Вінницький національний технічний університет
(повне найменування вищого навчального закладу)
Факультет інформаційних технологій та комп'ютерної інженерії
(назва факультету (відділення))
Кафедра програмного забезпечення
(повна назва кафедри (предметної, циклової комісії))

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Розробка методу і програмних засобів системи тренування і оцінювання робіт зі спортивного програмування. Частина 2 Клієнтський додаток»

Виконав: студент 2-го курсу, групи 2ПІ-20м (д/н)
спеціальності 121 – Інженерія програмного забезпечення
(шифр і назва напряму підготовки, спеціальності)
Кузнєцов Л. Г.
(прізвище та ініціали)
Керівник: к.т.н., доц. Бурбело С. М.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)
«___» _____ 2021 р.
Опонент: к.т.н., проф. Месюра В. І.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)
«___» _____ 2021 р.

Допущено до захисту
Завідувач кафедрою ПЗ
д.т.н., проф., Романюк О. Н.
«___» _____ 2021 р.

Вінниця ВНТУ – 2021 рік

Вінницький національний технічний університет
Факультет інформаційних технологій та комп'ютерної інженерії
Кафедра програмного забезпечення
Рівень вищої освіти ІІ-й (магістерський)
Галузь знань 12 Інформаційні технології
Спеціальність 121 Інженерія програмного забезпечення
Освітньо-професійна програма 121 Інженерія програмного забезпечення

УЗГОДЖУЮ
Директор КЗ «ВТЛ»
_____ О. М. Козяр
«____» _____ 2021 р.

ЗАТВЕРДЖУЮ
Завідувач кафедрою ПЗ
_____ О. Н. Романюк
« 13 » вересня 2021 р.

З А В Д А Н Н Я

НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТУ

Кузнєцову Леоніду Геннадійовичу

1. Тема роботи: Розробка методу і програмних засобів системи тренування і оцінювання робіт зі спортивного програмування. Частина 2 Клієнтський додаток.

керівник роботи: к.т.н., доц. кафедри ПЗ Бурбело С. М. затверджені наказом вищого навчального закладу від « 24 » вересня 2021 року № 277.

2. Строк подання студентом роботи до « 1 » грудня 2021 року.

3. Вихідні дані до роботи: середовище розробки – Microsoft Visual Studio Code, мови розробки – Ruby, JavaScript, C++, WebAssembly; операційна система – Linux; підтримка задач з одним варіантом відповіді та без інтерактивної взаємодії; взаємодія з серверною частиною через JSON-API за принципами REST; підтримка тестування на стороні клієнта.

4. Зміст розрахунково–пояснювальної записки (перелік питань, які потрібно розробити): вступ; аналіз стану питання оцінювання робіт зі спортивного програмування та постановка задач дослідження; удосконалення методу оцінювання розв’язків; удосконалення методу виконання тестування робіт; розробка методу і програмних засобів клієнтського додатку; тестування; висновки; список використаних джерел; додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень): мета, об'єкт та предмет дослідження; аналіз аналогів; діаграма компонентів веб-ресурсу «Codelabs»; діаграми послідовності варіантів використання системи; блок-схеми алгоритмів надсилання і оцінювання розв'язку, формування таблиці результатів, імпорту задачі з архіву; тестування.

6. Консультанти розділів магістерської кваліфікаційної роботи

Розділ	Прізвище, ініціали та посада Консультанта	Підпис, дата	
		завдання видав	завдання прийняв
1–4	к.т.н., доц. Бурбело С. М.		
5	д.е.н., проф. Бурєнікова Н. В.		

7. Дата видачі завдання « 14 » вересня 2021 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз проблеми, обґрунтування актуальності розробки системи та постановка задачі	15.09.2021 – 20.09.2021	Вик.
2	Розробка методів та засобів роботи системи	20.09.2021 – 02.10.2021	Вик.
3	Вибір середовища та мови розробки	02.10.2021 – 10.10.2021	Вик.
4	Розробка програмного продукту	10.10.2021 – 25.10.2021	Вик.
5	Тестування роботи системи	25.10.2021 – 05.11.2021	Вик.
6	Економічна частина	05.11.2021 – 20.11.2021	Вик.
7	Оформлення матеріалів до захисту МКР	20.11.2021 – 30.11.2021	Вик.

Студент _____ **Кузнєцов Л.Г.**
(підпис) **(прізвище та ініціали)**

Керівник магістерської кваліфікаційної роботи _____ **Бурбело С.М.**

АНОТАЦІЯ

УДК. 004.421+004.75:004.455

Кузнєцов. Л. Г. Розробка методу і програмних засобів системи тренування і оцінювання робіт зі спортивного програмування. Частина 2 Клієнтський додаток. Магістерська кваліфікаційно робота зі спеціальності 121 – Інженерія програмного забезпечення, освітня програма – 121 Інженерія програмного забезпечення. Вінниця: ВНТУ, 2021. 158 с.

На укр. мові. Бібліогр.: 39 назв; рис.: 60; табл. 9.

У магістерській кваліфікаційній роботі удосконалено метод виконання тестування робіт зі спортивного програмування шляхом винесення процесів компіляції вихідного коду та тестування отриманого виконуваного файлу у браузер на стороні клієнта, що дозволяє знизити навантаження на серверну систему та скоротити час очікування для користувача. Також проведено роботу над удосконаленням оцінювання розв’язків задач зі спортивного програмування, а саме запропоновано використати вагові коефіцієнти для налаштування обмежень у часі й пам’яті, що б враховували особливості мов програмування та компіляторів.

У результаті розроблено клієнтський додаток для системи тренування і оцінювання робіт зі спортивного програмування, що призначений для взаємодії з користувачем через веб-інтерфейс, надає можливість переглядати умови до задач, відправляти власні рішення й отримати результати їх перевірки, організовувати користувачів у групи та проводити змагання. Клієнтська частина зберігає усі дані про користувачів й задачі, використовуючи реляційну базу даних PostgreSQL та додаткове сховище даних Redis. Засоби програмного продукту реалізовані на мовах програмування Ruby, C++ та JavaScript під операційну систему Linux.

Ключові слова: спортивне програмування, веб-ресурс, тестування, компіляція, WebAssembly, Ruby on Rails.

ANNOTATION

The master's qualification thesis describes the improvement of the method of testing competitive programming solutions by moving processes of compiling source code and testing the resulting executable file in a browser on client side, which reduces load on server systems and reduces waiting time for users. In addition, the thesis describes the improvement of the method of rating competitive programming solutions by introducing weights to adjust time and memory constraints, that take into account the specifics of programming languages and compilers.

As a result, a client application was developed for the system of training and verification of competitive programming solutions, which is designed to interact with the user via a web interface, provides the ability to view task’s descriptions, send own solutions and get their verification results, organize users into groups for competitions. The client part stores all user data and tasks using PostgreSQL relational database and additional Redis data storage system. The software is implemented in Ruby, C ++ and JavaScript programming languages for the Linux operating system.

Keywords: competitive programming, web resource, testing, compilation, WebAssembly, Ruby on Rails.

ЗМІСТ

ВСТУП

Обґрунтування вибору теми дослідження. Наукова діяльність є невід'ємною складовою сучасного освітнього процесу та здійснюється з метою підвищення рівня наукової підготовки спеціалістів, виявлення талановитої молоді. На всіх етапах навчання на перше місце виходить завдання вибору ефективної оціночної системи, яка не тільки відображала б об'єктивний рівень знання студентів, а й мотивувала їх вчитися краще.

Одним з видів наукових заходів для студентів та учнів є олімпіада. Різноманітні інтелектуальні змагання проводяться з метою виявлення і розвитку творчих здібностей, підвищення інтересу до науково-дослідницької діяльності, сприяння професійній орієнтації, мотивування поглибленого вивчення [1]. У олімпіадах зі спортивного програмування потрібно розробити програму, що б розв'язувала поставлену задачу. Для її оцінювання необхідно запустити отриманий файл і перевірити результат його виконання на вичерпному наборі тестів. У наш час можливою є автоматизація цього процесу перевірки з використанням передових інформаційних та телекомунікаційних технологій.

Також, щоб отримати максимально ефективний результат від проведення олімпіади, їй має передувати етап підготовки, під час якого проводять заочні олімпіади, організовують семінари або гуртки, тощо [2]. Використання системи автоматизованої перевірки робіт на таких тренуваннях дозволяє наблизитись до олімпіадних умов, тобто дається можливість використати задачі й систему оцінювання, подібні до тих, які зустрінуться на змаганні.

Найкращим методом оцінювання знань студентів та учнів у процесі навчання програмуванню є перевірка створених ними програм у відповідних середовищах з різноманітними наборами тестів. Автоматична перевірка програм дозволяє забезпечити швидкість та об'єктивність оцінювання, додаючи до навчання елемент змагання (здобуття рейтингу), що спрямовано на підвищення мотивації і зацікавленості в вивчені предмету.

Таким чином, для підготовки та проведення олімпіад зі спортивного програмування, а також з метою покращення оцінювання знань студентів й учнів в процесі навчання програмуванню, доцільно використовувати системи автоматичного оцінювання робіт. Існуючі системи, такі як Codeforces, E-olymp, Ejudge і Algotester, мають певні недоліки та обмеження. Наприклад, не дозволяється виконати гнучке налаштування обмежень з урахуванням особливостей різних мов програмування, складність створення власних задач, неможливість переглянути деталі тестування, тощо. Все це обмежує сферу використання таких систем. Тому актуальною є розробка нової системи тренування і оцінювання робіт зі спортивного програмування.

Зв'язок роботи з науковими програмами, планами, темами. Робота виконувалася згідно плану виконання наукових досліджень на кафедрі програмного забезпечення.

Мета та завдання дослідження. Метою роботи є підвищення якості систем дистанційного проведення олімпіад і автоматичного оцінювання задач зі спортивного програмування за рахунок удосконалення методів оцінювання розв'язків задач спортивного програмування і виконання тестування робіт зі спортивного програмування, розробки програмних засобів клієнтського додатку нової системи, орієнтованої під специфіку обробки розв'язків задач зі спортивного програмування, що дозволяє підвищити об'єктивність оцінювання результатів і ефективність навчання чи тренування.

Основними задачами роботи є:

- удосконалення методу оцінювання розв'язків задач спортивного програмування;
- удосконалення методу виконання тестування робіт зі спортивного програмування;
- розробка діаграм взаємодії компонентів та блок-схем основних алгоритмів веб-ресурсу;
- розробка інтерфейсу програмного продукту;

- розробка підсистем для комунікації з серверною частиною, надсилання та оцінювання розв’язків, проведення змагань, створення задач й адміністрування, обробки сесій, особистого кабінету й комунікації з користувачами через електронну пошту;
- проведення тестування програмного продукту декількома різними методами.

Об’єкт дослідження – процес розробки системи для дистанційного проведення олімпіад зі спортивного програмування, технології зберігання задач, тестів і надісланих робіт, процес автоматичної перевірки розв’язків.

Предмет дослідження – методи та засоби розробки клієнтської частини системи тренування і оцінювання робіт зі спортивного програмування для розміщення задач, тренування й дистанційного проведення олімпіад.

Методи дослідження. У процесі дослідження використовувались:

- методи реляційних баз даних і нормальних форм для організації збереженої інформації на ресурсі;
- методи хешування для тимчасового збереження результатів складних обрахунків і пришвидшення доступу до ресурсу;
- методи теорії алгоритмів і принципи об’єктно-орієнтованого програмування для розробки алгоритмів і вихідного коду моделей, контролерів, представлень й сервісів;
- методи теорії кольорів для розробки дизайну користувацького інтерфейсу.

Наукова новизна отриманих результатів.

1. Подальшого розвитку отримав метод оцінювання розв’язків задач спортивного програмування, який, на відміну від існуючих, вводить до використання вагові коефіцієнти для налаштування обмежень у часі й пам’яті для задачі під окремі мови програмування та компілятори, що дозволяє виконати більш гнучке і точне налаштування тестуючої системи за обмеженнями автора задачі, оцінювати рішення на різних мовах програмування з урахуванням особливостей їх середовища виконання чи процесу запуску.

2. Подальшого розвитку отримав метод виконання тестування робіт зі спортивного програмування, який, на відміну від існуючих, пропонує перенести процеси компіляції вихідного коду рішення користувача, виконання отриманої програми та її тестування у браузер на стороні клієнта, що дозволяє знизити навантаження на серверну систему та скоротити час очікування для користувача.

Практична цінність отриманих результатів. Практична цінність одержаних результатів полягає в запропонованих алгоритмах та розробленій клієнтській частині системи тренування і оцінювання робіт зі спортивного програмування для розміщення задач, тренування й дистанційного проведення олімпіад із автоматичним оцінюванням розв’язків.

Впровадження. Результати досліджень використовуються у Комунальному закладі «Вінницький технічний ліцей» Вінницької міської ради, що підтверджується відповідним актом впровадження (додаток Б).

Особистий внесок здобувача. Усі наукові результати, викладені у магістерської кваліфікаційної роботі, отримані автором особисто. У наукових працях, опублікованих у співавторстві, автору належать алгоритми роботи і програмна реалізація клієнтської частини системи «CodeLabs», його архітектурна схема [3], аналіз статистичного дослідження результатів оцінювання часу роботи програми за різними методами [4], аналіз перспектив використання технології WebAssembly для перенесення процесів компіляції й запуску тестування рішень у браузер на стороні клієнта [5].

Апробація матеріалів магістерської кваліфікаційної роботи. Основні положення магістерської кваліфікаційної роботи доповідалися та обговорювалися на конференціях: XLIX Науково-технічна конференції факультету інформаційних технологій та комп'ютерної інженерії (Вінниця, 2020), IX Міжнародна науково-практична конференції молодих вчених та студентів «Молодь у світі сучасних технологій» (Херсон, 2020), Всеукраїнська науково-практична Інтернет-конференція «Електронні інформаційні ресурси: створення, використання, доступ» (Вінниця, 2021). Результати роботи

доповідалися на Всеукраїнському конкурсі студентських наукових робіт із спеціальності «Інженерія програмного забезпечення» й було отримано диплом переможця 2 ступеня (Тернопіль, 2021).

Публікації. Результати роботи опубліковані в трьох наукових працях – тезах-доповідях на XLIX Науково-технічній конференції факультету інформаційних технологій та комп'ютерної інженерії (Вінниця, 2020) [3], на IX Міжнародній науково-практичній конференції молодих вчених та студентів «Молодь у світі сучасних технологій» (Херсон, 2020) [4], на Всеукраїнській науково-практичній Інтернет-конференції «Електронні інформаційні ресурси: створення, використання, доступ» (Вінниця, 2021) [5].

Структура та обсяг роботи. Магістерська кваліфікаційна робота складається зі вступу; п’яти розділів; висновку; списку літератури, що містить 39 найменувань, 5 додатків. Робота містить 60 ілюстрацій, 9 таблиць.

1 АНАЛІЗ СТАНУ ПИТАННЯ ОЦІНЮВАННЯ РОБІТ ЗІ СПОРТИВНОГО ПРОГРАМУВАННЯ ТА ПОСТАНОВКА ЗАДАЧ ДОСЛІДЖЕННЯ

1.1 Аналіз стану питання оцінювання робіт зі спортивного програмування

На всіх етапах навчання на перше місце виходить проблема вибору ефективної оціночної системи, яка не тільки відображала б об’єктивний рівень знання студентів, а й мотивувала їх вчитися краще. Також, у сучасному навчальному процесі як студентів, так і учнів, важливу роль відіграє науково-дослідницька діяльність. З метою підвищення рівня наукової та практичної підготовки спеціалістів використовують різноманітні підходи, організовують олімпіади чи проводять змагання, хакатони.

Спортивне програмування – один із типів змагання за допомогою комп’ютеру, де кожен учасник може продемонструвати свої здібності у програмуванні та вирішенні алгоритмічних задач [6]. Змагання або олімпіади зі спортивного програмування влаштовують для підтримки змагального духу в процесі навчання студентів програмуванню, що сприяє виявленню та розвитку творчих здібностей, підвищує інтерес, мотивує на самостійне поглиблене вивчення складного матеріалу.

Завданнями системи тренування і оцінювання робіт зі спортивного програмування є допомога у навчанні студентів програмуванню і алгоритмам, організація та автоматизація процесу тренування до змагань зі спортивного програмування. У своїй основі, система має містити перелік доступних для розв’язування алгоритмічних задач та інструменти для перевірки їх рішень від користувачів, відображення детальних результатів такого тестування.

Під час проведення подібних змагань важливо швидко й точно виконувати оцінювання виконаних робіт. Особливо актуальним це питання постає в олімпіадах з великою кількістю учасників. Такі вимоги можна задовільнити, використовуючи передові досягнення в сфері інформаційних та телекомунікаційних технологій для автоматизації процесу проведення олімпіад

й перевірки рішень. Разом з цим забезпечується можливість дистанційного проведення турнірів, одночасну участь учасників з різних міст та навіть з різних країн. В ручному режимі майже неможливо забезпечити таку точність і швидкість оцінювання при одночасному проведенні олімпіад на кількох віддалених один від одного майданчиках, яку дозволяють досягти сучасні технології. Тому поширюється використання різних систем, що впроваджуються в навчальний процес і допомагають якісніше й швидше засвоїти навчальний матеріал, проводити різні змагання та інші заохочуючі заходи.

Наявні рішення для проведення змагань зі спортивного програмування дозволяють організовувати дистанційні змагання з автоматизованим оцінюванням. Деякі також надають можливість проводити тренування та підготовку до олімпіад, чи дозволяють тестувати додаткові рішення задач після закінчення змагання для перевірки нових ідей чи удосконалення своїх рішень.

Часто, подібні системи виконують у форматі веб-сайту із користувацьким інтерфейсом й серверної підсистеми для оцінювання рішень користувачів, що дозволяє їх використання як в навчальних установах, так і вдома для самостійної роботи. Вони працюють на будь-якому сучасному пристрої, що має можливість роботи з мережею Інтернет, та не потребують додаткового встановлення чи налаштування на кінцевому робочому місці користувача [7].

В основному, задача зі сфери спортивного програмування передбачає розробку програми, яка розв’язує певну проблему. Програма отримує певний набір вхідних даних в текстовому форматі через заздалегідь відомий файл, або читаючи стандартний потік введення консольного додатку. Свій результат програма повинна також записати у файл, або стандартний потік виведення, формат якого строго регламентовано умовою задачі.

Для перевірки рішень користувачів до кожної задачі створюють тести відповідно до її умови. Ці тести можуть містити вхідні дані для запуску розв’язку та очікуваний результат у вигляді файлу. Або можлива генерація вхідних даних на основі спеціально заданого алгоритму чи вихідних –

запускаючи авторське рішення. Якісні тести повинні повністю відповідати умові й технічним обмеженням задачі, покривати усі критичні точки можливих алгоритмів розв’язку. Разом з результатом роботи програм-розв’язків аналізується також затрачений в процесі їх роботи час та об’єм оперативної пам’яті. Обмеження на ці ресурси вказуються автором задачі, щоб виявляти рішення, які видають правильний результат, але не ефективно використовують системні ресурси через надлишкову складність алгоритму.

Для системи тренування і оцінювання робіт зі спортивного програмування необхідна підсистема, що виконуватиме компіляцію вихідних кодів рішень користувачів, запуск отриманих виконуваних файлів відповідно до тестів із задачі та аналіз результатів. Класичним підходом до розробки такої підсистеми є проведення компіляції вихідного коду, запуску програми та аналіз її роботи на стороні серверу. Проте, в такому випадку постають наступні питання ефективності й безпеки.

По-перше, необхідно виділити додаткові серверні ресурсах для швидкої обробки відправлень від усіх користувачів. Окрім ресурсів на роботу власне веб-сайту, потрібні додаткові ресурси серверу на підсистему оцінювання При цьому, оскільки доступні ресурси зазвичай обмежені, роботи поміщаються в певну чергу й перевіряються поступово, отже користувач може отримати результати перевірки своїх рішень із затримкою.

По-друге, існує вірогідність отримати зловмисний код від користувача. Тому необхідно обмежити можливості виконуваного коду, впроваджувати додаткові заходи безпеки для збереження працездатності системи й блокування нечесних рішень. Зазвичай це досягається певною ізоляцією чи віртуалізацією та потребує додаткової розробки й системних ресурсів.

По-третє, оскільки перевірка робіт користувачів відбувається централізовано, при перевантаженні ресурсів такої системи оцінювання чи її непрацездатності з певних причин, користувачі втрачають можливість отримати будь-яку інформації щодо їх розв’язків. Тривала відсутність

зворотного зв’язку критична під час змагання і може блокувати роботу команди.

1.2 Аналіз методів розробки клієнтської частини системи

Клієнтська частина веб-ресурсу для розміщення і автоматизованої перевірки олімпіадних задач матиме вигляд сайту. Розробка сайту може бути здійснена кількома різними методами.

Найбільш поширеним методом розробки є проектування й програмування одного програмного продукту, що включатиме усі необхідні функції. Такий спосіб розробки також називають монолітною архітектурою. Як видно з рисунку 1.1, монолітна архітектура передбачає об’єднання усіх підсистем в єдину програму, що одночасно реалізує і бізнес-логіку, і роботу з даними на нижчому рівні, і увесь інтерфейс користувача [8].

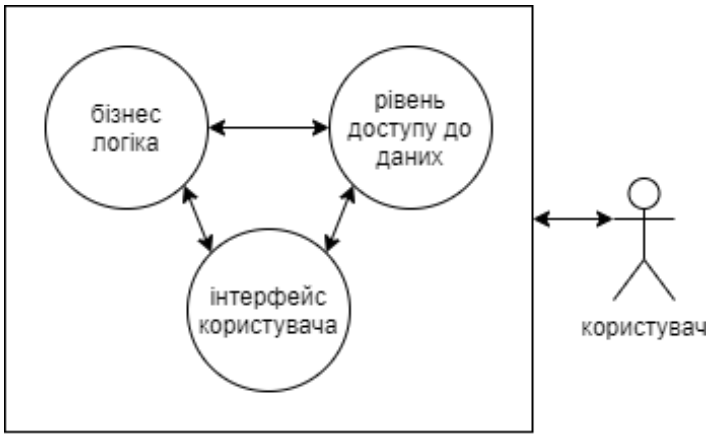


Рисунок 1.1 – Схема монолітної архітектури [8]

Перевагою такого підходу є простота початкової розробки, швидкість отримання мінімально життєздатного продукту. Однак подальші модифікації можуть стати занадто складними через надлишкову зв’язність між підсистемами.

Іншим способом є розробка сайту використовуючи так звану мікросервісну архітектуру (рисунок 1.2). У такому випадку ресурс поділяється на невеликі підсистеми, що програмуються окремо одна від одної у вигляді

незалежних програм. Кожна підсистема виконує якусь одну функцію, наприклад реєстрація користувачів чи оцінка задач. Цей метод розробки складніший та потребує більше часу на початку, однак дозволяє вносити подальші модифікації ефективно, оскільки всі складові є невеликими і простими для розуміння. Внутрішня схема кожного мікросервісу подібна до монолітної – він так само містить реалізацію бізнес-логіки і роботу з даними. Однак логіка та дані всередині одного мікросервісу стосуються лише його функції.

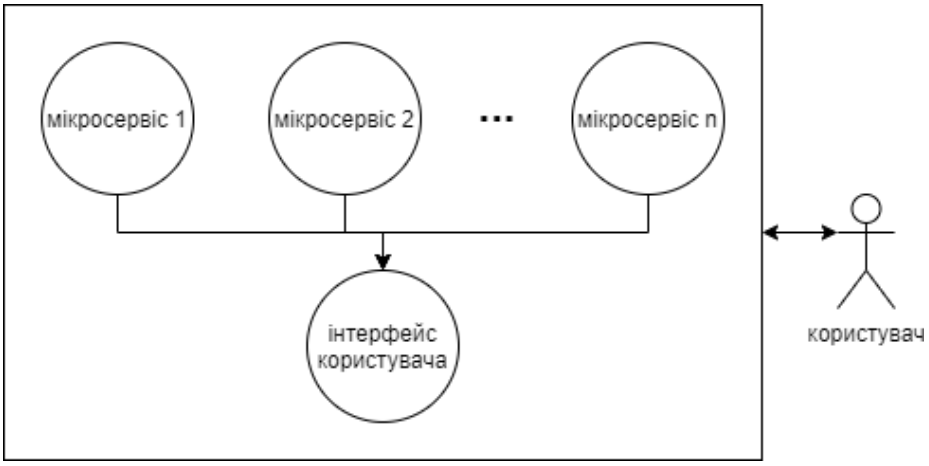


Рисунок 1.2 – Схема мікросервісної архітектури [8]

Хоча мікросервісна архітектура і має свої переваги, вона складніша в реалізації, потребуватиме більше часу та ресурсів на початковому етапі розробки. Тому для цієї роботи краще обрати монолітну архітектуру, що можна вважати класичним підходом для розробки подібних веб-ресурсів.

Веб-додатки за монолітною архітектурою реалізовуються в вигляді прикладної програми, розробленої за принципом «клієнт-сервер», що використовує в якості клієнта браузер і працює на стороні сервера [9]. На одному комп’ютері відбувається централізоване збереження й обробка всіх даних, необхідних для функціонування ресурсу: опис задач та їх тестів, персональна інформація про користувачів, відомості про надіслані розв’язки, оцінки й групи. Усі користувачі мають можливість працювати з системою

одночасно, відсилати розв’язки на перевірку та бачити доступну інформацію в режимі реального часу.

Інтерфейс веб-додатку виконується у вигляді сторінок в браузері з використанням технологій JavaScript, HTML і CSS. Для серверної частини застосування різних мов програмування практично не обмежено, однак певні краще підходять для цих цілей, оскільки вже містять бібліотеки для роботи з мережею Інтернет та необхідними протоколами.

Програма серверної частини повинна оперативно отримувати інформацію про надіслані до системи розв’язки щоб виконати їх перевірку. Також має бути можливість надсилання нею результатів тестування програм для подальшого запису у систему. Для такої взаємодії зазвичай використовують мережу «Інтернет», що дозволяє з’єднуватись кільком комп’ютерам для передачі даних. Потрібно розробити прикладний програмний інтерфейс для обміну даними між клієнтською та серверною частинами. Також необхідно визначити схему передачі даних та спосіб організації доступних ресурсів.

1.3 Порівняльний аналіз аналогів

Серед систем для розміщення і автоматизованої перевірки олімпіадних задач зі спортивного програмування найбільш примітні наступні:

Codeforces – тренувальна й навчальна веб-платформа, створена групою програмістів з Саратовського університету [10]. На платформі Codeforces регулярно проводяться змагання, а минулі можуть бути використані для підготовки. Система Codeforces не має відкритий вихідний код, дозволяє проводити турніри чи тренуватись розв’язуючи минулі задачі. Також дозволяється переглядати деталі оцінки по кожному з тестів. Інтерфейс зображено на рисунку 1.3.

E-Olymp – веб-портал організаційно-методичного забезпечення дистанційних олімпіад, що розроблено Житомирським державним університетом імені Івана Франка [11]. Сайт надає можливість тренуватися

розв’язувати задачі з програмування, брати участь в змаганнях, а також містить деякі навчальні матеріали. Інтерфейс зображено на рисунку 1.4.

Ejudge – система для проведення різних заходів, в яких необхідна автоматична перевірка програм [12]. Підтримується проведення турнірів по чотирьох системах – ACM, Kirow, Moscow, Olympiad. На відміну від інших, це рішення потрібно встановлювати на власному сервері, що робить його складнішим у використанні, однак, після налаштування, дозволяє проводити власні змагання по своїм задачам. Ця система часто використовується для проведення деяких етапів олімпіади ICPC та інших. Підтримується багато можливостей, серед яких можна виділити відображення детального результату до кожного тесту, робота з кількома одночасно-працюючими серверами перевірки рішень. Також доступний відкритий вихідний код програми. Інтерфейс зображено на рисунку 1.5.

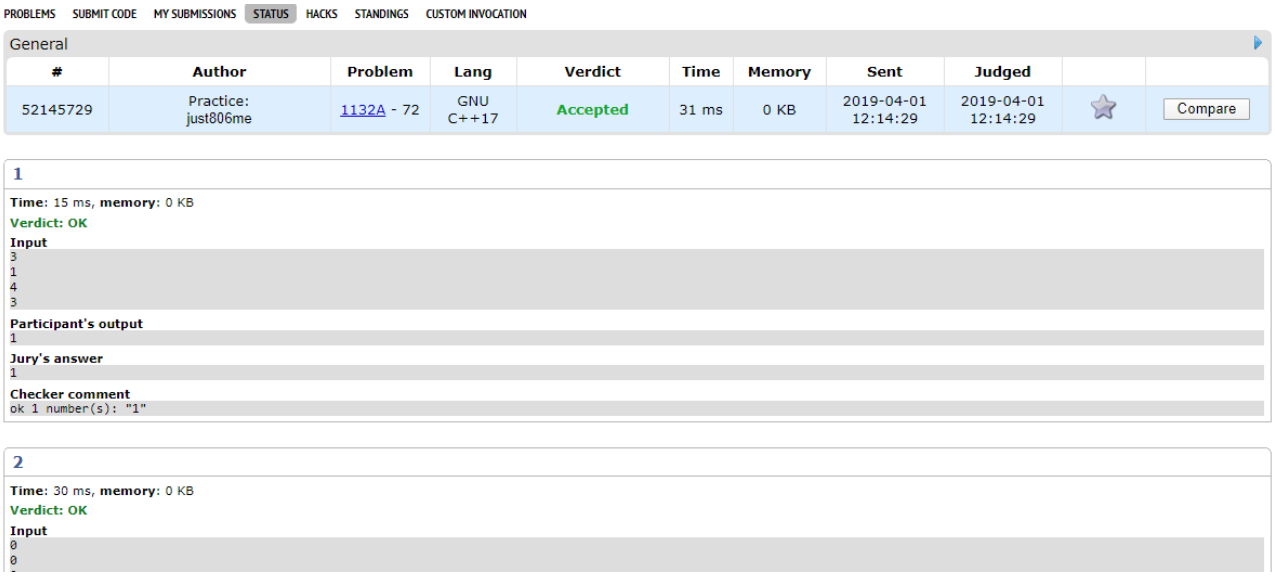


Рисунок 1.3 – Інтерфейс платформи Codeforces [10]

Algotester – система автоматичного тестування розв’язків з великим набором цікавих алгоритмічних задач різноманітної складності, ресурс для створення та публікації власних задач з можливістю проведення змагань [13]. Ресурс розроблено на базі Львівського національного університету імені Івана Франка. В ЛНУ та інших університетах Львова ця платформа регулярно

використовується для проведення змагань й тренувань, участь в яких доступна одночасно і для інших користувачів онлайн. Платформа також дозволяє користувачам реєструватись для відстеження власного прогресу. Інтерфейс зображено на рисунку 1.6.

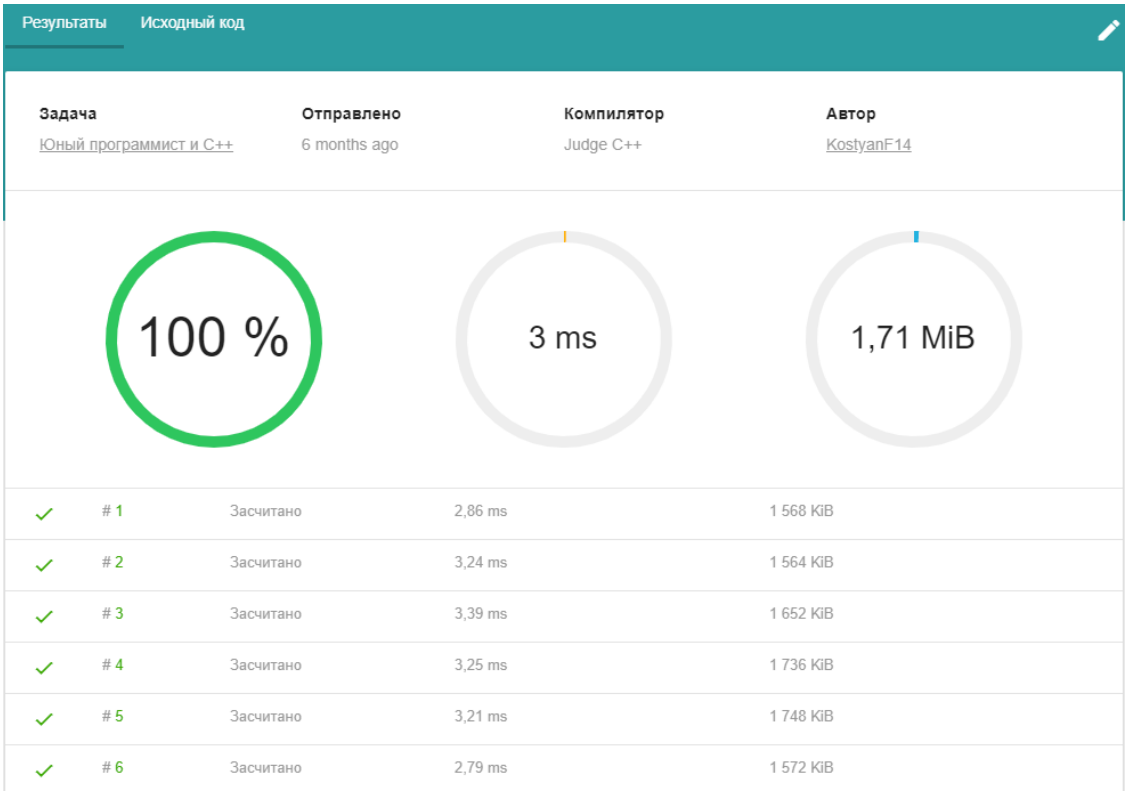


Рисунок 1.4 – Інтерфейс платформи E-Olymp [11]

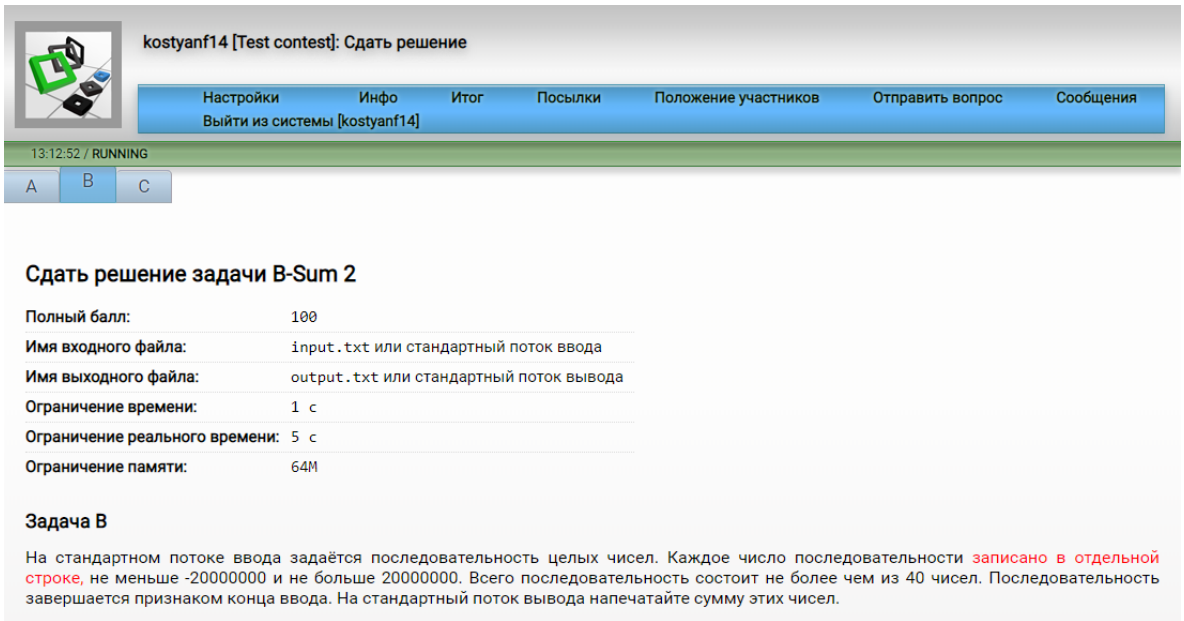


Рисунок 1.5 – Інтерфейс платформи Ejudge [12]

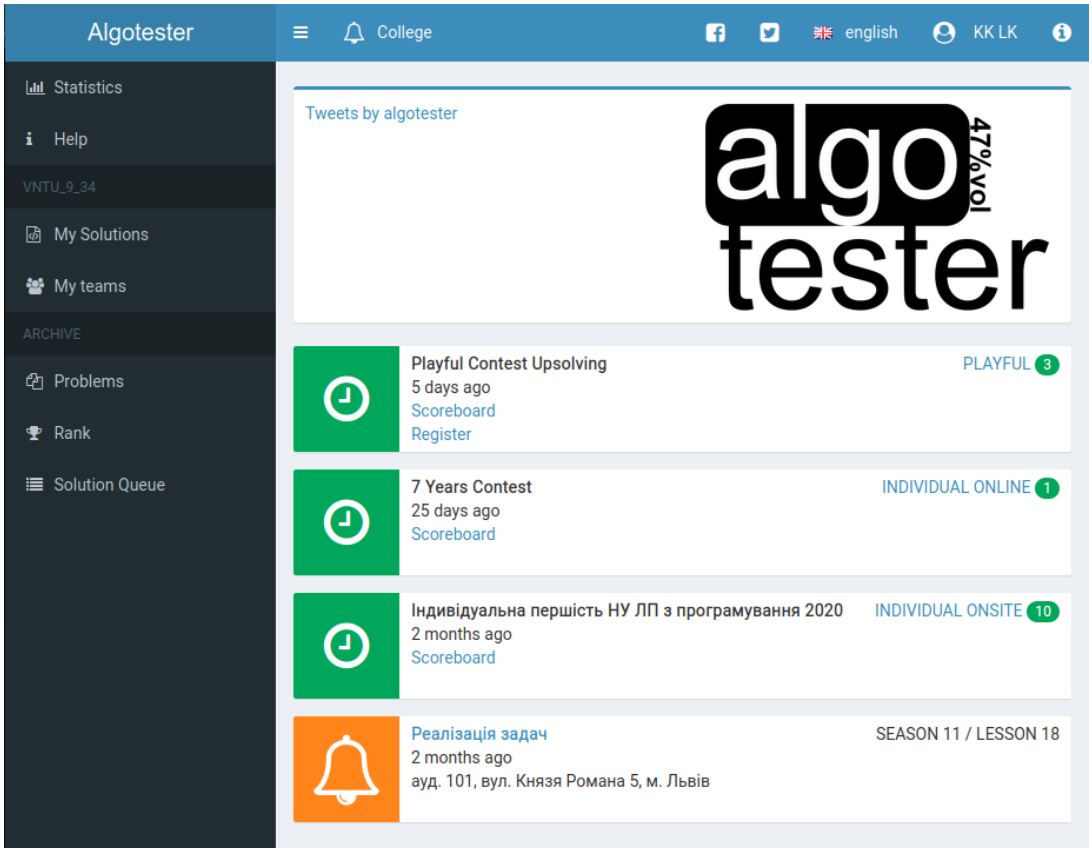


Рисунок 1.6 – Інтерфейс платформи Algotester [13]

Проаналізувавши доступні аналоги, було визначено їхні можливості та недоліки, які враховувались при створенні власного програмного забезпечення з назвою «CodeLabs». Побудована таблиця порівняльних характеристик показала, що розробка програмного продукту є доцільною (табл. 1.1).

У результаті отримаємо продукт, що забезпечує зручне розміщення і автоматизовану перевірку олімпіадних задач, дозволяє проводити контести й тренування під час навчання, має зручний графічний інтерфейс. Веб-ресурс «CodeLabs» покриватиме недоліки існуючих рішень: має відкритий вихідний код, що дозволяє запускати його на власній системі; підтримує роботу з кількома серверами тестування одночасно; відображає деталі тестування, що дає необхідну під час навчання чи тренування інформацію; дозволяє створювати власні задачі; вводить до використання вагові коефіцієнти для налаштування обмежень в часі й пам’яті під окремі мови програмування, що дозволяє розв’язувати задачі без огляду на накладні витрати обраної мови програмування.

Таблиця 1.1 – Порівняльні характеристики програмних продуктів

Критерій	Codeforces	E-olymp	Ejudge	Algotester	CodeLabs
Відкритий вихідний код	-	-	+	-	+
Особистий кабінет користувача	+	+	-	+	+
Кілька одночасно-працюючих серверів тестування	-	-	+	-	+
Перегляд деталей по кожному тесту	+	-	+	-	+
Коефіцієнти обмежень для різних мов	-	-	-	-	+
Створення власних задач	+/-	-	+	+	+
Тестування на стороні клієнту	-	-	-	-	+

1.4 Постановка задач дослідження

Після аналізу поточного стану питання розміщення олімпіадних задач з програмування й автоматизованого тестування їх розв’язків, порівняння існуючих рішень цієї проблеми за визначеними критеріями було визначено завдання, які необхідно виконати для дослідження і розробки клієнтського додатку сервісу «Codelabs»:

- удосконалення методу оцінювання розв’язків задач спортивного програмування;
- удосконалення методу виконання тестування робіт зі спортивного програмування;
- розробка діаграм взаємодії компонентів та блок-схем основних алгоритмів веб-ресурсу;
- розробка інтерфейсу програмного продукту, який буде зрозумілим для користувачів;
- розробка підсистем для комунікації з серверною частиною, надсилання та оцінювання розв’язків, проведення змагань, створення задач й

адміністрування, обробки сесій, особистого кабінету й комунікації з користувачами через електронну пошту;

- проведення тестування програмного продукту декількома різними методами.

1.5 Висновки

У першому розділі було розглянуто актуальний стан питання автоматизованої перевірки задач під час турнірів чи олімпіад на сьогоднішній день та доступні можливості використання автоматизованих систем в навчанні з метою покращення навичок, здобутих студентами під час навчального процесу. Було виявлено проблеми ефективності й безпеки при тестуванні вихідного коду користувачів на стороні серверу.

Також було проаналізовано можливі методи вирішення цього питання, технічні особливості реалізації програми, її інтерфейсу та взаємодії між частинами. Обрано розробку програми з використанням сучасних веб-технологій із монолітною архітектурою. Взаємодію з серверною частиною вирішено виконати через мережевий програмний інтерфейс.

Разом з цим було розглянуто аналоги системи тренування і оцінювання робіт зі спортивного програмування та порівняно їх з розроблюваною програмою за переліком критеріїв. Основними аналогами є системи Codeforces, E-olymp, Ejudge і Algotester. Серед їх недоліків можна виділити відсутність відкритого вихідного коду, підтримку лише одного серверу тестування, недостатня детальність оцінки, складність створення власних задач, використання однакових обмежень для всіх мов програмування, а також відсутня підтримка тестування на стороні клієнту.

У результаті порівняння було відображено доцільність розробки програмного продукту магістерської кваліфікаційної роботи. З огляду на недоліки аналогів для подальшої розробки було визначено основні завдання, які необхідно розв’язати. Такі завдання враховують недоліки розглянутих аналогів і включають розробку діаграм взаємодії між компонентами, розробку

підсистеми комунікації клієнтської та серверної частин, проектування зрозумілого інтерфейсу, розробку модулів для роботи з задачами, розв’язками, змаганнями, сесіями та особистими даними користувача, удосконалення методу виконання тестування робіт зі спортивного програмування шляхом перенесення певних етапів на клієнтську сторону, а також проведення кінцевого тестування отриманої програми.

2 РОЗРОБКА МЕТОДІВ І ЗАСОБІВ КЛІЄНТСЬКОГО ДОДАТКУ СИСТЕМИ ТРЕНУВАННЯ І ОЦІНЮВАННЯ РОБІТ

2.1 Аналіз інформаційного забезпечення системи

Веб-додатки за монолітною архітектурою розробляються у два етапи: створення застосунку для серверного комп’ютера та розробка інтерфейсу для його користувачів. Також, для організації взаємодії з серверною частиною потрібно розробити спеціальний мережевий інтерфейс.

Для розробки серверного застосунку використовують багато різноманітних технологій і мов програмування. Така програма повинна приймати HTTP-запити від клієнтів, видавати їм HTTP-відповіді, зазвичай разом зі сторінкою, що містить дані для відображення. Організацією і зберіганням даних на сервері займається система керування базою даних (СКБД), яка надає можливості створення, збереження, оновлення та пошуку інформації з контролем доступу до неї.

Для веб-ресурсів зазвичай використовують реляційну СКБД, що дозволяє описувати всі сутності на сайті і залежності між ними в єдиному місці. Також є можливість використання нереляційних СКБД, які не потребують чіткого опису схеми об’єктів, проте не дозволяють вказувати залежності, що додає накладних витрат при обробці певних типів даних.

Основними характеристиками задачі зі спортивного програмування можна вважати її текстовий опис, технічні умови і обмеження по часу та пам’яті, класифікацію для зручного пошуку, список розроблених автором тестів, вихідний код програми перевірки. Оскільки задача може містити кілька описів на різних мовах, ці дані зручніше зберігати окремою сутністю перекладу, вказавши в ній відповідне посилання на задачу.

Список тестів до кожної задачі краще зберігати окремою сутністю. Така сутність повинна містити вхідні й вихідні дані для перевірки, а також кількість балів за успішне виконання.

З метою зберігання й обробки надісланих користувачами розв’язків потрібно записувати час надсилання, обрану мову програмування, вихідний код, а також посилання на обрану задачу. Також необхідною є інформація про проведення тестування, така як статус тестування, кількість набраних балів та детальний опис результатів кожного тесту. Опис результатів доцільно винести в окрему сутність.

У підсистемі особистого кабінету й комунікації з користувачами також потрібна інформація про користувача, така як його електронну адресу, пароль, ім’я та зображення профілю.

Також, для забезпечення гнучкого налаштування тестуючої системи необхідно зберігати коефіцієнти обмежень для різних мов та інші налаштування компілятора. Тому потрібна сутність, яка буде зберігати та надсилати до серверної частини ці дані.

Збереження простих даних, таких як текст, числа чи дати, забезпечуються звичайними полями. Проте файли краще зберігати з допомогою іншого сховища, а в атрибутах буде містити лише посилання чи шлях до файлу. Це дозволить не втратити швидкодію при роботі з великим об’ємом даних.

Користувач взаємодіє з веб-застосунком через інтерфейс у браузері. Класичним методом створення веб-інтерфейсів є використання HTML для визначення вмісту веб-сторінок, CSS для визначення презентації веб-сторінок та JavaScript для визначення поведінки веб-сторінок [14]. Браузер будує сторінку на основі цих інструкцій, що повністю описують її вигляд та поведінку. Основними вимогами до веб-інтерфейсів є їх однаковий зовнішній вигляд і однакова функціональність при роботі в різних браузерах, зовнішня привабливість, адаптивність до різних розмірів екрану.

Hypertext Markup Language використовується для розбиття інформації, що відображається на сторінці, в послідовність елементів, таких як [15]:

- заголовки, абзаци, списки, таблиці;
- зображення, аудіо, відео чи інші типи медіа;
- гіпертекстові посилання на інші документи;

- інтерактивні форми, кнопки, поля для введення інформації.

Cascading Style Sheets являє собою правила, що застосовуються до елементів HTML і керують їх візуальним представленням – колір, розмір, вирівнювання, шрифт, відступи, рамки, видимість та інші [15]. Правила можуть застосовуватись як до одного елемента за його ідентифікатором, так і до групи елементів за їх класом або назвою тегу.

JavaScript – мова програмування, яка виконується у браузері й дозволяє керувати властивостями відображуваного документа, інтерактивно змінювати їх, реагувати на дії користувача, виконувати запити до серверу, тощо. Довгий час керувати поведінкою сторінки у браузері можна було лише з використанням JavaScript. Проте, сучасні браузери також підтримують технологію WASM. З її допомогою можна виконувати у браузері код, написаний на будь-яких інших мовах програмування, таких як C++, C# чи Rust. Ця технологія досить нова та має певні обмеження, такі як неможливість прямої взаємодії з DOM [16]. Тому її використання доцільно коли потрібно виконувати складні обрахунки на стороні користувача, а для програмування простої взаємодії зі сторінкою продовжується переважне використання JavaScript.

2.2 Розробка схеми взаємодії з серверною частиною

Програма серверної частини виконує компіляцію та тестування розв’язків, надісланих користувачами. З міркувань безпеки серверна частина знаходиться на окремому комп’ютері. Проте, вона повинна оперативно отримувати всю інформацію про отримані клієнтською частиною розв’язки, щоб виконати їх перевірку. Також має бути зворотній зв’язок щоб отримати результати тестування. Для взаємодії програм, що розташовані на різних системах, використовують локальну мережу чи мережу «Інтернет». Для роботи в мережі необхідно визначити схему передачі даних та спосіб організації доступних ресурсів.

Протокол взаємодії з серверною частиною повинен відповідати певним вимогам до інформаційного забезпечення. Основні вимоги, що повинні бути

дотримані при створенні програмного застосунку: цілісність, достовірність, контроль, захист від несанкціонованого доступу, єдність і гнучкість, стандартизація та уніфікація, адаптивність, мінімальна кількість помилок введення-виведення інформації [17]. Передачу даних необхідно здійснювати у машино-читаємому форматі JSON, XML чи Protocol Buffers. Для забезпечення безпеки під час передачі даних потрібно використовувати сучасні методи шифрування на рівні транспортного потоку, а також авторизацію.

Шифрування протоколу передачі даних у веб-додатках класичним чином реалізується з допомогою Secure Sockets Layer [18]. Це забезпечує шифрування більшості складових пакету даних та захист від аналізу трафіку та атак типу «MITM». Для авторизації можна використати кілька розповсюджених підходів: базова авторизація HTTP та авторизація через передачу токену.

Базова авторизація HTTP – це найпростіша схема, за якою ім'я користувача та пароль користувача передаються в заголовку Authorization в незашифрованому вигляді, але з кодуванням base64 [19]. Недоліком є потреба в як імені користувача, так і паролю. Такий спосіб зручний для авторизації користувачів, однак додає накладні витрати при використанні для програмної взаємодії. Схему базової авторизації HTTP зображено на рисунку 2.1 [19].

Рисунок 2.1 – Схема базової авторизації HTTP [19]

Авторизація через передачу токену подібна до базової, проте більш підходить для реалізації програмної взаємодії. Сервер генерує унікальний токен, який захищеним способом передається до серверної частини. Після цього цей токен додається до кожного запиту у вигляді відповідного параметру або заголовку. Часто цей спосіб також називають «Bearer Tokens».

Під час розробки схеми взаємодії з серверною частиною обрано використати форматі JSON для передачі та авторизацію по токену. Для організації доступних ресурсів зручно використати підхід REST. Його вимогами є розділення доступного функціоналу на ресурси, що мають певний шлях та ідентифікатори. Для роботи з такими ресурсами пропонується користуватись стандартними методами протоколу HTTP – GET для отримання інформації, POST для додання нової інформації, PATCH для оновлення існуючої інформації та DELETE для видалення створених ресурсів.

Однією з вимог до клієнтської частини веб-ресурсу є підтримка кількох одночасно-працюючих серверів тестування для забезпечення швидкодії системи при великому навантаженні. Для забезпечення цього функціоналу розроблено спеціальний ресурс і відповідну сутність в СКБД. Ресурс «Worker» реалізовуватиме такі функції: реєстрація нового серверу, оновлення відомостей про сервер для відслідковування його статусу, видалення серверу для відслідковування його зупинки. У відповідь на запит реєстрації сервер отримуватиме свій ідентифікатор, що використовуватиметься в подальшому для авторизації його дій.

Щоб забезпечити роботу серверної частини, окрім сервісного ресурсу, також розроблені ресурси керування потрібними для цього сутностями: розв’язки користувача, задачі та тести, налаштування і коефіцієнти компіляторів, результати перевірки. Серверна частина має можливість отримати всі відомості про задачу й розв’язок, що перевіряються, а також завантажити інформацію про тестування, оновити статус розв’язку і його оцінку.

2.3 Розробка інтерфейсу користувача

Інтерфейс програмного продукту складається з веб-сторінок, кожна з яких відповідає основним його функціям: реєстрація користувача, вхід, особистий кабінет, список задач, деталі задачі й надсилання рішення, список надісланих робіт, деталі оцінки, список груп, основна інформація про групу, таблиця результатів змагання групи.

Сторінка реєстрації користувача (рисунок 2.2) дозволяє ввести коротку особисту інформацію, обрати логін та пароль, зареєструватись в системі. Також сторінка повинна містити захист від спаму та автоматизованих реєстрацій ботами, що досягається інтеграцією з сервісом ReCaptcha. При розробці цієї сторінки використано такі основні елементи керування: поля редагування для введення імені, логіну та паролю, кнопка надсилання введених даних, мітки з основною інформацією про сайт.

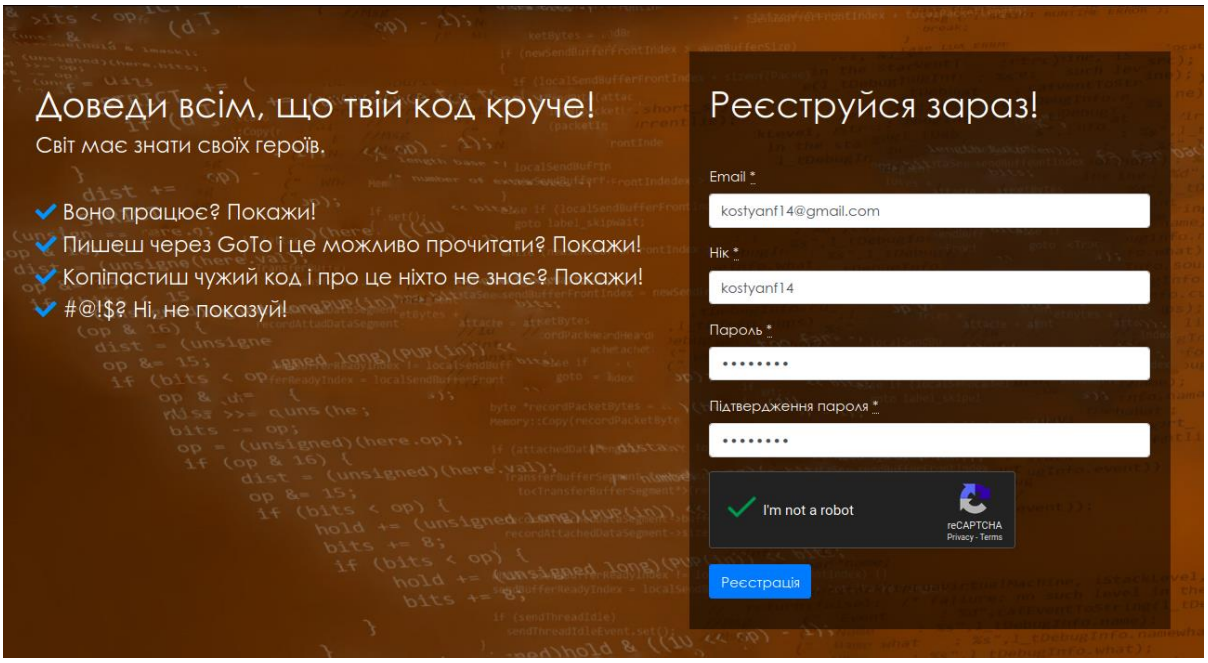


Рисунок 2.2 – Сторінка реєстрації користувача

Сторінка входу користувача (рисунок 2.3) потрібна, щоб користувачі могли авторизуватись у системі після реєстрації, або на інших комп’ютерах. Для захисту від автоматизованого підбору паролів також потрібно використати ReCaptcha. При розробці цієї сторінки використано такі основні елементи

керування: поля редагування для введення логіну з паролем, кнопка входу та відновлення паролю.

Сторінка особистого кабінету (рисунок 2.4) – це головна сторінка веб-ресурсу після входу користувача. Вона дозволяє керувати своїми налаштуваннями, вказати відображуване ім’я та аватар, переглядати власні розв’язки та виконувати інші дії стосовно профілю користувача. Для її розробки використано: поля редагування, кнопка оновлення інформації, кнопки з посиланнями на список розв’язків та виконання інших дій.

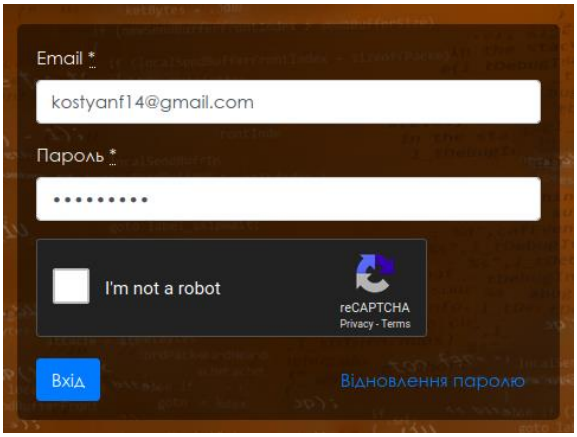


Рисунок 2.3 – Сторінка входу користувача

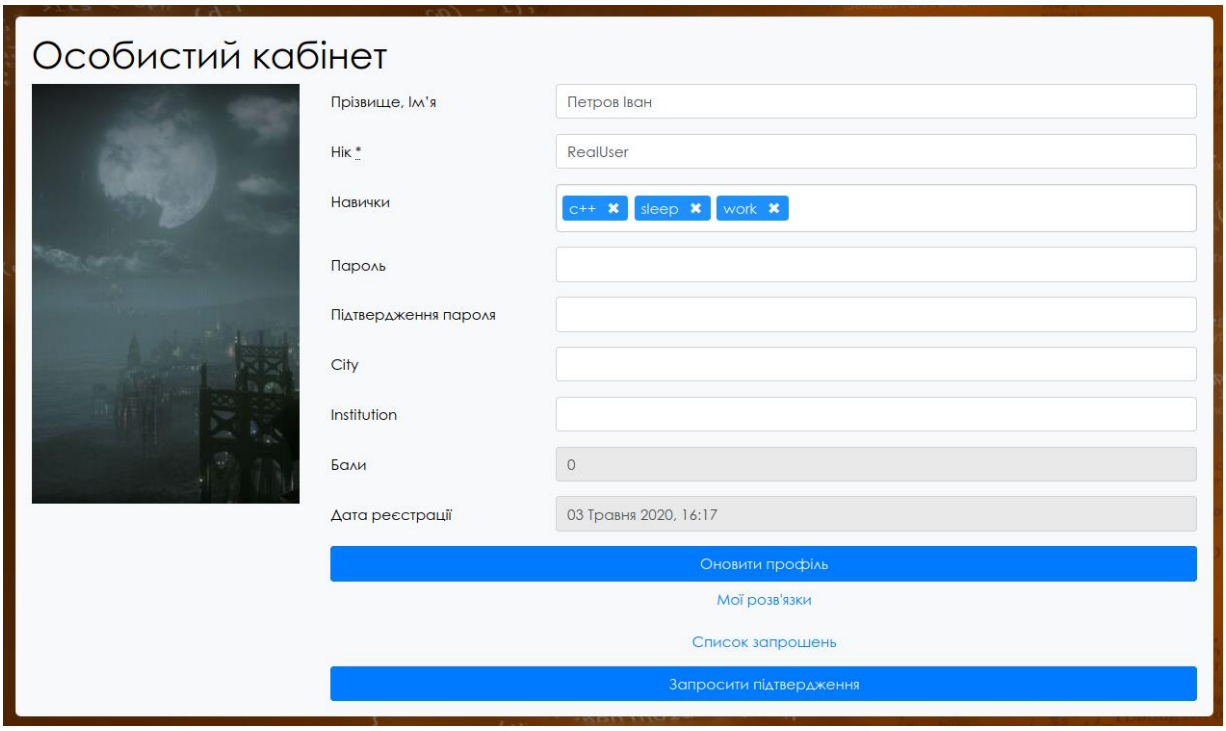


Рисунок 2.4 – Сторінка особистого кабінету

Сторінка списку задач (рисунок 2.5) використовується для відображення доступних для розв’язування задач і містить посилання на них. З метою спрощення навігації на цій сторінці використовується пагінація, так як кількість задач може бути великою. Для її розробки використано елемент для відображення табличних даних та кнопки переходу між сторінками.

Сторінка з деталями задачі (рисунок 2.6) дає можливість переглянути умову обраної задачі, її технічні обмеження, а також надіслати своє рішення для перевірки. При розробці цієї сторінки використано мітки з інформацією про задачу, а також спадне меню вибору підтримуваних мов програмування, кнопка вибору файлу й кнопка надсилання розв’язку.

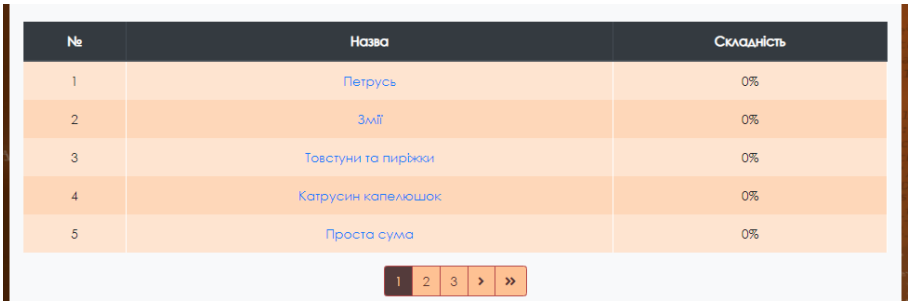


Рисунок 2.5 – Сторінка списку задач

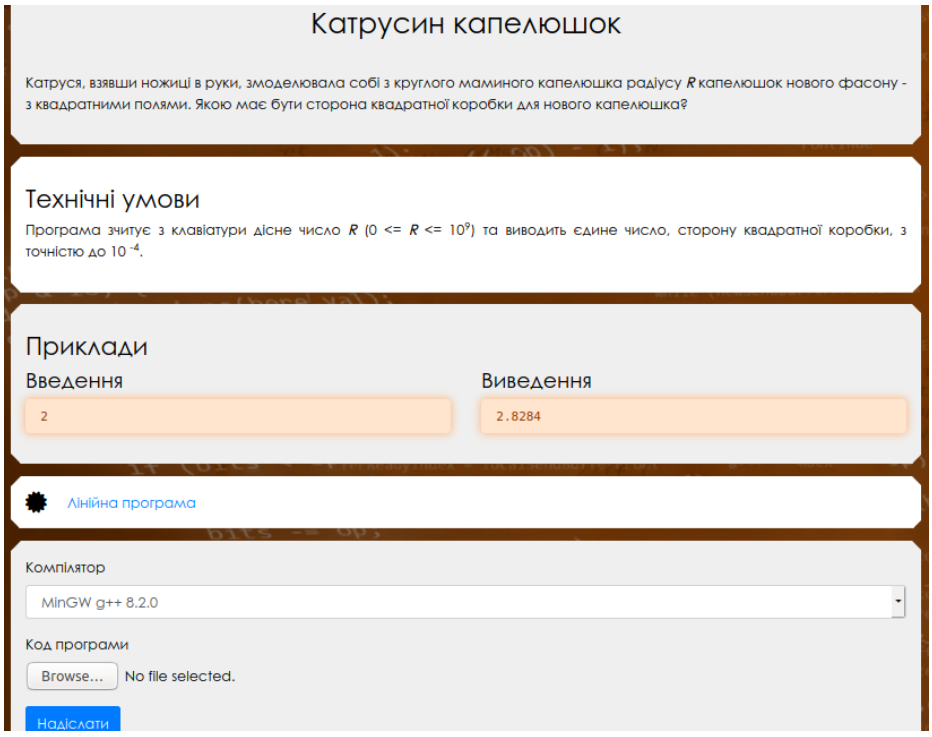


Рисунок 2.6 – Сторінка з деталями задачі

Сторінка зі списком надісланих робіт (рисунок 2.7) дозволяє побачити список робіт, що надіслані на перевірку кожним користувачем сайту, та містить посилання на деталі оцінки. З метою спрощення навігації на цій сторінці використовується пагінація, так як кількість надісланих робіт може бути великою. При розробці цієї сторінки використано елемент для відображення табличних даних та кнопки переходу між сторінками.

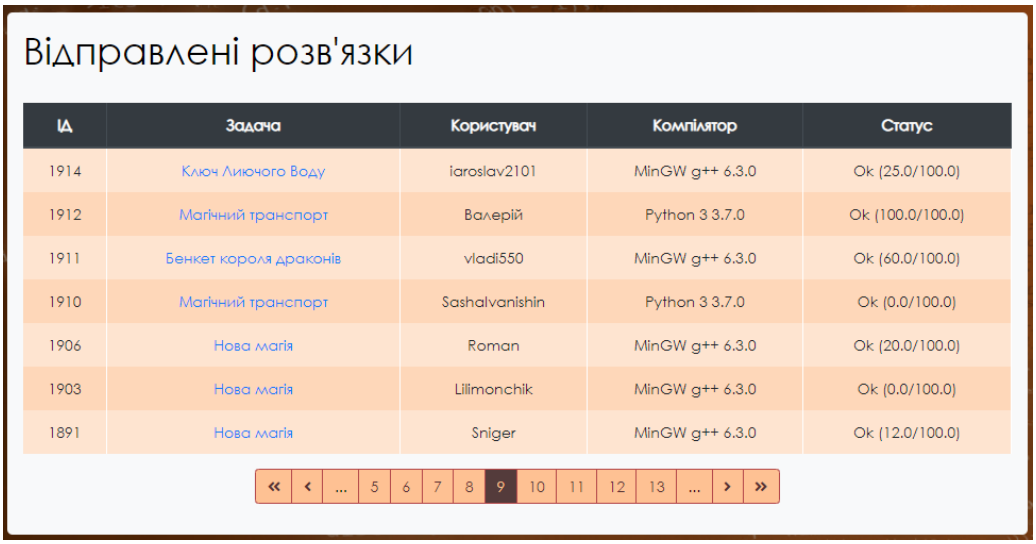


Рисунок 2.7 – Сторінка зі списком надісланих робіт

Сторінка деталей оцінки (рисунок 2.8) використовується для перегляду всієї інформації стосовно виконаного тестування розв’язків. Для її розробки використано такі елементи керування, як елемент для відображення табличних даних, мітки з додатковою інформацією та кнопки переходу між вкладками.

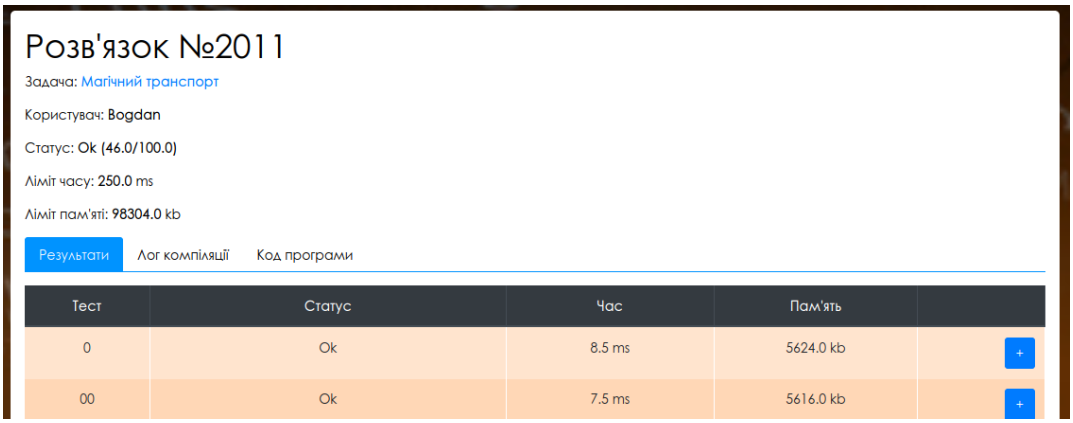


Рисунок 2.8 – Сторінка деталей оцінки

Сторінка списку груп (рисунок 2.9) відображує список існуючих груп, використовуючи елемент для відображення табличних даних. Для кращої навігації також використовується пагінація.

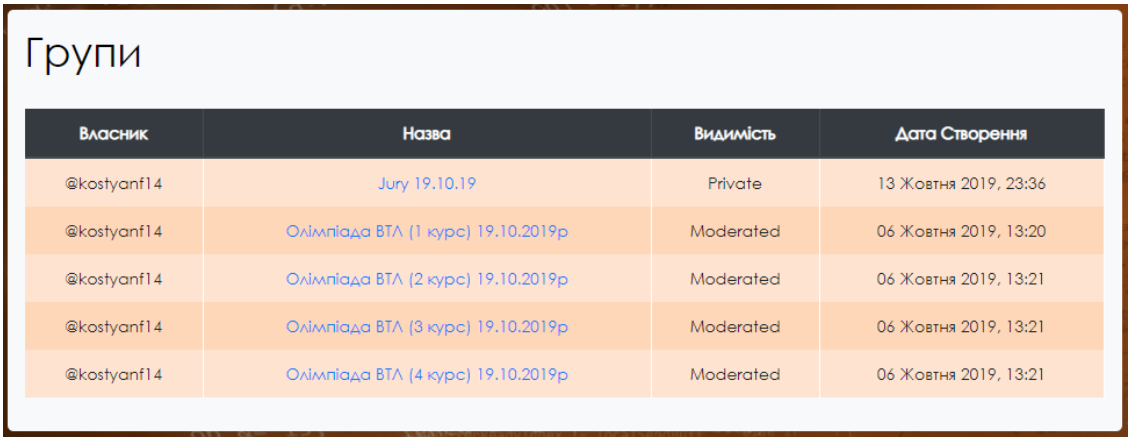


Рисунок 2.9 – Сторінка списку груп

Сторінка основної інформації про групу (рисунок 2.10) дозволяє вступити в групу, переглянути її опис та список учасників, виконати інші операції з нею, а також містить кнопку з посиланням на таблицю результатів для цієї групи. Під час розробки цієї сторінки використано: мітки з інформацією про групу, кнопки для вступу в групу та виконання інших операцій, список з учасниками.

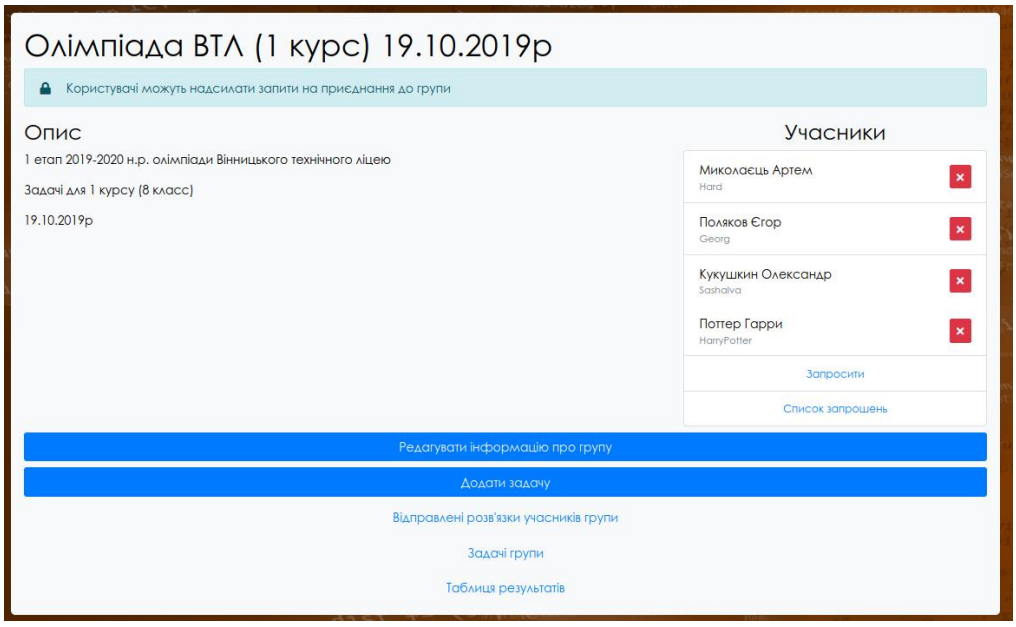


Рисунок 2.10 – Сторінка основної інформації про групу

Сторінка з таблицею результатів змагання групи (рисунок 2.11) використовується для перегляду сумарних результатів під час змагання. В ній використано лише елемент керування таблиця.

User	A	B	C	D
Поттер Гарри	84.0	18.0	0.0	
Холмс Шерлок	84.0	44.0	48.0	
Грейнджер Герміона	84.0	48.0	4.0	20.0
Абдулаєв Богдан	76.0	0.0	0.0	
Веретенко Денис	84.0		0.0	
Король Артур	84.0	100.0	0.0	
Добрій Рустам	84.0	76.0	0.0	76.0
Бітфур Микола	0.0	0.0		
Ровавюк Тамара	84.0	94.0	0.0	0.0
Невманежний Максим	8.0	22.0		
Баретенько Артем	0.0	0.0		

Рисунок 2.11 – Сторінка результатів змагання групи

Для навігації між сторінками потрібно додати елемент навігації на кожну зі сторінок. Цей елемент містить посилання на кожну з інших сторінок для переходу між ними, відображає поточну сторінку, дозволяє змінити мову інтерфейсу. Також для цілей адміністрування потрібно додати кілька сторінок, що використовуватимуться для керування доступними компіляторами, під’єднаними тестувальними серверами та іншою сервісною інформацією. На рисунку 2.12 зображено вигляд елементу навігації для адміністратора.

Рисунок 2.12 – Елемент навігації між сторінками

Отже, для клієнтської частини веб-ресурсу розроблено інтерфейс користувача, що забезпечує виконання усіх необхідних функцій розроблюваної системи. Інтерфейс складається з сторінок, виконаних за технологіями HTML, CSS і JavaScript.

2.4 Розробка структурних схем і моделей роботи програми

Модульне програмування – спосіб організації програми як сукупності невеликих незалежних модулів, які виконують певну частину функціоналу всієї системи. Модульне програмування має такі переваги, як можливість паралельної розробки, зрозумілість (можливо вивчати систему частинами), гнучкість продукту (можливо змінювати функціонал одного модуля без зміни інших) [20].

З огляду на основні функції розроблюваної програми, її можна розділити на такі модулі:

- модуль комунікації з серверною частиною;
- модуль зберігання та оцінювання розв’язків;
- модуль організації груп і проведення змагань;
- модуль створення, збереження і відображення задач;
- модуль комунікації з користувачами можливостями електронної пошти;
- модуль роботи з сесіями та особистим кабінетом;
- модуль для тестування на стороні клієнту
- модуль адміністрування.

Для відображення взаємозв’язку між всіма модулями розроблюваної програми та іншими використаними системами побудовано діаграму компонентів, що зображено на рисунку 2.13.

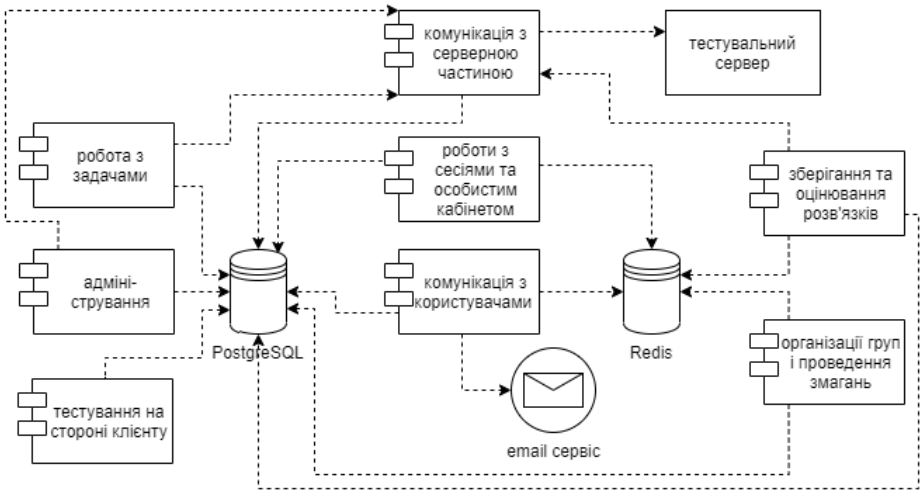


Рисунок 2.13 – Діаграма компонентів програмного продукту

Також, для розуміння моделей роботи програми побудовано діаграму послідовності (рисунок 2.14), що описує взаємодію модулів і систем між собою та з користувачем під час одного з варіантів використання програмного продукту – відправка розв’язку задачі під час змагання.

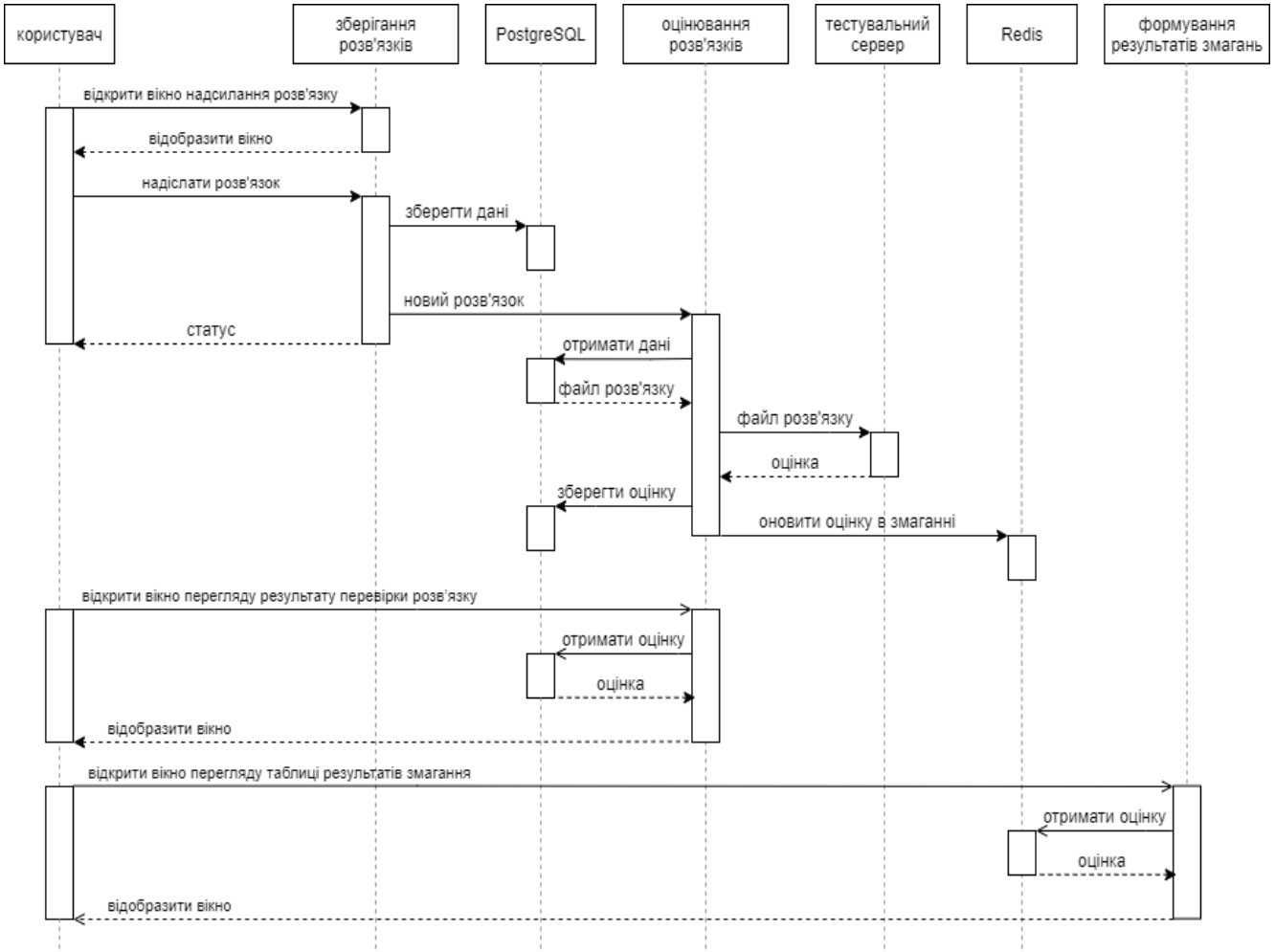


Рисунок 2.14 – Діаграма послідовності одного варіанту використання

2.5 Удосконалення методу оцінювання розв’язків задач спортивного програмування

Задачі зі сфери спортивного програмування передбачають розробку алгоритму для її вирішення та написання програми на одній з доступних для перевірки мов програмування. Після написання програми користувач надсилає вихідний код на сервер. Сервер виконує компіляцію рішення, запускає отриманий бінарний файл для тестування його роботи та визначає результат

оцінювання. Перевірка рішень виконується з використанням тестів – сутності, що містять вхідні дані та очікуваний результат, або програму їх генерації для більш складних задач. Власне перевірка виконується спеціальною програмою, написаною автором задачі, що аналізує отримані дані та параметри роботи рішення користувача й видає вердикт.

На результат оцінювання впливають такі параметри, як відповіді, надані програмою, час її роботи, а також об’єм використаної в процесі оперативної пам’яті. Якщо програма вивела не вірну відповідь на деякі з тестів, в різних системах оцінювання користувач може отримати різний результат. Так, система ACM оцінює рішення лише як «прийнято» чи «не прийнято», а система Kirov та інші подібні нараховує користувачу бали відповідно до пройдених тестів [12]. Тобто, деякі системи оцінювання передбачають можливість здачі не повного рішення. Однак, якщо програма не вклалась в обмеження по часу чи пам’яті, але видала правильний результат, в будь-якій системі результат її оцінки буде незадовільним. Обмеження на ці ресурси вказуються автором задачі, щоб виявляти рішення, які видають правильний результат, але мають надлишково-складний алгоритм, що недоцільно використовує системні ресурси.

Передбачається, що користувачі повинні оптимізувати алгоритм свого рішення та вкластись в обмеження, визначені умовою. Разом з цим, через певні технічні особливості, один й той самий алгоритм може виконуватись за різні проміжки часу чи використовувати різні об’єми оперативної пам’яті на різних мовах програмування. Це може бути пов’язано з проведенням початкової ініціалізації середовища виконання, що особливо помітно для інтерпретованих мов програмування, чи з особливостями механізмів керування пам’яттю й різними оптимізаціями.

Автори задач часто пишуть лише одне рішення, зазвичай на мові програмування C++, та вказують технічні обмеження опираючись на параметри його роботи і не перевіряють роботу цього ж алгоритму на інших мовах. Інколи це призводить до того, що на більш повільних мовах програмування буває

неможливо здати задачу – алгоритм вже є оптимальним, однак обмеження в часі чи пам’яті дуже низькі та не дозволяють програмі успішно виконатись.

З одного боку, потрібно вказувати збалансовані технічні обмеження, що б дозволяли здавати рішення на усіх підтримуваних мовах. Однак, з іншого боку, занадто високі обмеження дозволять здавати рішення з неоптимальними алгоритмами, але запрограмовані на швидких мовах програмування.

Тому, щоб забезпечити можливість успішної здачі розв’язків на усіх мовах програмування, потрібно вносити корективи в технічні обмеження на роботу програми. Це дозволить вказати правильні обмеження для кожної з мов програмування та забезпечить перевірку саме алгоритму і розв’язку користувача, не залежно від технологій, що він використовує.

Проте вручну вказувати обмеження на кожну з підтримуваних мов для кожної задачі неефективно – автору потрібно реалізувати той самий алгоритм правильного рішення на усіх цих мовах, заміряти параметри роботи і записати їх до задачі. Досягнути достатнього результату відносно-невеликими затратами можна, використавши вагові коефіцієнти для налаштування обмежень в часі й пам’яті для задачі під окремі мови програмування та компілятори.

Спершу потрібно визначити еталонні значення, з якими виконуватимемо порівняння продуктивності роботи програм на тій чи іншій мові. Оскільки більшість авторів використовують мову програмування C++ для своїх рішень, доцільно буде використати саме її. Також варто розробити кілька типових алгоритмів, що будуть використовуватись для подальших замірів. Після цього, під час додання підтримки на сервер для нової мови програмування, потрібно заміряти продуктивність роботи обраних алгоритмів на цій мові та порівняти з еталонними значеннями. За результатами порівняння можна визначити наближену функцію, що б відображала еталонні значення часу й пам’яті до випробуваних. Достатню точність може забезпечити навіть лінійна функція. Така функція містить лише два значення – фіксоване зміщення значення відносно еталонного та коефіцієнт пропорційності.

Лінійна функція є простим, але ефективним способом приблизного відображення часу та пам’яті роботи програми на іншій мові програмування відносно еталонного рішення автора. Так, коефіцієнт «a» може вказувати на накладні витрати пам’яті для зберігання кожного об’єкту у мовах з динамічною типізацією, а коефіцієнт «b» – на додаткові затрати часу для ініціалізації середовища виконання. Оскільки еталонною мовою обрано C++, для неї значення коефіцієнтів будуть однаковими для пам’яті й часу і матимуть такі значення: «a» – 1, «b» – 0. Для інших мов програмування потрібно провести описане дослідження використовуваних ресурсів і визначити приблизні значення цих коефіцієнтів як для часу роботи програми, так і для оперативної пам’яті.

Таким чином, опишемо покроковий алгоритм удосконаленого методу оцінювання розв’язків задач спортивного програмування з використанням вагових коефіцієнтів:

1. Початок.
2. Вибір мови та введення вихідного коду розв’язку користувачем.
3. Завантаження даних про задачу – тести, обмеження.
4. Завантаження вагових коефіцієнтів для обраної мови.
5. Корекція обмежень задачі на основі завантажених вагових коефіцієнтів.
6. Компіляція вихідного коду у виконуваний файл.
7. Для кожного тесту:
    a. запуск виконуваного файлу відповідно до вхідних даних тесту;
    b. аналіз затрачених при роботі часу та оперативної пам’яті з урахуванням скорегованих обмежень;
    c. запуск програми чекеру для отримання вердикту перевірки для поточного тесту.
8. Вивід результатів перевірки користувачу із коментарями чекера.
9. Кінець.

2.6 Удосконалення методу тестування робіт зі спортивного програмування

WebAssembly/WASM – це низькорівневий бінарний формат виконуваного коду, що запускається у браузері [21]. Цей формат може виступати ціллю компіляції для багатьох високорівневих мов програмування, зокрема C та C++, що дозволяє запускати такі програми в середовищі браузера.

Можливості технології WebAssembly дозволяють перенести етапи компіляції вихідних кодів рішень користувачів, запуску отриманих виконуваних файлів та аналіз результатів їх роботи на сторону клієнта. Для багатьох сучасних мов програмування існують компілятори, що дозволяють із вихідного коду програми отримати байт-код WASM, який можна запустити прямо у браузері користувача. Так, для мов C/C++ існує декілька утиліт для цього, наприклад emscripten [22] і clang [23]. У WASM байт-код для роботи у браузері можна перевести й сам компілятор clang [24]. Все це дозволяє виконати компіляцію вихідного коду користувача й запуск програми на тестах до задачі прямо у браузері без застосування серверу.

Перевагами такого підходу є повне усунення питань безпеки запуску невідомого коду, наданого користувачем, на сервері. Разом з цим, для процесів компіляції й тестування використовуються ресурси комп’ютера користувача, а не серверу. Таким чином ліквідуються можливі затримки в отриманні результатів тестування при навантаженні на сервер, а розробка і підтримка роботи самої системи тренування і оцінювання робіт зі спортивного програмування стає дешевшою.

Проте, пропонований підхід й технологія WebAssembly мають певні недоліки. Розглянемо основні такі недоліки та їх вплив на можливість використати такий підхід у подібних системах.

Продуктивність роботи програм була одним з головних мотиваційних факторів для розробки технології WebAssembly. У реальності програми, скомпільовані в такий формат, гарно показують себе в алгоритмах зі складними масивними обчисленнями, проте програють при великій кількості операцій з

пам’яттю чи взаємодії із зовнішнім середовищем через повільний інтероп із JavaScript [21]. За результатами дослідження [25], WASM в середньому на 30% швидший, ніж такий же код на JavaScript, проте на 50% повільніший, ніж «нативний» код. Для рішень задач у сфері спортивного програмування одним із важливих критеріїв є оптимальність реалізованого алгоритму, яка оцінюється за часом роботи програми. Оскільки код, скомпільований у формат WebAssmebly і виконаний у браузері, працює повільніше, може виникнути проблема з оцінюванням оптимальності алгоритму. Частково цю проблему можна нівелювати, якщо використати коригуючі коефіцієнти для приведення часу роботи програми у браузері до нормалізованого виду. Також така проблема має менший вплив при використанні в процесі тренування, а не під час змагань.

Програми, скомпільовані у байт-код WASM, під час виконання у браузері не мають самостійного доступу до будь-яких зовнішніх ресурсів, а для реалізації такого доступу можливо лише використовувати JavaScript як «міст» [26]. Таким чином, необхідно виділити можливості, що мають бути доступні для програм користувачів у цілях вирішення алгоритмічних задач (наприклад, функції введення та виведення), реалізувати їх за допомогою JavaScript та передати їх у середовище виконання програми користувача. Для вирішення цієї проблеми також можна використати існуючі рішення, наприклад WebAssembly System Interface [27].

Також, постає питання довіри до результатів перевірки рішення на стороні користувача. Адже користувач із зловмисним наміром може змінити вердикт перевірки на свою користь перед його надсиланням на сервер. Також, для запуску тестування у браузері, системі користувача необхідно передати повний набір вхідних даних та відповідей. Це дає можливість переглянути тести, що зазвичай не мають бути видимі. Доступ до тестів дає користувачу несправедливу перевагу та багато можливостей нечесного вирішення задачі. Наприклад, замість алгоритму, що вирішує поставлену задачу, достатньо написати умови для вхідних даних.

Тому, перенесення описаних етапів тестування робіт на сторону клієнта має місце лише в тренуваннях. Під час реальних змагань описані недоліки мають значно більший вплив на хід змагання, отже застосування цих технологій можливе лише для попереднього тестування робіт на публічно-доступних тестах, а повне оцінювання має відбуватись на стороні серверу.

Таким чином, опишемо покроковий алгоритм удосконаленого методу тестування робіт зі спортивного програмування з перенесенням певних етапів у браузер клієнта:

1. Початок.
2. Введення вихідного коду розв’язку користувачем.
3. Завантаження клієнтом даних про задачу – тести, обмеження.
4. Завантаження клієнтом вихідного коду програми чекера, оптимізованого під роботу у браузері.
5. Завантаження і запуск модуля для компіляції clang, попередньо зібраного у формат WASM.
6. Компіляція вихідних кодів у виконувані модулі за допомогою clang і модифікованої під WebAssembly стандартної бібліотеки C++.
7. Для кожного тесту:
    a. запуск виконуваного модулю розв’язку відповідно до вхідних даних тесту;
    b. аналіз затрачених при роботі часу та оперативної пам’яті з урахуванням обмежень задачі;
    c. запуск виконуваного модулю чекеру для отримання вердикту перевірки для поточного тесту.
8. Вивід результатів перевірки користувачу із коментарями чекера.
9. Кінець.

2.7 Розробка основних алгоритмів роботи програми

Розробка системи полягає в тому, що необхідно розробити алгоритми для всіх її модулів, частин і функцій. Розроблюваний веб-ресурс призначено для

розміщення олімпіадних задач з програмування і автоматизованої перевірки їх рішень. Розповсюдженим форматом зберігання таких задач є архів з файлом конфігурації, що містить умову задачі, приклади вхідних й вихідних даних, описує обмеження для задачі, тощо. Тому побудуємо алгоритми основних сценаріїв взаємодії з системою – надсилання розв’язку (рисунок 2.15), оцінювання розв’язку (рисунок 2.16), формування таблиці з результатом змагань (рисунок 2.17), імпортування задачі з архівного файлу (рисунок 2.18).

Алгоритм надсилання розв’язку складається з наступних кроків:

Крок 1. Початок.

Крок 2. Відображення вікна надсилання розв’язку у браузері користувача.

Крок 3. Введення користувачем компілятора і файлу з кодом розв’язку.

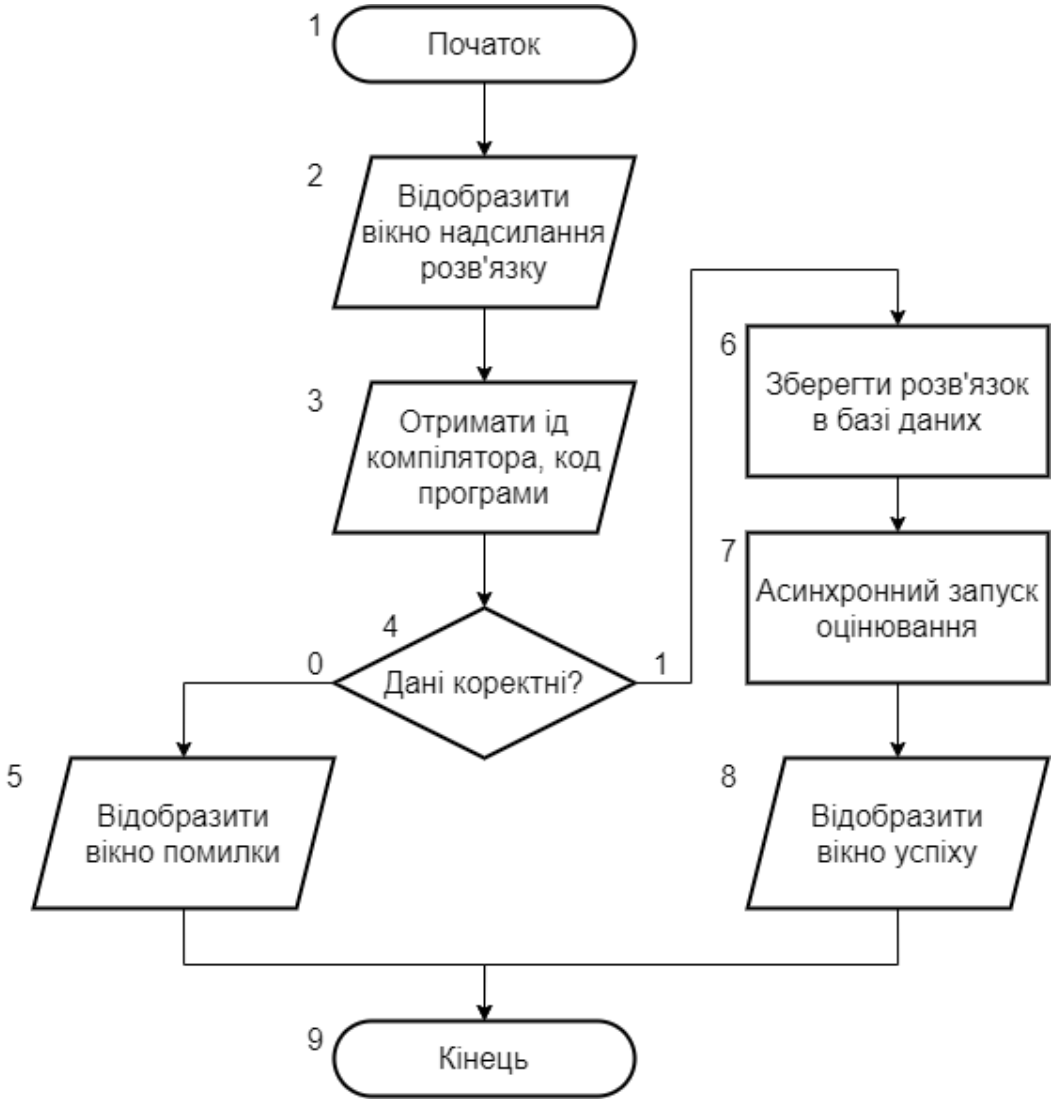


Рисунок 2.15 – Алгоритм надсилання розв’язку

Крок 4. Перевірка на коректність введених даних – присутність коду, підтримка компілятора. Якщо перевірка успішна – перейти до кроку 6.

Крок 5. Відобразити вікно помилки у браузері користувача. Перейти до кроку 9.

Крок 6. Зберегти інформацію щодо розв’язку у базі даних.

Крок 7. Запустити оцінювання розв’язку асинхронно.

Крок 8. Відобразити вікно успіху у браузері користувача.

Крок 9. Кінець.

Алгоритм оцінювання розв’язку складається з наступних кроків:

Крок 1. Початок.

Крок 2. Завантажити інформацію щодо розв’язку з бази даних.

Крок 3. Надіслати код розв’язку на тестувальний сервер.

Крок 4. Присвоїти змінним s і ms 0.

Крок 5. Цикл по тестам поточної задачі. Після циклу перейти до кроку 9.

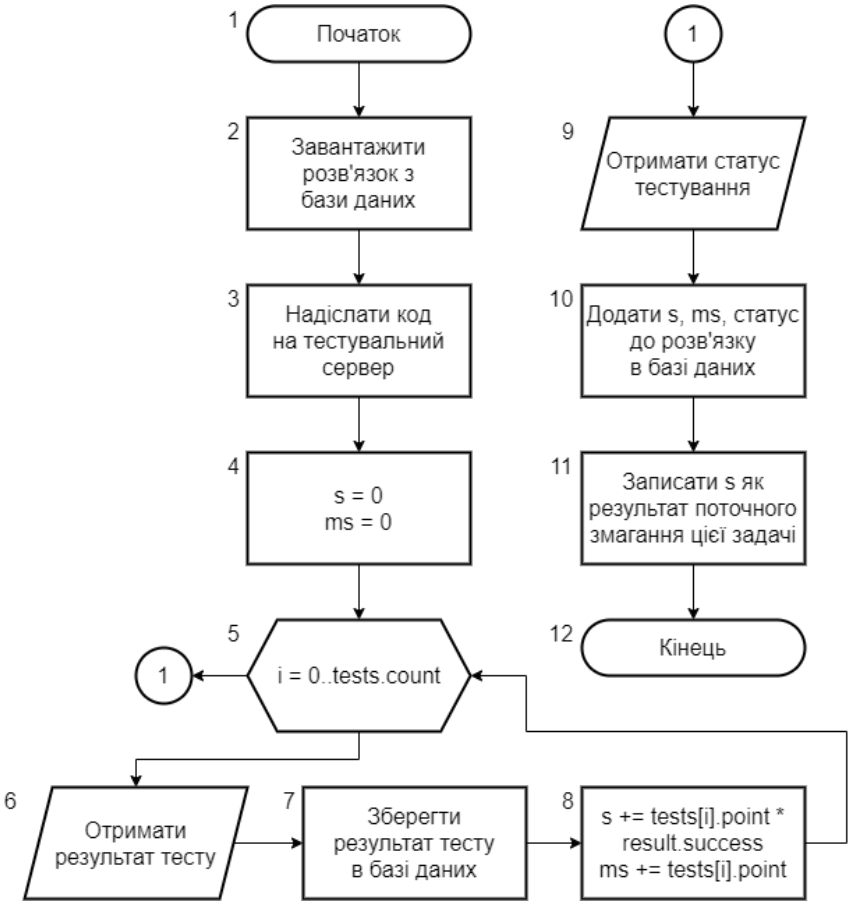


Рисунок 2.16 – Алгоритм оцінювання розв’язку

Крок 6. Введення результату поточного тесту тестувальним сервером.

Крок 7. Збереження результату поточного тесту в базі даних.

Крок 8. Додати до змінної s кількість балів поточного тесту, помножену на показник успішності результату; до змінної ms кількість балів поточного тесту. Перейти до кроку 5.

Крок 9. Введення статусу тестування сервером.

Крок 10. Додати s як оцінку, ms як максимальну оцінку та статус до розв’язку в базі даних.

Крок 11. Записати s як результат поточної задачі для поточного користувача в поточному змаганні.

Крок 12. Кінець.

Алгоритм формування таблиці з результатом змагань складається з наступних кроків:

Крок 1. Початок.

Крок 2. Ініціалізація двовимірного масиву d з розмірами кількість користувачів та кількість задач + 1.

Крок 3. Цикл зі змінною i по користувачам в поточному змаганні. Після циклу перейти до кроку 9.

Крок 4. У масив d, комірку з індексом «i, 0» помістити ім’я поточного користувача.

Крок 5. Цикл зі змінною j по задачам в поточному змаганні. Після циклу перейти до кроку 3.

Крок 6. Перевірка, чи знаходиться оцінка в кеші. Якщо так – перейти до кроку 8.

Крок 7. Пошук останнього успішного розв’язку поточного задачі поточного користувача, запис його оцінки в кеш..

Крок 8. Запис оцінки з кешу в масив d, комірку з індексом i, j + 1.

Крок 9. Відобразити вікно перегляду таблиці результатів на змагання основі масиву d у браузері користувача.

Крок 10. Кінець.

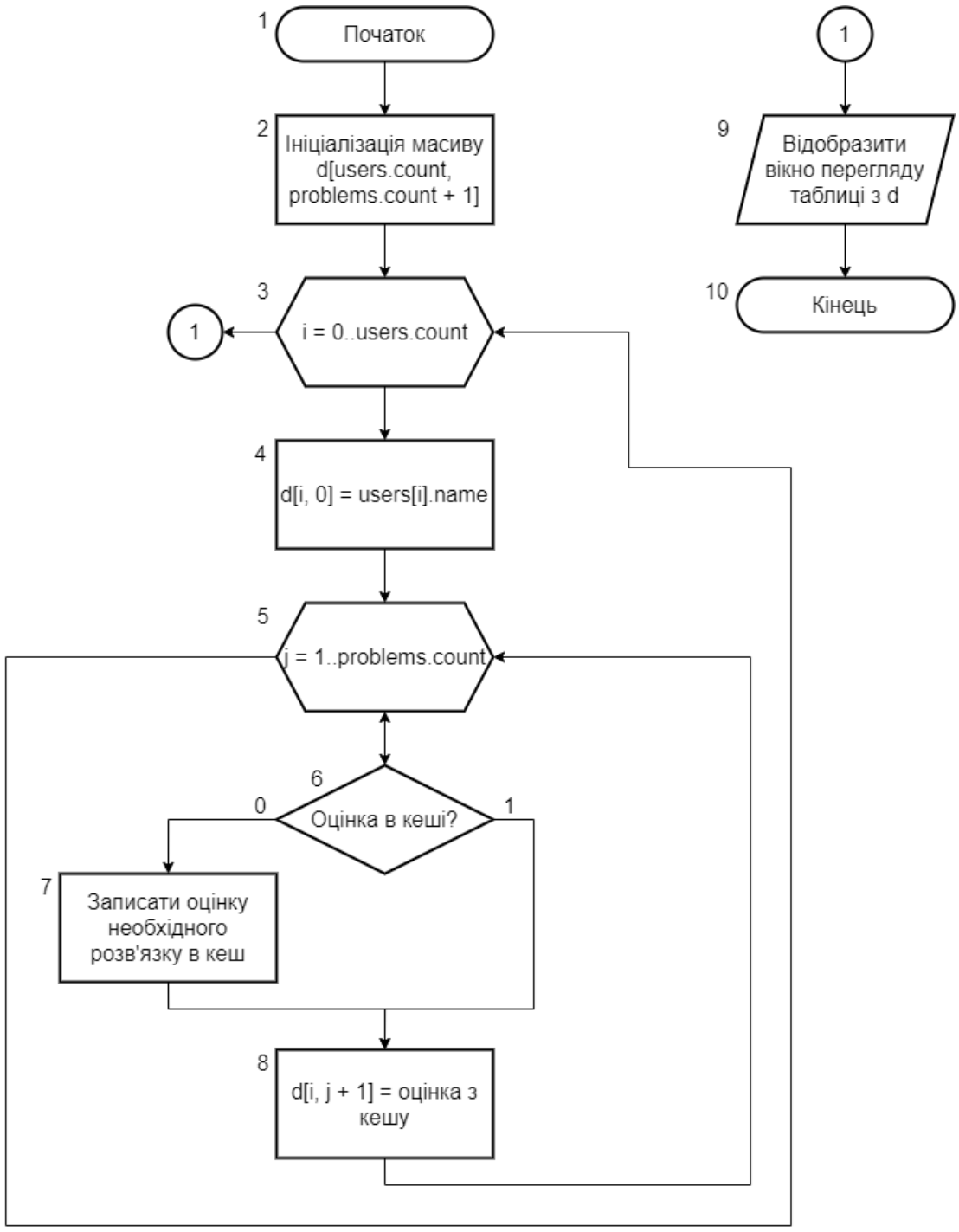


Рисунок 2.17 – Алгоритм формування таблиці з результатом змагань

Алгоритм імпортування задачі з архівного файлу складається з наступних кроків:

Крок 1. Початок.

Крок 2. Вибір користувачем файлу архіву з описом задач.

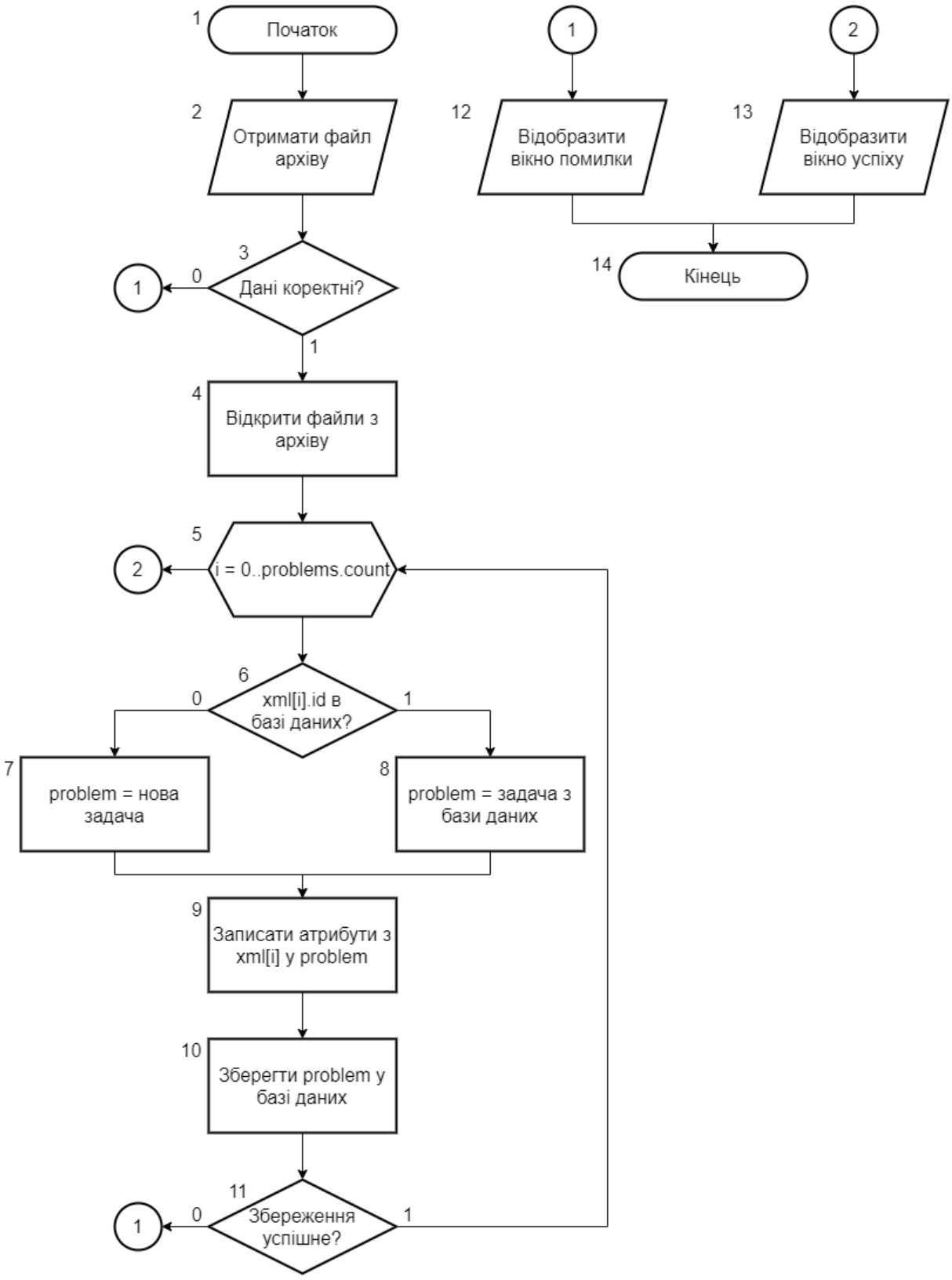


Рисунок 2.18 – Алгоритм імпортування задачі з архівного файлу

Крок 3. Перевірка на коректність введених даних – присутність архіву та xml-файлу в ньому, правильність структури xml-файлу. Якщо перевірка неуспішна – перейти до кроку 12.

Крок 4. Завантажити архів, розпакувати його, відкрити xml-файл з описом задач та файли тестових пар.

Крок 5. Цикл зі змінною і по задачам в поточному архіві. Після циклу перейти до кроку 13.

Крок 6. Перевірка на присутність ідентифікатору задачі з індексом і в базі даних. Якщо перевірка успішна – перейти до кроку 8.

Крок 7. Створення нової задачі з таким ідентифікатором, запис її до змінної problem. Перейти до кроку 9.

Крок 8. Завантаження задачі з бази даних по ідентифікатору в «problem».

Крок 9. Читання усіх атрибутів поточної задачі з xml-файлу, додання їх у задачу зі змінної problem. Для опису файлів тестових пар використовується атрибут, що вказує на шлях відповідних файлів всередині архіву.

Крок 10. Спроба зберегти отримані дані зі змінної problem у базу даних.

Крок 11. Перевірка чи успішне зберігання в базу даних. Якщо так – перейти до кроку 5.

Крок 12. Відобразити вікно помилки у браузері користувача. Перейти до кроку 14.

Крок 13. Відобразити вікно успіху у браузері користувача.

Крок 14. Кінець.

2.8 Висновки

У другому розділі було проаналізовано особливості побудови веб-інтерфейсу та характеристики сутностей інформаційного забезпечення, з якими повинен працювати ресурс. Збереження простих даних вирішено винести до СКБД, а файли зберігати з допомогою спеціального сховища. Розроблено схему взаємодії з серверною частиною із шифруванням Secure Sockets Layer, авторизацією через передачу токену, форматом даних JSON. Доступні методи організовано за принципами REST. Виділено сервісний ресурс для роботи з серверними частинами, ресурси для читання чи оновлення розв'язків, результатів перевірки, задач, тестів та коефіцієнтів компіляторів. Розроблено

інтерфейс користувача програмного продукту з десяти основних сторінок, елементу навігації та сторінок адміністрування. Розроблено ключові алгоритми роботи системи – надсилання розв’язку, автоматичного оцінювання розв’язку, формування таблиці з результатом змагань, імпортування задачі з архівного файлу. Побудовано діаграми компонентів та послідовності для відображення взаємодії між модулями додатку, серверною частиною та іншими сервісами.

Проаналізовано можливість удосконалення методу оцінювання розв’язків задач спортивного програмування за рахунок вагових коефіцієнтів для налаштування обмежень у часі й пам’яті для задачі, що враховують особливості різних мов програмування та компіляторів. Використання різних обмежень для різних мов програмування забезпечує можливість успішної здачі розв’язків на усіх підтримуваних мовах, оскільки задача матиме правильні обмеження для кожної них, що забезпечить перевірку саме алгоритму. Запропонований метод використання вагових коефіцієнтів спрощує таке налаштування обмежень, оскільки автору задачі більше не потрібно реалізувати однаковий алгоритм правильного рішення на усіх мовах програмування, що підтримуються системою тестування.

Також, розглянуто можливості перенесення різних етапів тестування робіт у браузер клієнта за допомогою технології WebAssembly. Виявлено переваги та недоліки у порівнянні з вирішенням таких задач на стороні серверу. Значна частина роботи переноситься на клієнтську сторону, що призводить до зниження навантаження на систему й скорочення часу очікування вердикту. Проте, існуючі недоліки дають можливість використати такий підхід в повній мірі лише під час тренування до змагань зі спортивного програмування та навчання вирішенню алгоритмічних задач, але не на самих змаганнях через проблему довіри до результатів перевірки лише на стороні клієнта.

3 РОЗРОБКА ПРОГРАМНИХ ЗАСОБІВ КЛІЄНТСЬКОЇ ЧАСТИНИ

3.1 Варіантний аналіз і обґрунтування вибору засобів реалізації

Перед розробкою будь-якої програми необхідно обрати засоби її реалізації. Правильний вибір таких засобів дуже важливий, оскільки від нього залежить складність розробки, ефективність і швидкість роботи програми, безпека зберігання даних. Для реалізації серверного застосунку потрібно обрати середовище його роботи (операційну систему), мову програмування з необхідними бібліотеками й фреймворками, а також технології баз даних для зберігання даних.

Windows, Mac OS і Linux – популярні сучасні операційні системи, що отримують оновлення, підтримку та регулярний випуск нових версій [28].

ОС Windows випускається компанією Microsoft та є найбільш популярною системою у всьому світі. Так, за даними сайту NetApplications, на березень 2020 року Windows була встановлена на 89.21% комп’ютерів, Mac OS – 8.94%, а Linux – 1.36% [29]. Також існує спеціальна версія цієї операційної системи, спроектована для роботи в ролі сервера.

Mac OS – достатньо поширена операційна система, розроблена в корпорації Apple. Підтримується робота лише на комп’ютерах Mac, що обмежує сферу її використання. Ця система спроектована, в основному, для використання на персональних комп’ютерах. Зважаючи на деякі її обмеження та цінову політику Apple, Mac OS недоцільно використовувати для розміщення веб-сайту.

Linux – загальна назва UNIX-подібних операційних систем на основі однойменного ядра. Вихідні коди ядра Linux та більшості програм у ній доступні будь-кому для безперешкодного використання, зміни та поширення. Завдяки підтримці спільноти та великих компаній ОС набула популярності для серверного застосування. UNIX вважається ефективною у роботі з мережевою платформою для будь-якого типу серверу [30].

У таблиці 3.1 зведено результати порівняльного аналізу операційних систем з огляду на розробку веб-додатків.

Таблиця 3.1 – Порівняння операційних систем

Характеристика	Операційна система		
	Windows	Mac OS	Linux
Безкоштовна	-	+/-	+
Відкритий вихідний код	-	-	+
Наявність серверної версії	+	-	+
Наявність технічної підтримки	+	+	-
Низькі технічні вимоги	-	-	+

Для створення веб-ресурсів зазвичай застосовують реляційну базу даних. Реляційна база даних – тіло зв’язаної інформації, що зберігається у двовимірних таблицях, нагадуючи адресну чи телефонну книгу [31]. Серед таких баз даних найбільше поширення отримали PostgreSQL та MySQL.

PostgreSQL – це потужна об'єктно-реляційна база даних із відкритим кодом, яка використовує та розширює мову SQL у поєднанні з багатьма функціями, які безпечно зберігають та масштабують найскладніші навантаження даних, заслужила сильну репутацію за свою перевірену архітектуру, надійність, підтримку цілісності даних [32]. Ще однією вагомою перевагою PostgreSQL є те, що вона намагається якомога більше відповідати стандарту SQL.

MySQL – найпопулярніша, на сьогодні, база даних. Найбільше відома за високу швидкодію обробки великої кількості даних та простоту використання. Однак, для досягнення своєї швидкості, ця база даних має певні розходження зі стандартом SQL, ігнорує деякі перевірки цілісності.

Разом з реляційною базою даних також часто використовують простіше сховище даних з метою кешування. Для таких цілей використовуються Redis чи Memcached. Це системи зберігання даних в оперативній пам'яті, які забезпечують швидкий доступ до часто використовуваних даних. Memcached випереджає Redis у швидкодії, коли кількість зберігаємих даних перевищує 100 000 записів, однак у випадку невеликих об’ємів виграє Redis [33]. Також Redis підтримує набагато більше типів даних, включаючи рядки, хеш, список, набір і сортований набір.

Одними з найпоширеніших серверних мов програмування є JavaScript (Node.js), Ruby, C#.

Ruby – інтерпретована мова програмування, що виконує принципи об'єктно-орієнтованого програмування. Мова вирізняється простотою та ефективністю розробки програм, елегантним синтаксисом, який приємно читати і писати [34]. Основну популярність мова здобула завдяки фреймворку для створення веб-застосунків – Rails. Він надає простий спосіб опису семантики веб-застосунків і потребує мінімум налаштувань завдяки принципу угоди над конфігурацією.

JavaScript – інтерпретована прототипна мова програмування з динамічною типізацією. Спочатку задумувалась для простих сценаріїв у веб-браузері, однак її почали використовувати і для написання серверних застосунків. Використання однієї мови для кожного шару додатку – це чудовий спосіб зменшити складність програмного забезпечення, дозволяє уникати невідповідностей та полегшує повторне використання коду [35]. Проте ця мова має деякі проблеми, допущені при початковому проектуванні: немає підтримки цілих чисел, використовується слабка типізація та агресивне приведення типів, тощо.

C# – компільована мова програмування з безпечною системою типізації, розроблена в Microsoft. Мова пропонує об’єктно-орієнтований підхід для розробки, має багато можливостей і постійно розвивається. Також підтримується асинхронний код, на кшталт сучасного JavaScript. Для розробки

веб-серверів з її використанням існує фреймворк ASP.NET, який дещо подібний до Rails, оскільки обидва використовують моделі MVC та принцип TDD [36]. Однак, C# і ASP.NET націлені переважно на використання інших технологій фірми Microsoft, зокрема для сервера та бази даних. На мою думку, ASP.NET менше підходить для розробки невеликих проектів через свою складність та громіздкість.

Результати порівняльного аналізу мов програмування для реалізації веб-додатку наведено у таблиці 3.2.

Таблиця 3.2 – Порівняння мов програмування

Характеристика	Мова програмування		
	C#	JavaScript (Node.JS)	Ruby
Об'єктно-орієнтована	+	+/-	+
Компільована	+/-	-	-
Підтримка Linux	+/-	+	+
Бібліотеки для веб-застосунків	+	+	+
Проста інтеграція з базою даних	-	+	+

Єдиним середовищем роботи веб-інтерфейсу є браузери, які підтримують лише технології HTML, CSS і JavaScript. Вибір засобів його реалізації полягає лише у підборі необхідних бібліотек, що можуть спростити розробку чи дати якісніший результат. Зазвичай вони містять шаблони CSS та HTML для типографіки, форм, кнопок, навігації й інших компонентів інтерфейсу, а також додаткові розширення JavaScript. Bootstrap, розроблений в Twitter, є найпопулярнішою бібліотекою, містить багато компонентів та сторонніх плагінів, надає широкі можливості з налаштування. Semantic UI – ще одна бібліотека компонентів інтерфейсу, розроблена як альтернатива Bootstrap, що

намагається використовувати аспекти природної мови для опису інтерфейсу. Однак підтримка Semantic UI наразі призупинена.

Таким чином, враховуючи наведений аналіз і таблиці порівняння мов програмування та операційних систем, для реалізації клієнтської частини веб-ресурсу для розміщення і автоматизованої перевірки олімпіадних задач використаємо операційну систему Linux, мову програмування Ruby з фреймворком Rails, реляційну базу даних PostgreSQL та систему для кешування Redis. А для розробки інтерфейсу використаємо бібліотеку Bootstrap.

3.2 Розробка модулів для роботи з розв'язками, змаганнями і задачами

Фреймворк Rails побудований за принципами Model-View-Controller. Таким чином вихідний код кожного з ресурсів програми потрібно розділити на:

- моделі, що відповідають за роботу з базою даних та перевірку коректності своїх атрибутів;
- представлення, які виконують необхідні перетворення для відображення даних з моделей користувачеві;
- контролери, що здійснюють обробку запитів від користувача та є свого роду вхідними точками програми.

Усі моделі представлені відповідною таблицею у СКБД. Їх атрибути не мають надлишковості та є нормалізованими відповідно до нормальних форм, що забезпечує мінімальні розміри та достатню швидкодію бази даних.

Окрім цих класів існують інші класи, які містять додаткову бізнес-логіку ресурсів, реалізацію фонових задач та методи надсилання електронних листів. Для розробки представлень інколи потрібно запрограмувати певний функціонал на стороні клієнту чи редагувати стилі відображення, що здійснюється доданням відповідних файлів з розширенням js чи css.

Для роботи з програмою використовується веб-браузер, що надсилає HTTP-запити до програми і відображає сторінки, що надходять у відповідь на ці запити. Розробка веб-ресурсу ведеться за принципами REST, тому для

функціонал поділяється на окремі ресурси. Кожен такий ресурс містить запити для відображення та редагування його інформації.

Модуль для роботи з задачами містить ресурси problems та archives, що реалізовано у класах ProblemsController, ArchivesController, Problem, ProblemTranslation, Example, Test, Archive, ProcessProblemArchiveJob та відповідних файлах представлення.

Запит «GET /problems» використовує метод index контролера ProblemsController, для відображення – представлення problems/index, для бази даних – методи моделі Problem: all щоб отримати список задач, includes щоб одночасно отримати автора задачі, order для сортування по ідентифікатору, page для пагінації. Вхідні дані відсутні. Вихідні дані – сторінка зі списком задач.

Запит «GET /problems/:id» використовує метод show контролера ProblemsController, для відображення – представлення problems/show, щоб отримати задачу з бази даних – метод find моделі Problem. Вхідні дані – ідентифікатор задачі. Вихідні дані – сторінка з описом задачі.

Запит «GET /problems/new» використовує метод new контролера ProblemsController, для відображення – представлення problems/new. Вхідні дані відсутні. Вихідні дані – сторінка з формою створення задачі.

Запит «POST /problems» використовує метод create контролера ProblemsController, для відображення – представлення problems/new чи problems/show в залежності від успіху операції, щоб записати задачу у базу даних – методи new і save моделі Problem. Вхідні дані – ідентифікатор користувача, відомості задачі, перекладів, прикладів, тестів. Вихідні дані – сторінка зі статусом операції та формою створення задачі чи її описом.

Запит «GET /problems/:id/edit» використовує метод edit контролера ProblemsController, для відображення – представлення problems/edit, щоб отримати задачу з бази даних – метод find моделі Problem. Вхідні дані – ідентифікатор задачі. Вихідні дані – сторінка з формою редагування задачі.

Запит «PATCH /problems/:id» використовує метод update контролера ProblemsController, для відображення – представлення problems/edit чи problems/show в залежності від успіху операції, щоб оновити задачу у базі даних – метод update моделі Problem. Вхідні дані – ідентифікатор та інші відомості про задачу, переклади, приклади, тести. Вихідні дані – сторінка зі статусом операції та формою редагування задачі чи її описом.

Запит «DELETE /problems/:id» використовує метод destroy контролера ProblemsController, для відображення – представлення problems/index, щоб видалити задачу з бази даних – метод destroy моделі Problem. Вхідні дані – ідентифікатор задачі. Вихідні дані – сторінка зі списком залишених задач.

Запит «GET /archives/new» використовує метод new контролера ArchivesController, для відображення – представлення archives/new. Вхідні дані відсутні. Вихідні дані – сторінка з формою імпорту задачі з архівного файлу.

Запит «POST /archives» використовує метод create контролера ArchivesController, щоб імпортувати архівний файл – методи new і save сервісного класу Archive. Вхідні дані – ідентифікатор користувача, архівний файл, ідентифікатор каналу для повідомлень. Вихідні дані відсутні.

Модель Problem відповідає за дані задачі. Атрибути – ідентифікатор, час створення, час редагування, ідентифікатор компілятору чекера, ідентифікатор автора, обмеження пам'яті та часу. Асоціації – файл чекера, переклади, приклади, тести, розв’язки. Також модель приймає атрибути для асоціацій, що не мають пов’язаних ресурсів.

Модель ProblemTranslation відповідає за дані перекладу. Атрибути – ідентифікатор, час створення, час редагування, ідентифікатор задачі, мова, назва, автор, умова, технічні обмеження.

Модель Example відповідає за дані прикладу. Атрибути – ідентифікатор, час створення, час редагування, ідентифікатор задачі, текст введення, текст виведення.

Модель Test відповідає за дані тесту. Атрибути – ідентифікатор, час створення, час редагування, ідентифікатор задачі, номер, бал. Асоціації – файл введення, файл виведення. Також модель має індекс по номеру для сортування.

Імпорт задачі з архівного файлу може зайняти деякий час, тому цей процес винесено у фонову задачу. Сервісний клас Archive використовується для тимчасового збереження надісланого файлу задачі та запуску задачі на його обробку. Клас ProcessProblemArchiveJob представляє цю фонову задачу й містить реалізацію алгоритму імпортування задачі з архівного файлу.

Модуль для роботи з розв’язками містить ресурс submissions, що реалізовано у класах SubmissionsController, Submission, Compiler та відповідних файлах представлення.

Запит «GET /submissions» використовує метод index контролера SubmissionsController, для відображення – представлення submissions/index, для бази даних – методи моделі Submission: all щоб отримати список розв’язків, where для пошуку по задачі, автору чи змаганню, includes щоб одночасно отримати компілятор, автора розв’язку й автора задачі, order для сортування по часу надсилання, page для пагінації. Вхідні дані – ідентифікатори задачі, користувача та змагання. Вихідні дані – сторінка зі списком надісланих розв’язків.

Запит «GET /submissions/:id» використовує метод show контролера SubmissionsController, для відображення – представлення submissions/show, щоб отримати розв’язок з бази даних – метод find моделі Submission. Вхідні дані – ідентифікатор розв’язку. Вихідні дані – сторінка з деталями розв’язку.

Запит «POST /submissions» використовує метод create контролера SubmissionsController, для відображення – представлення submissions/new чи submissions/show в залежності від успіху операції, щоб записати розв’язок у базу даних – методи new і save моделі Submission. Вхідні дані – ідентифікатор користувача, файл вихідного коду, ідентифікатор компілятора. Вихідні дані – сторінка зі статусом операції та формою створення розв’язку чи його деталями.

Запит «DELETE /submissions/:id» використовує метод destroy контролера SubmissionsController, для відображення – представлення submissions/index, щоб видалити розв’язок з бази даних – метод destroy моделі Submission. Вхідні дані – ідентифікатор розв’язку. Вихідні дані – сторінка зі списком залишених розв’язків.

Модель Submission відповідає за дані розв’язку. Атрибути – ідентифікатор, час створення, час редагування, ідентифікатор задачі, ідентифікатор компілятора, ідентифікатор автора, статус тестування, кількість спроб тестування, оцінка, максимальна оцінка. Асоціації – результати та деталі тестування.

Модель Compiler відповідає за дані компілятора. Атрибути – ідентифікатор, час створення, час редагування, назва, версія, коефіцієнти часу, коефіцієнти пам’яті, тип видимості.

Модуль для роботи зі змаганнями містить ресурси groups, memberships й standing, що реалізовано у класах GroupsController, MembershipsController, StandingsController, Group, Membership, StandingRedisStore та відповідних файлах представлення.

Запит «GET /groups» використовує метод index контролера GroupsController, для відображення – представлення groups/index, для бази даних – методи моделі Group: all щоб отримати список змагань, includes щоб одночасно отримати організатора, order для сортування по назві, page для пагінації. Вхідні дані відсутні. Вихідні дані – сторінка зі списком змагань.

Запит «GET /groups/:id» використовує метод show контролера GroupsController, для відображення – представлення groups/show, щоб отримати змагання з бази даних – метод find моделі Group. Вхідні дані – ідентифікатор змагання. Вихідні дані – сторінка з інформацією про змагання.

Запит «GET /groups/new» використовує метод new контролера GroupsController, для відображення – представлення groups/new. Вхідні дані відсутні. Вихідні дані – сторінка з формою створення змагання.

Запит «POST /groups» використовує метод create контролера GroupsController, для відображення – представлення groups/new чи groups/show в залежності від успіху операції, щоб записати змагання у базу даних – методи new і save моделі Group. Вхідні дані – ідентифікатор користувача, тип видимості, назва та опис змагання. Вихідні дані – сторінка зі статусом операції та формою створення змагання чи його деталями.

Запит «GET /groups/:id/edit» використовує метод edit контролера GroupsController, для відображення – представлення groups/edit, щоб отримати змагання з бази даних – метод find моделі Group. Вхідні дані – ідентифікатор змагання. Вихідні дані – сторінка з формою редагування змагання.

Запит «PATCH /groups/:id» використовує метод update контролера GroupsController, для відображення – представлення groups/edit чи groups/show в залежності від успіху операції, щоб оновити змагання у базі даних – метод update моделі Group. Вхідні дані – ідентифікатор, тип видимості, назва та опис змагання. Вихідні дані – сторінка зі статусом операції та формою редагування змагання чи його деталями.

Запит «DELETE /groups/:id» використовує метод destroy контролера GroupsController, для відображення – представлення groups/index, щоб видалити змагання з бази даних – метод destroy моделі Group. Вхідні дані – ідентифікатор змагання. Вихідні дані – сторінка зі списком залишених змагань.

Запит «GET /memberships» використовує метод index контролера MembershipsController, для відображення – представлення memberships/index, для бази даних – методи моделі Membership: all щоб отримати список запрошень, where для пошуку по статусу, змаганню чи користувачу, order для сортування по ідентифікатору, page для пагінації. Вхідні дані – ідентифікатор змагання та користувача. Вихідні дані – сторінка зі списком запрошень в очікуванні.

Запит «GET /groups/:group_id/memberships/new» використовує метод new контролера MembershipsController, для відображення – представлення

memberships/new. Вхідні дані – ідентифікатор змагання. Вихідні дані – сторінка з формою створення запрошення для змагання.

Запит «POST /groups/:group_id/memberships» використовує метод create контролера MembershipsController, для відображення – представлення memberships/new чи groups/show в залежності від успіху операції, щоб записати запрошення у базу даних – методи new і save моделі Membership. Вхідні дані – ідентифікатор користувача та змагання, тип запрошення. Вихідні дані – сторінка зі статусом операції та формою створення запрошення чи деталями змагання.

Запит «PATCH /memberships/:id» використовує метод update контролера MembershipsController, для відображення – представлення groups/show, щоб прийняти запрошення у базі даних – метод update моделі Membership. Вхідні дані – ідентифікатор запрошення. Вихідні дані – сторінка зі статусом операції та інформацією про змагання.

Запит «DELETE /memberships/:id» використовує метод destroy контролера MembershipsController, для відображення – представлення memberships/index, щоб відхилити запрошення у базі даних – метод destroy моделі Membership. Вхідні дані – ідентифікатор запрошення. Вихідні дані – сторінка зі статусом операції та інформацією про змагання.

Запит «GET /groups/:group_id/standing» використовує метод show контролера StandingsController, для відображення – представлення standings/show, щоб отримати змагання з бази даних – метод find моделі Group. Вхідні дані – ідентифікатор змагання. Вихідні дані – сторінка з таблицею результатів змагання.

Модель Group відповідає за дані змагання. Атрибути – ідентифікатор, час створення, час редагування, ідентифікатор організатора, назва, тип видимості. Асоціації – запрошення та користувачі, задачі, розв’язки. Також модель має індекс по назві для сортування.

Модель Membership відповідає за дані запрошення. Атрибути – ідентифікатор, час створення, час редагування, ідентифікатор користувача,

ідентифікатор групи, статус. Асоціації – запрошення та користувачі, задачі, розв’язки. Після створення запрошення приймає значення статусу: «запитано» для приватного змагання у разі запиту на приєднання – такий запит повинен прийняти організатор, «запрошено» для приватного змагання у разі запрошення від організатора – потрібно прийняти учаснику, «прийнято» для публічного змагання.

Сервісний клас StandingRedisStore використовується для реалізації алгоритму формування таблиці з результатом змагань. Він шукає у базі даних останній розв’язок користувача, що надіслано під час змагання, та записує його оцінку у кеш, використовуючи сховище даних Redis.

3.3 Розробка модуля для роботи з обліковими записами користувачів

Модуль для обробки сесій, особистого кабінету й комунікації з користувачами містить ресурси users, session, profile, password_recovery, password, що реалізовано у класах UsersController, SessionsController, ProfilesController, PasswordRecoveriesController, PasswordsController, User, AuthToken, Session, PasswordRecovery, UserMailer, PasswordRecoveryMailer та відповідних файлах представлення.

Запит «POST /users» використовує метод create контролера UsersController, для відображення – представлення users/new чи sessions/new в залежності від успіху операції, щоб записати користувача у базу даних – методи new і save моделі User. Вхідні дані – відомості recaptcha, нік, електронна адреса, пароль і підтвердження паролю. Вихідні дані – сторінка зі статусом операції та формою реєстрації чи входу.

Запит «GET /session/new» використовує метод new контролера SessionsController, для відображення – представлення sessions/new. Вхідні дані відсутні. Вихідні дані – сторінка з формою входу.

Запит «POST /session» використовує метод create контролера SessionsController, для відображення – представлення sessions/new чи profiles/show в залежності від успіху операції, щоб записати авторизацію у базу

даних – методи new і save сервісу Session. Вхідні дані – відомості recaptcha, електронна адреса, пароль. Вихідні дані – сторінка зі статусом операції та формою входу чи даними облікового запису користувача.

Запит «DELETE /session» використовує метод destroy контролера SessionsController, для відображення – представлення home/show, щоб відхилити видалити токен у базі даних – метод destroy сервісу Session. Вхідні дані – ідентифікатор токену. Вихідні дані – стартова сторінка ресурсу.

Запит «GET /profile» використовує метод show контролера ProfilesController, для відображення – представлення profiles/show, щоб отримати користувача з бази даних – метод find моделі User. Вхідні дані відсутні. Вихідні дані – сторінка з даними облікового запису користувача.

Запит «PATCH /profile» використовує метод update контролера ProfilesController, для відображення – представлення profiles/show, щоб оновити користувача у базі даних – метод update моделі User. Вхідні дані – нік, ім’я, новий пароль і підтвердження, навички, місто, університет. Вихідні дані – сторінка зі статусом операції та даними облікового запису користувача.

Запит «GET /password_recovery/new» використовує метод new контролера PasswordRecoveriesController, для відображення – представлення password_recoveries/new. Вхідні дані відсутні. Вихідні дані – сторінка з формою запиту на відновлення пароля.

Запит «POST /password_recovery» використовує метод create контролера PasswordRecoveriesController, для відображення – представлення password_recoveries/create, щоб записати відновлення у базу даних – методи new і save сервісу PasswordRecovery. Вхідні дані – відомості recaptcha, електронна адреса. Вихідні дані – сторінка зі статусом операції.

Запит «GET /password/edit» використовує метод edit контролера PasswordsController, для відображення – представлення passwords/edit, щоб отримати користувача з бази даних – метод find моделі User. Вхідні дані – токен відновлення пароля. Вихідні дані – сторінка з формою відновлення пароля.

Запит «PATCH /password» використовує метод update контролера PasswordsController, для відображення – представлення passwords/edit чи sessions/new в залежності від успіху операції, щоб оновити пароль у базі даних – метод update моделі User. Вхідні дані – токен відновлення, новий пароль та підтвердження. Вихідні дані – сторінка зі статусом операції та формою відновлення паролю чи входу.

Модель User відповідає за дані користувача. Атрибути – ідентифікатор, час створення, час редагування, електронна адреса, хеш паролю, нік, ім’я, навички, місто, університет, ролі, токен відновлення паролю. Асоціації – файл зображення, задачі, токени авторизації, розв’язки, змагання. Також модель має індекс по електронній адресі, імені, ніку для пошуку. Після створення нового користувача створюється йому надсилається електронний лист через UserMailer.

Модель AuthToken відповідає за дані авторизації. Атрибути – ідентифікатор, час створення, час редагування, ідентифікатор користувача.

Сервісний клас Session відповідає за процес авторизації. Виконується перевірка введених користувачем даних, генерація токену авторизації та його запис у сховище сесії.

Сервісний клас PasswordRecovery відповідає за процес запиту відновлення пароля. Виконується пошук користувача по електронній адресі, генерація нового токену відновлення пароля та надсилається електронний лист користувачеві про здійснений запит через PasswordRecoveryMailer.

Робота з електронною поштою здійснюється через спеціальні класи UserMailer для повідомлення про реєстрацію, та PasswordRecoveryMailer для повідомлення про відновлення пароля. Класи використовують представлення user_mailer/email та password_recovery_mailer/email відповідно. Також налаштовано сервіс Amazon Simple Email Service, що виконує доставляння листів.

3.4 Розробка модуля комунікації з серверною частиною

Модуль для комунікації з серверною частиною містить ресурси api/submissions, api/problems, api/results, api/logs, api/compilers, api/workers, що реалізовано у класах Api::SubmissionsController, Api::ProblemsController, Api::ResultsController, Api::LogsController, Api::CompilersController, Api::WorkersController, Result, Log, Worker, Release та відповідних файлах представлення.

Модуль комунікації реалізовує схему взаємодії з серверною частиною. Авторизація програми серверної частини здійснюється через передачу розміщеного на сервері унікального токену. Дані передаються у машино-читаємому форматі JSON, тому ресурси не використовують представлення. Для підтримки одночасної роботи кількох серверів використовуються механізми транзакції та блокування на рівні бази даних.

Запит «GET /api/submissions» використовує метод index контролера Api::SubmissionsController, для бази даних – методи моделі Submission: all щоб отримати список розв’язків, where для фільтрації по статусу, includes щоб одночасно отримати задачу, компілятор і вихідний код. Вхідні дані відсутні. Вихідні дані – список розв’язків, що очікують на перевірку.

Запит «GET /api/submissions/:submission_id/take» використовує метод take контролера Api::SubmissionsController, щоб оновити статус розв’язку у базі даних – метод take! моделі Submission. Вхідні дані – ідентифікатор розв’язку. Вихідні дані – статус операції.

Запит «GET /api/submissions/:submission_id/release» використовує метод release контролера Api::SubmissionsController, щоб оновити розв’язок у базі даних – методи new і save сервісу Release. Вхідні дані – ідентифікатор розв’язку. Вихідні дані – статус операції.

Запит «GET /api/submissions/:submission_id/fail» використовує метод fail контролера Api::SubmissionsController, щоб оновити статус розв’язку у базі даних – метод fail! моделі Submission. Вхідні дані – ідентифікатор розв’язку. Вихідні дані – статус операції.

Запит «GET /api/problems/:id» використовує метод show контролера Api::ProblemsController, щоб отримати задачу з бази даних – метод find моделі Problem. Вхідні дані – ідентифікатор задачі. Вихідні дані – тести і чекер задачі.

Запит «POST /api/results» використовує метод create контролера Api:: ResultsController, щоб записати результат у базу даних – методи new і save моделі Result. Вхідні дані – ідентифікатор розв’язку, ідентифікатор тесту, статус, деталі, затрачені пам’ять та час. Вихідні дані – статус операції та помилки.

Запит «POST /api/submissions/:submission_id/logs» використовує метод create контролера Api:: LogsController, щоб записати деталі у базу даних – методи new і save моделі Log. Вхідні дані – ідентифікатор розв’язку, дані та їх тип. Вихідні дані – статус операції та інформація про помилки.

Запит «GET /api/compilers» використовує метод index контролера Api:: CompilersController, щоб отримати список компіляторів з бази даних – метод all моделі Compiler. Вхідні дані відсутні. Вихідні дані – список компіляторів.

Запит «POST /api/workers» використовує метод create контролера Api::WorkersController, щоб записати сервер у базу даних – методи new і save моделі Worker. Вхідні дані – час останньої активності, тип і версія інтерфейсу, назва, статус, статуси задач, IP-адреса. Вихідні дані – статус операції та інформація про помилки чи створений сервер.

Запит «PATCH /api/workers/:id» використовує метод update контролера Api::WorkersController, щоб оновити сервер у базі даних – метод update моделі Worker. Вхідні дані – ідентифікатор сервера, час останньої активності, тип і версія інтерфейсу, назва, статус, статуси задач, IP-адреса. Вихідні дані – статус операції та інформація про помилки чи створений сервер.

Модель Result відповідає за результат перевірки розв’язку на одному тесті. Атрибути – ідентифікатор, ідентифікатор розв’язку, ідентифікатор тесту, час створення, час редагування, статус, деталі, затрачені пам’ять та час.

Модель Log відповідає за деталі тестування. Атрибути – ідентифікатор, ідентифікатор розв’язку, час створення, час редагування, дані, тип.

Модель Worker відповідає за сервер тестування. Атрибути – час останньої активності, тип і версія інтерфейсу, назва, статус, статуси задач, IP-адреса.

Сервісний клас Release відповідає за процес завершення оцінювання розв’язку. Він виконує оновлення статусу розв’язку та підрахунок кінцевої оцінки. Після цього оновлюється інформація у StandingRedisStore для підтримки таблиці результатів актуальною.

3.5 Висновки

У третьому розділі було проведено варіантний аналіз різноманітних технологій та засобів для розробки веб-ресурсів, такі як мови програмування, операційні системи й бази даних. Під час аналізу визначено переваги і недоліки розглянутих засобів, обґрунтовано вибір тих, що найкраще підходять для реалізації магістерської кваліфікаційної роботи – Linux в якості операційної системи, Ruby в якості мови програмування, реляційну базу даних PostgreSQL та систему для кешування Redis, бібліотеку компонентів інтерфейсу Bootstrap. Ці програмні засоби забезпечують простоту програмування веб-ресурсу й ефективну роботу з інформаційним забезпеченням системи.

Також були розроблені модулі для роботи з розв’язками, змаганнями, задачами, обліковими записами користувачів та комунікації з серверною частиною. Описано усі запити, на які повинні відповідати модулі, з допомогою методів контролерів. Розглянуто потрібні для роботи системи моделі, що зберігають і перевіряють дані, доступні їм атрибути та індекси. Для певних ресурсів реалізовано сервісні класи, так як вони не представлені моделями, а лише містять певну бізнес-логіку. Щоб надсилати повідомлення на електронні адреси користувачів розроблено класи для кожного повідомлення й налаштовано SES.

4 ТЕСТУВАННЯ ПРОГРАМИ

4.1 Опис методів тестування

Тестування програмного забезпечення – процес аналізу програмного засобу та супутньої документації з метою виявлення дефектів і підвищення якості продукту. Основними характеристиками сучасного тестування є використання гнучких методологій, глибока інтеграція з процесом розробки, широке використання автоматизації, набір різних технологій та інструментів [37]. Техніка тестування включає як процес пошуку помилок чи інших дефектів, так і випробування програмних складових з метою оцінки. Можна виділити кілька методів тестування, що відрізняються предметом дослідження та відомими даними про систему.

Метод статичного тестування передбачає перевірку документації й результатів розробки програми, наприклад технічного завдання, специфікації, вихідного коду програми. Проводиться аналіз дотримання стандартів програмування, відповідності заданим критеріям і вимогам.

Тестування методом «білої скриньки» полягає у дослідженні внутрішньої поведінки програми, коректність побудови окремих модулів чи їх складових, правильність взаємодії модулів між собою. Для такого тестування необхідно знати та розуміти внутрішню структуру програми. Тестування методом «білої скриньки» має деякі недоліки, наприклад повне тестування окремих модулів програми може не гарантувати відповідність усім вимогам, а також складно виявити деякі типи помилок в ізольованому середовищі.

Інший підхід забезпечується методом тестування «чорної скриньки», коли той, хто проводить тестування, не знає нічого про внутрішню структуру програми і все тестування проходить через інтерфейс користувача. В такому випадку досліджується робота кожної функції сервісу разом, в середовищі, що максимально наближено до користувача. Такий метод дозволяє забезпечити повну перевірку всіх функціональних вимог до програми, визначити помилки, що не можливо знайти іншими методами. Однак, неможливо провести

вичерпне тестування всіх можливих сценаріїв роботи, оскільки навіть в невеликих програмах їх кількість може бути дуже великою. Тому, таким методом зазвичай перевіряють критичну функціональність продукту за прогнозованими сценаріями використання.

Для пришвидшення тестування варто автоматизувати його там, де можливо. Автоматизоване тестування – це набір підходів й технологій, які дозволяють виключити людину з виконання деяких задач в процесі тестування. Автоматизація має такі переваги, як набагато швидше проведення тестування, виключення впливу людського фактору на результат, можливість обробки значно більшої кількості інформації [37]. Також такий підхід дозволяє використати концепції керованої тестами (TDD) чи поведінкою (BDD) розробки. Це схожі підходи до розробки програмного засобу, які починаються з написання тестів чи опису поведінки програми, після чого відбувається написання самої програми до того моменту, поки тести не проходитимуть успішно. RSpec – фреймворк для автоматичного тестування на Rails з підтримкою BDD, що допомагає писати ефективні тести для веб-додатків. RSpec пропонує особливі концепти, з використанням яких можна описувати специфікацію та поведінку певної частини коду в близькому до природної мови форматі, забезпечуючи при цьому її автоматичне тестування [38].

4.2 Тестування роботи клієнтської частини веб-ресурсу

Оскільки кожен з методів тестування має свої переваги й недоліки, необхідно використати їх комбінацію для якісного тестування програмного продукту. Для виконання тестування методом «білої скриньки» використаємо можливості фреймворку RSpec. Для кожного з модулів та його складових частин, таких як контролери, моделі, сервісні класи і декоратори, необхідно написати специфікацію засобами RSpec. Вихідний код такої специфікації наведено в додатку В. Після цього необхідно запустити відповідну програму для виконання перевірки поведінки вихідного коду розробленого сервісу. З результату перевірки (рисунок 4.1) видно, що було розроблено 800 тестових

випадків, кожен з яких проходить успішно. При цьому виконання усіх тестів займає всього 3,49 секунди.

```
just806me@archer CodeLabs master
$ bundle exec rspec

Randomized with seed 38571
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
....................................................................................................
..........................................................

Finished in 3.49 seconds (files took 1.68 seconds to load)
800 examples, 0 failures

Randomized with seed 38571
```

Рисунок 4.1 – Результат тестування фреймворком RSpec

Також виконаємо тестування методом «чорної скриньки», а саме перевіримо роботу працюючого веб-ресурсу через його інтерфейс користувача. Для цього потрібно відкрити головну сторінку, заповнити відповідні поля форми та зареєструватись на сайті для подальшої роботи (рисунок 4.2).

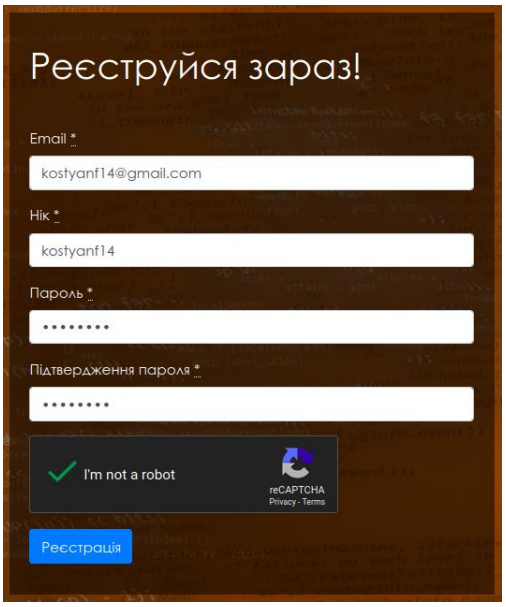


Рисунок 4.2 – Реєстрація на сайті

Після успішної реєстрації користувач потрапляє до сторінки особистого кабінету (рисунок 4.3). На цій сторінці він має можливість заповнити

персональну інформацію про себе, змінити пароль, переглянути свої розв’язки та запрошення до групи.

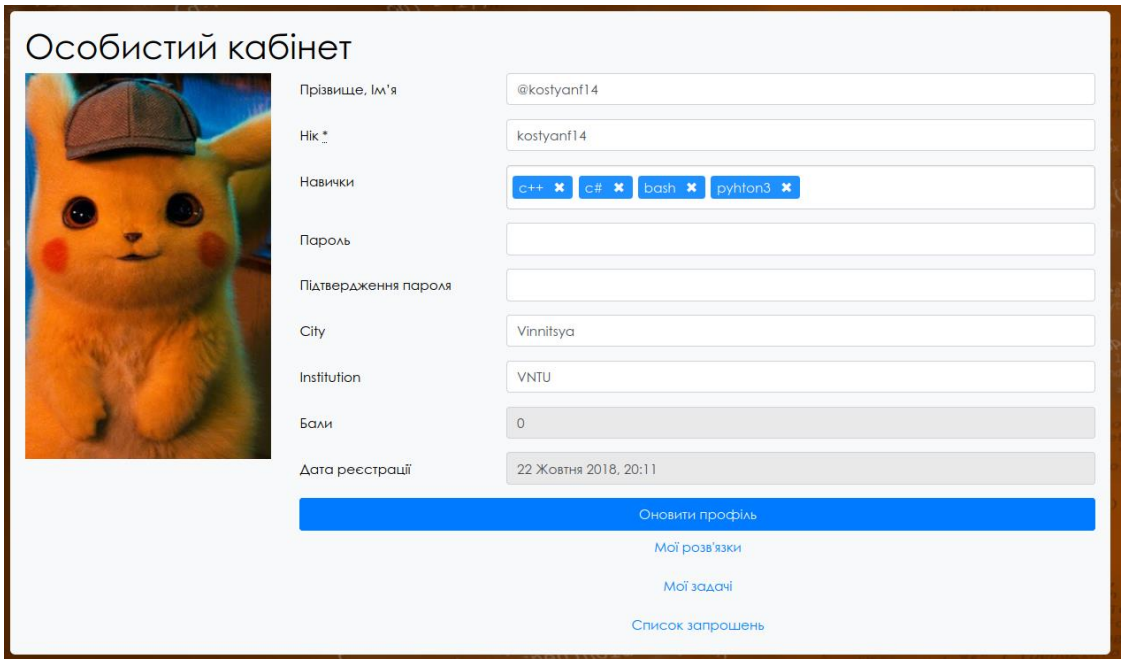


Рисунок 4.3 – Особистий кабінет

Після заповнення особистого кабінету, з допомогою елементу навігації, перейдемо до сторінки задач (рисунок 4.4), де повинен відображатись список усіх доступних для рішення задач.

Рисунок 4.4 – Сторінка зі списком задач

Обравши потрібну задачу, перейдемо до її опису за допомогою відповідного посилання. На цій сторінці описана умова задачі, технічні вимоги до її рішення, приклади вхідних та вихідних даних, а також її класифікація (рисунок 4.5).

Рисунок 4.5 – Сторінка умови обраної задачі

Для відправки вихідного коду розв’язку для задачі, необхідно перейти до відповідного елементу відправлення розв’язку та заповнити усі поля – компілятор MinGW g++ 8.4.2, код програми solution.cpp (рисунок 4.6).

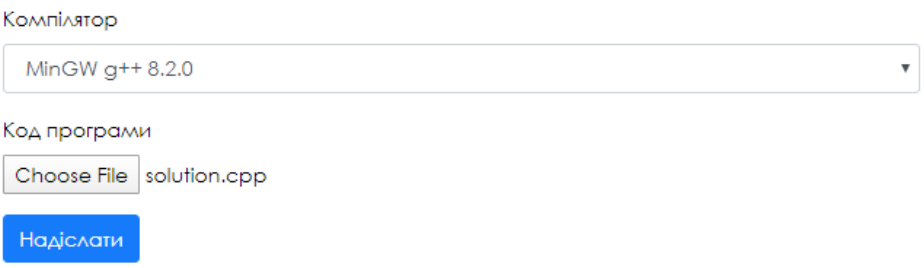


Рисунок 4.6 – Вікно відправлення розв’язку з заповненою інформацією

Далі необхідно натиснути кнопку «Надіслати», після чого має відкритись вікно перегляду результату розв’язку з оцінкою 22 бали із 22 і правильно пройденими усіма тестами (рисунок 4.3).

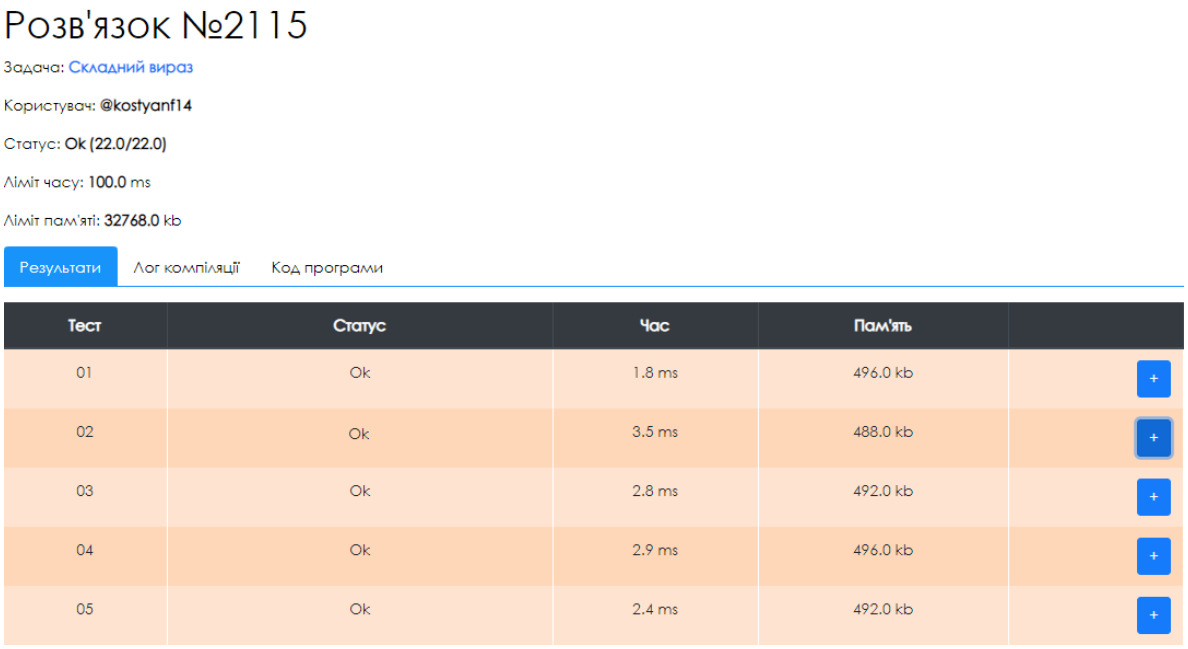


Рисунок 4.3 – Перегляд результату надісланого розв’язку

Також необхідно перевірити роботу таблиці результатів змагання. Для цього потрібно створити змагання з обраною задачею і додати туди кілька користувачів. Після цього відкрити вікно перегляду таблиці результатів цього змагання, де в користувача @kostyanf14 повинно бути здана задача на таку саму кількість балів, що отримані під час перевірки (рисунок 4.4).

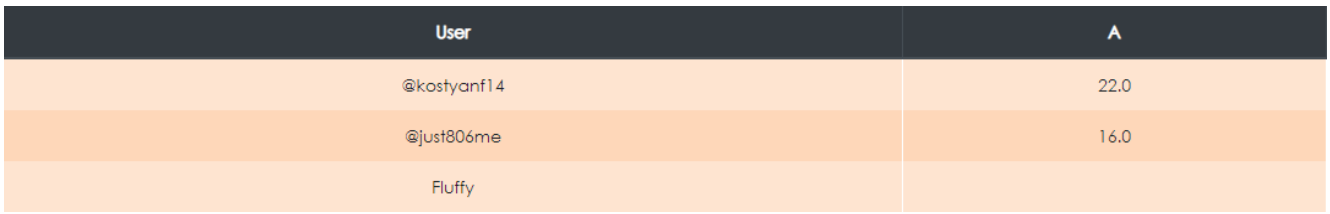


Рисунок 4.4 – Перегляд таблиці результатів змагання

Усі перевірені сторінки відображають правильну інформацію, що вказує на коректність роботи необхідного функціоналу розробленого веб-ресурсу для

розміщення та автоматизованої перевірки олімпіадних задач з програмування. А саме, були протестовані: реєстрація користувачів, робота з особистим кабінетом, перегляд доступних задач у вигляді списку та з детальним описом умови, відправка рішень на перевірку до клієнтської частини, проведення змагань за допомогою груп й формування таблиці з результатами таких змагань.

4.3 Висновки

У четвертому розділі було описано різні методи тестування та інструменти для його проведення в автоматичному режимі, проаналізовано їх переваги й недоліки, сфери використання. Також виконано тестування розробленої клієнтської частини веб-ресурсу методами «чорної скриньки» та «білої скриньки». Така комбінація дозволить провести ефективну перевірку усіх компонентів в автоматичному режимі, а також впевнитися в правильній роботі системи в цілому та її основних компонентів за рахунок більш детального аналізу в наближених до реальних умовах експлуатації.

При проведенні тестування отримана повна відповідність вхідних даних і вихідних результатів. Помилок при роботі програми не виявлено і це підтверджує нормальний режим роботи. У результаті доведено повну працездатність веб-ресурсу та його відповідність поставленому технічному завданню.

5 ЕКОНОМІЧНА ЧАСТИНА

5.1 Оцінювання комерційного потенціалу розробки

Метою проведення комерційного та технологічного аудиту є оцінювання комерційного потенціалу розробка методу і програмних засобів системи тренування і оцінювання робіт зі спортивного програмування.

Для проведення технологічного аудиту було залучено 3-х незалежних експертів Вінницького національного технічного університету: Войтко Вікторія Володимирівна (к.т.н., доц. кафедри ПЗ ВНТУ), Черноволик Галина Олександрівна (к.т.н., доц. кафедри ПЗ ВНТУ), Бурбело Сергій Михайлович (к.т.н., ст.в. кафедри ПЗ ВНТУ). Для проведення технологічного аудиту було використано таблицю 5.1, в якій за п’ятибальною шкалою використовуючи 12 критеріїв здійснено оцінку комерційного потенціалу розробки.

Таблиця 5.1 – Рекомендовані критерії оцінювання комерційного потенціалу розробки та їх можлива бальна оцінка

Критерії оцінювання та бали (за 5-ти бальною шкалою)					
Кри-терій	0	1	2	3	4
Технічна здійсненність концепції:					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними висновками	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено роботоздатність продукту в реальних умовах
Ринкові переваги (недоліки):					
2	Багато аналогів на малому ринку	Мало аналогів на малому ринку	Кілька аналогів на великому ринку	Один аналог на великому ринку	Продукт не має аналогів на великому ринку
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам аналогів	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно гірші, ніж в аналогів	Технічні та споживчі властивості продукту трохи гірші, ніж в аналогів	Технічні та споживчі властивості продукту на рівні аналогів	Технічні та споживчі властивості продукту трохи кращі, ніж в аналогів	Технічні та споживчі властивості продукту значно кращі, ніж в аналогів

Продовження таблиці 5.1

5	Експлуатаційні витрати значно вищі, ніж в аналогів	Експлуатаційні витрати дещо вищі, ніж в аналогів	Експлуатаційні витрати на рівні експлуатаційних витрат аналогів	Експлуатаційні витрати трохи нижчі, ніж в аналогів	Експлуатаційні витрати значно нижчі, ніж в аналогів
Ринкові перспективи					
6	Ринок малий і не має позитивної динаміки	Ринок малий, але має позитивну динаміку	Середній ринок з позитивною динамікою	Великий стабільний ринок	Великий ринок з позитивною динамікою
7	Активна конкуренція великих компаній на ринку	Активна конкуренція	Помірна конкуренція	Незначна конкуренція	Конкурентів немає
Практична здійсненність					
8	Відсутні фахівці як з технічної, так і з комерційної реалізації ідеї	Необхідно наймати фахівців або витрачати значні кошти та час на навчання наявних фахівців	Необхідне незначне навчання фахівців та збільшення їх штату	Необхідне незначне навчання фахівців	Є фахівці з питань як з технічної, так і з комерційної реалізації ідеї
9	Потрібні значні фінансові ресурси, які відсутні. Джерела фінансування ідеї відсутні	Потрібні незначні фінансові ресурси. Джерела фінансування відсутні	Потрібні значні фінансові ресурси. Джерела фінансування є	Потрібні незначні фінансові ресурси. Джерела фінансування є	Не потребує додаткового фінансування
10	Необхідна розробка нових матеріалів	Потрібні матеріали, що використовуються у військово-промисловому комплексі	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуються у виробництві
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій більше 10-ти років	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х до 5-ти років	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше 3-х років
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на виробництво та реалізацію продукту	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що вимагає значних коштів та часу	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів та часу	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь-які регламентні обмеження на виробництво та реалізацію продукту

Таблиця 5.2 – Рівні комерційного потенціалу розробки

Середньоарифметична сума балів СБ, розрахована на основі висновків експертів	Рівень комерційного потенціалу розробки
0-10	Низький
11-20	Нижче середнього
21-30	Середній
31-40	Вище середнього
41-48	Високий

В таблиці 5.3 наведено результати оцінювання експертами комерційного потенціалу розробки.

Таблиця 5.3 – Результати оцінювання комерційного потенціалу розробки

Критерії	Прізвище, ініціали, посада експерта		
	Войтко В.В	Бурбело С.М.	Черноволик Г.О.
	Бали, виставлені експертами:		
1	4	4	4
2	2	2	2
3	3	2	3
4	3	4	3
5	2	2	2
6	4	3	3
7	3	3	3
8	4	4	4
9	2	2	1
10	3	2	3
11	3	4	4
12	4	3	3
Сума балів	$СБ_1$=37	$СБ_2$=35	$СБ_3$=35
Середньоарифметична сума балів $\overline{СБ}$	$\overline{СБ} = \frac{\sum_1^3 СБ_i}{3} = \frac{37+35+35}{3} = 35.6$		

Розрахована нами на основі висновків експертів середньоарифметична сума балів склала 35.6 бали. Згідно таблиці 5.2 вважається, що рівень комерційного потенціалу проведених досліджень є вище середнього.

Методи і програмні засоби системи тренування і оцінювання робіт зі спортивного програмування, що розробляються в магістерській роботі, будуть цікаві школам та університетам, що приймають участь у міжнародних олімпіадах зі спортивного програмування, для підготовки учнів та студентів і навчанню їх основним принципам вирішення олімпіадних задач. Враховуючи особливості цільової аудиторії та освітню спрямованість проекту, дана розробка є некомерційною.

Порівняємо нову розробку, що розробляється в магістерській роботі, з аналогом, який існує на ринку.

Аналогом нашої розробки обрано інтернет-портал організаційно-методичного забезпечення дистанційних олімпіад з програмування для обдарованої молоді навчальних закладів України – E-Olymp. Основним недоліком аналога є використання однакових обмежень по часу та пам’яті для різних мов програмування й недостатня детальність результатів тестування, що ускладнює навчальний процес.

У розробці дані проблеми вирішуються за рахунок введення вагових коефіцієнтів та модернізації програми «чекера» для збереження детального ходу тестування.

Проведемо оцінку якості і конкурентоспроможності нової розробки порівняно з аналогом. В таблиці 5.4 наведені основні техніко-економічні показники аналога і нової розробки.

Таблиця 5.4 – Основні параметри нової розробки та товару-конкурента

Показник	Варіанти		Відносний показник якості	Коефіцієнт вагомості параметра
	Базовий	Новий		
Точність обмежень для різних мов програмування, %	60	90	1,5	30%
Детальність результатів тестування, бали	5	8	1,6	30%
Середній час відповіді на запит *(менше – краще)*, мс	500	280	1,79	20%
Кількість підтримуваних мов програмування, шт	24	6	0,25	10%
Використання ресурсів комп’ютера, %	20	20	1	10%

Проведемо оцінку якості продукції, яка є найефективнішим засобом забезпечення вимог споживачів та порівняємо її з аналогом.

Визначимо відносні одиничні показники якості по кожному параметру за формулами 5.1 та 5.2 і занесемо їх у відповідну колонку табл. 5.5.

$$q_i = \frac{P_{Hi}}{P_{Бi}} \tag{5.1}$$

$$q_i = \frac{P_{Бi}}{P_{Hi}} \tag{5.2}$$

де P_{Hi}, $P_{Бi}$ – числові значення *i*-го параметру відповідно нового і базового виробів.

$$q_1 = \frac{90}{60} = 1{,}5;$$
$$q_2 = \frac{8}{5} = 1{,}6;$$
$$q_3 = \frac{500}{280} = 1{,}79;$$
$$q_4 = \frac{6}{24} = 0{,}25;$$
$$q_5 = \frac{20}{20} = 1.$$

Відносний рівень якості нової розробки визначаємо за формулою 5.3:

$$К_{я.в.} = \sum_{i=1}^{n} q_i \cdot \alpha_i \tag{5.3}$$

$$К_{я.в.} = 1{,}5 \cdot 0{,}3 + 1{,}6 \cdot 0{,}3 + 1{,}79 \cdot 0{,}2 + 0{,}25 \cdot 0{,}1 + 1 \cdot 0{,}1 = 1{,}413$$

Загальний показник конкурентоспроможності інноваційного рішення (К) з урахуванням вищезазначених груп показників можна визначити за наведеною нижче формулою 5.4.

$$K = \frac{I_{m.n.}}{I_{e.n.}}, \tag{5.4}$$

де $I_{m.n.}$ – індекс технічних параметрів;

$I_{e.n.}$ – індекс економічних параметрів.

Індекс технічних параметрів є відносним рівнем якості інноваційного рішення. Індекс економічних параметрів визначається за формулою (5.5):

$$I_{e.n.} = \frac{\sum_{i=1}^{n} P_{Hei}}{\sum_{i=1}^{n} P_{Бei}}, \quad (5.5)$$

де P_{Hei} , $P_{Бei}$ – економічні параметри (ціна придбання та споживання товару) відповідно нового та базового товарів.

Зважаючи на некомерційний характер розробки і аналогу, приймемо індекс економічних параметрів за 1.

$$\mathrm{I_{е.п.}} = 1;$$

$$\mathrm{К} = \frac{1{,}413}{1} = 1{,}413.$$

Виходячи з розрахунків, можна зробити висновок, що нова розробка буде більш конкурентоспроможною, ніж конкурентний товар.

5.2 Прогнозування витрат на виконання науково-дослідної роботи

Витрати, пов’язані з проведенням науково-дослідної роботи групуються за такими статтями: витрати на оплату праці, витрати на соціальні заходи, матеріали, паливо та енергія для науково-виробничих цілей, витрати на службові відрядження, програмне забезпечення для наукових робіт, інші витрати, накладні витрати.

1. Основна заробітна плата кожного із дослідників $З_0$, якщо вони працюють в наукових установах бюджетної сфери визначається за наведеною нижче формулою 5.6.

$$З_О = \frac{М}{Т_р} \cdot t \text{ (грн)} \quad (5.6)$$

де M – місячний посадовий оклад конкретного розробника (інженера, дослідника, науковця тощо), грн.;

$T_Р$ – число робочих днів в місяці; приблизно $T_Р \approx 21...23$ дні;

t – число робочих днів роботи дослідника.

Для розробки програмних засобів системи тренування і оцінювання робіт зі спортивного програмування необхідно залучити трьох програмістів: розробник C++ з посадовим окладом 120000 грн, розробник Ruby з посадовим окладом 135000 грн, розробник JavaScript з посадовим окладом 130000 грн; одного QA-інженера з посадовим окладом 115000 грн; одного системного інженера з посадовим окладом 135000 грн [39]. Кількість робочих днів у місяці складає 22, кількість робочих днів програмістів Ruby і JavaScript складає 33, програміста C++ – 10, QA-інженера – 15, системного інженера – 5. Зведемо сумарні розрахунки до таблиці 5.6.

Таблиця 5.6 – Заробітна плата спеціалістів розробки програмних засобів

Найменування посади	Місячний посадовий оклад, грн.	Оплата за робочий день, грн.	Число днів роботи	Витрати на заробітну плату грн.
Керівник	12000	545,5	5	2727
Розробник C++	120000	5454,6	10	54546
Розробник Ruby	135000	6136,4	33	202501,2
Розробник JavaScript	130000	5909,0	33	194997
QA-інженер	115000	5227,3	15	172500,9
Системний інженер	135000	6136,4	5	30682
Всього				657954,1

2. Розрахунок додаткової заробітної плати робітників

Додаткова заробітна плата $З_д$ всіх розробників та робітників, які приймали устають в розробці нового технічного рішення розраховується як 10-12% від основної заробітної плати робітників за формулою 5.7.

На даному підприємстві додаткова заробітна плата начисляється в розмірі 10% від основної заробітної плати.

$$З_д = \frac{(З_о + З_р) * Н_{дод}}{100\%} \quad (5.7)$$

$$З_д = 0{,}10 \cdot 657954{,}1 = 65795{,}41 \text{ (грн)}$$

3. Нарахування на заробітну плату $Н_{ЗП}$ дослідників та робітників, які брали участь у виконанні даного етапу роботи, розраховуються за формулою (5.8):

$$Н_{ЗП} = \frac{(З_о + З_д) * \beta}{100} \quad (5.8)$$

де $З_о$ – основна заробітна плата розробників, грн.;

$З_д$ – додаткова заробітна плата всіх розробників та робітників, грн.;

β – ставка єдиного внеску на загальнообов'язкове державне соціальне страхування, %

Дана діяльність відноситься до бюджетної сфери, тому ставка єдиного внеску на загальнообов'язкове державне соціальне страхування буде складати 22%, тоді:

$$Н_{ЗП} = \frac{(657954{,}1 + 65795{,}41) \cdot 22}{100} = 159224{,}9 \text{ (грн)}$$

4. Витрати на матеріали М та комплектуючі К, що були використані під час виконання даного етапу роботи, розраховуються по кожному виду матеріалів за формулою 5.9:

$$M = \sum_{1}^{n} Н_i \cdot Ц_i \cdot К_i - \sum_{1}^{n} В_i \cdot Ц_в \quad (5.9)$$

де $Н_i$ – витрати матеріалу *і*-го найменування, кг;

$Ц_i$ – вартість матеріалу *і*-го найменування, грн./кг.;

$К_i$ – коефіцієнт транспортних витрат, $К_i$ = (1,1…1,15);

$В_i$ – маса відходів матеріалу *і*-го найменування, кг;

$Ц_в$ – ціна відходів матеріалу *і*-го найменування, грн/кг;

n – кількість видів матеріалів.

Таблиця 5.7 – Матеріали, що використані на розробку

Найменування матеріалу	Ціна за одиницю, грн.	Витрачено	Вартість витраченого матеріалу, грн.
Папір	140	1	140
Ручка	20	1	20
CD-диск	12	1	12
Флешка	155	1	155
Всього			327
З врахуванням коефіцієнта транспортування			359,7

5. Програмне забезпечення для наукової роботи включає витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення необхідного для проведення дослідження. Для нової розробки використовувались безкоштовні програмні засоби.

6. Амортизація обладнання, комп’ютерів та приміщень, які використовувались під час виконання даного етапу роботи

Дані відрахування розраховують по кожному виду обладнання, приміщенням тощо за формулою 5.10.

$$A = \frac{Ц \cdot T}{Ткор \cdot 12} \quad [грн], \tag{5.10}$$

де Ц – балансова вартість даного виду обладнання (приміщень), грн.;

$Т_{кор}$ – час користування;

Т – термін використання обладнання (приміщень), цілі місяці.

Згідно пункту 137.3.3 Податкового кодексу амортизація нараховується на основні засоби вартістю понад 2500 грн. В нашому випадку для написання магістерської роботи використовувалися п’ять персональних комп’ютерів кожен вартістю 45000 грн.

$$А = \frac{45000 \cdot 5}{2 \cdot 12} = 9375 \text{ (грн)}$$

7. До статті «Паливо та енергія для науково-виробничих цілей» відносяться витрати на всі види палива й енергії, що використовуються з технологічною метою на проведення досліджень, та розраховуються за формулою 5.11.

$$В_е = \sum_{i=1}^{n} \frac{W_{yt} \cdot t_i \cdot Ц_е \cdot К_{впі}}{\eta_і} \tag{5.11}$$

де W_{yt} – встановлена потужність обладнання, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_е$ – вартість 1 кВт-години електроенергії, грн;

$К_{впі}$ – коефіцієнт, що враховує використання потужності, $К_{впі} < 1$;

$\eta_і$ – коефіцієнт корисної дії обладнання, $\eta_і < 1$.

Для написання магістерської роботи використовувалися чотири персональних комп'ютери для яких розрахуємо витрати на електроенергію.

$$В_е = \frac{0,5 \cdot 250 \cdot 4,1 \cdot 0,5}{0,8} + \frac{0,5 \cdot 250 \cdot 4,1 \cdot 0,5}{0,8} + \frac{0,5 \cdot 250 \cdot 4,1 \cdot 0,5}{0,8} + \frac{0,5 \cdot 250 \cdot 4,1 \cdot 0,5}{0,8} + \frac{0,5 \cdot 250 \cdot 4,1 \cdot 0,5}{0,8} = 1601,56$$

Витрати на службові відрядження, витрати на роботи, які виконують сторонні підприємства, установи, організації та інші витрати в нашому досліджені не враховуються оскільки їх не було.

Накладні (загальновиробничі) витрати $В_{нзв}$ охоплюють: витрати на управління організацією, оплата службових відряджень, витрати на утримання, ремонт та експлуатацію основних засобів, витрати на опалення, освітлення, водопостачання, охорону праці тощо. Накладні (загальновиробничі) витрати $В_{нзв}$ можна прийняти як (100…150)% від суми основної заробітної плати розробників та робітників, які виконували дану МКНР, що розраховуються за формулою 5.12:

$$В_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}, \quad (5.12)$$

де $Н_{нзв}$ – норма нарахування за статтею «Інші витрати».

$$В_{нзв} = 657954,1 \cdot \frac{100}{100\%} = 657954,1 \text{ (грн)}$$

Сума всіх попередніх статей витрат дає витрати, які безпосередньо стосуються даного розділу МКНР:

$$В = 657954,1 + 65795,41 + 159224,9 + 359,7 + 9375 + 1601,56 + 657954,1 = 1552264,77 \text{ (грн)}$$

Прогнозування загальних втрат ЗВ на виконання та впровадження результатів виконаної МКНР здійснюється за формулою 5.13:

$$ЗВ = \frac{В}{\eta}, \quad (5.13)$$

де η – коефіцієнт, який характеризує стадію виконання даної НДР.

Оскільки, робота знаходиться на стадії впровадження, то коефіцієнт $\eta = 0{,}9$. Звідси:

$$ЗВ = \frac{1552264{,}77}{0{,}9} = 1724738{,}63 \text{ (грн.)}$$

5.3 Оцінювання важливості та наукової значимості науково-дослідної роботи

Застосовані методи і розроблені програмні засоби системи тренування і оцінювання робіт зі спортивного програмування можуть бути цікаві профільним школам та закладам вищої освіти, що приймають участь у міжнародних олімпіадах зі спортивного програмування, а також зацікавленим студентам. Враховуючи особливості цільової аудиторії та освітню спрямованість проекту, магістерська кваліфікаційна робота є науково-дослідного й некомерційного характеру. Тому, розрахуємо комплексний показник $К_р$ рівня важливості та наукової значимості науково-дослідної роботи за формулою 5.14:

$$К_р = \frac{I^n \cdot T_c \cdot R}{B \cdot t} \quad (5.14)$$

де I – коефіцієнт важливості роботи, I = 2...5;

n – коефіцієнт використання результатів роботи; n = 0, коли результати роботи не будуть використовуватись; n = 1, коли результати роботи будуть

використовуватись частково; n = 2, коли результати роботи будуть використовуватись в дослідно-конструкторських розробках; n = 3 , коли результати роботи можуть використовуватись навіть без проведення дослідно-конструкторських розробок;

T_c – коефіцієнт складності роботи, T_c = 1...3;

R – коефіцієнт результативності роботи; якщо результати роботи плануються вище відомих, то R = 4 ; якщо результати роботи відповідають відомому рівню, то R = 3 ; якщо нижче відомих результатів, то R = 1;

B – вартість науково-дослідної роботи, тис. грн;

t – час проведення дослідження, років.

Визначимо показники I, n, T_c, R, B, t на основі нормативів. У результаті підрахунків отримаємо наступне значення показника рівня науково-дослідної роботи:

$$К_р = \frac{5^2 \cdot 3 \cdot 4}{1552{,}26477 \cdot \frac{2}{12}} = 1{,}16$$

Оскільки розрахований показник рівня науково-дослідної роботи $К_р > 1$, то науково-дослідну роботу можна вважати ефективною з високим науковим, технічним і економічним рівнями.

5.4 Висновки до економічного розділу

Було проведено оцінку комерційного потенціалу методів та програмних засобів системи тренування і оцінювання робіт зі спортивного програмування, який є на рівні вище середнього. При порівнянні нової розробки з аналогом виявлено, що вона є якіснішою і більш конкурентоспроможною в порівнянні з аналогом, а також є кращою по технічним і економічним показникам.

Прогнозування витрат на виконання науково-дослідної роботи по кожній з статей витрат складе 1552264,77 грн. Загальна ж величина витрат на виконання та впровадження результатів даної НДР буде складати 1724738,63 грн.

Було розраховано показник рівня важливості та наукової значимості науково-дослідної роботи $К_р$. За його значенням науково-дослідну роботу можна вважати ефективною з високим науковим, технічним і економічним рівнями.

Мова про те, що вкладені у проект інвестиції окупляться, вестися не може, тому що розробка (проект) є освітньою, а результати її впровадження не передбачають отримання прибутку. Розробка має на меті надання вільного доступу до знань та інструментів для навчання, тому її фінансування може бути цікавим для профільних ЗВО та шкіл.

ВИСНОВКИ

У магістерської кваліфікаційної роботі було розроблено клієнтську частину системи тренування і оцінювання робіт зі спортивного програмування. Такий веб-ресурс може забезпечити як автоматичне проведення олімпіади через мережу Інтернет, так і дозволяє тренуватись до майбутніх олімпіад, відправляючи на перевірку рішення до доступних задач.

Було проаналізовано стан питання оцінювання робіт на сучасних олімпіадах. Розглянуто основні аналоги пропонованої системи, визначено їх особливості та недоліки, порівняно з власним програмним продуктом, за рахунок чого визначено актуальність і задачі розробки.

Після цього спроектовано структуру роботи програми, розділено її на модулі, взаємодію яких проілюстровано діаграмами компонентів і послідовності, а також розроблено блок-схеми необхідних алгоритмів.

Для вибору засобів реалізації системи проведено варіантний аналіз, за результатом якого обрано операційну систему Linux, мову програмування Ruby з фреймворком Rails, реляційну базу даних PostgreSQL, систему для кешування Redis і бібліотеку елементів інтерфейсу Bootstrap.

У результаті роботи над магістерською кваліфікаційною роботою було виконано такі задачі: розроблено схеми та алгоритми роботи веб-сайту; розроблено інтерфейс програмного; розроблено підсистему комунікації з серверною частиною; розроблено підсистеми для надсилання та оцінювання розв’язків, проведення змагань, створення задач, адміністрування, обробки сесій, особистого кабінету. Також були запропоновані удосконалення методів для оцінювання й тестування розв’язків задач спортивного програмування, а саме введення вагових коефіцієнтів для різних мов програмування та перенесення тестування на сторону клієнта, використовуючи можливості WebAssembly.

Для тестування було проаналізовано існуючі методи і підходи, виконано перевірку роботи веб-ресурсу за їх допомогою, що довело повну працездатність розробленої системи та її відповідність поставленому технічному завданню.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Олімпіади. – Міністерство освіти і науки України. URL: https://mon.gov.ua/ua/tag/olimpiadi (дата звернення 10.09.2021).
2. Нікітіна Н. С. Олімпіада як науковий захід для студентів. – К.: ЦУЛ, 2015. 175 с.
3. Войтко В.В., Бевз С. В., Бурбело С. М., Кузнєцов Л. Г., Костюк К. А. Розробка веб-ресурсу для розміщення та автоматизованої перевірки олімпіадних. – XLIX Науково-технічна конференція факультету інформаційних технологій та комп'ютерної інженерії, 2020. URL: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2020/paper/view/9620/8008 (дата звернення 10.09.2021)
4. Бурбело С. М., Костюк К. А., Кузнєцов Л. Г. Особливості використання процесорних тактів при оцінюванні часу роботи програм. – Конференція «Молодь у світі сучасних технологій», 2020. URL: http://conference.ho.ua/index.php (дата звернення 10.09.2021)
5. Войтко В. В., Коваленко О. О., Бевз С. В., Бурбело С. М., Кузнєцов Л. Г., Костюк К. А. Застосування wasm у системі тренування і оцінювання робіт зі спортивного програмування. – Всеукраїнська науково-практична Інтернет-конференція «Електронні інформаційні ресурси: створення, використання, доступ», 2021. URL: http://konferencia.voipopp.vn.ua/ (дата звернення 30.11.2021).
6. Спортивне програмування. – Державний університет телекомунікацій, 2017. URL: http://www.dut.edu.ua/ua/news-1-1009-4178-sportivne-programuvannya_kafedra-inzhenerii-programnogo-zabezpechennya (дата звернення 28.09.2021)
7. Розробка Web-додатків для бізнесу. URL: https://tqm.com.ua/ua/rozrobka-veb-dodatkiv-servisiv-web-application-development/rozrobka-web-dodatkiv-dlia-bizniesu (дата звернення 28.09.2021)

8. Монолитная vs Микросервисная архитектура. 2019. URL: https://proglib.io/p/monolitnaya-vs-mikroservisnaya-arhitektura-2019-09-16 (дата звернення 28.09.2021)
9. Тузовський А. Ф. Проектування і розробка web-додатків. – М.: Юрайт, 2019. 218 с.
10. Mirzayanov M. Codeforces: Часто задаваемые вопросы. 2011. URL: https://codeforces.com/help?locale=ru (дата звернення 02.10.2021)
11. E-Olymp – про проект. URL: https://www.e-olymp.com/uk/pages/about (дата звернення 02.10.2021)
12. Система проведення змагань ejudge. URL: https://ejudge.pit.org.ua/files/ru.pdf (дата звернення 02.10.2021)
13. Перша в Україні школа мистецтва розв'язування алгоритмічно складних задач. URL: http://algotester.com (дата звернення 02.10.2021)
14. Flanagan D. JavaScript – The definitive guide. 2011. 1100 с.
15. Кириченко А. В., Хрусталев А. А. HTML5 + CSS3. Основы современного дизайна. – СПб.: Наука и техника, 2018. 352 с.
16. Rourke M. Learn WebAssembly. Birmingham: Packt Publishing, 2018. 328 с.
17. Годун В. М. В., Орленко Н. С., Сендзюк М. А. Інформаційні системи і технології в статистиці. – К.: КНЕУ, 2003. 267 с.
18. Nguyen N. H. Essential Cyber Security Handbook. 2018. 382 с.
19. HTTP authentication. MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Web/HTTP/Authentication (дата звернення 11.10.2021)
20. Parnas D. L. On the Criteria to be Used in decomposing Systems into Modules. – Communications of the ACM, 1972.
21. Nzeemin. WebAssembly: что и как. 2019. URL: https://habr.com/en/post/475778/ (дата звернення 16.10.2021)
22. Walton Z. Easily Port C++ To HTML5/JavaScript With Emscripten. 2012. URL: https://web.archive.org/web/20130730202900/http:/www.webpronews.com/e

asily-port-c-to-html5javascript-with-emscripten-2012-04 (дата звернення 16.10.2021)

23. Compiling C to WebAssembly using clang/LLVM and WASI. URL: https://00f.net/2019/04/07/compiling-to-webassembly-with-llvm-and-clang/ (дата звернення 16.10.2021)
24. Fleming T. Clang In Browser (cib). 2018. URL: https://github.com/tbfleming/cib (дата звернення 16.10.2021)
25. Jangda A., Powers B., Emery B., Arjun G. Not So Fast: Analyzing the Performance of WebAssembly vs. Native Code. – Renton, WA, USA: University of Massachusetts Amherst, 2019. 15 с.
26. What are disadvantages of WebAssembly compared to current HTML/JS?. 2019. URL: https://www.quora.com/What-are-disadvantages-of-WebAssembly-compared-to-current-HTML-JS (дата звернення 18.10.2021)
27. Tulka T. Learning WebAssembly #7: Introducing WASI. 2021. URL: https://blog.ttulka.com/learning-webassembly-7-introducing-wasi (дата звернення 18.10.2021)
28. Tomsho G. Guide to Operating Systems. – Cengage Learning, 2016. 688 с.
29. Operating system market share. 2020. URL: https://netmarketshare.com/operating-system-market-share.aspx (дата звернення 25.10.2021)
30. Хошаба О. М. Операційні системи комп'ютерних мереж. Навчальній посібник. – Вінниця: ВНТУ, 2004. 115 с.
31. Грабер М. Введение в SQL. – М.: Лори, 2010. 227 с.
32. PostgreSQL: About. URL: URL: https://www.postgresql.org/about/ (дата звернення 03.11.2021)
33. Cloud A. Redis vs. Memcached: In-Memory Data Storage Systems. 2018. URL: https://medium.com/@Alibaba_Cloud/redis-vs-memcached-in-memory-data-storage-systems-3395279b0941 (дата звернення 05.11.2021)
34. Язык программирования Ruby. URL: https://www.ruby-lang.org/ru/ (дата звернення 05.11.2021)

35. Wilson J. Node.js 8 the Right Way. – Raleigh: The Pragmatic Programmers, 2018. 334 с.
36. Freeman A. Pro ASP.Net MVC 5. – APress, 2013. 832 с.
37. Куликов С. Тестирование програмного обеспечения. Базовый курс. – Мінськ: Четыре четверти, 2017. 312 с.
38. Marston M., Dees I. Effective Testing with RSpec 3: Build Ruby Apps with Confidence. – Pragmatic Bookshelf, 2017. 356 с.
39. Статистика зарплат програмістів, тестувальників і PM в Україні | DOU. URL: https://jobs.dou.ua/salaries/ (дата звернення 30.11.2021)

ДОДАТКИ

Додаток А – Технічне завдання

Вінницький національний технічний університет

УЗГОДЖУЮ
Директор КЗ «ВТЛ»
_____ О. М. Козяр
"____" _____ 2021 р.

ЗАТВЕРДЖУЮ
Завідувач кафедрою ПЗ
_____ О. Н. Романюк
"____" _____ 2021 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на магістерську кваліфікаційну роботу
зі спеціальності 121 «Інженерія програмного забезпечення»
студенту групи 2ПІ-20м Кузнєцову Леоніду Геннадійовичу

1.1 Найменування та галузь застосування

Розробка методу і програмних засобів системи тренування і оцінювання робіт зі спортивного програмування. Частина 2 Клієнтський додаток.

1.2 Підстава для проведення робіт

Завдання на роботу, яке затверджене на засіданні кафедри програмного забезпечення – протокол № 277 від « 24 » вересня 2021 року.

1.3 Мета та призначення роботи

Мета виконання магістерської кваліфікаційної роботи – підвищення якості систем дистанційного проведення олімпіад і автоматичного оцінювання задач зі спортивного програмування за рахунок розробки та програмної реалізації клієнтської частини спеціалізованого веб-ресурсу, орієнтованого під специфіку обробки розв’язків олімпіадних задач, що дозволяє підвищити об’єктивність і ефективність автоматичного оцінювання результатів.

Об’єктом дослідження є процес дистанційного проведення олімпіад зі спортивного програмування, технології зберігання задач, тестів і надісланих робіт, процес автоматичної перевірки розв’язків.

Предметом дослідження є методи та засоби розробки клієнтської частини веб-ресурсу для розміщення задач, дистанційного проведення олімпіад і автоматизованого оцінювання розв’язків зі спортивного програмування.

Для досягнення поставленої мети в роботі вирішуються такі завдання:

- удосконалення методу оцінювання розв’язків задач спортивного програмування;
- удосконалення методу виконання тестування робіт зі спортивного програмування;
- розробка діаграм взаємодії компонентів та блок-схем основних алгоритмів веб-ресурсу;
- розробка інтерфейсу програмного продукту;
- розробка підсистем для комунікації з серверною частиною, надсилання та оцінювання розв’язків, проведення змагань, створення задач й адміністрування, обробки сесій, особистого кабінету й комунікації з користувачами через електронну пошту;
- проведення тестування програмного продукту декількома різними методами.

1.4 Технічні вимоги

- Операційна система Linux з ядром 5.5 і вище.
- Оперативна пам’ять системи не менше 512 МБ.
- Пам’ять на жорсткому диску не менше 10 ГБ.

1.5 Перелік технічної документації, що пред’являється по закінченню робіт:

- технічне завдання;
- технічне обґрунтування;

- лістинги модулів програмного додатку.

1.6 Стадії і етапи розробки

№ з/п	Назва етапів магістерської кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1	Аналіз проблеми, обґрунтування актуальності розробки системи та постановка задачі	15.09.2021 – 20.09.2021	Вик.
2	Розробка методів та засобів роботи системи	20.09.2021 – 02.10.2021	Вик.
3	Вибір середовища та мови розробки	02.10.2021 – 10.10.2021	Вик.
4	Розробка програмного продукту	10.10.2021 – 25.10.2021	Вик.
5	Тестування роботи системи	25.10.2021 – 05.11.2021	Вик.
6	Економічна частина	05.11.2021 – 20.11.2021	Вик.
7	Оформлення матеріалів до захисту МКР	20.11.2021 – 30.11.2021	Вик.

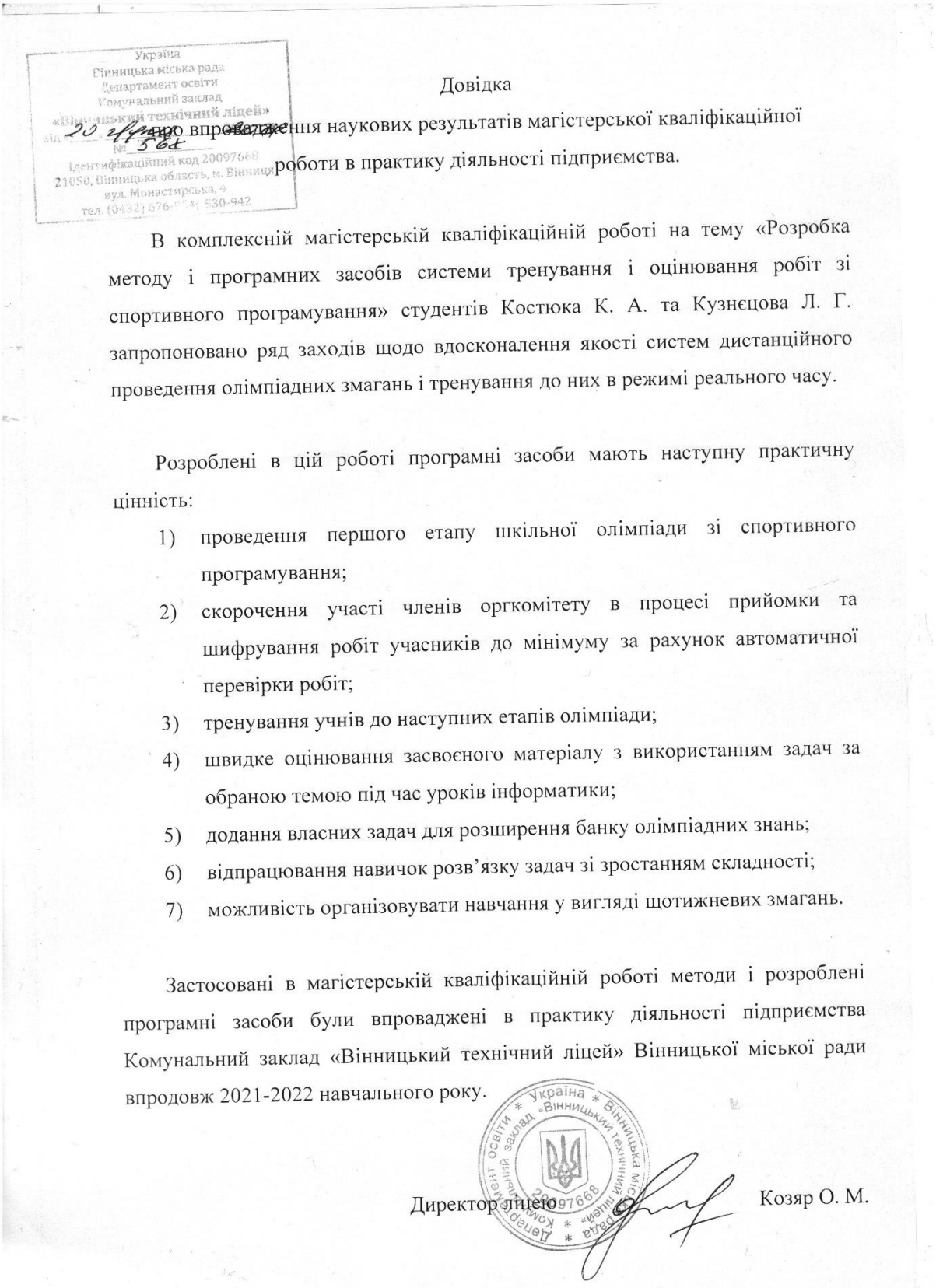
1.7 Порядок контролю і приймання

Порядок контролю і приймання роботи регламентується відповідними документами ВНТУ і державними стандартами.

Завдання отримав _____ Кузнєцов Л. Г.
(підпис) (прізвище та ініціали)

Науковий керівник _____ Бурбело С. М.
(підпис) (прізвище та ініціали)

Додаток Б – Акт впровадження



Україна
Вінницька міська рада
Департамент освіти
Комунальний заклад
«Вінницький технічний ліцей»
від 20.04.2022
№ 368
Ідентифікаційний код 20097668
21050, Вінницька область, м. Вінниця
вул. Монастирська, 9
тел. (0432) 676-[illegible], 530-942

Довідка

про впровадження наукових результатів магістерської кваліфікаційної роботи в практику діяльності підприємства.

В комплексній магістерській кваліфікаційній роботі на тему «Розробка методу і програмних засобів системи тренування і оцінювання робіт зі спортивного програмування» студентів Костюка К. А. та Кузнєцова Л. Г. запропоновано ряд заходів щодо вдосконалення якості систем дистанційного проведення олімпіадних змагань і тренування до них в режимі реального часу.

Розроблені в цій роботі програмні засоби мають наступну практичну цінність:

1) проведення першого етапу шкільної олімпіади зі спортивного програмування;
2) скорочення участі членів оргкомітету в процесі прийомки та шифрування робіт учасників до мінімуму за рахунок автоматичної перевірки робіт;
3) тренування учнів до наступних етапів олімпіади;
4) швидке оцінювання засвоєного матеріалу з використанням задач за обраною темою під час уроків інформатики;
5) додання власних задач для розширення банку олімпіадних знань;
6) відпрацювання навичок розв'язку задач зі зростанням складності;
7) можливість організовувати навчання у вигляді щотижневих змагань.

Застосовані в магістерській кваліфікаційній роботі методи і розроблені програмні засоби були впроваджені в практику діяльності підприємства Комунальний заклад «Вінницький технічний ліцей» Вінницької міської ради впродовж 2021-2022 навчального року.

Директор ліцею Козяр О. М.

Рисунок Б.1 – Акт впровадження

Додаток В – Протокол перевірки роботи на плагіат

ПРОТОКОЛ ПЕРЕВІРКИ НАВЧАЛЬНОЇ (КВАЛІФІКАЦІЙНОЇ) РОБОТИ

Назва роботи: **Розробка методу і програмних засобів системи тренування і оцінювання робіт зі спортивного програмування. Частина 2. Клієнтський додаток.**

Тип роботи: кваліфікаційна робота

Підрозділ: кафедра програмного забезпечення, ФІТКІ, 1ПІ – 20м

Науковий керівник: к.т.н. доц. Бурбело С. М.

Unicheck	
Оригінальність	**91,5 %**
Схожість	8,5 %

Аналіз звіту подібності

■ **Запозичення, виявлені у роботі, оформлені коректно і не містять ознак плагіату.**

 Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її автора. Роботу направити на доопрацювання.

 Виявлені у роботі запозичення є недобросовісними і мають ознаки плагіату та/або в ній містяться навмисні спотворення тексту, що вказують на спроби приховування недобросовісних запозичень.

Заявляю, що ознайомлена з повним звітом подібності, який був згенерований Системою щодо роботи «Розробка методу і програмних засобів системи тренування і оцінювання робіт зі спортивного програмування. Частина 2. Клієнтський додаток».

Автор _____ Кузнєцов Леонід Геннадійович

Опис прийнятого рішення: **допустити до захисту**

Особа, відповідальна за перевірку _____ Черноволик Г. О.
(підпис) (прізвище, ініціали)

Експерт _____ _____
(за потреби) (підпис) (прізвище, ініціали, посада)

Додаток Г – Лістинг коду

Файл Capfile
```
require 'capistrano/setup'
require 'capistrano/deploy'
require 'capistrano/scm/git'
require 'capistrano/rails'
install_plugin Capistrano::SCM::Git
Dir.glob('lib/capistrano/tasks/*.rake').ea
ch { |r| import r }
```
Файл Rakefile
```
require_relative 'config/application'
Rails.application.load_tasks
```
Файл app/models/tag_translation.rb
```
class TagTranslation < ApplicationRecord
  validates :language, :name, presence:
true
  belongs_to :tag
  enum language: I18n.available_locales
end
```
Файл app/models/sharing.rb
```
class Sharing < ApplicationRecord
  validates :problem_id, uniqueness: {
scope: :group_id }
  belongs_to :group
  belongs_to :problem
  delegate :owner, to: :group, prefix:
true
  delegate :user, to: :problem, prefix:
true
end
```
Файл app/models/user.rb
```
class User < ApplicationRecord
  validates :email, presence: true, email:
true, uniqueness: { case_sensitive: false
}
  validates :username, presence: true,
uniqueness: { case_sensitive: false }
  has_one_attached :avatar
  has_one :confirmation_request,
dependent: :destroy
  has_many :problems, dependent: :nullify
  has_many :auth_tokens, dependent:
:destroy
  has_many :submissions, dependent:
:destroy
  has_many :owned_groups, class_name:
'Group', foreign_key: :owner_id,
dependent: :destroy
  has_many :pending_memberships, -> {
where.not state: :accepted }, class_name:
'Membership', dependent: :destroy
  has_many :accepted_memberships, -> {
where state: :accepted }, class_name:
'Membership', dependent: :destroy
  has_many :pending_groups, through:
:pending_memberships, source: :group,
class_name: 'Group'
  has_many :accepted_groups, through:
:accepted_memberships, source: :group,
class_name: 'Group'
  has_many :sharings, through:
:accepted_groups
  has_many :shared_problems, class_name:
'Problem', through: :sharings, source:
:problem
  has_secure_password
  bitmask :roles, as: %i[confirmed
moderator administrator], null: false
  after_create_commit :send_email
  delegate :as_json, to: :decorate
  values_for_roles.each do |value|
    define_method("#{ value }?") { roles?
value }
  end
  private
  def send_email
    UserMailer.email(self).deliver_later
  end
end
```
Файл app/models/example.rb
```
class Example < ApplicationRecord
  validates :input, :answer, presence:
true
  belongs_to :problem
end
```
Файл app/models/log.rb
```
class Log < ApplicationRecord
  self.inheritance_column = ''
  validates :data, presence: true
  belongs_to :submission
  enum type: { source: 0, checker: 1 }
  delegate :user, :problem_user, to:
:submission, prefix: true
end
```
Файл app/models/problems_tag.rb
```
class ProblemsTag < ApplicationRecord
  belongs_to :problem
  belongs_to :tag, counter_cache:
:problems_count
end
```
Файл app/models/compiler.rb
```
class Compiler < ApplicationRecord
  validates :name, :version, :status,
presence: true
  validates :memory_a, :memory_b, :time_a,
:time_b, presence: true, numericality:
true
  enum status: { in_test: 0, reserved: 1,
public: 2 }, _prefix: true
end
```
Файл app/models/application_record.rb
```
class ApplicationRecord <
ActiveRecord::Base
  self.abstract_class = true
end
```
Файл app/models/problem_translation.rb
```
class ProblemTranslation <
ApplicationRecord
  validates :caption, :author, :text,
:technical_text, presence: true
  validates :language, presence: true,
uniqueness: { scope: :problem_id }
  validates :default, inclusion: { in:
[true, false] }
  belongs_to :problem
  enum language: I18n.available_locales
end
```
Файл app/models/worker.rb
```
class Worker < ApplicationRecord
  validates :name, :ips, :status,
:api_type, :alive_at, presence: true
  validates :api_version, presence: true,
numericality: { only_integer: true }
  validates :webhook_supported, inclusion:
{ in: [true, false] }
  enum api_type: { HTTP: 0, WS: 1, }
  enum status: { disabled: 0, ok: 1,
failed: 2, stale: 3, stopped: 4 }
  delegate :as_json, to: :decorate
end
```
Файл app/models/confirmation_request.rb
```
class ConfirmationRequest <
ApplicationRecord
  belongs_to :user
  enum status: { pending: 0, accepted: 1,
rejected: 2 }
  delegate :name, :username, to: :user,
prefix: true
end
```
Файл app/models/submission.rb
```
class Submission < ApplicationRecord
  include AASM
  validates :source, byte_size: { maximum:
5.kilobytes }
  validate :source_must_be_attached
  belongs_to :problem
  belongs_to :user
  belongs_to :compiler
  has_one_attached :source
  has_many :results, dependent: :destroy
  has_many :logs, dependent: :destroy
  enum test_state: { pending: 0,
in_progress: 1, done: 2, failed: 3 },
_prefix: true
  enum test_result: { ok: 0,
compiler_error: 1 }, _prefix: true
  delegate :as_json, to: :decorate
  delegate :user, :private?, to: :problem,
prefix: true
  after_commit :update_standings
  aasm column: :test_state,
whiny_transitions: false, enum: true do
    state :pending, initial: true
    state :in_progress, :done, :failed
    event(:take) { transitions from:
:pending, to: :in_progress }
    event(:release) { transitions from:
:in_progress, to: :done }
    event :fail, after: -> { increment!
:fails_count } do
      transitions from: :in_progress, to:
:pending, if: -> { fails_count < 5 },
after: -> { results.delete_all }
      transitions from: :in_progress, to:
:failed
    end
  end
  private
  def source_must_be_attached
    errors.add :source, :blank unless
source.attached?
  end
  def update_standings
    StandingRedisStore.update_if_exists
user_id, problem_id
  end
end
```
Файл app/models/problem.rb
```
class Problem < ApplicationRecord
  validates :memory_limit, :time_limit,
:real_time_limit, presence: true,
numericality: true
  validates :private, inclusion: { in:
[true, false] }
  validates :checker_source, byte_size: {
maximum: 5.kilobytes }
  belongs_to :user, optional: true
  belongs_to :checker_compiler,
class_name: 'Compiler'
  has_one_attached :checker_source
  has_one :translation, -> { where
language: I18n.locale }, class_name:
'ProblemTranslation'
  has_one :default_translation, -> { where
default: true }, class_name:
'ProblemTranslation'
  has_many :translations, dependent:
:destroy, class_name: 'ProblemTranslation'
  has_many :tests, dependent: :destroy
  has_many :examples, dependent: :destroy
  has_many :submissions, dependent:
:destroy
  has_many :sharings, dependent: :destroy
  has_many :problems_tags, dependent:
:destroy
  has_many :groups, through: :sharings
  has_many :tags, through: :problems_tags
  default_scope { includes :translation,
:default_translation }
  delegate :as_json, :caption, to:
:decorate
  accepts_nested_attributes_for :examples,
:tests, :translations, :submissions,
allow_destroy: true
  to_param :caption
end
```
Файл app/models/result.rb
```
class Result < ApplicationRecord
  validates :status, presence: true
  validates :memory, :time, presence:
true, numericality: true
  belongs_to :submission
  belongs_to :test
  enum status: {
    ok: 0,
    wrong_answer: 1,
    presentation_error: 2,
    fail: 3,
    dirt: 4,
    points: 5,
    bad_test: 6,
    unexpected_eof: 8,
    runtime_error: 10,
    memory_limit_exceded: 14,
    time_limit_exceded: 15,
    partilly_correct: 16
  }
  delegate :num, to: :test, prefix: true,
allow_nil: true
end
```
Файл app/models/group.rb
```
class Group < ApplicationRecord
  validates :name, :visibility, presence:
true
  belongs_to :owner, class_name: 'User'
  has_many :sharings, dependent: :destroy
  has_many :memberships, dependent:
:destroy
  has_many :pending_memberships, -> {
where.not state: :accepted }, class_name:
'Membership'
  has_many :accepted_memberships, -> {
where state: :accepted }, class_name:
'Membership'
  has_many :problems, -> { order :id },
through: :sharings
  has_many :pending_users, through:
:pending_memberships, source: :user,
class_name: 'User'
  has_many :accepted_users, through:
:accepted_memberships, source: :user,
class_name: 'User'
  has_many :submissions, through:
:accepted_users
  enum visibility: { private: 0,
moderated: 1, public: 2 }, _prefix: true
end
```
Файл app/models/tag.rb
```
class Tag < ApplicationRecord
  has_one :translation, -> { where
language: I18n.locale }, class_name:
'TagTranslation'
  has_many :translations, class_name:
'TagTranslation', dependent: :destroy
  has_many :problems_tags, dependent:
:destroy
  has_many :problems, through:
:problems_tags
  default_scope { includes :translation }
  delegate :as_json, to: :decorate
  delegate :name, to: :translation,
allow_nil: true
end
```
Файл app/models/test.rb
```
class Test < ApplicationRecord
  attr_accessor :input_text, :answer_text
  validates :num, presence: true,
uniqueness: { scope: :problem_id }
  validates :point, presence: true,
numericality: { only_integer: true }
```

```
  belongs_to :problem, touch: true
  has_one_attached :input
  has_one_attached :answer
  has_many :results, dependent: :nullify
  default_scope {
with_attached_input.with_attached_answer.o
rder :num }
  delegate :as_json, to: :decorate
end
Файл app/models/membership.rb
class Membership < ApplicationRecord
  validates :state, presence: true
  validates :user, uniqueness: { scope:
:group }
  validate :user_must_not_be_group_owner
  belongs_to :user
  belongs_to :group
  enum state: %i[requested invited
accepted], _prefix: true
  private
  def user_must_not_be_group_owner
    return unless group.present?
    errors.add :user, :taken if user ==
group.owner
  end
end
Файл app/models/auth_token.rb
class AuthToken < ApplicationRecord
  belongs_to :user
end
Файл app/policies/compiler_policy.rb
class CompilerPolicy < ApplicationPolicy
  def create?
    !!user&.administrator?
  end
  def index?
    !!user&.administrator?
  end
  def show?
    !!user&.administrator?
  end
  def update?
    !!user&.administrator?
  end
  def destroy?
    !!user&.administrator? &&
resource.persisted?
  end
end
Файл app/policies/membership_policy.rb
class MembershipPolicy < ApplicationPolicy
  def index?
    return false if user.blank?
    params[:parent] ? user ==
params[:parent].owner : true
  end
  def create?
    return false if user.blank?
    return true if resource.group.blank?
|| resource.user.blank? ||
resource.state.blank?
    case resource.state
    when 'invited'
      return true if user ==
resource.group.owner
      resource.group.visibility_public? &&
resource.group.accepted_users.include?(use
r)
    when 'requested'
      resource.group.visibility_moderated?
&& user == resource.user
    when 'accepted'
      resource.group.visibility_public? &&
user == resource.user
    end
  end
  def destroy?
    return false if user.blank?
    user == resource.user || user ==
resource.group.owner
  end
  def update?
    return false if user.blank?
    case resource.state
    when 'requested'
      user == resource.group.owner
    when 'invited'
      user == resource.user
    end
  end
end
Файл app/policies/worker_policy.rb
class WorkerPolicy < ApplicationPolicy
  def index?
    !!user&.administrator?
  end
  def destroy?
    !!user&.administrator?
  end
  def create?
    true
  end
  def update?
    true
  end
end
Файл app/policies/problem_policy.rb
class ProblemPolicy < ApplicationPolicy
  def create?
    !!user&.confirmed?
  end
  def update?
    return false if user.blank?
    user.moderator? || user ==
resource.user
  end
  def destroy?
    return false if user.blank?
    user.moderator? || user ==
resource.user
  end
  def index?
    true
  end
  def show?
    return true unless resource.private?
    return false if user.blank?
    return true if user.moderator? || user
== resource.user
    user.shared_problems.include? resource
  end
end
Файл app/policies/avatar_policy.rb
class AvatarPolicy < ApplicationPolicy
  def create?
    return false if user.blank?
    user == resource.user
  end
  def destroy?
    return false if user.blank?
    user == resource.user
  end
end
Файл app/policies/tag_policy.rb
class TagPolicy < ApplicationPolicy
  def index?
    true
  end
end
Файл app/policies/group_policy.rb
class GroupPolicy < ApplicationPolicy
  def index?
    user.present?
  end
  def create?
    !!user&.confirmed?
  end
  def update?
    return false if user.blank?
    user == resource.owner
  end
  def destroy?
    return false if user.blank?
    user == resource.owner
  end
  def show?
    return false if user.blank?
    return true if
resource.visibility_moderated? ||
resource.visibility_public?
    return true if user == resource.owner
    resource.accepted_users.include? user
  end
  def new_invite?
    return false if user.blank?
    return true if user == resource.owner
    resource.visibility_public? &&
resource.accepted_users.include?(user)
  end
  def new_sharing?
    return false if user.blank?
    user == resource.owner
  end
  def index_memberships?
    return false if user.blank?
    user == resource.owner
  end
end
Файл app/policies/session_policy.rb
class SessionPolicy < ApplicationPolicy
  def create?
    !user.present?
  end
  def destroy?
    user.present?
  end
end
Файл
app/policies/confirmation_request_policy.r
b
class ConfirmationRequestPolicy <
ApplicationPolicy
  def create?
    return false if user.blank? ||
user.roles?

user.reload.confirmation_request.blank?
  end
  def index?
    !!user&.administrator?
  end
end
Файл
app/policies/submission/retest_policy.rb
class Submission::RetestPolicy <
ApplicationPolicy
  def create?
    !!user&.administrator?
  end
end
Файл
app/policies/confirmation_request/reject_p
olicy.rb
class ConfirmationRequest::RejectPolicy <
ApplicationPolicy
  def create?
    return false if user.blank?
    return false unless
resource.confirmation_request_pending?
    user.moderator?
  end
end
Файл
app/policies/confirmation_request/accept_p
olicy.rb
class ConfirmationRequest::AcceptPolicy <
ApplicationPolicy
  def create?
    return false if user.blank?
    return false unless
resource.confirmation_request_pending?
    user.moderator?
  end
end
Файл app/policies/password_policy.rb
class PasswordPolicy < ApplicationPolicy
  def update?
    user.blank? && resource.present?
  end
end
Файл
app/policies/password_recovery_policy.rb
class PasswordRecoveryPolicy <
ApplicationPolicy
  def create?
    user.blank?
  end
end
Файл app/policies/application_policy.rb
class ApplicationPolicy
  attr_reader :user, :resource, :params
  def initialize user, resource, **params
    @user, @resource, @params = user,
resource, params
  end
  %i[index? show? create? update?
destroy?].each do |name|
    define_method(name) { false }
  end
  def new?
    create?
  end
  def edit?
    update?
  end
end
Файл app/policies/user_policy.rb
class UserPolicy < ApplicationPolicy
  def index?
    true
  end
  def create?
    !user.present?
  end
  def show?
    return false if user.blank?
    user.id == resource.id
  end
  def update?
    return false if user.blank?
    user.id == resource.id
  end
end
Файл app/policies/result_policy.rb
class ResultPolicy < ApplicationPolicy
  def create?

resource.submission.test_state_in_progress
?
  end
end
Файл app/policies/submission_policy.rb
class SubmissionPolicy < ApplicationPolicy
  def index?
    true
  end
  def create?
    return false if user.blank?
    return true unless
resource.problem_private?
    return true if user.moderator? || user
== resource.problem_user
    user.shared_problems.include?
resource.problem
  end
  def show?
    return false if user.blank?
    user.administrator? || user ==
resource.user || user ==
resource.problem_user
  end
  def destroy?
    !!user&.administrator?
  end
end
Файл app/policies/sharing_policy.rb
class SharingPolicy < ApplicationPolicy
  def new?
    return false if user.blank?
    user == resource.group_owner
  end
```

```
  def create?
    return false if user.blank?
    user == resource.group_owner && user
== resource.problem_user
  end
end
```

Файл app/policies/log_policy.rb

```
class LogPolicy < ApplicationPolicy
  def create?

resource.submission.test_state_in_progress
?
  end
  def show?
    return false if user.blank?
    return true if user.administrator?
    return true if resource.source? &&
resource.submission_user == user
    resource.submission_problem_user ==
user
  end
end
```

Файл app/policies/archive_policy.rb

```
class ArchivePolicy < ApplicationPolicy
  def create?
    !!user&.administrator?
  end
end
```

Файл app/decorators/membership_decorator.rb

```
class MembershipDecorator <
Draper::Decorator
  delegate_all
  decorates_associations :user, :group
  delegate :name, to: :user, prefix: true,
allow_nil: true
  delegate :name, to: :group, prefix:
true, allow_nil: true
end
```

Файл app/decorators/worker_decorator.rb

```
class WorkerDecorator < Draper::Decorator
  delegate_all
  def as_json *_args
    { id: id }
  end
  def status_class
    case status.to_sym
    when :disabled        then 'border-
dark'
    when :ok              then 'border-
success'
    when :failed          then 'border-
danger'
    when :stale, :stopped then 'border-
warning'
    end
  end
end
```

Файл app/decorators/group_decorator.rb

```
class GroupDecorator < Draper::Decorator
  delegate_all
  decorates_associations :owner,
:accepted_users, :problems
  delegate :name, to: :owner, prefix: true
  delegate :state_requested?,
:state_invited?, :state_accepted?, to:
:current_user_membership, prefix: true,
allow_nil: true
  def visibility_icon
    h.content_tag :i, '', class:
visibility_icon_class, title:
visibility.humanize,
      data: { toggle: :tooltip, placement:
:bottom }
  end
  def current_user_membership
    @current_user_membership ||=
memberships.find_by user: h.current_user
if h.current_user.present?
  end
  private
  def visibility_icon_class
    case visibility.to_sym
    when :private, :moderated
      'mr-3 fas fa-lock'
    when :public
      'mr-3 fas fa-unlock'
    end
  end
end
```

Файл app/decorators/user_decorator.rb

```
class UserDecorator < Draper::Decorator
  delegate_all
  def as_json *args
    { id: id, name: name,
search_suggestion: search_suggestion }
  end
  def name
    super.presence || username
  end
  def search_suggestion
    body = ''.html_safe
    body << h.content_tag(:p, name, class:
'mb-0')
    body << h.content_tag(:small,
username, class: 'text-muted')
    h.content_tag :div, body
  end
end
```

Файл app/decorators/compiler_decorator.rb

```
class CompilerDecorator <
Draper::Decorator
  delegate_all
  def as_json *args
    {
      id: id,
      name: name,
      version: version,
      memory_a: memory_a,
      memory_b: memory_b,
      time_a: time_a,
      time_b: time_b
    }
  end
end
```

Файл app/decorators/avatar_decorator.rb

```
class AvatarDecorator < Draper::Decorator
  delegate_all
  def as_json *args
    { url: url }
  end
  def url
    return unless user.avatar.attached?
    helpers.url_for user.avatar.variant
combine_options: { resize: '300x400^',
crop: '300x400+0+0', gravity: 'center' }
  end
end
```

Файл app/decorators/test_decorator.rb

```
class TestDecorator < Draper::Decorator
  delegate_all
  def as_json *args
    {
      id: id,
      num: num,
      input_url: input_url,
      answer_url: answer_url
    }
  end
  def input_url
    helpers.url_for input if
input.attached?
  end
  def answer_url
    helpers.url_for answer if
answer.attached?
  end
end
```

Файл app/decorators/tag_decorator.rb

```
class TagDecorator < Draper::Decorator
  delegate_all
  def as_json *args
    { value: id, text: name }
  end
end
```

Файл app/decorators/submission_decorator.rb

```
class SubmissionDecorator <
Draper::Decorator
  delegate_all
  decorates_association :problem, context:
:submission
  decorates_association :user
  delegate :username, to: :user, prefix:
true
  delegate :caption, to: :problem, prefix:
true
  def as_json *args
    {
      id: id,
      compiler_id: compiler_id,
      problem: problem,
      source_url: source_url,
      test_state:
Submission.test_states[test_state],
      fails_count: fails_count,
      memory_limit: memory_limit,
      time_limit: time_limit
    }
  end
  def source_url
    helpers.url_for source if
source.attached?
  end
  def memory_limit
    problem.memory_limit *
compiler.memory_a + compiler.memory_b
  end
  def time_limit
    problem.time_limit * compiler.time_a +
compiler.time_b
  end
  def state
    case
    when submission.test_state_done? &&
submission.test_result_ok?
      "#{ submission.test_result.humanize
} (#{ submission.score }/#{
submission.max_score })"
    when submission.test_state_done?
      submission.test_result.humanize
    else
      submission.test_state.humanize
    end
  end
end
```

Файл app/decorators/problem_decorator.rb

```
class ProblemDecorator < Draper::Decorator
  delegate_all
  delegate :caption, :author, :text,
:technical_text, to: :translation,
allow_nil: true
  def as_json *args
    case context
    when :submission
      {
        id: id,
        updated_at: updated_at,
        checker_compiler_id:
checker_compiler_id
      }
    else
      {
        id: id,
        updated_at: updated_at,
        checker_compiler_id:
checker_compiler_id,
        checker_source_url:
checker_source_url,
        tests: tests
      }
    end
  end
  def checker_source_url
    helpers.url_for checker_source if
checker_source.attached?
  end
  def translation
    super || default_translation
  end
  def language
    translation.language.to_sym if
translation
  end
end
```

Файл app/factories/membership_factory.rb

```
class MembershipFactory <
ApplicationFactory
  def initialize current_user, params
    @current_user, @params = current_user,
params
  end
  def build
    Membership.new
@params.slice(:group).merge(user: user,
state: state)
  end
  private
  def user
    case @params[:type]
    when 'invite'
      User.find_by id: @params[:user_id]
    when 'request'
      @current_user
    end
  end
  def state
    return unless @params[:group]
    case @params[:type]
    when 'invite' then :invited
    when 'request'
      case @params[:group].visibility
      when 'moderated' then :requested
      when 'public'    then :accepted
      end
    end
  end
end
```

Файл app/factories/application_factory.rb

```
class ApplicationFactory
  class << self
    def build *args
      new(*args).build
    end
  end
end
```

Файл app/assets/stylesheets/btn-default.scss

```
.btn-default {
  color: #ffffff;
  background-color: #0093FF;
  border-color: #0093FF;
  transition: all .2s ease-in-out;
  &:focus {
    color: #ffffff;
    background-color: #0093FF;
    border-color: #0093FF;
  }
  &:hover {
    color: lightcyan;
    background-color: #000000;
    border-color: #000000;
  }
}
```

Файл app/assets/stylesheets/member-list-actions.scss

```
.member-list-actions {
  margin-top: -1px;
  .btn.list-group-item {
    border-radius: 0;
    &:last-child {
      border-bottom-right-radius: 0.25rem;
      border-bottom-left-radius: 0.25rem;
    }
  }
}
```

Файл app/assets/stylesheets/nav-tabs.scss

```
.nav-tabs {
  margin-bottom: 1rem;
  border-color: #0093FF;
```

```
  .nav-link {
    color: #000000;
    &.active {
      color: #ffffff;
      background-color: #0093FF;
      border-color: #0093FF;
    }
  }
  .nav-link:hover {
    border-color: #eee #eee #0093FF;
  }
}
```

Файл app/assets/stylesheets/form-control-feedback.css

```
i.form-control-feedback {
  font-size: 1.5em;
  color: #000000;
}
```

Файл app/assets/stylesheets/member-list.scss

```
.member-list {
  max-height: 496px;
  overflow-y: auto;
  &__item {
    display: flex;
    justify-content: space-between;
    align-items: center;
    &:last-child {
      border-radius: 0;
    }
  }
}
```

Файл app/assets/stylesheets/process-problem-archive-log.css

```
#process-problem-archive-log {
  margin-top: 2rem;
}
```

Файл app/assets/stylesheets/tags.scss

```
.tags {
  margin: 0;
  list-style: none;
  li {
    display: inline-block;
    &:not(:first-child)::before {
      padding: 0 5px;
      color: #ccc;
      content: "\ff5c";
    }
  }
}
```

Файл app/assets/stylesheets/bootstrap-tagsinput.scss

```
.bootstrap-tagsinput {
  width: 100%;
  .label {
    background-color: #1e90ff;
  }
  .tag {
    padding: 3px 6px;
    border-radius: 3px;
  }
}
```

Файл app/assets/stylesheets/header.scss

```
.header {
  width: 100%;
  background: #000000;
  display: flex;
  justify-content: space-between;
  height: 70px;
  line-height: 70px;
  padding: 0 15px;
  &__logo {
    margin: 10px;
  }
}
```

Файл app/assets/stylesheets/home-section.scss

```
.home-section {
  padding: 30px 20px;
  .home-section__title {
    margin-bottom: 40px;
  }
  .home-section__content {
    .fa {
      color: #0093FF;
    }
  }
}
```

Файл app/assets/stylesheets/avatar.css

```
.avatar {
  object-fit: cover;
  height: 400px;
  width: 100%;
}
@media (max-width: 990px) {
  .avatar {
    margin-bottom: 20px;
  }
}
```

Файл app/assets/stylesheets/pagination.scss

```
.pagination {
  .page-item {
    .page-link {
      color: #593131;
      background-color: #ffc18e;
      border-color: #b64949;
      &:hover {
        color: #ffc18e;
        background-color: #543b3b;
        border-color: #b64949;
      }
    }
    &.active {
      .page-link,
      .page-link:hover {
        color: #ffc18e;
        background-color: #543b3b;
        border-color: #b64949;
      }
    }
    &.disabled {
      .page-link,
      .page-link:hover {
        color: #000000;
        background-color: #ffc18e;
        border-color: #b64949;
      }
    }
  }
}
```

Файл app/assets/stylesheets/section.scss

```
.section {
  background: #fff;
  color: #000000;
  border-radius: 5px;
  margin-bottom: 12px;
  clip-path: polygon(10px 0%, calc(100% - 10px) 0%, 100% 10px, 100% calc(100% - 10px), calc(100% - 10px) 100%, 10px 100%, 0% calc(100% - 10px), 0% 10px);
  padding: 15px;
  &:not(:last-child) {
    border-bottom: 1px solid #eee;
  }
  h2, h3 {
    margin-top: 10px
  }
  &__title {
    text-align: center;
    padding-bottom: 1.75rem;
  }
  &__content {
    text-align: justify;
    list-style-position: inside;
  }
  &--warning {
    text-align: center;
    background-color: #fcf8e3;
    color: #8a6d3b;
  }
  &--striped:nth-child(even) {
    background: #efefef;
  }
  &--with-icon {
    display: flex;
    align-items: center;
    i {
      font-size: 1.5em;
      margin-right: 20px;
    }
  }
}
```

Файл app/assets/stylesheets/pagination-wrapper.css

```
.pagination-wrapper {
  width: 100%;
  display: flex;
  align-items: center;
  flex-direction: column;
}
```

Файл app/assets/stylesheets/side-buttons.css

```
.side-buttons {
  position: absolute;
  left: 100%;
  top: 0;
  padding: 1rem 2rem;
  display: flex;
  flex-direction: column;
}
```

Файл app/assets/stylesheets/page.scss

```
.page {
  margin-top: 30px;
  margin-bottom: 30px;
  padding: 15px 0;
  border-radius: 5px;
  &:not(.row) {
    padding: 15px;
  }
  &__title {
    display: flex;
    align-items: center;
    justify-content: space-between;
    margin-bottom: 1.5rem;
  }
}
```

Файл app/assets/stylesheets/application.scss

```
@import "bootstrap-tagsinput.min";
@import "overlay";
@import "bootstrap";
@import "toastr/toastr";
@import "sweetalert2";
@import "*";
@font-face {
  font-family: "Century Gothic";
  src: asset-url("CenturyGothic.ttf") format("truetype");
}
* {
  margin: 0;
  padding: 0;
}
html {
  font-size: 14px;
}
body {
  background: #6d3606 image-url("background-1.jpg");
  background-size: cover;
  font-family: "Century Gothic", sans-serif;
  position: relative;
  min-height: 100vh;
  padding-bottom: 80px;
}
::selection {
  background: #93440180;
}
:focus {
  outline: none;
}
input[readonly] {
  background: #e9e9e9 !important;
}
textarea {
  resize: vertical;
  min-height: 35px;
}
form {
  width: 100%;
}
pre {
  padding: 12px;
  background: #e1e1e1;
  border-radius: 5px;
  color: #545454;
}
```

Файл app/assets/stylesheets/header-section.scss

```
.header-section {
  display: inline-flex;
  justify-content: space-between;
  &__item > a {
    display: inline-block;
    cursor: pointer;
    color: #ffffff !important;
    background-color: #000000;
    padding: 0 20px;
    font-size: 16px;
    text-transform: uppercase;
    text-decoration: none;
    transition: all .2s ease-in-out;
    white-space: nowrap;
    &:hover {
      color: #ffffff;
      background-color: #0093FF;
      text-decoration: none;
    }
    &:focus {
      color: #ffffff;
      text-decoration: none;
    }
    &--active {
      color: #0093FF;
      &:focus {
        color: #0093FF;
      }
      &:hover {
        color: #ffffff;
      }
    }
  }
}
```

Файл app/assets/stylesheets/result-log.css

```
.result-log {
  text-align: left;
  width: 250px;
  padding: 12px;
  background: #e1e1e1;
  border-radius: 5px;
  color: #545454;
}
```

Файл app/assets/stylesheets/toast-close-button.css

```
.toast-close-button {
  right: unset;
  top: -1px;
}
```

Файл app/assets/stylesheets/bg.css

```
.bg-dark {
  background-color: rgba(0, 0, 0, 0.5) !important;
  color: #ffffff
}
.bg-light {
  background-color: #f6f6f6;
  color: #000000;
}
.bg-white {
  background-color: #ffffff;
  color: #000000;
}
.bg-transparent {
  background-color: transparent;
  color: #ffffff;
}
```

Файл app/assets/stylesheets/translation.scss

```
.translation {
  box-shadow: #924301 0 0 5px -0.5px;
  padding: 15px;
```

```
  margin-bottom: 20px;
  &__top {
    .float-left {
      width: calc(100% - 35px);
    }
    .float-right {
      margin-top: 1.8rem;
    }
  }
}
```

Файл app/assets/stylesheets/ckeditor.css

```
.ck-content {
  list-style-position: inside;
}
```

Файл app/assets/stylesheets/problem-example.css

```
.problem-example {
  background-color: #ffe4ce;
  border-radius: 5px;
  box-shadow: 0 0 7px 1.2px #ffbd88;
  color: #933706;
  font-family: monospace;
  padding: 7.5px 15px;
}
```

Файл app/assets/stylesheets/dropdown-item.css

```
.dropdown-item {
  line-height: 30px;
}
```

Файл app/assets/stylesheets/form-page.css

```
.form-page {
  margin: 50px auto 0 auto;
  padding: 30px 20px;
  width: 420px;
}
```

Файл app/assets/stylesheets/footer.scss

```
.footer {
  position: absolute;
  bottom: 0;
  width: 100%;
  text-align: center;
  color: #ffffff;
  background-color: rgba(0, 0, 0, 0.25);
  line-height: 20px;
  padding: 20px;
  &--revision {
    font-size: 0.85em;
    color: rgba(255, 255, 255, 0.2);
  }
  &:hover {
    .footer--revision {
      color: rgba(255, 255, 255, 1);
    }
  }
}
```

Файл app/assets/stylesheets/show.scss

```
.winter-is-coming {
  z-index: -1;
  pointer-events: none;
  overflow: hidden;
  position: absolute;
  top: 0;
  bottom: 0;
  left: 0;
  right: 0;
}
.snow {
  z-index: -1;
  pointer-events: none;
  position: absolute;
  top: 0; right: 0; bottom: 0; left: 0;
  animation: falling linear infinite both;
  transform: translate3D(0,-100%,0);
  &--near {
    animation-duration: 10s;
    background-image:
url('https://dl6rt3mwcjzxg.cloudfront.net/
assets/snow/snow-large-
075d267ecbc42e3564c8ed43516dd557.png');
    background-size: contain;
    & + .snow--alt {
      animation-delay: (5s);
    }
  }
  &--mid {
    animation-duration: 20s;
    background-image:
url('https://dl6rt3mwcjzxg.cloudfront.net/
assets/snow/snow-medium-
0b8a5e0732315b68e1f54185be7a1ad9.png');
    background-size: contain;
    & + .snow--alt {
      animation-delay: (10s);
    }
  }
  &--far {
    animation-duration: 30s;
    background-image:
url('https://dl6rt3mwcjzxg.cloudfront.net/
assets/snow/snow-small-
1ecd03b1fce08c24e064ff8c0a72c519.png');
    background-size: contain;
    & + .snow--alt {
      animation-delay: (15s);
    }
  }
}
@keyframes falling {
  0% {
    transform: translate3D(-7.5%,-100%,
0);
  }
  100% {
    transform: translate3D(7.5%,100%, 0);
  }
}
```

Файл app/assets/stylesheets/toast-container.css

```
#toast-container {
  line-height: 14px;
  top: 82px;
}
```

Файл app/assets/stylesheets/typeahead.scss

```
.tt-menu {
  width: 20rem;
  border-radius: 5px;
  background: #fff;
  box-shadow: 2px 2px 5px 2px #eee;
}
.tt-hint {
  display: none;
}
.tt-input {
  background: #fff !important;
  color: #000 !important;
}
.tt-suggestion {
  padding: 7px;
  border-radius: 5px;
  width: 100%;
}
.tt-suggestion:hover {
  background: #f3f3f3;
}
.form-group > .twitter-typeahead {
  display: block !important;
}
```

Файл app/assets/stylesheets/form-group-invalid.scss

```
.form-group-invalid {
  color: #CB0B0A;
  .invalid-feedback {
    display: block;
  }
}
```

Файл app/assets/stylesheets/image-input.scss

```
.image-input {
  position: relative;
  .image-input__prompt {
    position: absolute;
    left: 0;
    right: 0;
    top: 0;
    bottom: 0;
    display: flex;
    opacity: 0;
    background-color: rgba(0, 0, 0, 0.5);
    transition: opacity .2s ease-in-out;
    cursor: pointer;
    .image-input__prompt-icon {
      text-align: center;
      align-self: center;
      width: 100%;
      color: #ffffff;
      font-size: 42px;
    }
  }
  &:hover {
    .image-input__prompt {
      opacity: 1;
    }
  }
}
```

Файл app/assets/stylesheets/table-custom.scss

```
.table-custom {
  th, td {
    border-top: none;
    border-right: 1px solid #ffffff;
  }
  th {
    text-transform: capitalize;
    font-weight: bold;
  }
  tr {
    text-align: center;
    background-color: #ffd7b6;
    &:nth-of-type(odd) {
      background-color: #ffe4ce;
    }
    &:hover {
      background-color: darken(#ffd7b6,
10%)
    }
  }
}
```

Файл app/assets/config/manifest.js

```
// app/assets/config/manifest.js
//
//= link application.css
//
//= link application.js
//= link ckeditor.js
//
//= link_tree ../images
//
```

Файл app/assets/javascripts/turbolinks-persistent-csp-nonce.js

```
(function () {
  var nonce = undefined;
  document.addEventListener('turbolinks:load
', function () {
    if (nonce === undefined)
      nonce = $('meta[name="csp-
nonce"]').prop('content');
    else
      $('meta[name="csp-
nonce"]').prop('content', nonce);
  });
})();
```

Файл app/assets/javascripts/application.js

```
//= require jquery3
//= require popper
//= require bootstrap
//= require rails-ujs
//= require activestorage
//= require turbolinks
//= require toastr/toastr
//= require cocoon
//= require sweetalert2
//= require sweet-alert2-rails
//= require bootstrap-tagsinput.min.js
//= require typeahead.min.js
//= require bs-custom-file-input/dist/bs-
custom-file-input.min.js
//= require_tree .
$(function () {
  $('input[readonly]').on('focus',
function () { $(this).blur(); });
  $('[data-toggle="tooltip"]').tooltip();
});
```

Файл app/assets/javascripts/cable.js

```
//= require action_cable
//= require_self
(function () {
  if (!this.App) this.App = {};
  App.cable =
ActionCable.createConsumer();
}).call(this);
```

Файл app/assets/javascripts/textarea-ckeditor.js

```
(function () {
  function replaceTextareas() {
    $('[data-
textarea="ckeditor"]').each(function () {

ClassicEditor.create(this).catch(function
(err) { console.error(err.stack) });
      $(this).attr('data-textarea',
'replaced');
    })
  }
  $(document).on('turbolinks:load',
replaceTextareas);
  $(document).on('cocoon:after-insert',
replaceTextareas);
})();
```

Файл app/assets/javascripts/new_problem.js

```
(function () {
  function newProblem() {
    var form = $(this);
    form.find(':submit').on('click',
function () {
      this.disabled = true;
      $('#process-problem-archive-
log').html('');
      form.submit();
    });
  }
  jQuery.fn.extend({ newProblem: function
() { return this.each(newProblem); } });

document.addEventListener('turbolinks:load
', function () {
$('form#new_problem').newProblem(); });
})();
```

Файл app/assets/javascripts/image-input.js

```
(function () {
  function imageInput() {
    var block = $(this);
    var input = block.children('input');
    var image = block.children('img');
    block.children('.image-
input__prompt').click(function () {
input.click(); });
    function success(data) {
image.attr('src', data.url); }
    input.change(function () {
      var files = input.prop('files');
      var data = new FormData();
      data.append('avatar[file]', files &&
files[0]);
      $.ajax({
        url: '/avatar',
        cache: false,
        contentType: false,
        processData: false,
        method: 'POST',
        dataType: 'json',
        data: data,
        success: success
      });
    });
  }
  jQuery.fn.extend({ imageInput: function
() { return this.each(imageInput); } });

document.addEventListener('turbolinks:load
', function () { $('.image-
input').imageInput(); });
```

```
})();
Файл app/assets/javascripts/sort.js
(function () {
  function onSortSuccess(result) {
$('table tbody').html(result); }
  function sortTable(data) {
    $.ajax({ url: '', data:
JSON.stringify(data), success:
onSortSuccess });
  }
  function getRemote(remote_url) {
    return JSON.parse($.ajax({ type:
"GET", url: remote_url, async: false
}).responseText);
  }
  function initializeSortBy() {
    var usernames = new Bloodhound({
      datumTokenizer:
Bloodhound.tokenizers.whitespace,
      queryTokenizer:
Bloodhound.tokenizers.whitespace,
      local:
getRemote('/v2/tests/users.json')
    });
    $('#sortByUser').typeahead(null, {
name: 'usernames', source: usernames });
    var problems = new Bloodhound({
      datumTokenizer:
Bloodhound.tokenizers.whitespace,
      queryTokenizer:
Bloodhound.tokenizers.whitespace,
      local:
getRemote('/v2/tests/problems.json')
    });
    $('#sortByTask').typeahead(null, {
name: 'problems', source: problems });
    var groups = new Bloodhound({
      datumTokenizer:
Bloodhound.tokenizers.whitespace,
      queryTokenizer:
Bloodhound.tokenizers.whitespace,
      local:
getRemote('/v2/tests/groups.json')
    });
    $('#sortByGroup').typeahead(null, {
name: 'groups', source: groups });
    // -------------- Process the
selection -------------- //
    var data = { status: null, user: null,
problem: null, group: null };
    $('#sortByStatus').change(function ()
{
      data.status =
$(this).find("option:selected").val();
      sortTable(data);
    });

$('#sortByTask').bind('typeahead:select',
function (e) {
      data.problem = e.target.value;
      sortTable(data);
    });

$('#sortByUser').bind('typeahead:select',
function (e) {
      data.user = e.target.value;
      sortTable(data);
    });

$('#sortByGroup').bind('typeahead:select',
function (e) {
      data.group = e.target.value;
      sortTable(data);
    });
  }
  $(document).on('turbolinks:load',
function () {
    if (typeof sortByUser !== 'undefined')
      initializeSortBy();
  });
})();
Файл app/assets/javascripts/tags-select.js
(function () {
  var data = new Bloodhound({
    datumTokenizer:
Bloodhound.tokenizers.obj.whitespace('text
'),
    queryTokenizer:
Bloodhound.tokenizers.whitespace,
    prefetch: { url: "/tags.json", cache:
false }
  });
  data.initialize();
  function tagsSelect() {
    var block = $(this);
    block.tagsinput({
      itemValue: 'value',
      itemText: 'text',
      typeaheadjs: { name: 'cities',
displayKey: 'text', source:
data.ttAdapter() }
    });
    $.map(this.options, function (option)
{
      block.tagsinput('add', { value:
option.value, text: option.text });
    });
  }
  jQuery.fn.extend({ tagsSelect: function
() { return this.each(tagsSelect); } });

document.addEventListener('turbolinks:load
', function () { $('[data-role=tags-
select]').tagsSelect(); });
})();
Файл app/assets/javascripts/autorefresh-
workers-page.js
(function () {
  var regex = /^\/[a-z][a-
z]\/workers\/?$/i;
  var handle = null;
  function refresh() {

Turbolinks.visit(window.location.toString(
));
  }

document.addEventListener('turbolinks:load
', function () {
    clearTimeout(handle)
    if
(regex.test(window.location.pathname))
      handle = setTimeout(refresh, 5000);
  });
})();
Файл app/assets/javascripts/flash-
messages.js
toastr.options.closeButton = true;
toastr.options.timeOut = 1500;
toastr.recaptcha_error = toastr.error;
document.addEventListener('turbolinks:load
', function () {
  $('body').data('flash').forEach(function
(m) { toastr[m[0]](m[1]); });
});
Файл app/assets/javascripts/tagsinput.js
document.addEventListener('turbolinks:load
', function () { $('[data-
role="tagsinput"]').tagsinput(); });
Файл app/assets/javascripts/test-fields-
swap.js
function initializeTestFieldsSwap() {
  $(document).on('click', '[data-
toggle=test-fields-swap]', function () {
    var row = $(this).closest('tr');
    var items = [
      row.find('.problem_tests_input'),

row.find('.problem_tests_input_text'),
      row.find('.problem_tests_answer'),

row.find('.problem_tests_answer_text')
    ];
    items.forEach(function (item) {
item.toggleClass('d-none'); });
  });
}
document.addEventListener('turbolinks:load
', function () {
initializeTestFieldsSwap(); });
Файл app/assets/javascripts/mathjax.js
(function () {
  var regex = /^\/[a-z][a-
z]\/problems\/[\-a-zA-Z0-9]+\/?$/i;

document.addEventListener('turbolinks:load
', function () {
    if
(regex.test(window.location.pathname))
      MathJax.Hub.Queue(['Typeset',
MathJax.Hub]);
  });
})();
$(function () { MathJax.Hub.Config({
skipStartupTypeset: true }); })
Файл app/assets/javascripts/process-
problem-archive-log.js
(function () {
  function processProblemArchiveLog() {
    var block = $(this);
    function received(data) {
      block.append('<p>' + data + '</p>');
      if (data === 'Done.')
        $('form#new_problem
:submit').prop('disabled', false);
    }
    App.processProblemArchiveSubscription
= App.cable.subscriptions.create(
      { channel:
'ProcessProblemArchiveChannel', id:
block.data('channel-id') },
      { received: received }
    );
  }
  jQuery.fn.extend({
processProblemArchiveLog: function () {
return
this.each(processProblemArchiveLog); } });

document.addEventListener('turbolinks:load
', function () {
    $('#process-problem-archive-
log').processProblemArchiveLog();
  });

document.addEventListener('turbolinks:befo
re-render', function () {
    if
(App.processProblemArchiveSubscription) {
App.processProblemArchiveSubscription.unsu
bscribe();

App.processProblemArchiveSubscription.cons
umer.disconnect();

App.processProblemArchiveSubscription =
undefined;
    }
  });
})();
Файл app/assets/javascripts/bs-custom-
file-input.js
document.addEventListener('turbolinks:load
', function () {
  bsCustomFileInput.init();
});
Файл app/assets/javascripts/user-search.js
(function () {
  function suggestion(data) { return
data.search_suggestion; }
  function onSelect(block) {
    var input =
block.closest('form').find(block.data('tar
get'));
    return function (_, selected) {
input.val(selected.id); };
  }
  var bloodhoundOptions = {
    datumTokenizer:
Bloodhound.tokenizers.whitespace,
    queryTokenizer:
Bloodhound.tokenizers.whitespace,
    remote: { wildcard: '%Q%', url:
'/users?query=%Q%' }
  };
  var typeaheadOptions = { highlight:
true, minLength: 3 }
  var typeaheadDatasets = {
    display: 'name',
    source: new
Bloodhound(bloodhoundOptions),
    templates: { suggestion: suggestion }
  }
  function initializeUserSearch() {
    var block = $(this);
    block.typeahead(typeaheadOptions,
typeaheadDatasets);
    block.bind('typeahead:select',
onSelect(block));
  }
  $.fn.extend({ userSearch: function () {
return this.each(initializeUserSearch); }
});
})();
document.addEventListener('turbolinks:load
', function () { $('input[data-role=user-
search]').userSearch(); });
Файл app/searchers/application_searcher.rb
class ApplicationSearcher
  attr_reader :relation, :params
  def initialize relation, params = {}
    @relation = relation
    @params = params
  end
  def search
    return relation unless params
    conditions = params.to_unsafe_hash.map
do |attribute, value|
      method_name = :"search_by_#{
attribute }"
      send method_name, value if
respond_to? method_name, true
    end
    conditions.compact.reduce(:merge) ||
relation
  end
  class << self
    def search *args
      new(*args).search
    end
  end
end
Файл app/searchers/user_searcher.rb
class UserSearcher < ApplicationSearcher
  def search_by_query query
    relation.where 'username ILIKE :query
OR name ILIKE :query', query: "%#{ query
}%" if query.present?
  end
end
Файл app/views/workers/_worker.html.erb
<%= content_tag :div, class: "card #{
worker.status_class } mb-3 worker-card" do
%>
  <div class="card-header d-flex justify-
content-between align-items-center">
    <%= worker.name %>
    <%= link_to worker, class: 'btn btn-
danger', remote: true, method: :delete do
%>
      <i class="fas fa-times"></i>
    <% end %>
  </div>
  <div class="card-body">
    <p class><b>IPs:</b> <%=
worker.ips.to_sentence(two_words_connector
: ', ', last_word_connector: ', ') %></h5>
    <p class="card-text">
```

```
      <%= worker.api_type %> api version
<%= worker.api_version %>
      <%= worker.webhook_supported ?
'with' : 'without' %> webhook support
    </p>
    <% if worker.task_status? %>
      <p class="card-
title"><b>Tasks:</b></p>
      <% worker.task_status.each do
|status| %>
        <p class="card-text"><%= status
%></p>
      <% end %>
    <% end %>
    <p class="card-text"><small
class="text-muted">
      Last update at <%= l
worker.updated_at %> (<%=
time_ago_in_words worker.updated_at,
include_seconds: true %> ago)
    </small></p>
  </div>
<% end %>
Файл app/views/workers/index.html.erb
<div class="page bg-light">
  <h1><%= t 'workers.index.title' %></h1>
  <div>
    <% collection.decorate.each_slice 3 do
|slice| %>
      <div class="card-deck">
        <% slice.each do |worker| %>
          <%= render worker %>
        <% end %>
      </div>
    <% end %>
  </div>
</div>
Файл
app/views/password_recoveries/create.html.
erb
<div class="page form-page bg-dark row">
  <div class="col-12">
    <h1><%= t '.title' %></h1>
  </div>
  <div class="col-12">
    <%= t '.content' %>
  </div>
</div>
Файл
app/views/password_recoveries/new.html.erb
<div class="page form-page bg-dark">
  <%= simple_form_for resource, url:
:password_recovery, remote: true do |f| %>
    <%= f.input :email %>
    <div class="form-group">
      <%= recaptcha_tags theme: :dark %>
    </div>
    <%= f.submit class: 'btn btn-primary'
%>
  <% end %>
</div>
Файл app/views/application/index.json.erb
<%= sanitize collection.to_json %>
Файл app/views/application/errors.json.erb
<%= sanitize({ errors: resource.errors
}.to_json) %>
Файл app/views/application/show.json.erb
<%= sanitize resource.to_json %>
Файл app/views/application/create.json.erb
<%= sanitize resource.to_json %>
Файл
app/views/compilers/_compiler.html.erb
<tr>
  <td><%= compiler.id  %></td>
  <td><%= link_to compiler.name, [:edit,
compiler] %></td>
  <td><%= compiler.version %></td>
  <td><%= compiler.status.humanize %></td>
</tr>
Файл app/views/compilers/edit.html.erb
<%= render 'form' %>
Файл app/views/compilers/_form.html.erb
<%= simple_form_for resource, remote: true
do |f| %>
  <div class="page row bg-light">
    <div class="col-12">
      <%= f.input :id, readonly: true %>
      <%= f.input :name %>
    </div>
    <div class="col-6">
      <%= f.input :version %>
      <%= f.input :memory_a %>
      <%= f.input :memory_b %>
    </div>
    <div class="col-6">
      <%= f.input :status, as: :select,
collection: Compiler.statuses.keys,
include_blank: false %>
      <%= f.input :time_a %>
      <%= f.input :time_b %>
    </div>
    <div class="col-12">
      <%= f.submit class: 'btn btn-primary
float-left' %>
      <%= link_to
t('compiler.form.destroy'), resource,
method: :delete, remote: true, class: 'btn
btn-danger float-right',
        data: { confirm: t('confirm') } if %>
policy(resource).destroy? %>
    </div>
  </div>
<% end %>
Файл app/views/compilers/index.html.erb
<div class="page bg-light">
  <div class="page__title">
    <h1><%= t 'compiler.index.title'
%></h1>
    <%= link_to t('compiler.index.new'),
%i(new compiler), class: 'btn btn-primary'
%>
  </div>
  <table class="table table-custom">
    <thead class="thead-dark">
      <tr>
        <th><%= t 'compiler.index.id'
%></th>
        <th><%= t 'compiler.index.name'
%></th>
        <th><%= t 'compiler.index.version'
%></th>
        <th><%= t 'compiler.index.status'
%></th>
      </tr>
    </thead>
    <tbody>
      <%= render collection %>
    </tbody>
  </table>
  <%= paginate collection %>
</div>
Файл app/views/compilers/new.html.erb
<%= render 'form' %>
Файл
app/views/kaminari/_prev_page.html.erb
<li class="page-item">
  <%= link_to url, class: 'page-link',
rel: 'prev', remote: remote do %>
    <i class="fas fa-angle-left"></i>
  <% end %>
</li>
Файл app/views/kaminari/_gap.html.erb
<li class="page-item disabled">
  <%= link_to '...', '#', class: 'page-
link' %>
</li>
Файл
app/views/kaminari/_paginator.html.erb
<%= paginator.render do -%>
  <nav class="pagination-wrapper">
    <ul class="pagination">
      <%  unless current_page.first? %>
        <%= first_page_tag %>
        <%= prev_page_tag %>
      <% end %>
      <% each_page do |page| -%>
        <% if page.left_outer? ||
page.right_outer? || page.inside_window? -
%>
          <%= page_tag page %>
        <% elsif !page.was_truncated? -%>
          <%= gap_tag %>
        <% end -%>
      <% end -%>
      <% unless current_page.last? %>
        <%= next_page_tag %>
        <%= last_page_tag %>
      <% end %>
    </ul>
  </nav>
<% end -%>
Файл
app/views/kaminari/_first_page.html.erb
<li class="page-item">
  <%= link_to url, class: 'page-link',
remote: remote do %>
    <i class="fas fa-angle-double-
left"></i>
  <% end %>
</li>
Файл app/views/kaminari/_page.html.erb
<li class="page-item <%= 'active' if
page.current? %>">
  <%= link_to page, url, class: 'page-
link', remote: remote %>
</li>
Файл
app/views/kaminari/_next_page.html.erb
<li class="page-item">
  <%= link_to url, class: 'page-link',
rel: 'next', remote: remote do %>
    <i class="fas fa-angle-right"></i>
  <% end %>
</li>
Файл
app/views/kaminari/_last_page.html.erb
<li class="page-item">
  <%= link_to url, class: 'page-link',
remote: remote do %>
    <i class="fas fa-angle-double-
right"></i>
  <% end %>
</li>
Файл app/views/problems/_problem.html.erb
<tr>
  <td><%= problem.id %></td>
  <td>
    <% if policy(problem).show? %>
      <%= link_to problem.caption, problem
%>
    <% else %>
      <%= problem.caption %>
    <% end %>
  </td>
  <td>0%</td>
</tr>
Файл
app/views/problems/_translations.html.erb
<div id="translations-tab" class="tab-pane
fade">
  <div id="translations"
class="translations">
    <%= f.simple_fields_for :translations
do |builder| %>
      <%= render 'translation_fields', f:
builder %>
    <% end %>
  </div>
  <%= link_to_add_association '+', f,
:translations, class: 'btn btn-warning',
    data: { association_insertion_node:
'#translations',
association_insertion_method: :append } %>
</div>
Файл
app/views/problems/_test_fields.html.erb
<tr class="nested-fields">
  <td><%= f.input :num, wrapper: :table
%></td>
  <td><%= f.input :point, wrapper: :table
%></td>
  <td>
    <%= f.input :input, wrapper:
:table_file %>
    <%= f.input :input_text, as: :text,
wrapper: :table, wrapper_html: { class:
:'d-none' } %>
  </td>
  <td>
    <%= f.input :answer, wrapper:
:table_file %>
    <%= f.input :answer_text, as: :text,
wrapper: :table, wrapper_html: { class:
:'d-none' } %>
  </td>
  <td>
    <button type="button" class="btn btn-
sm btn-success" data-toggle="test-fields-
swap"><i class="fas fa-sync"></i></button>
  </td>
  <td>
    <%= link_to_remove_association f,
class: 'btn btn-sm btn-danger' do %>
      <i class="fas fa-times"></i>
    <% end %>
  </td>
</tr>
Файл app/views/problems/show.html.erb
<div class="page bg-transparent position-
relative">
  <div class="side-buttons">
    <%= link_to [:edit, resource], class:
'btn btn-primary mb-2' do %>
      <i class="far fa-edit"></i>
    <% end if policy(resource).edit? %>
    <%= link_to resource, method: :delete,
remote: true, class: 'btn btn-danger',
data: { confirm: t('confirm') } do %>
      <i class="fas fa-times"></i>
    <% end if policy(resource).destroy? %>
  </div>
  <%= content_tag :div, class: 'section
section--warning' do %>
    <%= t
'problems.show.missing_translation',
language: I18n.t(resource.language, scope:
'problems.show.languages') %>
  <% end if I18n.locale !=
resource.language %>
  <div class="section section--striped">
    <h2 class="section__title"><%=
resource.caption %></h2>
    <div class="section__content">
      <%= sanitize_for_problem
resource.text %>
    </div>
  </div>
  <div class="section section--striped">
    <h3><%= t
'problems.show.technical_text' %></h3>
    <div class="section__content">
      <%= sanitize_for_problem
resource.technical_text %>
    </div>
  </div>
  <%= content_tag :div, class: 'section
section--striped' do %>
    <h3><%= t 'problems.show.examples'
%></h3>
    <div class="row">
      <div class="col-md-6 col-12">
        <h4><%= t
'problems.show.example_input' %></h4>
      </div>
      <div class="col-md-6 col-12">
        <h4><%= t
'problems.show.example_output' %></h4>
      </div>
    </div>
    <% resource.examples.each do |example|
%>
      <div class="row">
        <div class="col-md-6 col-12">
```

```
          <pre class="problem-example"><%=
example.input %></pre>
        </div>
        <div class="col-md-6 col-12">
          <pre class="problem-example"><%=
example.answer %></pre>
        </div>
      </div>
    <% end %>
  <% end if resource.examples.any? %>
  <%= content_tag :div, class: 'section
section--striped section--with-icon' do %>
    <i class="fa fa-certificate" aria-
hidden="true"></i>
    <ol class="tags">
      <% resource.tags.each do |tag| %>
        <li><%= link_to tag.name, [tag,
:problems] %></li>
      <% end %>
    </ol>
  <% end if resource.tags.any? %>
  <div class="section section--striped
section--with-icon">
    <i class="far fa-address-book" aria-
hidden="true"></i>
    <span><%= resource.author %></span>
  </div>
  <%= content_tag :div, class: 'section
section--striped' do %>
    <%= render 'submissions/form', parent:
resource, resource: Submission.new %>
  <% end if current_user %>
  <div class="section section--striped">
    <%= link_to
t('problems.show.submissions'), [resource,
:submissions], class: 'btn btn-block btn-
link' %>
  </div>
</div>
Файл app/views/problems/edit.html.erb
<%= render 'form' %>
Файл
app/views/problems/_translation_fields.htm
l.erb
<div class="translation nested-fields">
  <div class="clearfix translation__top">
    <div class="row float-left">
      <div class="col-4">
        <%= f.input :language, collection:
language_select_options, include_blank:
false,
            wrapper: :input_group, prepend:
f.input_field(:default) %>
      </div>
      <div class="col-4">
        <%= f.input :caption %>
      </div>
      <div class="col-4">
        <%= f.input :author %>
      </div>
    </div>
    <div class="float-right">
      <%= link_to_remove_association f,
class: 'btn btn-danger' do %>
        <i class="fas fa-times"></i>
      <% end %>
    </div>
  </div>
  <%= f.input :text, as: :text,
:input_html => { :data => {:textarea =>
"ckeditor"} } %>
  <%= f.input :technical_text, as: :text,
:input_html => { :data => {:textarea =>
"ckeditor"} } %>
</div>
Файл app/views/problems/_form.html.erb
<div class="page row bg-light"
style="padding: 15px">
  <%= simple_form_for resource do |f| %>
    <div class="col-12" style="padding-
left: 0">
      <h1>
        <%= t 'problems.new.title' %>
        <%= f.submit class: 'btn btn-
primary float-right' %>
      </h1>
    </div>
    <ul class="nav nav-tabs">
      <li class="nav-item">
        <%= link_to
t('problems.new.tabs.basic'), '#basic-
tab', 'data-toggle': 'tab', class: 'nav-
link active' %>
      </li>
      <li class="nav-item">
        <%= link_to
t('problems.new.tabs.examples'),
'#examples-tab', 'data-toggle': 'tab',
class: 'nav-link' %>
      </li>
      <li class="nav-item">
        <%= link_to
t('problems.new.tabs.tests'), '#tests-
tab', 'data-toggle': 'tab', class: 'nav-
link' %>
      </li>
      <li class="nav-item">
        <%= link_to
t('problems.new.tabs.translations'),
'#translations-tab', 'data-toggle': 'tab',
class: 'nav-link' %>
      </li>
    </ul>
    <div class="tab-content">
      <%= render 'basic', f: f %>
      <%= render 'examples', f: f %>
      <%= render 'tests', f: f %>
      <%= render 'translations', f: f %>
    </div>
  <% end %>
</div>
Файл app/views/problems/_tests.html.erb
<div id="tests-tab" class="tab-pane fade">
  <table class="table table-custom">
    <thead>
      <tr>
        <th>Num</th>
        <th>Point</th>
        <th>Input</th>
        <th>Answer</th>
        <th style="width: 52px"></th>
        <th style="width: 52px"></th>
      </tr>
    </thead>
    <tbody id="tests">
      <%= f.simple_fields_for :tests do
|builder| %>
        <%= render 'test_fields', f:
builder %>
      <% end %>
    </tbody>
  </table>
  <%= link_to_add_association '+', f,
:tests, class: 'btn btn-warning',
    data: { association_insertion_node:
'#tests', association_insertion_method:
:append } %>
</div>
Файл app/views/problems/_basic.html.erb
<div id="basic-tab" class="tab-pane fade
show active">
  <div class="row">
    <div class="col-md-6">
      <%= f.input :memory_limit, wrapper:
:input_group, append: 'kb',
        input_html: { value:
resource.memory_limit || 32768 } %>
      <%= f.input :time_limit, wrapper:
:input_group, append: 'ms',
        input_html: { value:
resource.time_limit || 100 } %>
      <%= f.input :real_time_limit,
wrapper: :input_group, append: 'ms',
        input_html: { value:
resource.real_time_limit || 5000 } %>
      <%= f.input :private %>
    </div>
    <div class="col-md-6">
      <%= f.input :checker_compiler_id,
collection: checker_compilers,
include_blank: false %>
      <div class="form-group <%= 'form-
group-invalid' if
resource.errors[:checker_source].present?
%>">
        <%= f.label :checker_source,
class: 'form-control-label' %>
        <div class="input-group">
          <% if
resource.checker_source.attached? %>
            <div class="input-group-
prepend">
              <span class="input-group-
text"><%= link_to
t('problems.show.download'),
resource.checker_source %></span>
            </div>
          <% end %>
          <div class="custom-file">
            <%= f.input_field
:checker_source, class: 'custom-file-
input' %>
            <label class="custom-file-
label"><%= t 'problems.show.select_file'
%></label>
          </div>
        </div>
        <div class="invalid-feedback d-
block"><%= f.error :checker_source
%></div>
      </div>
      <%= f.input :tag_ids, as: :select,
collection: resource.tags, input_html: {
multiple: true, 'data-role': 'tags-select'
} %>
    </div>
  </div>
</div>
Файл app/views/problems/index.html.erb
<div class="page row bg-light">
  <div class="col-12 mb-4">
    <h1>
      <%= t 'problems.index.title' %>
      <%= link_to
t('problems.index.new_archive'), %i(new
archive),
        class: 'btn btn-primary float-
right' if policy(Archive).new? %>
      <%= link_to t('problems.index.new'),
%i(new problem), class: 'btn btn-primary
float-right',
        style: 'margin-right: 1rem' if
policy(Problem).new? %>
    </h1>
  </div>
  <div class="col-12">
    <table class="table table-custom">
      <thead class="thead-dark">
        <tr>
          <th><%= t
'problems.index.number' %></th>
          <th><%= t 'problems.index.name'
%></th>
          <th><%= t
'problems.index.difficulty' %></th>
        </tr>
      </thead>
      <tbody>
        <%= render collection.decorate %>
      </tbody>
    </table>
    <%= paginate collection %>
  </div>
</div>
Файл
app/views/problems/_example_fields.html.er
b
<tr class="nested-fields">
  <td><%= f.input :input, wrapper: :table,
as: :text %></td>
  <td><%= f.input :answer, wrapper:
:table, as: :text %></td>
  <td>
    <%= link_to_remove_association f,
class: 'btn btn-danger' do %>
      <i class="fas fa-times"></i>
    <% end %>
  </td>
</tr>
Файл app/views/problems/_examples.html.erb
<div id="examples-tab" class="tab-pane
fade">
  <table class="table table-custom">
    <thead>
      <tr>
        <th>Input</th>
        <th>Answer</th>
        <th style="width: 52px"></th>
      </tr>
    </thead>
    <tbody id="examples">
      <%= f.simple_fields_for :examples do
|builder| %>
        <%= render 'example_fields', f:
builder %>
      <% end %>
    </tbody>
  </table>
  <%= link_to_add_association '+', f,
:examples, class: 'btn btn-warning',
    data: { association_insertion_node:
'#examples', association_insertion_method:
:append } %>
</div>
Файл app/views/problems/new.html.erb
<%= render 'form' %>
Файл
app/views/memberships/_request_form.html.e
rb
<%= simple_form_for [parent, resource],
remote: true do |f| %>
  <%= f.hidden_field :type, value:
'request' %>
  <%= f.submit
t('membership.new.request'), class: 'btn
btn-primary btn-block' %>
<% end %>
Файл
app/views/memberships/_membership.html.erb
<tr>
  <td><%= membership.user_name %></td>
  <td><%= membership.group_name %></td>
  <td><%= membership.state.humanize
%></td>
  <td><%= l membership.created_at, format:
:long %></td>
  <td>
    <%= link_to membership, method:
:patch, remote: true, class: 'btn btn-sm
btn-success' do %>
      <i class="fas fa-check"></i>
    <% end if policy(membership).update?
%>
    <%= link_to membership, method:
:delete, remote: true, class: 'btn btn-sm
btn-danger' do %>
      <i class="fas fa-times"></i>
    <% end if policy(membership).destroy?
%>
  </td>
</tr>
Файл app/views/memberships/index.html.erb
<div class="page bg-white">
  <table class="table table-custom">
    <thead class="thead-dark">
      <th><%= t
'membership.index.user_name' %></th>
      <th><%= t
'membership.index.group_name' %></th>
      <th><%= t 'membership.index.state'
%></th>
```

```
      <th><%= t
'membership.index.created_at' %></th>
      <th></th>
    </thead>
    <tbody>
      <%= render collection.decorate %>
    </tbody>
  </table>
  <%= paginate collection %>
</div>
```

Файл app/views/memberships/new.html.erb

```
<div class="page bg-light">
  <%= simple_form_for [parent, resource],
remote: true do |f| %>
    <%= f.hidden_field :type, value:
'invite' %>
    <div class="form-group<%= ' form-
group-invalid' if
resource.errors[:user].present? %>">
      <label class="form-control-
label">User</label>
      <input type="text" data-role="user-
search" data-target="#membership_user_id"
value="<%= resource.decorate.user_name %>"
       class="form-control<%= ' is-
invalid' if
resource.errors[:user].present? %>">
      <%= f.full_error :user, class:
'invalid-feedback' %>
    </div>
    <%= f.hidden_field :user_id %>
    <%= f.submit
t('membership.new.invite'), class: 'btn
btn-primary btn-block' %>
  <% end %>
</div>
```

Файл app/views/groups/show.html.erb

```
<div class="page row bg-light">
  <div class="col-12">
    <h1><%= resource.name %></h1>
    <div class="alert alert-info">
      <%= resource.visibility_icon %>
      <%= t resource.visibility, scope:
'group.show.visibility' %>
    </div>
  </div>
  <div class="col-8">
    <h3><%= t 'group.show.description'
%></h3>
    <%= simple_format resource.description
%>
  </div>
  <% if current_user == resource.owner ||
resource.current_user_membership_state_acc
epted? %>
    <div class="col-4">
      <h3 class="text-center"><%= t
'group.show.members' %></h3>
      <div class="list-group mb-2">
        <div class="member-list">
          <%
resource.accepted_memberships.each do
|membership| %>
            <div class="list-group-item
member-list__item">
              <div class="d-flex flex-
column">
                <%= membership.user.name
%>
                <small class="text-
muted"><%= membership.user.username
%></small>
              </div>
              <%= link_to membership,
method: :delete, class: 'btn btn-sm btn-
danger' do %>
                <i class="fas fa-
times"></i>
              <% end if
policy(membership).destroy? %>
            </div>
          <% end %>
        </div>
        <div class="member-list-actions">
          <%= link_to
t('group.show.new_membership'), [:new,
resource, :membership],
            class: 'list-group-item btn
btn-outline-primary btn-sm btn-block' if
policy(resource).new_invite? %>
          <%= link_to
t('group.show.membership_list'),
[resource, :memberships],
            class: 'list-group-item btn
btn-outline-primary btn-sm btn-block mt-0'
if policy(resource).index_memberships? %>
        </div>
      </div>
    </div>
    <div class="col-12">
      <%= link_to t('group.show.edit'),
[:edit, resource], class: 'btn btn-primary
btn-block' if policy(resource).edit? %>
      <%= link_to
t('group.show.create_sharing'), [:new,
resource, :sharing], class: 'btn btn-
primary btn-block' \
        if policy(resource).new_sharing?
%>
      <%= link_to
t('group.show.submission_list'),
[resource, :submissions], class: 'btn btn-
link btn-block' %>
      <%= link_to
t('group.show.problem_list'), [resource,
:problems], class: 'btn btn-link btn-
block' %>
      <%= link_to
t('group.show.standings'), [resource,
:standing], class: 'btn btn-link btn-
block' %>
    </div>
  <% elsif
resource.current_user_membership_state_req
uested? ||
resource.current_user_membership_state_inv
ited? %>
    <div class="col-12">
      <div class="alert alert-warning">
        <%= t 'group.show.pending_member'
%>
        <%= link_to
resource.current_user_membership, method:
:delete, remote: true, class: 'badge
badge-danger ml-3' do %>
          <i class="fas fa-times"></i>
        <% end %>
      </div>
    </div>
  <% else %>
    <div class="col-12">
      <div class="alert alert-warning"><%=
t 'group.show.not_a_member' %></div>
      <%= render
'memberships/request_form', parent:
resource, resource: Membership.new %>
    </div>
  <% end %>
</div>
```

Файл app/views/groups/edit.html.erb

```
<div class="page row bg-light">
  <div class="col-12">
    <h1><%= t 'group.edit.title' %></h1>
  </div>
  <div class="col-12">
    <%= render 'form' %>
  </div>
</div>
```

Файл app/views/groups/_group.html.erb

```
<tr>
  <td><%= group.owner_name %></td>
  <td>
    <% if policy(group).show? %>
      <%= link_to group.name, group %>
    <% else %>
      <%= group.name %>
    <% end %>
  </td>
  <td><%= group.visibility.humanize
%></td>
  <td><%= l group.created_at, format:
:long %></td>
</tr>
```

Файл app/views/groups/_form.html.erb

```
<%= simple_form_for resource do |f| %>
  <%= f.input :name %>
  <%= f.input :visibility, as: :select,
collection: Group.visibilities.keys,
include_blank: false %>
  <%= f.input :description %>
  <%= f.submit class: 'btn btn-primary' %>
<% end %>
```

Файл app/views/groups/index.html.erb

```
<div class="page row bg-light">
  <div class="col-12 mb-4">
    <h1>
      <%= t 'group.index.title' %>
      <%= link_to t('group.index.new'),
%i(new group), class: 'btn btn-primary
float-right',
          style: 'margin-right: 1rem' if
policy(Group).new? %>
    </h1>
  </div>
  <div class="col-12">
    <table class="table table-custom">
      <thead class="thead-dark">
        <tr>
          <th><%= t 'group.index.owner'
%></th>
          <th><%= t 'group.index.name'
%></th>
          <th><%= t
'group.index.visibility' %></th>
          <th><%= t
'group.index.created_at' %></th>
        </tr>
      </thead>
      <tbody>
        <%= render collection.decorate %>
      </tbody>
    </table>
    <%= paginate collection %>
  </div>
</div>
```

Файл app/views/groups/new.html.erb

```
<div class="page row bg-light">
  <div class="col-12">
    <h1><%= t 'group.new.title' %></h1>
  </div>
  <div class="col-12">
    <%= render 'form' %>
  </div>
</div>
```

Файл app/views/layouts/mailer.html.erb

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type"
content="text/html; charset=utf-8" />
    <style>
      /* Email styles need to be inline */
    </style>
  </head>
  <body>
    <%= yield %>
  </body>
</html>
```

Файл app/views/layouts/application.html.erb

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <meta name="theme-color"
content="#8c3f01">
    <meta http-equiv="X-UA-Compatible"
content="IE=edge">
    <title>CodeLabs</title>
    <%= csrf_meta_tags %>
    <%= csp_meta_tag %>
    <meta name="turbolinks-cache-control"
content="no-cache">
    <%= stylesheet_link_tag :application,
media: :all, 'data-turbolinks-track':
:reload %>
    <%= stylesheet_link_tag
'https://use.fontawesome.com/releases/v5.0
.13/css/all.css',
      media: :all, crossorigin: :anonymous
%>
    <%= javascript_include_tag
'https://cdnjs.cloudflare.com/ajax/libs/ma
thjax/2.7.5/MathJax.js?config=TeX-MML-
AM_CHTML' %>
    <%= javascript_include_tag :ckeditor,
'data-turbolinks-track': :reload %>
    <%= javascript_include_tag
:application, 'data-turbolinks-track':
:reload %>
  </head>
  <body data-flash="<%= flash.to_json %>">
    <%= render 'widgets/header' %>
    <div class="container">
      <%= yield %>
    </div>
    <%= render 'widgets/footer' %>
  </body>
</html>
```

Файл app/views/passwords/edit.html.erb

```
<div class="page form-page bg-dark">
  <%= simple_form_for resource, url:
:password, remote: true do |f| %>
    <%= hidden_field_tag :token,
params[:token] %>
    <%= f.input :password %>
    <%= f.input :password_confirmation %>
    <%= f.submit class: 'btn btn-primary'
%>
  <% end %>
</div>
```

Файл app/views/submissions/_submission.html.erb

```
<tr>
  <td>
    <% if policy(submission).show? %>
      <%= link_to submission.id,
submission %>
    <% else %>
      <%= submission.id %>
    <% end %>
  </td>
  <td>
    <% if policy(submission.problem).show?
%>
      <%= link_to
submission.problem_caption,
submission.problem %>
    <% else %>
      <%= submission.problem_caption %>
    <% end %>
  </td>
  <td><%= submission.user_username %></td>
  <td><%= submission.compiler.name %></td>
  <td><%= submission.state %></td>
</tr>
```

Файл app/views/submissions/show.html.erb

```
<div class="page bg-white position-
relative">
  <div class="side-buttons">
    <%= link_to [resource, :retest],
class: 'btn btn-warning mb-2',
      data: { confirm: t('confirm'),
remote: true, method: :post } do %>
      <i class="fas fa-sync"></i>
    <% end if
policy(Submission::Retest.new
resource).new? %>
    <%= link_to resource, method: :delete,
remote: true, class: 'btn btn-danger',
data: { confirm: t('confirm') } do %>
      <i class="fas fa-times"></i>
    <% end if policy(resource).destroy? %>
```

```
  </div>
  <div>
    <h1><%= t 'submissions.show.title',
id: resource.id %></h1>
    <p><%= t 'submissions.show.problem'
%>: <b><%= link_to
resource.problem_caption, resource.problem
%></b></p>
    <p><%= t 'submissions.show.user' %>:
<b><%= resource.user_username %></b></p>
    <p><%= t 'submissions.show.state' %>:
<b><%= resource.state %></b></p>
    <p><%= t 'submissions.show.time_limit'
%>: <b><%= resource.time_limit %></b>
ms</p>
    <p><%= t
'submissions.show.memory_limit' %>: <b><%=
resource.memory_limit %></b> kb</p>
  </div>
  <div>
    <ul class="nav nav-tabs">
      <li class="nav-item">
        <%= link_to
t('submissions.show.results'), '#results',
class: 'nav-link active', 'data-toggle':
'tab' %>
      </li>
      <li class="nav-item">
        <%= link_to
t('submissions.show.compiler_log'),
'#compiler-log', class: 'nav-link', 'data-
toggle': 'tab' %>
      </li>
      <li class="nav-item">
        <%= link_to
t('submissions.show.source'), '#source',
class: 'nav-link', 'data-toggle': 'tab' %>
      </li>
    </ul>
    <div class="tab-content">
      <div class="tab-pane fade show
active" id="results">
        <% if resource.test_state_failed?
%>
          <p><%= t
'submissions.show.test_state_failed'
%></p>
        <% elsif
resource.test_result_compiler_error? %>
          <p><%= t
'submissions.show.test_result_compiler_err
or' %></p>
        <% else %>
          <table class="table table-
custom">
            <thead class="thead-dark">
              <tr>
                <th><%= t
'results.index.num' %></th>
                <th><%= t
'results.index.status' %></th>
                <th><%= t
'results.index.time' %></th>
                <th><%= t
'results.index.memory' %></th>
                <th></th>
              </tr>
            </thead>
            <tbody>
              <%= render
resource.results.includes(:test).order('te
sts.num') %>
            </tbody>
          </table>
        <% end %>
      </div>
      <div class="tab-pane fade"
id="compiler-log"><%= render
'compiler_log' %></div>
      <div class="tab-pane fade"
id="source"><%= render 'source' %></div>
    </div>
  </div>
</div>
Файл app/views/submissions/_form.html.erb
<%= simple_form_for [parent, resource] do
|f| %>
  <%= f.input :compiler_id, collection:
visible_compilers, include_blank: false %>
  <%= f.input :source %>
  <%= f.submit class: 'btn btn-primary' %>
<% end %>
Файл
app/views/submissions/_source.html.erb
<pre><%= resource.source.download.encode
'UTF-8', invalid: :replace, undef:
:replace %></pre>
Файл app/views/submissions/index.html.erb
<div class="page row bg-light">
  <div class="col-12 mb-4">
    <h1><%= t 'submissions.index.title'
%></h1>
  </div>
  <%#
  <div class="col-12">
    <h3>Сортировать по </h3>
    <div class="row" style="margin-bottom:
15px;">
      <div class="col-12 col-lg-6 col-md-6
col-sm-6">
        Пользователю:
        <input id="sortByUser"
class="form-control" type="text">
      </div>
      <div class="col-12 col-lg-6 col-md-6
col-sm-6">
        Задаче:
        <input id="sortByTask"
class="form-control" type="text">
      </div>
      <div class="col-12 col-lg-6 col-md-6
col-sm-6">
        Статусу:
        <select id="sortByStatus"
class="form-control">
          <option
value="fail">Failed</option>
          <option value="ok">Ok</option>
          <option value="error">Compiler
error</option>
        </select>
      </div>
      <div class="col-12 col-lg-6 col-md-6
col-sm-6">
        Группе:
        <input id="sortByGroup"
class="form-control" type="text">
      </div>
    </div>
  </div>
  %>
  <div class="col-12">
    <table class="table table-custom">
      <thead class="thead-dark">
        <tr>
          <th><%= t 'submissions.index.id'
%></th>
          <th><%= t
'submissions.index.problem' %></th>
          <th><%= t
'submissions.index.user' %></th>
          <th><%= t
'submissions.index.compiler' %></th>
          <th><%= t
'submissions.index.state' %></th>
        </tr>
      </thead>
      <tbody>
        <%= render collection.decorate %>
      </tbody>
    </table>
    <%= paginate collection %>
  </div>
</div>
Файл app/views/submissions/new.html.erb
<div class="page row bg-light">
  <div class="col-12">
    <h1><%= t 'submissions.new.title',
problem_caption: parent.decorate.caption
%></h1>
  </div>
  <div class="col-12">
    <%= render 'form' %>
  </div>
</div>
Файл
app/views/submissions/_compiler_log.html.e
rb
<% resource.logs.each do |log| %>
  <%= content_tag(:pre) { log.data } if
policy(log).show? %>
<% end %>
Файл app/views/home/show.html.erb
<div class="page row">
  <div class="col-md-7 col-7 home-section
bg-transparent">
    <div class="home-section__title">
      <h1><%= t
'home.show.sections.first.title' %></h1>
      <h4><%= t
'home.show.sections.first.subtitle'
%></h4>
    </div>
    <div class="home-section__content">
      <h4><i class="fa fa-check"></i> <%=
t 'home.show.sections.first.list.0'
%></h4>
      <h4><i class="fa fa-check"></i> <%=
t 'home.show.sections.first.list.1'
%></h4>
      <h4><i class="fa fa-check"></i> <%=
t 'home.show.sections.first.list.2'
%></h4>
      <h4><i class="fa fa-check"></i> <%=
t 'home.show.sections.first.list.3'
%></h4>
    </div>
  </div>
  <div class="col-md-5 col-5 home-section
bg-dark">
    <h1 class="home-section__title"><%= t
'home.show.sections.second.title' %></h1>
    <%= render 'users/form', resource:
User.new %>
  </div>
</div>
Файл app/views/avatars/_form.html.erb
<div class="image-input">
  <input type="file" hidden>
  <div class="image-input__prompt">
    <i class="image-input__prompt-icon fas
fa-upload"></i>
  </div>
  <% if resource.attached? %>
    <%= image_tag resource.url, class:
'avatar' %>
  <% else %>
    <%= image_tag 'placeholder.png',
class: 'avatar' %>
  <% end %>
</div>
Файл app/views/archives/new.html.erb
<div class="page row bg-light">
  <div class="col-12">
    <h1><%= t 'archives.new.title' %></h1>
  </div>
  <div class="col-md-6 col-12">
    <%= simple_form_for resource, remote:
true, authenticity_token: true do |f| %>
      <%= f.hidden_field :channel_id %>
      <%= f.input :file, as: :file,
input_html: { accept: 'application/zip' }
%>
      <%= f.submit class: 'btn btn-
primary' %>
    <% end %>
  </div>
  <div id="process-problem-archive-log"
class="col-md-6 col-12" data-channel-
id="<%= resource.channel_id %>"></div>
</div>
Файл app/views/results/_result.html.erb
<tr>
  <td><%= result.test_num %></td>
  <td><%= result.status.humanize %></td>
  <td><%= result.time %> ms</td>
  <td><%= result.memory %> kb</td>
  <td>
    <% if result.log? %>
      <button class="btn btn-primary btn-
xs float-right" data-toggle="collapse"
data-target="#log-<%= result.test_num %>">
        +
      </button>
      <div id="log-<%= result.test_num %>"
aria-expanded="false"
class="collapse"><pre class="result-
log"><%= result.log %></pre></div>
    <% end %>
  </td>
</tr>
Файл app/views/sharings/new.html.erb
<div class="page row bg-light">
  <div class="col-12">
    <h1><%= t 'sharing.new.title' %></h1>
  </div>
  <div class="col-12">
    <%= simple_form_for [parent, resource]
do |f| %>
      <%= f.association :problem,
collection:
current_user.problems.decorate.map { |p|
[p.caption, p.id] } %>
      <%= f.submit class: 'btn btn-primary
btn-block' %>
    <% end %>
  </div>
</div>
Файл app/views/profiles/show.html.erb
<div class="page row bg-light">
  <div class="col-12">
    <h1><%= t 'profile.show.title' %></h1>
  </div>
  <%= render 'form' %>
  <div class="offset-md-3 col-md-9 col-
12">
    <%= link_to
t('profile.show.submissions'),
[current_user, :submissions], class: 'btn
btn-block btn-link' %>
    <%= link_to
t('profile.show.problems'), [current_user,
:problems], class: 'btn btn-block btn-
link' \
      if policy(Problem).new? %>
    <%= link_to
t('profile.show.membership_list'),
:memberships, class: 'btn btn-block btn-
link' %>
    <%= link_to
t('profile.show.ask_for_confirmation'),
:confirmation_requests, class: 'btn btn-
block btn-primary',
      method: :post if
policy(ConfirmationRequest).new? %>
  </div>
</div>
Файл app/views/profiles/_form.html.erb
<div class="col-md-3 col-12">
  <%= render 'avatars/form', resource:
Avatar.new(user: current_user) %>
</div>
<div class="col-md-9 col-12">
  <%= simple_form_for resource, url:
:profile, wrapper: :horizontal_form,
remote: true do |f| %>
    <%= f.input :name %>
    <%= f.input :username %>
    <%= f.input :skills, input_html: {
'data-role': :tagsinput } %>
    <%= f.input :password %>
    <%= f.input :password_confirmation %>
    <%= f.input :city %>
```

```
    <%= f.input :institution %>
    <%= f.input :score, readonly: true,
as: :string %>
    <%= f.input :created_at, readonly:
true, as: :string, input_html: { value:
l(resource.created_at, format: :long) } %>
    <%= f.submit class: 'btn btn-block
btn-primary' %>
  <% end %>
</div>
```

Файл app/views/sessions/_form.html.erb

```
<%= simple_form_for resource, url:
:session, remote: true do |f| %>
  <%= f.input :email %>
  <%= f.input :password %>
  <div class="form-group">
    <%= recaptcha_tags theme: :dark %>
  </div>
  <%= f.submit class: 'btn btn-primary' %>
  <%= link_to t('.reset_password'), %i[new
password_recovery], class: 'btn btn-link
float-right' %>
<% end %>
```

Файл app/views/sessions/new.html.erb

```
<div class="page form-page bg-dark">
  <%= render 'form' %>
</div>
```

Файл app/views/widgets/_snow.html.erb

```
<div class="winter-is-coming">
  <div class="snow snow--near"></div>
  <div class="snow snow--near snow--
alt"></div>
  <div class="snow snow--mid"></div>
  <div class="snow snow--mid snow--
alt"></div>
  <div class="snow snow--far"></div>
  <div class="snow snow--far snow--
alt"></div>
</div>
```

Файл app/views/widgets/_footer.html.erb

```
<div class="footer">
  ¾ VTL + ¾ KNU &copy; 2017-<%=
Time.zone.now.year %>
  <br />
  <span class="footer--revision"><%=
REVISION %></span>
</div>
```

Файл app/views/widgets/_header.html.erb

```
<div class="header">
  <%= link_to current_user ? :profile :
:root do %>
    <%= image_tag 'codelabs.png' %>
  <% end %>
  <div class="header-section">
    <div class="header-section__item
dropdown show">
      <a data-toggle="dropdown"
class="dropdown-toggle"><%= t
'header.problems' %></a>
      <div class="dropdown-menu dropdown-
menu-right">
        <%= link_to
t('header.problems_all'), :problems,
class: 'dropdown-item' if
policy(Problem).index? %>
        <%= link_to
t('header.problems_tags'), :tags, class:
'dropdown-item' if policy(Tag).index? %>
      </div>
    </div>
    <%= content_tag :div, class: 'header-
section__item' do %>
      <%= link_to t('header.submissions'),
:submissions %>
    <% end if policy(Submission).index? %>
    <%= content_tag :div, class: 'header-
section__item' do %>
      <%= link_to t('header.groups'),
:groups %>
    <% end if policy(Group).index? %>
    <%= content_tag :div, class: 'header-
section__item dropdown show' do %>
      <a data-toggle="dropdown"
class="dropdown-toggle"><%= t
'header.administration' %></a>
      <div class="dropdown-menu dropdown-
menu-right">
        <%= link_to
t('header.confirmation_requests'),
:confirmation_requests, class: 'dropdown-
item' \
          if
policy(ConfirmationRequest).index? %>
        <%= link_to t('header.compilers'),
:compilers, class: 'dropdown-item' if
policy(Compiler).index? %>
        <%= link_to t('header.workers'),
:workers, class: 'dropdown-item' if
policy(Worker).index? %>
      </div>
    <% end if current_user&.administrator?
%>
  </div>
  <div class="header-section">
    <div class="header-section__item">
      <% if current_user %>
        <%= link_to t('header.signout'),
:session, method: :delete, remote: true %>
      <% else %>
        <%= link_to t('header.signin'),
%i(new session) %>
      <% end %>
    </div>
    <div class="header-section__item
dropdown show">
      <a data-toggle="dropdown"
class="dropdown-toggle"><%= I18n.locale
%></a>
      <div class="dropdown-menu dropdown-
menu-right">
        <% I18n.available_locales.each do
|locale| %>
          <%= link_to t(:language, locale:
locale), { language: locale }, class:
'dropdown-item' %>
        <% end %>
      </div>
    </div>
  </div>
</div>
```

Файл app/views/tags/_tag.html.erb

```
<tr>
  <td><%= link_to tag.name, [tag,
:problems] %></td>
  <td><%= tag.problems_count %></td>
</tr>
```

Файл app/views/tags/index.html.erb

```
<div class="page row bg-light">
  <div class="col-12 mb-4">
    <h1><%= t 'tags.index.title' %></h1>
  </div>
  <div class="col-12">
    <table class="table table-custom">
      <thead class="thead-dark">
        <tr>
          <th><%= t 'tags.index.name'
%></th>
          <th><%= t
'tags.index.problems_count' %></th>
        </tr>
      </thead>
      <tbody>
        <%= render collection %>
      </tbody>
    </table>
    <%= paginate collection %>
  </div>
</div>
```

Файл app/views/standings/show.html.erb

```
<div class="page">
  <table class="table table-custom">
    <thead class="thead-dark">
      <tr>
        <th>User</th>
        <%
resource.problems.zip('A'..).each do
|problem, letter| %>
          <th data-toggle="tooltip"
title="<%= problem.caption %>">
            <%= link_to letter, problem,
style: 'color: inherit; text-decoration:
none' %>
          </th>
        <% end %>
      </tr>
    </thead>
    <tbody>
      <% resource.accepted_users.each do
|user| %>
        <tr>
          <td><%= user.name %></td>
          <% resource.problems.each do
|problem, name| %>
            <td><%= StandingRedisStore.get
user.id, problem.id %></td>
          <% end %>
        </tr>
      <% end %>
    </tbody>
  </table>
</div>
```

Файл app/views/confirmation_requests/_confirmation_request.html.erb

```
<tr>
  <td><%= confirmation_request.user_name
%></td>
  <td><%=
confirmation_request.user_username %></td>
  <td><%=
confirmation_request.status.humanize
%></td>
  <td><%= l
confirmation_request.created_at, format:
:long %></td>
  <td>
    <%= link_to [confirmation_request,
:accept], method: :post, class: 'btn btn-
sm btn-success' do %>
      <i class="fas fa-check"></i>
    <% end if
policy(ConfirmationRequest::Accept.new
confirmation_request).create? %>
    <%= link_to [confirmation_request,
:reject], method: :post, class: 'btn btn-
sm btn-danger' do %>
      <i class="fas fa-times"></i>
    <% end if
policy(ConfirmationRequest::Reject.new
confirmation_request).create? %>
  </td>
</tr>
```

Файл app/views/confirmation_requests/index.html.erb

```
<div class="page row bg-light">
  <div class="col-12 mb-4">
    <h1><%= t
'confirmation_requests.index.title'
%></h1>
  </div>
  <div class="col-12">
    <table class="table table-custom">
      <thead class="thead-dark">
        <tr>
          <th><%= t
'confirmation_requests.index.user_name'
%></th>
          <th><%= t
'confirmation_requests.index.user_username
' %></th>
          <th><%= t
'confirmation_requests.index.status'
%></th>
          <th><%= t
'confirmation_requests.index.created_at'
%></th>
          <th></th>
        </tr>
      </thead>
      <tbody>
        <%= render partial:
'confirmation_request', collection:
collection %>
      </tbody>
    </table>
    <%= paginate collection %>
  </div>
</div>
```

Файл app/views/user_mailer/email.html.erb

```
<p>Hello <%= @user.username %>.</p>
<p>
  Thank you for your interest in CodeLabs.
You have been successfully registered in
the system. You can follow
  <%= link_to 'this link', %i[new session]
%> to log in.
</p>
<p>CodeLabs Team</p>
```

Файл app/views/users/_form.html.erb

```
<%= simple_form_for resource, remote: true
do |f| %>
  <%= f.input :email %>
  <%= f.input :username %>
  <%= f.input :password, required: true %>
  <%= f.input :password_confirmation,
required: true %>
  <div class="form-group">
    <%= recaptcha_tags theme: :dark %>
  </div>
  <%= f.submit class: 'btn btn-primary' %>
<% end %>
```

Файл app/views/users/new.html.erb

```
<div class="page form-page bg-dark">
  <%= render 'form' %>
</div>
```

Файл app/views/password_recovery_mailer/email.html.erb

```
<p>Hello <%= @user.username %>.</p>
<p>
  You received this email because someone
requested a new password for your account.
You can follow
  <%= link_to 'this link', [:edit,
:password, token:
@user.password_recovery_token] %> to reset
your
  password.
</p>
<p>If you did not request a new password
do not take any action and ignore this
letter.</p>
<p>CodeLabs Team</p>
```

Файл app/channels/application_cable/connection.rb

```
module ApplicationCable
  class Connection <
ActionCable::Connection::Base
    identified_by :current_user
    def connect
      reject_unauthorized_connection
unless current_user
    end
    def current_user
      return unless
cookies.encrypted[:auth_token]
      @current_user ||=
User.joins(:auth_tokens).find_by(auth_toke
ns: { id: cookies.encrypted[:auth_token]
})
    end
  end
end
```

Файл app/channels/application_cable/channel.rb

```
module ApplicationCable
  class Channel <
ActionCable::Channel::Base
  end
end
```

```
Файл
app/channels/process_problem_archive_chann
el.rb
class ProcessProblemArchiveChannel <
ApplicationCable::Channel
  def subscribed
    stream_from "ProcessProblemArchive:#{
params[:id] }"
  end
end
Файл app/mailers/application_mailer.rb
class ApplicationMailer <
ActionMailer::Base
  default from: ENV['SMTP_FROM']
  layout 'mailer'
end
Файл app/mailers/user_mailer.rb
class UserMailer < ApplicationMailer
  def email user
    @user = user
    mail subject: 'Successful registration
at CodeLabs', to: user.email
  end
end
Файл
app/mailers/password_recovery_mailer.rb
class PasswordRecoveryMailer <
ApplicationMailer
  def email user
    @user = user
    mail subject: 'Password recovery at
CodeLabs', to: user.email
  end
end
Файл app/validators/byte_size_validator.rb
class ByteSizeValidator <
ActiveModel::EachValidator
  def validate_each record, attribute,
value
    return unless value.attached?
    record.errors.add attribute,
:too_long, count: options[:maximum] if
value.byte_size > options[:maximum]
  end
end
Файл
app/controllers/submissions_controller.rb
class SubmissionsController <
ApplicationController
  skip_before_action :authenticate!, only:
:index
  def create
    render :new and return unless
resource.save
    redirect_to resource
  end
  def destroy
    resource.destroy
    redirect_to :submissions
  end
  private
  def collection
    @collection ||=
submissions.includes(:compiler, :user,
problem: :user).order(created_at:
:desc).page(params[:page])
  end
  def submissions
    parent ? parent.submissions :
Submission.all
  end
  def parent
    @parent ||= \
      case
      when params[:group_id]
        Group.find params[:group_id]
      when params[:problem_id]
        Problem.find params[:problem_id]
      when params[:user_id]
        User.find params[:user_id]
      end
  end
  def resource
    @resource ||=
Submission.find(params[:id]).decorate
  end
  def resource_params

params.require(:submission).permit(:compil
er_id, :source).merge(user: current_user)
  end
  def build_resource
    @resource = parent.submissions.new
resource_params
  end
end
Файл
app/controllers/passwords_controller.rb
class PasswordsController <
ApplicationController
  skip_before_action :authenticate!, only:
%i[edit update]
  def update
    if resource.update resource_params
      flash[:success] = I18n.t
'flash.save.success'
      redirect_to %i[new session]
    else
      flash.now[:error] = I18n.t
'flash.save.failure'
      render :edit, turbolinks: true
    end
  end
  private
  def resource
    @resource ||= User.find_by
password_recovery_token: params[:token] if
params[:token].present?
  end
  def resource_params

params.require(:user).permit(:password,
:password_confirmation).merge(password_rec
overy_token: nil)
  end
  def authorize_resource
    authorize resource, policy_class:
PasswordPolicy
  end
end
Файл
app/controllers/compilers_controller.rb
class CompilersController <
ApplicationController
  def create
    if resource.save
      flash[:success] = I18n.t
'flash.save.success'
      redirect_to :compilers
    else
      flash.now[:error] = I18n.t
'flash.save.failure'
      render :new, turbolinks: true
    end
  end
  def update
    if resource.update resource_params
      flash.now[:success] = I18n.t
'flash.save.success'
    else
      flash.now[:error] = I18n.t
'flash.save.failure'
    end
    render :edit, turbolinks: true
  end
  def destroy
    resource.destroy
    redirect_to :compilers
  end
  private
  def collection
    @collection ||=
Compiler.order(:id).page(params[:page])
  end
  def resource
    @resource ||= Compiler.find
params[:id]
  end
  def resource_params

params.require(:compiler).permit(:name,
:version, :time_a, :time_b, :memory_a,
:memory_b, :status)
  end
  def initialize_resource
    @resource = Compiler.new
  end
  def build_resource
    @resource = Compiler.new
resource_params
  end
end
Файл app/controllers/avatars_controller.rb
class AvatarsController <
ApplicationController
  def create
    render :errors, status: 422 and return
unless resource.save
  end
  def destroy
    resource.destroy
    head 204
  end
  private
  def resource
    @resource ||= build_resource
  end
  def resource_params

params.require(:avatar).permit(:file).merg
e(user: current_user)
  end
  def build_resource
    @resource = Avatar.new resource_params
  end
end
Файл
app/controllers/profiles_controller.rb
class ProfilesController <
ApplicationController
  def update
    if resource.update resource_params
      flash.now[:success] = I18n.t
'flash.save.success'
    else
      flash.now[:error] = I18n.t
'flash.save.failure'
    end
    render :show, turbolinks: true
  end
  private
  alias_method :resource, :current_user
  def resource_params

params.require(:user).permit(:username,
:name, :password, :password_confirmation,
:skills, :city, :institution)
  end
end
Файл
app/controllers/problems_controller.rb
class ProblemsController <
ApplicationController
  skip_before_action :authenticate!, only:
%i[index show]
  def create
    if resource.save
      flash[:success] = I18n.t
'flash.save.success'
      redirect_to resource
    else
      flash.now[:error] = I18n.t
'flash.save.failure'
      render :new
    end
  end
  def update
    if resource.update resource_params
      flash[:success] = I18n.t
'flash.save.success'
      redirect_to resource
    else
      flash.now[:error] = I18n.t
'flash.save.failure'
      render :edit
    end
  end
  def destroy
    resource.destroy
    redirect_to :problems
  end
  private
  def collection
    @collection ||=
problems.includes(:user).order(:id).page(p
arams[:page])
  end
  def problems
    parent ? parent.problems : Problem.all
  end
  def parent
    @parent ||= \
      case
      when params[:tag_id]
        Tag.find params[:tag_id]
      when params[:user_id]
        User.find params[:user_id]
      when params[:group_id]
        Group.find params[:group_id]
      end
  end
  def resource
    @resource ||=
Problem.find(params[:id]).decorate
  end
  def resource_params
    params.require(:problem).permit \
      :memory_limit, :time_limit,
:real_time_limit, :checker_compiler_id,
:checker_source, :private, tag_ids: [],
      examples_attributes: %i[id input
answer _destroy],
      tests_attributes: %i[id num point
input answer _destroy],
      translations_attributes: %i[id
language caption author text
technical_text default _destroy]
  end
  def initialize_resource
    @resource = current_user.problems.new
  end
  def build_resource
    @resource = current_user.problems.new
resource_params
  end
end
Файл
app/controllers/standings_controller.rb
class StandingsController <
ApplicationController
  private
  def resource
    @resource ||=
Group.find(params[:group_id]).decorate
  end
end
Файл
app/controllers/submission/retests_control
ler.rb
class Submission::RetestsController <
ApplicationController
  def create
    flash[:error] = I18n.t
'flash.save.failure' unless resource.save
    redirect_to parent
  end
  private
  attr_reader :resource
  def parent
    @parent ||= Submission.find
params[:submission_id]
  end
  def build_resource
```

```
    @resource = Submission::Retest.new
parent
  end
end
Файл
app/controllers/confirmation_request/accep
ts_controller.rb
class
ConfirmationRequest::AcceptsController <
ApplicationController
  def create
    flash[:error] = I18n.t
'flash.save.failure' unless resource.save
    redirect_back fallback_location:
:confirmation_requests, allow_other_host:
false
  end
  private
  attr_reader :resource
  def parent
    @parent ||= ConfirmationRequest.find
params[:confirmation_request_id]
  end
  def build_resource
    @resource =
ConfirmationRequest::Accept.new parent
  end
end
Файл
app/controllers/confirmation_request/rejec
ts_controller.rb
class
ConfirmationRequest::RejectsController <
ApplicationController
  def create
    flash[:error] = I18n.t
'flash.save.failure' unless resource.save
    redirect_back fallback_location:
:confirmation_requests, allow_other_host:
false
  end
  private
  attr_reader :resource
  def parent
    @parent ||= ConfirmationRequest.find
params[:confirmation_request_id]
  end
  def build_resource
    @resource =
ConfirmationRequest::Reject.new parent
  end
end
Файл
app/controllers/password_recoveries_contro
ller.rb
class PasswordRecoveriesController <
ApplicationController
  attr_reader :resource
  skip_before_action :authenticate!, only:
%i[new create]
  def create
    if verify_recaptcha
      resource.save
      render :create, turbolinks: true
    else
      render :new, turbolinks: true
    end
  end
  private
  def resource_params

params.require(:password_recovery).permit(
:email)
  end
  def build_resource
    @resource = PasswordRecovery.new
resource_params
  end
  def initialize_resource
    @resource = PasswordRecovery.new
  end
end
Файл
app/controllers/application_controller.rb
class ApplicationController <
ActionController::Base
  include Pundit
  before_action :authenticate!,
:set_locale
  before_action :initialize_resource,
only: :new
  before_action :build_resource, only:
:create
  before_action :authorize_resource,
except: :index
  before_action :authorize_collection,
only: :index
  helper_method :current_user, :parent,
:collection, :resource
  rescue_from Pundit::NotAuthorizedError
do
    flash[:error] = I18n.t
'flash.not_authorized'
    redirect_back fallback_location:
:root, allow_other_host: false
  end
  private
  def current_user
    return unless
cookies.encrypted[:auth_token]
    @current_user ||=
User.joins(:auth_tokens).find_by(auth_toke
ns: { id: cookies.encrypted[:auth_token]
})
  end
  def authenticate!
    unless current_user
      session[:redirect] =
request.fullpath
      redirect_to %i[new session]
    end
  end
  def authorize_resource
    authorize resource
  end
  def authorize_collection
    authorize collection
  end
  def set_locale
    I18n.locale =
I18n.locale_available?(params[:language])
? params[:language] : I18n.default_locale
  end
  def default_url_options
    { language: I18n.locale }
  end
end
Файл app/controllers/tags_controller.rb
class TagsController <
ApplicationController
  skip_before_action :authenticate!, only:
:index
  private
  def collection
    @collection ||=
Tag.order(problems_count:
:desc).page(params[:page])
  end
end
Файл app/controllers/users_controller.rb
class UsersController <
ApplicationController
  skip_before_action :authenticate!, only:
%i[new create]
  def create
    render :new, turbolinks: true and
return unless verify_recaptcha &&
resource.save
    redirect_to %i[new session]
  end
  private
  attr_reader :resource
  def collection
    @collection ||=
UserSearcher.search(User,
params).page(params[:page])
  end
  def resource_params

params.require(:user).permit(:username,
:email, :password, :password_confirmation)
  end
  def initialize_resource
    @resource = User.new
  end
  def build_resource
    @resource = User.new resource_params
  end
end
Файл app/controllers/home_controller.rb
class HomeController <
ApplicationController
  skip_before_action :authenticate!,
:authorize_resource
  def show
    redirect_to :profile if current_user
  end
end
Файл app/controllers/groups_controller.rb
class GroupsController <
ApplicationController
  def create
    render :new and return unless
resource.save
    redirect_to resource
  end
  def update
    render :edit and return unless
resource.update resource_params
    redirect_to resource
  end
  def destroy
    resource.destroy
    redirect_to :groups
  end
  private
  def resource
    @resource ||=
Group.find(params[:id]).decorate
  end
  def collection
    @collection ||=
Group.includes(:owner).order(:name).page(p
arams[:page])
  end
  def resource_params
    params.require(:group).permit(:name,
:visibility, :description)
  end
  def initialize_resource
    @resource =
current_user.owned_groups.new
  end
  def build_resource
    @resource =
current_user.owned_groups.new
resource_params
  end
end
Файл
app/controllers/archives_controller.rb
class ArchivesController <
ApplicationController
  def create
    render :new, turbolinks: true and
return unless resource.save
    head 204
  end
  private
  attr_reader :resource
  def resource_params
    params.require(:archive).permit(:file,
:channel_id).merge(user: current_user)
  end
  def initialize_resource
    @resource = Archive.new channel_id:
SecureRandom.uuid
  end
  def build_resource
    @resource = Archive.new
resource_params
  end
end
Файл
app/controllers/sessions_controller.rb
class SessionsController <
ApplicationController
  skip_before_action :authenticate!, only:
%i[new create]
  def create
    render :new, turbolinks: true and
return unless verify_recaptcha &&
resource.save
    cookies.encrypted[:auth_token] =
resource.auth_token.id
    redirect_to session.delete(:redirect)
|| :profile
  end
  def destroy
    resource.destroy
    cookies.encrypted[:auth_token] = nil
    redirect_to :root
  end
  private
  def resource
    @resource ||= Session.new auth_token:
AuthToken.find(cookies.encrypted[:auth_tok
en])
  end
  def resource_params

params.require(:session).permit(:email,
:password)
  end
  def initialize_resource
    @resource = Session.new
  end
  def build_resource
    @resource = Session.new
resource_params
  end
end
Файл
app/controllers/api/submissions_controller
.rb
class Api::SubmissionsController <
Api::ApplicationController
  skip_before_action :authorize_collection
  private
  def collection
    @collection ||=
Submission.pending.with_attached_source.in
cludes :problem, :compiler
  end
end
Файл
app/controllers/api/constants_controller.r
b
class Api::ConstantsController <
Api::ApplicationController
  skip_before_action :authorize_collection
  def collection
    Constants
  end
end
Файл
app/controllers/api/logs_controller.rb
class Api::LogsController <
Api::ApplicationController
  def create
    render :errors, status: 422 and return
unless resource.save
    head 204
  end
  private
  attr_reader :resource
  def parent
    @parent ||= Submission.find
params[:submission_id]
  end
  def resource_params
```

```
    params.require(:log).permit(:data,
:type)
  end
  def build_resource
    @resource = parent.logs.new
resource_params
  end
end
Файл
app/controllers/api/compilers_controller.r
b
class Api::CompilersController <
Api::ApplicationController
  skip_before_action :authorize_collection
  def collection
    @collection ||= Compiler.all
  end
end
Файл
app/controllers/api/problems_controller.rb
class Api::ProblemsController <
Api::ApplicationController
  skip_before_action
:authorize_collection, :authorize_resource
  private
  def resource
    @resource ||= Problem.find params[:id]
  end
end
Файл
app/controllers/api/submission/release_con
troller.rb
class Api::Submission::ReleaseController <
Api::ApplicationController
  skip_before_action :authorize_resource
  def create
    head resource.save ? 204 : 422
  end
  private
  attr_reader :resource
  def resource_params

params.require(:release).permit(:test_resu
lt)
  end
  def parent
    @parent ||= Submission.find
params[:submission_id]
  end
  def build_resource
    @resource = Release.new parent,
resource_params
  end
end
Файл
app/controllers/api/submission/take_contro
ller.rb
class Api::Submission::TakeController <
Api::ApplicationController
  skip_before_action :build_resource,
:authorize_resource
  around_action :wrap_in_transaction,
only: :create
  def create
    head parent.take! ? 204 : 422
  end
  private
  def parent
    @parent ||= Submission.lock.find
params[:submission_id]
  end
  def wrap_in_transaction
    ActiveRecord::Base.transaction { yield
}
  end
end
Файл
app/controllers/api/submission/fail_contro
ller.rb
class Api::Submission::FailController <
Api::ApplicationController
  skip_before_action :build_resource,
:authorize_resource
  def create
    head parent.fail! ? 204 : 422
  end
  private
  def parent
    @parent ||= Submission.find
params[:submission_id]
  end
end
Файл
app/controllers/api/results_controller.rb
class Api::ResultsController <
Api::ApplicationController
  def create
    render :errors, status: 422 and return
unless resource.save
    head 204
  end
  private
  attr_reader :resource
  def resource_params

params.require(:result).permit(:status,
:log, :memory, :time, :test_id,
:submission_id)
  end
  def build_resource
    @resource = Result.new resource_params
  end
end
Файл
app/controllers/api/application_controller
.rb
class Api::ApplicationController <
ApplicationController
  skip_before_action
:verify_authenticity_token
  before_action -> { response.status = 201
}, only: :create
  rescue_from(Pundit::NotAuthorizedError)
{ head 403 }
  private
  def authenticate!
    head 401 unless
ActiveSupport::SecurityUtils.secure_compar
e params[:access_token] || '',
ENV['API_ACCESS_TOKEN']
  end
  def current_user
    nil
  end
  def default_url_options
    {}
  end
  def set_locale
    I18n.locale = :en
  end
end
Файл
app/controllers/api/workers_controller.rb
class Api::WorkersController <
Api::ApplicationController
  def create
    render :errors, status: 422 and return
unless resource.save
  end
  def update
    render :errors, status: 422 and return
unless resource.update resource_params
    head 204
  end
  private
  def resource
    @resource ||= Worker.find params[:id]
  end
  def resource_params
    params.require(:worker).permit \
      :alive_at,
      :api_type,
      :api_version,
      :name,
      :status,
      :webhook_supported,
      ips: [],
      task_status: []
  end
  def build_resource
    @resource = Worker.new resource_params
  end
end
Файл app/controllers/workers_controller.rb
class WorkersController <
ApplicationController
  def destroy
    resource.destroy
    redirect_to :workers
  end
  private
  def resource
    @resource ||= Worker.find params[:id]
  end
  def collection
    @collection ||= Worker.order alive_at:
:desc
  end
end
Файл
app/controllers/memberships_controller.rb
class MembershipsController <
ApplicationController
  def create
    if resource.save
      redirect_to resource.group
    else
      render :new, turbolinks: true
    end
  end
  def update
    resource.update resource_params
    redirect_back fallback_location:
resource.group
  end
  def destroy
    resource.destroy
    redirect_back fallback_location:
resource.group
  end
  private
  def parent
    @parent ||= Group.find
params[:group_id] if params[:group_id]
  end
  def collection
    @collection ||= (parent ||
current_user).pending_memberships.order(id
: :desc).page(params[:page])
  end
  def resource
    @resource ||= Membership.find
params[:id]
  end
  def resource_params
    { state: :accepted }
  end
  def create_resource_params

params.require(:membership).permit(:user_i
d, :type).merge(group: parent)
  end
  def initialize_resource
    @resource =
parent.pending_memberships.new
  end
  def build_resource
    @resource = MembershipFactory.build
current_user, create_resource_params
  end
  def policy record
    policy =
PolicyFinder.new(record).policy!
    policy == MembershipPolicy ?
MembershipPolicy.new(current_user, record,
parent: parent) : super
  end
end
Файл
app/controllers/sharings_controller.rb
class SharingsController <
ApplicationController
  def create
    render :new and return unless
resource.save
    redirect_to parent
  end
  private
  attr_reader :resource
  def parent
    @parent ||= Group.find
params[:group_id]
  end
  def resource_params

params.require(:sharing).permit(:problem_i
d).merge(group: parent)
  end
  def initialize_resource
    @resource = Sharing.new group: parent
  end
  def build_resource
    @resource = Sharing.new
resource_params
  end
end
Файл
app/controllers/confirmation_requests_cont
roller.rb
class ConfirmationRequestsController <
ApplicationController
  def create
    if resource.save
      flash[:success] = I18n.t
'flash.save.success'
    else
      flash[:error] = I18n.t
'flash.save.failure'
    end
    redirect_to :profile
  end
  private
  attr_reader :resource
  def collection
    @collection ||=
ConfirmationRequest.includes(:user).page(p
arams[:page])
  end
  def build_resource
    @resource = ConfirmationRequest.new
user: current_user
  end
end
Файл app/jobs/application_job.rb
class ApplicationJob < ActiveJob::Base
end
Файл
app/jobs/process_problem_archive_job.rb
class ProcessProblemArchiveJob <
ApplicationJob
  def perform user, archive_path,
channel_id
    @channel_id = channel_id
    log 'Reading the zip file..'
    zip = Zip::File.open archive_path
    xml = Nokogiri::XML
zip.get_input_stream 'problem.xml'
    xml.xpath('problems/problem').each do
|problem_xml|
      log 'Looking for the problem or
creating a new one..'
      problem =
Problem.find_or_initialize_by(id:
problem_xml[:id]) { |problem| problem.user
= user }
      log 'Parsing attributes..'
      problem.assign_attributes
Archive::ProblemParser.attributes
problem_xml, zip, user
      log problem.save ? 'Saved the
problem.' : "Validation errors
occured:\n#{
```

```
problem.errors.full_messages.to_sentence
}"
    end
  rescue StandardError => e
    log "An expection occured: #{ e
}.\n\n#{ e.backtrace }"
    raise
  ensure
    zip.close if zip
    FileUtils.remove archive_path
    log 'Done.'
  end
  private
  def log message
    ActionCable.server.broadcast
"ProcessProblemArchive:#{ @channel_id }",
message
  end
end
Файл app/helpers/application_helper.rb
module ApplicationHelper
  ALLOWED_TAGS = %w[b br em i p span
strong sub sup table tbody td th thead tr
img ul ol li].freeze
  ALLOWED_ATTRIBUTES = %w[class src
alt].freeze
  def language_select_options
    I18n.available_locales.map { |locale|
[translate(:language, locale: locale),
locale] }
  end
  def visible_compilers
    current_user.administrator? ?
Compiler.all : Compiler.status_public
  end
  def checker_compilers
    current_user.administrator? ?
Compiler.all : Compiler.where.not(status:
:in_test)
  end
  def sanitize_for_problem text
    sanitize text, tags: ALLOWED_TAGS,
attributes: ALLOWED_ATTRIBUTES
  end
end
Файл app/services/release.rb
class Release
  def initialize submission, params
    @submission = submission
    @test_result = params[:test_result]
  end
  def save
    return false unless
@submission.release
    @submission.update test_result:
@test_result, score: score, max_score:
max_score
  end
  private
  def score

@submission.results.joins(:test).where(sta
tus: :ok).sum('tests.point')
  end
  def max_score
    Test.unscoped.where(problem_id:
@submission.problem_id).sum(:point)
  end
end
Файл app/services/standing_redis_store.rb
class StandingRedisStore
  def initialize user_id, problem_id
    @user_id, @problem_id = user_id,
problem_id
    @key = "#{ user_id }_#{ problem_id }"
  end
  def get
    $redis_standings.set @key,
score_from_database unless
$redis_standings.exists @key
    $redis_standings.get @key
  end
  def update_if_exists
    $redis_standings.set @key,
score_from_database if
$redis_standings.exists @key
  end
  private
  def score_from_database
    params = { user_id: @user_id,
problem_id: @problem_id, test_state:
:done, test_result: :ok }

Submission.select(:score).order(created_at
: :desc).find_by(params)&.score
  end
  class << self
    def get *args
      new(*args).get
    end
    def update_if_exists *args
      new(*args).update_if_exists
    end
  end
end
Файл app/services/avatar.rb
class Avatar
  include ActiveModel::Validations
  include Draper::Decoratable
  attr_accessor :user, :file
  validates :user, :file, presence: true
  validate :blob_must_be_variable,
:blob_size_must_not_be_greater_that_2_mega
bytes
  delegate :avatar, to: :user, allow_nil:
true
  delegate :attached?, to: :avatar,
allow_nil: true
  delegate :as_json, :url, to: :decorate
  def initialize params = {}
    @user, @file = params.values_at :user,
:file
  end
  def save
    user.avatar = blob if valid?
  end
  def destroy
    avatar&.purge_later
  end
  private
  def blob
    @blob ||=
ActiveStorage::Blob.build_after_upload \
      io: file.open, filename:
file.original_filename, content_type:
file.content_type
  end
  def blob_must_be_variable
    return if file.blank?
    errors.add :file, :invalid unless
blob.variable?
  end
  def
blob_size_must_not_be_greater_that_2_megab
ytes
    return if file.blank?
    errors.add :file, :too_long, count:
2.megabytes if blob.byte_size >
2.megabytes
  end
end
Файл app/services/password_recovery.rb
class PasswordRecovery
  include ActiveModel::Validations
  attr_accessor :email
  validate :user_must_be_present
  validates :email, presence: true
  def initialize params = {}
    @email = params[:email]
  end
  def to_key; end
  def persisted?; false; end
  def save
    return unless valid?
    user.update! password_recovery_token:
SecureRandom.uuid

PasswordRecoveryMailer.email(user).deliver
_later
  end
  private
  def user
    @user ||= User.find_by 'lower(email) =
lower(?)', email
  end
  def user_must_be_present
    errors.add :email, :invalid if
email.present? && user.blank?
  end
end
Файл app/services/session.rb
class Session
  include ActiveModel::Validations
  attr_accessor :email, :password
  validates :email, :password, presence:
true
  validate :user_must_be_present,
:password_must_pass_authentication
  delegate :destroy, to: :auth_token
  def initialize params = {}
    @email, @password, @auth_token =
params.values_at :email, :password,
:auth_token
  end
  def to_key; end
  def persisted?; false; end
  def save
    valid? ? auth_token.save : false
  end
  def user
    @user ||= User.find_by 'lower(email) =
lower(?)', email
  end
  def auth_token
    @auth_token ||= user.auth_tokens.build
  end
  private
  def user_must_be_present
    return if email.blank?
    errors.add :email, :invalid if
user.blank?
  end
  def password_must_pass_authentication
    return if password.blank? ||
user.blank?
    errors.add :password, :invalid unless
user.authenticate password
  end
end
Файл app/services/constants.rb
class Constants
  class << self
    def as_json *args
      {
        log_types: Log.types,
        problem_translation_languages:
ProblemTranslation.languages,
        result_statuses: Result.statuses,
        submission_test_states:
Submission.test_states,
        submission_test_results:
Submission.test_results,
        tag_translation_languages:
TagTranslation.languages
      }
    end
  end
end
Файл app/services/archive.rb
class Archive
  include ActiveModel::Validations
  attr_accessor :user, :file, :channel_id
  validates :user, :file, :channel_id,
presence: true
  def initialize params = {}
    @user, @file, @channel_id =
params.values_at :user, :file, :channel_id
  end
  def to_key; end
  def persisted?; false; end
  def to_model; self; end
  def save
    return false unless valid?
    FileUtils.copy file.path, filename
    ProcessProblemArchiveJob.perform_later
user, filename, channel_id
    true
  end
  private
  def filename
    @filename ||= File.join Dir.tmpdir,
SecureRandom.uuid
  end
end
Файл
app/services/archive/submission_parser.rb
class Archive::SubmissionParser
  def initialize xml, zip, user
    @xml, @zip, @user = xml, zip, user
  end
  def attributes
    { user: @user, compiler_id:
@xml[:compiler_id], source: source
}.compact
  end
  def source
    Archive::FileParser.blob
@xml[:source], @zip
  end
  class << self
    def attributes *args
      new(*args).attributes
    end
  end
end
Файл
app/services/archive/example_parser.rb
class Archive::ExampleParser
  class << self
    def attributes xml
      { id: xml[:id], _destroy:
xml[:_destroy], input: xml[:input],
answer: xml[:answer] }.compact
    end
  end
end
Файл
app/services/archive/problem_parser.rb
class Archive::ProblemParser
  def initialize xml, zip, user
    @xml, @zip, @user = xml, zip, user
  end
  def attributes
    {
      memory_limit: @xml[:memory_limit],
      time_limit: @xml[:time_limit],
      real_time_limit:
@xml[:real_time_limit],
      private: @xml[:private],
      checker_compiler_id:
@xml[:checker_compiler_id],
      checker_source: checker_source,
      tag_ids: tag_ids,
      examples_attributes:
examples_attributes,
      tests_attributes: tests_attributes,
      translations_attributes:
translations_attributes,
      submissions_attributes:
submissions_attributes
    }.compact
  end
  private
  def checker_source
    Archive::FileParser.blob
@xml[:checker_source], @zip
  end
  def tag_ids
    return if @xml.xpath('tags').blank?
    @xml.xpath('tags/tag').map { |tag_xml|
tag_xml[:id] }.compact
  end
  def examples_attributes
```

```
    return if
@xml.xpath('examples').blank?
    @xml.xpath('examples/example').map {
|example_xml|
Archive::ExampleParser.attributes
example_xml }
  end
  def tests_attributes
    return if @xml.xpath('tests').blank?
    @xml.xpath('tests/test').map {
|test_xml| Archive::TestParser.attributes
test_xml, @zip, @xml[:id] }
  end
  def translations_attributes
    return if
@xml.xpath('translations').blank?

@xml.xpath('translations/translation').map
do |translation_xml|

Archive::TranslationParser.attributes
translation_xml, @xml[:id]
    end
  end
  def submissions_attributes
    return if
@xml.xpath('submissions').blank?

@xml.xpath('submissions/submission').map
do |submission_xml|
      Archive::SubmissionParser.attributes
submission_xml, @zip, @user
    end
  end
  class << self
    def attributes *args
      new(*args).attributes
    end
  end
end
Файл app/services/archive/test_parser.rb
class Archive::TestParser
  def initialize xml, zip, problem_id
    @xml, @zip, @problem_id = xml, zip,
problem_id
  end
  def attributes
    { id: id, _destroy: @xml[:_destroy],
num: @xml[:num], point: @xml[:point],
input: input, answer: answer }.compact
  end
  private
  def id
    return nil if @problem_id.blank? ||
@xml[:num].blank?

Test.unscoped.select(:id).find_by(problem_
id: @problem_id, num: @xml[:num])&.id
  end
  def input
    Archive::FileParser.blob @xml[:input],
@zip
  end
  def answer
    Archive::FileParser.blob
@xml[:answer], @zip
  end
  class << self
    def attributes *args
      new(*args).attributes
    end
  end
end
Файл app/services/archive/file_parser.rb
class Archive::FileParser
  class << self
    def blob path, zip

ActiveStorage::Blob.build_after_upload io:
zip.get_input_stream(path), filename:
File.basename(path)
    rescue Errno::ENOENT
      nil
    end
  end
end
Файл
app/services/archive/translation_parser.rb
class Archive::TranslationParser
  def initialize xml, problem_id
    @xml, @problem_id = xml, problem_id
  end
  def attributes
    {
      id: id,
      _destroy: @xml[:_destroy],
      default: @xml[:default],
      language: @xml[:language],
      caption: @xml[:caption],
      author: @xml[:author],
      text: text,
      technical_text: technical_text
    }.compact
  end
  private
  def id
    return nil if @problem_id.blank? ||
@xml[:language].blank?

ProblemTranslation.select(:id).find_by(pro
blem_id: @problem_id, language:
@xml[:language])&.id
  end
  def text
    @xml.at_xpath('text')&.inner_html
  end
  def technical_text

@xml.at_xpath('technical_text')&.inner_htm
l
  end
  class << self
    def attributes *args
      new(*args).attributes
    end
  end
end
Файл app/services/submission/retest.rb
class Submission::Retest
  attr_reader :submission
  def initialize submission
    @submission = submission
  end
  def save
    Submission.transaction do
      submission.results.delete_all
      submission.logs.delete_all
      submission.update test_state:
:pending,  fails_count: 0, score: nil,
test_result: nil, max_score: nil
    end
  end
end
Файл
app/services/confirmation_request/reject.r
b
class ConfirmationRequest::Reject
  attr_reader :confirmation_request
  delegate :pending?, to:
:confirmation_request, prefix: true
  def initialize confirmation_request
    @confirmation_request =
confirmation_request
  end
  def save
    confirmation_request.update status:
:rejected
  end
end
Файл
app/services/confirmation_request/accept.r
b
class ConfirmationRequest::Accept
  attr_reader :confirmation_request
  delegate :pending?, to:
:confirmation_request, prefix: true
  def initialize confirmation_request
    @confirmation_request =
confirmation_request
  end
  def save
    confirmation_request.update(status:
:accepted).tap do |result|
      confirmation_request.user.update!
roles: :confirmed if result
    end
  end
end
Файл config/environment.rb
require_relative 'application'
Rails.application.initialize!
Файл config/locales/flash.uk.yml
---
uk:
  flash:
    not_authorized: Ви не маєте дозволу на
виконання цієї дії
    save:
      failure: Помилки валідації
      success: Успішно збережено
Файл config/locales/ru.yml
---
ru:
  activerecord:
    errors:
      messages:
        confirmation: не совпадает
  confirm: Вы уверены?
  errors:
    messages:
      exists: уже существует
      model_invalid: 'Возникли ошибки:
%{errors}'
      other_than: должно отличаться от
%{count}
      present: нужно оставить пустым
      required: не может отсутствовать
  helpers:
    submit:
      problem:
        create: Загрузить архив
      session:
        create: Войти
      submission:
        create: Отправить
      user:
        create: Регистрация
        update: Обновить профиль
      password_recovery:
        create: Восстановить пароль
      password_recovery:
        create: Сохранить новый пароль
  language: Русский
  number:
    errors:
      messages:
        restrict_dependent_destroy:
          has_many: 'Невозможно удалить
запись, так как существуют зависимости:
%{record}'
          has_one: 'Невозможно удалить
запись, так как существует зависимость:
%{record}'
    human:
      storage_units:
        units:
          eb: ЭБ
          pb: ПБ
Файл
config/locales/attributes/submission.ru.ym
l
---
ru:
  activerecord:
    attributes:
      submission:
        compiler_id: Компилятор
        source: Исходный код
Файл config/locales/attributes/user.ru.yml
---
ru:
  activerecord:
    attributes:
      user:
        created_at: Дата регистрации
        name: Фамилия, Имя
        password: Пароль
        password_confirmation:
Подтверждение пароля
        score: Балы
        skills: Навыки
        username: Ник
Файл
config/locales/attributes/submission.uk.ym
l
---
uk:
  activerecord:
    attributes:
      submission:
        compiler_id: Компілятор
        source: Код програми
Файл
config/locales/attributes/submission.en.ym
l
---
en:
  activerecord:
    attributes:
      submission:
        compiler_id: Compiler
        source: Source
Файл config/locales/attributes/user.uk.yml
---
uk:
  activerecord:
    attributes:
      user:
        created_at: Дата реєстрації
        name: Прізвище, Ім’я
        password: Пароль
        password_confirmation:
Підтвердження пароля
        score: Бали
        skills: Навички
        username: Нік
Файл
config/locales/attributes/session.ru.yml
---
ru:
  activemodel:
    attributes:
      session:
        password: Пароль
Файл
config/locales/attributes/problem.ru.yml
---
ru:
  activerecord:
    attributes:
      problem:
        archive: Архив
Файл config/locales/attributes/user.en.yml
---
en:
  activerecord:
    attributes:
      user:
        created_at: Registration date
        name: Full name
        password: Password
        password_confirmation: Password
confirmation
        score: Score
        skills: Skills
Файл
config/locales/attributes/problem.en.yml
---
en:
  activerecord:
    attributes:
      problem:
```

```
      archive: Archive
Файл
config/locales/attributes/problem.uk.yml
---
uk:
  activerecord:
    attributes:
      problem:
        archive: Архів
Файл
config/locales/attributes/session.en.yml
---
en:
  activemodel:
    attributes:
      session:
        password: Password
Файл
config/locales/attributes/session.uk.yml
---
uk:
  activemodel:
    attributes:
      session:
        password: Пароль
Файл config/locales/flash.ru.yml
---
ru:
  flash:
    not_authorized: Вы не авторизованы для
выполнения этого действия
    save:
      failure: Ошибки валидации
      success: Успешно сохранено
Файл config/locales/flash.en.yml
---
en:
  flash:
    not_authorized: You are not authorized
to perform this action
    save:
      failure: Validation errors
      success: Saved successfully
Файл config/locales/views/workers.en.yml
---
en:
  workers:
    index:
      title: Workers
Файл
config/locales/views/membership.uk.yml
---
uk:
  membership:
    index:
      created_at: Дата створення
      group_name: Група
      state: Тип
      user_name: Користувач
    new:
      invite: Запросити до групи
      request: Приєднатись до групи
Файл
config/locales/views/submissions.ru.yml
---
ru:
  submissions:
    index:
      compiler: Компилятор
      id: ИД
      problem: Задача
      state: Статус
      title: Попытки
      user: Пользователь
    new:
      title: Отправить решение задачи
"%{problem_caption}"
    show:
      compiler_log: Лог компиляции
      memory_limit: Лимит памяти
      problem: Задача
      results: Результаты
      source: Исходный код
      state: Статус
      test_result_compiler_error: Ошибка
компиляции решения
      test_state_failed: Ошибка
тестирования решения
      time_limit: Лимит времени
      title: Попытка №%{id}
      user: Пользователь
Файл config/locales/views/header.en.yml
---
en:
  header:
    administration: Administration
    compilers: Compilers
    confirmation_requests: Confirmation
requests
    groups: Groups
    problems: Problems
    problems_all: All problems
    problems_tags: Classification
    signin: Sign in
    signout: Sign out
    submissions: Submissions
    workers: Workers
Файл config/locales/views/sessions.en.yml
---
en:
  sessions:
    form:
      reset_password: Reset password
Файл config/locales/views/archives.en.yml
---
en:
  archives:
    new:
      title: Create a task from the
archive
Файл config/locales/views/problems.ru.yml
---
ru:
  problems:
    index:
      difficulty: Сложность
      name: Название
      number: "№"
      title: Задачи
      new: Создать новую задачу
      new_archive: Загрузить задачу из
архива
    new:
      tabs:
        basic: Основная информация
        examples: Примеры
        tests: Тесты
        translations: Переводы
      title: Создать задачу
    show:
      download: Скачать
      example_input: Ввод
      example_output: Вывод
      examples: Примеры
      languages:
        en: английском
        uk: украинском
      missing_translation: Перевод на
выбраном языке отсутствует, условия
отображены на %{language} языке.
      select_file: Выберите файл
      submissions: Отправленые решения для
этой задачи
      technical_text: Технические условия
Файл config/locales/views/results.en.yml
---
en:
  results:
    index:
      memory: Memory
      num: Test
      status: Status
      time: Time
Файл config/locales/views/home.en.yml
---
en:
  home:
    show:
      sections:
        first:
          list:
            '0': It works? Show!
            '1': Using GoTo and the code
is still readable? Show!
            '2': Copypasting other's code
and no one knows? Show!
            '3': "??? No, please don't
show!"
          subtitle: The world should know
its heroes.
          title: Prove to everybody that
your code is cooler!
        second:
          title: Sign up now!
Файл config/locales/views/home.uk.yml
---
uk:
  home:
    show:
      sections:
        first:
          list:
            '0': Воно працює? Покажи!
            '1': Пишеш через GoTo і це
можливо прочитати? Покажи!
            '2': Копіпастиш чужий код і
про це ніхто не знає? Покажи!
            '3': Стоїть не свій код? Ні,
не показуй!
          subtitle: Світ має знати своїх
героїв.
          title: Доведи всім, що твій код
круче!
        second:
          title: Реєструйся зараз!
Файл config/locales/views/problems.uk.yml
---
uk:
  problems:
    index:
      difficulty: Складність
      name: Назва
      number: "№"
      title: Задачі
      new: Створити нову задачу
      new_archive: Завантажити задачу з
архіву
    new:
      tabs:
        basic: Основна інформація
        examples: Приклади
        tests: Тести
        translations: Переклади
      title: Створити задачу
    show:
      download: Завантажити
      example_input: Введення
      example_output: Виведення
      examples: Приклади
      languages:
        en: англійською
        ru: російською
      missing_translation: Переклад
обраною мовою відсутній, умову відображено
%{language} мовою.
      select_file: Оберіть файл
      submissions: Відправлені розв'язки
для цієї задачі
      technical_text: Технічні умови
Файл
config/locales/views/confirmation_requests
.uk.yml
---
uk:
  confirmation_requests:
    index:
      created_at: Дата відправлення
      status: Статус
      title: Запити на підтвердження
      user_name: Ім'я
      user_username: Нік
Файл config/locales/views/tags.en.yml
---
en:
  tags:
    index:
      name: Tag name
      problems_count: Problems count
      title: Classification
Файл
config/locales/views/submissions.en.yml
---
en:
  submissions:
    index:
      compiler: Compiler
      id: ID
      problem: Problem
      state: State
      title: Submissions
      user: User
    new:
      title: Submit a solution for the
"%{problem_caption}" problem
    show:
      compiler_log: Compiler log
      memory_limit: Memory limit
      problem: Problem
      results: Results
      source: Source code
      state: State
      test_result_compiler_error:
Compilation failed
      test_state_failed: Testing failed
      time_limit: Time limit
      title: 'Submission #%{id}'
      user: User
Файл config/locales/views/profile.uk.yml
---
uk:
  profile:
    show:
      ask_for_confirmation: Запросити
підтвердження
      membership_list: Список запрошень
      problems: Мої задачі
      submissions: Мої розв'язки
      title: Особистий кабінет
Файл config/locales/views/workers.ru.yml
---
ru:
  workers:
    index:
      title: Воркеры
Файл config/locales/views/compiler.en.yml
---
en:
  compiler:
    form:
      destroy: Delete
    index:
      id: ID
      name: Name
      new: New
      status: Status
      title: Compilers
      version: Version
Файл config/locales/views/tags.ru.yml
---
ru:
  tags:
    index:
      name: Название тега
      problems_count: Количество задач
      title: Класификация
Файл config/locales/views/tags.uk.yml
---
uk:
  tags:
    index:
      name: Назва тегу
      problems_count: Кількість задач
      title: Класифікація
```

Файл config/locales/views/workers.uk.yml
```
---
uk:
  workers:
    index:
      title: Воркери
```
Файл config/locales/views/membership.ru.yml
```
---
ru:
  membership:
    index:
      created_at: Дата создания
      group_name: Группа
      state: Тип
      user_name: Пользователь
    new:
      invite: Пригласить в группу
      request: Присоединиться к группе
```
Файл config/locales/views/group.en.yml
```
---
en:
  group:
    edit:
      title: Edit group
    index:
      created_at: Creation date
      name: Name
      owner: Owner
      title: Groups
      visibility: Visibility
      new: Create a new group
    new:
      title: Create a new group
    show:
      create_sharing: Add a problem
      description: Description
      edit: Edit group information
      members: Members
      membership_list: Invitation list
      new_membership: Invite
      not_a_member: You are not a group
member
      pending_member: Participation
request has been sent
      problem_list: Group problems
      standings: Standings
      submission_list: Group submissions
      visibility:
        moderated: Users can send
participation requests
        private: Only group owner can add
users
        public: Every member can join and
invite others to the group
```
Файл config/locales/views/archives.ru.yml
```
---
ru:
  archives:
    new:
      title: Создать задачу из архива
```
Файл config/locales/views/sessions.ru.yml
```
---
ru:
  sessions:
    form:
      reset_password: Восстановление
пароля
```
Файл config/locales/views/profile.en.yml
```
---
en:
  profile:
    show:
      ask_for_confirmation: Ask for
confirmation
      membership_list: Invitation list
      problems: My problems
      submissions: My solutions
      title: Personal cabinet
```
Файл config/locales/views/password_recoveries.u k.yml
```
---
uk:
  password_recoveries:
    create:
      title: Відновлення паролю
      content: Інструкції надіслані до
вашого пов’язаного облікового запису електронної пошти. Перевірте свою електронну пошту та дотримуйтесь інструкцій для скидання пароля.
```
Файл config/locales/views/group.ru.yml
```
---
ru:
  group:
    edit:
      title: Редактирование группы
    index:
      created_at: Дата создания
      name: Название
      owner: Владелец
      title: Группы
      visibility: Видимость
      new: Создать новую группу
    new:
      title: Создать новую группу
    show:
      create_sharing: Добавить задачу
      description: Описание
      edit: Редагтировать информацию про
группу
      members: Участники
      membership_list: Список приглашений
      new_membership: Пригласить
      not_a_member: Вы не являетесь
участником группы
      pending_member: Отправлено запрос на
добавление в группу
      problem_list: Задачи группы
      standings: Положение
      submission_list: Отправленные
решения участников группы
      visibility:
        moderated: Пользователи могут
отправлять запросы на добавление в группу
        private: Только владелец группы
может добавлять пользователей
        public: Каждый участник может
присоединиться и пригласить других в группу
```
Файл config/locales/views/password_recoveries.e n.yml
```
---
en:
  password_recoveries:
    create:
      title: Password recovery
      content: Instructions have been sent
to your associated email account. Check your email and follow the instructions to reset your password.
```
Файл config/locales/views/group.uk.yml
```
---
uk:
  group:
    edit:
      title: Редагування групи
    index:
      created_at: Дата створення
      name: Назва
      owner: Власник
      title: Групи
      visibility: Видимість
      new: Створити нову групу
    new:
      title: Створити нову групу
    show:
      create_sharing: Додати задачу
      description: Опис
      edit: Редагувати інформацію про
групу
      members: Учасники
      membership_list: Список запрошень
      new_membership: Запросити
      not_a_member: Ви не є учасником
групи
      pending_member: Надіслано запит на
приєднання до групи
      problem_list: Задачі групи
      standings: Таблиця результатів
      submission_list: Відправлені
розв'язки учасників групи
      visibility:
        moderated: Користувачі можуть
надсилати запити на приєднання до групи
        private: Лише власник групи може
додавати користувачів
        public: Кожен користувач може
приєднатись і запросити інших до групи
```
Файл config/locales/views/compiler.uk.yml
```
---
uk:
  compiler:
    form:
      destroy: Видалити
    index:
      id: ІД
      name: Назва
      new: Додати
      status: Статус
      title: Компілятори
      version: Версія
```
Файл config/locales/views/confirmation_request. ru.yml
```
---
ru:
  confirmation_requests:
    index:
      created_at: Дата отправления
      status: Статус
      title: Запросы на подтверждение
      user_name: Имя
      user_username: Ник
```
Файл config/locales/views/confirmatiom_request. en.yml
```
---
en:
  confirmation_requests:
    index:
      created_at: Sent date
      status: Status
      title: Confirmation requests
      user_name: Name
      user_username: Username
```
Файл config/locales/views/password_recoveries.r u.yml
```
---
ru:
  password_recoveries:
    create:
      title: Восстановление пароля
      content: Инструкции были отправлены
на ваш связанный адрес электронной почты. Проверьте свою электронную почту и следуйте инструкциям для сброса пароля.
```
Файл config/locales/views/header.uk.yml
```
---
uk:
  header:
    administration: Адміністрування
    compilers: Компілятори
    confirmation_requests: Запити на
підтвердження
    groups: Групи
    problems: Задачі
    problems_all: Усі задачі
    problems_tags: Класифікація
    signin: Вхід
    signout: Вихід
    submissions: Відправлені розв'язки
    workers: Воркери
```
Файл config/locales/views/profile.ru.yml
```
---
ru:
  profile:
    show:
      ask_for_confirmation: Запросить
подтверждение
      membership_list: Список приглашений
      problems: Мои задачи
      submissions: Мои решения
      title: Личный кабинет
```
Файл config/locales/views/sharing.en.yml
```
---
en:
  sharing:
    new:
      title: Add problem to the group
```
Файл config/locales/views/sessions.uk.yml
```
---
uk:
  sessions:
    form:
      reset_password: Відновлення паролю
```
Файл config/locales/views/header.ru.yml
```
---
ru:
  header:
    administration: Администрирование
    compilers: Компиляторы
    confirmation_requests: Запросы на
подтверждение
    groups: Группы
    problems: Задачи
    problems_all: Все задачи
    problems_tags: Класификация
    signin: Войти
    signout: Выйти
    submissions: Отправленые решения
    workers: Воркеры
```
Файл config/locales/views/sharing.ru.yml
```
---
ru:
  sharing:
    new:
      title: Добавить задачу в группу
```
Файл config/locales/views/results.ru.yml
```
---
ru:
  results:
    index:
      memory: Память
      num: Тест
      status: Статус
      time: Время
```
Файл config/locales/views/results.uk.yml
```
---
uk:
  results:
    index:
      memory: Пам'ять
      num: Тест
      status: Статус
      time: Час
```
Файл config/locales/views/home.ru.yml
```
---
ru:
  home:
    show:
      sections:
        first:
          list:
            '0': Оно работает? Покажи!
            '1': Пишеш через GoTo и это
можно прочитать? Покажи!
            '2': Копипастишь чужой код и
об этом никто не знает? Покажи!
            '3': Стоит на свой код? Нет,
не показывай!
          subtitle: Мир должен знать своих
героев.
          title: Докажи всем, что твой код
круче!
        second:
          title: Регистрируйся сейчас!
```
Файл config/locales/views/membership.en.yml

```
---
en:
  membership:
    index:
      created_at: Creation date
      group_name: Group
      state: Type
      user_name: User
    new:
      invite: Invite to the group
      request: Join the group
Файл config/locales/views/problems.en.yml
---
en:
  problems:
    index:
      difficulty: Difficulty
      name: Name
      number: "#"
      title: Problems
      new: Create a new problem
      new_archive: Create a new problem
from archive
    new:
      tabs:
        basic: Basic information
        examples: Examples
        tests: Tests
        translations: Translations
      title: Create a problem
    show:
      download: Download
      example_input: Input
      example_output: Output
      examples: Examples
      languages:
        ru: russian
        uk: ukrainian
      missing_translation: Translation for
selected language is missing, problem is
displayed in %{language}.
      select_file: Select file
      submissions: Submitted solutions for
this problem
      technical_text: Specifications
Файл config/locales/views/compiler.ru.yml
---
ru:
  compiler:
    form:
      destroy: Удалить
    index:
      id: ИД
      name: Название
      new: Добавить
      status: Статус
      title: Компиляторы
      version: Версия
Файл config/locales/views/sharing.uk.yml
---
uk:
  sharing:
    new:
      title: Додати задачу до групи
Файл config/locales/views/archives.uk.yml
---
uk:
  archives:
    new:
      title: Створити задачу з архіва
Файл
config/locales/views/submissions.uk.yml
---
uk:
  submissions:
    index:
      compiler: Компілятор
      id: ІД
      problem: Задача
      state: Статус
      title: Відправлені розв'язки
      user: Користувач
    new:
      title: Відправити рішення задачі
"%{problem_caption}"
    show:
      compiler_log: Лог компіляції
      memory_limit: Ліміт пам'яті
      problem: Задача
      results: Результати
      source: Код програми
      state: Статус
      test_result_compiler_error: Помилка
компіляції рішення
      test_state_failed: Помилка
тестування рішення
      time_limit: Ліміт часу
      title: Розв'язок №%{id}
      user: Користувач
Файл config/locales/uk.yml
---
uk:
  activerecord:
    errors:
      messages:
        confirmation: не співпадає
  confirm: Ви впевнені?
  errors:
    messages:
      exists: вже існує
      model_invalid: 'Виникли помилки:
%{errors}'
      other_than: має відрізнятися від
%{count}
      present: треба залишити пустим
      required: не може бути відсутнім
  helpers:
    submit:
      problem:
        create: Зберегти
      session:
        create: Вхід
      submission:
        create: Надіслати
      user:
        create: Реєстрація
        update: Оновити профіль
      password_recovery:
        create: Відновити пароль
      password_recovery:
        create: Зберегти новий пароль
  language: Українська
  number:
    errors:
      messages:
        restrict_dependent_destroy:
          has_many: 'Неможливо видалити
записи, так як існують залежності:
%{record}'
          has_one: 'Неможливо видалити
запис, так як існує залежність: %{record}'
    human:
      storage_units:
        units:
          eb: ЕБ
          pb: ПБ
Файл config/locales/simple_form.uk.yml
---
uk:
  simple_form:
    error_notification:
      default_message: 'Будь ласка,
продивіться наведені нижче проблеми:'
    'no': Ні
    placeholders:
      defaults:
        email: one@users.com
      group:
        name: Введіть назву групи
      problem:
        memory_limit: Допустимі затрати
пам'яті
        real_time_limit: Час до смерті
        time_limit: Час на виконання
    required:
      mark: "*"
      text: обов'язково
    'yes': Так
Файл config/locales/simple_form.ru.yml
---
ru:
  simple_form:
    error_notification:
      default_message: 'Пожалуйста,
просмотрите приведенные ниже проблемы:'
    'no': Нет
    placeholders:
      defaults:
        email: one@users.com
      group:
        name: Введите название группы
      problem:
        memory_limit: Допустимые затраты
памяти
        real_time_limit: Время до смерти
        time_limit: Время на выполнение
    required:
      mark: "*"
      text: обязательно
    'yes': Да
Файл config/locales/simple_form.en.yml
---
en:
  simple_form:
    error_notification:
      default_message: 'Please review the
problems below:'
    'no': 'No'
    placeholders:
      defaults:
        email: one@users.com
      group:
        name: Enter group name
      problem:
        memory_limit: Memory limit
        real_time_limit: Time to death
        time_limit: Time to complete
    required:
      mark: "*"
      text: required
    'yes': 'Yes'
Файл config/locales/en.yml
---
en:
  activerecord:
    errors:
      messages:
        confirmation: does not match
  confirm: Are your sure?
  errors:
    messages:
      exists: already exists
      model_invalid: 'Errors ocured:
%{errors}'
      other_than: must be different from
%{count}
      present: must be empty
      required: can not be absent
  helpers:
    submit:
      problem:
        create: Upload archive
      session:
        create: Sign in
      submission:
        create: Submit
      user:
        create: Registration
        update: Update profile
      password_recovery:
        create: Reset password
      password:
        create: Save new password
  language: English
  number:
    errors:
      messages:
        restrict_dependent_destroy:
          has_many: 'Can not delete the
record, because there are dependencies:
%{record}'
          has_one: 'Can not delete the
record, because there is a dependency:
%{record}'
    human:
      storage_units:
        units:
          eb: EB
          pb: PB
Файл config/application.conf
upstream backend {
  server
unix:///home/user/application/current/tmp/
sockets/server.sock fail_timeout=0;
}
server {
  listen *:80;
  return 301 https://$host$request_uri;
}
server {
  listen *:443;
  client_max_body_size 1024M;
  root
/home/user/application/current/public/;
  error_log
/home/user/application/current/log/nginx_e
rrors.log;
  gzip on;
  gzip_comp_level 6;
  gzip_proxied any;
  gzip_types text/plain text/css
application/javascript application/octet-
stream;
  location @backend {
    proxy_pass http://backend;
    proxy_redirect     off;
    proxy_set_header   Host
$host;
    proxy_set_header   X-Real-IP
$remote_addr;
    proxy_set_header   X-Forwarded-For
$proxy_add_x_forwarded_for;
  }
  location /cable {
    proxy_pass http://backend;
    proxy_set_header   Connection
Upgrade;
    proxy_set_header   Upgrade
websocket;
    proxy_set_header   Host
$host;
    proxy_set_header   X-Real-IP
$remote_addr;
    proxy_set_header   X-Forwarded-For
$proxy_add_x_forwarded_for;
  }
  location /assets {
    expires max;
    try_files $uri @backend;
  }
  location / {
    try_files $uri @backend;
  }
}
Файл config/application.service
[Unit]
  Description = Codelabs Application
  After = network.service
[Service]
  User = user
  Group = user
  WorkingDirectory =
/home/user/application/current
  Environment = 'RAILS_ENV=production'
  Environment =
'SOCKET=unix:///home/user/application/curr
ent/tmp/sockets/server.sock'
  Environment =
'PID=/home/user/application/current/tmp/pi
ds/server.pid'
```

```
  Environment =
'PATH=/home/user/.rubies/current/bin:/bin:
/usr/bin:/usr/local/bin'
  ExecStart = /usr/bin/env bundle exec
puma --config
/home/user/application/current/config/puma
.rb
  Restart = always
[Install]
  WantedBy = multi-user.target
Файл config/database.yml
default: &default
  adapter: postgresql
  encoding: unicode
  pool: <%= ENV.fetch('RAILS_MAX_THREADS',
5) %>
  timeout: 5000
development:
  <<: *default
  database: codelabs_development
test:
  <<: *default
  database: codelabs_test
  url: <%= ENV['DATABASE_URL'] %>
production:
  <<: *default
  url: <%= ENV['DATABASE_URL'] %>
Файл config/boot.rb
ENV['BUNDLE_GEMFILE'] ||=
File.expand_path('../Gemfile', __dir__)
require 'bundler/setup'
require 'bootsnap/setup'
Файл config/cable.yml
development:
  adapter: async
test:
  adapter: async
production:
  adapter: redis
  url: <%= ENV.fetch('REDIS_URL') {
'redis://localhost:6379/1' } %>
  channel_prefix: CodeLabs_production
Файл config/application.rb
require_relative 'boot'
require 'rails/all'
Bundler.require(*Rails.groups)
module CodeLabs
  class Application < Rails::Application
    config.load_defaults 5.2
    config.time_zone = 'Kyiv'
    config.active_record.default_timezone
= :local
  end
end
Файл config/environments/development.rb
Rails.application.configure do
  config.cache_classes = false
  config.eager_load = false
  config.consider_all_requests_local =
true
  if Rails.root.join('tmp', 'caching-
dev.txt').exist?

config.action_controller.perform_caching =
true
    config.cache_store = :memory_store
    config.public_file_server.headers = {
      'Cache-Control' => "public, max-
age=#{2.days.to_i}"
    }
  else

config.action_controller.perform_caching =
false
    config.cache_store = :null_store
  end
  config.active_storage.service = :local

config.action_mailer.raise_delivery_errors
= false
  config.action_mailer.perform_caching =
false
  config.active_support.deprecation = :log
  config.active_record.migration_error =
:page_load
  config.active_record.verbose_query_logs
= true
  config.assets.debug = true
  config.assets.quiet = true
  config.file_watcher =
ActiveSupport::EventedFileUpdateChecker
end
Файл config/environments/production.rb
Rails.application.configure do
  config.cache_classes = true
  config.eager_load = true
  config.consider_all_requests_local
= false
  config.action_controller.perform_caching
= true
  config.require_master_key = true
  config.public_file_server.enabled =
ENV['RAILS_SERVE_STATIC_FILES'].present?
  config.assets.js_compressor = :uglifier
  config.assets.compile = false
  config.active_storage.service = :local
  config.log_level = :debug
  config.log_tags = [ :request_id ]
  config.action_mailer.perform_caching =
false
  config.i18n.fallbacks =
[I18n.default_locale]
  config.active_support.deprecation =
:notify
  config.log_formatter =
::Logger::Formatter.new
  if ENV['RAILS_LOG_TO_STDOUT'].present?
    logger           =
ActiveSupport::Logger.new(STDOUT)
    logger.formatter =
config.log_formatter
    config.logger    =
ActiveSupport::TaggedLogging.new(logger)
  end

config.active_record.dump_schema_after_mig
ration = false
end
Файл config/environments/test.rb
Rails.application.configure do
  config.cache_classes = true
  config.eager_load = false
  config.public_file_server.enabled = true
  config.public_file_server.headers = {
    'Cache-Control' => "public, max-
age=#{1.hour.to_i}"
  }
  config.consider_all_requests_local
= true
  config.action_controller.perform_caching
= false
  config.action_dispatch.show_exceptions =
false

config.action_controller.allow_forgery_pro
tection = false
  config.active_storage.service = :test
  config.action_mailer.perform_caching =
false
  config.action_mailer.delivery_method =
:test
  config.active_support.deprecation =
:stderr
end
Файл config/routes.rb
Rails.application.routes.draw do
  scope '(:language)', language:
/ru|en|uk/ do
    root controller: :home, action: :show
    resource :session, only: %i[new create
destroy]
    resource :profile, only: %i[show
update]
    resource :avatar, only: %i[create
destroy]
    resources :users, only: %i[index
create] do
      resources :problems, only: :index
      resources :submissions, only: :index
    end
    resources :problems do
      resources :submissions, only:
%i[index create]
    end
    resources :tags, only: :index do
      resources :problems, only: :index
    end
    resources :submissions, only: %i[index
show destroy] do
      resource :retest, only: :create,
module: :submission
    end
    resources :archives, only: %i[new
create]
    resources :compilers, except: :show
    resources :workers, only: %i[index
destroy]
    resources :groups do
      resources :memberships, except:
%i[show edit], shallow: true
      resources :sharings, only: %i[new
create]
      resources :submissions, :problems,
only: :index
      resource :standing, only: :show
    end
    resources :confirmation_requests,
only: %i[index create] do
      resource :reject, :accept, only:
:create, module: :confirmation_request
    end
    resources :memberships, only: :index
    resource :password_recovery, only:
%i[new create]
    resource :password, only: %i[edit
update]
  end
  namespace :api do
    resources :submissions, only: :index
do
      resources :take, :release, :fail,
only: :create, module: :submission
      resources :logs, only: :create
    end
    resources :problems, only: :show
    resources :results, only: :create
    resources :compilers, :constants,
only: :index
    resources :workers, only: %i[create
update]
  end
end
Файл config/deploy.rb
set :application, 'CodeLabs'
set :repo_url,
'git@gitlab.com:kostyanf14/CodeLabs.git'
set :rails_env, :production
set :default_env, { path:
"$HOME/.rubies/current/bin:$PATH" }
append :linked_files, '.env',
'config/master.key'
append :linked_dirs, 'log', 'tmp/pids',
'tmp/cache', 'tmp/sockets',
'public/system', 'storage'
set :deploy_to, '/home/user/application'
namespace :deploy do
  after :finishing, 'application:restart'
  after :finishing, 'nginx:reload'
  after :finishing, 'bundler:clean'
end
Файл config/storage.yml
test:
  service: Disk
  root: <%= Rails.root.join('tmp/storage')
%>
local:
  service: Disk
  root: <%= Rails.root.join('storage') %>
Файл config/initializers/recaptcha.rb
Recaptcha.configure do |config|
  config.site_key =
ENV['RECAPTCHA_SITE_KEY']
  config.secret_key =
ENV['RECAPTCHA_SECRET_KEY']
  config.hostname =
ENV['RECAPTCHA_HOSTNAME']
end
module Recaptcha
  module Adapters
    module ControllerMethods
      def recaptcha_flash_supported?
        true
      end
    end
  end
end
Файл
config/initializers/sizable_zip_input_stre
am.rb
class Zip::InputStream
  def size
    @size ||= read.size.tap { rewind }
  end
end
Файл config/initializers/assets.rb
Rails.application.configure do
  config.assets.version = '1.0'
  config.assets.paths <<
Rails.root.join('node_modules')
end
Файл config/initializers/revision.rb
REVISION = \
  if Rails.env.production? &&
ENV.key?('REVISION_FILE') &&
File.exist?(ENV['REVISION_FILE'])

IO.readlines(ENV['REVISION_FILE']).last.st
rip
  else
    ''
  end.freeze
Файл
config/initializers/turbolinks_render.rb
Rails.application.config.turbolinks_render
.render_with_turbolinks_by_default = false
Файл config/initializers/redis.rb
redis_connection = Redis.new
$redis_standings = Redis::Namespace.new
"#{ Rails.env }_standings", redis:
redis_connection
Файл
config/initializers/wrap_parameters.rb
ActiveSupport.on_load(:action_controller)
{ wrap_parameters format: [:json] }
Файл config/initializers/action_mailer.rb
Rails.application.configure do

config.action_mailer.raise_delivery_errors
= true
  config.action_mailer.default_url_options
= { host: ENV['APPLICATION_URL'] }
  config.action_mailer.smtp_settings = {
    address: ENV['SMTP_ADDRESS'],
    port: ENV['SMTP_PORT'],
    user_name: ENV['SMTP_USER_NAME'],
    password: ENV['SMTP_PASSWORD'],
    authentication: :login,
    enable_starttls_auto: true
  }
end
Файл
config/initializers/cookies_serializer.rb
Rails.application.config.action_dispatch.c
ookies_serializer = :json
Файл config/initializers/generators.rb
Rails.application.configure do
  config.generators do |g|
    g.helper false
    g.controller assets: false, decorator:
false
  end
end
```

Файл config/initializers/bullet.rb

```
Rails.application.configure do
  config.after_initialize do
    Bullet.enable = true
    Bullet.bullet_logger = true
    Bullet.rails_logger = true
  end
end if Rails.env.development?
```

Файл config/initializers/content_security_policy.rb

```
if Rails.env.production?

Rails.application.config.content_security_policy do |policy|
    policy.default_src :self, :https
    policy.font_src    :self, :https, :data
    policy.img_src     :self, :https, :data
    policy.object_src  :none
    policy.script_src  :self, :https
    policy.style_src   :self, :https, :unsafe_inline
    policy.connect_src :self, :https, :wss
  end

Rails.application.config.content_security_policy_nonce_generator = -> request { SecureRandom.base64(16) }
end
```

Файл config/initializers/rescue_responses.rb

```
Rails.application.configure do

config.action_dispatch.rescue_responses['Pundit::NotAuthorizedError'] = :forbidden
end
```

Файл config/initializers/simple_form_bootstrap.rb

```
Dir[Rails.root.join('lib/components/**/*.rb')].each { |f| require f }
SimpleForm.setup do |config|
  config.button_class = 'btn'
  config.boolean_label_class = 'form-check-label'
  config.label_text = lambda { |label, required, explicit_label| "#{label} #{required}" }
  config.boolean_style = :inline
  config.item_wrapper_tag = :div

config.include_default_input_wrapper_class = false
  config.error_notification_class = 'alert alert-danger'
  config.error_method = :to_sentence
  config.input_field_error_class = 'is-invalid'
  config.wrappers :vertical_form, tag: 'div', class: 'form-group', error_class: 'form-group-invalid', valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.use :placeholder
    b.optional :maxlength
    b.optional :minlength
    b.optional :pattern
    b.optional :min_max
    b.optional :readonly
    b.use :label, class: 'form-control-label'
    b.use :input, class: 'form-control', error_class: 'is-invalid'
    b.use :full_error, wrap_with: { tag: 'div', class: 'invalid-feedback' }
    b.use :hint, wrap_with: { tag: 'small', class: 'form-text text-muted' }
  end
  config.wrappers :vertical_boolean, tag: 'fieldset', class: 'form-group', error_class: 'form-group-invalid', valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.optional :readonly
    b.wrapper :form_check_wrapper, tag: 'div', class: 'form-check' do |bb|
      bb.use :input, class: 'form-check-input', error_class: 'is-invalid'
      bb.use :label, class: 'form-check-label'
      bb.use :full_error, wrap_with: { tag: 'div', class: 'invalid-feedback' }
      bb.use :hint, wrap_with: { tag: 'small', class: 'form-text text-muted' }
    end
  end
  config.wrappers :vertical_collection, item_wrapper_class: 'form-check', tag: 'fieldset', class: 'form-group', error_class: 'form-group-invalid', valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.optional :readonly
    b.wrapper :legend_tag, tag: 'legend', class: 'col-form-label pt-0' do |ba|
      ba.use :label_text
    end
    b.use :input, class: 'form-check-input', error_class: 'is-invalid'
    b.use :full_error, wrap_with: { tag: 'div', class: 'invalid-feedback d-block' }
    b.use :hint, wrap_with: { tag: 'small', class: 'form-text text-muted' }
  end
  config.wrappers :vertical_file, tag: 'div', class: 'form-group', error_class: 'form-group-invalid', valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.use :placeholder
    b.optional :maxlength
    b.optional :minlength
    b.optional :readonly
    b.use :label
    b.use :input, class: 'form-control-file', error_class: 'is-invalid'
    b.use :full_error, wrap_with: { tag: 'div', class: 'invalid-feedback d-block' }
    b.use :hint, wrap_with: { tag: 'small', class: 'form-text text-muted' }
  end
  config.wrappers :vertical_multi_select, tag: 'div', class: 'form-group', error_class: 'form-group-invalid', valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.optional :readonly
    b.use :label, class: 'form-control-label'
    b.wrapper tag: 'div', class: 'd-flex flex-row justify-content-between align-items-center' do |ba|
      ba.use :input, class: 'form-control mx-1', error_class: 'is-invalid'
    end
    b.use :full_error, wrap_with: { tag: 'div', class: 'invalid-feedback d-block' }
    b.use :hint, wrap_with: { tag: 'small', class: 'form-text text-muted' }
  end
  config.wrappers :vertical_range, tag: 'div', class: 'form-group', error_class: 'form-group-invalid', valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.use :placeholder
    b.optional :readonly
    b.optional :step
    b.use :label
    b.use :input, class: 'form-control-range', error_class: 'is-invalid'
    b.use :full_error, wrap_with: { tag: 'div', class: 'invalid-feedback d-block' }
    b.use :hint, wrap_with: { tag: 'small', class: 'form-text text-muted' }
  end
  config.wrappers :horizontal_form, tag: 'div', class: 'form-group row', error_class: 'form-group-invalid', valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.use :placeholder
    b.optional :maxlength
    b.optional :minlength
    b.optional :pattern
    b.optional :min_max
    b.optional :readonly
    b.use :label, class: 'col-sm-3 col-form-label'
    b.wrapper :grid_wrapper, tag: 'div', class: 'col-sm-9' do |ba|
      ba.use :input, class: 'form-control', error_class: 'is-invalid'
      ba.use :full_error, wrap_with: { tag: 'div', class: 'invalid-feedback' }
      ba.use :hint, wrap_with: { tag: 'small', class: 'form-text text-muted' }
    end
  end
  config.wrappers :horizontal_boolean, tag: 'div', class: 'form-group row', error_class: 'form-group-invalid', valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.optional :readonly
    b.wrapper tag: 'label', class: 'col-sm-3' do |ba|
      ba.use :label_text
    end
    b.wrapper :grid_wrapper, tag: 'div', class: 'col-sm-9' do |wr|
      wr.wrapper :form_check_wrapper, tag: 'div', class: 'form-check' do |bb|
        bb.use :input, class: 'form-check-input', error_class: 'is-invalid'
        bb.use :label, class: 'form-check-label'
        bb.use :full_error, wrap_with: { tag: 'div', class: 'invalid-feedback d-block' }
        bb.use :hint, wrap_with: { tag: 'small', class: 'form-text text-muted' }
      end
    end
  end
  config.wrappers :horizontal_collection, item_wrapper_class: 'form-check', tag: 'div', class: 'form-group row', error_class: 'form-group-invalid', valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.optional :readonly
    b.use :label, class: 'col-sm-3 form-control-label'
    b.wrapper :grid_wrapper, tag: 'div', class: 'col-sm-9' do |ba|
      ba.use :input, class: 'form-check-input', error_class: 'is-invalid'
      ba.use :full_error, wrap_with: { tag: 'div', class: 'invalid-feedback d-block' }
      ba.use :hint, wrap_with: { tag: 'small', class: 'form-text text-muted' }
    end
  end
  config.wrappers :horizontal_collection_inline, item_wrapper_class: 'form-check form-check-inline', tag: 'div', class: 'form-group row', error_class: 'form-group-invalid', valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.optional :readonly
    b.use :label, class: 'col-sm-3 form-control-label'
    b.wrapper :grid_wrapper, tag: 'div', class: 'col-sm-9' do |wr|
      wr.use :input, class: 'form-check-input', error_class: 'is-invalid'
      wr.use :label, class: 'form-check-label'
      wr.use :full_error, wrap_with: { tag: 'div', class: 'invalid-feedback d-block' }
      wr.use :hint, wrap_with: { tag: 'small', class: 'form-text text-muted' }
    end
  end
  config.wrappers :horizontal_file, tag: 'div', class: 'form-group row', error_class: 'form-group-invalid', valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.use :placeholder
    b.optional :maxlength
    b.optional :minlength
    b.optional :readonly
    b.use :label, class: 'col-sm-3 form-control-label'
    b.wrapper :grid_wrapper, tag: 'div', class: 'col-sm-9' do |ba|
      ba.use :input, error_class: 'is-invalid'
      ba.use :full_error, wrap_with: { tag: 'div', class: 'invalid-feedback d-block' }
      ba.use :hint, wrap_with: { tag: 'small', class: 'form-text text-muted' }
    end
  end
  config.wrappers :horizontal_multi_select, tag: 'div', class: 'form-group row', error_class: 'form-group-invalid', valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.optional :readonly
    b.use :label, class: 'col-sm-3 control-label'
    b.wrapper :grid_wrapper, tag: 'div', class: 'col-sm-9' do |ba|
      ba.wrapper tag: 'div', class: 'd-flex flex-row justify-content-between align-items-center' do |bb|
        bb.use :input, class: 'form-control mx-1', error_class: 'is-invalid'
      end
      ba.use :full_error, wrap_with: { tag: 'div', class: 'invalid-feedback d-block' }
      ba.use :hint, wrap_with: { tag: 'small', class: 'form-text text-muted' }
    end
  end
  config.wrappers :horizontal_range, tag: 'div', class: 'form-group row', error_class: 'form-group-invalid', valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.use :placeholder
    b.optional :readonly
```

```
    b.optional :step
    b.use :label, class: 'col-sm-3 form-
control-label'
    b.wrapper :grid_wrapper, tag: 'div',
class: 'col-sm-9' do |ba|
      ba.use :input, class: 'form-control-
range', error_class: 'is-invalid'
      ba.use :full_error, wrap_with: {
tag: 'div', class: 'invalid-feedback d-
block' }
      ba.use :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
    end
  end
  config.wrappers :inline_form, tag:
'span', error_class: 'form-group-invalid',
valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.use :placeholder
    b.optional :maxlength
    b.optional :minlength
    b.optional :pattern
    b.optional :min_max
    b.optional :readonly
    b.use :label, class: 'sr-only'
    b.use :input, class: 'form-control',
error_class: 'is-invalid'
    b.use :error, wrap_with: { tag: 'div',
class: 'invalid-feedback' }
    b.optional :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
  end
  config.wrappers :inline_boolean, tag:
'span', class: 'form-check flex-wrap
justify-content-start mr-sm-2',
error_class: 'form-group-invalid',
valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.optional :readonly
    b.use :input, class: 'form-check-
input', error_class: 'is-invalid'
    b.use :label, class: 'form-check-
label'
    b.use :error, wrap_with: { tag: 'div',
class: 'invalid-feedback' }
    b.optional :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
  end
  config.wrappers :custom_boolean, tag:
'fieldset', class: 'form-group',
error_class: 'form-group-invalid',
valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.optional :readonly
    b.wrapper :form_check_wrapper, tag:
'div', class: 'custom-control custom-
checkbox' do |bb|
      bb.use :input, class: 'custom-
control-input', error_class: 'is-invalid'
      bb.use :label, class: 'custom-
control-label'
      bb.use :full_error, wrap_with: {
tag: 'div', class: 'invalid-feedback' }
      bb.use :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
    end
  end
  config.wrappers :custom_boolean_switch,
tag: 'fieldset', class: 'form-group',
error_class: 'form-group-invalid',
valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.optional :readonly
    b.wrapper :form_check_wrapper, tag:
'div', class: 'custom-control custom-
checkbox-switch' do |bb|
      bb.use :input, class: 'custom-
control-input', error_class: 'is-invalid'
      bb.use :label, class: 'custom-
control-label'
      bb.use :full_error, wrap_with: {
tag: 'div', class: 'invalid-feedback' }
      bb.use :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
    end
  end
  config.wrappers :custom_collection,
item_wrapper_class: 'custom-control', tag:
'fieldset', class: 'form-group',
error_class: 'form-group-invalid',
valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.optional :readonly
    b.wrapper :legend_tag, tag: 'legend',
class: 'col-form-label pt-0' do |ba|
      ba.use :label_text
    end
    b.use :input, class: 'custom-control-
input', error_class: 'is-invalid'
    b.use :full_error, wrap_with: { tag:
'div', class: 'invalid-feedback d-block' }
    b.use :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
  end
  config.wrappers
:custom_collection_inline,
item_wrapper_class: 'custom-control
custom-control-inline', tag: 'fieldset',
class: 'form-group', error_class: 'form-
group-invalid', valid_class: 'form-group-
valid' do |b|
    b.use :html5
    b.optional :readonly
    b.wrapper :legend_tag, tag: 'legend',
class: 'col-form-label pt-0' do |ba|
      ba.use :label_text
    end
    b.use :input, class: 'custom-control-
input', error_class: 'is-invalid'
    b.use :full_error, wrap_with: { tag:
'div', class: 'invalid-feedback d-block' }
    b.use :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
  end
  config.wrappers :custom_file, tag:
'div', class: 'form-group', error_class:
'form-group-invalid', valid_class: 'form-
group-valid' do |b|
    b.use :html5
    b.use :placeholder
    b.optional :maxlength
    b.optional :minlength
    b.optional :readonly
    b.use :label, class: 'form-control-
label'
    b.wrapper :custom_file_wrapper, tag:
'div', class: 'custom-file' do |ba|
      ba.use :input, class: 'custom-file-
input', error_class: 'is-invalid'
      ba.use :label, class: 'custom-file-
label'
      ba.use :full_error, wrap_with: {
tag: 'div', class: 'invalid-feedback' }
    end
    b.use :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
  end
  config.wrappers :custom_multi_select,
tag: 'div', class: 'form-group',
error_class: 'form-group-invalid',
valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.optional :readonly
    b.use :label, class: 'form-control-
label'
    b.wrapper tag: 'div', class: 'd-flex
flex-row justify-content-between align-
items-center' do |ba|
      ba.use :input, class: 'custom-select
mx-1', error_class: 'is-invalid'
    end
    b.use :full_error, wrap_with: { tag:
'div', class: 'invalid-feedback d-block' }
    b.use :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
  end
  config.wrappers :custom_range, tag:
'div', class: 'form-group', error_class:
'form-group-invalid', valid_class: 'form-
group-valid' do |b|
    b.use :html5
    b.use :placeholder
    b.optional :readonly
    b.optional :step
    b.use :label, class: 'form-control-
label'
    b.use :input, class: 'custom-range',
error_class: 'is-invalid'
    b.use :full_error, wrap_with: { tag:
'div', class: 'invalid-feedback d-block' }
    b.use :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
  end
  config.wrappers :input_group, tag:
'div', class: 'form-group', error_class:
'form-group-invalid', valid_class: 'form-
group-valid' do |b|
    b.use :html5
    b.use :placeholder
    b.optional :maxlength
    b.optional :minlength
    b.optional :pattern
    b.optional :min_max
    b.optional :readonly
    b.use :label, class: 'form-control-
label'
    b.wrapper :input_group_tag, tag:
'div', class: 'input-group' do |ba|
      ba.optional :prepend
      ba.use :input, class: 'form-
control', error_class: 'is-invalid'
      ba.optional :append
    end
    b.use :full_error, wrap_with: { tag:
'div', class: 'invalid-feedback d-block' }
    b.use :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
  end
  config.wrappers :floating_labels_form,
tag: 'div', class: 'form-label-group',
error_class: 'form-group-invalid',
valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.use :placeholder
    b.optional :maxlength
    b.optional :minlength
    b.optional :pattern
    b.optional :min_max
    b.optional :readonly
    b.use :input, class: 'form-control',
error_class: 'is-invalid'
    b.use :label, class: 'form-control-
label'
    b.use :full_error, wrap_with: { tag:
'div', class: 'invalid-feedback' }
    b.use :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
  end
  config.wrappers :floating_labels_select,
tag: 'div', class: 'form-label-group',
error_class: 'form-group-invalid',
valid_class: 'form-group-valid' do |b|
    b.use :html5
    b.optional :readonly
    b.use :input, class: 'custom-select
custom-select-lg', error_class: 'is-
invalid'
    b.use :label, class: 'form-control-
label'
    b.use :full_error, wrap_with: { tag:
'div', class: 'invalid-feedback' }
    b.use :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
  end
  config.wrappers :table, tag: 'div' do
|b|
    b.use :html5
    b.use :placeholder
    b.optional :maxlength
    b.optional :minlength
    b.optional :pattern
    b.optional :min_max
    b.optional :readonly
    b.use :input, class: 'form-control',
error_class: 'is-invalid'
    b.use :full_error, wrap_with: { tag:
'div', class: 'invalid-feedback' }
    b.use :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
  end
  config.wrappers :table_file, tag: 'div'
do |b|
    b.use :html5
    b.use :placeholder
    b.optional :maxlength
    b.optional :minlength
    b.optional :readonly
    b.use :input, class: 'form-control-
file', error_class: 'is-invalid'
    b.use :full_error, wrap_with: { tag:
'div', class: 'invalid-feedback d-block' }
    b.use :hint, wrap_with: { tag:
'small', class: 'form-text text-muted' }
  end
  config.default_wrapper = :vertical_form
  config.wrapper_mappings = {
    boolean:       :vertical_boolean,
    check_boxes:   :vertical_collection,
    date:          :vertical_multi_select,
    datetime:      :vertical_multi_select,
    file:          :vertical_file,
    radio_buttons: :vertical_collection,
    range:         :vertical_range,
    time:          :vertical_multi_select
  }
end
Файл config/initializers/i18n.rb
Rails.application.configure do
  config.i18n.available_locales = %i[ru en
uk]
  config.i18n.default_locale = :uk
  config.i18n.load_path +=
Dir[Rails.root.join('config', 'locales',
'**', '*.{rb,yml}')]
end
Файл
config/initializers/filter_parameter_loggi
ng.rb
Rails.application.config.filter_parameters
+= %i[password auth_token access_token]
Файл config/puma.rb
threads_count = ENV.fetch
'RAILS_MAX_THREADS', 5
rails_env = ENV.fetch 'RAILS_ENV',
'development'
threads threads_count, threads_count
environment rails_env
if rails_env == 'production'
  bind ENV['SOCKET']
  pidfile ENV['PID']
else
  port ENV.fetch 'PORT', 3000
end
plugin :tmp_restart
Файл config/spring.rb
%w[
  .ruby-version
  .rbenv-vars
  tmp/restart.txt
  tmp/caching-dev.txt
].each { |path| Spring.watch(path) }
Файл config/i18n-tasks.yml
base_locale: en
data:
  read:
    - config/locales/%{locale}.yml
    - config/locales/**/*.%{locale}.yml
  write:
  external:
  yaml:
    write:
      line_width: -1
```

```
search:
  exclude:
    - app/assets/images
    - app/assets/fonts
    - app/assets/videos
Файл config/deploy/staging.rb
set :branch, :master
server 'codelabs.site:10016', user:
'user', roles: %i[app web db]
Файл config/deploy/production.rb
set :branch, :production
server 'codelabs.site:10014', user:
'user', roles: %i[app web db]
Файл .gitlab-ci.yml
stages:
  - test
  - deploy
variables:
  POSTGRES_PASSWORD: "postgres"
  DATABASE_URL:
"postgres://postgres:postgres@postgres/cod
elabs_test"
  REDIS_URL: "redis://redis/"
  APPLICATION_URL: "http://localhost/"
  SMTP_FROM: "codelabs@codelabs.local"
test:
  environment: test
  stage: test
  script:
    - gem update --system
    - gem update bundler --force
    - bundle config set path 'vendor/ruby'
    - bundle install
    - RAILS_ENV=test bundle exec rake
db:create
    - RAILS_ENV=test bundle exec rake
db:schema:load
    - bundle exec rake
  cache:
    paths:
      - vendor/ruby
  services:
    - postgres
    - redis
deploy_staging:
  image: ruby:alpine
  environment: development
  stage: deploy
  script:
    - apk add openssh-client
    - eval $(ssh-agent -s)
    - echo "${SSH_PRIVATE_KEY}" | tr -d
'\r' | ssh-add -
    - gem install capistrano capistrano-
rails
    - cap staging deploy
  only:
    - master
deploy_production:
  image: ruby:alpine
  environment: production
  stage: deploy
  script:
    - apk add openssh-client
    - eval $(ssh-agent -s)
    - echo "${SSH_PRIVATE_KEY}" | tr -d
'\r' | ssh-add -
    - gem install capistrano capistrano-
rails
    - cap production deploy
  only:
    - production
Файл package.json
{
  "name": "CodeLabs",
  "private": true,
  "dependencies": {
    "bs-custom-file-input": "^1.3.1",
    "toastr": "^2.1.4"
  }
}
Файл
db/migrate/20181013153232_add_worker_alive
_at_time.rb
class AddWorkerAliveAtTime <
ActiveRecord::Migration[5.2]
  def change
    add_column :workers, :alive_at,
:datetime
    add_index :workers, :alive_at
  end
end
Файл
db/migrate/20180807083719_add_limits.rb
class AddLimits <
ActiveRecord::Migration[5.2]
  def change
    add_column :compilers, :memory_a,
:float
    add_column :compilers, :memory_b,
:float
    add_column :compilers, :time_a, :float
    add_column :compilers, :time_b, :float
    add_column :problems, :memory_limit,
:float
    add_column :problems, :time_limit,
:float
    change_column_null :compilers,
:memory_a, false, 1
    change_column_null :compilers,
:memory_b, false, 0
    change_column_null :compilers,
:time_a, false, 1
    change_column_null :compilers,
:time_b, false, 0
    change_column_null :problems,
:memory_limit, false, 32*1024
    change_column_null :problems,
:time_limit, false, 100
  end
end
Файл
db/migrate/20181006142956_create_invites.r
b
class CreateInvites <
ActiveRecord::Migration[5.2]
  def change
    create_table :invites do |t|
      t.belongs_to :sender, foreign_key: {
to_table: :users }, index: true, null:
false
      t.belongs_to :receiver, foreign_key:
{ to_table: :users }, index: true, null:
false
      t.belongs_to :group, foreign_key:
true, index: true, null: false
      t.timestamps
    end
  end
end
Файл
db/migrate/20180705095143_create_examples.
rb
class CreateExamples <
ActiveRecord::Migration[5.2]
  def change
    create_table :examples do |t|
      t.string :input
      t.string :answer
      t.belongs_to :problem, foregin_key:
true, index: true, null: false
      t.timestamps
    end
  end
end
Файл
db/migrate/20180703110705_create_problem_t
ranslations.rb
class CreateProblemTranslations <
ActiveRecord::Migration[5.2]
  def change
    create_table :problem_translations do
|t|
      t.integer :language
      t.string :caption
      t.string :author
      t.text :text
      t.text :technical_text
      t.belongs_to :problem, index: true,
foreign_key: true, null: false
      t.timestamps
    end
  end
end
Файл
db/migrate/20181017174725_add_private_to_p
roblems.rb
class AddPrivateToProblems <
ActiveRecord::Migration[5.2]
  def change
    add_column :problems, :private,
:boolean, null: false, default: false
  end
end
Файл
db/migrate/20180807081936_add_checker_comp
iler_to_problems.rb
class AddCheckerCompilerToProblems <
ActiveRecord::Migration[5.2]
  def change
    add_reference :problems,
:checker_compiler, index: true
    add_foreign_key :problems, :compilers,
column: :checker_compiler_id
    change_column_null :problems,
:checker_compiler_id, false, 1
  end
end
Файл
db/migrate/20180623192309_enable_extention
_pgcrypto.rb
class EnableExtentionPgcrypto <
ActiveRecord::Migration[5.2]
  def change
    enable_extension 'pgcrypto'
  end
end
Файл
db/migrate/20180819160126_index_compilers_
on_name.rb
class IndexCompilersOnName <
ActiveRecord::Migration[5.2]
  def change
    add_index :compilers, :name
  end
end
Файл
db/migrate/20180704135114_add_user_id_to_s
ubmissions.rb
class AddUserIdToSubmissions <
ActiveRecord::Migration[5.2]
  def change
    Submission.destroy_all
    add_reference :submissions, :user,
index: true, foreign_key: true, null:
false
  end
end
Файл
db/migrate/20180623192259_create_users.rb
class CreateUsers <
ActiveRecord::Migration[5.2]
  def change
    create_table :users do |t|
      t.string :email, index: true
      t.string :password_digest
      t.timestamps
    end
  end
end
Файл
db/migrate/20180905090623_create_logs.rb
class CreateLogs <
ActiveRecord::Migration[5.2]
  def change
    create_table :logs do |t|
      t.text :data, null: false
      t.integer :type, null: false
      t.belongs_to :submission,
foregin_key: true, index: true, null:
false
      t.timestamps
    end
  end
end
Файл
db/migrate/20180807074747_create_compilers
.rb
class CreateCompilers <
ActiveRecord::Migration[5.2]
  def change
    create_table :compilers do |t|
      t.string :name, null: false
      t.string :version, null: false
      t.timestamps
    end
  end
end
Файл
db/migrate/20180624142413_add_profile_attr
s_to_users.rb
class AddProfileAttrsToUsers <
ActiveRecord::Migration[5.2]
  def change
    add_column :users, :name, :string
    add_column :users, :score, :bigint,
default: 0
    add_column :users, :skills, :string,
array: true, default: []
  end
end
Файл
db/migrate/20180819142135_add_administrato
r_to_users.rb
class AddAdministratorToUsers <
ActiveRecord::Migration[5.2]
  def change
    add_column :users, :administrator,
:boolean, default: false, null: false
  end
end
Файл
db/migrate/20180703110702_create_problems.
rb
class CreateProblems <
ActiveRecord::Migration[5.2]
  def change
    create_table :problems do |t|
      t.timestamps
    end
  end
end
Файл
db/migrate/20181013093828_create_workers.r
b
class CreateWorkers <
ActiveRecord::Migration[5.2]
  def change
    create_table :workers, id: :uuid do
|t|
      t.string :name, null: false
      t.string :ips, array: true, default:
[], null: false
      t.integer :api_version, null: false
      t.integer :api_type, null: false
      t.boolean :webhook_supported, null:
false
      t.string :task_status, array: true,
default: [], null: false
      t.integer :status, default: 0, null:
false
      t.timestamps
    end
  end
end
Файл
db/migrate/20180929141465_change_compiler_
status.rb
class ChangeCompilerStatus <
ActiveRecord::Migration[5.2]
  def change
    add_column :compilers, :status,
:integer, default: 0
```

```
    remove_column :compilers, :visible,
:boolean
  end
end
Файл
db/migrate/20180705153243_create_tag_trans
lations.rb
class CreateTagTranslations <
ActiveRecord::Migration[5.2]
  def change
    create_table :tag_translations do |t|
      t.integer :language
      t.string :name
      t.belongs_to :tag, index: true,
foreign_key: true, null: false
      t.timestamps
    end
  end
end
Файл
db/migrate/20180703115032_create_tests.rb
class CreateTests <
ActiveRecord::Migration[5.2]
  def change
    create_table :tests do |t|
      t.string :test_num
      t.belongs_to :problem, foregin_key:
true, index: true, null: false
      t.timestamps
    end
  end
end
Файл
db/migrate/20181209211125_add_problems_cou
nt_to_tags.rb
class AddProblemsCountToTags <
ActiveRecord::Migration[5.2]
  def change
    add_column :tags, :problems_count,
:integer, default: 0, null: false, index:
true
  end
end
Файл
db/migrate/20180929141405_change_users_ski
lls.rb
class ChangeUsersSkills <
ActiveRecord::Migration[5.2]
  def change
    remove_column :users, :skills,
:string, array: true
    add_column :users, :skills, :string
  end
end
Файл
db/migrate/20180703112139_add_foreign_key_
and_not_null_to_auth_tokens_user_id.rb
class
AddForeignKeyAndNotNullToAuthTokensUserId
< ActiveRecord::Migration[5.2]
  def change
    add_foreign_key :auth_tokens, :users
    change_column_null :auth_tokens,
:user_id, false
  end
end
Файл
db/migrate/20180703172255_create_test_requ
ests.rb
class CreateTestRequests <
ActiveRecord::Migration[5.2]
  def change
    create_table :test_requests do |t|
      t.integer :solution_compiler, null:
false
      t.belongs_to :problem, index: true,
foreign_key: true, null: false
      t.timestamps
    end
  end
end
Файл
db/migrate/20181016162627_remove_admnistra
tor_from_users.rb
class RemoveAdmnistratorFromUsers <
ActiveRecord::Migration[5.2]
  def change
    User.where(administrator:
true).update_all(roles: 7)
    remove_column :users, :administrator
  end
end
Файл
db/migrate/20181006100525_add_index_to_gro
ups_name.rb
class AddIndexToGroupsName <
ActiveRecord::Migration[5.2]
  def change
    add_index :groups, :name
  end
end
Файл
db/migrate/20190214091143_remove_groups_us
ers_and_invites.rb
class RemoveGroupsUsersAndInvites <
ActiveRecord::Migration[5.2]
  def change
    drop_table :groups_users
    drop_table :invites
  end
end
Файл
db/migrate/20181021190935_drop_groups_prob
lems.rb
class DropGroupsProblems <
ActiveRecord::Migration[5.2]
  def change
    drop_table :groups_problems
  end
end
Файл
db/migrate/20180704114305_rename_submissio
ns_solution_compiler_to_compiler.rb
class
RenameSubmissionsSolutionCompilerToCompile
r < ActiveRecord::Migration[5.2]
  def change
    rename_column :submissions,
:solution_compiler, :compiler
  end
end
Файл
db/migrate/20181017132550_create_confirmat
ion_requests.rb
class CreateConfirmationRequests <
ActiveRecord::Migration[5.2]
  def change
    create_table :confirmation_requests do
|t|
      t.belongs_to :user, foreign_key:
true, index: true, null: false
      t.integer :status, default: 0, null:
false
      t.timestamps
    end
  end
end
Файл
db/migrate/20180705104902_add_foreign_key_
examples_problems.rb
class AddForeignKeyExamplesProblems <
ActiveRecord::Migration[5.2]
  def change
    add_foreign_key :examples, :problems
  end
end
Файл
db/migrate/20181006094117_create_table_gro
ups_users.rb
class CreateTableGroupsUsers <
ActiveRecord::Migration[5.2]
  def change
    create_table :groups_users, id: false
do |t|
      t.belongs_to :group, foreign_key:
true, index: true, null: false
      t.belongs_to :user, foreign_key:
true, index: true, null: false
    end
  end
end
Файл
db/migrate/20180930141400_add_additional_u
ser_information.rb
class AddAdditionalUserInformation <
ActiveRecord::Migration[5.2]
  def change
    enable_extension "pg_trgm"
    add_column :users, :city, :string
    add_index :users, :city, using: :gist,
opclass: :gist_trgm_ops
    add_column :users, :institution,
:string
    add_index :users, :institution, using:
:gist, opclass: :gist_trgm_ops
  end
end
Файл
db/migrate/20181006092120_create_groups.rb
class CreateGroups <
ActiveRecord::Migration[5.2]
  def change
    create_table :groups do |t|
      t.string :name, null: false
      t.integer :visibility, null: false,
default: 0
      t.belongs_to :owner, foreign_key: {
to_table: :users }, index: true, null:
false
      t.timestamps
    end
  end
end
Файл
db/migrate/20180909160839_add_visible_to_c
ompilers.rb
class AddVisibleToCompilers <
ActiveRecord::Migration[5.2]
  def change
    add_column :compilers, :visible,
:boolean, default: false, null: false
    add_index :compilers, :visible
  end
end
Файл
db/migrate/20180807083125_add_log_to_resul
ts.rb
class AddLogToResults <
ActiveRecord::Migration[5.2]
  def change
    add_column :results, :log, :text
  end
end
Файл
db/migrate/20191103134959_add_password_rec
overy_token_to_users.rb
class AddPasswordRecoveryTokenToUsers <
ActiveRecord::Migration[5.2]
  def change
    add_column :users,
:password_recovery_token, :uuid
    add_index :users,
:password_recovery_token, unique: true
  end
end
Файл
db/migrate/20180807065900_create_results.r
b
class CreateResults <
ActiveRecord::Migration[5.2]
  def change
    create_table :results do |t|
      t.integer :status, null: false
      t.belongs_to :submission,
foregin_key: true, index: true, null:
false
      t.belongs_to :test, foregin_key:
true, index: true, null: false
      t.timestamps
    end
  end
end
Файл
db/migrate/20181006122214_add_description_
to_groups.rb
class AddDescriptionToGroups <
ActiveRecord::Migration[5.2]
  def change
    add_column :groups, :description,
:text
  end
end
Файл
db/migrate/20190911195729_add_index_users_
on_name.rb
class AddIndexUsersOnName <
ActiveRecord::Migration[5.2]
  def change
    add_index :users, :name, using: :gist,
opclass: :gist_trgm_ops
  end
end
Файл
db/migrate/20181008093116_add_status_to_in
vite.rb
class AddStatusToInvite <
ActiveRecord::Migration[5.2]
  def change
    add_column :invites, :status,
:integer, null: false, default: 0
  end
end
Файл
db/migrate/20180721222157_add_test_state_t
o_submissions.rb
class AddTestStateToSubmissions <
ActiveRecord::Migration[5.2]
  def change
    add_column :submissions, :test_state,
:integer, default: 0, null: false
    add_index :submissions, :test_state
  end
end
Файл
db/migrate/20181017181648_create_groups_pr
oblems.rb
class CreateGroupsProblems <
ActiveRecord::Migration[5.2]
  def change
    create_table :groups_problems, id:
false do |t|
      t.belongs_to :group, index: true,
foreign_key: true, null: false
      t.belongs_to :problem, index: true,
foreign_key: true, null: false
    end
  end
end
Файл
db/migrate/20180705153038_create_tags.rb
class CreateTags <
ActiveRecord::Migration[5.2]
  def change
    create_table :tags do |t|
      t.timestamps
    end
  end
end
Файл
db/migrate/20190214092557_create_membershi
ps.rb
class CreateMemberships <
ActiveRecord::Migration[5.2]
  def change
    create_table :memberships do |t|
      t.belongs_to :user, foreign_key:
true, index: true
      t.belongs_to :group, foreign_key:
true, index: true
      t.integer :state, null: false
      t.timestamps
    end
  end
```

```
end
Файл
db/migrate/20181021190739_create_sharings.
rb
class CreateSharings <
ActiveRecord::Migration[5.2]
  def change
    create_table :sharings do |t|
      t.belongs_to :group, index: true,
foreign_key: true, null: false
      t.belongs_to :problem, index: true,
foreign_key: true, null: false
      t.timestamps
    end
  end
end
Файл
db/migrate/20180703174334_add_foreign_key_
tests_problems.rb
class AddForeignKeyTestsProblems <
ActiveRecord::Migration[5.2]
  def change
    add_foreign_key :tests, :problems
  end
end
Файл
db/migrate/20180907085443_add_test_result_
to_submissions.rb
class AddTestResultToSubmissions <
ActiveRecord::Migration[5.2]
  def change
    add_column :submissions, :test_result,
:integer
  end
end
Файл
db/migrate/20180907083612_fix_not_null_for
_results_memoty_and_time.rb
class FixNotNullForResultsMemotyAndTime <
ActiveRecord::Migration[5.2]
  def change
    change_column_null :results, :memory,
false, 0
    change_column_null :results, :time,
false, 0
  end
end
Файл
db/migrate/20181124172711_make_results_tes
t_id_nullable.rb
class MakeResultsTestIdNullable <
ActiveRecord::Migration[5.2]
  def change
    change_column_null :results, :test_id,
true
  end
end
Файл
db/migrate/20181028160041_add_points_for_t
est.rb
class AddPointsForTest <
ActiveRecord::Migration[5.2]
  def change
    add_column :tests, :point, :integer,
null: false, default: 1
  end
end
Файл
db/migrate/20181014145628_add_owner_to_pro
blems.rb
class AddOwnerToProblems <
ActiveRecord::Migration[5.2]
  def change
    add_reference :problems, :user, index:
true, foreign_key: true
  end
end
Файл
db/migrate/20181028215626_add_index_tests_
on_num.rb
class AddIndexTestsOnNum <
ActiveRecord::Migration[5.2]
  def change
    add_index :tests, :num
  end
end
Файл
db/migrate/20180807073852_remove_compiler_
enum_from_submissions.rb
class RemoveCompilerEnumFromSubmissions <
ActiveRecord::Migration[5.2]
  def change
    remove_column :submissions, :compiler
  end
end
Файл
db/migrate/20181016154234_add_roles_to_use
rs.rb
class AddRolesToUsers <
ActiveRecord::Migration[5.2]
  def change
    add_column :users, :roles, :integer,
null: false, default: 0
  end
end
Файл
db/migrate/20191103110356_add_index_users_
on_lower_email.rb
class AddIndexUsersOnLowerEmail <
ActiveRecord::Migration[5.2]
  def change
    add_index :users, 'lower(email)'
  end
end
Файл
db/migrate/20180807075923_add_compiler_to_
submissions.rb
class AddCompilerToSubmissions <
ActiveRecord::Migration[5.2]
  def change
    add_reference :submissions, :compiler,
index: true
    add_foreign_key :submissions,
:compilers
    change_column_null :submissions,
:compiler_id, false, 1
  end
end
Файл
db/migrate/20191103110350_remove_index_use
rs_on_email.rb
class RemoveIndexUsersOnEmail <
ActiveRecord::Migration[5.2]
  def change
    remove_index :users, :email
  end
end
Файл
db/migrate/20180704110205_rename_table_tes
t_request_to_submissions.rb
class RenameTableTestRequestToSubmissions
< ActiveRecord::Migration[5.2]
  def change
    rename_table :test_requests,
:submissions
  end
end
Файл
db/migrate/20180705154704_create_problems_
tags.rb
class CreateProblemsTags <
ActiveRecord::Migration[5.2]
  def change
    create_table :problems_tags do |t|
      t.belongs_to :problem, index: true,
foreign_key: true, null: false
      t.belongs_to :tag, index: true,
foreign_key: true, null: false
      t.timestamps
    end
  end
end
Файл
db/migrate/20181013133854_add_username_to_
user.rb
class AddUsernameToUser <
ActiveRecord::Migration[5.2]
  def change
    add_column :users, :username, :string
    change_column_null :users, :username,
false, -> { 'gen_random_uuid()' }
    add_index :users, :username, using:
:gist, opclass: :gist_trgm_ops
  end
end
Файл
db/migrate/20180623192711_create_auth_toke
ns.rb
class CreateAuthTokens <
ActiveRecord::Migration[5.2]
  def change
    create_table :auth_tokens, id: :uuid
do |t|
      t.belongs_to :user, index: true
      t.timestamps
    end
  end
end
Файл
db/migrate/20180907082949_add_score_to_sub
missions.rb
class AddScoreToSubmissions <
ActiveRecord::Migration[5.2]
  def change
    add_column :submissions, :score,
:float
  end
end
Файл
db/migrate/20181028181705_add_max_score_fo
r_submission.rb
class AddMaxScoreForSubmission <
ActiveRecord::Migration[5.2]
  def change
    add_column :submissions, :max_score,
:float
  end
end
Файл
db/migrate/20180707184901_rename_tests_tes
t_num_to_num.rb
class RenameTestsTestNumToNum <
ActiveRecord::Migration[5.2]
  def change
    rename_column :tests, :test_num, :num
  end
end
Файл
db/migrate/20180725161854_add_fails_count_
to_submissions.rb
class AddFailsCountToSubmissions <
ActiveRecord::Migration[5.2]
  def change
    add_column :submissions, :fails_count,
:integer, default: 0, null: false
  end
end
Файл
db/migrate/20180807090413_add_memory_and_t
ime_to_results.rb
class AddMemoryAndTimeToResults <
ActiveRecord::Migration[5.2]
  def change
    add_column :results, :memory, :float,
null: false
    add_column :results, :time, :float,
null: false
  end
end
Файл
db/migrate/20180922075414_add_default_to_p
roblem_translations.rb
class AddDefaultToProblemTranslations <
ActiveRecord::Migration[5.2]
  def change
    add_column :problem_translations,
:default, :boolean, default: false, null:
false
  end
end
Файл
db/migrate/20181014155123_alive_cant_be_nu
ll.rb
class AliveCantBeNull <
ActiveRecord::Migration[5.2]
  def change
    change_column_null :workers,
:alive_at, false
  end
end
Файл
db/migrate/20180624132057_create_active_st
orage_tables.active_storage.rb
class CreateActiveStorageTables <
ActiveRecord::Migration[5.2]
  def change
    create_table :active_storage_blobs do
|t|
      t.string   :key,        null: false
      t.string   :filename,   null: false
      t.string   :content_type
      t.text     :metadata
      t.bigint   :byte_size,  null: false
      t.string   :checksum,   null: false
      t.datetime :created_at, null: false
      t.index [ :key ], unique: true
    end
    create_table
:active_storage_attachments do |t|
      t.string     :name,     null: false
      t.references :record,   null: false,
polymorphic: true, index: false
      t.references :blob,     null: false
      t.datetime :created_at, null: false
      t.index [ :record_type, :record_id,
:name, :blob_id ], name:
"index_active_storage_attachments_uniquene
ss", unique: true
    end
  end
end
Файл
db/migrate/20180929114039_add_real_time_li
mit_to_problems.rb
class AddRealTimeLimitToProblems <
ActiveRecord::Migration[5.2]
  def change
    add_column :problems,
:real_time_limit, :float
    change_column_null :problems,
:real_time_limit, false, 5000
  end
end
Файл db/schema.rb
ActiveRecord::Schema.define(version:
2019_11_03_134959) do
  enable_extension "pg_trgm"
  enable_extension "pgcrypto"
  enable_extension "plpgsql"
  create_table
"active_storage_attachments", force:
:cascade do |t|
    t.string "name", null: false
    t.string "record_type", null: false
    t.bigint "record_id", null: false
    t.bigint "blob_id", null: false
    t.datetime "created_at", null: false
    t.index ["blob_id"], name:
"index_active_storage_attachments_on_blob_
id"
    t.index ["record_type", "record_id",
"name", "blob_id"], name:
"index_active_storage_attachments_uniquene
ss", unique: true
  end
  create_table "active_storage_blobs",
force: :cascade do |t|
    t.string "key", null: false
    t.string "filename", null: false
    t.string "content_type"
    t.text "metadata"
    t.bigint "byte_size", null: false
    t.string "checksum", null: false
```

```
    t.datetime "created_at", null: false
    t.index ["key"], name:
"index_active_storage_blobs_on_key",
unique: true
  end
  create_table "auth_tokens", id: :uuid,
default: -> { "gen_random_uuid()" },
force: :cascade do |t|
    t.bigint "user_id", null: false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.index ["user_id"], name:
"index_auth_tokens_on_user_id"
  end
  create_table "compilers", force:
:cascade do |t|
    t.string "name", null: false
    t.string "version", null: false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.float "memory_a", null: false
    t.float "memory_b", null: false
    t.float "time_a", null: false
    t.float "time_b", null: false
    t.integer "status", default: 0
    t.index ["name"], name:
"index_compilers_on_name"
  end
  create_table "confirmation_requests",
force: :cascade do |t|
    t.bigint "user_id", null: false
    t.integer "status", default: 0, null:
false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.index ["user_id"], name:
"index_confirmation_requests_on_user_id"
  end
  create_table "examples", force: :cascade
do |t|
    t.string "input"
    t.string "answer"
    t.bigint "problem_id", null: false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.index ["problem_id"], name:
"index_examples_on_problem_id"
  end
  create_table "groups", force: :cascade
do |t|
    t.string "name", null: false
    t.integer "visibility", default: 0,
null: false
    t.bigint "owner_id", null: false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.text "description"
    t.index ["name"], name:
"index_groups_on_name"
    t.index ["owner_id"], name:
"index_groups_on_owner_id"
  end
  create_table "logs", force: :cascade do
|t|
    t.text "data", null: false
    t.integer "type", null: false
    t.bigint "submission_id", null: false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.index ["submission_id"], name:
"index_logs_on_submission_id"
  end
  create_table "memberships", force:
:cascade do |t|
    t.bigint "user_id"
    t.bigint "group_id"
    t.integer "state", null: false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.index ["group_id"], name:
"index_memberships_on_group_id"
    t.index ["user_id"], name:
"index_memberships_on_user_id"
  end
  create_table "problem_translations",
force: :cascade do |t|
    t.integer "language"
    t.string "caption"
    t.string "author"
    t.text "text"
    t.text "technical_text"
    t.bigint "problem_id", null: false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.boolean "default", default: false,
null: false
    t.index ["problem_id"], name:
"index_problem_translations_on_problem_id"
  end
  create_table "problems", force: :cascade
do |t|
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.bigint "checker_compiler_id", null:
false
    t.float "memory_limit", null: false
    t.float "time_limit", null: false
    t.float "real_time_limit", null: false
    t.bigint "user_id"
    t.boolean "private", default: false,
null: false
    t.index ["checker_compiler_id"], name:
"index_problems_on_checker_compiler_id"
    t.index ["user_id"], name:
"index_problems_on_user_id"
  end
  create_table "problems_tags", force:
:cascade do |t|
    t.bigint "problem_id", null: false
    t.bigint "tag_id", null: false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.index ["problem_id"], name:
"index_problems_tags_on_problem_id"
    t.index ["tag_id"], name:
"index_problems_tags_on_tag_id"
  end
  create_table "results", force: :cascade
do |t|
    t.integer "status", null: false
    t.bigint "submission_id", null: false
    t.bigint "test_id"
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.text "log"
    t.float "memory", null: false
    t.float "time", null: false
    t.index ["submission_id"], name:
"index_results_on_submission_id"
    t.index ["test_id"], name:
"index_results_on_test_id"
  end
  create_table "sharings", force: :cascade
do |t|
    t.bigint "group_id", null: false
    t.bigint "problem_id", null: false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.index ["group_id"], name:
"index_sharings_on_group_id"
    t.index ["problem_id"], name:
"index_sharings_on_problem_id"
  end
  create_table "submissions", force:
:cascade do |t|
    t.bigint "problem_id", null: false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.bigint "user_id", null: false
    t.integer "test_state", default: 0,
null: false
    t.integer "fails_count", default: 0,
null: false
    t.bigint "compiler_id", null: false
    t.float "score"
    t.integer "test_result"
    t.float "max_score"
    t.index ["compiler_id"], name:
"index_submissions_on_compiler_id"
    t.index ["problem_id"], name:
"index_submissions_on_problem_id"
    t.index ["test_state"], name:
"index_submissions_on_test_state"
    t.index ["user_id"], name:
"index_submissions_on_user_id"
  end
  create_table "tag_translations", force:
:cascade do |t|
    t.integer "language"
    t.string "name"
    t.bigint "tag_id", null: false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.index ["tag_id"], name:
"index_tag_translations_on_tag_id"
  end
  create_table "tags", force: :cascade do
|t|
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.integer "problems_count", default:
0, null: false
  end
  create_table "tests", force: :cascade do
|t|
    t.string "num"
    t.bigint "problem_id", null: false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.integer "point", default: 1, null:
false
    t.index ["num"], name:
"index_tests_on_num"
    t.index ["problem_id"], name:
"index_tests_on_problem_id"
  end
  create_table "users", force: :cascade do
|t|
    t.string "email"
    t.string "password_digest"
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.string "name"
    t.bigint "score", default: 0
    t.string "skills"
    t.string "city"
    t.string "institution"
    t.string "username", null: false
    t.integer "roles", default: 0, null:
false
    t.uuid "password_recovery_token"
    t.index "lower((email)::text)", name:
"index_users_on_lower_email"
    t.index ["city"], name:
"index_users_on_city", opclass:
:gist_trgm_ops, using: :gist
    t.index ["institution"], name:
"index_users_on_institution", opclass:
:gist_trgm_ops, using: :gist
    t.index ["name"], name:
"index_users_on_name", opclass:
:gist_trgm_ops, using: :gist
    t.index ["password_recovery_token"],
name:
"index_users_on_password_recovery_token",
unique: true
    t.index ["username"], name:
"index_users_on_username", opclass:
:gist_trgm_ops, using: :gist
  end
  create_table "workers", id: :uuid,
default: -> { "gen_random_uuid()" },
force: :cascade do |t|
    t.string "name", null: false
    t.string "ips", default: [], null:
false, array: true
    t.integer "api_version", null: false
    t.integer "api_type", null: false
    t.boolean "webhook_supported", null:
false
    t.string "task_status", default: [],
null: false, array: true
    t.integer "status", default: 0, null:
false
    t.datetime "created_at", null: false
    t.datetime "updated_at", null: false
    t.datetime "alive_at", null: false
    t.index ["alive_at"], name:
"index_workers_on_alive_at"
  end
  add_foreign_key "auth_tokens", "users"
  add_foreign_key "confirmation_requests",
"users"
  add_foreign_key "examples", "problems"
  add_foreign_key "groups", "users",
column: "owner_id"
  add_foreign_key "memberships", "groups"
  add_foreign_key "memberships", "users"
  add_foreign_key "problem_translations",
"problems"
  add_foreign_key "problems", "compilers",
column: "checker_compiler_id"
  add_foreign_key "problems", "users"
  add_foreign_key "problems_tags",
"problems"
  add_foreign_key "problems_tags", "tags"
  add_foreign_key "sharings", "groups"
  add_foreign_key "sharings", "problems"
  add_foreign_key "submissions",
"compilers"
  add_foreign_key "submissions",
"problems"
  add_foreign_key "submissions", "users"
  add_foreign_key "tag_translations",
"tags"
  add_foreign_key "tests", "problems"
end
```

Файл db/seeds.rb

```
Compiler.create id: 1, name: 'g++',
version: '6.3.0-1', memory_a: 1, memory_b:
0, time_a: 1, time_b: 0, status: :public
```

Файл spec/models/tag_spec.rb

```
require 'rails_helper'
RSpec.describe Tag, type: :model do
  it { should
have_one(:translation).conditions(language
:
I18n.locale).class_name('TagTranslation')
}
  it { should
have_many(:translations).dependent(:destro
y).class_name('TagTranslation') }
  it { should
have_many(:problems_tags).dependent(:destr
oy) }
  it { should
have_many(:problems).through(:problems_tag
s) }
  it { should
delegate_method(:name).to(:translation) }
  describe '.default_scope' do
    subject { described_class.all }
    let(:expected) {
described_class.unscoped.includes
:translation }
    its(:includes_values) { should eq
expected.includes_values }
    its(:to_sql) { should eq
expected.to_sql }
  end
end
```

Файл spec/models/group_spec.rb

```
require 'rails_helper'
RSpec.describe Group, type: :model do
  it { should validate_presence_of :name }
  it { should validate_presence_of
:visibility }
  it { should
belong_to(:owner).class_name('User') }
  it { should
have_many(:sharings).dependent(:destroy) }
```

```
  it { should
have_many(:memberships).dependent(:destroy
) }
  it { should
have_many(:pending_memberships).class_name
('Membership') }
  it { should
have_many(:accepted_memberships).condition
s(state:
:accepted).class_name('Membership') }
  it { should
have_many(:problems).through(:sharings).or
der(:id) }
  it { should
have_many(:pending_users).through(:pending
_memberships).source(:user).class_name('Us
er') }
  it { should
have_many(:accepted_users).through(:accept
ed_memberships).source(:user).class_name('
User') }
  it { should
have_many(:submissions).through(:accepted_
users) }
  it { should
define_enum_for(:visibility).with_values(p
rivate: 0, moderated: 1, public:
2).with_prefix }
end
Файл spec/models/log_spec.rb
require 'rails_helper'
RSpec.describe Log, type: :model do
  it {
expect(described_class.inheritance_column)
.to eq '' }
  it { should validate_presence_of :data }
  it { should belong_to :submission }
  it { should
define_enum_for(:type).with_values(source:
0, checker: 1) }
  it { should
delegate_method(:user).to(:submission).wit
h_prefix }
  it { should
delegate_method(:problem_user).to(:submiss
ion).with_prefix }
end
Файл spec/models/tag_translation_spec.rb
require 'rails_helper'
RSpec.describe TagTranslation, type:
:model do
  it { should validate_presence_of
:language }
  it { should validate_presence_of :name }
  it { should belong_to :tag }
  it { should
define_enum_for(:language).with_values(I18
n.available_locales) }
end
Файл spec/models/compiler_spec.rb
require 'rails_helper'
RSpec.describe Compiler, type: :model do
  it { should validate_presence_of :name }
  it { should validate_presence_of
:version }
  it { should validate_presence_of :status
}
  it { should validate_presence_of
:memory_a }
  it { should validate_presence_of
:memory_b }
  it { should validate_presence_of :time_a
}
  it { should validate_presence_of :time_b
}
  it { should validate_numericality_of
:memory_a }
  it { should validate_numericality_of
:memory_b }
  it { should validate_numericality_of
:time_a }
  it { should validate_numericality_of
:time_b }
  it { should
define_enum_for(:status).with_values(in_te
st: 0, reserved: 1, public: 2).with_prefix
}
end
Файл spec/models/example_spec.rb
require 'rails_helper'
RSpec.describe Example, type: :model do
  it { should validate_presence_of :input
}
  it { should validate_presence_of :answer
}
  it { should belong_to :problem }
end
Файл spec/models/submission_spec.rb
require 'rails_helper'
RSpec.describe Submission, type: :model do
  it { should belong_to :problem }
  it { should belong_to :user }
  it { should belong_to :compiler }
  it { should have_one :source_attachment
}
  it { should
have_many(:results).dependent(:destroy) }
  it { should
have_many(:logs).dependent(:destroy) }
  it { should
define_enum_for(:test_state).with_values(p
ending: 0, in_progress: 1, done: 2,
failed: 3).with_prefix }
  it { should
define_enum_for(:test_result).with_values(
ok: 0, compiler_error: 1).with_prefix }
  it { should
delegate_method(:as_json).to(:decorate) }
  it { should
delegate_method(:user).to(:problem).with_p
refix }
  it { should
delegate_method(:private?).to(:problem).wi
th_prefix }
  it { should
callback(:update_standings).after(:commit)
}
  it { should have_state :pending }
  it { should
transition_from(:pending).to(:in_progress)
.on_event(:take) }
  it { should
transition_from(:in_progress).to(:done).on
_event(:release) }
  describe '#fail' do
    before { expect(subject).to
receive(:increment!).with(:fails_count) }
    context do
      subject { stub_model
described_class, fails_count: 0 }
      before { expect(subject).to
receive_message_chain(:results,
:delete_all) }
      it { should
transition_from(:in_progress).to(:pending)
.on_event(:fail) }
    end
    context do
      subject { stub_model
described_class, fails_count: 5 }
      it { should
transition_from(:in_progress).to(:failed).
on_event(:fail) }
    end
  end
  describe '#source_must_be_attached' do
    before { allow(subject).to
receive(:source).and_return(source) }
    let(:call) { -> { subject.send
:source_must_be_attached } }
    context do
      let(:source) { double attached?:
true }
      it { expect(&call).to_not change {
subject.errors.details } }
    end
    context do
      let(:source) { double attached?:
false }
      it { expect(&call).to change {
subject.errors.details[:source] }.to
[{error: :blank}] }
    end
  end
end
Файл spec/models/worker_spec.rb
require 'rails_helper'
RSpec.describe Worker, type: :model do
  it { should validate_presence_of :name }
  it { should validate_presence_of :ips }
  it { should validate_presence_of :status
}
  it { should validate_presence_of
:api_type }
  it { should validate_presence_of
:alive_at }
  it { should validate_presence_of
:api_version }
  it { should
validate_numericality_of(:api_version).onl
y_integer }
  it { should_not
allow_value(nil).for(:webhook_supported) }
  it { should
define_enum_for(:api_type).with_values(HTT
P: 0, WS: 1) }
  it { should
define_enum_for(:status).with_values(disab
led: 0, ok: 1, failed: 2, stale: 3,
stopped: 4) }
  it { should
delegate_method(:as_json).to(:decorate) }
end
Файл spec/models/membership_spec.rb
require 'rails_helper'
RSpec.describe Membership, type: :model do
  it { should validate_presence_of :state
}
  pending { should
validate_uniqueness_of(:user).scoped_to(:g
roup) }
  it { should belong_to(:user).required }
  it { should belong_to(:group).required }
  it { should
define_enum_for(:state).with_values(%i[req
uested invited accepted]).with_prefix }
  describe '#user_must_not_be_group_owner'
do
    fixtures :users
    subject { stub_model described_class,
group: group, user: users(:two) }
    before { subject.valid? }
    context do
      let(:group) { nil }
      its('errors.details') { should_not
include :user }
    end
    context do
      let(:group) { stub_model Group,
owner: users(:one) }
      its('errors.details') { should_not
include :user }
    end
    context do
      let(:group) { stub_model Group,
owner: users(:two) }
      its('errors.details') { should
include user: [error: :taken] }
    end
  end
end
Файл spec/models/sharing_spec.rb
require 'rails_helper'
RSpec.describe Sharing, type: :model do
  pending { should
validate_uniqueness_of(:problem_id).scoped
_to(:group_id) }
  it { should belong_to :problem }
  it { should belong_to :group }
  it { should
delegate_method(:owner).to(:group).with_pr
efix }
  it { should
delegate_method(:user).to(:problem).with_p
refix }
end
Файл spec/models/test_spec.rb
require 'rails_helper'
RSpec.describe Test, type: :model do
  it { should validate_presence_of :num }
  it { should validate_presence_of :point
}
  it { should
validate_numericality_of(:point).only_inte
ger }
  pending { should
validate_uniqueness_of(:num).scoped_to(:pr
oblem_id) }
  it { should belong_to(:problem).touch }
  it { should have_one :input_attachment }
  it { should have_one :answer_attachment
}
  it { should
have_many(:results).dependent(:nullify) }
  it { should
delegate_method(:as_json).to(:decorate) }
  describe '.default_scope' do
    subject { described_class.all }
    let(:expected) {
described_class.unscoped.with_attached_inp
ut.with_attached_answer.order :num }
    its(:includes_values) { should eq
expected.includes_values }
    its(:to_sql) { should eq
expected.to_sql }
  end
end
Файл spec/models/user_spec.rb
require 'rails_helper'
RSpec.describe User, type: :model do
  fixtures :users
  subject { users(:one) }
  it { should validate_presence_of :email
}
  it { should
validate_uniqueness_of(:email).case_insens
itive }
  it { should validate_presence_of
:username }
  it { should
validate_uniqueness_of(:username).case_ins
ensitive }
  it { should have_one :avatar_attachment
}
  it { should
have_one(:confirmation_request).dependent(
:destroy) }
  it { should
have_many(:problems).dependent(:nullify) }
  it { should
have_many(:auth_tokens).dependent(:destroy
) }
  it { should
have_many(:submissions).dependent(:destroy
) }
  it { should
have_many(:owned_groups).class_name('Group
').with_foreign_key(:owner_id).dependent(:
destroy) }
  it { should
have_many(:pending_memberships).class_name
('Membership').dependent(:destroy) }
  it { should
have_many(:accepted_memberships).condition
```

```
s(state:
:accepted).class_name('Membership').depend
ent(:destroy) }
  it { should
have_many(:pending_groups).through(:pendin
g_memberships).source(:group).class_name('
Group') }
  it { should
have_many(:accepted_groups).through(:accep
ted_memberships).source(:group).class_name
('Group') }
  it { should
have_many(:sharings).through(:accepted_gro
ups) }
  it { should
have_many(:shared_problems).through(:shari
ngs).source(:problem).class_name('Problem'
) }
  it { should have_secure_password }
  it { should allow_values(%i[confirmed
moderator administrator]).for(:roles) }
  it { should
callback(:send_email).after(:commit).on(:c
reate) }
  it { should
delegate_method(:as_json).to(:decorate) }
  %i[confirmed moderator
administrator].each do |value|
    describe "##{ value }?" do
      before { expect(subject).to
receive(:roles?).with(value).and_return(:r
esult) }
      its("#{ value }?") { should eq
:result }
    end
  end
  describe '#send_email' do
    it do
      expect(UserMailer).to
receive(:email).with(subject) do
        double.tap { |a| expect(a).to
receive(:deliver_later) }
      end
    end
    after { subject.send :send_email }
  end
end
Файл spec/models/auth_token_spec.rb
require 'rails_helper'
RSpec.describe AuthToken, type: :model do
  it { should belong_to :user }
end
Файл
spec/models/problem_translation_spec.rb
require 'rails_helper'
RSpec.describe ProblemTranslation, type:
:model do
  it { should validate_presence_of
:language }
  it { should validate_presence_of
:caption }
  it { should validate_presence_of :author
}
  it { should validate_presence_of :text }
  it { should validate_presence_of
:technical_text }
  pending { should
validate_uniqueness_of(:language).scoped_t
o(:problem_id) }
  it { should belong_to :problem }
  it { should
define_enum_for(:language).with_values(I18
n.available_locales) }
end
Файл spec/models/problems_tag_spec.rb
require 'rails_helper'
RSpec.describe ProblemsTag, type: :model
do
  it { should belong_to :problem }
  it { should
belong_to(:tag).counter_cache(:problems_co
unt) }
end
Файл spec/models/problem_spec.rb
require 'rails_helper'
RSpec.describe Problem, type: :model do
  it { should validate_presence_of
:memory_limit }
  it { should validate_presence_of
:time_limit }
  it { should validate_presence_of
:real_time_limit }
  it { should validate_numericality_of
:memory_limit }
  it { should validate_numericality_of
:time_limit }
  it { should validate_numericality_of
:real_time_limit }
  it { should_not
allow_value(nil).for(:private) }
  it { should
allow_value(true).for(:private) }
  it { should
allow_value(false).for(:private) }
  it { should belong_to(:user).optional }
  it { should
belong_to(:checker_compiler).class_name('C
ompiler') }
  it { should have_one
:checker_source_attachment }
  it { should
have_one(:translation).conditions(language
:
I18n.locale).class_name('ProblemTranslatio
n') }
  it { should
have_one(:default_translation).conditions(
default:
true).class_name('ProblemTranslation') }
  it { should
have_many(:translations).dependent(:destro
y).class_name('ProblemTranslation') }
  it { should
have_many(:tests).dependent(:destroy) }
  it { should
have_many(:examples).dependent(:destroy) }
  it { should
have_many(:submissions).dependent(:destroy
) }
  it { should
have_many(:sharings).dependent(:destroy) }
  it { should
have_many(:problems_tags).dependent(:destr
oy) }
  it { should
have_many(:groups).through(:sharings) }
  it { should
have_many(:tags).through(:problems_tags) }
  it { should
delegate_method(:as_json).to(:decorate) }
  it { should
delegate_method(:caption).to(:decorate) }
  describe '#to_param' do
    subject { stub_model described_class,
id: 621 }
    before { expect(subject).to
receive(:caption).and_return('N-digit
numbers') }
    its(:to_param) { should eq '621-n-
digit-numbers' }
  end
  describe '.default_scope' do
    subject { described_class.all }
    let(:expected) {
described_class.unscoped.includes
:translation, :default_translation }
    its(:includes_values) { should eq
expected.includes_values }
    its(:to_sql) { should eq
expected.to_sql }
  end
end
Файл
spec/models/confirmation_request_spec.rb
require 'rails_helper'
RSpec.describe ConfirmationRequest, type:
:model do
  it { should belong_to :user }
  it { should
define_enum_for(:status).with_values(pendi
ng: 0, accepted: 1, rejected: 2) }
  it { should
delegate_method(:name).to(:user).with_pref
ix }
  it { should
delegate_method(:username).to(:user).with_
prefix }
end
Файл spec/models/result_spec.rb
require 'rails_helper'
RSpec.describe Result, type: :model do
  it { should validate_presence_of :status
}
  it { should validate_presence_of :memory
}
  it { should validate_presence_of :time }
  it { should validate_numericality_of
:memory }
  it { should validate_numericality_of
:time }
  it { should belong_to :submission }
  it { should belong_to :test }
  it do
    should
define_enum_for(:status).with_values \
      ok: 0,
      wrong_answer: 1,
      presentation_error: 2,
      fail: 3,
      dirt: 4,
      points: 5,
      bad_test: 6,
      unexpected_eof: 8,
      runtime_error: 10,
      memory_limit_exceded: 14,
      time_limit_exceded: 15,
      partilly_correct: 16
  end
  it { should
delegate_method(:num).to(:test).with_prefi
x.allow_nil }
end
Файл
spec/support/action_controller_parameters.
rb
module ActionControllerParameters
  def acp params
    ActionController::Parameters.new
params
  end
end
Файл
spec/support/shoulda/matchers/action_contr
oller/render_template_with_status_matcher.
rb
module Shoulda
  module Matchers
    module ActionController
      class RenderTemplateMatcher
        def with_status status

RenderTemplateWithStatusMatcher.new
@options, status, @message, @context
        end
      end
      class
RenderTemplateWithStatusMatcher
        def initialize template, status,
message, context
          @render_template =
RenderTemplateMatcher.new template,
message, context
          @respond_with =
RespondWithMatcher.new status
        end
        def matches? controller
          @controller = controller
          renders_template? &&
responds_with?
        end
        def description
          "#{ @render_template.description
} and #{ @respond_with.description }"
        end
        def failure_message
          @render_template.failure_message
|| @respond_with.failure_message
        end
        def failure_message_when_negated
          "Did not expect to render #{
template } and respond with #{ status }"
        end
        private
        def renders_template?
          @render_template.matches?
@controller
        end
        def responds_with?
          @respond_with.matches?
@controller
        end
        def template

@render_template.instance_variable_get
:@template
        end
        def status

@respond_with.instance_variable_get
:@status
        end
      end
    end
  end
end
Файл
spec/support/shared_examples/controller.rb
RSpec.shared_examples :show do |params|
  let(:default_params) { { anonymous:
false, unauthorized: false, format: :html,
params: { id: 1 } } }
  let(:resource) { double }
  include_examples :parse_params, params
  include_examples :authenticate_user
  include_examples :authorize_resource
  describe '#show' do
    before { allow(subject).to
receive(:resource).and_return(resource) }
    before { get :show, params:
request_params, format: format }
    it { should render_template :show }
  end
end
RSpec.shared_examples :new do |params|
  let(:default_params) { { anonymous:
false, unauthorized: false, format: :html,
params: {} } }
  let(:resource) { :resource }
  include_examples :parse_params, params
  include_examples :authenticate_user
  include_examples :authorize_resource
  describe '#new' do
    before { allow(subject).to
receive(:resource).and_return(resource) }
    before { expect(subject).to
receive(:initialize_resource) }
    before { get :new, params:
request_params, format: format }
    it { should render_template :new }
  end
end
RSpec.shared_examples :create do |params|
  let(:default_params) { { anonymous:
false, unauthorized: false, format: :html,
params: {} } }
  include_examples :parse_params, params
  include_examples :authenticate_user
  include_examples :authorize_resource
  describe '#create' do
    before { allow(subject).to
receive(:resource).and_return(resource) }
```

```
    before { expect(subject).to
receive(:build_resource) }
    context do
      before { expect(resource).to
receive(:save).and_return(true) }
      before { post :create, params:
request_params, format: format }
      it { success.call }
    end
    context do
      before { expect(resource).to
receive(:save).and_return(false) }
      before { post :create, params:
request_params, format: format }
      it { failure.call }
    end
  end
end
RSpec.shared_examples :edit do |params|
  let(:default_params) { { anonymous:
false, unauthorized: false, format: :html,
params: { id: 1 } } }
  let(:resource) { :resource }
  include_examples :parse_params, params
  include_examples :authenticate_user
  include_examples :authorize_resource
  describe '#edit' do
    before { allow(subject).to
receive(:resource).and_return(resource) }
    before { get :edit, params:
request_params, format: format }
    it { should render_template :edit }
  end
end
RSpec.shared_examples :update do |params|
  let(:default_params) { { anonymous:
false, unauthorized: false, format: :html,
params: { id: 1 } } }
  include_examples :parse_params, params
  include_examples :authenticate_user
  include_examples :authorize_resource
  describe '#update' do
    before { allow(subject).to
receive(:resource).and_return(resource) }
    before { expect(subject).to
receive(:resource_params).and_return(:reso
urce_params) }
    context do
      before { expect(resource).to
receive(:update).with(:resource_params).an
d_return(true) }
      before { patch :update, params:
request_params, format: format }
      it { success.call }
    end
    context do
      before { expect(resource).to
receive(:update).with(:resource_params).an
d_return(false) }
      before { patch :update, params:
request_params, format: format }
      it { failure.call }
    end
  end
end
RSpec.shared_examples :destroy do |params|
  let(:default_params) { { anonymous:
false, unauthorized: false, format: :html,
params: { id: 1 } } }
  let(:resource) { double }
  include_examples :parse_params, params
  include_examples :authenticate_user
  include_examples :authorize_resource
  describe '#destroy' do
    before { allow(subject).to
receive(:resource).and_return(resource) }
    before { expect(resource).to
receive(:destroy).and_return(true) }
    before { delete :destroy, params:
request_params, format: format }
    it { success.call }
  end
end
RSpec.shared_examples :index do |params|
  let(:default_params) { { anonymous:
false, unauthorized: false, format: :html,
params: {} } }
  include_examples :parse_params, params
  include_examples :authenticate_user
  describe '#index' do
    before { expect(subject).to
receive(:authorize_collection).and_return(
true) unless unauthorized }
    before { get :index, params:
request_params, format: format }
    it { should render_template :index }
  end
end
RSpec.shared_examples :parse_params do
|params|
  let(:anonymous) { (params &&
params[:anonymous]) ||
default_params[:anonymous] }
  let(:unauthorized) { (params &&
params[:unauthorized]) ||
default_params[:unauthorized] }
  let(:format) { (params &&
params[:format]) ||
default_params[:format] }
  let(:request_params) { (params &&
params[:params]) ||
default_params[:params] }
  let(:policy) { (params &&
params[:policy]) ||
default_params[:policy] }
end
RSpec.shared_examples :authenticate_user
do
  before { expect(subject).to
receive(:authenticate!) unless anonymous }
end
RSpec.shared_examples :authorize_resource
do
  before { expect(subject).to
receive(:authorize).with(resource).and_ret
urn(true) unless unauthorized }
end
Файл spec/spec_helper.rb
RSpec.configure do |config|
  config.expect_with :rspec do
|expectations|

expectations.include_chain_clauses_in_cust
om_matcher_descriptions = true
  end
  config.mock_with :rspec do |mocks|
    mocks.verify_partial_doubles = true
  end
  config.shared_context_metadata_behavior
= :apply_to_host_groups
end
Файл spec/policies/avatar_policy_spec.rb
require 'rails_helper'
RSpec.describe AvatarPolicy do
  subject { described_class }
  fixtures :users
  let(:resource) { Avatar.new user:
users(:one) }
  permissions :create?, :destroy? do
    it { should permit users(:one),
resource }
    it { should_not permit nil, resource }
    it { should_not permit users(:two),
resource }
  end
end
Файл
spec/policies/membership_policy_spec.rb
require 'rails_helper'
RSpec.describe MembershipPolicy do
  subject { described_class }
  fixtures :users, :groups, :memberships
  permissions :destroy? do
    it { should_not permit nil, double }
    it { should_not permit users(:three),
memberships(:two) }
    it { should permit users(:one),
memberships(:two) }
    it { should permit users(:two),
memberships(:two) }
  end
  permissions :create? do
    it { should_not permit nil, double }
    context do
      let(:resource) { stub_model
Membership, group: stub_model(Group),
user: stub_model(User) }
      it { should permit double, resource
}
    end
    context do
      let(:resource) { stub_model
Membership, group: stub_model(Group),
state: :requested }
      it { should permit double, resource
}
    end
    context do
      let(:resource) { stub_model
Membership, state: :invited, user:
stub_model(User) }
      it { should permit double, resource
}
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
stub_model(User), state: :invited }
      it { should permit users(:one),
resource }
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
stub_model(User), state: :invited }
      before { resource.group.visibility =
:public }
      it { should permit users(:two),
resource }
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
stub_model(User), state: :invited }
      before { resource.group.visibility =
:private }
      it { should_not permit users(:two),
resource }
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
stub_model(User), state: :invited }
      before { resource.group.visibility =
:moderated }
      it { should_not permit users(:two),
resource }
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
users(:one), state: :requested }
      before { resource.group.visibility =
:moderated }
      it { should permit users(:one),
resource }
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
users(:one), state: :requested }
      before { resource.group.visibility =
:moderated }
      it { should_not permit users(:two),
resource }
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
users(:one), state: :requested }
      before { resource.group.visibility =
:private }
      it { should_not permit users(:one),
resource }
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
users(:one), state: :requested }
      before { resource.group.visibility =
:public }
      it { should_not permit users(:one),
resource }
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
users(:one), state: :accepted }
      before { resource.group.visibility =
:public }
      it { should permit users(:one),
resource }
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
users(:one), state: :accepted }
      before { resource.group.visibility =
:public }
      it { should_not permit users(:two),
resource }
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
users(:one), state: :accepted }
      before { resource.group.visibility =
:private }
      it { should_not permit users(:one),
resource }
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
users(:one), state: :accepted }
      before { resource.group.visibility =
:moderated }
      it { should_not permit users(:one),
resource }
    end
  end
  permissions :update? do
    it { should_not permit nil, double }
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
users(:three), state: :requested }
      it { should permit users(:one),
resource }
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
users(:three), state: :requested }
      it { should_not permit
users(:three), resource }
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
users(:three), state: :invited }
      it { should permit users(:three),
resource }
    end
    context do
      let(:resource) { stub_model
Membership, group: groups(:one), user:
users(:three), state: :invited }
      it { should_not permit users(:one),
resource }
    end
```

```
  end
  describe '#index?' do
    context do
      subject { described_class.new nil,
double }
      its(:index?) { should be false }
    end
    context do
      subject { described_class.new
users(:one), memberships(:one), parent:
groups(:one) }
      its(:index?) { should be true }
    end
    context do
      subject { described_class.new
users(:two), memberships(:one), parent:
groups(:one) }
      its(:index?) { should be false }
    end
    context do
      subject { described_class.new
users(:two), memberships(:one) }
      its(:index?) { should be true }
    end
  end
end
Файл spec/policies/user_policy_spec.rb
require 'rails_helper'
RSpec.describe UserPolicy do
  subject { described_class }
  fixtures :users
  let(:resource) { users :one }
  permissions :index? do
    it { should permit nil, resource }
    it { should permit double, resource }
  end
  permissions :new?, :create? do
    it { should permit nil, resource }
    it { should_not permit double,
resource }
  end
  permissions :show?, :edit?, :update? do
    it { should_not permit nil, resource }
    it { should_not permit users(:two),
resource }
    it { should permit users(:one),
resource }
  end
end
Файл spec/policies/problem_policy_spec.rb
require 'rails_helper'
RSpec.describe ProblemPolicy do
  subject { described_class }
  permissions :new?, :create? do
    let(:resource) { double }
    it { should_not permit nil, resource }
    context do
      let(:user) { stub_model User, roles:
[] }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:administrator] }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:moderator] }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:confirmed] }
      it { should permit user, resource }
    end
  end
  permissions :edit?, :update?, :destroy?
do
    let(:owner) { stub_model User }
    let(:resource) { double user: owner }
    it { should_not permit nil, resource }
    context do
      let(:user) { stub_model User, roles:
[:confirmed] }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:administrator] }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:moderator] }
      it { should permit user, resource }
    end
    it { should permit owner, resource }
  end
  permissions :index? do
    it { should permit nil, double }
  end
  permissions :show? do
    context do
      let(:resource) { stub_model Problem,
private: false }
      it { should permit nil, resource }
      it { should permit stub_model(User),
resource }
    end
    context do
      let(:owner) { stub_model User }
      let(:resource) { stub_model Problem,
private: true, user: owner }
      it { should permit owner, resource }
      context do
        let(:user) { stub_model User,
roles: [:moderator] }
        it { should permit user, resource
}
      end
      context do
        let(:user) { stub_model User }
        before { expect(user).to
receive(:shared_problems).and_return([reso
urce]) }
        it { should permit user, resource
}
      end
      context do
        let(:user) { stub_model User }
        before { expect(user).to
receive(:shared_problems).and_return([]) }
        it { should_not permit user,
resource }
      end
      context do
        let(:user) { stub_model User,
roles: [:administrator] }
        before { expect(user).to
receive(:shared_problems).and_return([]) }
        it { should_not permit user,
resource }
      end
      context do
        let(:user) { stub_model User,
roles: [:confirmed] }
        before { expect(user).to
receive(:shared_problems).and_return([]) }
        it { should_not permit user,
resource }
      end
    end
  end
end
Файл spec/policies/archive_policy_spec.rb
require 'rails_helper'
RSpec.describe ArchivePolicy do
  subject { described_class }
  let(:resource) { double }
  permissions :new?, :create? do
    context do
      let(:user) { stub_model User }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:moderator] }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:confirmed] }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:administrator] }
      it { should permit user, resource }
    end
  end
end
Файл spec/policies/group_policy_spec.rb
require 'rails_helper'
RSpec.describe GroupPolicy do
  subject { described_class }
  let(:resource) { double }
  let(:user) { stub_model User }
  permissions :index? do
    it { should_not permit nil, resource }
    it { should permit user, resource }
  end
  permissions :new?, :create? do
    it { should_not permit nil, resource }
    it { should_not permit user, resource
}
    context do
      let(:user) { stub_model User, roles:
[:confirmed] }
      it { should permit user, resource }
    end
  end
  permissions :edit?, :update?, :destroy?
do
    context do
      let(:user) { nil }
      it { should_not permit user,
resource }
    end
    context do
      let(:resource) { stub_model Group,
owner: stub_model(User) }
      it { should_not permit user,
resource }
    end
    context do
      let(:resource) { stub_model Group,
owner: user }
      it { should permit user, resource }
    end
  end
  permissions :show? do
    it { should_not permit nil, resource }
    context do
      let(:resource) { stub_model Group,
visibility: :private, owner:
stub_model(User) }
      before { expect(resource).to
receive(:accepted_users).and_return([stub_
model(User)]) }
      it { should_not permit user,
resource }
    end
    context do
      let(:resource) { stub_model Group,
visibility: :private, owner:
stub_model(User) }
      before { expect(resource).to
receive(:accepted_users).and_return([stub_
model(User), user]) }
      it { should permit user, resource }
    end
    context do
      let(:resource) { stub_model Group,
visibility: :private, owner: user }
      it { should permit user, resource }
    end
    context do
      let(:resource) { stub_model Group,
visibility: :moderated }
      it { should permit user, resource }
    end
    context do
      let(:resource) { stub_model Group,
visibility: :public }
      it { should permit user, resource }
    end
  end
  permissions :new_sharing?,
:index_memberships? do
    fixtures :users, :groups
    it { should_not permit nil, resource }
    it { should permit users(:one),
groups(:one) }
    it { should_not permit users(:two),
groups(:one) }
    it { should_not permit users(:three),
groups(:one) }
  end
  permissions :new_invite? do
    fixtures :users, :groups, :memberships
    it { should_not permit nil, resource }
    it { should permit users(:one),
groups(:one) }
    context do
      let(:resource) { groups :one }
      before { resource.visibility =
:public }
      it { should permit users(:two),
groups(:one) }
      it { should_not permit
users(:three), groups(:one) }
    end
    context do
      let(:resource) { groups :one }
      before { resource.visibility =
:private }
      it { should_not permit users(:two),
groups(:one) }
      it { should_not permit
users(:three), groups(:one) }
    end
    context do
      let(:resource) { groups :one }
      before { resource.visibility =
:moderated }
      it { should_not permit users(:two),
groups(:one) }
      it { should_not permit
users(:three), groups(:one) }
    end
  end
end
Файл spec/policies/sharing_policy_spec.rb
require 'rails_helper'
RSpec.describe SharingPolicy do
  subject { described_class }
  fixtures :users
  permissions :new? do
    let(:resource) { stub_model Sharing,
group: stub_model(Group, owner:
users(:one)) }
    it { should permit users(:one),
resource }
    it { should_not permit users(:two),
resource }
    it { should_not permit users(:three),
resource }
    it { should_not permit nil, resource }
  end
  permissions :create? do
    it { should_not permit nil, double }
    let(:resource) { stub_model Sharing,
group: group, problem: problem }
    context do
```

```
      let(:group) { stub_model Group,
owner: users(:one) }
      let(:problem) { stub_model Problem,
user: users(:three) }
      it { should_not permit users(:one),
resource }
      it { should_not permit users(:two),
resource }
      it { should_not permit
users(:three), resource }
    end
    context do
      let(:group) { stub_model Group,
owner: users(:one) }
      let(:problem) { stub_model Problem,
user: users(:one) }
      it { should permit users(:one),
resource }
      it { should_not permit users(:two),
resource }
      it { should_not permit
users(:three), resource }
    end
  end
end
```

Файл spec/policies/submission/retest_policy_spec.rb

```
require 'rails_helper'
RSpec.describe Submission::RetestPolicy do
  subject { described_class }
  let(:resource) { double }
  permissions :new?, :create? do
    context do
      let(:user) { stub_model User }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:moderator] }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:confirmed] }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:administrator] }
      it { should permit user, resource }
    end
  end
end
```

Файл spec/policies/confirmation_request/accept_policy_spec.rb

```
require 'rails_helper'
RSpec.describe
ConfirmationRequest::AcceptPolicy do
  subject { described_class }
  let(:sender) { stub_model User }
  let(:status) { :pending }
  let(:confirmation_request) { stub_model
ConfirmationRequest, status: status, user:
sender }
  let(:resource) {
ConfirmationRequest::Accept.new
confirmation_request }
  permissions :new?, :create? do
    it { should_not permit nil, resource }
    it { should_not permit sender,
resource }
    it { should_not permit
stub_model(User), resource }
    context do
      let(:user) { stub_model User, roles:
[:administrator] }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:confirmed] }
      it { should_not permit user,
resource }
    end
    context do
      let(:status) { :accepted }
      let(:user) { stub_model User, roles:
[:moderator] }
      it { should_not permit user,
resource }
    end
    context do
      let(:status) { :rejected }
      let(:user) { stub_model User, roles:
[:moderator] }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:moderator] }
      it { should permit user, resource }
    end
  end
end
```

Файл spec/policies/confirmation_request/reject_policy_spec.rb

```
require 'rails_helper'
RSpec.describe
ConfirmationRequest::RejectPolicy do
  subject { described_class }
  let(:sender) { stub_model User }
  let(:status) { :pending }
  let(:confirmation_request) { stub_model
ConfirmationRequest, status: status, user:
sender }
  let(:resource) {
ConfirmationRequest::Accept.new
confirmation_request }
  permissions :new?, :create? do
    it { should_not permit nil, resource }
    it { should_not permit sender,
resource }
    it { should_not permit
stub_model(User), resource }
    context do
      let(:user) { stub_model User, roles:
[:administrator] }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:confirmed] }
      it { should_not permit user,
resource }
    end
    context do
      let(:status) { :accepted }
      let(:user) { stub_model User, roles:
[:moderator] }
      it { should_not permit user,
resource }
    end
    context do
      let(:status) { :rejected }
      let(:user) { stub_model User, roles:
[:moderator] }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:moderator] }
      it { should permit user, resource }
    end
  end
end
```

Файл spec/policies/password_recovery_policy_spec.rb

```
require 'rails_helper'
RSpec.describe PasswordRecoveryPolicy do
  subject { described_class }
  fixtures :users
  permissions :new?, :create? do
    it { should permit nil, double }
    it { should_not permit users(:one),
double }
  end
end
```

Файл spec/policies/password_policy_spec.rb

```
require 'rails_helper'
RSpec.describe PasswordPolicy do
  subject { described_class }
  fixtures :users
  permissions :edit?, :update? do
    it { should permit nil, double }
    it { should_not permit nil, nil }
    it { should_not permit users(:one),
nil }
    it { should_not permit users(:one),
double }
  end
end
```

Файл spec/policies/session_policy_spec.rb

```
require 'rails_helper'
RSpec.describe SessionPolicy do
  subject { described_class }
  fixtures :users
  permissions :new?, :create? do
    it { should permit nil, double }
    it { should_not permit users(:one),
double }
  end
  permissions :destroy? do
    it { should permit users(:one), double
}
    it { should_not permit nil, double }
  end
end
```

Файл spec/policies/log_policy_spec.rb

```
require 'rails_helper'
RSpec.describe LogPolicy do
  subject { described_class }
  fixtures :users
  permissions :create? do
    let(:resource) { stub_model Log,
submission: stub_model(Submission,
test_state: test_state) }
    context do
      let(:test_state) { :pending }
      it { should_not permit nil, resource
}
    end
    context do
      let(:test_state) { :in_progress }
      it { should permit nil, resource  }
    end
    context do
      let(:test_state) { :done }
      it { should_not permit nil, resource
}
    end
    context do
      let(:test_state) { :failed }
      it { should_not permit nil, resource
}
    end
  end
  permissions :show? do
    it { should_not permit nil, double }
    it { should permit users(:two), double
}
    let(:resource) { stub_model Log, type:
:source, submission: submission }
    context do
      let(:submission) { stub_model
Submission, user: users(:one) }
      it { should permit users(:one),
resource }
    end
    context do
      let(:submission) { stub_model
Submission, user: users(:two), problem:
stub_model(Problem, user: users(:one)) }
      it { should permit users(:one),
resource }
    end
    context do
      let(:submission) { stub_model
Submission, user: users(:two), problem:
stub_model(Problem, user: users(:three)) }
      it { should_not permit users(:one),
resource }
    end
  end
end
```

Файл spec/policies/tag_policy_spec.rb

```
require 'rails_helper'
RSpec.describe TagPolicy do
  subject { described_class }
  permissions :index? do
    it { should permit nil, double }
  end
end
```

Файл spec/policies/compiler_policy_spec.rb

```
require 'rails_helper'
RSpec.describe CompilerPolicy do
  subject { described_class }
  fixtures :users
  permissions :index?, :new?, :edit?,
:create?, :update? do
    it { should_not permit nil, double }
    it { should_not permit users(:one),
double }
    it { should permit users(:two), double
}
  end
  permissions :destroy? do
    context do
      let(:resource) { double persisted?:
true }
      it { should_not permit nil, resource
}
      it { should_not permit users(:one),
resource }
      it { should permit users(:two),
resource }
    end
    context do
      let(:resource) { double persisted?:
false }
      it { should_not permit nil, resource
}
      it { should_not permit users(:one),
resource }
      it { should_not permit users(:two),
resource }
    end
  end
end
```

Файл spec/policies/result_policy_spec.rb

```
require 'rails_helper'
RSpec.describe ResultPolicy do
  subject { described_class }
  permissions :create? do
    let(:resource) { stub_model Result,
submission: stub_model(Submission,
test_state: test_state) }
    context do
      let(:test_state) { :pending }
      it { should_not permit nil, resource
}
    end
    context do
      let(:test_state) { :in_progress }
      it { should permit nil, resource  }
    end
    context do
      let(:test_state) { :done }
      it { should_not permit nil, resource
}
    end
    context do
      let(:test_state) { :failed }
```

```
      it { should_not permit nil, resource
}
    end
  end
end
Файл
spec/policies/confirmation_request_policy_
spec.rb
require 'rails_helper'
RSpec.describe ConfirmationRequestPolicy
do
  subject { described_class }
  fixtures :users, :confirmation_requests
  permissions :new?, :create? do
    let(:resource) {
ConfirmationRequest.new user: user }
    context do
      let(:user) { nil }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { users(:one) }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { users(:two) }
      it { should_not permit user,
resource }
    end
    context do
      let(:user) { users(:three) }
      it { should permit user, resource }
    end
  end
  permissions :index? do
    it { should_not permit nil, double }
    it { should_not permit users(:one),
double }
    it { should_not permit users(:three),
double }
    it { should permit users(:two), double
}
  end
end
Файл spec/policies/worker_policy_spec.rb
require 'rails_helper'
RSpec.describe WorkerPolicy do
  subject { described_class }
  fixtures :users
  permissions :create?, :update? do
    it { should permit nil, double }
  end
  permissions :index?, :destroy? do
    it { should_not permit users(:one),
double }
    it { should permit users(:two), double
}
  end
end
Файл
spec/policies/submission_policy_spec.rb
require 'rails_helper'
RSpec.describe SubmissionPolicy do
  subject { described_class }
  permissions :index? do
    it { should permit nil, double }
  end
  permissions :show? do
    it { should_not permit nil, double }
    context do
      let(:resource) { stub_model
Submission, problem: stub_model(Problem) }
      it { should_not permit
stub_model(User), resource }
    end
    context do
      let(:user) { stub_model User, roles:
[:administrator] }
      let(:resource) { stub_model
Submission }
      it { should permit user, resource }
    end
    context do
      let(:user) { stub_model User }
      let(:resource) { stub_model
Submission, user: user }
      it { should permit user, resource }
    end
    context do
      let(:user) { stub_model User }
      let(:resource) { stub_model
Submission, problem: stub_model(Problem,
user: user) }
      it { should permit user, resource }
    end
  end
  permissions :create? do
    it { should_not permit nil, double }
    let(:resource) { stub_model
Submission, problem: problem }
    context do
      let(:problem) { stub_model Problem,
private: false }
      it { should permit double, resource
}
    end
    context do
      let(:problem) { stub_model Problem,
private: true }
      let(:user) { stub_model User, roles:
[:moderator] }
      it { should permit user, resource }
    end
    context do
      let(:user) { stub_model User }
      let(:problem) { stub_model Problem,
private: true, user: user }
      it { should permit user, resource }
    end
    context do
      let(:problem) { stub_model Problem,
private: true }
      let(:user) { stub_model User }
      before { expect(user).to
receive(:shared_problems).and_return([prob
lem]) }
      it { should permit user, resource }
    end
    context do
      let(:problem) { stub_model Problem,
private: true }
      let(:user) { stub_model User }
      before { expect(user).to
receive(:shared_problems).and_return([]) }
      it { should_not permit user,
resource }
    end
  end
  permissions :destroy? do
    context do
      let(:user) { stub_model User }
      it { should_not permit user, double
}
    end
    context do
      let(:user) { stub_model User, roles:
[:moderator] }
      it { should_not permit user, double
}
    end
    context do
      let(:user) { stub_model User, roles:
[:confirmed] }
      it { should_not permit user, double
}
    end
    context do
      let(:user) { stub_model User, roles:
[:administrator] }
      it { should permit user, double }
    end
  end
end
Файл
spec/policies/application_policy_spec.rb
require 'rails_helper'
RSpec.describe ApplicationPolicy do
  subject { described_class }
  permissions :index?, :show?, :new?,
:create?, :edit?, :update?, :destroy? do
    it { should_not permit double, double
}
  end
end
Файл
spec/decorators/test_decorator_spec.rb
require 'rails_helper'
RSpec.describe TestDecorator do
  let(:resource) { stub_model Test, id:
21, num: '10 A' }
  subject { resource.decorate }
  describe '#as_json' do
    before { expect(subject).to
receive(:input_url).and_return(:input_url)
}
    before { expect(subject).to
receive(:answer_url).and_return(:answer_ur
l) }
    its(:as_json) { should eq id: 21, num:
'10 A', input_url: :input_url, answer_url:
:answer_url }
  end
  describe '#input_url' do
    before { allow(subject).to
receive(:input).and_return(input) }
    context do
      let(:input) { double attached?:
false }
      its(:input_url) { should be_nil }
    end
    context do
      let(:input) { double attached?: true
}
      before do
        expect(subject).to
receive_message_chain(:helpers,
:url_for).with(input).

and_return('https://test.host/input')
      end
      its(:input_url) { should eq
'https://test.host/input' }
    end
  end
  describe '#answer_url' do
    before { allow(subject).to
receive(:answer).and_return(answer) }
    context do
      let(:answer) { double attached?:
false }
      its(:answer_url) { should be_nil }
    end
    context do
      let(:answer) { double attached?:
true }
      before do
        expect(subject).to
receive_message_chain(:helpers,
:url_for).with(answer).

and_return('https://test.host/answer')
      end
      its(:answer_url) { should eq
'https://test.host/answer' }
    end
  end
end
Файл
spec/decorators/problem_decorator_spec.rb
require 'rails_helper'
RSpec.describe ProblemDecorator do
  let(:translation) { stub_model
ProblemTranslation, language: 'ru' }
  let(:default_translation) { stub_model
ProblemTranslation, language: 'uk',
default: true }
  let(:resource) { stub_model Problem, id:
24, updated_at: Time.zone.now,
checker_compiler_id: 1 }
  before { allow(resource).to
receive(:translation).and_return(translati
on) }
  before { allow(resource).to
receive(:default_translation).and_return(d
efault_translation) }
  subject { resource.decorate }
  it { should
delegate_method(:caption).to(:translation)
}
  it { should
delegate_method(:author).to(:translation)
}
  it { should
delegate_method(:text).to(:translation) }
  it { should
delegate_method(:technical_text).to(:trans
lation) }
  describe '#as_json' do
    context do
      before { expect(subject).to
receive(:checker_source_url).and_return(:c
hecker_source_url) }
      before { expect(subject).to
receive(:tests).and_return(:tests) }
      its :as_json do
        should eq id: 24, updated_at:
Time.zone.now, checker_compiler_id: 1,
          checker_source_url:
:checker_source_url, tests: :tests
      end
    end
    context do
      subject { resource.decorate context:
:submission }
      its(:as_json) { should eq id: 24,
updated_at: Time.zone.now,
checker_compiler_id: 1 }
    end
  end
  describe '#checker_source_url' do
    before { allow(subject).to
receive(:checker_source).and_return(checke
r_source) }
    context do
      let(:checker_source) { double
attached?: false }
      its(:checker_source_url) { should
be_nil }
    end
    context do
      let(:checker_source) { double
attached?: true }
      before do
        expect(subject).to
receive_message_chain(:helpers,
:url_for).with(checker_source).

and_return('https://test.host/checker_sour
ce')
      end
      its(:checker_source_url) { should eq
'https://test.host/checker_source' }
    end
  end
  describe '#translation' do
    its(:translation) { should eq
translation }
    context do
      let(:translation) { nil }
      its(:translation) { should eq
default_translation }
    end
  end
  describe '#language' do
    its(:language) { should eq :ru }
    context do
      let(:translation) { nil }
      let(:default_translation) { nil }
      its(:language) { should be_nil }
    end
  end
```

```
end
Файл spec/decorators/tag_decorator_spec.rb
require 'rails_helper'
RSpec.describe TagDecorator do
  let(:resource) { stub_model Tag, id: 5,
name: 'Simple Problem' }
  subject { resource.decorate }
  its(:as_json) { should eq  value: 5,
text: 'Simple Problem' }
end
Файл
spec/decorators/user_decorator_spec.rb
require 'rails_helper'
RSpec.describe UserDecorator do
  fixtures :users
  let(:resource) { users(:one) }
  subject { resource.decorate }
  its(:search_suggestion) { should eq
'<div><p class="mb-0">User One</p><small
class="text-muted">user-one</small></div>'
}
  describe '#as_json' do
    before { expect(subject).to
receive(:search_suggestion).and_return(:se
arch_suggestion) }
    its(:as_json) { should eq id:
resource.id, name: 'User One',
search_suggestion: :search_suggestion }
  end
  describe '#name' do
    its(:name) { should eq 'User One' }
    context do
      let(:resource) { stub_model User,
username: 'kostyanf14' }
      its(:name) { should eq 'kostyanf14'
}
    end
  end
end
Файл
spec/decorators/avatar_decorator_spec.rb
require 'rails_helper'
RSpec.describe AvatarDecorator do
  let(:resource) { Avatar.new }
  subject { resource.decorate }
  describe '#as_json' do
    before { expect(subject).to
receive(:url).and_return(:url) }
    its(:as_json) { should eq url: :url }
  end
  describe '#url' do
    before { allow(subject).to
receive_message_chain(:user,
:avatar).and_return(avatar) }
    context do
      let(:avatar) { double attached?:
false }
      its(:url) { should be_nil }
    end
    context do
      let(:avatar) { double attached?:
true }
      let(:options) { { resize:
'300x400^', crop: '300x400+0+0', gravity:
'center' } }
      before { expect(avatar).to
receive(:variant).with(combine_options:
options).and_return(:variant) }
      before { expect(subject).to
receive_message_chain(:helpers,
:url_for).with(:variant).and_return(:url)
}
      its(:url) { should eq :url }
    end
  end
end
Файл
spec/decorators/membership_decorator_spec.
rb
require 'rails_helper'
RSpec.describe MembershipDecorator do
  let(:resource) { stub_model Membership,
user: stub_model(User), group:
stub_model(Group) }
  subject { resource.decorate }
  its(:user) { should be_a UserDecorator }
  its(:group) { should be_a GroupDecorator
}
  it { should
delegate_method(:name).to(:user).with_pref
ix.allow_nil }
  it { should
delegate_method(:name).to(:group).with_pre
fix.allow_nil }
end
Файл
spec/decorators/group_decorator_spec.rb
require 'rails_helper'
RSpec.describe GroupDecorator do
  let(:owner) { stub_model User }
  let(:resource) { stub_model Group,
visibility: :moderated, owner: owner }
  subject { resource.decorate }
  its(:owner) { should be_a UserDecorator
}
  its(:accepted_users) { should be_a
Draper::CollectionDecorator }
  its(:problems) { should be_a
Draper::CollectionDecorator }
  it { should
delegate_method(:name).to(:owner).with_pre
fix }
  it { should
delegate_method(:state_requested?).to(:cur
rent_user_membership).with_prefix.allow_ni
l }
  it { should
delegate_method(:state_invited?).to(:curre
nt_user_membership).with_prefix.allow_nil
}
  it { should
delegate_method(:state_accepted?).to(:curr
ent_user_membership).with_prefix.allow_nil
}
  describe '#visibility_icon' do
    before { expect(subject).to
receive(:visibility_icon_class).and_return
('visibility_icon_class') }
    its :visibility_icon do
      should eq '<i
class="visibility_icon_class"
title="Moderated" data-toggle="tooltip"
data-placement="bottom"></i>'
    end
  end
  describe '#visibility_icon_class' do
    its(:visibility_icon_class) { should
eq 'mr-3 fas fa-lock' }
    context do
      let(:resource) { stub_model Group,
visibility: :private }
      its(:visibility_icon_class) { should
eq 'mr-3 fas fa-lock' }
    end
    context do
      let(:resource) { stub_model Group,
visibility: :public }
      its(:visibility_icon_class) { should
eq 'mr-3 fas fa-unlock' }
    end
  end
  describe '#current_user_membership' do
    before { allow(subject).to
receive_message_chain(:h,
:current_user).and_return(current_user) }
    context do
      let(:current_user) { nil }
      before { expect(resource).to_not
receive(:memberships) }
      its(:current_user_membership) {
should be_nil }
    end
    context do
      let(:current_user) { stub_model User
}
      before do
        expect(resource).to \

receive_message_chain(:memberships,
:find_by).with(user:
current_user).and_return(:current_user_mem
bership)
      end
      its(:current_user_membership) {
should eq :current_user_membership }
    end
    context do
      let(:current_user) { stub_model User
}
      before {
subject.instance_variable_set
:@current_user_membership,
:current_user_membership }
      before { expect(resource).to_not
receive(:memberships) }
      its(:current_user_membership) {
should eq :current_user_membership }
    end
  end
end
Файл
spec/decorators/compiler_decorator_spec.rb
require 'rails_helper'
RSpec.describe CompilerDecorator do
  let :resource do
    stub_model Compiler, id: 128, name:
'Visual C++', version: '14.0.12',
memory_a: 1, memory_b: 0, time_a: 1,
time_b: 0
  end
  subject { resource.decorate }
  its :as_json  do
    should eq  id: 128, name: 'Visual
C++', version: '14.0.12', memory_a: 1,
memory_b: 0, time_a: 1, time_b: 0
  end
end
Файл
spec/decorators/submission_decorator_spec.
rb
require 'rails_helper'
RSpec.describe SubmissionDecorator do
  let(:problem) { stub_model Problem, id:
24, checker_compiler_id: 1, memory_limit:
256, time_limit: 100 }
  let(:compiler) { stub_model Compiler,
memory_a: 12.28, memory_b: 5.112, time_a:
143, time_b: 3.14 }
  let(:resource) { stub_model Submission,
id: 125, compiler_id: 3, test_state: 0,
fails_count: 2 }
  before { allow(resource).to
receive(:problem).and_return(problem) }
  before { allow(resource).to
receive(:compiler).and_return(compiler) }
  subject { resource.decorate }
  it { should
delegate_method(:username).to(:user).with_
prefix }
  it { should
delegate_method(:caption).to(:problem).wit
h_prefix }
  its(:memory_limit) { should eq 3148.792
}
  its(:time_limit) { should eq 14303.14 }
  describe '#as_json' do
    before { expect(subject).to
receive(:source_url).and_return(:source_ur
l) }
    its :as_json do
      should eq id: 125, compiler_id: 3,
problem: problem, source_url: :source_url,
test_state: 0,
       fails_count: 2, memory_limit:
3148.792, time_limit: 14303.14
    end
  end
  describe '#source_url' do
    before { allow(subject).to
receive(:source).and_return(source) }
    context do
      let(:source) { double attached?:
false }
      its(:source_url) { should be_nil }
    end
    context do
      let(:source) { double attached?:
true }
      before do
        expect(subject).to
receive_message_chain(:helpers,
:url_for).with(source).

and_return('https://test.host/source')
      end
      its(:source_url) { should eq
'https://test.host/source' }
    end
  end
  describe '#state' do
    let(:resource) { stub_model
Submission, test_state: test_state,
test_result: test_result, score: 60,
max_score: 100 }
    context do
      let(:test_state) { :done }
      let(:test_result) { :ok }
      its(:state) { should eq 'Ok
(60.0/100.0)' }
    end
    context do
      let(:test_state) { :done }
      let(:test_result) { :compiler_error
}
      its(:state) { should eq 'Compiler
error' }
    end
    context do
      let(:test_state) { :in_progress }
      let(:test_result) { nil }
      its(:state) { should eq 'In
progress' }
    end
  end
end
Файл
spec/factories/application_factory_spec.rb
require 'rails_helper'
RSpec.describe ApplicationFactory do
  let(:described_class) do
    Class.new ApplicationFactory do
      def initialize *args; end
    end
  end
  describe '.build' do
    after { described_class.build :params
}
    it do
      expect(described_class).to
receive(:new).with(:params) do
        double.tap { |a| expect(a).to
receive(:build) }
      end
    end
  end
end
Файл
spec/factories/membership_factory_spec.rb
require 'rails_helper'
RSpec.describe MembershipFactory do
  subject { described_class.new
:current_user, params }
  let(:params) { { user_id: :user_id,
group: :group, type: :type } }
  it { should be_an ApplicationFactory }
  describe '#build' do
    before { expect(subject).to
receive(:user).and_return(:user) }
```

```
    before { expect(subject).to
receive(:state).and_return(:state) }
    before do
      expect(Membership).to
receive(:new).with(user: :user, group:
:group, state:
:state).and_return(:membership)
    end
    its(:build) { should eq :membership }
  end
  describe '#user' do
    context do
      let(:params) { { user_id: 6, type:
'invite' } }
      before { expect(User).to
receive(:find_by).with(id:
6).and_return(:user) }
      its(:user) { should eq :user }
    end
    context do
      let(:params) { { type: 'request' } }
      before { expect(User).to_not
receive(:find_by) }
      its(:user) { should eq :current_user
}
    end
    context do
      let(:params) { { type: 'unknown' } }
      before { expect(User).to_not
receive(:find_by) }
      its(:user) { should be_nil }
    end
  end
  describe '#state' do
    context do
      let(:params) { { type: 'invite',
group: :group } }
      its(:state) { should eq :invited }
    end
    context do
      let(:params) { { type: 'request',
group: group } }
      context do
        let(:group) { stub_model Group,
visibility: :public }
        its(:state) { should eq :accepted
}
      end
      context do
        let(:group) { stub_model Group,
visibility: :moderated }
        its(:state) { should eq :requested
}
      end
      context do
        let(:group) { stub_model Group,
visibility: :private }
        its(:state) { should be_nil }
      end
    end
    context do
      let(:params) { { type: 'unknown',
group: :group } }
      its(:state) { should be_nil }
    end
    context do
      let(:params) { { type: 'request' } }
      its(:state) { should be_nil }
    end
  end
end
Файл spec/fixtures/groups.yml
one:
  name: Group One
  owner: one
Файл spec/fixtures/memberships.yml
one:
  user: one
  group: one
  state: 2
two:
  user: two
  group: one
  state: 2
Файл spec/fixtures/users.yml
one:
  username: user-one
  name: User One
  email: one@users.com
  password_digest: XXXYYYZZZ
two:
  username: user-two
  name: User Two
  email: two@users.com
  password_digest: XXXYYYZZZ
  roles: 4
three:
  username: user-three
  name: User Three
  email: three@users.com
  password_digest: XXXYYYZZZ
Файл spec/fixtures/workers.yml
ok:
  name: Worker Ok
  api_version: 1
  api_type: HTTP
  webhook_supported: false
  status: ok
  alive_at: '20.09.2018 19:38:42'
  ips:
    - '8.8.8.8'
    - '1.1.1.1'
  task_status:
    - Working
    - Working
disabled:
  name: Worker Disabled
  api_version: 2
  api_type: HTTP
  webhook_supported: false
  status: disabled
  alive_at: '20.09.2018 19:38:42'
  ips:
    - '8.8.8.8'
    - '1.1.1.1'
  task_status:
    - Stopped
    - Stopped
failed:
  name: Worker Failed
  api_version: 3
  api_type: HTTP
  webhook_supported: false
  status: failed
  alive_at: '20.09.2018 19:38:42'
  ips:
    - '8.8.8.8'
    - '1.1.1.1'
  task_status:
    - Failed
    - Failed
stale:
  name: Worker Stale
  api_version: 4
  api_type: HTTP
  webhook_supported: false
  status: stale
  alive_at: '20.09.2017 19:38:42'
  ips:
    - '8.8.8.8'
    - '1.1.1.1'
  task_status:
    - Working
    - Working
stopped:
  name: Worker Stopped
  api_version: 2
  api_type: HTTP
  webhook_supported: false
  status: stopped
  alive_at: '20.09.2018 19:38:42'
  ips:
    - '8.8.8.8'
    - '1.1.1.1'
  task_status:
    - Working
    - Stopped
Файл
spec/fixtures/confirmation_requests.yml
one:
  user: one
  status: pending
Файл spec/searchers/user_searcher_spec.rb
require 'rails_helper'
RSpec.describe UserSearcher do
  let(:relation) { double }
  subject { UserSearcher.search relation,
params }
  describe '#search_by_query' do
    context do
      let(:params) { acp query: 'John' }
      let(:sql) { 'username ILIKE :query
OR name ILIKE :query' }
      before { expect(relation).to
receive(:where).with(sql, query:
'%John%').and_return(:result) }
      it { should eq :result }
    end
    context do
      let(:params) { acp query: '' }
      before { expect(relation).to_not
receive(:where) }
      it { should eq relation }
    end
  end
end
Файл
spec/searchers/application_searcher_spec.r
b
require 'rails_helper'
RSpec.describe ApplicationSearcher do
  let(:relation) { double }
  let(:params) { acp name: 'John', age:
'19' }
  subject { described_class.new relation,
params }
  its(:relation) { should eq relation }
  its(:params) { should eq params }
  describe '#search' do
    context do
      let(:params) { nil }
      its(:search) { should eq relation }
    end
    context do
      let(:described_class) { Class.new
ApplicationSearcher }
      its(:search) { should eq relation }
    end
    context do
      let :described_class do
        Class.new ApplicationSearcher do
          def search_by_name name; end
          def search_by_age age; end
        end
      end
      let(:conditions) { [double, double]
}
      before { expect(subject).to
receive(:search_by_name).with('John').and_
return(conditions[0]) }
      before { expect(subject).to
receive(:search_by_age).with('19').and_ret
urn(conditions[1]) }
      before { expect(conditions[0]).to
receive(:merge).with(conditions[1]).and_re
turn(:result) }
      its(:search) { should eq :result }
    end
  end
  describe '.search' do
    it do
      expect(described_class).to
receive(:new).with(:params) do
        double.tap { |a| expect(a).to
receive(:search) }
      end
    end
    after { described_class.search :params
}
  end
end
Файл
spec/mailers/previews/user_mailer_preview.
rb
class UserMailerPreview <
ActionMailer::Preview
  def email
    UserMailer.email User.new username:
'test-username'
  end
end
Файл
spec/mailers/previews/password_recovery_ma
iler_preview.rb
class PasswordRecoveryMailerPreview <
ActionMailer::Preview
  def email
    PasswordRecoveryMailer.email User.new
username: 'test-username',
password_recovery_token: SecureRandom.uuid
  end
end
Файл
spec/mailers/password_recovery_mailer_spec
.rb
require "rails_helper"
RSpec.describe PasswordRecoveryMailer,
type: :mailer do
  let(:user) { stub_model User, username:
'test', email: 'user@codelabs.site',
password_recovery_token: SecureRandom.uuid
}
  subject { described_class.email user }
  its(:subject) { should eq 'Password
recovery at CodeLabs' }
  its(:from) { should eq
[ENV['SMTP_FROM']] }
  its(:to) { should eq
['user@codelabs.site'] }
end
Файл spec/mailers/user_mailer_spec.rb
require "rails_helper"
RSpec.describe UserMailer, type: :mailer
do
  let(:user) { stub_model User, username:
'test', email: 'user@codelabs.site' }
  subject { described_class.email user }
  its(:subject) { should eq 'Successful
registration at CodeLabs' }
  its(:from) { should eq
[ENV['SMTP_FROM']] }
  its(:to) { should eq
['user@codelabs.site'] }
end
Файл spec/rails_helper.rb
ENV["RAILS_ENV"] ||= 'test'
require
File.expand_path('../../config/environment
', __FILE__)
require 'rspec/rails'
require 'pundit/rspec'
require 'aasm/rspec'
Dir[Rails.root.join('spec/support/**/*.rb'
)].each { |f| require f }
RSpec.configure do |config|
  config.mock_with :rspec
  config.order = :random
  config.use_transactional_fixtures = true
  config.fixture_path = Rails.root.join
'spec/fixtures'
  config.expect_with :rspec do |c|
    c.syntax = :expect
  end
  config.include
ActionControllerParameters
  config.include
ActiveSupport::Testing::TimeHelpers
  [:controller, :view, :request].each do
|type|
    config.include
Rails::Controller::Testing::TestProcess,
type: type
```

```
    config.include
Rails::Controller::Testing::TemplateAssert
ions, type: type
    config.include
Rails::Controller::Testing::Integration,
type: type
  end
  config.before { freeze_time }
end
Shoulda::Matchers.configure do |config|
  config.integrate do |with|
    with.test_framework :rspec
    with.library :rails
  end
end
ActiveRecord::Migration.maintain_test_sche
ma!
I18n.default_locale = :en
```

Файл spec/validators/byte_size_validator_spec.rb

```
require 'rails_helper'
ByteSizeValidator::Validatable =
Struct.new :source do
  include ActiveModel::Validations
  validates :source, byte_size: { maximum:
1.kilobytes }
end
RSpec.describe ByteSizeValidator do
  subject {
ByteSizeValidator::Validatable.new source
}
  before { subject.valid? }
  context do
    let(:source) { double attached?: false
}
    its(:errors) { should be_empty }
  end
  context do
    let(:source) { double attached?: true,
byte_size: 1.kilobytes }
    its(:errors) { should be_empty }
  end
  context do
    let(:source) { double attached?: true,
byte_size: 2.kilobytes }
    its(:errors) { should not be_empty }
    its('errors.details') { should eq
source: [{ count: 1024, error: :too_long
}] }
  end
end
```

Файл spec/controllers/avatars_controller_spec.rb

```
require 'rails_helper'
RSpec.describe AvatarsController, type:
:controller do
  it_behaves_like :create, format: :json
do
    let(:resource) { double }
    let(:success) { -> { should
render_template(:create).with_status(200)
} }
    let(:failure) { -> { should
render_template(:errors).with_status(422)
} }
  end
  it_behaves_like :destroy do
    let(:success) { -> { should
respond_with 204 } }
  end
  describe '#resource' do
    context do
      before {
subject.instance_variable_set :@resource,
:resource }
      its(:resource) { should eq :resource
}
    end
    context do
      before { expect(subject).to
receive(:build_resource).and_return(:resou
rce) }
      its(:resource) { should eq :resource
}
    end
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:resource_params).and_return(:reso
urce_params) }
    before { expect(Avatar).to
receive(:new).with(:resource_params).and_r
eturn(:resource) }
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
  describe '#resource_params' do
    let(:params) { acp avatar: { file: ''
} }
    before { expect(subject).to
receive(:params).and_return(params) }
    before { expect(subject).to
receive(:current_user).and_return(:current
_user) }
    its(:resource_params) { should eq
params[:avatar].permit!.merge(user:
:current_user) }
  end
end
```

Файл spec/controllers/workers_controller_spec.rb

```
require 'rails_helper'
RSpec.describe WorkersController, type:
:controller do
  it_behaves_like :index
  it_behaves_like :destroy do
    let(:success) { -> { should
redirect_to :workers } }
  end
  describe '#resource' do
    context do
      before {
subject.instance_variable_set :@resource,
:resource }
      its(:resource) { should eq :resource
}
    end
    context do
      before { expect(subject).to
receive(:params).and_return(id: 324) }
      before { expect(Worker).to
receive(:find).with(324).and_return(:resou
rce) }
      its(:resource) { should eq :resource
}
    end
  end
  describe '#collection' do
    context do
      before {
subject.instance_variable_set
:@collection, :collection }
      its(:collection) { should eq
:collection }
    end
    context do
      before { expect(Worker).to
receive(:order).with(alive_at:
:desc).and_return(:collection) }
      its(:collection) { should eq
:collection }
    end
  end
end
```

Файл spec/controllers/confirmation_requests_controller_spec.rb

```
require 'rails_helper'
RSpec.describe
ConfirmationRequestsController, type:
:controller do
  it_behaves_like :create do
    let(:resource) { double }
    let :success do
      lambda do
        should set_flash[:success]
        should redirect_to :profile
      end
    end
    let :failure do
      lambda do
        should set_flash[:error]
        should redirect_to :profile
      end
    end
  end
  describe '#resource' do
    before { subject.instance_variable_set
:@resource, :resource }
    its(:resource) { should eq :resource }
  end
  describe '#collection' do
    context do
      before {
subject.instance_variable_set
:@collection, :collection }
      its(:collection) { should eq
:collection }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(page: 23) }
      before do
        expect(ConfirmationRequest).to
receive(:includes).with(:user) do
          double.tap { |a| expect(a).to
receive(:page).with(23).and_return(:collec
tion) }
        end
      end
      its(:collection) { should eq
:collection }
    end
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:current_user).and_return(:current
_user) }
    before {
expect(ConfirmationRequest).to
receive(:new).with(user:
:current_user).and_return(:resource) }
    before  { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
```

Файл spec/controllers/password_recoveries_controller_spec.rb

```
require 'rails_helper'
RSpec.describe
PasswordRecoveriesController, type:
:controller do
  it_behaves_like :new, anonymous: true
  describe '#create' do
    let(:resource) { double }
    it_behaves_like :create, anonymous:
true do
      let(:success) { -> { should
render_template :create } }
      let(:failure) { -> {  should
render_template :create } }
      before { expect(subject).to
receive(:verify_recaptcha).and_return(true
) }
    end
    context do
      before { allow(subject).to
receive(:resource).and_return(resource) }
      before { expect(subject).to
receive(:authorize).with(resource).and_ret
urn(true) }
      before { expect(subject).to
receive(:build_resource) }
      before { expect(subject).to
receive(:verify_recaptcha).and_return(fals
e) }
      before { expect(resource).to_not
receive(:save) }
      before { post :create, format: :html
}
      it { should render_template :new }
    end
  end
  describe '#resource_params' do
    let(:params) { acp password_recovery:
{ email: 'one@users.com' } }
    before { expect(subject).to
receive(:params).and_return(params) }
    its(:resource_params) { should eq
params[:password_recovery].permit! }
  end
  describe '#resource' do
    before { subject.instance_variable_set
:@resource, :resource }
    its(:resource) { should eq :resource }
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:resource_params).and_return(:reso
urce_params) }
    before { expect(PasswordRecovery).to
receive(:new).with(:resource_params).and_r
eturn(:resource) }
    before { subject.send :build_resource}
    its(:resource) { should eq :resource }
  end
  describe '#initialize_resource' do
    before { expect(PasswordRecovery).to
receive(:new).with(no_args).and_return(:re
source) }
    before { subject.send
:initialize_resource}
    its(:resource) { should eq :resource }
  end
end
```

Файл spec/controllers/groups_controller_spec.rb

```
require 'rails_helper'
RSpec.describe GroupsController, type:
:controller do
  it_behaves_like :index
  it_behaves_like :show
  it_behaves_like :new
  it_behaves_like :edit
  it_behaves_like :create do
    let(:resource) { stub_model Group }
    let(:success) { -> { should
redirect_to resource } }
    let(:failure) { -> { should
render_template :new } }
  end
  it_behaves_like :update do
    let(:resource) { stub_model Group }
    let(:success) { -> { should
redirect_to resource } }
    let(:failure) { -> { should
render_template :edit } }
  end
  it_behaves_like :destroy do
    let(:success) { -> { should
redirect_to :groups } }
  end
  describe '#resource' do
    context do
      before {
subject.instance_variable_set :@resource,
:resource }
      its(:resource) { should eq :resource
}
    end
    context do
      before { expect(subject).to
receive(:params).and_return(id: 18) }
      before do
        expect(Group).to
receive(:find).with(18) do
```

```
          double.tap { |a| expect(a).to
receive(:decorate).and_return(:resource) }
        end
      end
      its(:resource) { should eq :resource
}
    end
  end
  describe '#collection' do
    context do
      before {
subject.instance_variable_set
:@collection, :collection }
      its(:collection) { should eq
:collection }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(page: 22) }
      before do
        expect(Group).to
receive(:includes).with(:owner) do
          double.tap do |a|
            expect(a).to
receive(:order).with(:name) do
              double.tap { |b|
expect(b).to
receive(:page).with(22).and_return(:collec
tion) }
            end
          end
        end
      end
      its(:collection) { should eq
:collection }
    end
  end
  describe '#resource_params' do
    let(:params) { acp_group: { name: '',
visibility: '', description: '' } }
    before { expect(subject).to
receive(:params).and_return(params) }
    its(:resource_params) { should eq
params[:group].permit! }
  end
  describe '#initialize_resource' do
    before { expect(subject).to
receive_message_chain(:current_user,
:owned_groups, :new).and_return(:resource)
}
    before { subject.send
:initialize_resource }
    its(:resource) { should eq :resource }
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:resource_params).and_return(:para
ms) }
    before do
      expect(subject).to
receive_message_chain(:current_user,
:owned_groups,
:new).with(:params).and_return(:resource)
    end
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
Файл
spec/controllers/submissions_controller_sp
ec.rb
require 'rails_helper'
RSpec.describe SubmissionsController,
type: :controller do
  it_behaves_like :show
  it_behaves_like :index, anonymous: true
  it_behaves_like :index, anonymous: true,
params: { problem_id: 7 }
  it_behaves_like :index, anonymous: true,
params: { group_id: 7 }
  it_behaves_like :index, anonymous: true,
params: { user_id: 7 }
  it_behaves_like :create, params: {
problem_id: 7 } do
    let(:resource) { stub_model Submission
}
    let(:success) { -> { should
redirect_to resource } }
    let(:failure) { -> { should
render_template :new } }
  end
  it_behaves_like :destroy do
    let(:success) { -> { should
redirect_to :submissions } }
  end
  describe '#collection' do
    context do
      before {
subject.instance_variable_set
:@collection, :collection }
      its(:collection) { should eq
:collection }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(page: 41) }
      before do
        expect(subject).to
receive_message_chain(:submissions,
:includes).with(:compiler, :user, problem:
:user) do
          double.tap do |a|
            expect(a).to
receive(:order).with(created_at: :desc) do
              double.tap { |b|
expect(b).to
receive(:page).with(41).and_return(:collec
tion) }
            end
          end
        end
      end
      its(:collection) { should eq
:collection }
    end
  end
  describe '#submissions' do
    before { allow(subject).to
receive(:parent).and_return(parent) }
    context do
      let(:parent) { double submissions:
:submissions }
      its(:submissions) { should eq
:submissions }
    end
    context do
      let(:parent) { nil }
      its(:submissions) { should eq
Submission.all }
    end
  end
  describe '#parent' do
    context do
      before {
subject.instance_variable_set :@parent,
:parent }
      its(:parent) { should eq :parent }
    end
    context do
      before { allow(subject).to
receive(:params).and_return(problem_id:
89) }
      before { expect(Problem).to
receive(:find).with(89).and_return(:parent
) }
      its(:parent) { should eq :parent }
    end
    context do
      before { allow(subject).to
receive(:params).and_return(group_id: 89)
}
      before { expect(Group).to
receive(:find).with(89).and_return(:parent
) }
      its(:parent) { should eq :parent }
    end
    context do
      before { allow(subject).to
receive(:params).and_return(user_id: 89) }
      before { expect(User).to
receive(:find).with(89).and_return(:parent
) }
      its(:parent) { should eq :parent }
    end
  end
  describe '#resource' do
    context do
      before {
subject.instance_variable_set :@resource,
:resource }
      its(:resource) { should eq :resource
}
    end
    context do
      before { expect(subject).to
receive(:params).and_return(id: 74) }
      before do
        expect(Submission).to
receive(:find).with(74) do
          double.tap { |a| expect(a).to
receive(:decorate).and_return(:resource) }
        end
      end
      its(:resource) { should eq :resource
}
    end
  end
  describe '#resource_params' do
    let(:params) { acp_submission: {
compiler_id: '128', source: '' } }
    before { expect(subject).to
receive(:params).and_return(params) }
    before { expect(subject).to
receive(:current_user).and_return(:current
_user) }
    its(:resource_params) { should eq
params[:submission].permit!.merge(user:
:current_user) }
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:resource_params).and_return(:reso
urce_params) }
    before do
      expect(subject).to
receive_message_chain(:parent,
:submissions,
:new).with(:resource_params).and_return(:r
esource)
    end
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
Файл
spec/controllers/archives_controller_spec.
rb
require 'rails_helper'
RSpec.describe ArchivesController, type:
:controller do
  it_behaves_like :new
  it_behaves_like :create do
    let(:resource) { double }
    let(:success) { -> { should
respond_with 204 } }
    let(:failure) { -> { should
render_template :new } }
  end
  describe '#resource' do
    before { subject.instance_variable_set
:@resource, :resource }
    its(:resource) { should eq :resource }
  end
  describe '#resource_params' do
    let(:params) { acp_archive: { file:
'', channel_id: 'XXX-YYY-ZZZ' } }
    before { expect(subject).to
receive(:params).and_return(params) }
    before { expect(subject).to
receive(:current_user).and_return(:current
_user) }
    its(:resource_params) { should eq
params[:archive].permit!.merge(user:
:current_user) }
  end
  describe '#initialize_resource' do
    before { expect(SecureRandom).to
receive(:uuid).and_return('XXX-YYY-ZZZ') }
    before { expect(Archive).to
receive(:new).with(channel_id: 'XXX-YYY-
ZZZ').and_return(:resource) }
    before { subject.send
:initialize_resource }
    its(:resource) { should eq :resource }
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:resource_params).and_return(:reso
urce_params) }
    before { expect(Archive).to
receive(:new).with(:resource_params).and_r
eturn(:resource) }
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
Файл
spec/controllers/sessions_controller_spec.
rb
require 'rails_helper'
RSpec.describe SessionsController, type:
:controller do
  it_behaves_like :new, anonymous: true
  describe '#create' do
    let(:resource) { Session.new
auth_token: stub_model(AuthToken, id:
'408a0176-06ef-430e-a97e-634cfb5bdfc4') }
    let(:failure) { -> { should
render_template :new } }
    context do
      let :success do
        lambda do

expect(cookies.encrypted[:auth_token]).to
eq('408a0176-06ef-430e-a97e-634cfb5bdfc4')
          should redirect_to :profile
        end
      end
      before { expect(subject).to
receive(:verify_recaptcha).and_return(true
) }
      it_behaves_like :create, anonymous:
true
    end
    context do
      let :success do
        lambda do

expect(cookies.encrypted[:auth_token]).to
eq('408a0176-06ef-430e-a97e-634cfb5bdfc4')
          expect(session[:redirect]).to
be_nil
          should redirect_to '/problems/1'
        end
      end
      before { session[:redirect] =
'/problems/1' }
      before { expect(subject).to
receive(:verify_recaptcha).and_return(true
) }
      it_behaves_like :create, anonymous:
true
    end
    context do
      before { allow(subject).to
receive(:resource).and_return(resource) }
      before { expect(subject).to
receive(:authorize).with(resource).and_ret
urn(true) }
```

```
      before { expect(subject).to
receive(:build_resource) }
      before { expect(subject).to
receive(:verify_recaptcha).and_return(fals
e) }
      before { expect(resource).to_not
receive(:save) }
      before { post :create, format: :html
}
      it { should render_template :new }
    end
  end
  it_behaves_like :destroy do
    before {
cookies.encrypted[:auth_token] = :uuid }
    let :success do
      lambda do

expect(cookies.encrypted[:auth_token]).to
be_nil
       should redirect_to :root
      end
    end
  end
  describe '#resource' do
    context do
      before {
subject.instance_variable_set :@resource,
:resource }
      its(:resource) { should eq :resource
}
    end
    context do
      before { expect(subject).to
receive_message_chain(:cookies,
:encrypted,
:[]).with(:auth_token).and_return(:uuid) }
      before { expect(AuthToken).to
receive(:find).with(:uuid).and_return(:aut
h_token) }
      before { expect(Session).to
receive(:new).with(auth_token:
:auth_token).and_return(:resource) }
      its(:resource) { should eq :resource
}
    end
  end
  describe '#resource_params' do
    let(:params) { acp session: { email:
'one@users.com', password: 'password' } }
    before { expect(subject).to
receive(:params).and_return(params) }
    its(:resource_params) { should eq
params[:session].permit! }
  end
  describe '#initialize_resource' do
    before { expect(Session).to
receive(:new).and_return(:resource) }
    before { subject.send
:initialize_resource }
    its(:resource) { should eq :resource }
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:resource_params).and_return(:reso
urce_params) }
    before { expect(Session).to
receive(:new).with(:resource_params).and_r
eturn(:resource) }
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
Файл
spec/controllers/standings_controller_spec
.rb
require 'rails_helper'
RSpec.describe StandingsController, type:
:controller do
  it_behaves_like :show, params: {
group_id: 1 }
  describe '#resource' do
    context do
      before {
subject.instance_variable_set :@resource,
:resource }
      its(:resource) { should eq :resource
}
    end
    context do
      before { expect(subject).to
receive(:params).and_return(group_id: 49)
}
      before do
        expect(Group).to
receive(:find).with(49) do
          double.tap { |a| expect(a).to
receive(:decorate).and_return(:resource) }
        end
      end
      its(:resource) { should eq :resource
}
    end
  end
end
Файл
spec/controllers/passwords_controller_spec
.rb
require 'rails_helper'
RSpec.describe PasswordsController, type:
:controller do
  it_behaves_like :edit, anonymous: true,
unauthorized: true do
    before { expect(subject).to
receive(:authorize).with(resource,
policy_class:
PasswordPolicy).and_return(true) }
  end
  it_behaves_like :update, anonymous:
true, unauthorized: true do
    let(:resource) { double }
    let :success do
      lambda do
        should set_flash[:success]
        should redirect_to %i[new session]
      end
    end
    let :failure do
      lambda do
        should set_flash.now[:error]
        render_template :edit
      end
    end
    before { expect(subject).to
receive(:authorize).with(resource,
policy_class:
PasswordPolicy).and_return(true) }
  end
  describe '#resource' do
    context do
      before { expect(subject).to
receive(:params).and_return(token: 'XXX-
YYY-ZZZ') }
      before {
subject.instance_variable_set :@resource,
:resource }
      its(:resource) { should eq :resource
}
    end
    context do
      before { expect(subject).to
receive(:params).twice.and_return(token:
'XXX-YYY-ZZZ') }
      before { expect(User).to
receive(:find_by).with(password_recovery_t
oken: 'XXX-YYY-ZZZ').and_return(:resource)
}
      its(:resource) { should eq :resource
}
    end
    context do
      before { expect(subject).to
receive(:params).and_return(Hash.new) }
      its(:resource) { should eq nil }
    end
  end
  describe '#resource_params' do
    let(:params) { acp user: { password:
'password', password_confirmation:
'password' } }
    before { expect(subject).to
receive(:params).and_return(params) }
    its(:resource_params) { should eq
params[:user].permit!.merge(password_recov
ery_token: nil) }
  end
  describe '#authorize_resource' do
    before { expect(subject).to
receive(:resource).and_return(:resource) }
    before { expect(subject).to
receive(:authorize).with(:resource,
policy_class:
PasswordPolicy).and_return(true) }
    its(:authorize_resource) { should eq
true }
  end
end
Файл
spec/controllers/submission/retests_contro
ller_spec.rb
require 'rails_helper'
RSpec.describe
Submission::RetestsController, type:
:controller do
  it_behaves_like :create, params: {
submission_id: 2 } do
    let(:resource) { double }
    let(:parent) { stub_model Submission }
    before { allow(subject).to
receive(:parent).and_return(parent) }
    let(:success) { -> { should
redirect_to parent } }
    let :failure do
      lambda do
        should set_flash[:error]
        should redirect_to parent
      end
    end
  end
  describe '#resource' do
    before { subject.instance_variable_set
:@resource, :resource }
    its(:resource) { should eq :resource }
  end
  describe '#parent' do
    context do
      before {
subject.instance_variable_set :@parent,
:parent }
      its(:parent) { should eq :parent }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(submission_id:
3) }
      before { expect(Submission).to
receive(:find).with(3).and_return(:parent)
}
      its(:parent) { should eq :parent }
    end
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:parent).and_return(:parent) }
    before { expect(Submission::Retest).to
receive(:new).with(:parent).and_return(:re
source) }
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
Файл
spec/controllers/confirmation_request/reje
cts_controller_spec.rb
require 'rails_helper'
RSpec.describe
ConfirmationRequest::RejectsController,
type: :controller do
  it_behaves_like :create, params: {
confirmation_request_id: 10 } do
    let(:resource) { double }
    let(:success) { -> { should
redirect_to :confirmation_requests } }
    let :failure do
      lambda do
        should set_flash[:error]
        should redirect_to
:confirmation_requests
      end
    end
  end
  describe '#resource' do
    before { subject.instance_variable_set
:@resource, :resource }
    its(:resource) { should eq :resource }
  end
  describe '#parent' do
    context do
      before {
subject.instance_variable_set :@parent,
:parent }
      its(:parent) { should eq :parent }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(confirmation_r
equest_id: 89) }
      before {
expect(ConfirmationRequest).to
receive(:find).with(89).and_return(:parent
) }
      its(:parent) { should eq :parent }
    end
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:parent).and_return(:parent) }
    before {
expect(ConfirmationRequest::Reject).to
receive(:new).with(:parent).and_return(:re
source) }
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
Файл
spec/controllers/confirmation_request/acce
pts_controller_spec.rb
require 'rails_helper'
RSpec.describe
ConfirmationRequest::AcceptsController,
type: :controller do
  it_behaves_like :create, params: {
confirmation_request_id: 10 } do
    let(:resource) { double }
    let(:success) { -> { should
redirect_to :confirmation_requests } }
    let :failure do
      lambda do
        should set_flash[:error]
        should redirect_to
:confirmation_requests
      end
    end
  end
  describe '#resource' do
    before { subject.instance_variable_set
:@resource, :resource }
    its(:resource) { should eq :resource }
  end
  describe '#parent' do
    context do
      before {
subject.instance_variable_set :@parent,
:parent }
      its(:parent) { should eq :parent }
    end
    context do
```

```
      before { expect(subject).to
receive(:params).and_return(confirmation_r
equest_id: 89) }
      before {
expect(ConfirmationRequest).to
receive(:find).with(89).and_return(:parent
) }
      its(:parent) { should eq :parent }
    end
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:parent).and_return(:parent) }
    before {
expect(ConfirmationRequest::Accept).to
receive(:new).with(:parent).and_return(:re
source) }
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
```

Файл spec/controllers/problems_controller_spec.rb

```
require 'rails_helper'
RSpec.describe ProblemsController, type:
:controller do
  it_behaves_like :index, anonymous: true
  it_behaves_like :index, params: {
tag_id: 25 }, anonymous: true
  it_behaves_like :index, params: {
user_id: 25 }, anonymous: true
  it_behaves_like :show, anonymous: true
  it_behaves_like :new
  it_behaves_like :edit
  it_behaves_like :create do
    let(:resource) { stub_model Problem }
    let :success do
      lambda do
        should set_flash[:success]
        should redirect_to resource
      end
    end
    let :failure do
      lambda do
        should set_flash.now[:error]
        should render_template :new
      end
    end
  end
  it_behaves_like :update do
    let(:resource) { stub_model Problem }
    let :success do
      lambda do
        should set_flash[:success]
        should redirect_to resource
      end
    end
    let :failure do
      lambda do
        should set_flash.now[:error]
        should render_template :edit
      end
    end
  end
  it_behaves_like :destroy do
    let(:resource) { stub_model Problem }
    let(:success) { -> { should
redirect_to :problems } }
  end
  describe '#resource_params' do
    let :params do
      acp_problem: {
        memory_limit: '1000',
        time_limit: '256',
        real_time_limit: '5000',
        checker_compiler_id: '1',
        checker_source: '',
        private: '0',
        tag_ids: ['1', '2'],
        examples_attributes: [{ id: '1',
input: '', answer: '', _destroy: '' }],
        tests_attributes: [{ id: '2', num:
'b', point: '10', input: '', answer: '',
_destroy: '' }],
        translations_attributes: [{
          id: '3',
          language: 'ru',
          caption: 'a problem',
          author: '@just806me',
          text: 'Just solve it',
          technical_text: '$$ f(x, z) =
x^2 + z^2 $$',
          default: '1',
          _destroy: ''
        }]
      }
    end
    before { expect(subject).to
receive(:params).and_return(params) }
    its(:resource_params) { should eq
params[:problem].permit! }
  end
  describe '#collection' do
    context do
      before {
subject.instance_variable_set
:@collection, :collection }
      its(:collection) { should eq
:collection }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(page: 12) }
      before do
        expect(subject).to
receive_message_chain(:problems,
:includes).with(:user) do
          double.tap do |a|
            expect(a).to
receive(:order).with(:id) do
              double.tap { |b|
expect(b).to
receive(:page).with(12).and_return(:collec
tion) }
            end
          end
        end
      end
      its(:collection) { should eq
:collection }
    end
  end
  describe '#problems' do
    before { allow(subject).to
receive(:parent).and_return(parent) }
    context do
      let(:parent) { double problems:
:problems }
      its(:problems) { should eq :problems
}
    end
    context do
      let(:parent) { nil }
      its(:problems) { should eq
Problem.all }
    end
  end
  describe '#parent' do
    context do
      before {
subject.instance_variable_set :@parent,
:parent }
      its(:parent) { should eq :parent }
    end
    context do
      before { allow(subject).to
receive(:params).and_return(tag_id: 5) }
      before { expect(Tag).to
receive(:find).with(5).and_return(:parent)
}
      its(:parent) { should eq :parent }
    end
    context do
      before { allow(subject).to
receive(:params).and_return(user_id: 5) }
      before { expect(User).to
receive(:find).with(5).and_return(:parent)
}
      its(:parent) { should eq :parent }
    end
    context do
      before { allow(subject).to
receive(:params).and_return(group_id: 5) }
      before { expect(Group).to
receive(:find).with(5).and_return(:parent)
}
      its(:parent) { should eq :parent }
    end
    context do
      before { allow(subject).to
receive(:params).and_return({}) }
      its(:parent) { should be_nil }
    end
  end
  describe '#resource' do
    context do
      before {
subject.instance_variable_set :@resource,
:resource }
      its(:resource) { should eq :resource
}
    end
    context do
      before { expect(subject).to
receive(:params).and_return(id: 28) }
      before do
        expect(Problem).to
receive(:find).with(28) do
          double.tap { |a| expect(a).to
receive(:decorate).and_return(:resource) }
        end
      end
      its(:resource) { should eq :resource
}
    end
  end
  describe '#initialize_resource' do
    before { expect(subject).to
receive_message_chain(:current_user,
:problems, :new).and_return(:resource) }
    before { subject.send
:initialize_resource }
    its(:resource) { should eq :resource }
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:resource_params).and_return(:reso
urce_params) }
    before do
      expect(subject).to
receive_message_chain(:current_user,
:problems,
:new).with(:resource_params).and_return(:r
esource)
    end
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
```

Файл spec/controllers/compilers_controller_spec.rb

```
require 'rails_helper'
RSpec.describe CompilersController, type:
:controller do
  it_behaves_like :index
  it_behaves_like :new
  it_behaves_like :edit
  it_behaves_like :create do
    let(:resource) { double }
    let :success do
      lambda do
        should set_flash[:success]
        should redirect_to :compilers
      end
    end
    let :failure do
      lambda do
        should set_flash.now[:error]
        should render_template :new
      end
    end
  end
  it_behaves_like :update do
    let(:resource) { double }
    let :success do
      lambda do
        should set_flash.now[:success]
        should render_template :edit
      end
    end
    let :failure do
      lambda do
        should set_flash.now[:error]
        should render_template :edit
      end
    end
  end
  it_behaves_like :destroy do
    let(:success) { -> { should
redirect_to :compilers } }
  end
  describe '#collection' do
    context do
      before {
subject.instance_variable_set
:@collection, :collection }
      its(:collection) { should eq
:collection }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(page: 10) }
      before do
        expect(Compiler).to
receive(:order).with(:id) do
          double.tap { |a| expect(a).to
receive(:page).with(10).and_return(:collec
tion) }
        end
      end
      its(:collection) { should eq
:collection }
    end
  end
  describe '#resource' do
    context do
      before {
subject.instance_variable_set :@resource,
:resource }
      its(:resource) { should eq :resource
}
    end
    context do
      before { expect(subject).to
receive(:params).and_return(id: 28) }
      before { expect(Compiler).to
receive(:find).with(28).and_return(:resour
ce) }
      its(:resource) { should eq :resource
}
    end
  end
  describe '#resource_params' do
    let :params do
      acp_compiler: {
        name: 'gcc', version: '6.4',
time_a: '100', time_b: '1', memory_a: '0',
memory_b: '3', status: 'public'
      }
    end
    before { expect(subject).to
receive(:params).and_return(params) }
    its(:resource_params) { should eq
params[:compiler].permit! }
  end
  describe '#initialize_resource' do
    before { expect(Compiler).to
receive(:new).and_return(:resource) }
```

```
    before { subject.send
:initialize_resource }
    its(:resource) { should eq :resource }
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:resource_params).and_return(:reso
urce_params) }
    before { expect(Compiler).to
receive(:new).with(:resource_params).and_r
eturn(:resource) }
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
```

Файл spec/controllers/memberships_controller_spec.rb

```
require 'rails_helper'
RSpec.describe MembershipsController,
type: :controller do
  it_behaves_like :index
  it_behaves_like :index, params: {
group_id: 3 }
  it_behaves_like :new, params: {
group_id: 3 }
  it_behaves_like :create, params: {
group_id: 4 } do
    let(:group) { stub_model Group }
    let(:resource) { double group: group }
    let(:success) { -> { should
redirect_to group } }
    let(:failure) { -> { should
render_template :new } }
  end
  it_behaves_like :destroy do
    let(:group) { stub_model Group }
    let(:resource) { double group: group }
    let(:success) { -> { should
redirect_to group } }
  end
  describe '#parent' do
    context do
      before { expect(subject).to
receive(:params).and_return(group_id: 17)
}
      before {
subject.instance_variable_set :@parent,
:parent }
      its(:parent) { should eq :parent }
    end
    context do
      before { expect(subject).to
receive(:params).twice.and_return(group_id
: 17) }
      before { expect(Group).to
receive(:find).with(17).and_return(:parent
) }
      its(:parent) { should eq :parent }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(Hash.new) }
      before { expect(Group).to_not
receive(:find) }
      its(:parent) { should be_nil }
    end
  end
  describe '#collection' do
    context do
      before {
subject.instance_variable_set
:@collection, :collection }
      its(:collection) { should eq
:collection }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(page: 6) }
      before do
        expect(subject).to
receive_message_chain(:parent,
:pending_memberships, :order).with(id:
:desc) do
          double.tap { |a| expect(a).to
receive(:page).with(6).and_return(:collect
ion) }
        end
      end
      its(:collection) { should eq
:collection }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(page: 6) }
      before { expect(subject).to
receive(:parent).and_return(nil) }
      before do
        expect(subject).to
receive_message_chain(:current_user,
:pending_memberships, :order).with(id:
:desc) do
          double.tap { |a| expect(a).to
receive(:page).with(6).and_return(:collect
ion) }
        end
      end
      its(:collection) { should eq
:collection }
    end
  end
  describe '#resource' do
    context do
      before {
subject.instance_variable_set :@resource,
:resource }
      its(:resource) { should eq :resource
}
    end
    context do
      before { expect(subject).to
receive(:params).and_return(id: 91) }
      before { expect(Membership).to
receive(:find).with(91).and_return(:resour
ce) }
      its(:resource) { should eq :resource
}
    end
  end
  its(:resource_params) { should eq state:
:accepted }
  describe '#create_resource_params' do
    let(:params) { acp_membership: {
user_id: '', type: '' } }
    before { expect(subject).to
receive(:params).and_return(params) }
    before { expect(subject).to
receive(:parent).and_return(:parent) }
    its(:create_resource_params) { should
eq
params[:membership].permit!.merge(group:
:parent) }
  end
  describe '#initialize_resource' do
    before do
      expect(subject).to
receive_message_chain(:parent,
:pending_memberships,
:new).with(no_args).and_return(:resource)
    end
    before { subject.send
:initialize_resource }
    its(:resource) { should eq :resource }
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:create_resource_params).and_retur
n(:params) }
    before { expect(subject).to
receive(:current_user).and_return(:current
_user) }
    before { expect(MembershipFactory).to
receive(:build).with(:current_user,
:params).and_return(:resource) }
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
  describe '#policy' do
    context do
      let(:record) { stub_model Problem }
      it { expect(subject.send :policy,
record).to be_a(ProblemPolicy) }
    end
    context do
      let(:record) { stub_model Membership
}
      before { expect(subject).to
receive(:current_user).and_return(:current
_user) }
      before { expect(subject).to
receive(:parent).and_return(:parent) }
      before { expect(MembershipPolicy).to
receive(:new).with(:current_user, record,
parent: :parent).and_return(:policy) }
      it { expect(subject.send :policy,
record).to eq(:policy) }
    end
  end
end
```

Файл spec/controllers/api/workers_controller_spec.rb

```
require 'rails_helper'
RSpec.describe Api::WorkersController,
type: :controller do
  it { should be_an
Api::ApplicationController }
  it_behaves_like :create, format: :json
do
    let(:resource) { double }
    let(:success) { -> { should
render_template(:create).with_status(201)
} }
    let(:failure) { -> { should
render_template(:errors).with_status(422)
} }
  end
  it_behaves_like :update, format: :json
do
    let(:resource) { double }
    let(:success) { -> { should
respond_with 204 } }
    let(:failure) { -> { should
render_template(:errors).with_status(422)
} }
  end
  describe '#resource' do
    context do
      before {
subject.instance_variable_set :@resource,
:resource }
      its(:resource) { should eq :resource
}
    end
    context do
      before { expect(subject).to
receive(:params).and_return(id: 324) }
      before { expect(Worker).to
receive(:find).with(324).and_return(:resou
rce) }
      its(:resource) { should eq :resource
}
    end
  end
  describe '#resource_params' do
    let :params do
      acp_worker: {
        alive_at: Date.today,
        api_type: 0,
        api_version: 1,
        name: 'First Worker',
        status: 1,
        webhook_supported: false,
        ips: ['127.0.0.1', '192.168.0.1'],
        task_status: ['Slot 1: Running',
'Slot 2: Stopped']
      }
    end
    before { expect(subject).to
receive(:params).and_return(params) }
    its(:resource_params) { should eq
params[:worker].permit! }
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:resource_params).and_return(:reso
urce_params) }
    before { expect(Worker).to
receive(:new).with(:resource_params).and_r
eturn(:resource) }
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
```

Файл spec/controllers/api/constants_controller_spec.rb

```
require 'rails_helper'
RSpec.describe Api::ConstantsController,
type: :controller do
  it { should be_an
Api::ApplicationController }
  it_behaves_like :index, unauthorized:
true, format: :json
  its(:collection) { should eq Constants }
end
```

Файл spec/controllers/api/submissions_controller_spec.rb

```
require 'rails_helper'
RSpec.describe Api::SubmissionsController,
type: :controller do
  it { should be_an
Api::ApplicationController }
  it_behaves_like :index, format: :json,
unauthorized: true
  describe '#collection' do
    context do
      before {
subject.instance_variable_set
:@collection, :collection }
      its(:collection) { should eq
:collection }
    end
    context do
      before do
        expect(Submission).to
receive_message_chain(:pending,
:with_attached_source, :includes).
          with(:problem,
:compiler).and_return(:collection)
      end
      its(:collection) { should eq
:collection }
    end
  end
end
```

Файл spec/controllers/api/submissions/fails_controller_spec.rb

```
require 'rails_helper'
RSpec.describe
Api::Submission::FailController, type:
:controller do
  it { should be_an
Api::ApplicationController }
  describe '#create' do
    let(:parent) { double }
    before { expect(subject).to
receive(:authenticate!) }
    before { allow(subject).to
receive(:parent).and_return(parent) }
    context do
      before { expect(parent).to
receive(:fail!).and_return(true) }
      before { post :create, params: {
submission_id: 2 }, format: :json }
      it { should respond_with 204 }
```

```
    end
    context do
      before { expect(parent).to
receive(:fail!).and_return(false) }
      before { post :create, params: {
submission_id: 2 }, format: :json }
      it { should respond_with 422 }
    end
  end
  describe '#parent' do
    context do
      before {
subject.instance_variable_set :@parent,
:parent }
      its(:parent) { should eq :parent }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(submission_id:
56) }
      before { expect(Submission).to
receive(:find).with(56).and_return(:parent
) }
      its(:parent) { should eq :parent }
    end
  end
end
Файл
spec/controllers/api/submissions/releases_
controller_spec.rb
require 'rails_helper'
RSpec.describe
Api::Submission::ReleaseController, type:
:controller do
  it { should be_an
Api::ApplicationController }
  it_behaves_like :create, params: {
submission_id: 93 }, unauthorized: true,
format: :json do
    let(:resource) { double }
    let(:success) { -> { should
respond_with 204 } }
    let(:failure) { -> { should
respond_with 422 } }
  end
  describe '#resource' do
    before { subject.instance_variable_set
:@resource, :resource }
    its(:resource) { should eq :resource }
  end
  describe '#parent' do
    context do
      before {
subject.instance_variable_set :@parent,
:parent }
      its(:parent) { should eq :parent }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(submission_id:
79) }
      before { expect(Submission).to
receive(:find).with(79).and_return(:parent
) }
      its(:parent) { should eq :parent }
    end
  end
  describe '#resource_params' do
    let(:params) { acp release: {
test_result: 0 } }
    before { expect(subject).to
receive(:params).and_return(params) }
    its(:resource_params) { should eq
params[:release].permit! }
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:resource_params).and_return(:reso
urce_params) }
    before { expect(subject).to
receive(:parent).and_return(:parent) }
    before { expect(Release).to
receive(:new).with(:parent,
:resource_params).and_return(:resource) }
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
Файл
spec/controllers/api/submissions/takes_con
troller_spec.rb
require 'rails_helper'
RSpec.describe
Api::Submission::TakeController, type:
:controller do
  it { should be_an
Api::ApplicationController }
  describe '#create' do
    let(:parent) { double }
    before { expect(subject).to
receive(:authenticate!) }
    before { expect(subject).to
receive(:wrap_in_transaction).and_yield }
    before { allow(subject).to
receive(:parent).and_return(parent) }
    context do
      before { expect(parent).to
receive(:take!).and_return(true) }
      before { post :create, params: {
submission_id: 2 }, format: :json }
      it { should respond_with 204 }
    end
    context do
      before { expect(parent).to
receive(:take!).and_return(false) }
      before { post :create, params: {
submission_id: 2 }, format: :json }
      it { should respond_with 422 }
    end
  end
  describe '#parent' do
    context do
      before {
subject.instance_variable_set :@parent,
:parent }
      its(:parent) { should eq :parent }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(submission_id:
56) }
      before { expect(Submission).to
receive_message_chain(:lock,
:find).with(56).and_return(:parent) }
      its(:parent) { should eq :parent }
    end
  end
  describe '#wrap_in_transaction' do
    let(:foo) { double }
    before { expect(ActiveRecord::Base).to
receive(:transaction).and_yield }
    it { expect(foo).to receive(:bar) }
    after {
subject.send(:wrap_in_transaction) {
foo.bar } }
  end
end
Файл
spec/controllers/api/problems_controller_s
pec.rb
require 'rails_helper'
RSpec.describe Api::ProblemsController,
type: :controller do
  it { should be_an
Api::ApplicationController }
  it_behaves_like :show, format: :json,
unauthorized: true
  describe '#resource' do
    context do
      before {
subject.instance_variable_set :@resource,
:resource }
      its(:resource) { should eq :resource
}
    end
    context do
      before { expect(subject).to
receive(:params).and_return(id: 52) }
      before { expect(Problem).to
receive(:find).with(52).and_return(:resour
ce) }
      its(:resource) { should eq :resource
}
    end
  end
end
Файл
spec/controllers/api/compilers_controller_
spec.rb
require 'rails_helper'
RSpec.describe Api::CompilersController,
type: :controller do
  it { should be_an
Api::ApplicationController }
  it_behaves_like :index, unauthorized:
true, format: :json
  describe '#collection' do
    context do
      before {
subject.instance_variable_set
:@collection, :collection }
      its(:collection) { should eq
:collection }
    end
    context do
      before { expect(Compiler).to
receive(:all).and_return(:collection) }
      its(:collection) { should eq
:collection }
    end
  end
end
Файл
spec/controllers/api/results_controller_sp
ec.rb
require 'rails_helper'
RSpec.describe Api::ResultsController,
type: :controller do
  it { should be_an
Api::ApplicationController }
  it_behaves_like :create, format: :json
do
    let(:resource) { double }
    let(:success) { -> { should
respond_with 204 } }
    let(:failure) { -> { should
render_template(:errors).with_status(422)
} }
  end
  describe '#resource' do
    before { subject.instance_variable_set
:@resource, :resource }
    its(:resource) { should eq :resource }
  end
  describe '#resource_params' do
    let :params do
      acp result: {
        status: 6,
        log: 'Number is correct',
        memory: 109.42,
        time: 98.24,
        test_id: 12,
        submission_id: 5
      }
    end
    before { expect(subject).to
receive(:params).and_return(params) }
    its(:resource_params) { should eq
params[:result].permit! }
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:resource_params).and_return(:reso
urce_params) }
    before { expect(Result).to
receive(:new).with(:resource_params).and_r
eturn(:resource) }
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
Файл
spec/controllers/api/logs_controller_spec.
rb
require 'rails_helper'
RSpec.describe Api::LogsController, type:
:controller do
  it { should be_an
Api::ApplicationController }
  it_behaves_like :create, params: {
submission_id: 47 }, format: :json do
    let(:resource) { double }
    let(:success) { -> { should
respond_with 204 } }
    let(:failure) { -> { should
render_template(:errors).with_status(422)
} }
  end
  describe '#resource' do
    before { subject.instance_variable_set
:@resource, :resource }
    its(:resource) { should eq :resource }
  end
  describe '#parent' do
    context do
      before {
subject.instance_variable_set :@parent,
:parent }
      its(:parent) { should eq :parent }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(submission_id:
138) }
      before { expect(Submission).to
receive(:find).with(138).and_return(:paren
t) }
      its(:parent) { should eq :parent }
    end
  end
  describe '#resource_params' do
    let(:params) { acp log: { data: '',
type: 0 } }
    before { expect(subject).to
receive(:params).and_return(params) }
    its(:resource_params) { should eq
params[:log].permit! }
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:resource_params).and_return(:reso
urce_params) }
    before { expect(subject).to
receive_message_chain(:parent, :logs,
:new).with(:resource_params).and_return(:r
esource) }
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
Файл
spec/controllers/api/application_controlle
r_spec.rb
require 'rails_helper'
RSpec.describe Api::ApplicationController,
type: :controller do
  its(:current_user) { should be_nil }
  its(:default_url_options) { should eq
Hash.new }
  describe '#authenticate!' do
    let(:uuid) { SecureRandom.uuid }
    before { ENV['API_ACCESS_TOKEN'] =
uuid }
    after { subject.send :authenticate! }
    context do
      before { expect(subject).to
receive(:params).and_return({}) }
      it { expect(subject).to
receive(:head).with(401) }
```

```
    end
    context do
      before { expect(subject).to
receive(:params).and_return(access_token:
'invalid') }
      it { expect(subject).to
receive(:head).with(401) }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(access_token:
uuid) }
      it { expect(subject).to_not
receive(:head) }
    end
  end
  describe '#set_locale' do
    after { I18n.locale =
I18n.default_locale }
    before { subject.send :set_locale }
    it { expect(I18n.locale).to eq(:en) }
  end
end
Файл
spec/controllers/home_controller_spec.rb
require 'rails_helper'
RSpec.describe HomeController, type:
:controller do
  describe '#show' do
    context do
      before { expect(subject).to
receive(:current_user).and_return(nil) }
      it_behaves_like :show, anonymous:
true, unauthorized: true
    end
    context do
      before { expect(subject).to
receive(:current_user).and_return(:current
_user) }
      it { expect(subject).to
receive(:redirect_to).with(:profile) }
      after { subject.show }
    end
  end
end
Файл
spec/controllers/profiles_controller_spec.
rb
require 'rails_helper'
RSpec.describe ProfilesController, type:
:controller do
  it_behaves_like :show
  it_behaves_like :update do
    let(:resource) { double }
    let :success do
      lambda do
        should set_flash.now[:success]
        should render_template :show
      end
    end
    let :failure do
      lambda do
        should set_flash.now[:error]
        should render_template :show
      end
    end
  end
  it {
expect(subject.method(:resource).original_
name).to eq :current_user }
  describe '#resource_params' do
    let :params do
      acp user: {
        username: 'pika',
        name: 'Pikachu',
        password: 'password',
        password_confirmation: 'password',
        skills: 'lighting rod, electro
ball',
        city: 'Vinnytsia',
        institution: 'VTL'
      }
    end
    before { expect(subject).to
receive(:params).and_return(params) }
    its(:resource_params) { should eq
params[:user].permit! }
  end
end
Файл
spec/controllers/sharings_controller_spec.
rb
require 'rails_helper'
RSpec.describe SharingsController, type:
:controller do
  it_behaves_like :create, params: {
group_id: 24 } do
    let(:resource) { double }
    let(:parent) { stub_model Group }
    before { allow(subject).to
receive(:parent).and_return(parent) }
    let(:success) { -> { should
redirect_to parent } }
    let(:failure) { -> { should
render_template :new } }
  end
  describe '#resource' do
    before { subject.instance_variable_set
:@resource, :resource }
    its(:resource) { should eq :resource }
  end
  describe '#parent' do
    context do
      before {
subject.instance_variable_set :@parent,
:parent }
      its(:parent) { should eq :parent }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(group_id: 28)
}
      before { expect(Group).to
receive(:find).with(28).and_return(:parent
) }
      its(:parent) { should eq :parent }
    end
  end
  describe '#resource_params' do
    let(:params) { acp sharing: {
problem_id: 57 } }
    before { expect(subject).to
receive(:params).and_return(params) }
    before { expect(subject).to
receive(:parent).and_return(:parent) }
    its(:resource_params) { should eq
params[:sharing].permit!.merge(group:
:parent) }
  end
  describe '#initialize_resource' do
    before { expect(subject).to
receive(:parent).and_return(:parent) }
    before { expect(Sharing).to
receive(:new).with(group:
:parent).and_return(:resource) }
    before { subject.send
:initialize_resource }
    its(:resource) { should eq :resource }
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:resource_params).and_return(:reso
urce_params) }
    before { expect(Sharing).to
receive(:new).with(:resource_params).and_r
eturn(:resource) }
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
Файл
spec/controllers/tags_controller_spec.rb
require 'rails_helper'
RSpec.describe TagsController, type:
:controller do
  it_behaves_like :index, anonymous: true
  it_behaves_like :index, anonymous: true,
format: :json
  describe '#collection' do
    context do
      before {
subject.instance_variable_set :@collection
, :collection }
      its(:collection) { should eq
:collection }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(page: 51) }
      before do
        expect(Tag).to
receive(:order).with(problems_count:
:desc) do
          double.tap { |a| expect(a).to
receive(:page).with(51).and_return(:collec
tion) }
        end
      end
      its(:collection) { should eq
:collection }
    end
  end
end
Файл
spec/controllers/application_controller_sp
ec.rb
require 'rails_helper'
RSpec.describe ApplicationController,
type: :controller do
  its(:default_url_options) { should eq
language: :en }
  describe '#current_user' do
    before { allow(subject).to
receive_message_chain(:cookies,
:encrypted,
:[]).with(:auth_token).and_return(auth_tok
en) }
    context do
      let (:auth_token) { nil }
      its(:current_user) { should be_nil }
    end
    context do
      let(:auth_token) { :auth_token }
      before {
subject.instance_variable_set
:@current_user, :current_user }
      its(:current_user) { should eq
:current_user }
    end
    context do
      let(:auth_token) { :auth_token }
      before do
        expect(User).to
receive(:joins).with(:auth_tokens) do
          double.tap { |a| expect(a).to
receive(:find_by).with(auth_tokens: { id:
:auth_token }).and_return(:current_user) }
        end
      end
      its(:current_user) { should eq
:current_user }
    end
  end
  describe '#authenticate!' do
    context do
      before { expect(subject).to
receive(:current_user).and_return(nil) }
      before { expect(subject).to
receive_message_chain(:request,
:fullpath).and_return('/profile') }
      before { expect(subject).to
receive(:redirect_to).with(%i[new
session]) }
      it { expect { subject.send
:authenticate! }.to change {
session[:redirect] }.to('/profile') }
    end
    context do
      before { expect(subject).to
receive(:current_user).and_return(:current
_user) }
      it { expect(subject).to_not
receive(:redirect_to) }
      after { subject.send :authenticate!
}
    end
  end
  describe '#authorize_resource' do
    before {
subject.define_singleton_method(:resource)
{ :resource } }
    before { expect(subject).to
receive(:authorize).with(:resource).and_re
turn(true) }
    it {
expect(subject.send(:authorize_resource)).
to eq(true) }
  end
  describe '#authorize_collection' do
    before {
subject.define_singleton_method(:collectio
n) { :collection } }
    before { expect(subject).to
receive(:authorize).with(:collection).and_
return(true) }
    it {
expect(subject.send(:authorize_collection)
).to eq(true) }
  end
  describe '#set_locale' do
    after { I18n.locale =
I18n.default_locale }
    before { allow(subject).to
receive(:params).and_return(params) }
    before { subject.send :set_locale }
    context do
      let(:params) { { language: 'ru' } }
      it { expect(I18n.locale).to eq(:ru)
}
    end
    context do
      let(:params) { { language: 'invalid'
} }
      it { expect(I18n.locale).to eq(:en)
}
    end
    context do
      let(:params) { { } }
      it { expect(I18n.locale).to eq(:en)
}
    end
  end
end
Файл
spec/controllers/users_controller_spec.rb
require 'rails_helper'
RSpec.describe UsersController, type:
:controller do
  it_behaves_like :index, format: :json
  describe '#create' do
    let(:resource) { double }
    it_behaves_like :create, anonymous:
true do
      let(:success) { -> { should
redirect_to %i[new session] } }
      let(:failure) { -> {  should
render_template :new } }
      before { expect(subject).to
receive(:verify_recaptcha).and_return(true
) }
    end
    context do
      before { allow(subject).to
receive(:resource).and_return(resource) }
      before { expect(subject).to
receive(:authorize).with(resource).and_ret
urn(true) }
      before { expect(subject).to
receive(:build_resource) }
      before { expect(subject).to
receive(:verify_recaptcha).and_return(fals
e) }
```

```
      before { expect(resource).to_not
receive(:save) }
      before { post :create, format: :html
}
      it { should render_template :new }
    end
  end
  describe '#resource' do
    before { subject.instance_variable_set
:@resource, :resource }
    its(:resource) { should eq :resource }
  end
  describe '#collection' do
    context do
      before {
subject.instance_variable_set
:@collection, :collection }
      its(:collection) { should eq
:collection }
    end
    context do
      before { expect(subject).to
receive(:params).and_return(name: 'John',
page: '28').twice }
      before do
        expect(UserSearcher).to
receive(:search).with(User, name: 'John',
page: '28') do
          double.tap { |a| expect(a).to
receive(:page).with('28').and_return(:coll
ection) }
        end
      end
      its(:collection) { should eq
:collection }
    end
  end
  describe '#resource_params' do
    let :params do
      acp user: { username: 'just806me',
email: 'one@users.com', password:
'password', password_confirmation:
'password' }
    end
    before { expect(subject).to
receive(:params).and_return(params) }
    its(:resource_params) { should eq
params[:user].permit! }
  end
  describe '#initialize_resource' do
    before { expect(User).to
receive(:new).and_return(:resource) }
    before { subject.send
:initialize_resource }
    its(:resource) { should eq :resource }
  end
  describe '#build_resource' do
    before { expect(subject).to
receive(:resource_params).and_return(:reso
urce_params) }
    before { expect(User).to
receive(:new).with(:resource_params).and_r
eturn(:resource) }
    before { subject.send :build_resource
}
    its(:resource) { should eq :resource }
  end
end
Файл
spec/jobs/process_problem_archive_job_spec
..rb
require 'rails_helper'
RSpec.describe ProcessProblemArchiveJob,
type: :job do
  pending
end
Файл
spec/helpers/application_helper_spec.rb
require 'rails_helper'
RSpec.describe ApplicationHelper, type:
:helper do
  subject { helper }
  fixtures :users
  its(:language_select_options) { should
eq [['Русский', :ru], ['English', :en],
['Українська', :uk]] }
  describe '#visible_compilers' do
    context do
      before { allow(subject).to
receive(:current_user).and_return(users
:one) }
      its(:visible_compilers) { should eq
Compiler.status_public }
    end
    context do
      before { allow(subject).to
receive(:current_user).and_return(users
:two) }
      its(:visible_compilers) { should eq
Compiler.all }
    end
  end
  describe '#checker_compilers' do
    context do
      before { allow(subject).to
receive(:current_user).and_return(users
:one) }
      its(:checker_compilers) { should eq
Compiler.where.not status: :in_test }
    end
    context do
      before { allow(subject).to
receive(:current_user).and_return(users
:two) }
      its(:checker_compilers) { should eq
Compiler.all }
    end
  end
  describe '#sanitize_for_problem' do
    let(:tags) { %w[b br em i p span
strong sub sup table tbody td th thead tr
img ul ol li] }
    let(:attributes) { %w[class src alt] }
    before { expect(subject).to
receive(:sanitize).with(:text, tags: tags,
attributes:
attributes).and_return(:sanitized) }
    it {
expect(subject.sanitize_for_problem
:text).to eq(:sanitized) }
  end
end
Файл spec/services/constants_spec.rb
require 'rails_helper'
RSpec.describe Constants do
  subject { described_class }
  its :as_json do
    should eq \
      log_types: { 'source' => 0,
'checker' => 1 },
      problem_translation_languages: {
'ru' => 0, 'en' => 1, 'uk' => 2 },
      result_statuses: {
        'ok' => 0,
        'wrong_answer' => 1,
        'presentation_error' => 2,
        'fail' => 3,
        'dirt' => 4,
        'points' => 5,
        'bad_test' => 6,
        'unexpected_eof' => 8,
        'runtime_error' => 10,
        'memory_limit_exceded' => 14,
        'time_limit_exceded' => 15,
        'partilly_correct' => 16
      },
      submission_test_states: { 'pending'
=> 0, 'in_progress' => 1, 'done' => 2,
'failed' => 3 },
      submission_test_results: { 'ok' =>
0, 'compiler_error' => 1 },
      tag_translation_languages: { 'ru' =>
0, 'en' => 1, 'uk' => 2 }
  end
end
Файл spec/services/avatar_spec.rb
require 'rails_helper'
RSpec.describe Avatar, type: :model do
  let(:user) { double }
  let(:file) { double open: :open,
original_filename: 'avatar.jpg',
content_type: 'image/jpeg'  }
  subject { described_class.new user:
user, file: file }
  its(:user) { should eq user }
  its(:file) { should eq file }
  it { should be_an
ActiveModel::Validations }
  it { should be_a Draper::Decoratable }
  it { should
delegate_method(:avatar).to(:user).allow_n
il }
  it { should
delegate_method(:attached?).to(:avatar).al
low_nil }
  it { should
delegate_method(:destroy).to(:avatar).as(:
purge_later).allow_nil }
  it { should
delegate_method(:as_json).to(:decorate) }
  it { should
delegate_method(:url).to(:decorate) }
  describe '#valid?' do
    before { expect(subject).to
receive(:blob_must_be_variable) }
    before { expect(subject).to
receive(:blob_size_must_not_be_greater_tha
t_2_megabytes) }
    it { should validate_presence_of :user
}
    it { should validate_presence_of :file
}
  end
  describe '#save' do
    context do
      before { expect(subject).to
receive(:valid?).and_return(true) }
      before { expect(subject).to
receive(:blob).and_return(:blob) }
      before { expect(user).to
receive(:avatar=).with(:blob) }
    end
    context do
      before { expect(subject).to
receive(:valid?).and_return(false) }
      before { expect(subject).to_not
receive(:blob) }
    end
  end
  describe '#blob' do
    context do
      before {
subject.instance_variable_set :@blob,
:blob }
      its(:blob) { should eq :blob }
    end
    context do
      let(:options) { { io: :open,
filename: 'avatar.jpg', content_type:
'image/jpeg' } }
      before {
expect(ActiveStorage::Blob).to
receive(:build_after_upload).with(options)
.and_return(:blob) }
      its(:blob) { should eq :blob }
    end
  end
  describe '#blob_must_be_variable' do
    let(:call) { -> { subject.send
:blob_must_be_variable } }
    let(:errors) { -> {
subject.errors.details } }
    context do
      let(:file) { nil }
      before { expect(subject).to_not
receive(:blob) }
      it { expect(&call).to_not
change(&errors) }
    end
    context do
      before { expect(subject).to
receive(:blob).and_return(double
variable?: true) }
      it { expect(&call).to_not
change(&errors) }
    end
    context do
      before { expect(subject).to
receive(:blob).and_return(double
variable?: false) }
      it { expect(&call).to
change(&errors).to(file: [{ error:
:invalid }]) }
    end
  end
  describe
'#blob_size_must_not_be_greater_that_2_meg
abytes' do
    let(:call) { -> { subject.send
:blob_size_must_not_be_greater_that_2_mega
bytes } }
    let(:errors) { -> {
subject.errors.details } }
    context do
      let(:file) { nil }
      before { expect(subject).to_not
receive(:blob) }
      it { expect(&call).to_not
change(&errors) }
    end
    context do
      before { expect(subject).to
receive(:blob).and_return(double
byte_size: 2.megabytes) }
      it { expect(&call).to_not
change(&errors) }
    end
    context do
      before { expect(subject).to
receive(:blob).and_return(double
byte_size: 3.megabytes) }
      it { expect(&call).to
change(&errors).to(file: [{ error:
:too_long, count: 2097152 }]) }
    end
  end
end
Файл spec/services/archive_spec.rb
require 'rails_helper'
RSpec.describe Archive, type: :model do
  let(:uuid) { SecureRandom.uuid }
  let(:user) { stub_model User }
  let(:file) {
ActionDispatch::Http::UploadedFile.new
tempfile: double(path: '/uploads/01'),
type: 'application/zip' }
  subject { described_class.new file:
file, channel_id: uuid, user: user }
  its(:file) { should eq file }
  its(:channel_id) { should eq uuid }
  its(:user) { should eq user }
  its(:to_key) { should be_nil }
  its(:persisted?) { should be false }
  its(:to_model) { should eq subject }
  it { should validate_presence_of :file }
  it { should validate_presence_of
:channel_id }
  it { should validate_presence_of :user }
  describe '#save' do
    context do
      before { expect(subject).to
receive(:valid?).and_return(false) }
      before {
expect(ProcessProblemArchiveJob).to_not
receive(:perform_later) }
      its(:save) { should be false }
    end
    context do
      before { expect(subject).to
receive(:valid?).and_return(true) }
```

```
      before { expect(subject).to
receive(:filename).and_return('/tmp/01').t
wice }
      before { expect(FileUtils).to
receive(:copy).with('/uploads/01',
'/tmp/01') }
      before {
expect(ProcessProblemArchiveJob).to
receive(:perform_later).with(user,
'/tmp/01', uuid) }
      its(:save) { should be true }
    end
  end
  describe '#filename' do
    context do
      before {
subject.instance_variable_set :@filename,
:filename }
      its(:filename) { should eq :filename
}
    end
    context do
      before { expect(SecureRandom).to
receive(:uuid).and_return(uuid) }
      its(:filename) { should eq File.join
Dir.tmpdir, uuid }
    end
  end
end
Файл
spec/services/submission/retest_spec.rb
require 'rails_helper'
RSpec.describe Submission::Retest do
  let(:submission) { stub_model Submission
}
  subject { described_class.new submission
}
  its(:submission) { should eq submission
}
  describe '#save' do
    before { expect(Submission).to
receive(:transaction).and_yield }
    before { expect(submission).to
receive_message_chain(:results,
:delete_all) }
    before { expect(submission).to
receive_message_chain(:logs, :delete_all)
}
    before do
      expect(submission).to
receive(:update).
        with(test_state: :pending,
fails_count: 0, score: nil, test_result:
nil, max_score: nil).and_return(:result)
    end
    its(:save) { should eq :result }
  end
end
Файл
spec/services/confirmation_request/reject_
spec.rb
require 'rails_helper'
RSpec.describe ConfirmationRequest::Reject
do
  let(:confirmation_request) { stub_model
ConfirmationRequest }
  subject { described_class.new
confirmation_request }
  its(:confirmation_request) { should eq
confirmation_request }
  it { should
delegate_method(:pending?).to(:confirmatio
n_request).with_prefix }
  describe '#save' do
    before {
expect(confirmation_request).to
receive(:update).with(status:
:rejected).and_return(:result) }
    its(:save) { should eq :result }
  end
end
Файл
spec/services/confirmation_request/accept_
spec.rb
require 'rails_helper'
RSpec.describe ConfirmationRequest::Accept
do
  let(:confirmation_request) { stub_model
ConfirmationRequest }
  subject { described_class.new
confirmation_request }
  its(:confirmation_request) { should eq
confirmation_request }
  it { should
delegate_method(:pending?).to(:confirmatio
n_request).with_prefix }
  describe '#save' do
    let(:user) { double }
    before {
allow(confirmation_request).to
receive(:user).and_return(user) }
    context do
      before {
expect(confirmation_request).to
receive(:update).with(status:
:accepted).and_return(true) }
      before { expect(user).to
receive(:update!).with(roles: :confirmed)
}
      its(:save) { should be true }
    end
    context do
      before {
expect(confirmation_request).to
receive(:update).with(status:
:accepted).and_return(false) }
      its(:save) { should be false }
    end
  end
end
Файл
spec/services/standing_redis_store_spec.rb
require 'rails_helper'
RSpec.describe StandingRedisStore do
  subject { described_class.new :user_id,
:problem_id }
  it {
expect(subject.instance_variable_get
:@key).to eq('user_id_problem_id') }
  context do
    let(:key) {
subject.instance_variable_get :@key }
    after { $redis_standings.del key }
    describe '#get' do
      context do
        before { expect(subject).to
receive(:score_from_database).and_return(2
0) }
        its(:get) { should eq('20') }
      end
      context do
        before { $redis_standings.set key,
50 }
        before { expect(subject).to_not
receive(:score_from_database) }
        its(:get) { should eq('50') }
      end
    end
    describe '#update_if_exists' do
      context do
        before { expect(subject).to_not
receive(:score_from_database) }
        before { subject.update_if_exists
}
        it { expect($redis_standings.get
key).to be_nil }
      end
      context do
        before { $redis_standings.set key,
50 }
        before { expect(subject).to
receive(:score_from_database).and_return(1
00) }
        before { subject.update_if_exists
}
        it { expect($redis_standings.get
key).to eq('100') }
      end
    end
    describe '#score_from_database' do
      let(:params) { { user_id: :user_id,
problem_id: :problem_id, test_state:
:done, test_result: :ok } }
      before do
        expect(Submission).to
receive(:select).with(:score) do
          double.tap do |a|
            expect(a).to
receive(:order).with(created_at: :desc) do
              double.tap { |b|
expect(b).to
receive(:find_by).with(params).and_return(
submission) }
            end
          end
        end
      end
      context do
        let(:submission) { double score:
90 }
        its(:score_from_database) { should
eq 90 }
      end
      context do
        let(:submission) { nil }
        its(:score_from_database) { should
be_nil }
      end
    end
  end
  describe '.get' do
    it do
      expect(described_class).to
receive(:new).with(:user_id, :problem_id)
do
        double.tap { |a| expect(a).to
receive(:get) }
      end
    end
    after { described_class.get :user_id,
:problem_id }
  end
  describe '.update_if_exists' do
    it do
      expect(described_class).to
receive(:new).with(:user_id, :problem_id,
55) do
        double.tap { |a| expect(a).to
receive(:update_if_exists) }
      end
    end
    after {
described_class.update_if_exists :user_id,
:problem_id, 55 }
  end
end
Файл
spec/services/password_recovery_spec.rb
require 'rails_helper'
RSpec.describe PasswordRecovery, type:
:model do
  subject { described_class.new email:
'one@users.com', password: 'password' }
  its(:email) { should eq 'one@users.com'
}
  its(:to_key) { should be_nil }
  its(:persisted?) { should be false }
  describe '#valid?' do
    before { expect(subject).to
receive(:user_must_be_present) }
    it { should validate_presence_of
:email }
  end
  describe '#save' do
    context do
      let(:user) { stub_model User }
      before { expect(subject).to
receive(:valid?).and_return(true) }
      before { expect(SecureRandom).to
receive(:uuid).and_return(:uuid) }
      before { expect(subject).to
receive(:user).twice.and_return(user) }
      before { expect(user).to
receive(:update!).with(password_recovery_t
oken: :uuid) }
      before do
        expect(PasswordRecoveryMailer).to
receive(:email).with(user) do
          double.tap { |a| expect(a).to
receive(:deliver_later) }
        end
      end
      it { expect { subject.save }.to_not
raise_error }
    end
    context do
      before { expect(subject).to
receive(:valid?).and_return(false) }
      before { expect(subject).to_not
receive(:user) }
      it { expect { subject.save }.to_not
raise_error }
    end
  end
  describe '#user' do
    context do
      before {
subject.instance_variable_set :@user,
:user }
      its(:user) { should eq :user }
    end
    context do
      before { expect(User).to
receive(:find_by).with('lower(email) =
lower(?)',
'one@users.com').and_return(:user) }
      its(:user) { should eq :user }
    end
  end
  describe '#user_must_be_present' do
    let(:call) { -> { subject.send
:user_must_be_present } }
    let(:errors) { -> {
subject.errors.details } }
    context do
      subject { described_class.new }
      it { expect(&call).to_not
change(&errors) }
    end
    context do
      before { allow(subject).to
receive(:user).and_return(double) }
      it { expect(&call).to_not
change(&errors) }
    end
    context do
      before { allow(subject).to
receive(:user).and_return(nil) }
      it { expect(&call).to
change(&errors).to(email: [{ error:
:invalid }]) }
    end
  end
end
Файл spec/services/session_spec.rb
require 'rails_helper'
RSpec.describe Session, type: :model do
  subject { described_class.new email:
'one@users.com', password: 'password' }
  its(:email) { should eq 'one@users.com'
}
  its(:password) { should eq 'password' }
  its(:to_key) { should be_nil }
  its(:persisted?) { should be false }
  it { should
delegate_method(:destroy).to(:auth_token)
}
  describe '#valid?' do
    before { expect(subject).to
receive(:user_must_be_present) }
```

```
    before { expect(subject).to
receive(:password_must_pass_authentication
) }
    it { should validate_presence_of
:email }
    it { should validate_presence_of
:password }
  end
  describe '#save' do
    context do
      before { expect(subject).to
receive(:valid?).and_return(false) }
      before { expect(subject).to_not
receive(:auth_token) }
      its(:save) { should be false }
    end
    context do
      before { expect(subject).to
receive(:valid?).and_return(true) }
      before { expect(subject).to
receive_message_chain(:auth_token,
:save).and_return(:result) }
      its(:save) { should eq :result }
    end
  end
  describe '#user' do
    context do
      before {
subject.instance_variable_set :@user,
:user }
      its(:user) { should eq :user }
    end
    context do
      before { expect(User).to
receive(:find_by).with('lower(email) =
lower(?)',
'one@users.com').and_return(:user) }
      its(:user) { should eq :user }
    end
  end
  describe '#auth_token' do
    context do
      before {
subject.instance_variable_set
:@auth_token, :auth_token }
      its(:auth_token) { should eq
:auth_token }
    end
    context do
      before { expect(subject).to
receive_message_chain(:user, :auth_tokens,
:build).and_return(:auth_token) }
      its(:auth_token) { should eq
:auth_token }
    end
  end
  describe '#user_must_be_present' do
    let(:call) { -> { subject.send
:user_must_be_present } }
    let(:errors) { -> {
subject.errors.details } }
    context do
      subject { described_class.new }
      it { expect(&call).to_not
change(&errors) }
    end
    context do
      before { allow(subject).to
receive(:user).and_return(double) }
      it { expect(&call).to_not
change(&errors) }
    end
    context do
      before { allow(subject).to
receive(:user).and_return(nil) }
      it { expect(&call).to
change(&errors).to(email: [{ error:
:invalid }]) }
    end
  end
  describe
'#password_must_pass_authentication' do
    let(:call) { -> { subject.send
:password_must_pass_authentication } }
    let(:errors) { -> {
subject.errors.details } }
    let(:user) { double }
    before { allow(subject).to
receive(:user).and_return(user) }
    context do
      subject { described_class.new }
      it { expect(&call).to_not
change(&errors) }
    end
    context do
      let(:user) { nil }
      it { expect(&call).to_not
change(&errors) }
    end
    context do
      before { expect(user).to
receive(:authenticate).with('password').an
d_return(true) }
      it { expect(&call).to_not
change(&errors) }
    end
    context do
      before { expect(user).to
receive(:authenticate).with('password').an
d_return(false) }
      it { expect(&call).to
change(&errors).to(password: [{ error:
:invalid }]) }
    end
  end
end
Файл spec/services/release_spec.rb
require 'rails_helper'
RSpec.describe Release do
  let(:problem) { stub_model Problem, id:
22 }
  let(:submission) { stub_model
Submission, problem: problem }
  subject { described_class.new
submission, test_result: :ok }
  describe '#score' do
    before do
      expect(submission).to
receive_message_chain(:results,
:joins).with(:test) do
        double.tap do |a|
          expect(a).to
receive(:where).with(status: :ok) do
            double.tap { |b| expect(b).to
receive(:sum).with('tests.point').and_retu
rn(:score) }
          end
        end
      end
    end
    its(:score) { should eq :score }
  end
  describe '#max_score' do
    before do
      expect(Test).to
receive_message_chain(:unscoped,
:where).with(problem_id: 22) do
        double.tap { |a| expect(a).to
receive(:sum).with(:point).and_return(:max
_score) }
      end
    end
    its(:max_score) { should eq :max_score
}
  end
  describe '#save' do
    context do
      before { expect(submission).to
receive(:release).and_return(false) }
      before { expect(submission).to_not
receive(:update) }
      its(:save) { should be false }
    end
    context do
      before { expect(submission).to
receive(:release).and_return(true) }
      before { expect(subject).to
receive(:score).and_return(:score) }
      before { expect(subject).to
receive(:max_score).and_return(:max_score)
}
      before do
        expect(submission).to \

receive(:update).with(test_result: :ok,
score: :score, max_score:
:max_score).and_return(:result)
      end
      its(:save) { should eq :result }
    end
  end
end
Файл Gemfile
source 'https://rubygems.org'
git_source(:github) { |repo|
"https://github.com/#{repo}.git" }
gem 'rails', '~> 5.2'
gem 'puma'
gem 'sassc-rails'
gem 'uglifier'
gem 'turbolinks'
gem 'turbolinks_render'
gem 'bcrypt'
gem 'bootsnap', require: false
gem 'tzinfo-data', platforms: [:mingw,
:mswin, :x64_mingw, :jruby]
gem 'dotenv-rails'
gem 'pg'
gem 'email_validator'
gem 'kaminari'
gem 'simple_form'
gem 'pundit'
gem 'jquery-rails'
gem 'mini_racer', platforms: :ruby
gem 'mini_magick'
gem 'hiredis'
gem 'redis', require: %w(redis
redis/connection/hiredis)
gem 'redis-namespace'
gem 'draper'
gem 'aasm'
gem 'rubyzip', require: 'zip'
gem 'nokogiri'
gem 'rails-i18n'
gem 'russian'
gem 'cocoon'
gem 'bootstrap'
gem 'rails-assets-sweetalert2', '7.29.1',
source: 'https://rails-assets.org'
gem 'sweet-alert2-rails', github:
'just806me/sweet-alert2-rails'
gem 'bitmask_attributes'
gem 'recaptcha'
group :development, :test do
  gem 'rspec-rails'
end
group :development do
  gem 'listen', '>= 3.0.5', '< 3.2'
  gem 'spring'
  gem 'spring-watcher-listen'
  gem 'i18n-tasks'
  gem 'capistrano', require: false
  gem 'capistrano-rails', require: false
  gem 'faker', require: false
  gem 'bullet'
end
group :test do
  gem 'shoulda-matchers'
  gem 'shoulda-callback-matchers', github:
'just806me/shoulda-callback-matchers'
  gem 'rspec-its'
  gem 'rspec-activemodel-mocks'
  gem 'rails-controller-testing'
end
Файл lib/tasks/tags.rake
namespace :tags do
  task export: :environment do
    tags = Tag.find_each.map do |tag|
      { id: tag.id,
translations_attributes:
tag.translations.as_json(only: %i[language
name]) }
    end
    open('tags.json', 'wb+') { |f| f.write
tags.to_json }
  end
  task import: :environment do
    Tag.accepts_nested_attributes_for
:translations
    open('tags.json', 'r') { |f|
Tag.create! JSON.parse f.read }
  end
end
Файл
lib/components/input_group_component.rb
module InputGroup
  def prepend wrapper_options = nil
    span_tag = content_tag :span,
options[:prepend], class: "input-group-
text"
    template.content_tag :div, span_tag,
class: "input-group-prepend"
  end
  def append wrapper_options = nil
    span_tag = content_tag :span,
options[:append], class: "input-group-
text"
    template.content_tag :div, span_tag,
class: "input-group-append"
  end
end
SimpleForm.include_component InputGroup
Файл lib/capistrano/tasks/application.rake
namespace :application do
  desc 'Start application'
  task :start do
    on roles(:app), in: :sequence, wait: 5
do
      within current_path do
        execute 'sudo systemctl start
application'
      end
    end
  end
  desc 'Stop application'
  task :stop do
    on roles(:app), in: :sequence, wait: 5
do
      within current_path do
        execute 'sudo systemctl stop
application'
      end
    end
  end
  desc 'Restart application'
  task :restart do
    on roles(:app), in: :sequence, wait: 5
do
      within current_path do
        execute 'sudo systemctl restart
application'
      end
    end
  end
end
Файл lib/capistrano/tasks/nginx.rake
namespace :nginx do
  desc 'Reload nginx'
  task :reload do
    on roles(:app), in: :sequence, wait: 5
do
      within current_path do
        execute 'sudo systemctl reload
nginx'
      end
    end
  end
end
```

Додаток Д

Ілюстративна частина

РОЗРОБКА МЕТОДУ І ПРОГРАМНИХ ЗАСОБІВ СИСТЕМИ ТРЕНУВАННЯ І ОЦІНЮВАННЯ РОБІТ ЗІ СПОРТИВНОГО ПРОГРАМУВАННЯ. ЧАСТИНА 2 КЛІЄНТСЬКИЙ ДОДАТОК

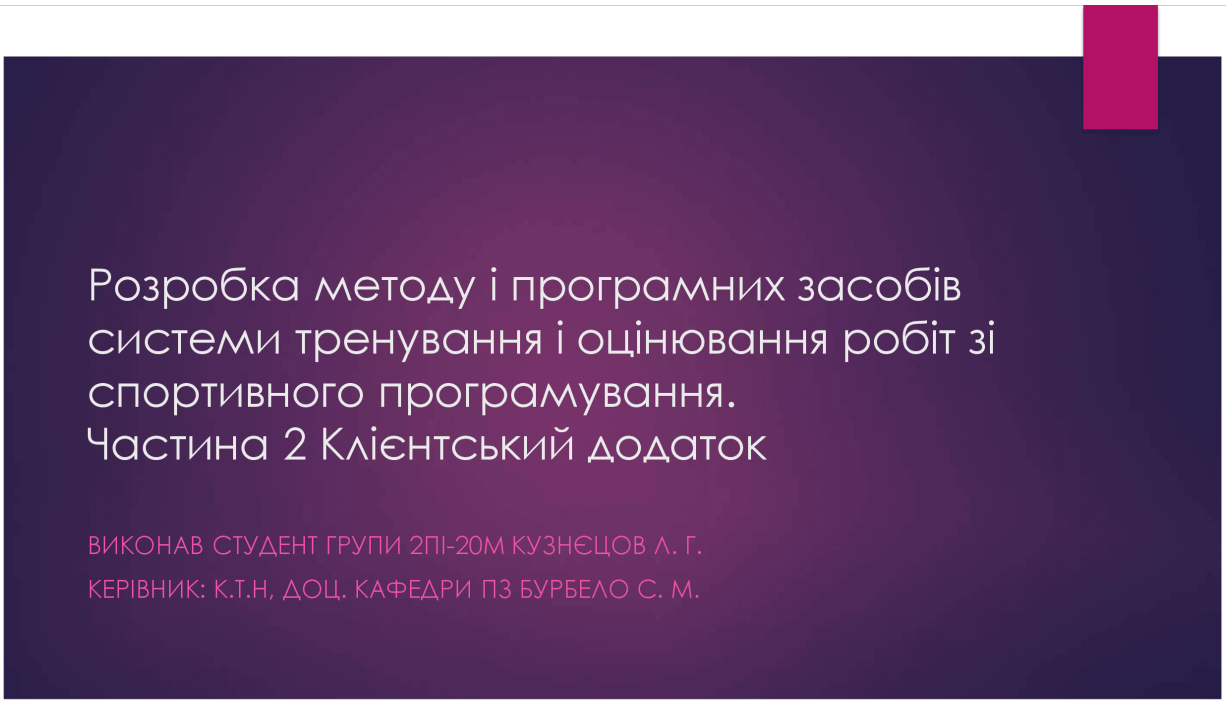


Рисунок Д.1 – Титульний лист презентації

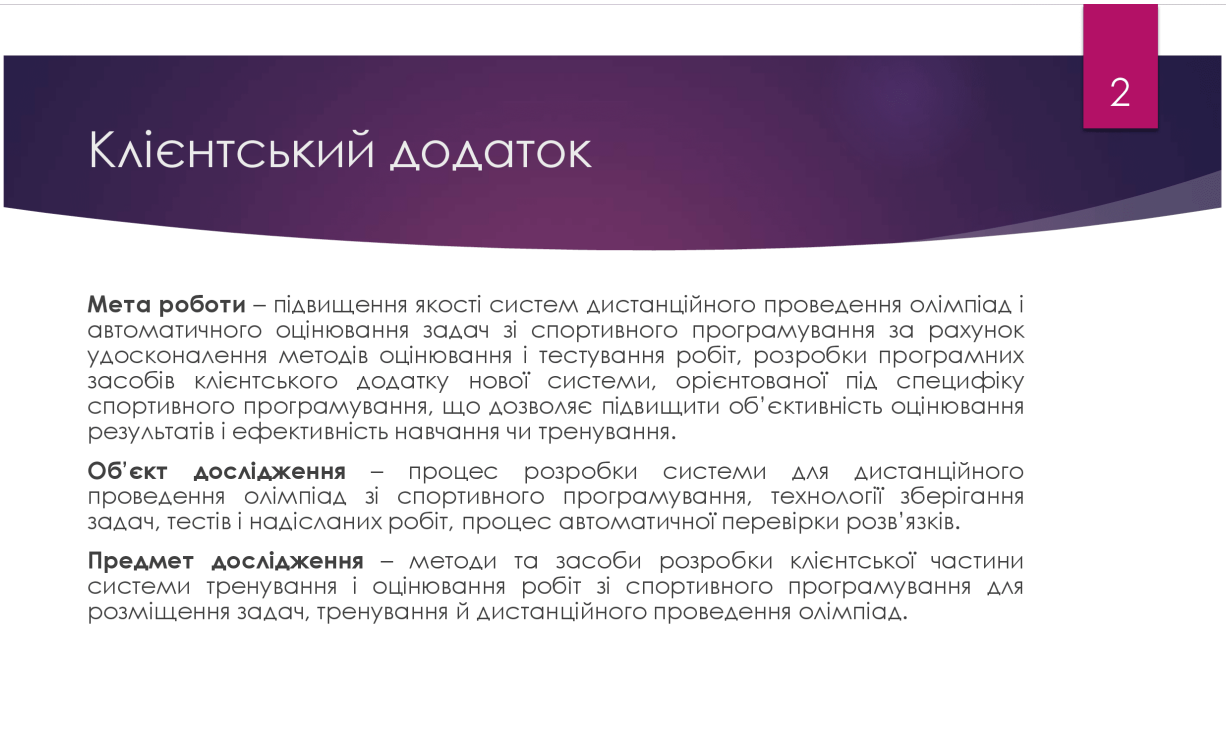


Рисунок Д.2 – Мета, об’єкт, предмет

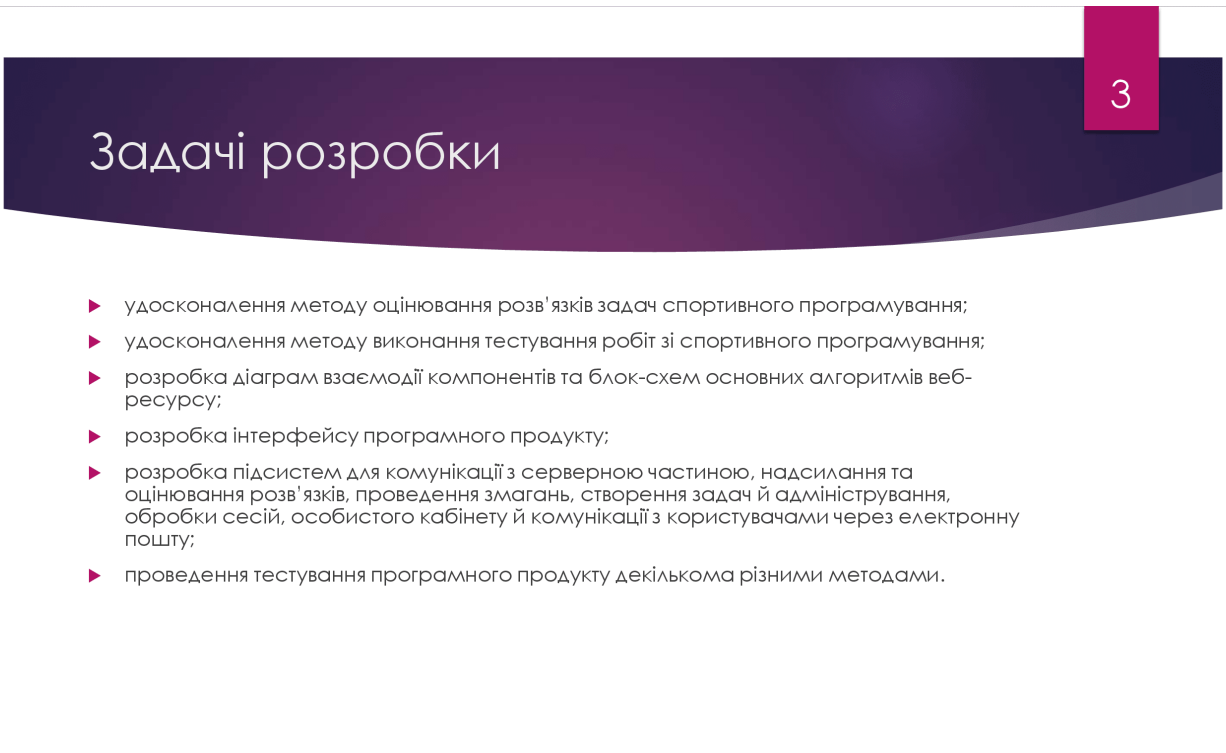


Рисунок Д.3 – Задачі розробки

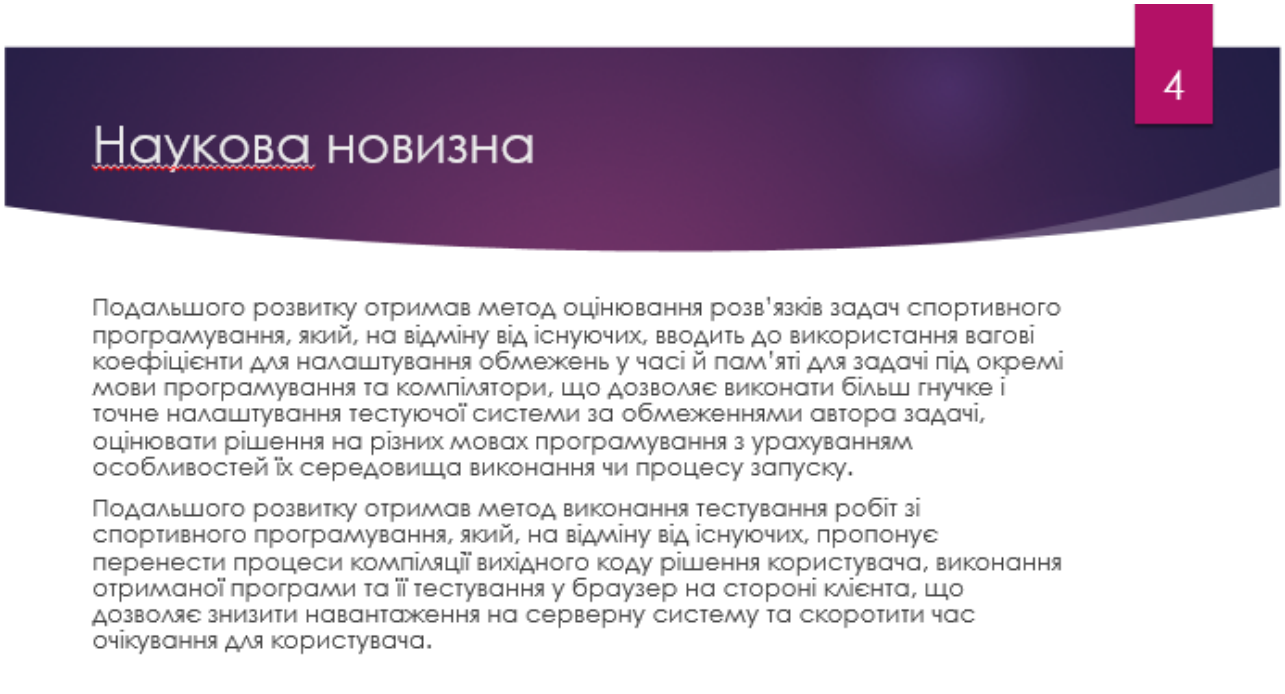


Рисунок Д.4 – Наукова новизна

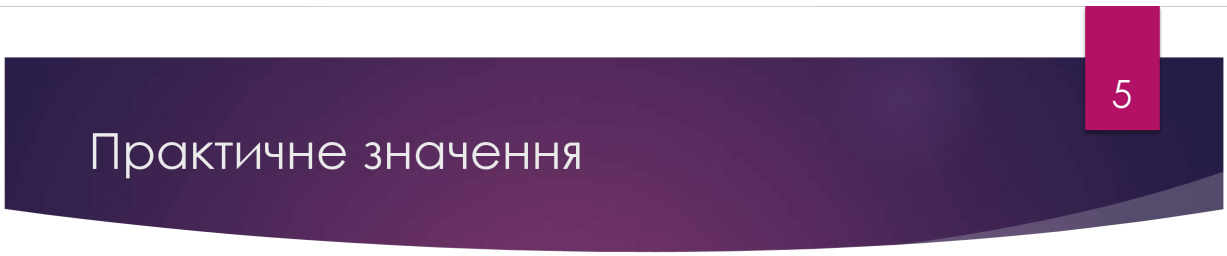


Рисунок Д.5 – Практичне значення

Рисунок Д.6 – Аналіз аналогів

7

Codeforces

Тренувальна й навчальна веб-платформа, створена групою у Саратовському університеті. На платформі регулярно проводяться змагання, а минулі можуть бути використані для підготовки.

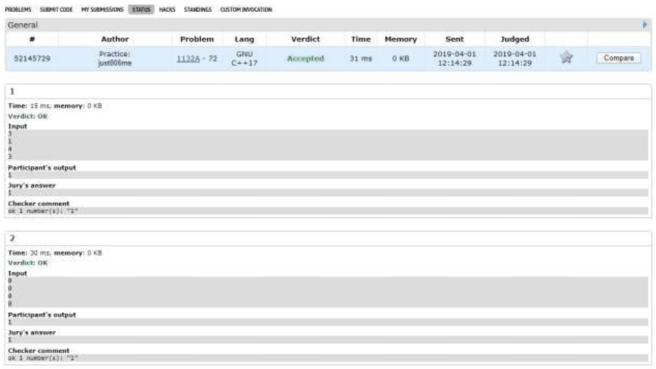


Рисунок Д.7 – Аналог Codeforces

8

E-Olymp

Веб-портал організаційно-методичного забезпечення дистанційних олімпіад, що розроблено Житомирським державним університетом ім. І. Франка. Надає можливість тренуватися розв'язувати задачі з програмування, брати участь в змаганнях, а також містить деякі навчальні матеріали.

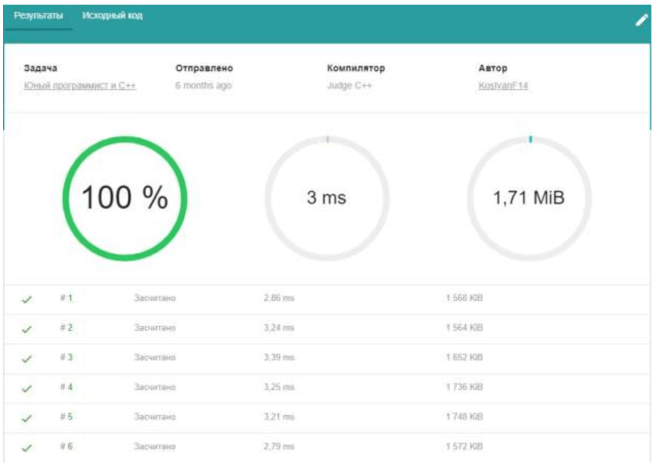


Рисунок Д.8 – Аналог E-Olymp

9

Ejudge

Система для проведення заходів, в яких необхідна автоматична перевірка програм. Підтримується системи ACM, Kirov, Moscow, Olympiad. Для роботи потрібно встановити на власному сервері. Часто використовується для проведення деяких етапів олімпіади ICPC та інших.

Рисунок Д.9 – Аналог Ejudge

10

Algotester

Система автоматичного тестування розв'язків з великим набором цікавих алгоритмічних задач різноманітної складності. Розроблено на базі ЛНУ ім. І. Франка. Регулярно використовується для проведення змагань й тренувань. Дозволяється реєстрація для відстеження власного прогресу.

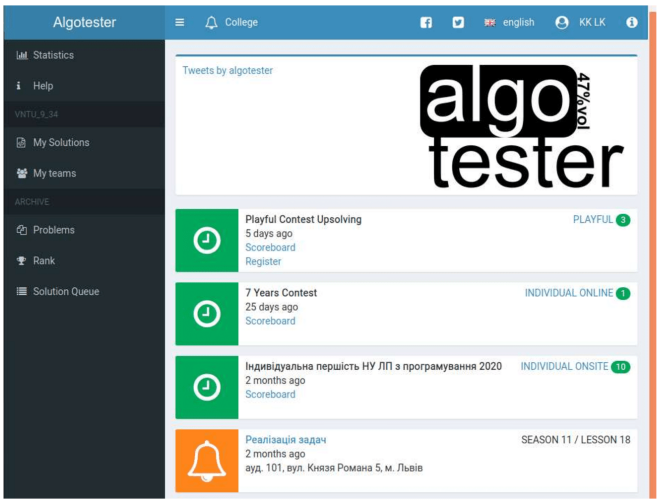


Рисунок Д.10 – Аналог Algotester

11

Порівняльні характеристики аналогів

Критерій	Codeforces	E-olymp	Ejudge	Algotester	CodeLabs
Відкритий вихідний код	-	-	+	-	+
Особистий кабінет користувача	+	+	-	+	+
Кілька одночасно-працюючих серверів тестування	-	-	+	-	+
Перегляд деталей по кожному тесту	+	-	+	-	+
Коефіцієнти обмежень для різних мов	-	-	-	-	+
Створення власних задач	+/-	-	+	+	+
Тестування на стороні клієнту	-	-	-	-	+

Рисунок Д.11 – Порівняльні характеристики аналогів

12

Діаграма компонентів

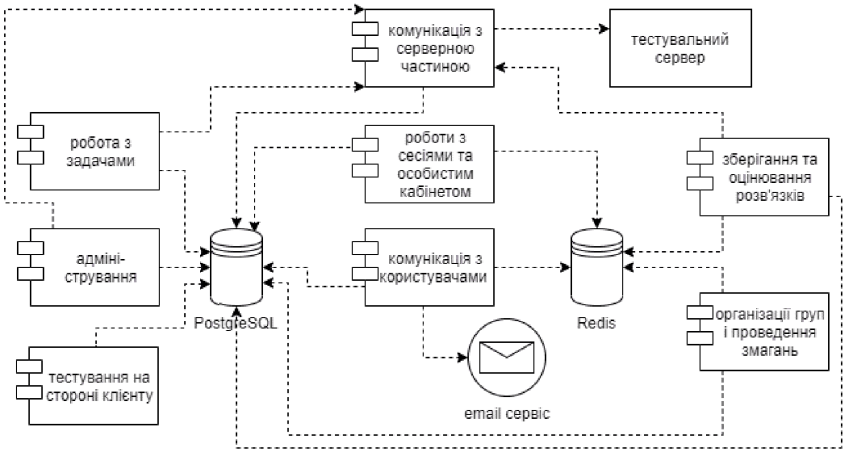


Рисунок Д.12 – Діаграма компонентів клієнтської частини

13

Діаграма послідовності

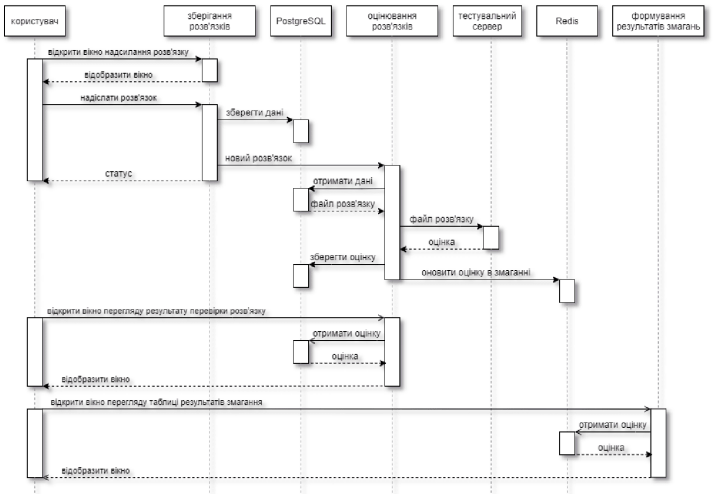


Рисунок Д.13 – Діаграма послідовності одного з варіантів використання

14

Удосконалення методу оцінювання розв’язків задач спортивного програмування

- На результат оцінювання впливають: вихідні дані, час роботи, об'єм використаної оперативної пам’яті.
- Через технічні особливості, однаковий алгоритм може виконуватись довше чи використовувати більше пам’яті на різних мовах програмування.
- Для успішної здачі розв’язків потрібно вносити корективи в технічні обмеження: занадто високі дозволять здавати неоптимальні алгоритми, занадто низькі спровокують обмеження для інтерпретованих/динамічних мов програмування.
- Для автоматичного налаштування обмежень можна скористатись ваговими коефіцієнтами, що представляють функцію відображення еталонних значень з умови задачі. Так, під час перевірки будуть використані обмеження, що відповідають показникам продуктивності обраної мови.

Рисунок Д.14 – Удосконалення методу оцінювання розв’язків задач спортивного програмування

15

Удосконалення методу тестування робіт зі спортивного програмування

- Для системи необхідно виконувати компіляцію рішень користувачів, їх запуск та аналіз результату роботи. Класичним підходом є виконання цих етапів на стороні серверу.
- При тестуванні на сервері постають питання ефективності й безпеки:
  - необхідність додаткових ресурсів серверу;
  - отримання результатів перевірки з затримкою;
  - блокування розв'язування задач при несправності серверу;
  - вірогідність отримати зловмисний код.
- Можливості WASM дозволяють перенести описані етапи у браузер клієнта без застосування серверу. При цьому вирішуються проблеми серверного підходу.
- Проте, існують недоліки:
  - код у форматі WASM у браузері працює повільніше, ніж «нативний»;
  - відкритість тестів користувачеві
  - можливість користувача впливати на процес і результат тестування;

Рисунок Д.15 – Удосконалення методу тестування робіт зі спортивного програмування

16

Блок-схема

алгоритму надсилання розв'язку

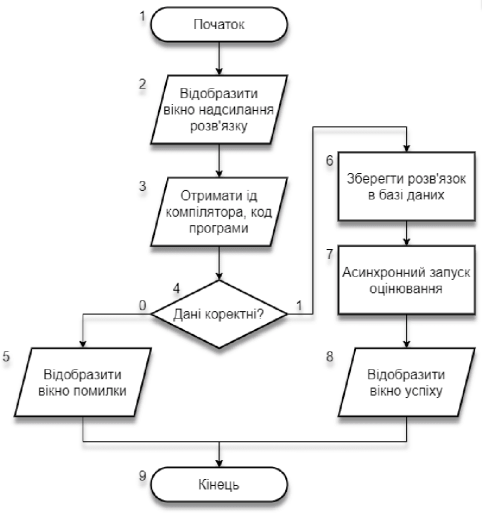


Рисунок Д.16 – Блок-схема алгоритму надсилання розв’язку

17

Блок-схема

алгоритму оцінювання розв'язку

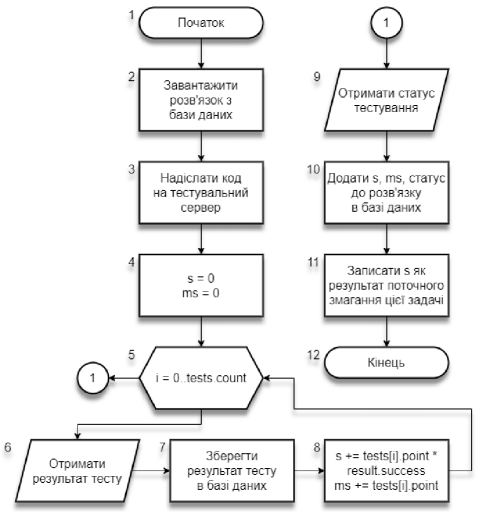


Рисунок Д.17 – Блок-схема алгоритму оцінювання розв’язку

18

Блок-схема

алгоритму формування таблиці з результатом змагань

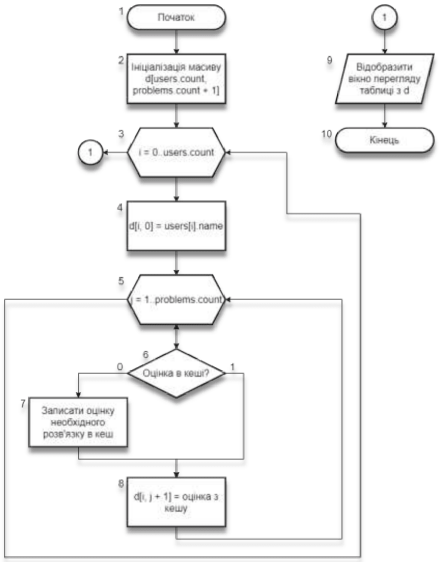


Рисунок Д.18 – Блок-схема алгоритму формування таблиці з результатом змагань

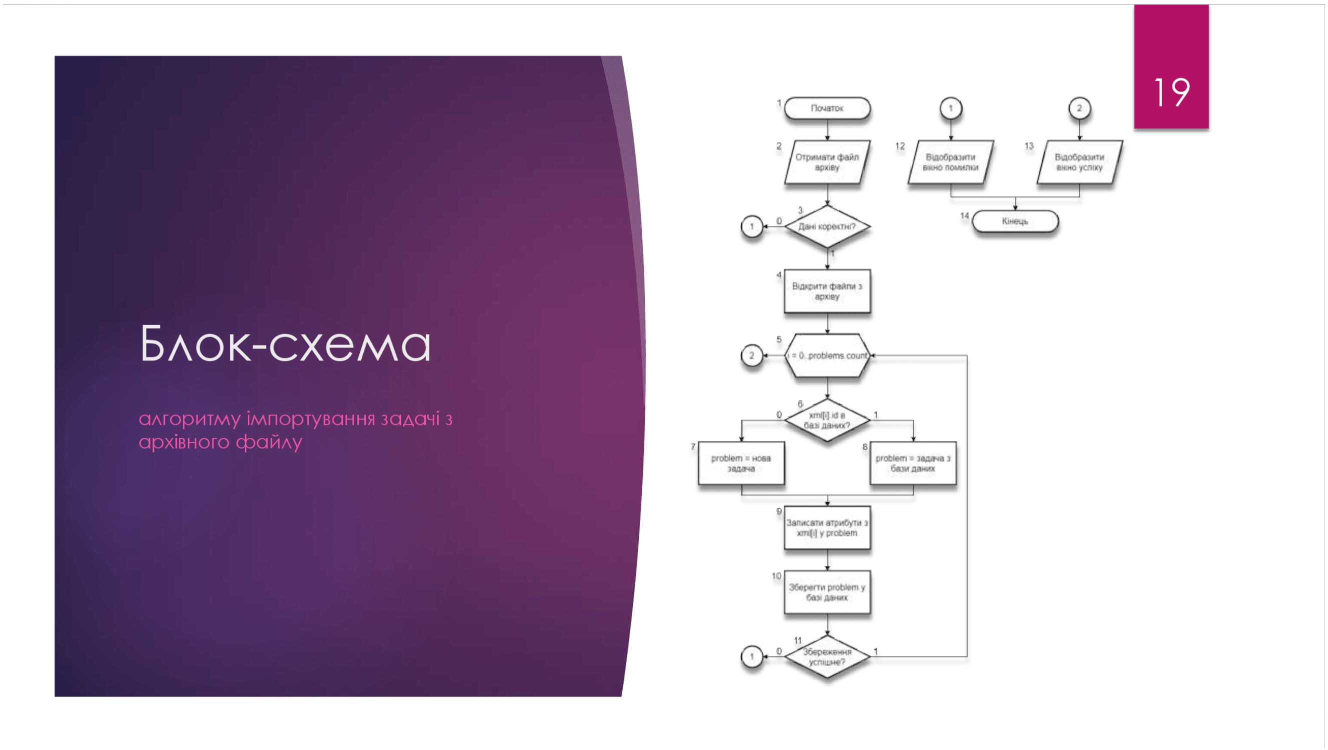


Рисунок Д.19 – Блок-схема алгоритму імпортування задачі з архівного файлу

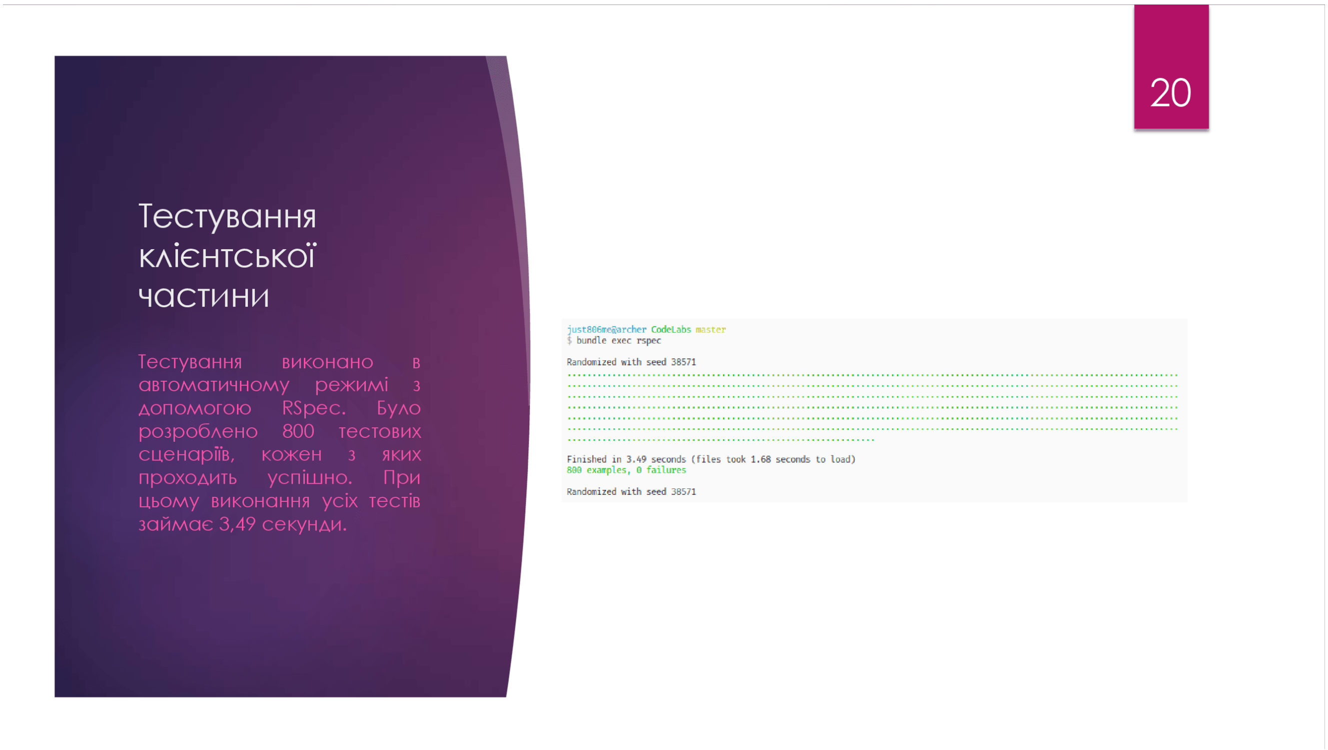


Рисунок Д.20 – Тестування клієнтської частини в автоматичному режимі

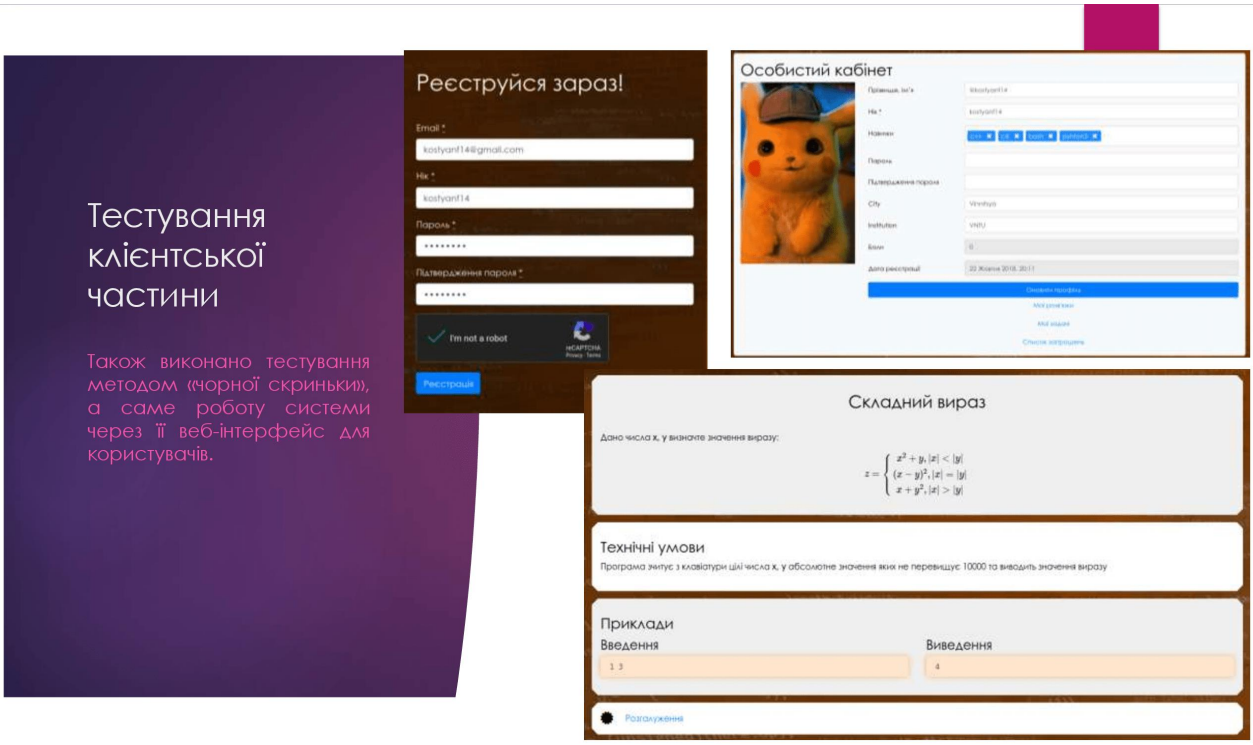


Рисунок Д.21 – Тестування клієнтської частини методом «чорної скриньки»: реєстрація

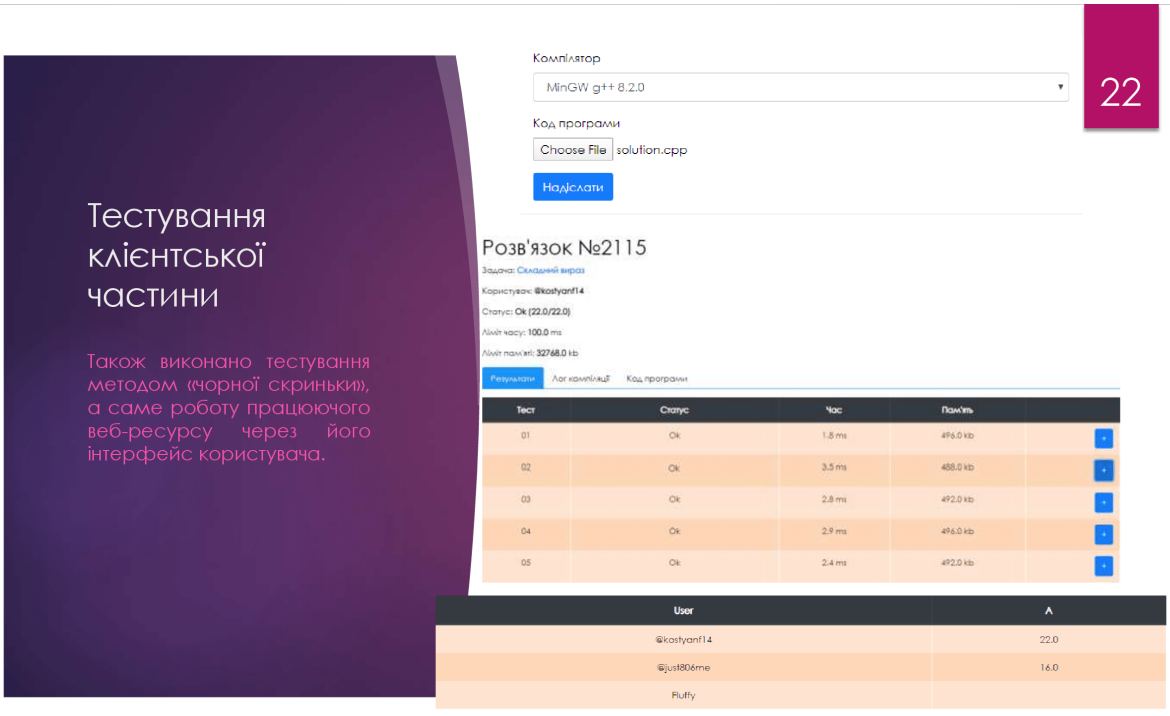


Рисунок Д.22 – Тестування клієнтської частини методом «чорної скриньки»: аналіз розв’язків

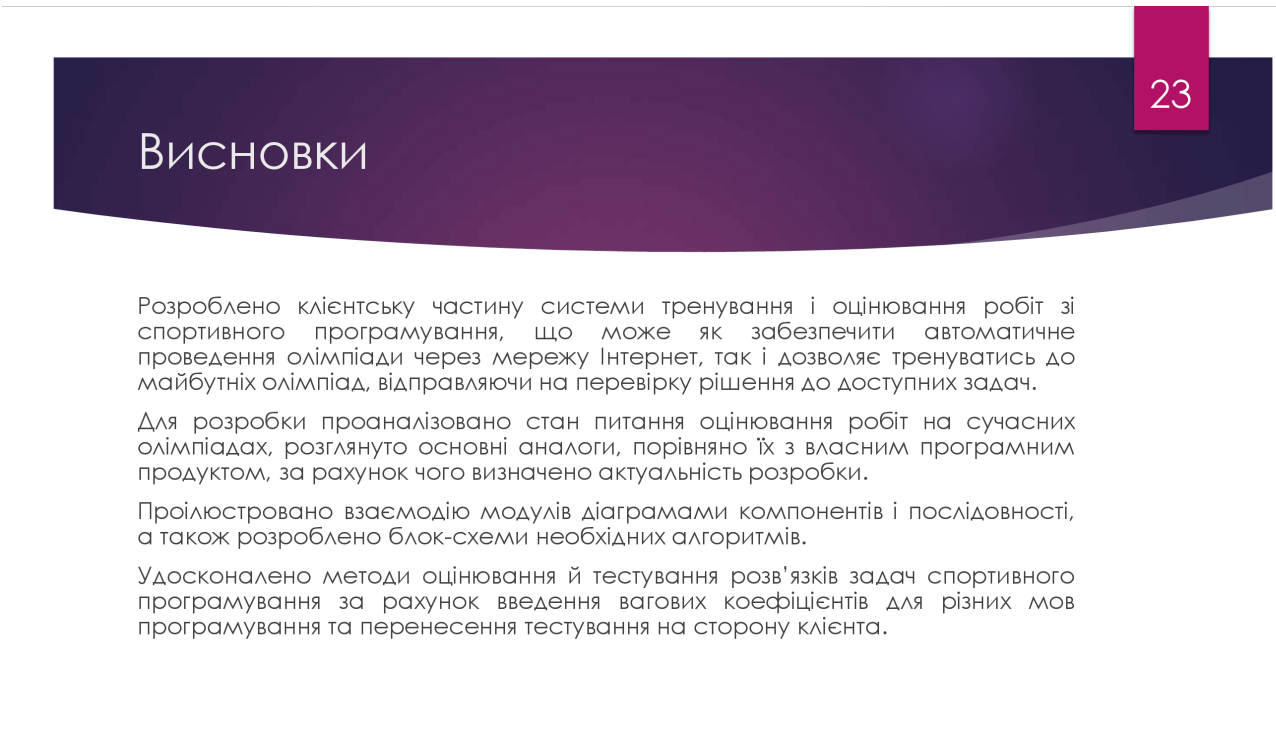


Рисунок Д.23 – Висновки

Рисунок Д.24 – Апробація

25

Публікації

- Войтко В.В., Бевз С. В., Бурбело С. М., Кузнєцов Л. Г., Костюк К. А. Розробка веб-ресурсу для розміщення та автоматизованої перевірки олімпіадних. – XLIX Науково-технічна конференція факультету інформаційних технологій та комп'ютерної інженерії, 2020. URL: https://conferences.vntu.edu.ua/index.php/all-fitki/all-fitki-2020/paper/view/9620/8008 (дата звернення 10.09.2021)
- Бурбело С. М., Костюк К. А., Кузнєцов Л. Г. Особливості використання процесорних тактів при оцінюванні часу роботи програм. – Конференція «Молодь у світі сучасних технологій», 2020. URL: http://conference.ho.ua/index.php (дата звернення 10.09.2021)
- Войтко В. В., Коваленко О. О., Бевз С. В., Бурбело С. М., Кузнєцов Л. Г., Костюк К. А. Застосування wasm у системі тренування і оцінювання робіт зі спортивного програмування. – Всеукраїнська науково-практична Інтернет-конференція «Електронні інформаційні ресурси: створення, використання, доступ», 2021. URL: http://konferencia.voipopp.vn.ua/ (дата звернення 30.11.2021).

Рисунок Д.25 – Публікації

1

Впровадження

Результати дослідження використовуються у Комунальному закладі «Вінницький технічний ліцей» Вінницької міської ради, що підтверджується відповідним актом впровадження.

Довідка

про впровадження наукових результатів магістерської кваліфікаційної роботи в практику діяльності підприємства.

В комплексній магістерській кваліфікаційній роботі на тему «Розробка методу і програмних засобів системи тренування і оцінювання робіт зі спортивного програмування» студентів Костюка К. А. та Кузнєцова Л. Г. запропоновано ряд заходів щодо вдосконалення якості систем дистанційного проведення олімпіадних змагань і тренування до них в режимі реального часу.

Розроблені в цій роботі програмні засоби мають наступну практичну цінність:

1) проведення першого етапу шкільної олімпіади зі спортивного програмування;
2) скорочення участі членів оргкомітету в процесі прийомки та шифрування робіт учасників до мінімуму за рахунок автоматичної перевірки робіт;
3) тренування учнів до наступних етапів олімпіади;
4) швидке оцінювання засвоєного матеріалу з використанням задач за обраною темою під час уроків інформатики;
5) додання власних задач для розширення банку олімпіадних знань;
6) відпрацювання навичок розв'язку задач зі зростанням складності;
7) можливість організовувати навчання у вигляді щотижневих змагань.

Застосовані в магістерській кваліфікаційній роботі методи і розроблені програмні засоби були впроваджені в практику діяльності підприємства Комунальний заклад «Вінницький технічний ліцей» Вінницької міської ради впродовж 2021-2022 навчального року.

Директор ліцею Козяр О. М.

Рисунок Д.26 – Впровадження