Вінницький національний технічний університет
Факультет інтелектуальних інформаційних технологій та автоматизації
Кафедра комп'ютерних систем управління

МАГІСТЕРСЬКА КВАЛІФІКАЦІЙНА РОБОТА
на тему:

Розробка автоматизованиї системи управління відвідуваністю для приватного закладу шкільної освіти

Виконав: студент 2 курсу, групи 2АКІТ-22м
спеціальності 151 – Автоматизація та комп'ютерно-інтегровані технології

Володимир МАЗУР
Ім'я ПРІЗВИЩЕ

Керівник: д.т.н., доцент, т.в.о. зав. кафедри КСУ
степінь, звання, посада

Марія ЮХИМЧУК
Ім'я ПРІЗВИЩЕ

«01» 12 2023 р.

Опонент: к.т.н., доцент, доцент кафедри АІІТ
степінь, звання, посада

Марія БАРАБАН
Ім'я ПРІЗВИЩЕ

«05» 12 2023 р.

Допущено до захисту
Т.в.о. зав. кафедри КСУ
Марія ЮХИМЧУК
«07» 12 2023

Вінниця ВНТУ – 2023 рік

Вінницький національний технічний університет

Факультет інтелектуальних інформаційних технологій та автоматизації

Кафедра комп'ютерних систем управління

Рівень вищої освіти другий (магістерський)

Галузь знань – 15 – Автоматизація та приладобудування

Спеціальність – 151 – Автоматизація та комп'ютерно-інтегровані технології

Освітньо - професійна програма – Інтелектуальні комп'ютерні системи

ЗАТВЕРДЖУЮ
Т.в.о.Зав. кафедри КСУ

Марія ЮХИМЧУК

"09" жовтня 2023 року

ЗАВДАННЯ
НА МАГІСТЕРСЬКУ КВАЛІФІКАЦІЙНУ РОБОТУ
студенту Мазуру Володимиру Юрійовичу
(прізвище, ім'я, по батькові)

1. Тема роботи. Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти
керівник роботи Юхимчук Марія Сергіївна
затверджені наказом ВНТУ від "18" вересня 2023 року №247
2. Термін подання студентом роботи "1 " грудня 2023 року
3. Вихідні дані до роботи: підвищення ефективності контролю учнів, fullstack-застосунок, використання сучасних веб-технологій, розробка інтуітивного інтерфейсу, простота у використанні, API, тестування.

4. Зміст текстової частини: вступ, аналіз проблеми, аналітичний огляд переваг та недоліків аналогів, функціональний огляд, розробка та детальний опис

5. Перелік ілюстративного матеріалу (з точним зазначенням обов'язкових креслень) вступ, актуальність, мета дослідження, об'єкт дослідження, предмет дослідження, новизна, бази даних, функціональність cms, послідовність під різними ролями, структура інтерфейсу, зовнішній вигляд розробленого додатку.

1. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання	виконання прийняв
4	Бурєннікова Н. В., д.е.н., професор кафедри ЕПВМ.		

2. Дата видачі завдання "09" жовтня 2023 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва та зміст етапу	Термін виконання		Примітка
		початок	закінчення	
1	Аналітичний огляд проблеми,	11.10.2023	20.10.2023	
2	Огляд теоретичного та методичного аспекту дослідження	21.10.2023	01.11.2023	
3	Розробка та тестування додатку	02.11.2023	21.11.2023	
4	Розрахунок економічної частини	22.11.2023	27.11.2023	
5	Оформлення матеріалів до захисту	28.11.2023	30.11.2023	

Студент Володимир МАЗУР
(підпис) (*Ім'я ПРІЗВИЩЕ*)

Керівник роботи Марія ЮХИМЧУК
(підпис) (*Ім'я ПРІЗВИЩЕ*)

АНОТАЦІЯ

УДК 37.018

Мазур В. Ю Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти. Магістерська кваліфікаційна робота зі спеціальності 151 – Автоматизація та комп’ютерно-інтегровані технології, освітня програма – Інтелектуальні комп’ютерні системи. Вінниця: ВНТУ, 2023. 114с.

На укр.мові. Бібліогр.: 43 назв; рис.: 20; табл. 13.

У магістерській кваліфікаційній роботі розроблено автоматизовану систему управління відвідуваністю для приватного закладу шкільної освіти. У оглядово-аналітичній частині роботи розглянуто особливості та важливість розробки системи, обрано сучасний стек технологій та обгрунтовано вибір кожної технології. У теоретично-методичній частині розглянуто функціональність сучасного веб-сайту, та олгянуто систему керування та навігації на веб-сайті. У практичній частині розроблено програмне забезпечення із використанням фреймворку NextJS, та базу даних за допомогою cms Strapi. У економічній частині проаналізований технічний рівень і розрахована собівартість та витрати на реалізацію проекту.

Ілюстративна частина складається з 13 плакатів із результатами роботи.

Ключові слова: автоматизована система, управління, стек, CMS, NextJS, API, додаток, веб-сайт.

ANNOTATION

UDC 37.018

Mazur V. Y Development of an automated attendance management system for a private school education institution. Master's thesis on specialty 151 - Automation and computer-integrated technologies, educational program - Intelligent computer systems. Vinnytsia: VNTU, 2023. 114p.

In Ukrainian speech Bibliography: 43 titles; Fig.: 20; table 13.

In the master's qualification work, an automated attendance management system for a private school education institution was developed. In the review and analytical part of the work, the features and importance of system development were considered, a modern stack of technologies was chosen and the choice of each technology was justified. In the theoretical and methodological part, the functionality of the modern website is considered, and the management and navigation system on the website is explained. In the practical part, the software was developed using the NextJS framework, and the database was developed using the Strapi cms. In the economic part, the technical level is analyzed and the cost price and costs for project implementation are calculated.

The illustrative part consists of 13 posters with the results of the work.

Keywords: automated system, management, stack, CMS, NextJS, API, application, website.

ЗМІСТ

ВСТУП

Актуальність. Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти має актуальність для України в зв'язку із покращенням ефективності та якісної організації навчального процесу. Ця ініціатива дозволяє оптимізувати робочий процес школи та впроваджувати сучасні підходи до управління освітнім середовищем.

Автоматизована система сприяє точному фіксуванню відвідувань учнів, що надає можливість вчителям та адміністрації швидко та ефективно контролювати присутність. Зокрема, це полегшує вирішення питань звітності, а також надає інструменти для аналізу даних, спрямованих на поліпшення якості освіти та прийняття обґрунтованих управлінських рішень.

Крім того, така система відповідає сучасним вимогам освітнього процесу, сприяючи використанню інноваційних технологій у сфері освіти. Це може позитивно позначитися на іміджі закладу та забезпечити йому конкурентоспроможність у сучасному освітньому середовищі України.

Мета дослідження. Мета розробки автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти полягає в створенні інноваційного інструменту, який сприятиме поліпшенню організації навчального процесу та забезпеченню ефективного контролю за присутністю учнів. Ця система має на меті забезпечити точність обліку відвідувань, дозволяючи швидко та ефективно відстежувати та аналізувати дані.

Об’єктом дослідження є система управління відвідуваністю для приватного закладу шкільної освіти

Предметом дослідження є ефективність використання системи управління відвідуваністю для підвищення якості освіти в приватному закладі

Новизна:

- Інтуїтивний та зручний інтерфейс;
- Оптимізація для мобільних пристроїв;
- Можливість легкого внесення та оновлення даних.

1 АНАЛІЗ СИСТЕМИ КОНТРОЛЮ ВІДВІДУВАНІСТЮ В ШКІЛЬНИХ ЗАКЛАДАХ ТА ОГЛЯД АНАЛОГІВ

1.1 Аналіз проблеми відвідуваності в шкільних закладах

Проблема відвідуваності у школах є серйозним викликом для освітньої системи. Часті відсутності учнів можуть призвести до відставання у навчанні та соціальної ізоляції. Не тільки академічні досягнення учнів страждають через низьку відвідуваність, але й їхні можливості розвитку та майбутнє кар'єрного зростання можуть бути відчутно обмежені.

Ця проблема особливо актуальна для сімей з низьким соціально-економічним статусом, де доступ до якісної освіти може бути обмеженим. Учні можуть відсутнювати через хвороби, неправильний розпорядок дня чи низький інтерес до навчання. Соціальний тиск та негативні впливи оточуючого середовища також можуть призводити до відмови від навчання. Учні можуть втратити інтерес до навчання через нудьгу, нецікавий матеріал або неправильні методи викладання. Крім того, сімейні обставини, економічні труднощі та психологічні проблеми можуть також бути чинниками, що призводять до регулярних відсутностей.

Соціальні фактори також грають важливу роль. Діти, які виростають у важких умовах, можуть відчувати неприв'язаність до школи та відчуженість від оточуючого середовища. Негативні взаємини з однокласниками чи вчителями можуть призводити до відсутності бажання відвідувати уроки.

Проблема відвідуваності може призвести до значного відставання в навчанні. Учні, які часто пропускають заняття, можуть пропустити важливий матеріал та навички, що може ускладнити їх подальше навчання. Це може вплинути на їхні можливості отримати вищу освіту та знайти стабільну роботу у майбутньому.

Ця проблема вимагає комплексного підходу для її вирішення. Школи повинні впроваджувати системи моніторингу та підтримки для відстеження

відвідуваності учнів і надання додаткової підтримки тим, хто стикається із цією проблемою. Також важливо залучати батьків до процесу, роз'яснювати їм важливість регулярного навчання та сприяти створенню сприятливого навчального середовища вдома[1].

Забезпечення стабільного та регулярного відвідування учнями школи є ключовим для їх успіху в навчанні та майбутньому житті. Боротьба з проблемою відвідуваності вимагає комплексного підходу. Навчальні заклади повинні вдосконалювати методи викладання, робити навчання цікавішим та стимулюючим для учнів. Також важливо залучати батьків до освітнього процесу, сприяти збереженню позитивного та підтримуючого середовища у школі, де кожен учень почувається важливим та підтриманим. Шляхом спільних зусиль вчителів, батьків та учнів можна подолати цю складну проблему та забезпечити всім дітям якісну освіту та можливість розвитку у майбутньому.

1.2 Обґрунтування важливості розробки автоматизованої системи управління

Автоматизована система управління[2] відвідуваністю для приватних шкіл є надзвичайно важливою і корисною з точки зору підвищення ефективності та оптимізації багатьох аспектів шкільної адміністрації. Враховуючи специфічні потреби приватних закладів, обґрунтування важливості такої системи виглядає наступним чином:

1. Автоматизація Обліку та Звітності:

Система дозволяє автоматично фіксувати відвідуваність учнів, учителів та іншого персоналу, спрощуючи процес обліку присутності та відсутності. Це значно зменшує ризик людських помилок та покращує точність звітності.

2. Моніторинг Точності та Керування Відвідуваністю:

Забезпечення точного моніторингу відвідуваності учнів та персоналу дозволяє оперативно реагувати на відсутність, сприяючи покращенню дисципліни та організації робочого процесу школи.

3. Забезпечення Безпеки та Контролю:

Автоматизована система забезпечує високий рівень безпеки шкільного простору, зокрема відслідковування присутності осіб в приміщеннях. Це може бути важливим для вирішення питань безпеки учнів.

4. Спрощення Взаємодії з Батьками:

Система дозволяє батькам в режимі реального часу відстежувати відвідуваність своїх дітей, отримувати повідомлення про відсутність та отримувати доступ до звітності. Це сприяє покращенню комунікації між школою та батьками.

5. Ефективне Управління Ресурсами:

Автоматизована система допомагає оптимізувати використання шкільних ресурсів, таких як класні кімнати та персонал, шляхом аналізу відвідуваності та розподілу обов'язків.

6. Адаптація до Сучасних Тенденцій в Освіті:

Впровадження автоматизованої системи відповідає сучасним тенденціям у розвитку освітніх технологій та підвищує конкурентоспроможність приватного закладу. Приватні школи сьогодні стикаються з суттєвими викликами в контексті ведення обліку відвідуваності. Одним із ключових аспектів, який потребує уваги та може бути вдосконалений через впровадження автоматизованої системи, є точність та швидкість обліку.

Ручний облік відвідуваності може призводити до неточностей та затримок у процесі. У сучасному освітньому середовищі, де швидкість та точність важливі для ефективного управління, це стає серйозним викликом для приватних шкіл. Підвищення об'єктивності та ефективності у веденні обліку відвідуваності стає стратегічно важливим завданням для оптимізації шкільних процесів та забезпечення якісної організації навчального середовища.

Ці аргументи обґрунтовують важливість та необхідність розробки автоматизованої системи управління відвідуваністю для приватних шкіл у контексті покращення робочих процесів.

1.3 Аналіз процесу навчання з використанням інтернету

Сучасні технології грають важливу роль у вирішенні проблеми низької відвідуваності у школах. Електронні системи моніторингу та сповіщення використовуються для точного відстеження відвідуваності учнів. Ці системи включають в себе біометричні сканери, картки доступу та мобільні додатки, що дозволяють автоматизувати процес реєстрації та повідомляти батьків про відсутність їхніх дітей.

Онлайн-платформи та навчальні додатки стають все більш популярними у навчальних закладах. Вони надають можливість дистанційного навчання та дозволяють учням звертатися до навчальних ресурсів в будь-який зручний для них час. Це зменшує бар'єри до доступу до освіти та сприяє збільшенню мотивації для навчання.

Крім того, використання інтерактивних методів навчання на основі комп'ютерних програм та віртуальної реальності може робити навчання цікавішим для учнів. Ігрові технології в навчанні можуть стимулювати учнів до активної участі та сприяти збільшенню зацікавленості у навчанні.

Технології Інтернету речей (IoT) можуть використовуватися для моніторингу присутності учнів у різних частинах шкільної будівлі та автоматизації систем сповіщення батьків чи вчителів у разі відсутності учнів без поважної причини.

Ці інноваційні підходи, які базуються на сучасних технологіях, допомагають забезпечити ефективну та систематичну відвідуваність учнями школи, створюючи сприятливі умови для навчання та розвитку.

1.4 Огляд аналогічних систем

В сучасному світі освіти, потреба в ефективних засобах управління навчанням стає все більш актуальною. У цьому розділі будуть розглянуті сучасні системи управління навчанням, серед таких інструментів виділяються системи

управління навчанням (LMS), які допомагають вчителям та адміністраторам створювати, впроваджувати та відстежувати навчальні програми, вони забезпечують широкий спектр можливостей для організації навчального процесу та взаємодії між вчителями та учнями.

LMS – це learning management system, являє собою веб-додаток, що дозволяє адмініструвати навчальний курс будь-якого напрямку в процесі дистанційного навчання[3]. Такі системи дуже широко поширилися останнім часом у зв’язку з пандемією і переходом багатьох навчальних закладів на віддалений або частково віддалений формат діяльності. Є обґрунтовані припущення, що дистанційна освіта буде широко використовуватися і надалі.

Наявність зручної системи управління навчанням дозволяє вирішувати цілий ряд задач, які раніше вимагали великої кількості розрізнених інструментів, в тому числі і паперових записів. Перш за все, LMS дозволяє налагодити процес навчання. Викладач може в рамках системи:

- призначати завдання;
- посилатися на курси і окремі навчальні матеріали;
- контролювати витрачений на вивчення і виконання завдань та тестів час;
- оцінювати досягнуті результати;
- скоротити час на донесення інформації.

Такі системи успішно застосовуються і в корпоративному навчанні, адже вони дозволяють донести до нового співробітника всі стандарти і весь комплекс професійних знань за гранично короткий термін і в зручному форматі. І звичайно, LMS буде незамінна для дистанційного навчання. Розглянемо детальніше найсучасніші і найпопулярніші з них, також дізнаємось про їх основні характеристики та переваги.

Перша з них це Moodle, що означає Modular Object-Oriented Dynamic Learning Environment (Модульне Об'єктно-Орієнтоване Динамічне Навчальне Середовище), представляє собою систему управління навчанням, також відому як LMS (Learning Management System), CMS (Course Management System), VLE

(Virtual Learning Environment) або просто навчальною платформою[4]. Moodle надає вчителям, учням і адміністраторам високорозвинений набір інструментів для комп'ютеризованого навчання, включаючи дистанційне навчання. Використовується він в різних контекстах, таких як освіта школярів і студентів, підвищення кваліфікації, бізнес-навчання та навчання в комп'ютерних класах або вдома.

Moodle є вільною та відкритою (Open Source) системою, що не тільки є безкоштовною, але і не потребує оплати за будь-яке програмне забезпечення. Це дозволяє кожному навчальному закладу використовувати абсолютно ліцензійну систему, не витрачаючи коштів на програмне забезпечення, а також адаптувати її код під свої потреби.

Moodle є однією з найпоширеніших та досконалих систем свого роду в Україні та у всьому світі. На сьогоднішній день кількість користувачів Moodle сягає приблизно 400 мільйонів по всьому світу, і ця платформа продовжує швидко розвиватися, обігаючи своїх конкурентів. На 2018 рік сталася значуща подія: за статистикою, використання Moodle перевищило використання всіх інших платформ разом узятими.

Moodle надає можливість використовувати розширений набір інструментів для взаємодії вчителя, учнів та адміністрації навчального закладу. Забезпечує можливість подавати навчальний матеріал у різних форматах, таких як текст, презентації, відео та веб-сторінки. Курс може бути структурований як сукупність веб-сторінок з можливістю виконання тестових завдань на проміжних етапах. Також, надає можливість проведення тестувань та опитувань учнів, використовуючи питання закритого (множинний вибір та порівняння) та відкритого типів. Учні можуть виконувати завдання та передавати відповідні файли. Розглянемо його переваги та недоліки:

Переваги:

1. Гнучкість та розширюваність:

   Moodle має модульну архітектуру, що дозволяє користувачам вибирати та впроваджувати лише ті функції, які вони потребують.

Можливість розширення функціоналу за допомогою плагінів та додаткових модулів, які розробляються спільнотою;

2. Спільнота та підтримка. Багата та допоміжна спільнота користувачів та розробників, що розвиває та підтримує Moodle. Наявність форумів та документації, що допомагають вирішувати проблеми та отримувати поради;
3. Безкоштовність та відкритий код. Використання Moodle є абсолютно безкоштовним, що робить його доступним для різних освітніх установ. Можливість змінювати та адаптувати код під специфічні потреби користувача;
4. Гнучкі інструменти оцінювання. Можливість виставляти оцінки за різноманітні завдання, тести та домашні роботи. Зручна система ведення електронного журналу з деталізованими записами оцінок;
5. Дистанційне навчання. Moodle ідеально підходить для організації дистанційного навчання, надаючи учням та вчителям доступ до матеріалів з будь-якого місця.

Недоліки:

1. Складність для новачків. Для новачків може здатися складним в освоєнні через велику кількість функцій та налаштувань. Використання у повному обсязі вимагає часу та навчання для розуміння всіх можливостей;
2. Естетика та дизайн. Зовнішній вигляд та дизайн може виглядати менш сучасно, порівняно з конкурентами. Деякі користувачі вказують на неінтуїтивність інтерфейсу;
3. Залежність від адміністратора. Підтримка та налаштування Moodle може вимагати наявності кваліфікованого адміністратора. Може бути важко впроваджувати та обслуговувати для менших організацій або тих, у яких обмежені технічні ресурси;

Наступна Google Classroom - це онлайн платформа від Google, яка дозволяє студентам та викладачам проводити віртуальні зустрічі для дистанційного

навчання[5]. Google Classroom надається безкоштовно для всіх користувачів, включаючи викладачів та студентів. З неї можна користуватися на ПК (через веб-версію) або мобільних телефонах, завдяки додаткам для Android та iPhone (iOS). Платформа також підтримує 38 мов, включаючи українську.

Google Classroom — це онлайн-клас, який сприяє вчителям у керуванні навчанням та створенні інтерактивних занять, що допомагають студентам розширювати знання за допомогою доступних інтернет-інструментів. Сервіс також дозволяє створювати різні класи, розподіляти завдання та обмінюватися нотатками та зворотнім зв'язком. Google Classroom є безкоштовним для державних і приватних шкіл та інтегрується з G Suite for Education.

Окрім проведення уроків, викладачі можуть надсилати повідомлення про заходи та завдання учням через комп'ютер чи мобільний телефон. Під час створення класу вчителі можуть додавати теми викладання, ставити питання учням та встановлювати бали за кожне завдання та визначати час для його виконання.

Матеріалими класу можна ділитися у форматі PDF, фотографій, відео та посилань на веб-сайти. Кожен студент має доступ до змісту безпосередньо зі свого пристрою. Google Classroom також інтегрується з кількома освітніми програмами. Наприклад, з Classcraft викладачі можуть імпортувати дані учнів та перетворювати діяльність в ігрові місії. У Quizizz можна ділитися іграми та перетворювати їх на завдання. Tynker надає безкоштовні курси та ресурси для навчання інформатики.

Створюючи різні аудиторії для кожного предмета, вчителі можуть індивідуально контролювати поведінку кожного учня та рівень викладання у кожному класі, а також змінювати зміст відповідно до рівня складності. Google Classroom також зберігає оцінки учнів та фіксує дати відправки кожної діяльності, допомагаючи вчителям стежити за успішністю кожного учня.

Переваги:

1. Простий та легко зрозумілий інтерфейс робить навігацію зручною для учнів та вчителів;

2. Додатки для Android та iOS надають можливість користуватися Classroom на будь-якому пристрої, забезпечуючи доступність де завгодно;
3. Дозволяє вчителям ефективно виставляти бали та стежити за прогресом учнів;
4. Захист конфіденційності інформації учнів та вчителів відповідно до стандартів безпеки;
5. Можливість ведення обговорень та взаємодії в реальному часі для активного навчання;
6. Підтримка різноманітних форматів матеріалів, включаючи PDF, фотографії, відео та посилання.

Недоліки:

1. Порівняно з іншими платформами, вона може мати менше функціональних можливостей;
2. Робота платформи пов'язана з доступністю Інтернет-з'єднання;
3. Порівняно з іншими платформами для освіти, може менше підходити для ігрового використання;
4. Деякі користувачі, особливо початківці, можуть потребувати часу для повного освоєння функцій та налаштувань;
5. Має деякі обмеження у варіативності налаштувань для індивідуального користування;
6. Для повноцінного використання Classroom може вимагати користування інши

Наступним є Canvas - це є електронним навчанням з відкритим вихідним кодом. Canvas LMS – це абсолютно безкоштовна система управління навчанням (LMS) для шкіл, університетів та навчальних центрів[6]. Він гнучкий, надійний, налаштований та ідеальний LMS для шкіл. Це також корисний LMS для бізнесу. Canvas LMS, Canvas Studio та Canvas Catalog є трьома компонентами програмного забезпечення для освіти Canvas. Canvas LMS організовує роботу та оптимізує

процеси навчання. Canvas Studio покращує навчання відео, роблячи його більш інтерактивним та цікавим. Реєстрація курсу спрощені та модернізовані за допомогою каталогу полотна. Користувачі можуть використовувати активну спільноту Canvas для обміну ідеями, ставити та відповідати на запитання, отримати доступ до навчальних посібників та взаємодіяти зі своїми однолітками.

Ви також можете покращити систему, інтегруючи її зі сторонніми послугами. Вчителі Canvas, Canvas Student та Parent Parent також можуть використовуватися для доступу до LMS Canvas з мобільного пристрою. Він розроблений у Ruby та JavaScript. Користувачі можуть знайти докладну документацію та вихідний код у GitHub. Ліцензія на систему навчання Canvas – загальна публічна ліцензія GNU Affero.

Створення курсів у системі Canvas зазвичай виконується викладачем або адміністратором освітнього закладу. Після входу в систему, викладач має можливість створити новий курс, вказавши необхідні деталі, такі як назва, опис та терміни проведення. Після цього викладач може додавати різноманітні матеріали для вивчення, такі як лекції, завдання, відео та інші ресурси. Основна інформація про курс включає також параметри доступності матеріалів, графік, методи оцінювання та інші аспекти. Після завершення налаштувань викладач має можливість опублікувати курс, роблячи його доступним для учнів. Розглянемо переваги і недоліки даної системи.

Переваги:

1. Canvas надає вчителям величезний функціонал для налаштування курсів відповідно до їхніх потреб, включаючи завдання, календар, та структуру курсу;
2. Підтримка віртуальних лекцій, відеоуроків, обговорень та інтерактивних завдань;
3. Canvas дозволяє вчителям встановлювати та відстежувати прогрес учнів через систему оцінок та аналітику;
4. Інтеграція чату, форумів та інших засобів комунікації дозволяє ефективно спілкуватися між учнями та вчителями;

5. Підтримка сторонніх додатків та сервісів дозволяє розширювати функціональність Canvas.

Недоліки:

1. Для новачків може бути важко орієнтуватися великою кількістю функцій та налаштувань.
2. У порівнянні з іншими LMS може бути менше інтегрованих ігрових можливостей.
3. Робота платформи пов'язана з наявністю Інтернет-з'єднання.
4. Для повного використання Canvas корисно мати певні технічні знання.
5. Деякі розширені функції або інтеграції можуть вимагати додаткової плати.

Отже, у нашому обговоренні були розглянуті три освітні платформи: Moodle, Google Classroom та Canvas LMS. Кожна з цих систем має свої унікальні риси та призначення, але всі вони спрямовані на поліпшення процесу навчання та забезпечення ефективного віддаленого навчання.

Moodle представляє собою відкриту та безкоштовну платформу з великою кількістю інструментів для комп'ютеризованого навчання, надаючи гнучкі можливості для налаштування процесу навчання. Велика активна спільнота та безкоштовний характер роблять її привабливою для різних типів освітніх закладів.

Google Classroom відзначається простотою використання та інтеграцією з іншими сервісами Google. Вона призначена для ефективного ведення класів, спільної роботи над завданнями та спрощення взаємодії між вчителями та учнями. Як частина Google Workspace for Education, вона підходить для шкіл та вищих навчальних закладів.

Canvas LMS є інноваційною та гнучкою освітньою платформою. Вона вражає широким функціоналом для налаштування курсів та акцентом на інтерактивність. Canvas забезпечує вчителів та студентів різноманітними

інструментами для активного навчання та ефективної співпраці.

Кожна система має свої переваги та недоліки, і вибір між ними може залежати від конкретних потреб, вимог та можливостей освітнього закладу. Загалом, важливо враховувати потреби користувачів, зручність використання та можливості для ефективного навчання при виборі платформи для освітнього процесу.

Вискновки до розділу

У ході цього розділу ми глибоко дослідили проблему відвідуваності в шкільних закладах, розглянули важливість впровадження автоматизованої системи управління, проаналізували процес навчання з використанням інтернет-технологій та оглянули аналогічні системи.

Виявлено, що проблема низької відвідуваності може бути успішно вирішена завдяки сучасним технологіям. Впровадження автоматизованої системи управління не лише поліпшить моніторинг відвідуваності, але й сприятиме ефективнішому управлінню шкільними ресурсами та взаємодії з батьками та учнями. Аналіз процесу навчання з використанням інтернет-технологій підкреслив важливість розвитку дистанційного навчання та доступу до освітніх ресурсів. Це не лише сприятатиме гнучкості навчання, але й зменшить бар'єри до освіти. Під час огляду аналогічних систем було виявлено, що існують ефективні веб-застосунки, які можуть слугувати вдосконаленням нашого підходу. Потрібно створити власну системи LMS у вигляді веб-застосунку з зручним, простим, інтуітивно зрозумілим інтерфейсом, щоб система була легка в управлінні для адміністратора, та зручною для ведення звітності вчителів, впровадження інноваційних елементів, та інтерактивні методи навчання, можуть сприяти більш ефективному вирішенню поставлених завдань.

Отже, застосування сучасних технологій у шкільному навчанні може значно покращити умови для освіти, забезпечуючи ефективний моніторинг, доступ до якісних навчальних ресурсів та створюючи стимулюючий інтерактивний навчальний процес.

2 ОГЛЯД ТЕОРЕТИЧНОГО ТА МЕТОДИЧНОГО АСПЕКТУ ДОСЛІДЖЕННЯ

2.1 Аналіз та вибір технологій для побудування інтерфейсів користувача

UI – User Interface, дослівно перекладається як «Інтерфейс користувача». Це певний процес візуалізації, який дає змогу реалізувати прототип сайту, програми або іншого веб-ресурсу[7]. Насамперед, UI включає активну роботу над графічною складовою інтерфейсу. Створюються анімації, ілюстрації, кнопки та інші елементи додатку, включаючи шрифти, кольори, форми.

Спочатку визначається досвід користувача і досліджується цільова аудиторія. Залежно від потреб відвідувачів визначається палітра кольорів, форми, а також структура розміщення об'єктів. Інтерфейс користувача має бути зрозумілим, і одне з головних завдань – визначити, чи зручний сайт візуально, чи легко потрапити по кнопках, чи текст читається добре. Потрібно приділити багато уваги розробці інтерфейсу користувача, щоб використання веб-додатку було легким та простим, врахувати всі аспекти UX (англ. User Experience, надалі UX), для того щоб користувачеві було не складно знайти потрібну для нього інформацію на сайті.

UX - дослівно означає «досвід користувача»[8]. У більш широкому сенсі це поняття про весь досвід, який отримує користувач при взаємодії з сайтом або додатком. UX-дизайн відповідає за функції, адаптивність продукту і те, які емоції він викликає у користувачів. Чим зрозуміліший інтерфейс, тим легше користувачеві отримати результат і зробити цільову дію.

Інтерфейс користувача складається з прочитаних браузером HTML-розмітки сайту, де розкладена вся потрібна інформація і елементи взаємодії по сторінці. Потім для сторінки та всіх її елементів читаються стилі, це каскадні таблиці стилів (Cascading Style Sheets – CSS), вся динамічна взаємодія з сайтом реалізується через мову програмування JavaScript.

Оберемо варіант самостійної розробки інтерфейсу користувача. Можна прописати всі умови відображення в браузері на «нативних» мовах – це HTML,

CSS та JavaScript, але щоб досягнути кращого результату за менший проміжок часу, потрібно обирати фреймворки та бібліотеки.

2.1.1 Аналіз і обгрунтування стеку для front-end частини

Для Front-end частини використовувались такі технології:

– Next Js 14.0;

– Tailwind CSS;

– Material UI/MUI.

Next.js - це фреймворк, заснований на React, який дозволяє створювати веб-додатки з покращеною продуктивністю та покращеним користувальницьким досвідом за допомогою додаткових функцій попереднього рендерингу, таких як повноцінний рендеринг на стороні сервера (SSR) (Див. рис 2.1 та 2.2)та статична генерація сторінок (SSG)[9]. Таким чином, програми Next.js використовують усі переваги бібліотеки React і просто додають додаткові функції:

Server Side Rendering. SSR дозволяє отримати доступ до всіх необхідних даних для побудови сторінки на сервері. Потім сторінка повністю відправляється назад у браузер і відразу відображається. SSR дозволяє веб-сторінкам завантажуватись за менший час і підвищує зручність роботи користувачів за рахунок підвищення швидкості відгуку[10].

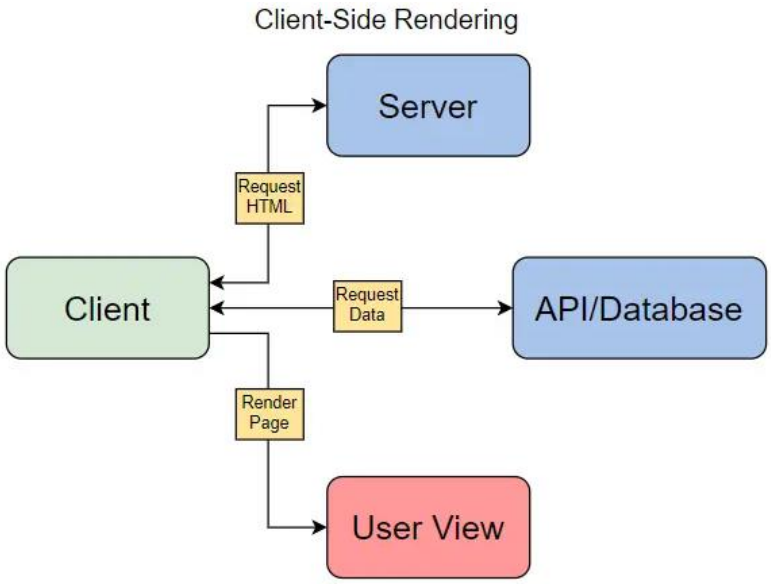


Рисунок 2.1 – Ілюстрація рендеринга на стороні клієнта

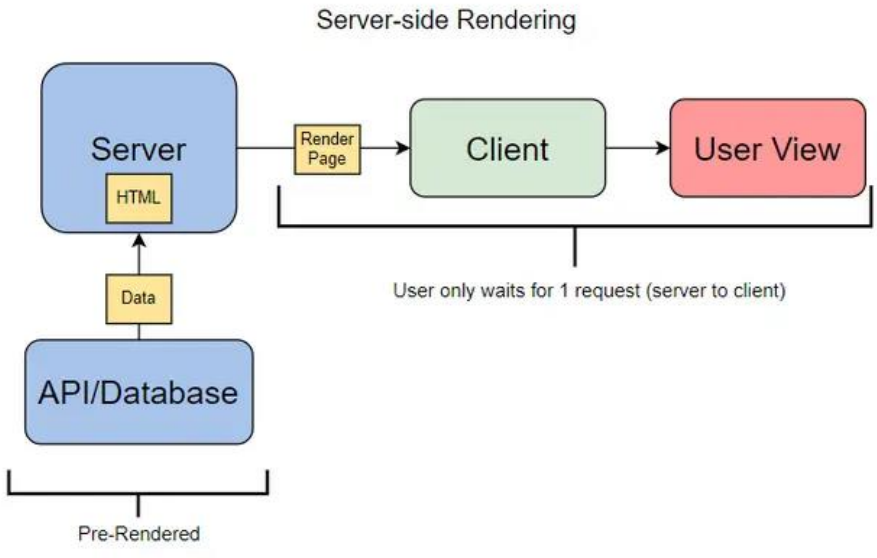


Рисунок 2.2 – Ілюстрація рендеринга на стороні сервера

SEO чи просто пошукова оптимізація. Використання SSR також дає вам перевагу у SEO, що допомагає вашому веб-додатку займати більш високі позиції на сторінках результатів пошукових систем. SSR підвищує рейтинг веб-сайтів для SEO, тому що вони завантажуються швидше та більше контенту сайту можна сканувати за допомогою трекерів SEO[11].

Next.js також дозволяє вам редагувати тег <head> сайту, який ви не можете зробити в React. Тег <head> є основною частиною метаданих веб-сторінки та сприяє підвищенню SEO-рейтингу веб-додатка.

Навіщо взагалі використовувати Next.js ?

Основна перевага Next.js – вбудована підтримка SSR для підвищення продуктивності та SEO. Рендеринг на стороні сервера працює шляхом зміни потоку запитів React, так що всі компоненти, крім клієнта, відправляють свою інформацію на сервер.

Маючи всю інформацію на сервері, клієнт може отримати попередню версію HTML-коду сторінки. Клієнт може надіслати один запит на сервер та отримати повноцінну HTML-сторінку замість того, щоб вимагати кожен компонент окремо з рендерингом на стороні клієнта.

Next.js - потужний фреймворк, який спрощує розробку веб-додатків на React та надає ряд корисних функцій для покращення продуктивності та робочого

процесу розробників. Документація React згадує Next.js серед «Рекомендованих наборів інструментів» і рекомендує його розробникам під час «створення серверного веб-сайту за допомогою Node.js. У той час як традиційні веб-додатки React можуть відтворювати свій вміст лише в браузері на стороні клієнта, Next.js розширює цю функціональність, додаючи можливість генерації веб-додатка на стороні сервера.

Авторські права та торгові марки для Next.js належать компанії Vercel, яка підтримує та очолює розробку фреймворку як програмного забезпечення з відкритим вихідним кодом, логотип Next.js показаний на рисунку 2.3.

Рисунок 2.3 – Логотип фреймворку Next JS

Плюси роботи з Next.js[12]:

- Програми Next.js завантажуються значно швидше, ніж програми React
- завдяки вбудованому рендерингу на стороні сервера;
- Підтримує функцію експорту статичних сайтів;
- Швидкий вхід для тих, хто вже працював з бібліотекою React.js;
- Автоматичне поділ коду для сторінок;
- Легко створювати внутрішні API-інтерфейси за допомогою вбудованих маршрутів API та створювати кінцеві точки API;
- Вбудована підтримка маршрутизації сторінок, CSS, JSX та TypeScript;
- Швидке додавання плагінів для налаштування Next.js відповідно до потреб вашої сторінки;
- Підтримує такі переваги, як інтуїтивно зрозуміле створення на основі компонентів, інтерфейсна система станів і висока популярність.

Мінуси Next.js:

– Єдиним реальним недоліком Next.js є те, що це самостійний фреймворк, тобто він має певний метод і набір інструментів, які він хоче, щоб ви використовували для створення своїх додатків.Однак можливості Next.js цілком підходять для більшості проектів.

Next.js відмінно підходить для розробки як статичних, так і динамічних веб-додатків. Використання його для статичних сторінок сприяє покращенню швидкості завантаження, а динамічні можливості фреймворку дозволяють створювати реактивні веб-додатки. Зручність управління маршрутизацією, автоматичне підняття зображень та можливість серверного рендерингу роблять Next.js потужним інструментом для розробки веб-додатків на основі React.

Далі розглянемо TailwindCSS – це поступово набираючий популярності CSS-фреймворк, що дозволяє вносити зміни в оформлення сайтів та додатків, не залишаючи HTML-розмітку (причому як у відповідних файлах, так і в компонентах типу React або Svelte) і не використовуючи тег <style>. Tailwind CSS дозволяє створювати інтерфейси користувача за допомогою набору готових класів. Він був розроблений для прискорення процесу розробки та полегшення підтримки коду[13]. Логотип фреймворку зображений на рисунку 2.4.

Рисунок 2.4 – Логотим фреймворку TailwindCSS

Ідея полягає в тому, щоб прописувати стилі безпосередньо в директиву class, а не під селекторами CSS-файлі. Це виглядає так:

Header

Основна перевага Tailwind CSS полягає в тому, що вона надає великий набір готових класів. Це значно скорочує час на створення інтерфейсу користувача, так як не потрібно писати щоразу нові стилі для кожного елемента.

Крім того, використання готових класів дозволяє легко і швидко змінювати зовнішній вигляд елементів, не торкаючись їхньої структури. Програмний код на TailwindCSS написаний на рисунку 2.5.

ample.html

```
<div class="bg-gray-50">
  <div class="max-w-screen-xl mx-auto py-12 px-4 sm:px-6 lg:py-16 lg:px-8
  lg:flex lg:items-center lg:justify-between">
    <h2 class="text-3xl leading-9 font-extrabold tracking-tight text-gray-900
    sm:text-4xl sm:leading-10">
      Ready to dive in?
      <br />
      <span class="text-indigo-600">Start your free trial today.</span>
    </h2>
    <div class="mt-8 flex lg:flex-shrink-0 lg:mt-0">
      <div class="inline-flex rounded-md shadow">
        <a href="#" class="inline-flex items-center justify-center px-5 py-3
        border border-transparent text-base leading-6 font-medium rounded-md
        text-white bg-indigo-600 hover:bg-indigo-500 focus:outline-none
        focus:shadow-outline transition duration-150 ease-in-out">
          Get started
        </a>
      </div>
      <div class="ml-3 inline-flex rounded-md shadow">
        <a href="#" class="inline-flex items-center justify-center px-5 py-3
        border border-transparent text-base leading-6 font-medium rounded-md
        text-indigo-600 bg-white hover:text-indigo-500 focus:outline-none
        focus:shadow-outline transition duration-150 ease-in-out">
          Learn more
```

Рисунок 2.5 – Приклад написаного коду на TailwindCSS

Ще одна перевага Tailwind CSS полягає в тому, що він дозволяє створювати адаптивні інтерфейси. Для цього у фреймворку є спеціальні класи, які дозволяють змінювати стилі в залежності від розміру екрана пристрою, на якому відображається сторінка, також він дозволяє налаштовувати та перевизначати готові класи, що дозволяє створювати унікальні стилі для кожного проекту.

Завдяки використанню Tailwind CSS, розробники мають можливість зосередитися на логіці та функціональності програми, замість того, щоб витрачати час на написання стилів. Ще однією перевагою Tailwind CSS є його модульність. Він складається з багатьох невеликих модулів, які можна

використовувати окремо або в комбінації один з одним. Це дає розробникам можливість вибирати ті модулі, які потрібні для конкретного проекту, і не навантажувати сторінку зайвим кодом. Крім того, Tailwind CSS надає безліч інструментів для спрощення розробки, таких як автодоповнення класів у редакторі коду, візуальні інструменти для налаштування колірної схеми та багато іншого.

У результаті, Tailwind CSS є дуже корисним інструментом для розробки інтерфейсів користувача, який дозволяє значно прискорити процес створення і підтримки коду, а також створювати унікальні та адаптивні інтерфейси без необхідності писати щоразу нові стилі.

Наступим розберемо Material UI/MUI - Material-UI (або MUI) - це популярна бібліотека для розробки інтерфейсу користувача (UI) для React-додатків, яка базується на дизайн-мові Google Material Design. Material Design - це система дизайну, розроблена Google, яка використовує концепції матеріалів та різних шарів для створення чітких та зрозумілих інтерфейсів. Material-UI був створений на основі концепцій та дизайну Google Material Design і спроектований як набір React-компонентів для легкого створення інтерфейсів користувача відповідно до цих принципів. Розробка Material-UI розпочалася в 2014 році, коли тема Material Design була вперше представлена Google. Спільнота React швидко прийняла ці ідеї, і виникла потреба у бібліотеці компонентів, яка б дозволяла легко впроваджувати цей дизайн в React-проекти[14].

Матеріали-UI були створені на основі відкритого підходу, і вони активно розвиваються завдяки спільноті та приймають участь в обговореннях та внесенні змін. З тих пір, як Material-UI був випущений відкритою бета-версією, він став дуже популярним серед розробників React.. Вона містить широкий спектр готових до використання компонентів, таких як кнопки, таблиці, меню, сповіщення та багато іншого. Material UI/MUI є відкритим джерелом і може бути повністю налаштований. Крім того, вона має набір інструментів для налаштування тем, які дозволяють легко налаштувати компоненти під свої потреби. Material UI/MUI є однією з найбільш популярних бібліотек компонентів React, яку використовують

найкращі команди продуктів у світі. Логотип бібліотеки зображений на рисунку 2.6.

Рисунок 2.6 – Логотип бібліотеки Material UI

Переваги Material UI[15]:

- Material Design - це чітко визначена мова дизайну, а Material UI/MUI надає готові компоненти, які відповідають цій мові дизайну. Це може допомогти забезпечити консистентний вигляд та відчуття вашого інтерфейсу користувача на різних пристроях та платформах;
- Material-UI надає готові до використання компоненти, але також дозволяє легко керувати їх стилями та розширювати функціональність, використовуючи власні компоненти;
- Надає широкий спектр готових до використання компонентів, які легко налаштовувати та використовувати. Багато з компонентів мають властивості, які дозволяють налаштувати їх вигляд та поведінку;
- Material UI/MUI має велику та активну спільноту розробників, які вносять свій вклад у проект. Це означає, що ви зазвичай можете отримати допомогу з будь-яких питань, з якими ви можете стикнутися. Спільнота також регулярно випускає нові функції та покращення;
- Завдяки Material-UI можна швидко і легко впроваджувати дизайн, який відповідає концепціям Material Design, що забезпечує професійний та сучасний вигляд додатку.

Недоліки Material UI:

- Material UI/MUI може додати значну кількість розміру до вашої програми, оскільки вона містить велику кількість готових компонентів. Це може вплинути на продуктивність вашої програми, особливо на пристроях з меншою продуктивністю;
- Хоча Material UI/MUI дозволяє настраювати деякі зі своїх компонентів, не завжди можливо досягти того самого вигляду та відчуття, яке ви хочете;
- Вибір та керування стилями може бути викликом для деяких розробників, особливо якщо вони хочуть повністю змінити вигляд компонентів;
- Для повноцінного використання Material-UI, користувачам слід бути знайомим із базовим React.

Приклад використання MUI:

```
import React from 'react';
import Button from '@mui/material/Button';
function MyButton() {
  return (
    <Button variant="contained" color="primary">
      Click me
    </Button>
  );
}
```

Тут Button - це компонент MUI, який автоматично додає стилі відповідно до Material Design.

Використання Material-UI сприяє швидкій розробці інтерфейсу з використанням популярної бібліотеки React, також сприяє зменшенню зусиль для створення стильного та сучасного дизайну, що робить його відмінним інструментом для розробників, які цінують ефективність та естетику у своєму процесі роботи.

2.1.2 Аналіз сучасних конкурентів до вибраного стеку front-end частини

На момент останньої оновленої інформації, Next.js був одним з найпопулярніших фреймворків для розробки веб-додатків на мові програмування JavaScript або TypeScript. Однак у світі веб-розробки існує багато конкурентів, які також пропонують рішення для створення сучасних та ефективних веб-додатків. Давайте розглянемо деякі з них:

1. React.js з Create React App (CRA):

– Опис: React є бібліотекою для інтерфейсів користувача, і Next.js використовує React як базову технологію. Create React App (CRA) є іншим інструментом швидкого старту реактивних додатків. React - це бібліотека для створення інтерфейсів користувача в мові JavaScript[16]. Заснована на компонентному підході, React дозволяє розбити інтерфейс на невеликі, незалежні компоненти, що полегшує розробку, тестування та обслуговування. Одним з ключових аспектів React є використання віртуального DOM, що сприяє оптимізації відображення змін. React також використовує JSX - це розширення синтаксису JavaScript, яке дозволяє писати HTML-подібний код безпосередньо в JavaScript файлі, що полегшує створення компонентів. Однією з суттєвих переваг React є його можливість працювати як на клієнтській стороні (client-side), так і на серверній стороні (server-side), що забезпечує високу гнучкість та широкі можливості для розробників.

– Відмінності від Next.js: CRA не надає статичних рендерингів та серверних рендерингів за замовчуванням, що може призводити до меншої продуктивності в окремих сценаріях.

2. Vue.js та Nuxt.js:

– Опис: Vue.js – це фреймворк для створення інтерфейсів

користувача на JavaScript. Він славиться своєю простотою використання та гнучкістю. Vue заснований на компонентній архітектурі, що дозволяє організувати код у вигляді відокремлених компонентів, полегшуючи розробку та підтримку[17]. Однією з ключових рис Vue є його реактивність - система, яка автоматично відстежує зміни даних і оновлює інтерфейс відповідно. Це дозволяє

створювати динамічні та відгукні інтерфейси без необхідності вручну втручатися у DOM. Vue.js також пропонує прогресивний підхід до використання, дозволяючи вам впроваджувати його поступово в проекти різного розміру. Ця гнучкість робить Vue популярним вибором серед розробників. а Nuxt.js – фреймворк, який базується на Vue.js та надає подібні можливості до Next.js[18].

– Відмінності від Next.js: Як і Next.js, Nuxt.js пропонує статичний та серверний рендеринг. Вибір між Next.js та Nuxt.js може залежати від ваших улюбленців та досвіду з використанням React або Vue.

3. Angular та Angular Universal:

– Angular - це потужний фреймворк для розробки веб-додатків, розроблений Google[19]. Однією з визначальних рис Angular є використання мови програмування TypeScript, що надає додаткові можливості та строго типізований підхід до коду. Основні характеристики Angular включають компонентну архітектуру, яка організована навколо компонентів, модульність та використання сервісів. Angular також володіє потужним системою двостороннього зв'язування даних, що дозволяє автоматично відслідковувати та оновлювати дані в інтерфейсі користувача. Завдяки своїм багатофункціональним можливостям та інструментам, Angular часто використовується для створення великих, масштабованих веб-додатків. Angular Universal - це модуль для рендерингу Angular додатків на стороні сервера[20].

– Відмінності від Next.js: Angular Universal та Next.js обидва підтримують серверний рендеринг, але мають відмінності в архітектурі та підходах до розробки.

4. Svelte та SvelteKit:

– Опис: Svelte - інноваційний фреймворк для створення веб-додатків, що відрізняється своєю унікальною концепцією компіляції[21]. Замість традиційного підходу, де код виконується на клієнтському браузері, Svelte використовує компіляцію коду в оптимізований JavaScript на етапі збірки. Суть Svelte полягає в тому, що весь код компонента видаляється під час компіляції, і генерується оптимізований JavaScript, який працює без великої кількості

фреймворк-коду на клієнтському браузері. Це дозволяє забезпечити ефективність та швидкодіючість веб-додатків, використовуючи простий і зрозумілий синтаксис. Основні переваги Svelte включають відсутність віртуального DOM, компактний код та простоту використання, що робить його привабливим вибором для розробників, які цінують продуктивність та простоту розробки.. SvelteKit - це фреймворк для створення веб-додатків на основі Svelte.

Відмінності від Next.js: Svelte та SvelteKit використовують іншу архітектуру, яка дозволяє зменшити обсяг коду та підвищити продуктивність. Однак, на момент останнього оновлення, екосистема Svelte була менш розвиненою порівняно з React або Vue.

Tailwind CSS є популярним інструментом для створення веб-інтерфейсів, проте на ринку є інші CSS-фреймворки та інструменти, які також пропонують зручний спосіб роботи зі стилями веб-додатків. Давайте розглянемо деяких конкурентів Tailwind CSS:

1. Bootstrap:

– Опис: Bootstrap є одним із найпопулярніших фреймворків для розробки веб-додатків. Його розробка була почата командою Twitter (Twitter Bootstrap) і вперше випущена у відкритий доступ в серпні 2011 року[22]. Основна мета фреймворку - полегшити процес веб-розробки, надаючи розробникам готові компоненти та інструменти для швидкого створення сучасних та адаптивних інтерфейсів. Він надає готові компоненти, сітку та стилі для швидкої побудови сучасних інтерфейсів. Bootstrap має обширний набір готових компонентів, таких як кнопки, форми, картки, що полегшує створення інтерфейсу. Фреймворк надає систему сітки, що робить сторінки адаптивними для різних розмірів екрану. Використання Flexbox і Grid спрощує організацію та розміщення елементів. Bootstrap забезпечує сумісність з більшістю сучасних браузерів. З недоліків Bootstrap великий розмір, включення всього фреймворку може призвести до великого обсягу коду, що може впливати на час завантаження. Сайти, побудовані на Bootstrap, можуть виглядати стандартно, оскільки багато проєктів використовують однакові стилі та компоненти. Важко досягти унікального

вигляду, оскільки призначена для швидкої розробки та спрощення, а не для індивідуальності.

– Відмінності від Tailwind CSS: У випадку Bootstrap, стилі визначаються класами, які вже є визначені у фреймворку, тоді як Tailwind CSS надає вам набір базових класів, які можна комбінувати для створення власних стилів.

2. Bulma:

– Опис: Bulma, фреймворк для веб-розробки, був створений Флоріаном Бігардом і випущений у 2016 році. Його створення було віддзеркаленням потреби в простоті та гнучкості для розробників, які хочуть побудувати інтерфейси користувача на основі стандартів CSS, без використання JavaScript[23]. Переваги Bulma: Bulma вражає своїм простим та зрозумілим синтаксисом, що спрощує використання та розгортання, особливо для розробників, які шукають простий інструмент для швидкої розробки. Bulma використовує потужну сіткову систему на основі Flexbox, що дозволяє легко контролювати розташування та вигляд компонентів на сторінці. Компоненти Bulma є модульними, що дозволяє вам вибирати та використовувати лише ті, які необхідні для вашого проєкту. Недоліки Bulma: Однією з недоліків Bulma є відсутність розширених JavaScript компонентів порівняно з іншими фреймворками, що може обмежувати можливості для розробників, які шукають великий функціонал на стороні клієнта. У порівнянні з деякими іншими фреймворками, Bulma може мати меншу спільноту та екосистему, що може вплинути на доступність розширень та підтримку. Не так широко використовується: У порівнянні з деякими іншими фреймворками, такими як Bootstrap або Tailwind CSS, Bulm a може бути менш популярним і менше використовуватися в галузі веб-розробки.

– Відмінності від Tailwind CSS: Bulma пропонує певний набір класів для стилізації елементів, але вони менш деталізовані та гнучкі порівняно з класами Tailwind CSS.

3. Materialize CSS:

– Опис: Materialize CSS - це інший фреймворк для веб-розробки, який

базується на дизайн-мові Google Material Design. Створений в 2014 році, Materialize CSS намагається принести естетику та концепції Material Design на веб-сайти та додатки[24]. Однією з основних переваг є відповідність концепціям Material Design, яка надає сучасний та чистий вигляд. Materialize CSS містить готові до використання компоненти, такі як кнопки, картки, форми, що полегшує розробку інтерфейсу. Фреймворк пропонує різноманітні анімації та ефекти, які можна легко використовувати для покращення взаємодії користувача. З недоліків виділяють великий обсягу коду, що може впливати на продуктивність та швидкість завантаження сторінки. Іноді важко здійснити повну кастомізацію стилів, оскільки Materialize CSS вже визначає частину зовнішнього вигляду. Деякі компоненти Materialize вимагають включення JavaScript для повного функціоналу, що може впливати на взаємодію з додатками та завданнями на сторінці. Загалом Materialize CSS може бути ефективним вибором для проєктів, які вимагають використання Material Design та забезпечують легку інтеграцію готових компонентів. Однак розробники повинні враховувати недоліки, такі як обсяг коду та обмежена кастомізація, при виборі цього фреймворку.

– Відмінності від Tailwind CSS: Materialize CSS використовує свої класи для стилізації, у той час як Tailwind CSS надає широкий набір класів, які можна використовувати в комбінаціях.

4. Semantic UI:

– Опис: Semantic UI - це інший фреймворк для розробки веб-інтерфейсів, який ставить за мету полегшити процес створення стильних та інтуїтивно зрозумілих інтерфейсів користувача[25]. Розроблений у 2013 році Джеком Луї (Jack Lukic), Semantic UI пропонує елегантний та семантичний код, а також гнучкість при роботі з різними проєктами. Semantic UI був розроблений з ідеєю надання розробникам інструменту, який працює на основі семантики HTML та призначений для створення чистого та зрозумілого коду. Запущений у 2013 році, фреймворк швидко здобув популярність завдяки своїй простоті та гнучкості. Semantic UI спрощує створення інтерфейсу користувача, використовуючи семантичний HTML та лаконічний CSS, має активну спільноту розробників та

широкий набір розширень, які полегшують роботу з різними компонентами. Фреймворк дозволяє вам легко керувати виглядом та поведінкою компонентів, а також забезпечує можливості кастомізації. З недоліків це завантаження усіх ресурсів Semantic UI може збільшити обсяг коду, що впливає на швидкість завантаження сторінки. Іноді інтеграція Semantic UI з іншими бібліотеками або фреймворками може бути складною. Semantic UI не так широко використовується як Bootstrap, і тому може бути менш популярним в деяких випадках.

– Відмінності від Tailwind CSS: Semantic UI працює з використанням класів та атрибутів даних для стилізації елементів, в той час як Tailwind CSS пропонує більш явний та деталізований підхід за допомогою класів.

5. Foundation:

– Опис: Foundation - це фреймворк для веб-розробки, який визначається своєю гнучкістю та можливістю швидко створювати адаптивні та стильні інтерфейси[26]. Розроблений компанією ZURB, Foundation відзначається досвідом інтерфейсного дизайну та вперше був представлений у 2011 році. Foundation забезпечує розробників потужними інструментами для швидкої інтеграції та розробки веб-інтерфейсів. Однією з головних переваг є його гнучка сітка, що робить можливим легке створення адаптивних та відзивчивих інтерфейсів для різних пристроїв. Foundation також славиться обширним набором стилів та компонентів, які дають розробникам велику свободу в кастомізації зовнішнього вигляду своїх проєктів. Однією з можливих недоліків є те, що Foundation може виглядати складним для новачків, оскільки він може включати велику кількість опцій та налаштувань, що може призвести до витрат часу на їх вивчення та розуміння. Також, Foundation, може не мати такої широкої спільноти та популярності, як, наприклад, Bootstrap, що може впливати на доступність ресурсів та підтримку в інтернеті.

– Відмінності від Tailwind CSS: Foundation також має свої класи для стилізації, але підходить до цього іншим способом порівняно з класами Tailwind CSS.

Material-UI (MUI) - це популярна бібліотека React компонентів, яка реалізує

дизайн матеріалу від Google. Ця бібліотека забезпечує готові компоненти, які відповідають концепціям та директивам матеріалу. Хоча Material-UI дуже популярна, існують інші конкуренти, які також пропонують компоненти, зрозумілі стилі та інші можливості для розробки інтерфейсів. Давайте розглянемо деяких з них:

1. Ant Design:

– Опис: Ant Design - це бібліотека React компонентів, яка надає широкий набір готових компонентів для створення сучасних веб-інтерфейсів[27]. Ant Design відзначається своєю компонентною архітектурою, базуючись на дизайн-мові Ant Design Language. Перше випуск був представлений у 2016 році. Ant Design надає розробникам ряд переваг у процесі створення веб-інтерфейсів. Його компоненти мають консистентний та сучасний дизайн, відповідаючи принципам Ant Design Language. Однією з ключових переваг є велика кількість готових компонентів, які полегшують створення складних інтерфейсів. Крім того, Ant Design славиться своєю гнучкістю та можливістю легко кастомізувати стилі та поведінку компонентів. З проблем може виникнути складність інтеграції Ant Design з іншими фреймворками чи бібліотеками, особливо якщо вони використовують власні стилі та компоненти. Іншим аспектом може бути те, що Ant Design більш широко використовується в азійських країнах, що може вплинути на ресурси та підтримку для користувачів, які не входять в цей регіон.

– Відмінності від Material-UI: Ant Design використовує дизайн, що відрізняється від матеріального, і надає власні компоненти та стилі для розробки.

2. Chakra UI:

– Опис: Chakra UI - це сучасна бібліотека компонентів для реактивного створення інтерфейсів користувача веб-додатків з використанням React. Її вирізняють елегантний та декларативний підхід до розробки, який робить код чистим та легким для розуміння[28]. Chakra UI дозволяє швидко створювати стильні та адаптивні інтерфейси, використовуючи готові компоненти та гнучкий API для кастомізації. Однією з головних переваг Chakra UI є його простота використання та ефективність в розробці. Інтуїтивний API та готові компоненти

полегшують роботу розробникам у створенні елегантних інтерфейсів. Бібліотека славиться високою ступенем гнучкості та можливістю кастомізації. Компоненти легко пристосовуються до дизайну проєкту, а система тем дозволяє легко змінювати зовнішній вигляд за допомогою темних або світлих тем. Важливою перевагою є активна підтримка та розвиток Chakra UI, а також його спільнота, що забезпечує надійність та регулярні оновлення. Одним з можливих недоліків може бути обмеження в готових компонентах порівняно з іншими бібліотеками. Хоча Chakra UI має велику кількість корисних компонентів, деякі можливо вам доведеться створити самостійно або використовувати додаткові бібліотеки. Також, через свою релятивно новизну порівняно з іншими фреймворками, Chakra UI може не мати такої широкої розповсюдженості та спільноти, що впливає на доступність ресурсів та підтримку.

– Відмінності від Material-UI: Chakra UI не копіює дизайн матеріалу, але надає свої стилі та компоненти, щоб прискорити процес розробки.

3. Semantic UI React:

– Опис: це фреймворк для розробки веб-інтерфейсів, який визначається своєю семантикою та натхненням від природних мов. Заснований на чіткій та зрозумілій структурі HTML, Semantic UI спрощує процес створення стильних та інтуїтивно зрозумілих інтерфейсів користувача[29]. Однією з головних переваг Semantic UI є лаконічний та семантичний HTML-код, що полегшує взаємодію з розміткою та поліпшує зрозумілість коду. Це робить його особливо привабливим для розробників, які ставлять акцент на семантику та доступність. Фреймворк володіє обширним набором готових компонентів та стилів, що спрощує розробку та полегшує створення стильних інтерфейсів без необхідності великої кількості власного CSS-коду. Одним з можливих недоліків є відносно маленька спільнота та менша популярність фреймворка порівняно з іншими аналогами, такими як Bootstrap чи Materialize CSS. Це може впливати на доступність ресурсів, документації та підтримку у випадку потреби. Semantic UI може вимагати деякого часу для вивчення та зрозуміння, особливо для новачків, оскільки його підходи можуть відрізнятися від інших фреймворків. Також, через його фокус на

семантиці, деякі розробники можуть виявити, що стилі не так легко кастомізувати за порівняно з іншими фреймворками..

– Відмінності від Material-UI: Semantic UI має свій власний дизайн і підходить до розробки за власними стилізованими компонентами.

4. Blueprint:

– Опис: Blueprint - це бібліотека компонентів, яка була створена компанією Palantir Technologies з метою полегшення процесу розробки інтерфейсів користувача для веб-додатків на базі React[30]. Вона була представлена як інструмент для швидкої та ефективної розробки професійних та стильних інтерфейсів. Історія Blueprint почалася з потреби створення набору готових компонентів, які б враховували сучасні стандарти дизайну та були легко відтворюваними в React-додатках. Запущена відносно недавно, Blueprint швидко здобула популярність серед розробників, які шукали ефективний та сучасний інструментарій для своїх проєктів. Одним з основних переваг Blueprint є його чистий та елегантний дизайн компонентів, який сприяє створенню професійного вигляду інтерфейсів. Гнучкість та легкість використання дозволяють розробникам швидко інтегрувати його в свої проекти, забезпечуючи швидкий розвиток. Однак, слід зазначити, що Blueprint може бути менш відомим або менше популярним порівняно з іншими бібліотеками, такими як Ant Design або Material-UI. Це може вплинути на доступність ресурсів, спільноти та документації, зокрема, якщо розробники шукають широкосприйняту підтримку. Загалом Blueprint становить конкурентоспроможну альтернативу для розробників, які шукають бібліотеку React-компонентів зі спрощеним та сучасним дизайном.

– Відмінності від Material-UI: Blueprint має свій стиль та компоненти, не пов'язані з дизайном матеріалу.

5. Evergreen:

– Опис: Evergreen - це бібліотека React-компонентів, розроблена компанією Segment[31]. Вона спрямована на створення сучасних та функціональних інтерфейсів користувача. Evergreen визначається своєю простотою та легкістю використання. Компоненти бібліотеки дозволяють швидко

створювати інтерфейси без значних витрат часу на кастомізацію та стилізацію. Однією з переваг є також чистий та сучасний дизайн компонентів, що відповідає сучасним тенденціям веб-дизайну. Evergreen надає деяку гнучкість в налаштуванні та адаптації компонентів до потреб конкретного проєкту, але при цьому зберігає простоту використання. Одним з можливих недоліків є менша кількість готових компонентів порівняно з іншими популярними бібліотеками, що може зробити необхідним самостійне створення деяких елементів інтерфейсу. Також, через те, що Evergreen не має такої широкої розповсюдженості, як деякі інші бібліотеки, може бути менше ресурсів, документації та спільноти для підтримки. Узагальнюючи, Evergreen є привабливою опцією для тих, хто цінує простоту та потребує базового набору сучасних компонентів для швидкої розробки веб-інтерфейсів.

– Відмінності від Material-UI: Evergreen не прив'язаний до конкретного дизайну та дозволяє розробникам створювати свої власні стилі.

2.2 Аналіз та вибір технологій для побудови логіки веб-додатку

Backend веб-додатку є серцем системи, відповідаючи за всі аспекти, які не є прямо відображеними для користувача, але визначають його функціональність та ефективність[32].

Ця частина програми відповідає за обробку всіх вхідних та вихідних запитів. Вона визначає логіку додатку, керує роботою бази даних, здійснює аутентифікацію користувачів та взаємодіє з іншими сервісами. Backend забезпечує збереження та обробку даних, а також виконання операцій, які потрібні для правильного функціонування програми.

Ця складна система включає в себе різноманітні елементи, такі як сервер, додаткові сервіси, бізнес-логіка, інтерфейси програмування додатків (API) та інше. Управління ресурсами, безпека даних, оптимізація продуктивності - це лише декілька аспектів, які backend вирішує для забезпечення стабільної та ефективної роботи веб-додатку.

Взагалі для написання backend-у використовують мови програмування такі як PHP, C#, Ruby, Java, Python, Go (Golang), але для розробки автоматизованиї системи управління для закладу шкільної освіти були обрано CMS (Content Management System), тому що вона має ряд вагомих переваг над мовами програмування, а саме:

– CMS зазвичай мають готовий функціонал, який можна швидко налаштувати та використовувати. Це може бути вигідним для швидкого старту проекту;
– CMS надає інтерфейс для адміністрування, що полегшує створення та управління контентом;
– CMS може включати в себе різні функції, такі як керування користувачами, SEO-оптимізація, аналітика тощо, що може заощадити час на розробці.

2.2.1 Аналіз і обгрунтування стеку для back-end частини

Для back-end частини було обрано CMS Strapi.

Strapi – це headless CMS, побудована на Node.js та React, яка дозволяє вам використовувати базу даних на ваш вибір. В даний час він підтримує SQLite, MongoDB, MySQL, Oracle (при використанні з плагіном "strapi-hook-orm") та Postgres. Не секрет, що Strapi позиціонує себе альтернативою WordPress. Логотип фреймворка Strapi зображений на рисунку 2.7.

Рисунок 2.7 – Логотип фреймворка Strapi

Ви можете використовувати Strapi для створення всього, що захочете, від простого веб-сайту до великої програми для електронної комерції[33]. На їхньому веб-сайті ви можете знайти безліч ресурсів, щоб дізнатися, як інтегрувати Strapi з найпопулярнішими фреймворками, такими як Gatsby.js, Next.js, Flutter та іншими.

В Strapi інтуїтивно зрозумілий інтерфейс адміністратора як для

користувачів, так і для розробників і редакторів. З точки зору розробника, ви можете легко визначити свою API-модель контенту завдяки їхньому спрощеному конструктору типів контенту. А з погляду редактора вас зустріне інтуїтивно зрозумілий інтерфейс користувача, що дозволяє швидко створювати контент.

Адмін - панель виглядає чистою. Ви можете швидко отримати доступ до функцій, які потрібно використовувати. Панель адміністратора в Strapi зображено на рисунку 2.8.

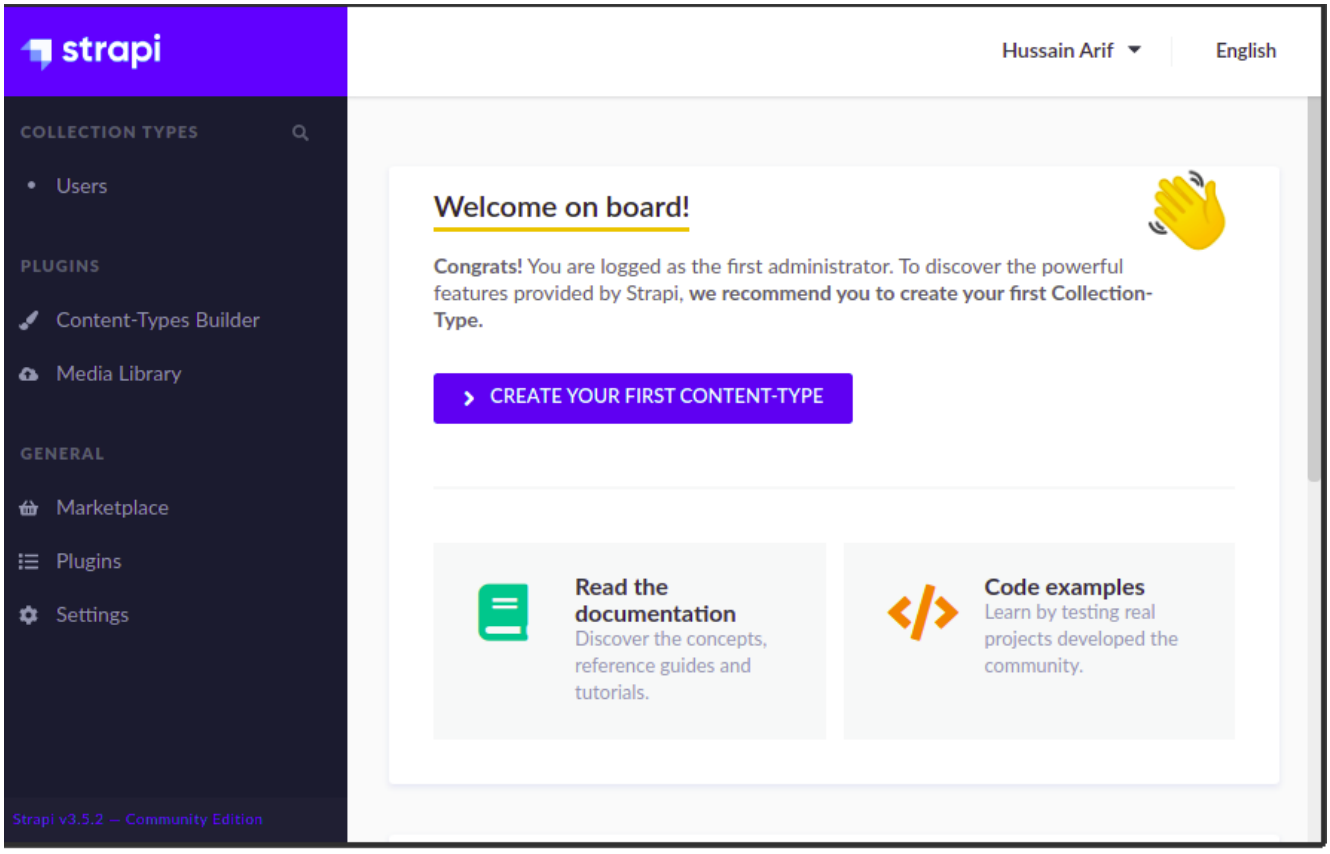


Рисунок 2.8 – Панель адміністратора Strapi

Отже, в основному у вас є ці розділи в галузі адміністрування[34]:

– Типи колекцій / окремі типи: тут мешкає ваша API-модель контенту. Редактори можуть отримати доступ сюди, щоб заповнити дані, які будуть надіслані на веб-сайт/додаток.
– Плагіни: тут можна побачити встановлені в даний момент плагіни. Один із них дуже важливий: Конструктор типів контенту, тому що за його допомогою розробник може створювати типи контенту для API.

– Загальні: перейдіть на ринок плагінів, встановіть нові та отримайте доступ до глобальних налаштувань панелі інструментів та плагінів.

Коли ви встановлюєте Strapi вперше, ви виявите, що деякі з плагінів встановлені за замовчуванням, які забезпечують відмінні функції для вашого проекту, а також інші, які ви, можливо, захочете встановити. Наприклад, нещодавно розробники випустили плагін, який займається інтернаціоналізацією. Цей плагін дозволяє вам створювати багатомовні веб-сайти/додатки та створювати кращий користувальницький досвід для вашої аудиторії.

Якщо ви не бачите функції/плагіна, які можуть знадобитися для вашого конкретного варіанту використання, ви можете створити його, трохи познайомившись з React.js і Node.js, оскільки Strapi створений з використанням цих технологій.

Особливості Strapi:

– Це open source-проект, повністю безкоштовний;
– Система розгортається локально на своєму сервері компанії, що забезпечує безпеку даних;
– CMS можна настроювати та масштабувати за допомогою системи плагінів;
– Передбачено безліч вбудованих можливостей: зручна адміністративна панель, керування автентифікацією та доступами, інструменти для роботи з контентом, генератор API та інше;
– Система має багато варіантів застосування. Вона може використовуватись для статичних сайтів, мобільних додатків, корпоративних ресурсів, електронної комерції;
– На Strapi створюються ультра-швидкі сучасні сайти та мобільні програми. Підвищена продуктивність досягається при використанні Headless CMS у зв'язці зі статичним генератором сайтів та обслуговування через CDN.

Переваги Strapi[35]:

6. Відкритий вихідний код. Система розроблена ентузіастами та підтримується сотнями учасників GitHub, які розвивають її відповідно до нових вимог та технологій. Вона завжди буде доступна та безкоштовна;
7. Широкі та гнучкі налаштування. Панель адміністратора, як і API, легко налаштовується. Функціонал розширюється за рахунок плагінів користувача в лічені секунди;
8. RESTful або GraphQL. CMS підтримує передачу даних у вигляді і REST, і GraphQL. Це розширює можливості взаємодії з різними клієнтами, мобільними програмами, IoT-пристроями;
9. Місцеве розміщення. Розміщення на власному сервері власника системи гарантує конфіденційність та забезпечує підвищений рівень захисту даних (у тому числі відповідно до європейського стандарту GDPR).
10. Одна мова. Система використовує JavaScript, що дозволяє працювати з однією мовою як CMS, так і у фронтенді;
11. Налаштування доступу. У системі реалізовано контроль доступу з урахуванням різних рівнів та ролей користувачів (адміністраторів).

Хоча Strapi має ряд переваг і може бути корисним інструментом для розробки API, він також має деякі недоліки:

Недоліки Strapi:

- Вам потрібні базові знання DevOps, щоб запустити його у власному середовищі;
- Погана підтримка перенесення чи розгортання змін даних між різними середовищами;
- Інтерфейс адміністратора не дуже чуйний;
- Базова підтримка розширеного текстового поля;
- Strapi має невелику спільноту розробників та обмежену кількість

плагінів та розширень, що може обмежити його функціональність;

- Хоча Strapi має хорошу документацію, вона може бути неповною або застарілою, що може ускладнити роботу з фреймворком для деяких розробників;
- Strapi не підтримує старі версії браузерів та операційних систем, що може стати проблемою для тих, хто хоче використати застарілу техніку.

Щоб створити проект на Strapi, вам потрібно:

1. Переконайтеся, що у вас встановлені Node.js та npm.
2. Відкрийте термінал та встановіть Strapi, виконавши команду: npm install strapi@alpha -g.
3. Створіть нову директорію для вашого проекту та перейдіть до неї.
4. Запустіть команду strapi new, щоб створити новий проект Strapi. Виберіть установки, такі як ім'я проекту, база даних тощо.
5. Після успішного встановлення виконайте команду cd my-project, де my-project - ім'я вашого проекту, щоб перейти до директорії проекту.
6. Запустіть команду strapi start, щоб запустити сервер Strapi. Тепер можна відкрити браузер і перейти за адресою http://localhost:1337, щоб відкрити панель управління Strapi.

Ви можете вибрати базу даних, яку ви хочете використовувати, під час встановлення Strapi. За замовчуванням Strapi використовує SQLite. Щоб змінити базу даних у Strapi, слід виконати наступні кроки:

1. Під час створення нового проекту Strapi: при використанні команди для створення нового проекту (strapi new project-name), вас буде попрошено вибрати базу даних. Виберіть опцію, яка вам потрібна (наприклад, PostgreSQL, MySQL, MongoDB), та слідуйте інструкціям.
2. Під час використання існуючого проекту Strapi: якщо ви вже створили проект Strapi і хочете змінити базу даних, вам слід редагувати конфігураційний файл.

- Відкрийте файл config/database.js у вашому проекті Strapi.
- Знайдіть розділ, що відповідає за налаштування бази даних.
- Змініть параметри, такі як client (тип бази даних), host, port, username, password, database тощо, відповідно до параметрів нової бази даних.

Наприклад, якщо ви хочете перейти з SQLite на PostgreSQL, змініть наступні параметри:

```
module.exports = ({ env }) => ({
  defaultConnection: 'default',
  connections: {
    default: {
      connector: 'bookshelf',
      settings: {
        client: 'postgresql', // змініть на 'postgresql'
        host: env('DATABASE_HOST', '127.0.0.1'),
        port: env.int('DATABASE_PORT', 5432),
        database: env('DATABASE_NAME', 'my-project'),
        username: env('DATABASE_USERNAME', 'user'),
        password: env('DATABASE_PASSWORD', 'password'),
        ssl: env.bool('DATABASE_SSL', false),
      },
      options: {}
    },
  },
});
```

Збережіть зміни, а потім перезапустіть ваш Strapi-сервер. Ці кроки дозволять вам змінити базу даних, яку використовує Strapi для вашого проекту.

2.2.2 Аналіз сучасних конкурентів до вибраного стеку back-end частини

Strapi - це відкрита система управління контентом (CMS), яка дозволяє

розробникам створювати API для веб-додатків з інтерфейсом для управління контентом. Це рішення дозволяє ефективно керувати змістом та надає розгалужені можливості для розробки, зокрема, за допомогою свого власного API. Однак існують інші рішення та конкуренти, які також пропонують подібні можливості. Давайте розглянемо кількох конкурентів Strapi та їхніх API:

1. Contentful:

– Опис: Contentful - це система управління контентом (CMS), яка визначається своєю гнучкістю та здатністю надавати контент для різних типів веб-додатків та платформ[36]. Contentful вирізняється гнучкістю у створенні та управлінні контентом. Його API-орієнтована архітектура дозволяє інтегрувати контент у різноманітні застосування та платформи, включаючи веб-сайти, мобільні додатки та інші. Інтерфейс користувача Contentful досить інтуїтивний, спрощуючи завдання створення, редагування та організації контенту. Це робить його доступним для користувачів різного рівня технічної підготовки. Contentful надає підтримку для роботи з різними типами контенту, включаючи текст, зображення, відео та інші медіафайли, що робить його відмінним вибором для проєктів з різноманітним контентом. Одним з можливих недоліків є вартість використання Contentful, особливо для великих проєктів або тих, де обсяг контенту є великим. Вартість може залежати від кількості та типу використовуваних ресурсів. Ще однією можливою обмеженістю є те, що в управлінні контентом Contentful може бути не настільки розширена, як у традиційних CMS, і це може вимагати більше зусиль для вирішення певних завдань. У цілому, Contentful - це потужний інструмент для управління контентом, особливо для проєктів, які вимагають гнучкості та розгортання контенту на різних платформах.

– Відмінності від Strapi: Contentful пропонує готовий хмарний сервіс, у той час як Strapi може бути самостійно розгорнутий на власному сервері. Strapi також може бути безкоштовним, тоді як Contentful має платні тарифи.

2. GraphCMS:

– Опис: GraphCMS - це система управління контентом (CMS), яка

використовує концепцію графів для забезпечення гнучкості та потужності в управлінні контентом[37]. GraphCMS володіє гнучким та потужним інтерфейсом, який дозволяє розробникам та редакторам легко створювати, редагувати та публікувати контент. Гнучкість визначення типів даних та зв'язків у графі дозволяє ефективно організовувати структуру контенту. Використання GraphQL як основного мовлення для отримання даних дозволяє точно визначати та отримувати необхідні дані, зменшуючи зайвий обсяг інформації, яку потрібно завантажувати. GraphCMS може легко інтегруватися з іншими інструментами та сервісами, що робить його практичним вибором для екосистеми з використанням різних технологій. Одним з можливих недоліків є вартість використання GraphCMS, особливо для підприємств або проєктів із обмеженими бюджетами. Вартість може бути значною, особливо при великому обсязі даних та потребі в розширених функціях. Іншим недоліком може бути складність для неініційованих користувачів або тих, хто не має досвіду з GraphQL. Вивчення та розуміння цього запитового мовлення може вимагати часу та зусиль. Усе ж, GraphCMS виконує важливу роль в інструментарії для управління контентом, особливо для проєктів, де важлива гнучкість та ефективність у роботі з контентом.

– Відмінності від Strapi: GraphCMS фокусується на використанні GraphQL API, тоді як Strapi підтримує як REST, так і GraphQL API.

3. Kentico Kontent:

– Опис: Kentico Kontent - це хмарний сервіс для управління контентом, який дозволяє розробникам та контент-редакторам спільно працювати над створенням та управлінням контентом для веб-додатків та інших каналів доставки[38]. З основних переваг Kentico Kontent є його хмарна архітектура, що дозволяє ефективно співпрацювати командам розробників та контент-редакторам з різних місць і віддалених регіонів. Платформа підтримує створення та управління контентом для різних каналів, включаючи веб-сайти, мобільні додатки та IoT-пристрої, що робить її відмінним вибором для проєктів, де необхідна багатоканальна стратегія. Інтерфейс користувача Kentico Kontent досить інтуїтивний та легко використовується, забезпечуючи зручний процес

редагування та розміщення контенту. З можливих недоліків є вартість використання платформи, особливо для великих проєктів або тих, де велика кількість користувачів взаємодіє з контентом. Вартість може бути високою, особливо при великому обсязі та потребі в розширених функціях. Ще однією можливою обмеженістю є те, що платформа може вимагати певного часу для освоєння та вивчення, особливо для нових користувачів чи тих, хто не має попереднього досвіду з CMS.

– Відмінності від Strapi: Kentico Kontent надає розширені можливості для створення моделей даних та керування контентом, але може бути менш гнучким у порівнянні з Strapi.

4. Directus:

– Опис: Directus - це відкритий додаток для самостійного управління контентом, який також пропонує API для роботи з даними. З ключових переваг Directus є його відкритий ісходний код та можливість редагування та адаптації за власними потребами[39]. Розробники можуть налаштовувати та розширювати функціонал, щоб відповідати унікальним вимогам проєктів. Платформа пропонує простий та інтуїтивний інтерфейс, який полегшує користування для контент-редакторів та інших учасників процесу, зокрема тих, хто не має технічного досвіду. Directus забезпечує роботу як з веб-сайтами, так і з мобільними додатками, роблячи його універсальним рішенням для розробки багатьох видів проєктів. З недоліків є те, що відкритий характер Directus може призводити до відсутності офіційного та стабільного сервісу підтримки. Це може вплинути на доступність ресурсів та швидкість вирішення проблем. Для користувачів, які звикли до готових рішень або CMS з широким спектром вбудованих функцій, Directus може виглядати менш повним за стандартними аналогами.

– Відмінності від Strapi: Directus, подібно до Strapi, надає можливість самостійного розгортання, але має інший підхід до налаштування схем даних та інтерфейсу.

5. Prismic:

– Опис: Prismic - це CMS, який надає API для створення та керування

контентом. Він використовує концепцію "slices" для створення різних типів контенту[40]. Prismic визначається своєю простотою використання та інтуїтивним інтерфейсом, що полегшує завдання створення, редагування та публікації контенту для редакторів та контент-менеджерів. Платформа дозволяє розробникам створювати гнучкі контентні моделі, що робить її підходящою для проєктів різного роду та масштабу. Prismic надає зручне керування зображеннями, медіафайлами та іншими активами, спрощуючи процес роботи з мультимедійним контентом. З недоліків може бути обмежена гнучкість у порівнянні з власними рішеннями на основі відкритого коду, оскільки певні аспекти платформи можуть бути менш змінюваними або менш кастомізованими. Також, вартість використання Prismic може бути значною, особливо для великих проєктів або тих, де обсяг контенту є великим.

– Відмінності від Strapi: Prismic має свою унікальну систему для створення контенту, яка використовує поняття "slices", і може бути спрощеним варіантом для деяких випадків в порівнянні з Strapi.

Висновки до розділу

Було проведено глибокий аналіз технологічних аспектів, пов'язаних з розробкою back-end частини веб-додатка. Визначено та обгрунтовано оптимальний стек технологій, враховуючи функціональність, продуктивність та можливості майбутнього розвитку.

Аналіз конкурентів сприяє вибору не лише технологій, а й стратегій, які дозволять побудувати ефективний та конкурентоспроможний back-end. Отримані висновки допомагають забезпечити стабільність, масштабованість та оптимальну продуктивність веб-додатка в умовах постійної зміни технологічного середовища.

Загалом, аналіз та вибір технологій є стратегічно важливим етапом, який визначає успіх розробки веб-додатка, забезпечуючи його високу якість та конкурентоспроможність на ринку.

3 РОЗРОБКА СИСТЕМИ КОНТОЛЮ ВІДВІДУВАНІСТЮ

3.1 Вибір та аналіз інструментів для розробки

Створення веб-додатку включає в себе використання різних інструментів та середовищ, які сприяють розробці, тестуванню та впровадженню веб-проекту. Ось ключові компоненти які використовувались при створенні проекту:

1. Текстовий редактор або Інтегроване Середовище Розробки (IDE):
   – Visual Studio Code - Використовується для написання та редагування коду. Забезпечує зручний інтерфейс та функції для автоматичного завершення коду.
2. Веб-браузер:
   – Google Chrome - Використовується для перевірки та тестування візуального вигляду веб-сайту. Різні браузери можуть відрізнятися в інтерпретації коду та відображенні стилів.
3. Мова програмування:
   – JavaScript - відповідає за динамічність та взаємодію на стороні клієнта.
4. Фреймворк:
   – Next.js - є фреймворком для розробки веб-додатків на мові програмування JavaScript або TypeScript. Він побудований поверх React.js, бібліотеки для створення інтерфейсу користувача, і надає додаткові можливості та інструменти для розробки реактивних веб-сайтів та додатків.
5. Бібліотеки:
   – Material-UI - це бібліотека компонентів для React, яка реалізує дизайн-мову Google Material Design. Material Design визначає стандарти та рекомендації щодо дизайну інтерфейсів, роблячи їх більш консистентними та користувацьки дружелюбними.
6. Система Контролю Версій:

- Git Дозволяє відстежувати зміни в коді, співпрацювати над проектом у команді та відновлювати попередні версії.

7. Система Управління Залежностями:
   - npm (Node Package Manager) Дозволяє встановлювати та оновлювати бібліотеки та інші залежності проекту.
8. API сервер:
   - Strapi - це відкрите програмне забезпечення для створення API. Основна функція Strapi - це надання інтерфейсу для управління та створення API, які використовуються для отримання доступу до бази даних або інших ресурсів. Strapi може використовувати різні бази даних (наприклад, MongoDB, PostgreSQL, MySQL) як сховище для даних, але він сам по собі не є базою даних. Це інструмент для розробки серверних додатків та API, які можна використовувати для різних цілей, таких як створення веб-сайтів, мобільних додатків тощо.

3.2 Розробка плану для реалізації проекту

Розробка веб-проекту передбачає виконання великого обсягу робіт, незалежно від розміру майбутнього проекту. Тому для реалізації успішного продукту необхідно ретельно продумати всі етапи створення веб-сторінки та дотримуватись наміченого плану.

Етапи розробки:

- Визначення тематики та основної мети проекту;
- Розробка технічного завдання;
- Прототипування, макетування та дизайн;
- Верстка та програмування;
- Наповнення контентом;
- Тестування ;
- Здача готового проекту.

Далі про кожен пункт детальніше:

1. Визначення тематики та основної мети проекту. З чого починається робота над веб-додатком? Спочатку потрібно визначити для чого він потрібен. Від цього залежатиме тип ресурсу, портрет цільової аудиторії та основні вимоги. Чітке розуміння мети та остаточного результату допоможе вибудувати ланцюг структури проекту та сформувати які етапи створення необхідні для досягнення мети. Задачею даної МКР є створення автоматизованиї системи управління відвідуваністю для приватного закладу шкільної освіти.

Усі ці особливості необхідно обговорювати на початковому етапі. Для цього потрібно працювати у тісній зв’язці з замовником. Клієнтам дуже часто не вдається висловити загальну ідею. Активна участь у переговорах допоможе розробити концепцію, сформулювати основні цілі та вибрати інструменти для їх досягнення. Тільки після досягнення порозуміння та визначення основних пріоритетів, можна переходити до наступного етапу.

2. Розробка технічного завдання. Технічне завдання – це офіційний документ та фундамент для подальшої роботи. В ньому прописуються всі деталі: структура або мапа додатку (кількість сторінок, розділів, категорій, блоків), вимоги стосовно дизайну, функціонального, візуального та текстового наповнення, а також технічні можливості.

План розробки веб-додатка або ТЗ потребує обов’язкової участі замовника та відповідності наступним вимогам:

- детальність – прописується кожен аспект і всі кроки, які виконуватиме фахівець;
- чіткість – у цьому документі не місце суб’єктивним формулюванням;
- зрозумілість – усі вимоги розписуються зрозумілою для фахівця мовою з використанням відповідної термінології.

Техзавдання – це інструкція, яка буде постійно використовуватися під час розробки сайту. Перейти до основних видів робіт можна буде лише після узгодження всіх питань.

3. Прототипування, макетування та дизайн як основні етапи створення веб-

додатка. На цьому етапі створюється макет, який перевтілить ідеї у реальний об’єкт. Мова не про розробку повноцінного веб-інструменту, готового до роботи, але ви зможете його роздивитися та оцінити переваги. Більшість помилково вважають, що цей пункт стосується лише зовнішнього оформлення сторінок. Насправді, найбільшу увагу спеціалісти приділяють правильному розташуванню елементів з урахуванням правил юзабіліті та інших технічних особливостей. Враховуючи всі ці етапи створення можна забезпечити швидку та стабільну роботу ресурсу.

Після підготовки прототипів та макетів, їх узгоджують з замовником. При необхідності вносяться необхідні зміни, поки проект не буде ухвалено остаточно.

4. Верстка та програмування. Наступний крок – технічна складова. Даний процес передбачає злиття дизайну з двигуном, що перетворює сайт в інструмент з робочими функціями. Спеціалісти для цього використовують знання з основ HTML, підключають CSS стилі, а потім з’єднують з CMS. Слід зазначити, що не всі сайти створюються на основі CMS. Наприклад, прості односторінкові ресурси здатні функціонувати без системи управління контентом.

Далі послідовність створення веб-сайту передбачає надання послуг з програмування. Фахівець повинен «оживити» сайт та наповнити його необхідним функціоналом. У більшості випадків програмування здійснюється на основі CMS, наприклад, на WordPress або сервісі Тільда, але в інших – потребується написання коду з нуля. Наприклад, для того щоб розробити унікальний функціональний блок тощо.

5. Наповнення контентом. Після завершення створення верстки, ми отримуємо по суті робочий інструмент, але з порожніми розділами та сторінками. Їх необхідно заповнити текстовими та графічними матеріалами. Важливо, щоб контент відповідав стандартам оптимізації для подальшого просування ресурсу в пошукових системах.

6. Тестування. Загалом ми вже розглянули основні етапи створення сайту, які стосуються безпосереднього процесу розробки. Тестування – це завершальний етап, який включає проведення різних видів перевірок на предмет помилок,

некоректного функціонування та загальної працездатності ресурсу. Виявлені помилки усуваються фахівцями до тих пір, поки не будуть повністю вирішені. Після розміщення сайту в інтернеті проводять фінальне тестування, щоб перевірити його працездатність.

7. Здача готового проекту. Після передачі замовнику готового проекту, фахівці проводять навчання роботі з сайтом. Це допоможе клієнту самостійно оновлювати інформацію на ресурсі, збирати аналітику, вносити певні зміни та діяти на свій розсуд.

За бажанням клієнт може продовжити роботу з розробником, адже будь-який ресурс потребує подальшого розвитку, підтримки та просування. Наприклад, сайт-візитка після завершення розробки та розміщення на хостингу не потребує особливої уваги або регулярного оновлення інформації. В той час як інтернет-магазин повинен постійно оновлюватися та утримувати позиції в пошукових системах.

3.3 Огляд додатку і розробка користувацьких інтерфейсів

Коли людина заходить на веб-додаток, вона спершу бачить головну сторінку (лендінг). У верхній частині сторінки розташована кнопка "Увійти". На рисунку 3.1 зображено голвне вікно при вході у веб-додаток.

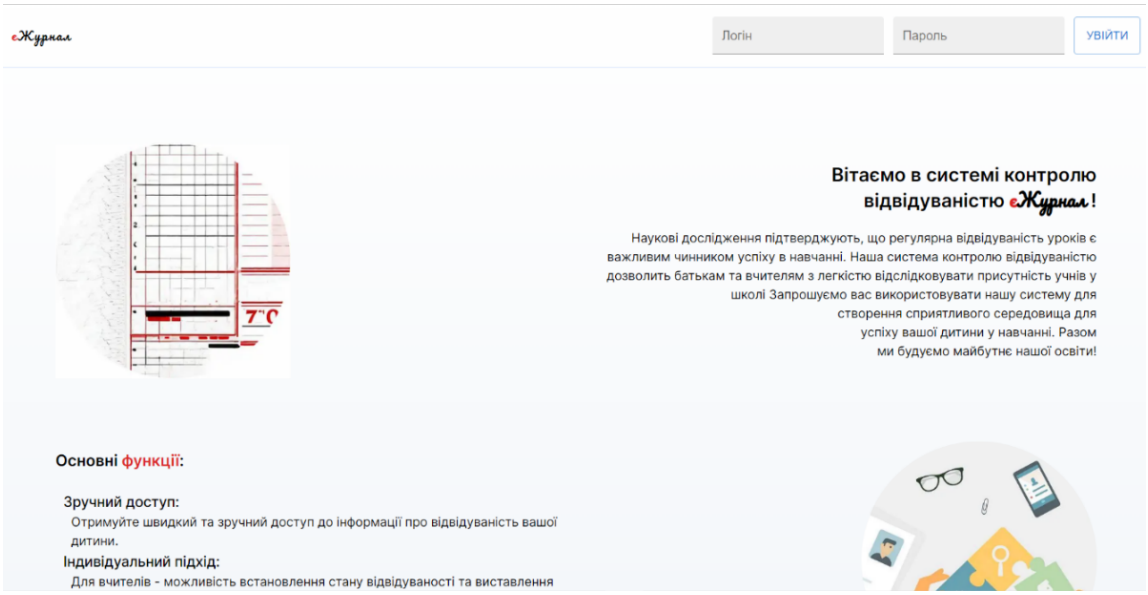


Рисунок 3.1 – Головна сторінка веб-додатку

Якщо користувач вже ввійшов у свій аккаунт, його автоматично перекидає на особистий кабінет, і кнопка "Увійти" змінюється на "Кабінет".

Важливо відзначити, що нові користувачі не можуть створити свій аккаунт самостійно - це може зробити лише адміністратор через спеціальний панель управління.

Коли користувач вводить свій логін і пароль, дані перевіряються спочатку на його комп'ютері для визначення їх правильності (валідація). Якщо дані вірні, вони надсилаються на сервер для перевірки. Якщо сервер підтверджує правильність введених даних, користувачеві надсилається унікальний JSON Web Token (JWT), який зберігається як файл cookie у його браузері. Цей токен використовується для ідентифікації користувача на сервері при кожному його візиті. У проекті буде використовуватись три ролі

– Батьки;
– Викладач;
– Адміністратор.

Розберемо кожну роль детальніше:

Батьки: Вхід здійснюється під профіль дитини, Показується актуальний розклад занять, предмети, клас, в якому навчається дитина, Є можливість перегляду оцінок дитини, виставлених викладачем, В тому ж журналі, якщо дитина була відсутня на занятті, оцінку вона оримати не могла, тому буде символ "В" - дитина була відсутня на занятті. На рисунку 3.2 зображено головне вікно для ролі “Батьки”.

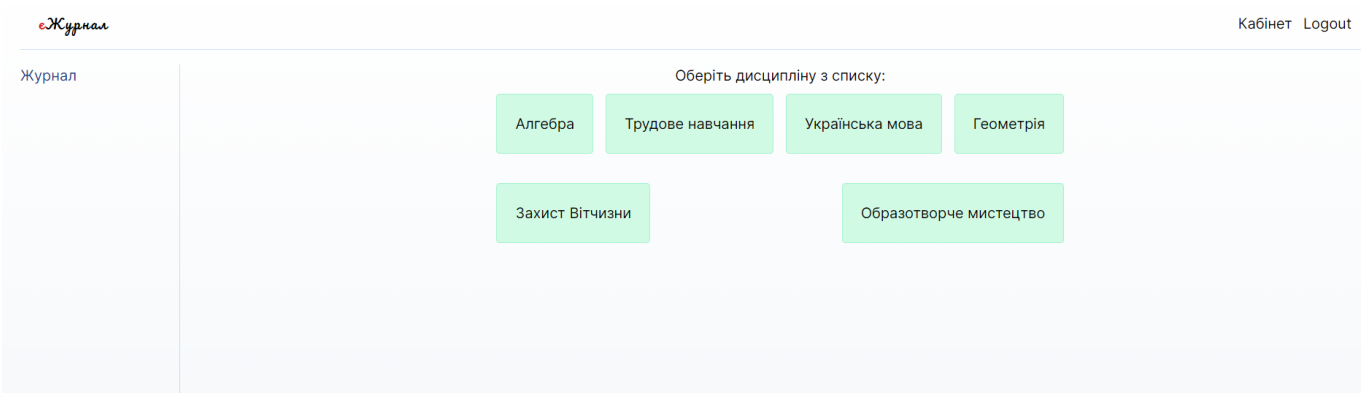


Рисунок 3.2 – Вигляд головного вікна від ролі “Батьки”

Викладач: Вхід здійснюється під профіль викладача, Розклад дисциплін - розклад уроків для викладача, Класи - сторінка, з списком класів, в яких викладач веде предмет. Як тільки обирається клас, викладачеві потрібно обрати предмет який він викладає в даному класі, щоб отримати сторінку журналу по ньому. Сторінка журналу отримується викладачем, після того як він обере клас та дисципліну, графи журналу визначаються самим викладачем, за замовчуванням викладачеві доступний список учнів класу, та декілька граф для оцінок. В графу кожного учня викладач може встановлювати оцінку, відсутність, або залишати графу пустою. Вигляд головного вікна для ролі “Викладач” зображено на рисунку 3.3 та вигляд журналу зображено на рисунку 3.4.

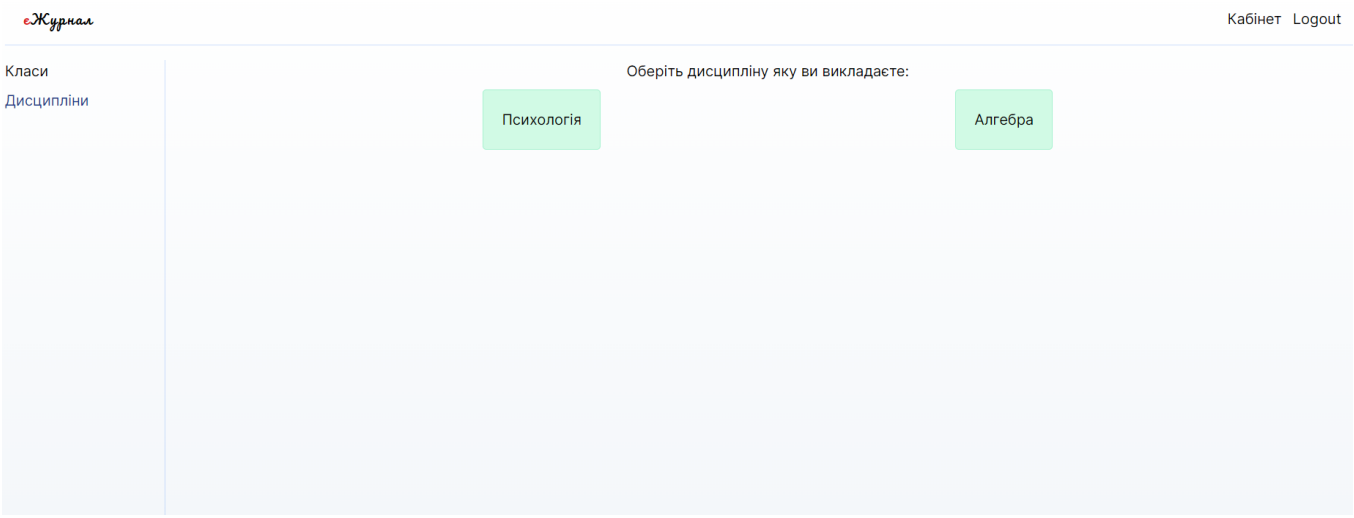


Рисунок 3.3 – Вигляд головного вікна від ролі “ Вчителі”

Рисунок 3.4 – Вигляд журналу викладача

Адміністрація:

Адміністратор має повний доступ до всієї інформаці, вхід здійснюється під профіль адміністратора, в кабінеті на сайдбарі є такі вкладки:

– Класи - Список класів школи, для кожного класу адміністратор може назначати розклад занять, рисунок 3.5;
– Викладачі - список викладачів, адміністратор може назначати викладачеві клас та дисципліну для ведення, також визначається розклад занять для викладача та класу, рисунок 3.6;
– Дисципліни - список всіх класів, та журналів по дисциплінах для них, рисунок 3.7.

Рисунок 3.5 – Список класів вигляд адміністратора

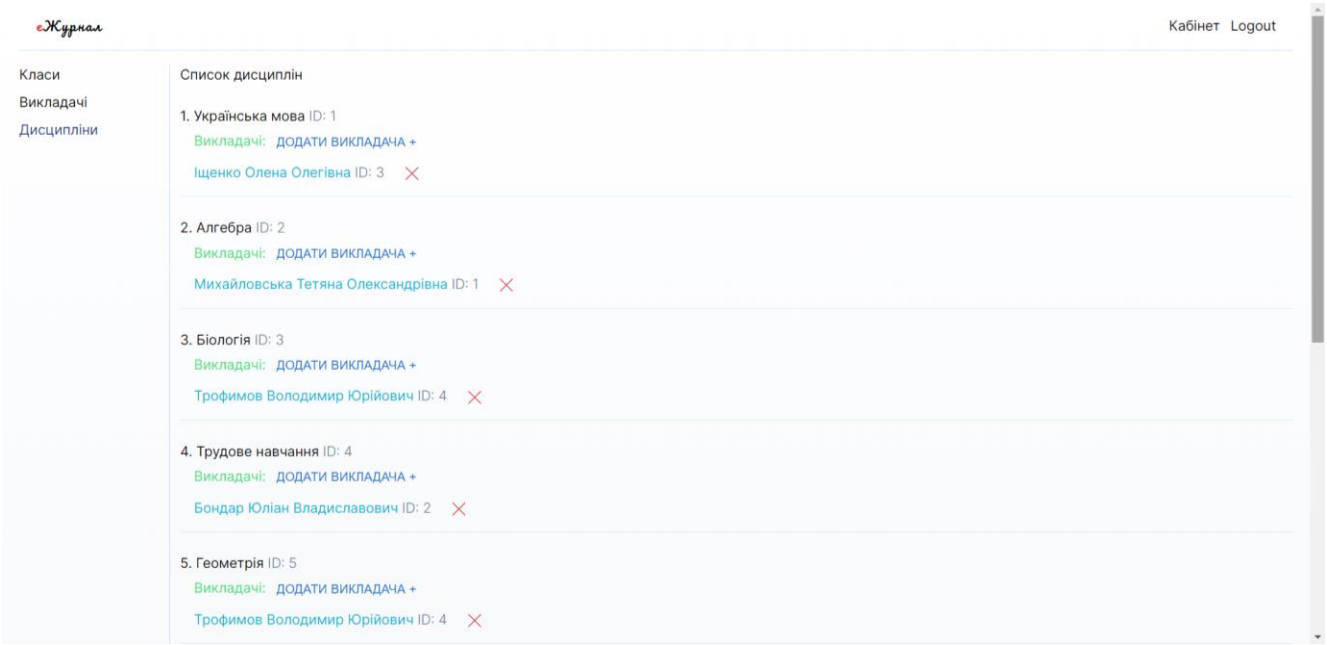


Рисунок 3.6 – Список дисциплін вигляд адміністратора

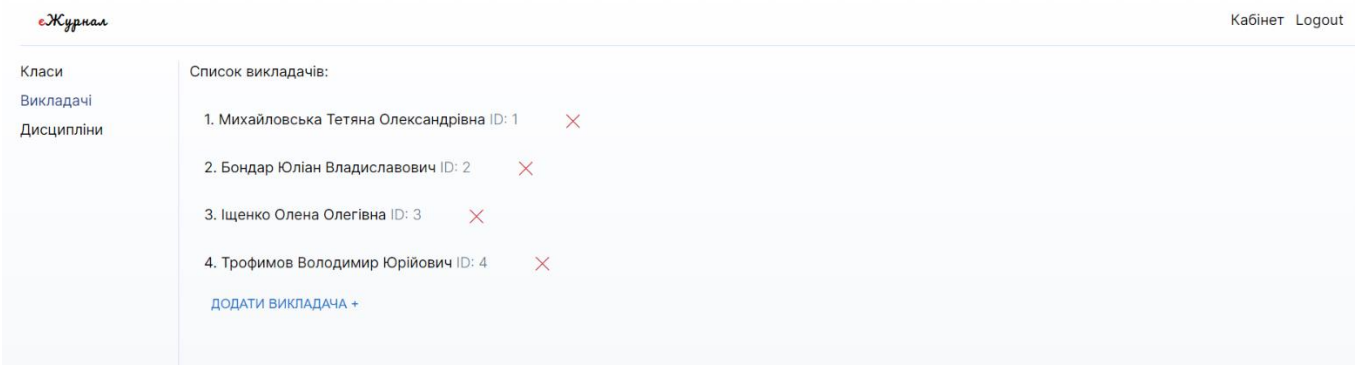


Рисунок 3.7 – Список викладачів вигляд адміністратора

Номер класу визначається роком вступу до навчання, та поточним роком навчання. Також класу присвоюється літера Лік починається з 1 вересня кожного року. В класу є унікальний цифровий ідентифікатор, щоб унеможливити спутування. До прикладу, у класу Class1 має бути літера та рік вступу. Припустимо для Class1 призначили літеру "А", та рік вступу - 2017. Зараз 23 листопада 2023 року - тоді це буде "23.11.2023 - 1.09.2017 - А" клас, тобто 6-а клас. Якщо Нинішня дата мінус дата вступу більше ніж 11, то клас поміщається в архів, та не береться до уваги ніде. Клас має список учнів, клас може мати багато учнів, а учень може мати один клас. Також клас має одного куратора (Класного керівника) з списку викладачів. Куратор може надсилати повідомлення до дитини з списку учнів класу, яким він кирує.

Адміністратор: Класи.

В кабінеті адміністратора є сторінка Класів, де є список всіх класів школи, обравши який адміністратор може призначити розклад занять, коригувати розклад занять. Адміністратор натискає на клас із списку, до прикладу "1-А" відкривається сторінка з опціями по класу:

– Список учнів;
– Список дисциплін.

Адміністратор може коригувати список учнів, додаючи або видаляючи їх, в списку дисциплін адміністратор встановлює, коригує або видаляє розклад занять.

Адміністратор: Викладачі.

В сайдбарі адміністратора міститься пункт "Викладачі", де зібраний список викладачів, обравши викладача із списку, адміністратор може додати чи видалити йому дисципліни які він може викладати. Вигляд головного вікна для адміністратора на корегування класа та викладачів зображено на рисунку 3.8.

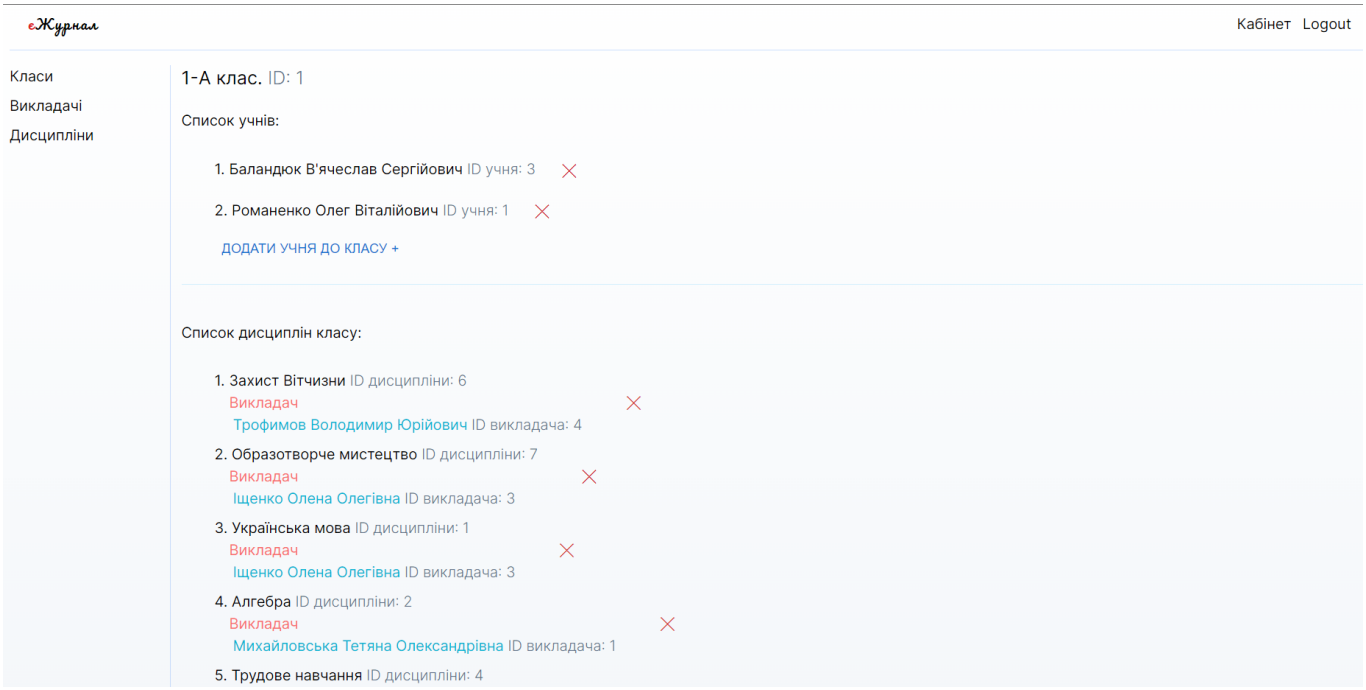


Рисунок 3.8 – Вигляд від адміністратора на корегування класа та викладачів

Виставлення оцінок. По задумці, викладач обирає клітинку в журналі, і коли натискає на неї, з'являється діалогове вікно, в якому вказана дата уроку, дисципліна, ПІБ дитини, з вибору 2 радіокнопки Присутній(я) - за замовчуванням, Відсутній(я) - якщо обрана відсутність, то оцінка бути виставленя не може. Далі йде input type number min=1 max=12, та кнопка виставити Інпут може бути використаний тільки при умові що обрано радіокнопку "Присутній(я)"

Виставлення оцінок: Першим йде запит на уроки за фільтром предмету та класу (кількість занять). Заняття створюються вчителем. Далі йде запит на учнів обраного класу, Після чого йде запит на оцінки учня по предмету, де вказано за який урок яка оцінка виставлена. Оцінок може не бути, тому клітинка буде пустою, Оцінкою може бути і не циферне значення, а відмітка "В" - відсутність. На рисунку 3.9 зображено вікно виставлення оцінки чи присутності учня.

Ідеал: Отримати всі данні в локальний стейт та відобразити все з нього, задля того, щоб не робити повторного get запиту на сервер після кожного виставлення оцінки. Процес такий: Всі вищеописані дані для відмалюваня журналу отримуються, та передаються в локальний стейт, як тільки викладач виставляє оцінку, відправляється POST-запит на сервер. Як тільки прийде відповідь (200), тоді змінюємо локальний стейт, якщо відповідь не 200 - виводимо помилку. Таким чином кожна оцінка викличе перемалюваня всієї таблиці, але не отримання всіх данних заново

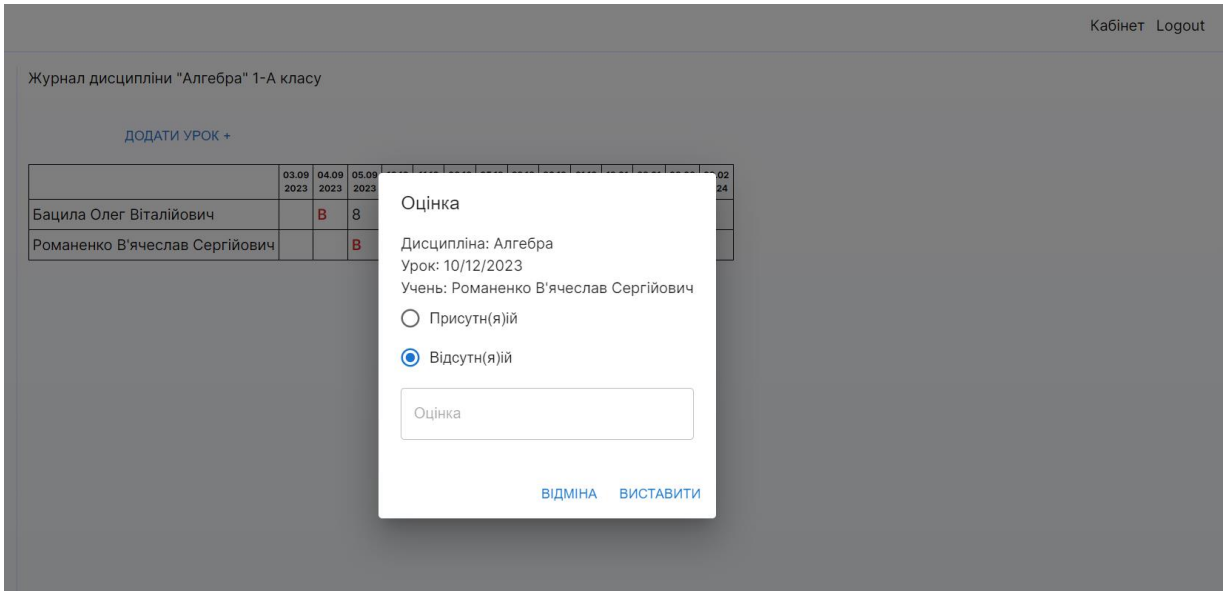


Рисунок 3.9 – Виставлення оцінки чи присутності учня

Для того щоб отримати оцінки, потрібно зробити запит на marks з фільтрами.

&filters[student][id][$eq]=${childId}&filters[subject][id][$eq]= ${disciplineId}

Таким чином отримаються дати та оцінки, а пусті клітинки – ні, для того щоб отримати дні, де учень не отримав оцінку, потрібно зробити запит на журнал дисципліни в класі. Щоб зробити запит на журнал по дисципліни в класі, потрібно запитати журнал з фільтрами клас та дисципліна. Клас береться з профілю батьків, id дисципліни обирається на фронтенді батьками (переходом на сторінку).

http://localhost:1337/api/test-journals?populate=*&filters[class][id][$eq]=1&filters[subject][id] [$eq]=1

Так ми отримуємо json:


```
{
  ('data');
  [
    {
      id: 1,
      attributes: {
        createdAt: '2023-11-30T18:09:46.768Z',
        updatedAt: '2023-11-30T18:11:31.712Z',
        publishedAt: '2023-11-30T18:11:31.709Z',
        class: {
          data: {
            id: 1,
            attributes: {
              identifier: '1',
              createdAt: '2023-11-27T13:09:25.047Z',
              updatedAt: '2023-11-28T18:40:16.443Z',
              publishedAt: '2023-11-27T13:10:11.533Z',
              admissionYear: '2023-09-01',
              classLetter: 'A',
            },
          },
        },
        teacher: {
          data: null,
        },
        subject: {
          data: {
            id: 1,
            attributes: {
              createdAt: '2023-11-27T13:11:32.519Z',
              updatedAt: '2023-11-27T13:11:51.816Z',
              publishedAt: '2023-11-27T13:11:51.807Z',
              name: 'Алгебра',
            },
```

```
        },
      },
      test_lessons: {
        data: [
          {
            id: 1,
            attributes: {
              lessonDate: '2023-11-26T08:45:00.000Z',
              createdAt: '2023-11-30T18:10:24.001Z',
              updatedAt: '2023-11-30T18:10:28.645Z',
              publishedAt: '2023-11-30T18:10:28.643Z',
            },
          },
          {
            id: 2,
            attributes: {
              lessonDate: '2023-11-28T11:30:00.000Z',
              createdAt: '2023-11-30T18:10:41.411Z',
              updatedAt: '2023-11-30T18:10:43.581Z',
              publishedAt: '2023-11-30T18:10:43.577Z',
            },
          },
          {
            id: 3,
            attributes: {
              lessonDate: '2023-11-29T07:30:00.000Z',
              createdAt: '2023-11-30T18:11:02.611Z',
              updatedAt: '2023-11-30T18:11:06.943Z',
              publishedAt: '2023-11-30T18:11:06.941Z',
            },
          },
        ],
      },
    },
  },
],
```

```
    'meta';
  {
    ('pagination');
    {
      ('page');
      1;
      ('pageSize');
      25, 'pageCount';
      1, 'total';
      1;
    }
  }
}
```

З якого беремо test_lessons.data та мапимо його в хедер таблиці. Скільки уроків, стільки й клітинок. Ролі користувачів, та варіанти використання системи зображені на рисунку 3.10.

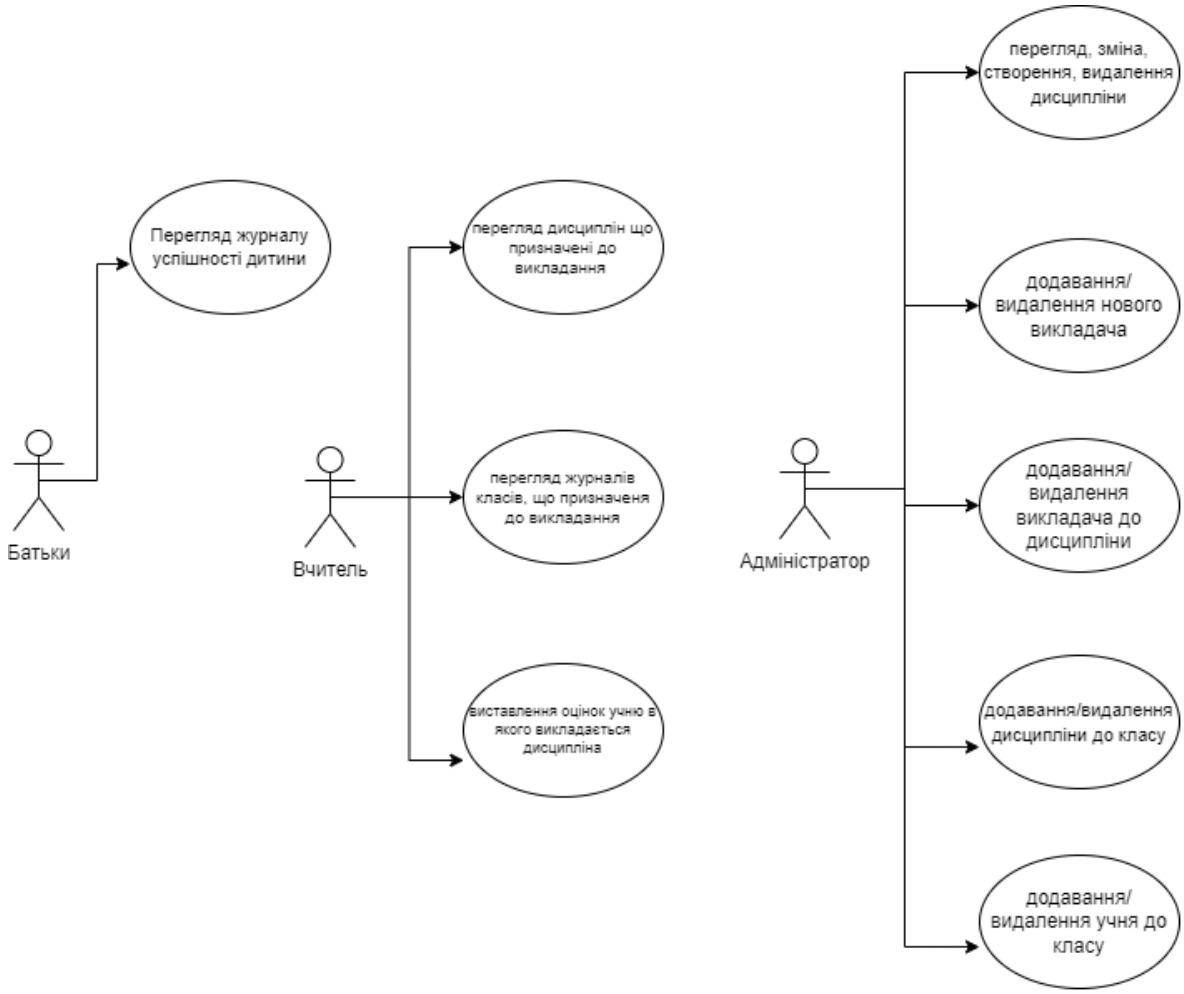


Рисунок 3.10 – UML діаграма варіантів використання ролей

3.4 Створення та налаштування системи керування вмістом

На цьому етапі розробки детально розглядається back-end частина веб-додатка, яка відповідає за обробку даних, виконання бізнес-логіки та взаємодію з базою даних. Вибір правильних технологій та інструментів для реалізації back-end є критичним завданням, що визначає ефективність та стабільність роботи веб-додатка. В даному проекті як відомо використовувалась система керування вмістом Strapi.

Всі дані зберігаються в базі даних і можуть бути відредаговані адміністратором через панель управління Strapi. Це забезпечує надійний та безпечний доступ до даних для кожного користувача, відповідно до його ролі та прав доступу

В Strapi, колекції (Collections) є ключовим елементом для організації та визначення структури даних. Кожна колекція представляє собою сукупність даних, яку ви хочете зберігати в системі. Strapi використовує поняття "колекцій" для управління структурою та типами даних в базі даних. Колекції є важливим елементом в Strapi, оскільки вони визначають схеми даних та їхні взаємовідношення.

Створення та управління колекціями в Strapi дозволяє розробникам визначати структуру даних, обмінюватись інформацією між різними частинами веб-додатка та забезпечує зручний інтерфейс для роботи з даними в адміністративному розділі Strapi. Кожна колекція може включати поля з різними типами даних та визначати їхні взаємовідношення, що робить моделювання даних гнучким та пристосованим до конкретних потреб веб-додатка. Основні аспекти колекцій у Strapi включають[41]:

1. Типи колекцій:
   - Однозначна (Single): Використовується для зберігання одного екземпляра даних, наприклад, налаштувань сайту.
   - Множинна (Collection): Використовується для зберігання кількох екземплярів даних, таких як статті в блозі чи

користувачі.

2. Поля колекцій:
   - Кожна колекція має свій набір полів, які визначають структуру даних. Наприклад, для колекції "Статті" можуть бути поля, такі як заголовок, зміст, дата публікації тощо.
3. Відносини:
   - Колекції можуть взаємодіяти між собою за допомогою відносин. Наприклад, статті можуть мати відносини з авторами (колекція "Користувачі").
4. Автоматична документація:
   - Strapi автоматично генерує документацію API на основі ваших колекцій, що полегшує розуміння доступних ендпоінтів та їх параметрів.
5. Доступ та авторизаія:
   - Можливість обмежувати доступ до колекцій за допомогою налаштувань ролей та авторизації.

Колекції у Strapi надають можливості для роботи з даними та їхнього управління в back-end частині веб-додатка. Ось деякі дії, які можна виконувати з колекціями:

- Розробники можуть визначати нові колекції та визначати їхню структуру за допомогою різних полів з різними типами даних;
- Змінення структури колекції, додавання або видалення полів, зміна типів даних;
- Додавання та редагування конкретних записів у колекціях для наповнення бази даних реальними даними;
- Встановлення зв'язків між різними колекціями для моделювання взаємозв'язків між об'єктами даних;
- Використання запитів для отримання конкретних даних з колекцій або виконання операцій фільтрації та сортування;

- Встановлення прав доступу до колекцій для забезпечення безпеки даних та обмеження доступу до ресурсів;
- Можливість експортувати та імпортувати дані з колекцій для зручного переносу та резервного копіювання інформації.

Ці можливості дозволяють розробникам ефективно управляти даними в back-end частині веб-додатка, створюючи гнучкі та динамічні структури для роботи з різними типами інформації.

Далі розглянемо типо колекцій які використовуються у проекті:

1. Тип колекції Class - учнівський клас. Має такі поля:
   - admissionYear типу Date - рік початку навчання. За замовчуванням 1 вересня. classLetter - літера класу (наприклад 11-А, 11-Б).

   Зв'язки:
   - Students: Class belongs to many (один до багатьох) Students;
   - Subjects: Class belongs to many (один до багатьох) Subjects;
   - Journals: Class belongs to many (один до багатьох) Journals;
   - Teachers: Class has and belongs to many (багато до багатьох) Teachers.
2. Тип колекції Journal - журнал класу по предмету. Не має полів, але має зв'язки:
   - Subject: Subject has many (багато до одного) Journals;
   - Lessons: Journal belongs to many (один до багатьох) Lessons;
   - Class: Class has many Journals (багато до одного);
   - Teacher: Teacher has many (багато до одного) Journals.
3. Тип колекції Lesson - урок. Має поле:
   - lessonDate типу Data - дата уроку.

   Зв'язки:
   - Journal: Journal has many Lessons;
   - Subject: Subject has many Lessons;

- Marks: Lesson belongs to many Marks.

4. Тип колекції Mark - оцінка учня за урок. Має поле:
   - value типу text (short text) - оцінка або відсутність.

   Зв'язки:
   - Student: Student has many Marks;
   - Subject: Subject has many Marks;
   - Lesson: Lesson has many Marks.
5. Тип колекції Student - учень. Має текстове поле:
   - name (short text) - ПІБ учня.

   Зв'язки:
   - Marks: Student belongs to many Marks;
   - Class: Class has many Students.
6. Тип колекції Subject - предмет (дисципліна). Має поле:
   - name типу Text (short text) - назва дисципліни.

   Зв'язки:
   - Marks: Subject belongs to many Marks;
   - Class: Class has many Subjects;
   - Journals: Subject belongs to many Journals;
   - Lessons: Subject belongs to many Lessons;
   - Teacher: Teacher has many Subjects.
7. Тип колекції Teacher - вчитель. Має поле:
   - name типу Text (short text).

   Зв'язки:
   - Subjects: Teacher belongs to many Subjects;
   - Journals: Teacher belongs to many Journals;
   - Classes: Teacher has and belongs to many Classes.

Вигляд всіх типів колекцій в Strapi у проекті зображено на рисунку 3.11 та головна панель управління зображена на рисунку 3.12.

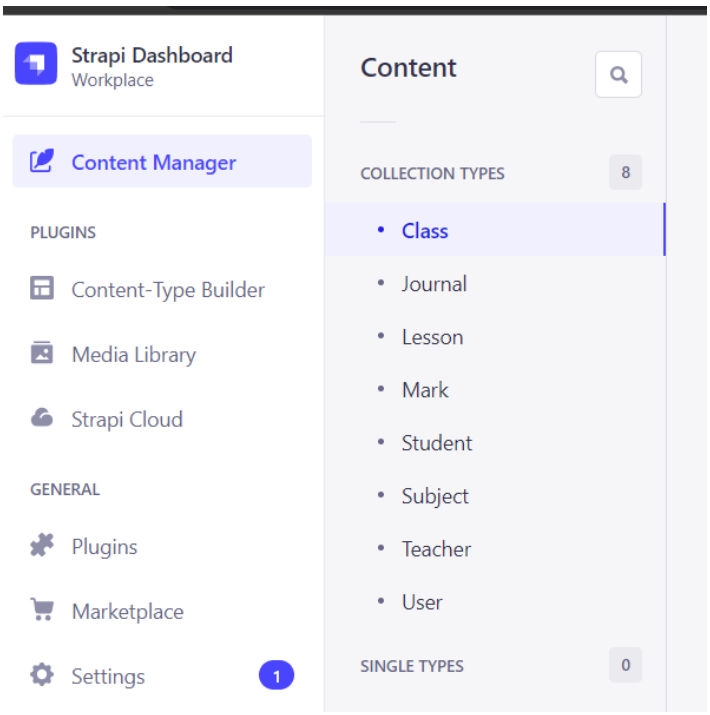


Рисунок 3.11 – Вигляд колекцій у проекті

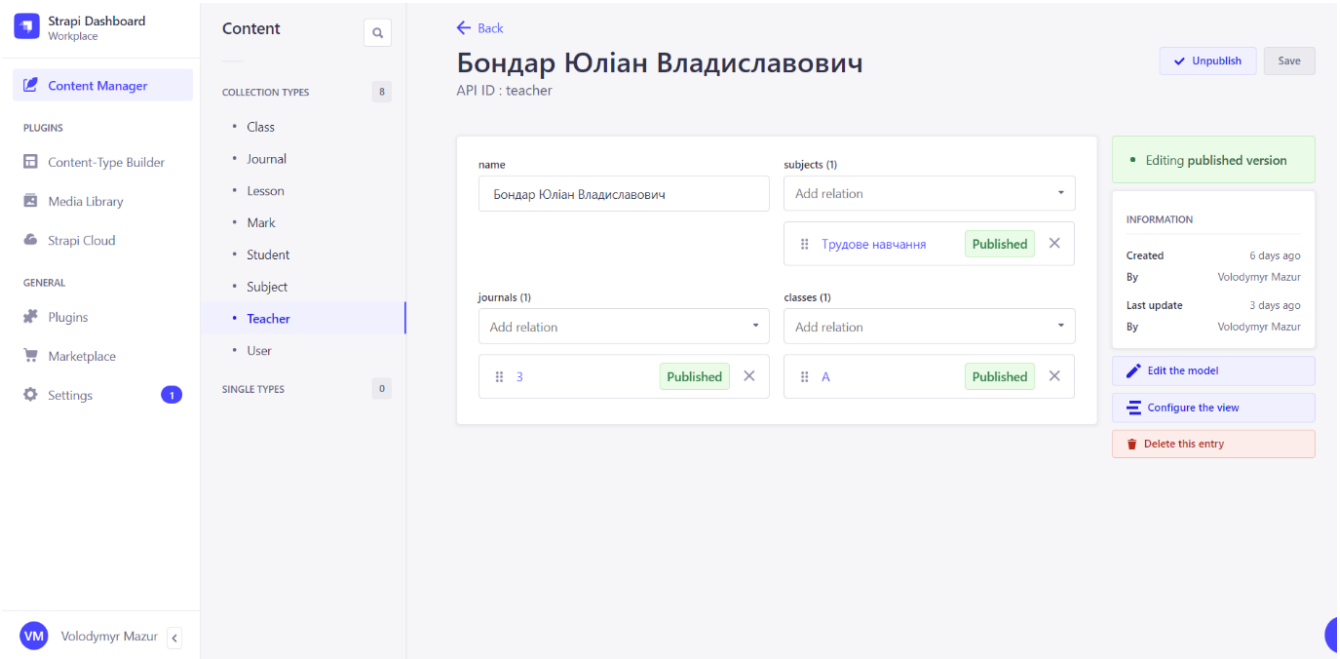


Рисунок 3.12 – Панель управління Strapi

Висновки до розділу

У процесі розробки веб-додатка важливо правильно вибрати інструменти для розробки, щоб забезпечити ефективність та якість проекту. Під час аналізу інструментів для розробки слід враховувати їхню сумісність, функціональність та

зручність у використанні. Розробка плану для реалізації проекту визначає порядок виконання завдань та дозволяє керувати процесом розробки, забезпечуючи вчасну доставку результатів.

Огляд додатку і розробка користувацьких інтерфейсів грають ключову роль у взаємодії користувачів з продуктом. Важливо враховувати зручність, інтуїтивність та естетичний аспект дизайну, щоб створити позитивний користувацький досвід.

Створення та налаштування системи керування вмістом є необхідним етапом для забезпечення ефективного управління контентом веб-додатка. Важливо вибрати або розробити систему, яка відповідає конкретним потребам проекту та дозволяє легко оновлювати та модифікувати вміст.

Інтеграція цих етапів дозволить забезпечити успішну та збалансовану розробку веб-додатка, враховуючи технічні, дизайнерські та управлінські аспекти.

4 ЕКОНОМІЧНА ЧАСТИНА

Науково-технічна розробка має право на існування та впровадження, якщо вона відповідає вимогам часу, як в напрямку науково-технічного прогресу та і в плані економіки. Тому для науково-дослідної роботи необхідно оцінювати економічну ефективність результатів виконаної роботи.

Магістерська кваліфікаційна робота за темою «Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти» відноситься до науково-технічних робіт, які плануються для використання безпосередньо самим розробником (замовником), тобто її результатами буде користуватися тільки одна особа – розробник (або замовник). У цьому випадку нам потрібно довести ефективність інвестицій, вкладених у цей проект самим розробником (замовником).

Для цього випадку необхідно виконати такі етапи робіт:

1) провести технологічний аудит власної науково-технічної розробки, тобто встановити її науково-технічний рівень;

2) розрахувати витрати на здійснення науково-технічної розробки;

3) провести розрахунок економічної ефективності науково-технічної розробки у випадку її впровадження розробником (замовником) на власному підприємстві та обґрунтувати економічну доцільність впровадження розробником (замовником) розробленого науково-технічного проекту.

4.1 Проведення комерційного та технологічного аудиту науково-технічної розробки

Метою проведення комерційного і технологічного аудиту дослідження за темою «Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти» є оцінювання науково-технічного рівня та рівня комерційного потенціалу розробки, створеної в результаті науково-технічної діяльності.

Оцінювання науково-технічного рівня розробки та її комерційного

потенціалу рекомендується здійснювати із застосуванням 5-ти бальної системи оцінювання за 12-ма критеріями, наведеними в табл. 4.1[42].

Таблиця 4.1 – Рекомендовані критерії оцінювання науково-технічного рівня і комерційного потенціалу розробки та бальна оцінка

Бали (за 5-ти бальною шкалою)					
	0	1	2	3	4
Технічна здійсненність концепції					
1	Достовірність концепції не підтверджена	Концепція підтверджена експертними	Концепція підтверджена розрахунками	Концепція перевірена на практиці	Перевірено працездатність продукту в
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
3	Ціна продукту значно вища за ціни аналогів	Ціна продукту дещо вища за ціни аналогів	Ціна продукту приблизно дорівнює цінам	Ціна продукту дещо нижче за ціни аналогів	Ціна продукту значно нижче за ціни аналогів
4	Технічні та споживчі властивості продукту значно	Технічні та споживчі влас-тивості продукту трохи гірші, ніж	Технічні та споживчі властивості продукту на рівні	Технічні та споживчі властивості продукту трохи	Технічні та споживчі властивості продукту значно
5	Експлуатаційні витрати значно вищі, ніж в	Експлуатаційні витрати дещо вищі, ніж в	Експлуатаційні витрати на рівні експлуатаційних	Експлуатаційні витрати трохи нижчі, ніж в	Експлуатаційні витрати значно нижчі, ніж в
Ринкові перспективи					
6	Ринок малий і не має позитивної	Ринок малий, але має позитивну	Середній ринок з позитивною	Великий стабільний ринок	Великий ринок з позитивною
7	Активна конкуренція великих	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	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	Потрібні незначні фінансові ресурси. Джерела фінансування	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	Потрібні дорогі матеріали	Потрібні досяжні та дешеві матеріали	Всі матеріали для реалізації ідеї відомі та давно використовуютьс
11	Термін реалізації ідеї більший за 10 років	Термін реалізації ідеї більший за 5 років. Термін окупності інвестицій	Термін реалізації ідеї від 3-х до 5-ти років. Термін окупності інвестицій більше	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій від 3-х	Термін реалізації ідеї менше 3-х років. Термін окупності інвестицій менше
12	Необхідна розробка регламентних документів та отримання великої кількості дозвільних документів на	Необхідно отримання великої кількості дозвільних документів на виробництво та реалізацію продукту, що	Процедура отримання дозвільних документів для виробництва та реалізації продукту вимагає незначних коштів	Необхідно тільки повідомлення відповідним органам про виробництво та реалізацію продукту	Відсутні будь- які регламентні обмеження на виробництво та реалізацію продукту

Результати оцінювання науково-технічного рівня та комерційного потенціалу науково-технічної розробки потрібно звести до таблиці.

Таблиця 4.2 – Результати оцінювання науково-технічного рівня і комерційного потенціалу розробки експертами

Критерії	Експерт (ПІБ, посада)		
	1	2	3
	Бали:		
1. Технічна здійсненність концепції	3	4	3
2. Ринкові переваги (наявність аналогів)	3	3	2
3. Ринкові переваги (ціна продукту)	3	2	2
4. Ринкові переваги (технічні властивості)	3	2	2

5. Ринкові переваги (експлуатаційні витрати)	2	2	2
6. Ринкові перспективи (розмір ринку)	2	2	2
7. Ринкові перспективи (конкуренція)	2	2	2
8. Практична здійсненність (наявність фахівців)	4	4	4
9. Практична здійсненність (наявність фінансів)	2	3	2
10. Практична здійсненність (необхідність нових матеріалів)	2	2	2
11. Практична здійсненність (термін реалізації)	3	4	4
12. Практична здійсненність (розробка документів)	4	4	4
Сума балів	33	34	31
Середньоарифметична сума балів $СБ_c$	32,7		

За результатами розрахунків, наведених в таблиці 4.2, зробимо висновок щодо науково-технічного рівня і рівня комерційного потенціалу розробки. При цьому використаємо рекомендації, наведені в табл. 4.3[42].

Таблиця 4.3 – Науково-технічні рівні та комерційні потенціали розробки

Середньоарифметична сума балів СБ ,розрахована на основі висновків експертів	Науково-технічний рівень та комерційний потенціал розробки
41…48	Високий
31…40	Вище середнього
21…30	Середній
11…20	Нижче середнього
0…10	Низький

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти» становить 32,7 бала, що, відповідно до таблиці 4.3, свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

4.2 Розрахунок узагальненого коефіцієнта якості розробки

Окрім комерційного аудиту розробки доцільно також розглянути технічний рівень якості розробки, розглянувши її основні технічні показники. Ці показники по-різному впливають на загальну якість проектної розробки.

Узагальнений коефіцієнт якості ($B_н$) для нового технічного рішення розрахуємо за формулою[43]:

$$B_н = \sum_{i=1}^{k} \alpha_i \cdot \beta_i \quad , \tag{4.1}$$

де k – кількість найбільш важливих технічних показників, які впливають на якість нового технічного рішення;

αi – коефіцієнт, який враховує питому вагу і-го технічного показника в загальній якості розробки. Коефіцієнт αi визначається експертним шляхом і при цьому має виконуватись умова $\sum_{i=1}^{k} \alpha_i = 1$;

β_i – відносне значення *і-го* технічного показника якості нової розробки.

Відносні значення β_i для різних випадків розраховуємо за такими формулами:

для показників, зростання яких вказує на підвищення в лінійній залежності якості нової розробки:

$$\beta_i = \frac{I_{ні}}{I_{аі}} \quad , \tag{4.2}$$

де $I_{ні}$ та $I_{на}$ – чисельні значення конкретного *і*-го технічного показника якості відповідно для нової розробки та аналога;

для показників, зростання яких вказує на погіршення в лінійній залежності якості нової розробки:

$$\beta_i = \frac{I_{аі}}{I_{ні}} \quad ; \tag{4.3}$$

Використовуючи наведені залежності можемо проаналізувати та порівняти техніко-економічні характеристики аналогу та розробки на основі отриманих

наявних та проектних показників, а результати порівняння зведемо до таблиці 4.4.

Таблиця 4.4 – Порівняння основних параметрів розробки та аналога.

Показники (параметри)	Одиниця вимірю-вання	Аналог	Проектований пристрій	Відношення параметрів нової розробки до аналога	Питома вага показника
Кількість показників відвідуваності що підлягають обліку	шт.	5	10	2	0,3
Кількість місць розміщення (хмара, мобільно, ПК)	шт.	2	3	1,5	0,1
Рівень захищеності баз даних (за 10-бальною шкалою)	бал	5	8	1,6	0,15
Доступність інтерфейсу	бал	7	9	1,26	0,25
Можлива кількість додатково інтегрованих модулів	шт.	2	4	2	0,2

Узагальнений коефіцієнт якості ($B_н$) для нового технічного рішення складе:

$$B_н = \sum_{i=1}^{k} \alpha_i \cdot \beta_i = 2 \cdot 0{,}3 + 1{,}5 \cdot 0{,}1 + 1{,}6 \cdot 0{,}15 + 1{,}26 \cdot 0{,}25 + 2 \cdot 0{,}2 = 1{,}71.$$

Отже за технічними параметрами, згідно узагальненого коефіцієнту якості розробки, науково-технічна розробка переважає існуючі аналоги приблизно в 1,71 рази.

4.3 Розрахунок витрат на проведення науково-дослідної роботи

Витрати, пов'язані з проведенням науково-дослідної роботи на тему «Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти», під час планування, обліку і калькулювання собівартості науково-дослідної роботи групуємо за відповідними статтями.

4.3.1 Витрати на оплату праці

До статті «Витрати на оплату праці» належать витрати на виплату основної та додаткової заробітної плати керівникам відділів, лабораторій, секторів і груп, науковим, інженерно-технічним працівникам, конструкторам, технологам, креслярам, копіювальникам, лаборантам, робітникам, студентам, аспірантам та іншим працівникам, безпосередньо зайнятим виконанням конкретної теми, обчисленої за посадовими окладами, відрядними розцінками, тарифними ставками згідно з чинними в організаціях системами оплати праці.

Основна заробітна плата дослідників

Витрати на основну заробітну плату дослідників ($З_о$) розраховуємо у відповідності до посадових окладів працівників, за формулою[42]:

$$З_о = \sum_{i=1}^{k} \frac{M_{ni} \cdot t_i}{T_p}, \qquad (4.4)$$

де k – кількість посад дослідників залучених до процесу досліджень;

M_{ni} – місячний посадовий оклад конкретного дослідника, грн;

t_i – число днів роботи конкретного дослідника, дн.;

T_p – середнє число робочих днів в місяці, T_p=22 дні.

$$З_о = 16950{,}00 \cdot 44 / 22 = 33900{,}00 \text{ грн.}$$

Проведені розрахунки зведемо до таблиці.

Таблиця 4.5 – Витрати на заробітну плату дослідників

Найменування посади	Місячний посадовий оклад, грн	Оплата за робочий день, грн	Число днів роботи	Витрати на заробітну плату, грн
Керівник проекту	16950,00	770,45	44	33900,00
Інженер-програміст	16900,00	768,18	40	30727,27
Інженер-проектувальник автоматизованих систем управління	16050,00	729,55	22	16050,00
Консультант (вчитель-методист вищої категорії)	12320,00	560,00	5	2800,00
Всього				83477,27

Основна заробітна плата робітників

Витрати на основну заробітну плату робітників ($З_р$) за відповідними найменуваннями робіт НДР на тему «Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти» розраховуємо за формулою:

$$З_р = \sum_{i=1}^{n} C_i \cdot t_i , \quad (4.5)$$

де C_i – погодинна тарифна ставка робітника відповідного розряду, за виконану відповідну роботу, грн/год;

t_i – час роботи робітника при виконанні визначеної роботи, год.

Погодинну тарифну ставку робітника відповідного розряду C_i можна визначити за формулою:

$$C_i = \frac{M_M \cdot K_i \cdot K_c}{T_p \cdot t_{зм}} , \quad (4.6)$$

де M_M – розмір прожиткового мінімуму працездатної особи, або мінімальної

місячної заробітної плати (в залежності від діючого законодавства), приймемо *Мм*=6700,00 грн;

K_i – коефіцієнт міжкваліфікаційного співвідношення для встановлення тарифної ставки робітнику відповідного розряду (табл. Б.2, додаток Б)[42];

K_c – мінімальний коефіцієнт співвідношень місячних тарифних ставок робітників першого розряду з нормальними умовами праці виробничих об'єднань і підприємств до законодавчо встановленого розміру мінімальної заробітної плати.

T_p – середнє число робочих днів в місяці, приблизно T_p = 22 дн;

$t_{зм}$ – тривалість зміни, год.

$C_1 = 6700{,}00 \cdot 1{,}10 \cdot 1{,}35 / (22 \cdot 8) = 56{,}53$ *грн.*

$З_{р1} = 56{,}53 \cdot 12{,}00 = 678{,}38$ *грн.*

Таблиця 4.6 – Величина витрат на основну заробітну плату робітників

Найменування робіт	Тривалість роботи, год	Розряд роботи	Тарифний коефіцієнт	Погодинна тарифна ставка, грн	Величина оплати на робітника грн
Установка обладнання	12,00	2	1,10	56,53	678,38
Підготовка робочого місця розробника системи	8,00	3	1,35	69,38	555,03
Інсталяція програмного забезпечення	6,20	4	1,50	77,09	477,95
Формування бази даних	7,50	3	1,35	69,38	520,34
Налагодження програмних блоків	6,00	5	1,70	87,37	524,20
Тестування	11,00	3	1,35	69,38	763,17

програмного забезпечення					
Всього					3519,07

Додаткова заробітна плата дослідників та робітників

Додаткову заробітну плату розраховуємо як 10 … 12% від суми основної заробітної плати дослідників та робітників за формулою:

$$З_{дод} = (З_о + З_р) \cdot \frac{Н_{дод}}{100\%}, \quad (4.7)$$

де $Н_{дод}$ – норма нарахування додаткової заробітної плати. Приймемо 10%.

$З_{дод}$ = (83477,27 + 3519,07) · 10 / 100% = 8699,63 грн.

4.3.2 Відрахування на соціальні заходи

Нарахування на заробітну плату дослідників та робітників розраховуємо як 22% від суми основної та додаткової заробітної плати дослідників і робітників за формулою:

$$З_н = (З_о + З_р + З_{дод}) \cdot \frac{Н_{зп}}{100\%} \quad (4.8)$$

де $Н_{зп}$ – норма нарахування на заробітну плату. Приймаємо 22%.

Зн = (83477,27 + 3519,07 + 8699,63) · 22 / 100% = 21053,12 грн.

4.3.3 Сировина та матеріали

До статті «Сировина та матеріали» належать витрати на сировину, основні та допоміжні матеріали, інструменти, пристрої та інші засоби і предмети праці, які придбані у сторонніх підприємств, установ і організацій та витрачені на проведення досліджень за темою «Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти».

Витрати на матеріали (*М*), у вартісному вираженні розраховуються окремо по кожному виду матеріалів за формулою:

$$M = \sum_{j=1}^{n} H_j \cdot Ц_j \cdot К_j - \sum_{j=1}^{n} B_j \cdot Ц_{вj}, \quad (4.9)$$

де H_j – норма витрат матеріалу j-го найменування, кг;

n – кількість видів матеріалів;

$Ц_j$ – вартість матеріалу j-го найменування, грн/кг;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ = *1,1 ... 1,15*);

B_j – маса відходів j-го найменування, кг;

$Ц_{вj}$ – вартість відходів j-го найменування, грн/кг.

$M_1 = 3 \cdot 193{,}00 \cdot 1{,}02 - 0 \cdot 0 = 590{,}58$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 4.7 – Витрати на матеріали

Найменування матеріалу, марка, тип, сорт	Ціна за 1 кг, грн	Норма витрат, кг	Величина відходів, кг	Ціна відходів, грн/кг	Вартість витраченого матеріалу, грн
Офісний папір 500 80 г\м	193,00	3	0	0	590,58
Папір для записів 100 70 г\м	101,00	3	0	0	309,06
Органайзер офісний	182,00	3	0	0	556,92
Набір офісний (канцелярське приладдя)	221,00	3	0	0	676,26
Картридж для принтера	1230,00	1	0	0	1254,60
Диск оптичний CD-R	28,40	5	0	0	144,84
Flesh-пам'ять 64 GB	239,00	1	0	0	243,78

Тека для паперів	101,00	5	0	0	515,10
Всього					4291,14

4.3.4 Розрахунок витрат на комплектуючі

Витрати на комплектуючі *($К_в$)*, які використовують при проведенні НДР на тему «Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти», розраховуємо, згідно з їхньою номенклатурою, за формулою:

$$К_в = \sum_{j=1}^{n} Н_j \cdot Ц_j \cdot К_j \qquad (4.10)$$

де $Н_j$ – кількість комплектуючих *j*-го виду, шт.;

$Ц_j$ – покупна ціна комплектуючих *j*-го виду, грн;

$К_j$ – коефіцієнт транспортних витрат, ($К_j$ *= 1,1 … 1,15*).

$К_в = 1 \cdot 2729,00 \cdot 1,02 = 2783,58$ грн.

Проведені розрахунки зведемо до таблиці.

Таблиця 4.8 – Витрати на комплектуючі

Найменування комплектуючих	Кількість, шт.	Ціна за штуку, грн	Сума, грн
Зовнішній жорсткий диск 2.5" 1TB Transcend (TS1TSJ25M3S)	1	2729,00	2783,58
Всього			2783,58

4.3.5 Спецустаткування для наукових (експериментальних) робіт

До статті «Спецустаткування для наукових (експериментальних) робіт» належать витрати на виготовлення та придбання спецустаткування необхідного для проведення досліджень, також витрати на їх проектування, виготовлення,

транспортування, монтаж та встановлення.

Балансову вартість спецустаткування розраховуємо за формулою:

$$B_{спец} = \sum_{i=1}^{k} Ц_i \cdot C_{пр.i} \cdot K_i \ , \qquad (4.11)$$

де $Ц_i$ – ціна придбання одиниці спецустаткування даного виду, марки, грн;

$C_{пр.i}$ –кількість одиниць устаткування відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує доставку, монтаж, налагодження устаткування тощо, ($K_i = 1,10…1,12$);

k – кількість найменувань устаткування.

$B_{спец} = 16613,00 \cdot 1 \cdot 1,03 = 17111,39$ *грн.*

Отримані результати зведемо до таблиці:

Таблиця 4.9 – Витрати на придбання спецустаткування по кожному виду

Найменування устаткування	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
Серверна платформа Supermicro CSE-813MFTQC-350CB	1	16613,00	17111,39
Маршрутизатор TP-Link ARCHER-C80	1	1799,00	1852,97
Всього			18964,36

4.3.6 Програмне забезпечення для наукових (експериментальних) робіт

До статті «Програмне забезпечення для наукових (експериментальних) робіт» належать витрати на розробку та придбання спеціальних програмних засобів і програмного забезпечення, (програм, алгоритмів, баз даних) необхідних для проведення досліджень, також витрати на їх проектування, формування та встановлення.

Балансову вартість програмного забезпечення розраховуємо за формулою:

$$B_{прг} = \sum_{i=1}^{k} Ц_{iпрг} \cdot C_{прг.i} \cdot K_i \ , \qquad (4.12)$$

де $Ц_{iпрг}$ – ціна придбання одиниці програмного засобу даного виду, грн;

$C_{прг.i}$ – кількість одиниць програмного забезпечення відповідного найменування, які придбані для проведення досліджень, шт.;

K_i – коефіцієнт, що враховує інсталяцію, налагодження програмного засобу тощо, ($K_i = 1,10…1,12$);

k – кількість найменувань програмних засобів.

$B_{прг} = 19999,00 \cdot 1 \cdot 1,02 = 20398,98$ *грн.*

Отримані результати зведемо до таблиці:

Таблиця 4.10 – Витрати на придбання програмних засобів по кожному виду

Найменування програмного засобу	Кількість, шт	Ціна за одиницю, грн	Вартість, грн
ПЗ для сервера Microsoft Windows Svr Essentials 2019 64Bit English DVD 1-2CPU (G3S-01299)	1	19999,00	20398,98
Прикладне ПЗ Visual Studio Code	1	7690,00	7843,80
Пакет розробки титульної частини nextjs, tailwindcss, material UI, JS	1	10320,00	10526,40
Всього			38769,18

4.3.7 Амортизація обладнання, програмних засобів та приміщень

В спрощеному вигляді амортизаційні відрахування по кожному виду обладнання, приміщень та програмному забезпеченню тощо, розраховуємо з використанням прямолінійного методу амортизації за формулою:

$$A_{обл} = \frac{Ц_б}{T_в} \cdot \frac{t_{вик}}{12}, \quad (4.13)$$

де $Ц_б$ – балансова вартість обладнання, програмних засобів, приміщень тощо, які використовувались для проведення досліджень, грн;

$t_{вик}$ – термін використання обладнання, програмних засобів, приміщень під час досліджень, місяців;

$T_в$ – строк корисного використання обладнання, програмних засобів, приміщень тощо, років.

$A_{обл} = (44599{,}00 \cdot 2) / (2 \cdot 12) = 3716{,}58$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 4.11 – Амортизаційні відрахування по кожному виду обладнання

Найменування обладнання	Балансова вартість, грн	Строк корисного використання, років	Термін використання обладнання, місяців	Амортизаційні відрахування, грн
Персональний комп'ютер розробника ПЗ	44599,00	2	2	3716,58
Персональний комп'ютер інженера-розробника автоматизованих систем управління	29420,00	2	2	2451,67
Робоче місце інженера-програміста	9620,00	5	2	320,67
Серверне	23400,00	4	2	975,00

обладнання				
Пристрої передачі даних	6660,00	4	2	277,50
Оргтехніка	8340,00	5	2	278,00
Приміщення лабораторії розробки ПЗ	417000,00	35	2	1985,71
ОС Windows	6520,00	2	2	543,33
Прикладний пакет Microsoft Office	6510,00	2	2	542,50
Принтер	8350,00	5	2	278,33
Всього				11369,30

4.3.8 Паливо та енергія для науково-виробничих цілей

Витрати на силову електроенергію (B_e) розраховуємо за формулою:

$$B_e = \sum_{i=1}^{n} \frac{W_{yi} \cdot t_i \cdot Ц_e \cdot K_{вni}}{\eta_i}, \quad (4.14)$$

де W_{yi} – встановлена потужність обладнання на визначеному етапі розробки, кВт;

t_i – тривалість роботи обладнання на етапі дослідження, год;

$Ц_e$ – вартість 1 кВт-години електроенергії, грн; (вартість електроенергії визначається за даними енергопостачальної компанії), приймемо $Ц_e$ = 7,50 грн;

$K_{вni}$ – коефіцієнт, що враховує використання потужності, $K_{вni} < 1$;

η_i – коефіцієнт корисної дії обладнання, $\eta_i < 1$.

$B_e = 0,26 \cdot 120,0 \cdot 7,50 \cdot 0,95 / 0,97 = 234,00$ *грн.*

Проведені розрахунки зведемо до таблиці.

Таблиця 4.12 – Витрати на електроенергію

Найменування обладнання	Встановлена потужність, кВт	Тривалість роботи, год	Сума, грн
Серверна платформа Supermicro CSE-813MFTQC-350CB	0,26	120,0	234,00
Маршрутизатор TP-Link ARCHER-C80	0,03	120,0	27,00
Персональний комп'ютер розробника ПЗ	0,42	320,0	1008,00
Персональний комп'ютер інженера-розробника автоматизованих систем управління	0,06	320,0	144,00
Робоче місце інженера-програміста	0,09	320,0	216,00
Серверне обладнання	0,36	120,0	324,00
Пристрої передачі даних	0,10	120,0	90,00
Оргтехніка	0,52	2,0	7,80
Принтер	0,20	3,5	5,25
Всього			2056,05

4.3.9 Службові відрядження

До статті «Службові відрядження» дослідної роботи на тему «Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти» належать витрати на відрядження штатних працівників, працівників організацій, які працюють за договорами цивільно-правового характеру, аспірантів, зайнятих розробленням досліджень, відрядження, пов’язані з проведенням випробувань машин та приладів, а також витрати на відрядження на наукові з’їзди, конференції, наради, пов’язані з виконанням конкретних досліджень.

Витрати за статтею «Службові відрядження» розраховуємо як 20…25% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{св} = (З_о + З_р) \cdot \frac{Н_{св}}{100\%}, \quad (4.15)$$

де $Н_{св}$ – норма нарахування за статтею «Службові відрядження», приймемо $Н_{св}$ = 20%.

$В_{св}$ = (83477,27 + 3519,07) · 20 / 100% = 17399,27 грн.

4.3.10 Витрати на роботи, які виконують сторонні підприємства, установи і організації

Витрати за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації» розраховуємо як 30…45% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{сп} = (З_о + З_р) \cdot \frac{Н_{сп}}{100\%}, \quad (4.16)$$

де $Н_{сп}$ – норма нарахування за статтею «Витрати на роботи, які виконують сторонні підприємства, установи і організації», приймемо $Н_{сп}$= 30%.

$В_{сп}$ = (83477,27 + 3519,07) · 30 / 100% = 26098,90 грн.

4.3.11 Інші витрати

До статті «Інші витрати» належать витрати, які не знайшли відображення у зазначених статтях витрат і можуть бути віднесені безпосередньо на собівартість досліджень за прямими ознаками.

Витрати за статтею «Інші витрати» розраховуємо як 50…100% від суми основної заробітної плати дослідників та робітників за формулою:

$$І_в = (З_о + З_р) \cdot \frac{Н_{ів}}{100\%}, \quad (4.17)$$

де $Н_{ів}$ – норма нарахування за статтею «Інші витрати», приймемо $Н_{ів}$ = 50%.

$І_в$ = (83477,27 + 3519,07) · 50 / 100% = 43498,17 грн.

4.3.12 Накладні (загальновиробничі) витрати

До статті «Накладні (загальновиробничі) витрати» належать: витрати, пов'язані з управлінням організацією; витрати на винахідництво та раціоналізацію; витрати на підготовку (перепідготовку) та навчання кадрів; витрати, пов'язані з набором робочої сили; витрати на оплату послуг банків; витрати, пов'язані з освоєнням виробництва продукції; витрати на науково-технічну інформацію та рекламу та ін.

Витрати за статтею «Накладні (загальновиробничі) витрати» розраховуємо як 100…150% від суми основної заробітної плати дослідників та робітників за формулою:

$$B_{нзв} = (З_о + З_р) \cdot \frac{Н_{нзв}}{100\%}\ , \qquad (4.18)$$

де $Н_{нзв}$ – норма нарахування за статтею «Накладні (загальновиробничі) витрати», приймемо $Н_{нзв}$ = 100%.

$B_{нзв} = (83477,27 + 3519,07) \cdot 100 / 100\% = 86996,34$ *грн.*

Витрати на проведення науково-дослідної роботи на тему «Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти» розраховуємо як суму всіх попередніх статей витрат за формулою:

$$B_{заг} = З_о + З_р + З_{дод} + З_н + М + К_в + B_{спец} + B_{прг} + A_{обл} + B_е + B_{св} + B_{сп} + I_в + B_{нзв}. \qquad (4.19)$$

$B_{заг}$= *83477,27 +3519,07 +8699,63 +21053,12 +4291,14 +2783,58 +18964,36 + 38769,18 + 11369,30 +2056,05 +17399,27 +26098,90 +43498,17 +86996,34 = =368975,39 грн.*

Загальні витрати *ЗВ* на завершення науково-дослідної (науково-технічної) роботи та оформлення її результатів розраховується за формулою:

$$ЗВ = \frac{B_{заг}}{\eta}, \qquad (4.20)$$

де η - коефіцієнт, який характеризує етап (стадію) виконання науково-дослідної роботи, приймемо η=0,95.

ЗВ = 368975,39 / 0,95 = 388395,14 грн.

4.4 Розрахунок економічної ефективності науково-технічної розробки від її впровадження безпосередньо розробником (замовником)

При виконанні даної роботи за темою «Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти» розглядається ситуація, коли замовник певної науково-технічної розробки використовує її тільки на своєму підприємстві (чи в організації) і не виводить її на ринок. У цьому випадку позитивним результатом від впровадження цієї науково-технічної розробки може бути покращення певних економічних та фінансових показників діяльності підприємства.

Для визначення величини майбутнього економічного ефекту та ефективності розробки визначимо певні характеристики підприємства.

Таблиця 4.13 – Вихідні дані замовника

Показник	Рік розробки	1-й рік	2-й рік	3-й рік	4-й рік
Чисельність працівників, які виконують визначені функції вручну, осіб	9	-	-	-	-
Середня заробітна плата працівника, який виконує відповідну функцію вручну, грн	9200,00	-	-	-	-
Приблизні витрати на розробку автоматизованої системи управління, грн	388395,14	-	-	-	-
Економія чисельності працівників виконання виробничої чи управлінської функції яких було автоматизовано у році що аналізується, осіб	-	6	3	0	0
Кількість функцій, які виконуються вручну у році до впровадження результатів нової науково-технічної розробки, шт	66000	-	-	-	-
Прогнозоване зростання кількості виробничих чи інформаційно-	-	44000	22000	0	0

технічних управлінських функцій, виконання яких автоматизується, у році що аналізується (відносно року до впровадження даної розробки), шт					

В даному випадку майбутній економічний ефект та ефективність буде формуватися на основі використання таких показників: $\Delta П_я$ – зростання прибутку підприємства внаслідок зниження витрат на оплату праці працівників, які виконують окремі виробничі чи інформаційно-технічні управлінські функції, грн. Причому $\Delta П_я$ може бути визначено як:

$$\Delta П_я = \frac{ЧП \cdot ЗП \cdot 12}{N} - \frac{(0,2...0,6) \cdot ЗВ}{\Delta N_i}, \qquad (4.21)$$

де $ЧП$ – чисельність працівників, які виконують визначені функції вручну, приймемо 9 осіб; $ЗП$ – середня заробітна плата працівника, який виконує відповідну функцію вручну, приймемо 9200,00 грн; $ЗВ$ – приблизні витрати на розробку автоматизованої системи управління, приймемо 388395,14 грн; N – кількість функцій, які виконуються вручну у році до впровадження результатів нової науково-технічної розробки, 66000 шт; ΔN_i – прогнозоване зростання кількості виробничих чи інформаційно-технічних управлінських функцій, виконання яких автоматизується, у році що аналізується (відносно року до впровадження даної розробки), шт.

Зростання прибутку підприємства в 1-й рік впровадження розробки

$\Delta П_я$ =9·9200,00·12/66000 - 0,45·388395,14/44000 = 11,08 грн/функц.

Зростання прибутку підприємства в 2-й рік впровадження розробки

$\Delta П_я$ =9·9200,00·12/66000 - 0,45·388395,14/66000 = 12,41 грн/функц.

Зростання прибутку підприємства в 3-й рік впровадження розробки

$\Delta П_я$ =9·9200,00·12/66000 - 0,45·388395,14/66000 = 12,41 грн/функц.

Зростання прибутку підприємства в 4-й рік впровадження розробки

$\Delta П_я$ =9·9200,00·12/66000 - 0,45·388395,14/66000 = 12,41 грн/функц.

$П_я$ – прибуток, який отримує підприємство від автоматизації виконання окремої виробничої чи інформаційно-технічної управлінської функції у кожному із років після впровадження науково-технічної розробки, грн. Даний прибуток можна приблизно оцінити виходячи з формули:

$$П_я = \frac{\Delta ЧП \cdot ЗП \cdot 12}{N}, \quad (4.22)$$

де $\Delta ЧП$ – економія чисельності працівників виконання виробничої чи управлінської функції яких було автоматизовано у році що аналізується, осіб;

Прибуток який отримує підприємство від автоматизації функції в 1-й рік $П_я$ =6·9200,00·12/66000 = 10,03636364 грн/функц.

Прибуток який отримує підприємство від автоматизації функції в 2-й рік $П_я$ =3·9200,00·12/66000 = 5,018181818 грн/функц.

Прибуток який отримує підприємство від автоматизації функції в 3-й рік $П_я$ =0·9200,00·12/66000 = 0 грн/функц.

Прибуток який отримує підприємство від автоматизації функції в 4-й рік $П_я$ =0·9200,00·12/66000 = 0 грн/функц.

Збільшення чистого прибутку підприємства $\Delta П_i$ для кожного із років, протягом яких очікується отримання позитивних результатів від можливого впровадження науково-технічної розробки, розраховуємо за формулою:

$$\Delta П_i = (\Delta П_я \cdot N + П_я \cdot \Delta N)_i, \quad (4.23)$$

де $\Delta П_я$ – покращення основного якісного показника від впровадження на підприємстві результатів науково-технічної розробки у році що аналізується;

N – основний кількісний показник, який визначає обсяг діяльності підприємства у році до впровадження результатів нової науково-технічної

розробки;

$П_я$ – основний якісний показник, який визначає результати діяльності підприємства у кожному із років після впровадження науково-технічної розробки;

ΔN – зміна основного кількісного показника діяльності підприємства в результаті впровадження науково-технічної розробки у році що аналізується.

Збільшення чистого прибутку підприємства в 1-й рік впровадження

$\Delta П_i$=11,08·66000+10,03636364·44000=1173033,28 грн.

Збільшення чистого прибутку підприємства в 2-й рік впровадження

$\Delta П_i$=12,41·66000+5,018181818·(44000+22000)=1150022,19 грн.

Збільшення чистого прибутку підприємства в 3-й рік впровадження

$\Delta П_i$=12,41·66000+0·(44000+22000+0)=818822,19 грн.

Збільшення чистого прибутку підприємства в 4-й рік впровадження

$\Delta П_i$=12,41·66000+0·(44000+22000+0+0)=818822,19 грн.

Приведена вартість збільшення всіх чистих прибутків *ПП*, що їх може отримати замовник від можливого впровадження та комерціалізації науково-технічної розробки:

$$ПП = \sum_{i=1}^{T} \frac{\Delta П_i}{(1+\tau)^t}, \qquad (4.24)$$

де $\Delta П_i$ – збільшення чистого прибутку у кожному з років, протягом яких виявляються результати впровадження науково-технічної розробки, грн;

T – період часу, протягом якого очікується отримання позитивних результатів від впровадження та комерціалізації науково-технічної розробки, роки;

τ – ставка дисконтування, за яку можна взяти щорічний прогнозований рівень інфляції в країні, τ=0,13;

t – період часу (в роках) від моменту початку впровадження науково-технічної розробки до моменту отримання замовником додаткових чистих прибутків у цьому році.

$ПП$=1173033,28/(1+0,13)^1+1150022,19/(1+0,13)^2+818822,19/(1+0,13)^3+

$+818822,19/(1+0,13)^4$=1038082,55+900636,06+567484,85+502198,98= = 3008402,44 грн.

Величина початкових інвестицій PV, які замовник має вкласти для впровадження і комерціалізації науково-технічної розробки:

$$PV = k_{інв} \cdot ЗВ, \tag{4.25}$$

де $k_{інв}$ – коефіцієнт, що враховує витрати інвестора на впровадження науково-технічної розробки та її комерціалізацію, приймаємо $k_{інв}$=2;

$ЗВ$ – загальні витрати на проведення науково-технічної розробки та оформлення її результатів, приймаємо 388395,14 грн.

$PV = k_{інв} \cdot ЗВ$ = 2 · 388395,14 = 776790,29 грн.

Абсолютний економічний ефект $E_{абс}$ для розробника від можливого впровадження науково-технічної розробки становитиме:

$$E_{абс} = ПП - PV \tag{4.26}$$

де $ПП$ – приведена вартість зростання всіх чистих прибутків від можливого впровадження та комерціалізації науково-технічної розробки, 3008402,44 грн;

PV – теперішня вартість початкових інвестицій, 776790,29 грн.

$E_{абс} = ПП - PV$ = 3008402,44 - 776790,29 = 2231612,15 грн.

Внутрішня економічна дохідність інвестицій $E_в$, які можуть бути вкладені розробником у впровадження та комерціалізацію науково-технічної розробки:

$$E_в = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1, \tag{4.27}$$

де $E_{абс}$ – абсолютний економічний ефект вкладених інвестицій, 2231612,15 грн;

PV – теперішня вартість початкових інвестицій, 776790,29 грн;

$T_{ж}$ – життєвий цикл науково-технічної розробки, тобто час від початку її розробки до закінчення отримування позитивних результатів від її впровадження, 4 роки.

$$E_{в} = \sqrt[Tж]{1 + \frac{E_{абс}}{PV}} - 1 = (1+2231612,15/776790,29)^{1/4} = 0,403.$$

Мінімальна внутрішня економічна дохідність вкладених інвестицій $\tau_{мін}$:

$$\tau_{мін} = d + f, \tag{4.28}$$

де d – середньозважена ставка за депозитними операціями в комерційних банках; в 2021 році в Україні d =0,1;

f – показник, що характеризує ризикованість вкладення інвестицій, приймемо 0,22.

$\tau_{мін}$ = 0,1+0,22 = 0,32 < 0,403 свідчить про те, що внутрішня економічна дохідність інвестицій $E_{в}$, які можуть бути вкладені розробником у впровадження та комерціалізацію науково-технічної розробки вища мінімальної внутрішньої дохідності. Тобто інвестувати в науково-дослідну роботу за темою «Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти» доцільно.

Період окупності інвестицій $T_{ок}$ які можуть бути вкладені розробником у впровадження та комерціалізацію науково-технічної розробки:

$$T_{ок} = \frac{1}{E_{в}}, \tag{4.29}$$

де $E_{в}$ – внутрішня економічна дохідність вкладених інвестицій.

$T_{ок}$ = 1 / 0,403 = 2,48 р.

$T_{ок}$ < 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати розробника профінансувати впровадження даної розробки для застосування в діяльності підприємства.

Висновки до розділу

Згідно проведених досліджень рівень комерційного потенціалу розробки за темою «Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти» становить 32,7 бала, що свідчить про комерційну важливість проведення даних досліджень (рівень комерційного потенціалу розробки вище середнього).

При оцінюванні за технічними параметрами, згідно узагальненого коефіцієнту якості розробки, науково-технічна розробка переважає існуючі аналоги приблизно в 1,71 рази.

Також термін окупності становить 2,48 р., що менше 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати розробника до впровадження даної розробки при отриманні ефекту в розмірі 2231612,15 грн.

Отже можна зробити висновок про доцільність проведення науково-дослідної роботи за темою «Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти».

ВИСНОВКИ

В результаті проведеного аналізу різних систем та технологій, спрямованих на створення сучасних веб-сайтів, виявлено, що кожна з розглянутих систем має свої унікальні переваги та недоліки. Це дозволяє зробити висновок, що вибір конкретної технології повинен бути обґрунтованим і залежати від вимог проекту та власних уподобань розробника.

У першому розділі, присвяченому проблемі відвідуваності у школах, виокремлені ключові аспекти, що демонструють актуальність створення автоматизованої системи управління відвідуваністю.

Другий розділ надає обґрунтування з різнобіччям різних технологій frontend частини, що дало широкий вибір та розуміння принципів роботи кожної технології. Вибір технології для реалізації цього проекту обгрунтовується різнобічними вимогами та особливостями самого проекту, були вивчені та вибрані технології для побудови інтерфейсів користувача та логіки веб-додатку. Аналіз і вибір конкретних технологій забезпечили раціональний підхід до створення системи.

У третьому розділі були розглянуті інструменти для розробки, було розроблено план реалізації проекту, огляд додатку, розробку користувацьких інтерфейсів та створення системи керування вмістом. Детальний огляд функціоналу та ролей у проекту сприяє глибшому розумінню важливості забезпечення повноцінної та зручної взаємодії з користувачем, що є критичним для успішної реалізації будь-якого проекту. Результатом цього розділу є створена ефективна система, яка відповідає визначеним вимогам.

Четвертий розділ присвячений економічній частині, становить важливий етап у забезпеченні повного розуміння не лише наукових та технічних аспектів роботи, а й її потенційного впливу на бізнесове середовище. Прогнозування витрат та економічне обґрунтування проведеної роботи підкреслюють значущість проекту з економічної точки зору. Результати дозволяють зробити висновок про доцільність інвестицій у впровадження розробленої системи, враховуючи її потенціал для підвищення ефективності управління відвідуваністю та

забезпечення конкурентноспроможності приватного навчального закладу.

У цілому, магістерська робота виконана з урахуванням сучасних тенденцій у розробці веб-додатків та висловлює глибокі рефлексії, що дозволяють зробити вагомий внесок у практичний аспект вибору технологій для веб-проектів. Загальний висновок магістерської роботи відображає важливість розгляду роботи з комплексного погляду, ураховуючи наукові, технічні та економічні аспекти. Такий інтегрований підхід дозволяє визначити потенціал розробленої системи та створює підґрунтя для її подальшого вдосконалення та впровадження у практику.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Освіторія. Мотивація під час навчання. [Електронний ресурс] – Режим доступу: https://osvitoria.media/experience/motyvatsiya-pid-chas-dystantsijnogo-navchannya-misiya-mozhlyva/
2. Stud.com. Автоматизовані системи управління. [Електронний ресурс] – Режим доступу: https://stud.com.ua/21186/informatika/ponyattya_avtomatizovani_sistemi_upravlinnya
3. Avada Media. Розробка LMS систем управління навчанням. [Електронний ресурс] – Режим доступу: https://avada-media.ua/ua/services/lms/
4. Moodle. Що таке Moodle. [Електронний ресурс] – Режим доступу: https://moodle.org/mod/page/view.php?id=8174
5. Future now. Що таке Google Classroom. [Електронний ресурс] – Режим доступу: https://futurenow.com.ua/shho-take-google-classroom-yak-neyu-korystuvatysya-yak-pratsyuye-ta-osnovni-mozhlyvosti/
6. Canvas. Canvas LMS для шкіл. [Електронний ресурс] – Режим доступу: https://products.containerize.com/ru/lms/canvas/
7. Wezom. Що таке UI і як він впливає на користувачів. [Електронний ресурс] – Режим доступу: https://wezom.com.ua/ua/blog/chto-takoe-ui-i-kak-polzovatelskij-interfejs-vliyaet-na-prodazhi-internet-magazina
8. Redstone. Що таке UX/UI дизайн. [Електронний ресурс] – Режим доступу: https://wezom.com.ua/ua/blog/chto-takoe-ui-i-kak-polzovatelskij-interfejs-vliyaet-na-prodazhi-internet-magazina
9. Vercel. Next.js Brand Guidelines. [Електронний ресурс] – Режим доступу: https://vercel.com/design/brands#next-js
10. DOU. Генерації сторінок SSR SSG. [Електронний ресурс] – Режим доступу: https://dou.ua/forums/topic/41585/
11. AG Marketing. Що таке SEO і чому це важливо. [Електронний ресурс] – Режим доступу: https://ag.marketing/blog/shcho-take-seo/

12. DOU. Переваги та недоліки NextJS. [Електронний ресурс] – Режим доступу: https://dou.ua/forums/topic/44219/
13. itProger. Вивчаємо TailwindCSS. [Електронний ресурс] – Режим доступу: https://itproger.com/ua/news/izuchenie-tailwind-css-za-chas-razrabotka-proekta-s-nulya
14. Продизайн. Які реальні заслуги у Material UI. [Електронний ресурс] – Режим доступу: https://prodesign.in.ua/2015/11/yaki-realni-zasluhy-u-material-design/
15. DevZone. UI-фреймворки та бібліотеки JavaScript. [Електронний ресурс] – Режим доступу: https://devzone.org.ua/post/ui-freimvorki-ta-biblioteki-javascript
16. UkLegacy. Посібник “Знайомство з React”. [Електронний ресурс] – Режим доступу: https://uk.legacy.reactjs.org/tutorial/tutorial.html
17. Wezom. Стаття “Vue vs React у 2022 році - який фреймворк вибрати і коли?”. [Електронний ресурс] – Режим доступу: https://wezom.com.ua/ua/blog/vue-vs-react
18. Brander. Що таке Nuxt.JS і Vue.JS і коли їх вигідно використовувати . [Електронний ресурс] – Режим доступу: https://brander.ua/technologies/nuxtjs
19. Apeirondb. “IT технології Angular”. [Електронний ресурс] – Режим доступу: https://apeirondb.com/ua/blog/it-technologies-angular
20. Webmaestro. Стаття “SEO + Angular = friends”. [Електронний ресурс] – Режим доступу: https://webmaestro.com.ua/ua/blog/seo-angular/
21. DOUСтаття “Svelte – нова надійність”. [Електронний ресурс] – Режим доступу: https://dou.ua/forums/topic/42867/
22. Wikipedia. Bootstrap. [Електронний ресурс] – Режим доступу: https://uk.wikipedia.org/wiki/Bootstrap
23. Romul name. Bulma сучасний CSS фреймворк. [Електронний ресурс] – Режим доступу: https://romul.name/blog/bulma-suchasnyj-css-frejmvork/#
24. HTML plus. Фреймворки каскадних таблиць стилів. [Електронний ресурс] – Режим доступу: https://html-plus.in.ua/css-frameworks/#framework-materialize
25. HTML plus. Semantic UI. [Електронний ресурс] – Режим доступу: https://html-plus.in.ua/css-frameworks/#semantic-ui

26. HTML plus. Foundation. [Електронний ресурс] – Режим доступу: https://html-plus.in.ua/css-frameworks/#framework-foundation
27. Офіційна документація Ant Design. [Електронний ресурс] – Режим доступу: https://ant.design/docs/spec/introduce
28. Hashdork. Посібник Chakra UI. [Електронний ресурс] – Режим доступу: https://hashdork.com/uk/chakra-ui
29. React. Semantic UI React. [Електронний ресурс] – Режим доступу: https://react.semantic-ui.com/
30. Gamedev. Blueprint. [Електронний ресурс] – Режим доступу: https://gamedev.dou.ua/blogs/visual-programming-blueprints/
31. Ikraine. EverGreen або Вічнозелений контент. [Електронний ресурс] – Режим доступу: https://ikraine.net/evergreen-ili-vechnozelenyj-kontent-kak-sozdat-materialy-kotorye-budut-populyarny-vsegda/
32. Buh24. Що таке backend. [Електронний ресурс] – Режим доступу: https://www.buh24.com.ua/shho-take-backend-rozrobka-i-chym-vona-vidriznyayetsya-vid-frontend/
33. Bluepick. Strapi - аналіз і порівняння. [Електронний ресурс] – Режим доступу: https://www.bluepick.de/cms/strapi
34. Офіційна документація Strapi. [Електронний ресурс] – Режим доступу: https://docs.strapi.io/dev-docs/intro
35. Стаття “Найкращі headless CMS” . [Електронний ресурс] – Режим доступу: https://dou.ua/forums/topic/43307/
36. Офіційна документація Contentful. Managing Content. [Електронний ресурс] – Режим доступу: https://www.contentful.com/help/content-tab/
37. UAspectr. GraphCMS. [Електронний ресурс] – Режим доступу: https://uaspectr.com/2021/10/05/top-7-headless-cms/
38. Wikipedia. Kentico Kontent Wiki. [Електронний ресурс] – Режим доступу: https://ru.wikipedia.org/wiki/Kentico_CMS
39. Офіційна документація Directus. [Електронний ресурс] – Режим доступу: https://docs.directus.io/

40. Офіційна документаці Prismic. [Електронний ресурс] – Режим доступу: https://prismic.io/developers
41. Academind. "Strapi: The Complete Guide" (Відеокурс). [Електронний ресурс] – Режим доступу: https://www.youtube.com/watch?v=6FnwAbd2SDY
42. Методичні вказівки до виконання економічної частини магістерських кваліфікаційних робіт / Уклад. : В. О. Козловський, О. Й. Лесько, В. В. Кавецький. – Вінниця : ВНТУ, 2021. – 42 с.
43. Кавецький В. В**.** Економічне обґрунтування інноваційних рішень: практикум / В. В. Кавецький, В. О. Козловський, І. В. Причепа – Вінниця : ВНТУ, 2016. – 113 с.

ДОДАТКИ

Додаток А

(обов'язковий)

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: **« Розробка автоматизованиї системи управління відвідуваністю для приватного закладу шкільної освіти»**

Тип роботи: Магістерська кваліфікаційна робота
(БДР, МКР)

Підрозділ КСУ, ФІІТА
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 87,3% Схожість 12,7%

Аналіз звіту подібності (відмітити потрібне)

ᵞ 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 розгляд експертної комісії кафедри.

□ 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.

Особа, відповідальна за перевірку _____ Володимир ДУБОВОЙ
(підпис) (прізвище, ініціали)

Ознайомлені з повним звітом подібності, який був згенерований системою Unichek щодо роботи.

Автор роботи _____ Володимир МАЗУР
(підпис) (прізвище, ініціали)

Керівник роботи _____ Марія ЮХИМЧУК
(підпис) (прізвище, ініціали)

Додаток Б

(обов'язковий)

ВНТУ

ЗАТВЕРДЖЕНО
Т.в.о. зав. кафедри КСУ ВНТУ,
д.т.н., доцент.
_____Марія ЮХИМЧУК
" 6 " жовтня 2023 р.

ТЕХНІЧНЕ ЗАВДАННЯ

на виконання магістерської кваліфікаційної роботи

"РОЗРОБКА АВТОМАТИЗОВАНОЇ СИСТЕМИ УПРАВЛІННЯ ВІДВІДУВАНІСТЮ ДЛЯ ПРИВАТНОГО ЗАКЛАДУ ШКІЛЬНОЇ ОСВІТИ"

08-33.МКР.15.00.000 ТЗ
номер

Студент групи 2АКІТ-22м

Підпис *Володимир МАЗУР*
Ім'я ПРІЗВИЩЕ

Керівник д.т.н., доцент, т.в.о.зав. кафедри КСУ

Підпис *Марія ЮХИМЧУК*
Ім'я ПРІЗВИЩЕ

Вінниця 2023

1. Назва та галузь застосування

1.1. Назва – Автоматизована система управління відвідуваністю в школах

1.2. Галузь застосування – сфера інформаційних технологій

2. Підстава для проведення розробки.

Тема магістерської кваліфікаційної роботи затверджена наказом по ВНТУ від №247 від 18-09-2023р.

3. Мета та призначення розробки.

Мета розробки автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти полягає в створенні інноваційного інструменту, який сприятиме поліпшенню організації навчального процесу та забезпеченню ефективного контролю за присутністю учнів.

4. Джерела розробки.

Магістерська кваліфікаційна робота виконується вперше. В ході проведення розробки повинні використовуватись такі документи:

1. Vercel. Next.js Brand Guidelines. [Електронний ресурс] – Режим доступу: https://vercel.com/design/brands#next-js
2. DOU. Генерації сторінок SSR SSG. [Електронний ресурс] – Режим доступу: https://dou.ua/forums/topic/41585/
3. AG marketing. Що таке SEO і чому це важливо. [Електронний ресурс] – Режим доступу: https://ag.marketing/blog/shcho-take-seo/

5. Вимоги до розробки.

5.1. Перелік головних функцій:

- керування базою даних;
- додавання, видалення, зміна інформації про учнів;
- можливість для батьків слідкувати за відвідуваністю.

5.2. Основні технічні вимоги до розробки.

5.2.1. Вимоги до програмної платформи:

- Google Chrome 101.
- Доступ до мережі інтернет
- Акаунт Gmail

5.2.2. Умови експлуатації системи:

- робота на веб-додатку;
- можливість цілодобового функціонування системи;
- робота на ПК;
- дані оновлюються і є актуальними.

6. Стадії та етапи розробки.

6.1 Пояснювальна записка:

- Аналіз технічного завдання «11» вересня 2023 р.
- Аналіз проблеми та інформаційний опис. «13» вересня 2023 р.
- Постановка задач дослідження «14»вересня 2023 р.
- Аналіз технічного завдання «15» вересня 2023 р.
- Аналіз сучаних технологій та вибір стеку «19» вересня 2023 р.
- Проектування архітектури проекту та користувацького інтерфейсу «23» вересня 2023 р.

6.2 Графічні матеріали:

- Створення структури ПЗ «24» вересня 2023 р.
- Опис інтерфейсу «2» жовтня 2023 р.

7. Порядок контролю і приймання.

7.1. Хід виконання роботи контролюється керівником роботи. Рубіжний контроль провести до «28»__11_ 2023 р.

7.2. Атестація МКР здійснюється на попередньому захисті. Попередній захист магістерської кваліфікаційної роботи провести до «11__12_ 2023 р.

7.3. Підсумкове рішення щодо оцінки якості виконання роботи приймається на засіданні ЕК. Захист магістерської кваліфікаційної роботи провести до «20»__12_ 2023 р.

Додаток В

(довідниковий)

Лістинг файлу авторизації “auth.js”

```
import Cookies from 'js-cookie';
import { authMeId, authMeUsername } from './actions/authorisation';

export const setToken = (data) => {
  if (typeof window === 'undefined') {
    return;
  }
  Cookies.set('id', data.user.id);
  Cookies.set('username', data.user.username);
  Cookies.set('jwt', data.jwt);
  if (Cookies.get('username') && Cookies.get('id') && Cookies.get('jwt')) {
    return true;
  }
};

export const unsetToken = () => {
  if (typeof window === 'undefined') {
    return;
  }
  Cookies.remove('id');
  Cookies.remove('username');
  Cookies.remove('jwt');
  return true;
};
export const getUserFromLocalCookie = async () => {
  const jwt = getTokenFromLocalCookie();

  if (jwt) {
    const response = await authMeUsername(jwt);
    return await response;
  } else {
    return;
```

```
  }
};
export const getIdFromLocalCookie = async () => {
  const jwt = getTokenFromLocalCookie();

  if (jwt) {
    return await authMeId(jwt);
  } else {
    return;
  }
};
export const getTokenFromLocalCookie = () => {
  return Cookies.get('jwt');
};
export const getTokenFromServerCookie = (req) => {
  if (!req.headers.cookie || '') {
    return undefined;
  }
  const jwtCookie = req.headers.cookie.split(';').find((c) => c.trim().startsWith('jwt='));
  if (!jwtCookie) {
    return undefined;
  }
  const jwt = jwtCookie.split('=')[1];
  return jwt;
};
export const getIdFromServerCookie = (req) => {
  if (!req.headers.cookie || '') {
    return undefined;
  }
  const idCookie = req.headers.cookie.split(';').find((c) => c.trim().startsWith('id='));
  if (!idCookie) {
    return undefined;
  }
  const id = idCookie.split('=')[1];
  return id;
};
```

Додаток В

(довідниковий)

Лістинг файлу авторизації “authProvider.js”

```
"use client"

import { createContext, useContext, useEffect, useState } from "react";
import { getUserFromLocalCookie } from "./auth";

let userState;

const User = createContext({user: null, loading: false});

export const UserProvider = ({value, children}) => {
   const {user} = value;
   useEffect(() => {
      if(!userState && user) {
         userState = user
      }
   }, [])
   return <User.Provider value={value}>{children}</User.Provider>;
}

export const useUser = () => useContext(User);

export const useFetchUser = () => {
   const [data, setUser] = useState({
      user: userState || null,
      loading: userState === undefined
   })
   useEffect(() => {
      if(userState !== undefined) {
         return
      }
      let isMounted = true;
```

```
    const resolveUser = async () => {
      const user = await getUserFromLocalCookie();
      if(isMounted) {
        setUser({user, loading: false})
      }
    }
    resolveUser();
    return () => {
      isMounted = false
    }
  }, [])
  return data;
}
```

Додатог Г

(обов'язковий)

ІЛЮСТРАТИВНА ЧАСТИНА

Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти

(тема)

1. Актуальність роботи
2. Мета, об'єкт та предмет дослідження
3. Задачі дослідження
4. Вибір стеку технологій
5. Послідовність розробки
6. UML – діаграма варіантів використання
7. Зовнішній вигляд розробленого додатку
8. Висновки

Студент групи 2АКІТ-22м

Підпис Володимир МАЗУР
Ім'я ПРІЗВИЩЕ

Керівник д.т.н., доцент, т.в.о.зав. кафедри КСУ

Підпис Марія ЮХИМЧУК
Ім'я ПРІЗВИЩЕ

1

Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти

Магістерська кваліфікаційна робота
Студента групи 2АКІТ-22м Мазура Володимира Юрійовича

Керівник роботи: Юхимчук Марія Сергіївна
д.т.н., доцент, т.в.о.зав. кафедри КСУ

2

Актуальність роботи

- Автоматизована система сприяє точному фіксуванню відвідувань учнів, що надає можливість вчителям та адміністрації швидко та ефективно контролювати присутність. Зокрема, це полегшує вирішення питань звітності, а також надає інструменти для аналізу даних, спрямованих на поліпшення якості освіти та прийняття обґрунтованих управлінських рішень. Крім того, така система відповідає сучасним вимогам освітнього процесу, сприяючи використанню інноваційних технологій у сфері освіти. Це може позитивно позначитися на іміджі закладу та забезпечити йому конкурентоспроможність у сучасному освітньому середовищі України.

Мета, об'єкт та предмет дослідження

3

- Мета дослідження:

Розробка автоматизованої системи управління відвідуваністю для шкільного закладу, яка полягає у створенні інструменту який сприятиме покращенню організації навчального процесу та заебезпечить ефективний контроль за присутністю учнів.

- Об'єкт дослідження:

Є система управління контролю відвідуваністю для шкільного закладу

- Предмет дослідження:

Є ефективніть використання системи управління відвідуваністю для підвищення якості освіти.

Задачі дослідження

4

- Вивчення різних систем та технологій для розробки веб-сайтів, аналіз їхніх переваг і недоліків у контексті конкретного проекту.
- Встановлення конкретних вимог та особливостей автоматизованої системи управління відвідуваністю школи.
- Визначення ключових аспектів, які підкреслюють важливість впровадження автоматизованої системи для вирішення проблеми відвідуваності в освітніх закладах.
- Аналіз економічної доцільності та конкурентоспроможності

Вибір стеку технологій

5

- Середовище розробки: Обрано Visual Studio Code який є потужним текстовим редактором для розробників, він є безкоштовним, має велику мовну розширюваність, легкість користування, зручний інтерфейс, підтримує роботу з git, є кросплатформеність що дає змогу переключатись між Windows, macOS, Linux якщо розробка йде на різник комп'ютерах
- Мова програмування: Обрано JS так як вона найпопулярніша і найкраще підходить для браузерів і має ряд переваг (швидкість виконання, робота з фреймворками та бібліотеками, підтримка для різних платформ)
- Вибір фреймворку: Обрано Next JS бо він чудово працює з JS, має суттєві переваги, а саме рендеринг на стороні сервера (SSR) та рендеринг на стороні клієнта (CSR) це суттєва перевага яка допомагає оптимізувати додаток як по навантеженю системи на слабких пристроях так і SEO, за допомогою CSR дана технологія дозволить створити інтерактивні веб-сторінки оскільки частина логіки обробляється на стороні клієнта, без перезавантаження всієї сторінки.
- Система управління БД: Обрано cms Strapi яка має хорошу швидкодію,безпеку, простота у використанні, безкоштовна, добре масштабується від невеликих особистих сайтів до складних підприємницьких рішень, також Strapi є headless CMS, що означає, що він забезпечує лише API для управління контентом. Це дає велику свободу у виборі технологій та фреймворків для фронтенду.
- Розробка інтерфейсу: Обрано Material UI це бібліотека компонентів для React, яка реалізує дизайн інтерфейсу і має такі переваги: готові компоненти, що в свою чергу не потребує від розробника приділяти багато часу на CSS код, кастомізація за допомогою якої можна легко змінити колір, шрифт та інші стилі, дана бібліотека повністю інтегрована з React що задовільняє вимоги до стеку, також має активну спільноту розробників. Наступним інструментом обрано Tailwind CSS який працює зі стилями у веб-розробці, його головна перевага це швидкість та зручність розробки.

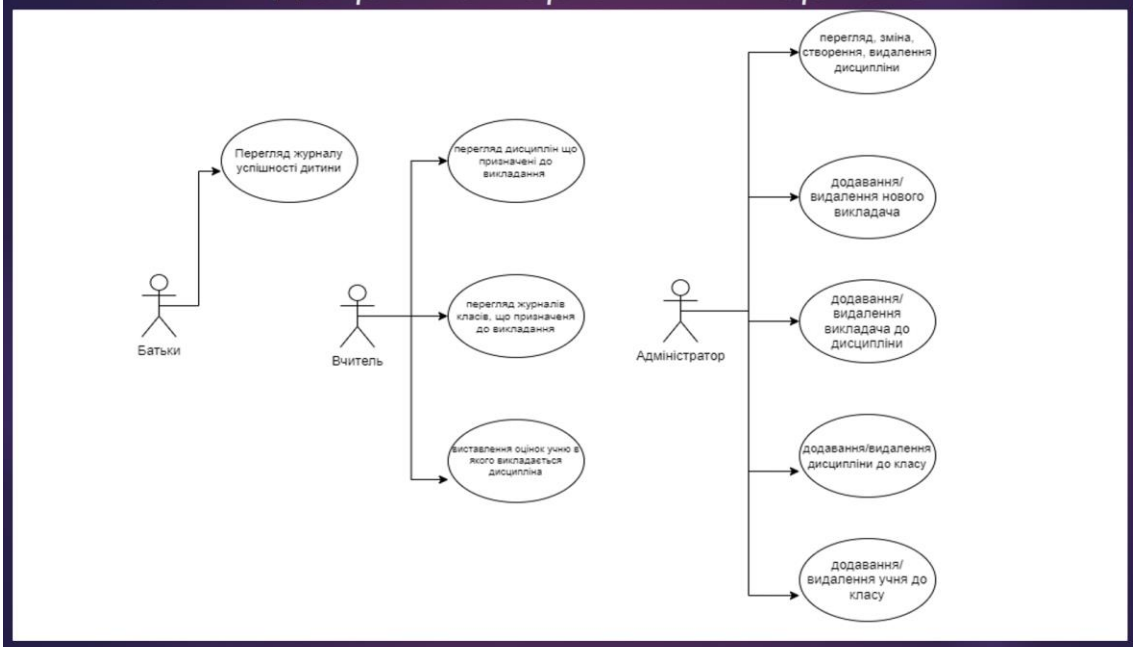
Послідовність розробки

6

1) Налаштування середовища розробки VS Code
2) Ініціалізація проетк NextJS
3) Розробка клієнтських компонент і сторінок без даних, що надходять з серверу (розробка інтерфейсу користувача)
4) Ініціалізація системи керування вмістом Strapi
5) Створення типів колекцій для адміністратора, викладачів та батьків
6) Заповнення колекцій тестовими даними
7) Встановлення взаємодії front-end частини з back-end частиною
8) Відображення даних в front-end частину що отримані з серверу smc Strapi
9) Впровадження процесу авторизації
10) Тестування системи

UML - діаграма варіантів використання

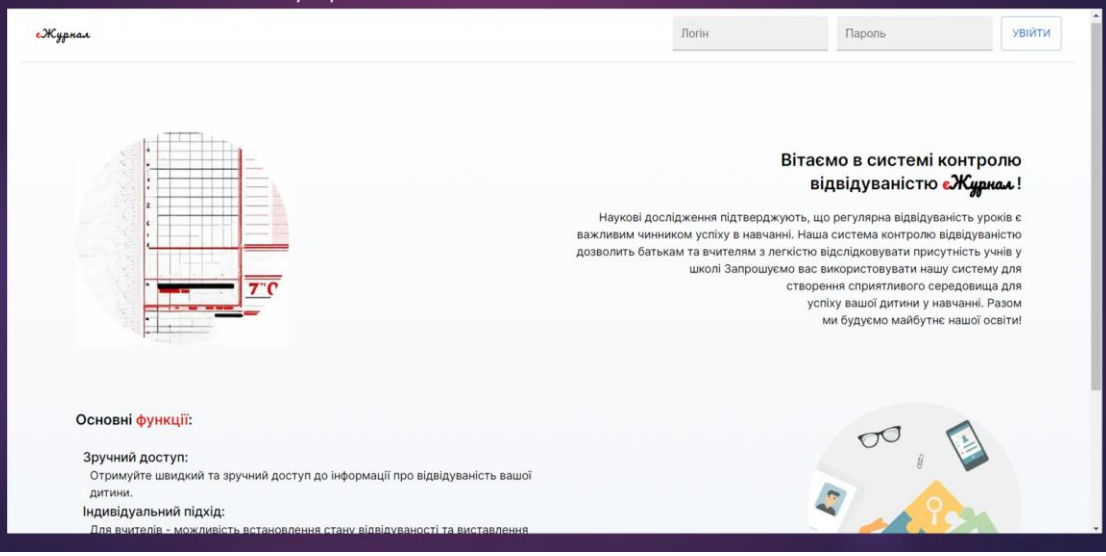
7



Зовнішній вигляд розробленого додатку

8

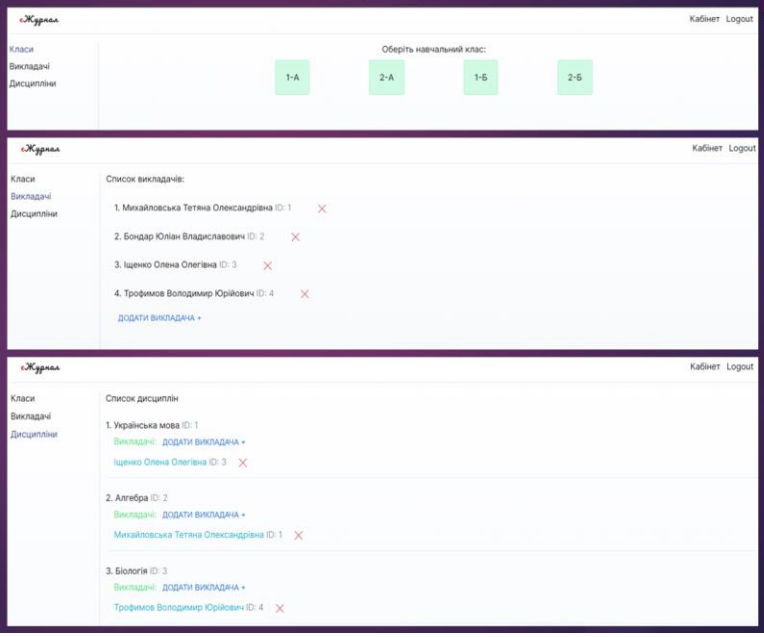
Головний вигляд додатку при вході



Головне вікно адміністратора при вході

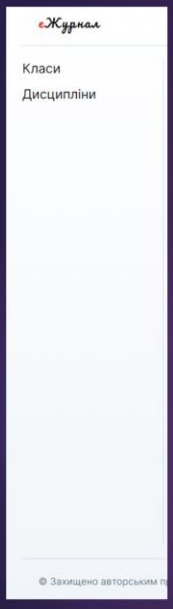
9

Вигляд бічної панелі адміністратора

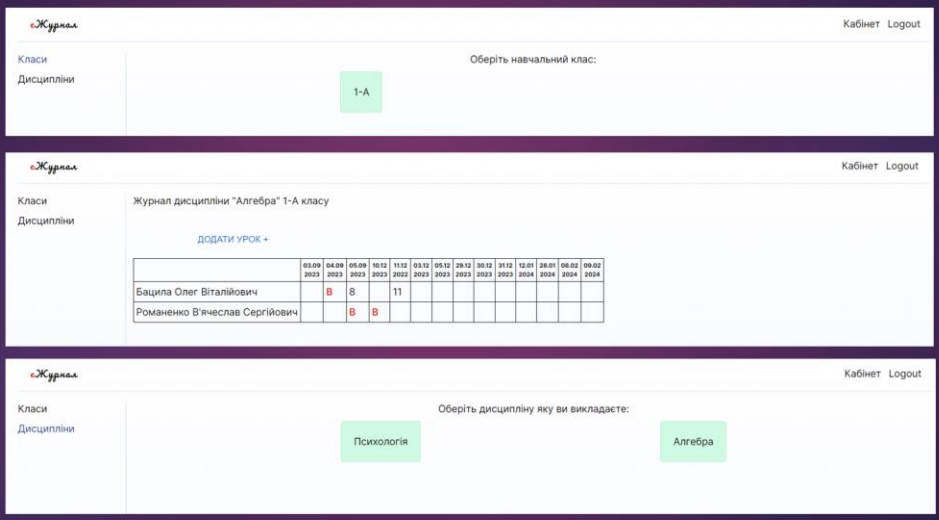


Головне вікно при вході вчителів

10

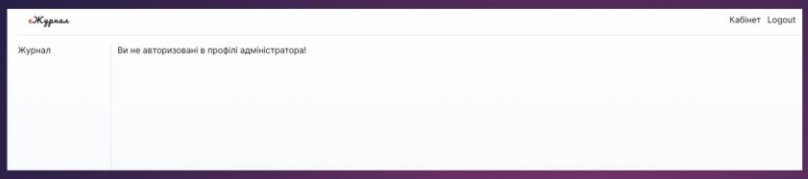


Бічна панель при натисканні на обраний список

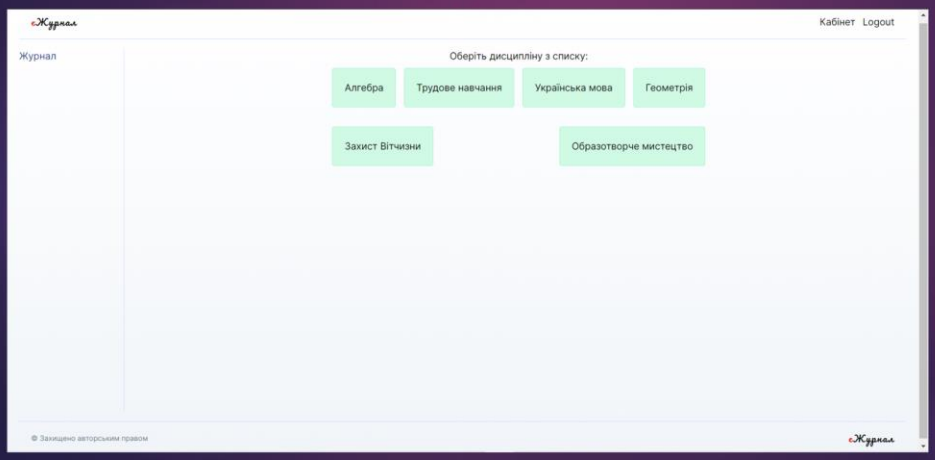


Головне вікно при вході бітьків

11



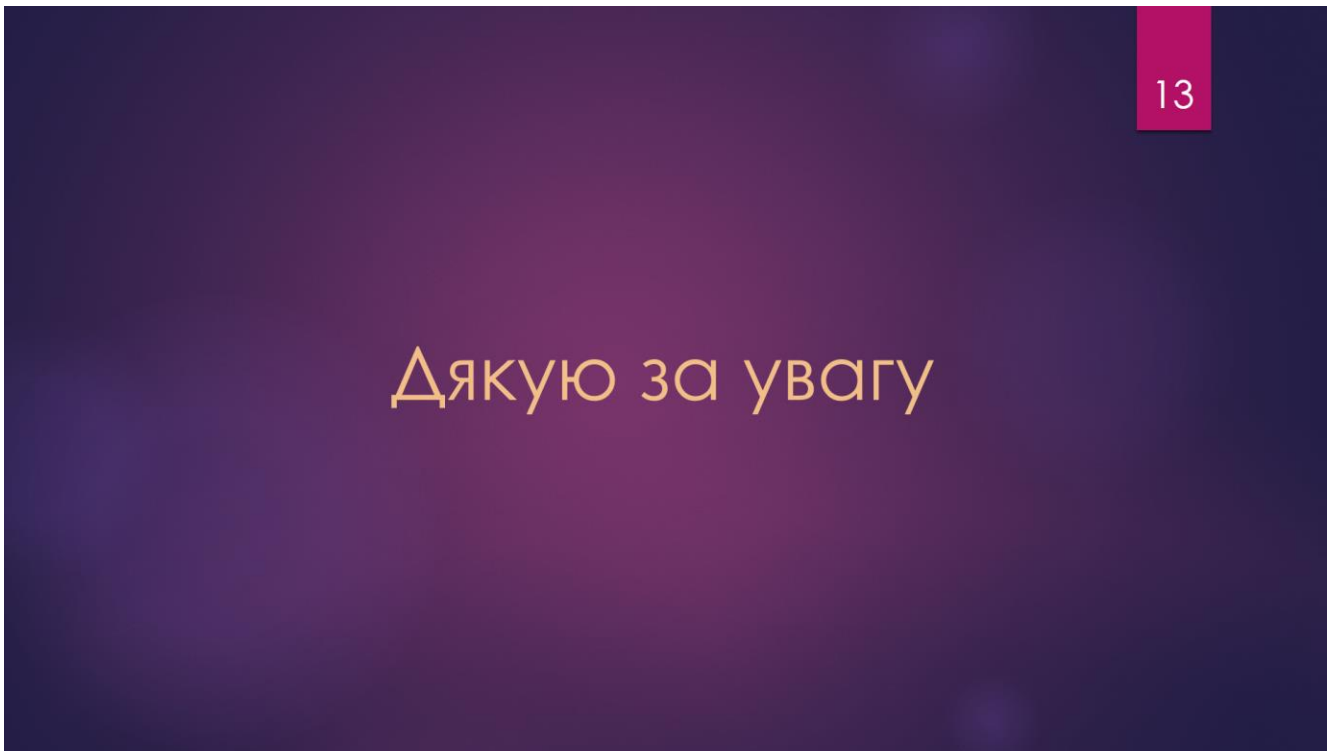
Бічна панель при натисканні на “Журнал”



Висновки

12

- Магістерська робота “Розробка автоматизованої системи управління відвідуваністю для приватного закладу шкільної освіти” вирішує актуальну проблему звітності, налагодження швидкодії робочого процесу.
- Було оглянуто усі сучасні пропозиції для розробки від конкурентів.
- Процес розробки включав вибір технологічного стеку, створення клієнтського інтерфейсу, створення back-end частини за допомогою cms Strapi.
- Проаналізовано розробку з економічної частини а саме розробка переважає існуючі аналоги приблизно в 1,71 рази, також термін окупності становить 2,48 р., що менше 3-х років, що свідчить про комерційну привабливість науково-технічної розробки і може спонукати розробника до впровадження даної розробки при отриманні ефекту в розмірі 2231612,15 грн.
- Розробка в подальшому також передбачається, а саме розширений функціонал системи, збільшення можливостей для роботи тих чи інших функцій для різних ролей (одніє з таких може бути створення можливусті ставити відгуки для учнів, щоб їх бачили батьки).


13
Дякую за увагу