Вінницький національний технічний університет
Факультет інформаційних електронних систем
Кафедра інфокомунікаційних систем і технологій

Бакалаврська дипломна робота на тему:

ТЕЛЕГРАМ-БОТ ДЛЯ ВИВЧЕННЯ ОБ'ЄКТНО-ОРІЄНТОВАНОЇ МОВИ ПРОГРАМУВАННЯ НА БАЗІ ПРОТОКОЛУ HTTP

Виконав: студент 4-го курсу, групи ПЗТ-22мс
спеціальності 172 – Телекомунікації та радіотехніка
_____ Канюк Д.В.

Керівник: к.т.н., доц. каф. ІКСТ
_____ Воловик А.Ю.
«11» 06 2024 р.

Рецензент: д.т.н., проф., професор каф. ІРТС
_____ Семенов А.О.
«11» 06 2024 р.

Допущено до захисту

Завідувач кафедри ІКСТ

_____ д.т.н., проф. Кичак В.М.

«11» 06 2024 р.

Вінниця ВНТУ - 2024 рік

Вінницький національний технічний університет
Факультет інформаційних електронних систем
Кафедра інфокомунікаційних систем та технологій
Рівень вищої освіти перший (бакалаврський)
Галузь знань 17 – Електроніка та телекомунікації
(шифр і назва)
Спеціальність 172 – Телекомунікації та радіотехніка
(шифр і назва)
Освітня програма – Програмне забезпечення телекомунікаційних систем

ЗАТВЕРДЖУЮ

Завідувач кафедри ІКСТ
д.т.н., професор В.М. Кичак
"06" 05 2024 року

З А В Д А Н Н Я
НА БАКАЛАВРСЬКУ ДИПЛОМНУ РОБОТУ СТУДЕНТУ

Канюку Дмитру Васильовичу
(прізвище, ім'я, по батькові)

1. Тема роботи: «Телеграм-бот для вивчення об'єктно-орієнтованої мови програмування на базі протоколу http»
керівник роботи Воловик Андрій Юрійович, к.т.н, доц.
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджені наказом ВНТУ від "11" березня 2024 року № 80
2. Строк подання студентом роботи 07 червня 2024 року

3. Вихідні дані до роботи: Тип інфокомунікаційної мережі – програмно-керована мережа; Режим роботи інфокомунікаційної мережі - багатоканальна; Тип стандарту зв'язку – IEEE 802.11; Сумарна швидкість інформаційних потоків - Ethernet 100 Мбіт/с згідно технічного завдання; Імовірність помилки в радіотракті - 10-6;Тип програмного забезпечення - віртуальні машини; Типи апаратного забезпечення – обладнання програмно-керованого центру оброблення даних; Метод доступу до програмного середовища – дистанційний; Режим роботи мережі доступу до навчального ресурсу – адаптивний зі зміною кількості користувачів та пропускної здатності сервісів;.

4. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		завдання видав	завдання прийняв
Спеціальна частина	Воловик А.Ю., доцент кафедри ІКСТ	06.05.2024	11.06.2024
Охорона праці	Демебірська С.В. проф. каф. ЦЖДПБ	06.05.2024	11.06.2024

5. Дата видачі завдання 06 травня 2024 року

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів бакалаврської дипломної роботи	Строк виконання етапів роботи	Примітка
1	Формування та затвердження теми та індивідуального завдання бакалаврської дипломної роботи (БДР)	06.05.2024р.	
2	Виконання спеціальної частини БДР. Перший рубіжний контроль виконання БДР	20.05.2024р.	
3	Виконання спеціальної частини БДР. Другий рубіжний контроль виконання БДР	31.05.2024р.	
4	Виконання розділу «Охорона праці»	04.06.2024р.	
5	Нормоконтроль БДР	05.06.2024р.	
6	Попередній захист БДР	06.06.2024р.	
7	Рецензування БДР	07.06.2024р.	
8	Захист БДР	12.06.2024р.	

Студент _____ Канюк Д.В.
(підпис)

Керівник роботи _____ Воловик А.Ю.
(підпис)

АНОТАЦІЯ

УДК 621.396

Канюк Д.В. Телеграм бот для вивчення об'єктно орієнтованої мови програмування на базі протоколу http – бакалаврська дипломна робота студента спеціальності 172 – Телекомунікації та радіотехніка – Вінниця: ВНТУ 2024 р. 91 стор., 18 – рис., 9 – табл. 19 -джерел.

У бакалаврській дипломній роботі розроблено телеграм бота що допомагає вивчати мову програмування Java. Протягом розробки було використано методи розробки клієнтського інтерфейсу у вигляду боту для телеграму; методи навчання; методи створення бази даних; методи задання ключових параметрів меню; методи оптимізації бекенд логіки та її клієнтського інтерфейсу; методи алгоритмізації модулів програмної системи. На кінцевому етапі роботи проводиться дослідження питань охорони праці.

Ключові слова: мова програмування, Java, телеграм бот, протокол передачі даних, http

ANNOTATION

Kaniuk D.V. Telegram bot for learning an object-oriented programming language based on the http protocol - bachelor thesis of a student of specialty 172 - Telecommunications and radio engineering - Vinnytsia: VNTU 2024. 91 pages, 18 - fig., 9 – table, 19 - sources.

In this course project, a telegram bot was developed that helps to learn the Java programming language. During the development, methods of developing a client interface in the form of a bot for Telegram were used; teaching methods; database creation methods; methods of setting key menu parameters; methods of optimizing the backend logic and its client interface; algorithmization methods of software system modules.

Keywords: programming language, Java, Telegram bot, data transfer protocol, http

ЗМІСТ

ВСТУП 5
1. ОБГРУНТУВАННЯ ВИБОРУ МОВИ, ПРОТОКОЛУ ТА СЕРЕДОВИЩА ПРОГРАМУВАННЯ ДОДАТКУ.....7
1.1 Аналіз стану галузі вивчення мови програмування Java.....7
1.2 Порівняльний аналіз аналогів розроблюваної системи.....12
1.3 Постановка задач проєкту.....15
1.4 Порівняльний аналіз протоколів передачі даних.....16
2 ТЕХНІЧНЕ ОБГРУНТУВАННЯ ОПТИМАЛЬНОГО ВАРІАНТА ВИРІШЕННЯ ОСНОВНОЇ ЗАДАЧІ З АНАЛІЗОМ СУЧАСНОГО СТАНУ РОЗРОБОК.....38
2.1 Аналіз та структурування телеграм боту.....38
2.2 Розробка структури інтерфейсу телеграм бота для вивчення мови програмування Java.....39
3 РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ КЛІЄНТСЬКОЇ ЧАСТИНИ ДЛЯ КЕРУВАННЯ ПАРАМЕТРАМИ ПРОГРАМНОГО СЕРЕДОВИЩА ВИВЧЕННЯ МОВИ ПРОГРАМУВАННЯ.....42
4 РОЗРОБЛЕННЯ ПРОГРАМНОГО ЩАБЕЗПЕЧЕННЯ СЕРВЕРНОЇ ЧАСТИНИ ТЕЛЕГРАМ БОТА.....48
5 ОХОРОНА ПРАЦІ.....53
5.1 Технічні рішення щодо безпечного виконання роботи.....53
5.2 Технічні рішення з гігієни праці та виробничої санітарії.....57
5.2.1 Мікроклімат.....57
5.2.2 Склад повітря робочої зони.....58
5.2.3 Виробниче освітлення.....59
5.2.4 Виробничий шум.....61
5.3 Пожежна безпека.....63
5.3.1 Технічні рішення системи запобігання пожежі.....64
5.3.2 Технічні рішення системи протипожежного захисту.....66
ВИСНОВКИ.....67
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ.....68
ДОДАТОК А (ілюстрації).....70
ДОДАТОК Б (антиплагіат).....77
ДОДАТОК В (лістинг).....78

ВСТУП

Сфера інформаційних технологій стає дедалі популярнішою, і все більше людей прагнуть долучитися до неї, адже, як стверджується, можна навчитися цьому у вільний час вдома. Проте, при спробі розпочати навчання, початківці часто стикаються з надлишком інформації і не знають, з чого почати. Найскладнішим є вибір мови програмування. Людині, незнайомій зі сферою інформаційних технологій, важко визначити, яку мову обрати, оскільки всі топ-15 мов актуальні і потрібні по-своєму. Важливо вибирати не просто мову, а напрямок, адже знання якоїсь мови програмування не забезпечує професії. Потрібно знати специфіку галузі та технології, що необхідні для тієї чи іншої професії.

Однією з найпотужніших високорівневих і компільованих мов є Java.

Java — одна з найпопулярніших мов програмування, розроблена компанією Sun Microsystems під керівництвом Джеймса Гослінга у 1995 році. Основною метою було створення інструменту, що дозволить розробникам писати код один раз і запускати його на будь-якій платформі без необхідності перекомпіляції (принцип WORA — Write Once, Run Anywhere). Java широко використовується в різних галузях для створення різноманітних програм завдяки своїй універсальності та численним перевагам.

Java залишається однією з найпопулярніших мов програмування у світі завдяки універсальності, надійності та широкому спектру бібліотек та інструментів. Вона використовується для розробки настільних додатків, ігор, фінансових програм та багато іншого. Постійне оновлення та розвиток гарантують її актуальність та затребуваність. Багато програмістів вивчили програмування самотужки завдяки доступним ресурсам в інтернеті. Створення таких ресурсів є дуже важливим, адже наявні ресурси часто містять лише документацію без чіткого шляху навчання, що може збити новачків з пантелику і відвернути їх від навчання. [1].

Велика частина програмістів стали ними завдяки вивчення програмування самотужки за допомогою доступних в інтернеті ресурсів. Саме по цій причині актуальність створення таких ресурсів є дуже високою в наш час.

Зазвичай на ресурсах для вивчання мов програмування доступна лише їх документація з поясненнями того як вони працюють, але без чіткого шляху як саме можна їх вивчити, а через це новачки можуть загубитися в інформації та можливо навіть залишити ідею навчання через думку, що це занадто складно.

Отже, розробка ресурсу, що допоможе в вивченні мови програмування Java, а ця мова є однією з найкращих для початківців в вигляді телеграм боту є актуальною, бо все більше людей бажають розпочати свій шлях в сфері інформаційних технологій і вони почати це робити просто в своєму телефоні не встановлюючи жодних програм.

Метою даної бакалаврської дипломної роботи є підвищення продуктивності вивчення мови програмування Java шляхом створення телеграм боту-помічника для самостійного навчання.

Задачами бакалаврської дипломної роботи є:

- аналіз стану галузі вивчення мови програмування Java;
- порівняльний аналіз аналогів;
- аналіз методів розв’язання задачі;
- постановка задач;
- розробка структури і алгоритмів програмного продукту;
- варіантний аналіз і обґрунтування вибору засобів для реалізації програмного засобу;
- розробка телеграм боту для вивчення мови програмування Java;
- тестування програми.

Протягом розробки було використано методи розробки клієнтського інтерфейсу у вигляду боту для телеграму; методи навчання; методи створення бази даних; методи задання ключових параметрів меню; методи оптимізації бекенд логіки та її клієнтського інтерфейсу; методи алгоритмізації модулів програмної системи. Основні ідеї роботи були доповідені і обговорені на науковій конференції ВНТУ у 2024 році.

1 ОБГРУНТУВАННЯ ВИБОРУ МОВИ, ПРОТОКОЛУ ТА СЕРЕДОВИЩА ПРОГРАМУВАННЯ ДОДАТКУ

1.1 Аналіз стану галузі вивчення мови програмування Java

Завдяки стрімкому розвитку IT-сфери на сьогоднішній день існує чимало сервісів для вивчення мов програмування для різних рівнів підготовки. Аби отримати нові знання, потрібно лише мати час та бажання. Але з розвитком постають і нові проблеми: неспроможність обрати саме ту програму, яка потрібна; багато різної інформації, але далеко не вся є достовірною та якісною; чимало вузькоспеціальної лексики, яку новачкам доводиться шукати окремо в мережі Інтернет.

Програма — це алгоритм, записаний мовою програмування для розв'язання конкретних завдань на комп'ютері. У програмному коді команди подаються мовою програмування.

Мова програмування — це штучна мова, яка є системою позначень і правил для запису алгоритмів у формі, придатній для виконання комп'ютером.

Складові мови програмування включають:

Алфавіт мови: набір символів, з яких утворюються команди та інші мовні конструкції.

Синтаксис мови: правила побудови команд мови програмування.

Семантика мови: правила виконання комп'ютером команд, записаних мовою програмування.

Мови програмування класифікують за декількома критеріями:

- За рівнем абстракції:

Мови програмування низького рівня базуються на машинних командах процесора і використовуються для розробки програм.

Мови програмування високого рівня оперують сутностями, зрозумілішими людині, такими як об'єкти, функції тощо.

- За областю застосування:

Універсальні мови використовуються для розв'язання різноманітних завдань.

Спеціалізовані мови призначені для вирішення завдань певного виду.

До універсальних мов належать Python, C/C++, Java, Object Pascal та інші. Спеціалізовані мови включають PHP, Perl, VBScript, JavaScript (призначені для вебпрограмування).

- За парадигмами програмування:

Парадигма програмування — це система ідей і понять, що визначають стиль написання програм та уявлення програміста про роботу програми.

Структурна парадигма: програма розглядається як послідовність дій. Основні поняття включають оператор (команда для опрацювання даних), змінну, базові алгоритмічні конструкції для керування послідовністю виконання операторів. Кожна конструкція має один вхід і один вихід.

Процедурна парадигма: програма складається з блоків команд — процедур або функцій, що дозволяє використовувати певний фрагмент коду, записавши його один раз і надавши йому назву.

Об'єктно-орієнтована парадигма: програма розглядається як сукупність об'єктів, що взаємодіють між собою. Об'єкти мають набір властивостей, виконують дії над даними і реагують на події.

Транслятор — це програма-перекладач, що перетворює програму з однієї з мов високого рівня у програму, що складається з машинних команд. Транслятори поділяються на інтерпретатори і компілятори.

Інтерпретатор: переводить і виконує програму рядок за рядком, перетворюючи невеликий фрагмент вихідної програми на машинний код, який одразу виконується процесором. Машинний код для повторного виконання не зберігається.

Компілятор: перетворює всю програму на машинні коди, зберігаючи їх у пам'яті комп'ютера без виконання. Скомпільовану програму можна зберегти для подальшого використання. Збережений результат компіляції називається виконуваним файлом (наприклад, з розширенням *.exe в ОС Windows).

Проблема розробки програми під себе, незнання того, що за чим потрібно вчити є доволі поширеною, адже мережа Інтернет – це величезна купа корисної та навпаки непотрібної і шкідливої інформації. Через це багато людей заробляють на таких новачках, розробляючи програму навчання, даючи вже підготовлену інформацію та просячи за те саме, що є у вільному доступі, гроші.

Головним завданням людини, яка бажає почати свій шлях в сфері інформаційних технологій є вибір мови програмування. Від цього вибору може залежати чи сподобається людині ця галузь і чи продовжить вона свій в ній шлях. Однією з найкращих мов для початківців є мова Java. Java є універсальною мовою програмування. Ось деякі з ключових сфер її застосування:

Веб-розробка: Java широко використовується для розробки back-end компонентів веб-сайтів та веб-застосунків. Її популярні фреймворки, такі як Spring MVC і JSF, полегшують створення масштабованих та надійних веб-рішень.

Розробка мобільних додатків: Java є офіційною мовою програмування для платформи Android, що робить її домінуючим вибором для розробки Android-додатків. Її інструменти та бібліотеки, такі як Android SDK і Kotlin, роблять процес створення мобільних додатків ефективним та зручним.

Розробка корпоративних програм: Java широко використовується для розробки корпоративних програм, таких як системи ERP, CRM та системи управління ланцюжками постачання. Її надійність, безпека та масштабованість роблять її ідеальною для критично важливих програмних рішень.

Вбудовані системи: Java використовується для розробки вбудованого програмного забезпечення для різних пристроїв, таких як смарт-телевізори, розумні будинки, медичні прилади та багато іншого. Її портативність та енергоефективність роблять її привабливим вибором для вбудованих систем.

Наукові обчислення: Java використовується в наукових обчисленнях для розробки складних алгоритмів і програмного забезпечення для моделювання. Її бібліотеки наукових обчислень, такі як Java Math Toolbox та Apache Commons Math, полегшують роботу з науковими даними та складними математичними розрахунками.

Великі дані: Java використовується в галузі великих даних для розробки програмного забезпечення для обробки та аналізу великих обсягів даних. Її фреймворки великих даних, такі як Hadoop і Spark, дозволяють ефективно обробляти та аналізувати розподілені набори даних.

Машинне навчання: Java використовується в машинному навчанні для розробки моделей машинного навчання та програмного забезпечення для аналізу даних. Її бібліотеки машинного навчання, такі як Weka і H2O, полегшують створення та навчання моделей машинного навчання.

Проте головна галузь її використання - це бекенд-розробка, тобто та частина програми або веб-сайту, яка працює на сервері та відповідає за обробку запитів користувачів і взаємодію з базою даних. Це основна складова системи, яку користувачі не бачать, але яка забезпечує її правильну роботу. Java має широкий спектр бібліотек і фреймворків, що робить його досить універсальною мовою програмування. Вона використовується для розробки не

лише бекенду веб-додатків, але й для створення мікросервісів, десктопних застосунків та навіть вбудованих систем. Підтримка Java є практично всюди. Будь-який пристрій, операційна система або серверний середовище здатні виконувати Java-код, що забезпечує стабільність та надійність роботи програм незалежно від того, де вони запускаються. Можливість використання Java в різних сферах відкриває безліч можливостей для розробників. Вона стає важливим інструментом для реалізації різноманітних проєктів та застосунків у різних галузях індустрії. [1].

Сучасна Java є мовою програмування, яка вважається "безпечною". Вона обмежує доступ до низькорівневих операцій з пам'яттю або процесором, оскільки спочатку була розроблена для застосування у браузерах, де такі функції не потрібні.Можливості Java на бекенді залежать від середовища, в якому вона виконується. Наприклад, у своєму середовищі вона підтримує операції читання/запису файлів, виконання мережевих запитів та багато іншого. На бекенді за допомогою Java можна здійснювати різноманітні операції, такі як:

- взаємодія з базою даних, зчитування та запис інформації;
- обробка запитів від клієнтів, включаючи обробку HTTP-запитів;
- відправлення повідомлень на різні сервери, отримання та обробка відповідей;
- робота з кукісами, сеансами та іншими аспектами стану клієнта;
- зберігання даних на сервері або в базі даних.

Java на бекенді забезпечує надійну та безпечну роботу веб-додатків та інших систем для яких вона використовується, де можна використовувати її для різноманітних завдань, пов'язаних з обробкою запитів та управлінням даними.

Через те, що у неосяжній мережі Інтернет занадто багато інформації, часто просто блукаєш і не знаєш, що саме тобі потрібно. Навіть якщо і обираєш для себе конкретний сервіс, потім розумієш, що це не зовсім те, що тобі потрібно. Через цю проблему багато людей можуть бути дезінформованими і, думаючи, що вони праві, поширювати неправду серед інших, породжуючи багато плутанини. Проблема обирання інформації, розрахованої на іншу аудиторію, є не менш поширеною, ніж попередні, але часто люди встигають зрозуміти, що дане скупчення знань не для їх рівня. Усе ж таки бувають моменти, коли на початку начебто все зрозуміло, але чим дальше – тим більш заплутано та незрозуміло.

Отже, враховуючи усе вищесказане, зроблено висновок, що потрібно розробити безкоштовну систему для вивчення мови програмування Java, у якій буде зібраний якісний матеріал з найкорисніших ресурсів, для поглинання якого не потрібно буде блукати мережею.

1.2 Порівняльний аналіз аналогів розроблюваної системи

Найперше, що потрібно визначити це платформу для якої буде розроблятися бот. Порівняно з іншими платформами, такими як Facebook Messenger або Slack, телеграм має деякі переваги. Наприклад, Facebook Messenger має велику аудиторію, але інтеграція з ним може бути складнішою. Slack часто використовується для командної роботи, тому він може бути корисним, якщо ваш бот призначений для командної роботи або спільного навчання. Вибір платформи залежить від вашої цільової аудиторії та функціональних вимог вашого бота.

Ось кілька основних переваг створення телеграм бота:

Доступність: Телеграм має широкий охоплення користувачів у багатьох країнах світу. Це означає, що ваш бот матиме потенціал доступу до великої аудиторії.

Простота використання: Телеграм відомий своєю інтуїтивно зрозумілою і легкою у використанні платформою. Це робить навчання Java через телеграм-бота більш привабливим для новачків.

Миттєва зворотний зв'язок: Телеграм дозволяє надсилати повідомлення в реальному часі. Це означає, що користувачі можуть одразу отримувати відповіді на свої запитання і швидко вирішувати проблеми.

Можливості інтеграції: Телеграм має широкий спектр можливостей інтеграції з іншими сервісами та платформами, що дозволяє розширити функціональність вашого бота.

Результати порівняння платформ наведено у таблиці 1.1.

Таблиця 1.1 – Порівняння платформ

Критерії оцінювання	Телеграм-бот для навчання Java	Бот для навчання Java для Discord	Бот для навчання Java для Facebook Messenger	Бот для навчання Java для Slack
Доступність аудиторії	Широкий охоплення користувачів	Головно геймерська аудиторія	Велика аудиторія, різноманітна демографія	Переважно для командної роботи
Функціональні можливості	Простий інтерфейс, легка навігація	Голосові канали, можливості спільної гри	Можливості чат-боту, інтерактивність	Можливості організації робочого процесу

Можливості інтеграції	Широкі можливості інтеграції	Можливості інтеграції, але API може бути складніше	Широкі можливості інтеграції з іншими сервісами	Широкі можливості інтеграції з іншими сервісами
Спільнота та підтримка	Активна спільнота розробників та користувачів	Активна спільнота розробників та користувачів	Активна спільнота розробників та користувачів	Активна спільнота розробників та користувачів

Найкращим шляхом визначення актуальності розробки системи є проведення аналізу її аналогів та їх порівняння за найголовнішими для навчальних ресурсів критеріями оцінювання.

Оскільки мова Java є однією з найпоширеніших, існує багато ресурсів, які допомагають з її вивченням. Майже кожен з них виконує свою окрему роль, але більшість являють собою документацію до того як працювати з цією мовою і представлені у вигляді веб-сторінок чи додатків на телефони. Документація не є гарним способом вивчення, тому що інформація там подана в не структурованому виді, і якщо початківець захоче дізнатися щось нове, він може наткнутися на тему для розуміння якої йому потрібно вже мати немалий багаж знань. Занурившись в інтернет в пошуках аналогів я не знайшов ботів у телеграмі які б допомагали саме вчити мову програмування Java українською. Більшість з існуючих – це англо/російсько-мовні боти з всілякими тестами на знання мови чи просто статтями. До того ж більшість з них не активні.

З розробленою системою під назвою “JavaHelper” було порівняно її аналоги за такими критеріями:

1. Можливість створення аккаунта – якщо в користувача є можливість авторизуватися на сайті, в нього є можливість зберігати,

наприклад, свій прогрес або підписуватися на якісь теми та їх оновлення якщо це доступно.

2. Наявність тестувань – можливість перевірки набутих знань. При перевірці своїх знань набагато легше дізнатися, які теми варто підтянути.

3. Структуроване навчання – на багатьох навчальних ресурсах інформація представлена без чуткої структури і початківцю може бути незрозуміло, які теми йому потрібно вчити на початку, а які необхідні для досвідчених користувачів.

Виходячи з результатів аналізу аналогів, можна зробити висновок, що телеграм бот для вивчення мови програмування Java є актуальним на сьогоднішній день та має переваги, які роблять його конкурентоспроможною.

1.3 Постановка задач

Проаналізувавши стан галузі вивчення програмування Java та дослідивши методи розробки продукту для розв’язання теми «Розробка системи для вивчення мови програмування Java», було виділено наступні задачі, які будуть забезпечувати високу якість продукту і його модулів:

− визначити найбільш ефективний метод вивчення мови програмування Java;

− розробити структуру телеграм боту;

− розробити клієнтську частину в якій користувачам буде зручно оперувати;

− розробити базу даних, яка буде містити інформацію про користувачів;

− здійснити тестування програмного модуля.

Отже, було поставлено задачі дослідження на тему «Розробка веб-системи для вивчення мови програмування Java».

1.4 Порівняльний аналіз протоколів передачі даних

Протокол мережі — це набір правил, що дозволяє обмінюватися даними між пристроями, забезпечуючи доставку пакетів і реалізацію певних процесів. Функціонування Інтернету базується на роботі кількох протоколів, розташованих один поверх іншого. Розглянемо основні види мережевих протоколів і значення їхніх абревіатур.

Основні види протоколів передачі даних:

- MAC (Media Access Control)

MAC (Media Access Control) є одним із ключових компонентів у структурі мережевих комунікацій, забезпечуючи контроль доступу до середовища передачі даних. MAC-адреса – це унікальний ідентифікатор, призначений для кожного мережевого інтерфейсу, що дозволяє пристроям у мережі ідентифікувати один одного та спілкуватися. Ця доповідь надасть детальний огляд функцій та значення MAC у мережевих технологіях.

MAC-адреса є 48-бітовим числом, зазвичай вираженим у вигляді шістнадцяткових цифр, розділених двокрапками або тире. Ця адреса поділяється на дві частини: перші 24 біти ідентифікують виробника обладнання (OUI - Organizationally Unique Identifier), а останні 24 біти є унікальним номером для конкретного пристрою. Завдяки цій структурі, кожна MAC-адреса є унікальною і дозволяє точно ідентифікувати пристрій у мережі.

MAC грає вирішальну роль у функціонуванні локальних мереж (LAN). Коли пристрій хоче відправити дані, він використовує MAC-адресу призначення для визначення, до якого конкретного пристрою слід доставити

ці дані. Це відбувається на рівні канального шару моделі OSI, забезпечуючи ефективну маршрутизацію кадрів даних між пристроями в межах тієї ж локальної мережі.

Один з основних принципів роботи MAC – це використання протоколу доступу до середовища передачі, такого як Ethernet. Ethernet використовує метод доступу CSMA/CD (Carrier Sense Multiple Access with Collision Detection), який дозволяє пристроям слухати середовище передачі перед відправленням даних, щоб уникнути колізій. Якщо виникає колізія, пристрої чекають випадковий проміжок часу перед повторною спробою передачі.

Інший важливий аспект MAC – це функція управління потоками даних. Вона забезпечує, щоб пристрої не переповнювали приймач великою кількістю даних, ніж той може обробити. Це досягається через механізми, такі як управління потоком IEEE 802.3x для дротових мереж Ethernet, що дозволяє пристроям сигналізувати відправнику про уповільнення швидкості передачі даних.

MAC-адреса використовується також у питаннях безпеки мереж. Наприклад, фільтрація MAC-адрес дозволяє мережевим адміністраторам обмежувати доступ до мережі тільки дозволеним пристроям. Проте, важливо зазначити, що MAC-адреси можуть бути підроблені або змінені, що робить цю форму захисту не завжди надійною без додаткових заходів безпеки.

MAC (Media Access Control) є критично важливою технологією в мережевих комунікаціях, забезпечуючи унікальну ідентифікацію пристроїв і управління доступом до середовища передачі даних. Завдяки своїм функціям та структурі, MAC сприяє ефективній та безпечній передачі даних у локальних мережах, роблячи її незамінною частиною сучасних мережевих технологій.

- IP

IP (Internet Protocol) є основним протоколом у мережевій архітектурі Інтернету, який забезпечує адресацію та маршрутизацію даних між комп'ютерами та іншими пристроями. Без IP було б неможливо визначити, куди слід доставляти дані в мережі, що робить його критично важливим для функціонування Інтернету. У цій доповіді ми розглянемо структуру, функції та значення IP у сучасних мережах.

IP -адреса – це унікальний числовий ідентифікатор, призначений кожному пристрою, підключеному до мережі. Існує дві основні версії IP: IPv4 та IPv6. IPv4 використовує 32-бітові адреси, що дозволяє створити приблизно 4,3 мільярда унікальних адрес. IPv6, у свою чергу, використовує 128-бітові адреси, що забезпечує практично необмежений простір адрес.

IP-адреси складаються з двох основних частин: мережевої та хостової. Мережева частина ідентифікує підмережу, до якої належить пристрій, тоді як хостова частина ідентифікує конкретний пристрій у цій підмережі. Завдяки такій структурі маршрутизатори можуть ефективно визначати шляхи для передачі даних від одного пристрою до іншого через різні мережі.

Таблиця 1.2 – Порівняльна таблиця для ipv6 та ipv4

ipv4	**ipv6**
Розмір пакету: 576 байт	Розмір пакету: 1280 байт
Безпека не передбачена	Включено аутентифікацію та шифрування
Використання в мережах: 99%	Використання в мережах: 1%
Доступний у всіх хостинг-провайдерів	Доступний у не всіх хостинг-провайдерів
Чотири десяткові числа розділених крапкою	Шестизначне число з розділенням двокрапкою

Основне завдання IP – це маршрутизація пакетів даних від джерела до місця призначення. Під час маршрутизації кожен пакет проходить через ряд маршрутизаторів, які використовують таблиці маршрутизації для визначення найкращого шляху до цільового IP-адреса. Кожен маршрут може включати кілька проміжних вузлів, і маршрутизатори можуть оновлювати свої таблиці маршрутизації на основі поточної мережевої топології та умов трафіку.

Протокол IP працює на мережевому рівні моделі OSI, забезпечуючи універсальний спосіб адресації та передачі даних незалежно від фізичного середовища мережі. Це дозволяє різним типам мереж взаємодіяти один з одним, створюючи єдиний глобальний Інтернет.

IP також відіграє важливу роль у забезпеченні безпеки мережі. Технології, такі як IPsec (Internet Protocol Security), забезпечують шифрування та автентифікацію IP-пакетів, що дозволяє створювати захищені з'єднання між пристроями. IP-адреси використовуються для контролю доступу та фільтрації трафіку, що дозволяє адміністраторам обмежувати доступ до мережі лише авторизованим користувачам.

IP (Internet Protocol) є основою сучасних мережевих технологій, забезпечуючи ефективну адресацію та маршрутизацію даних у глобальній мережі Інтернет. Завдяки своїй структурі та функціональності, IP дозволяє різноманітним пристроям взаємодіяти між собою, роблячи Інтернет надійним та масштабованим. Без IP сучасний Інтернет, яким ми його знаємо, просто не існував би.

- ICMP (Internet control message protocol)

ICMP (Internet Control Message Protocol) є одним з ключових протоколів у сімействі Інтернет-протоколів, забезпечуючи передачу повідомлень про

помилки та діагностичну інформацію між пристроями в мережі. ICMP не використовується для передачі даних, а виконує допоміжну функцію, забезпечуючи ефективну роботу та управління мережею. У цій доповіді ми розглянемо структуру, функції та значення ICMP у сучасних мережах.

ICMP працює на мережевому рівні моделі OSI і використовується переважно маршрутизаторами, хостами та іншими мережевими пристроями для обміну повідомленнями про помилки та іншою діагностичною інформацією. Повідомлення ICMP інкапсулюються у IP-пакети, що дозволяє їм бути переданими через мережу як частина стандартного трафіку IP.

Повідомлення ICMP складаються з заголовка та даних. Заголовок містить такі поля, як тип, код та контрольна сума. Поле "тип" визначає загальну категорію повідомлення (наприклад, помилка або запит), тоді як поле "код" надає додаткову інформацію про конкретний тип повідомлення. Дані повідомлення можуть включати IP-заголовок і перші 8 байтів даних оригінального пакета, що дозволяє відправнику ідентифікувати, яке повідомлення викликало помилку.

Однією з основних функцій ICMP є повідомлення про помилки. Наприклад, якщо маршрутизатор не може доставити пакет до місця призначення, він може відправити повідомлення ICMP типу "Destination Unreachable" до відправника, повідомляючи його про проблему. Інші типи повідомлень про помилки включають "Time Exceeded" (коли пакет перевищив допустимий час у мережі) і "Redirect" (коли маршрутизатор повідомляє відправника про кращий маршрут).

ICMP також використовується для діагностики мережі. Найвідомішим інструментом на основі ICMP є утиліта "ping", яка надсилає ICMP-запити типу "Echo Request" до цільового хоста і очікує відповіді "Echo Reply". Це дозволяє користувачам перевіряти доступність та час відгуку віддалених пристроїв.

Утиліта "traceroute" також використовує ICMP для визначення маршруту, який пакет проходить через мережу до свого місця призначення.

ICMP грає важливу роль у підтримці надійності та ефективності мережі, забезпечуючи засоби для виявлення та усунення проблем. Однак ICMP може бути використаний і для атак, таких як "ping flood" або "smurf attack", де зловмисники перевантажують мережу великою кількістю ICMP-запитів. Через це мережеві адміністратори часто обмежують або фільтрують ICMP-трафік для підвищення безпеки.

ICMP (Internet Control Message Protocol) є незамінним інструментом для управління та діагностики мереж. Завдяки своїй здатності передавати повідомлення про помилки та діагностичну інформацію, ICMP допомагає забезпечити ефективну роботу мережевих систем. Розуміння та правильне використання ICMP є критично важливим для підтримки надійності та безпеки мережевих інфраструктур.

- TCP (Transmission control protocol)

TCP (Transmission Control Protocol) є одним з основних протоколів у наборі Інтернет-протоколів, забезпечуючи надійну та впорядковану передачу даних між комп'ютерами в мережі. TCP працює разом з IP (Internet Protocol), формуючи TCP/IP модель, яка є основою сучасного Інтернету. У цій доповіді ми розглянемо структуру, функції та значення TCP у мережевих комунікаціях.

TCP є протоколом транспортного рівня, що забезпечує надійну передачу даних між хостами. Кожне з'єднання TCP починається зі встановлення сесії, відомого як тристороннє рукостискання (three-way handshake), і закінчується завершенням сесії. Це рукостискання складається з обміну трьома пакетами:

SYN, SYN-ACK і ACK, що дозволяє хостам синхронізувати свої лічильники послідовності і підтвердити готовність до передачі даних.

TCP-сегменти складаються з заголовка та даних. Заголовок містить такі важливі поля, як порт відправника, порт отримувача, номер послідовності, номер підтвердження, прапори (наприклад, SYN, ACK, FIN), вікно розміру, контрольна сума та інші. Номери послідовності та підтвердження дозволяють TCP забезпечувати впорядковану доставку даних та виявляти втрату пакетів.

Однією з ключових особливостей TCP є забезпечення надійності передачі даних. Коли відправник надсилає дані, він очікує підтвердження від отримувача. Якщо підтвердження не надходить у певний проміжок часу, дані пересилаються знову. Цей механізм автоматичного повторного запиту (ARQ) забезпечує високу надійність передачі навіть у мережах з високим рівнем втрат.

TCP також включає механізми управління потоком і управління перевантаженнями. Управління потоком забезпечує, що відправник не перевантажує отримувача, надсилаючи більше даних, ніж той може обробити. Це досягається за допомогою вікна розміру, яке отримувач повідомляє відправнику. Управління перевантаженнями, з іншого боку, дозволяє TCP адаптувати швидкість передачі даних відповідно до стану мережі, щоб уникнути перевантаження каналів зв'язку. Цей механізм включає такі алгоритми, як контроль повільного старту, уникнення перевантажень та швидке відновлення.

TCP є критично важливим для багатьох інтернет-додатків, включаючи веб-браузинг, електронну пошту, передавання файлів та інші сервіси, де надійність та впорядкованість передачі даних мають вирішальне значення. TCP забезпечує, що дані надходять без помилок, у правильному порядку та без втрат, що робить його незамінним для комунікацій в Інтернеті.

TCP (Transmission Control Protocol) є основним елементом сучасних мережевих технологій, забезпечуючи надійну та впорядковану передачу даних. Завдяки своїм механізмам управління потоком, перевантаженнями та забезпеченням надійності, TCP дозволяє підтримувати стабільну та ефективну роботу інтернет-додатків. Розуміння TCP є критично важливим для адміністраторів мереж і розробників програмного забезпечення, які прагнуть забезпечити надійність і ефективність своїх мережевих рішень.

- UDP

UDP (User Datagram Protocol) є одним з ключових транспортних протоколів у наборі Інтернет-протоколів, забезпечуючи швидку та ефективну передачу даних між комп'ютерами в мережі. На відміну від TCP (Transmission Control Protocol), UDP не гарантує надійність, впорядкованість або контроль потоку, що робить його легким і швидким протоколом для певних типів комунікацій. У цій доповіді ми розглянемо структуру, функції та значення UDP у мережевих технологіях.

UDP є протоколом транспортного рівня, який забезпечує передачу даних без встановлення з'єднання. Кожен UDP-пакет, відомий як датаграма, містить заголовок та дані. Заголовок UDP є мінімальним і складається з чотирьох полів: порт відправника, порт отримувача, довжина та контрольна сума. Це робить UDP дуже простим і малозатратним у порівнянні з TCP.

Порти відправника та отримувача використовуються для ідентифікації прикладних процесів на хостах, які надсилають і отримують дані. Поле довжини вказує загальну довжину датаграми, включаючи заголовок і дані. Поле контрольної суми забезпечує перевірку цілісності даних, але його використання є необов'язковим у деяких реалізаціях.

Основна характеристика UDP полягає в тому, що він є ненадійним протоколом. Це означає, що UDP не гарантує доставку пакетів, не відстежує їх порядок і не намагається виправляти помилки або втрати пакетів. Це робить UDP дуже швидким, оскільки не витрачається час на встановлення з'єднання, контроль потоку або повторну передачу даних.

UDP підходить для додатків, де швидкість і ефективність є важливішими за надійність. Типові застосування включають стрімінг аудіо та відео, онлайн-ігри, голосові виклики (VoIP) та інші реального часу сервіси, де затримки повинні бути мінімальними. У таких додатках втрата кількох пакетів незначно впливає на загальну якість сервісу, тоді як затримка може бути критичною.

UDP відіграє важливу роль у сучасних мережевих комунікаціях, забезпечуючи легкий і швидкий спосіб передачі даних. Завдяки своїй простоті, UDP забезпечує низькі накладні витрати, що робить його ідеальним для додатків, де важлива швидкість передачі та низька затримка. Наприклад, у стрімінгових сервісах UDP дозволяє забезпечити плавне відтворення медіа, навіть якщо деякі пакети губляться під час передачі.

Проте, через свою ненадійність, UDP не підходить для всіх типів додатків. Для передачі даних, де важлива цілісність і впорядкованість, таких як передача файлів або веб-серфінг, TCP є кращим вибором.

UDP (User Datagram Protocol) є важливим компонентом транспортного рівня Інтернет-протоколів, забезпечуючи швидку та ефективну передачу даних без накладних витрат, пов'язаних із надійністю та впорядкованістю. Завдяки своїм характеристикам, UDP ідеально підходить для додатків, де затримка має критичне значення, таких як стрімінг медіа та реального часу комунікації. Розуміння UDP є важливим для мережевих інженерів і розробників, що прагнуть оптимізувати свої сервіси для конкретних потреб користувачів.

- HTTP

HTTP (Hypertext Transfer Protocol) є одним з найважливіших протоколів у сучасному Інтернеті, забезпечуючи передачу гіпертекстових документів, таких як веб-сторінки, між веб-серверами та клієнтами. Створений у 1990 році Тімом Бернерсом-Лі, HTTP став основою Всесвітньої павутини (World Wide Web). У цій доповіді ми розглянемо структуру, функції та значення HTTP у сучасних мережевих технологіях.

HTTP є протоколом прикладного рівня, який використовує модель клієнт-сервер. Клієнт (зазвичай веб-браузер) відправляє запити на веб-сервер, а сервер відповідає на ці запити, надаючи ресурси, такі як HTML-документи, зображення, відео та інші файли. HTTP працює за принципом "запит-відповідь", де кожен запит та відповідь складаються з заголовка і тіла.

Основні типи запитів які використовуються в HTTP:

- GET: Запит на отримання ресурсу. Це найпоширеніший метод, який використовується для отримання даних з сервера.
- POST: Запит на відправку даних на сервер для обробки. Використовується для надсилання форм і завантаження файлів.
- PUT: Запит на заміну ресурсу на сервері.
- DELETE: Запит на видалення ресурсу на сервері.
- HEAD: Запит, аналогічний GET, але без тіла відповіді. Використовується для отримання заголовків відповіді.

HTTP-заголовки містять метадані про запит або відповідь, такі як тип контенту, довжина контенту, статус код (наприклад, 200 OK, 404 Not Found) та інші.

HTTP є безстанним протоколом, що означає, що кожен запит і відповідь є незалежними один від одного. Це спрощує протокол, але також створює виклики для розробників, які хочуть зберігати стан між запитами. Ці виклики вирішуються за допомогою технологій, таких як cookies, сесії та токени.

З плином часу HTTP еволюціонував, щоб відповідати зростаючим вимогам до безпеки та ефективності. HTTP/1.1, випущений у 1997 році, додав підтримку для з'єднань, що залишаються відкритими, та кілька інших поліпшень. HTTP/2, стандартизований у 2015 році, включає такі особливості, як мультиплексування потоків, стискання заголовків та сервер push, що покращують продуктивність.

HTTP є основою для більшості веб-додатків і послуг, що робить його критично важливим для функціонування Інтернету. Веб-сайти, API, мобільні додатки і багато інших сервісів використовують HTTP для обміну даними. Без HTTP сучасний Інтернет не зміг би функціонувати так, як ми його знаємо сьогодні.

Безпека HTTP також є важливим аспектом. HTTP-трафік може бути легко перехоплений і прочитаний, тому більшість сучасних веб-сайтів використовують HTTPS (HTTP Secure), який включає шифрування даних за допомогою SSL/TLS. Це забезпечує конфіденційність, цілісність і автентичність даних, що передаються між клієнтом і сервером.

HTTP (Hypertext Transfer Protocol) є критично важливим протоколом, який забезпечує функціонування Всесвітньої павутини. Завдяки своїй простоті, гнучкості та масштабованості, HTTP став основою для передачі даних в Інтернеті.

Розуміння HTTP є важливим для розробників, адміністраторів мереж і всіх, хто працює з веб-технологіями, оскільки цей протокол забезпечує основні можливості, які роблять Інтернет таким, яким ми його знаємо сьогодні.

- FTP

FTP (File Transfer Protocol) є одним з найстаріших і найважливіших протоколів Інтернету, який забезпечує передачу файлів між клієнтом і сервером. Створений ще в 1971 році, FTP був розроблений для спрощення обміну файлами в комп'ютерних мережах. Незважаючи на свій вік, FTP залишається важливим інструментом для адміністраторів мереж і розробників, забезпечуючи ефективний спосіб управління файлами на віддалених серверах. У цій доповіді ми розглянемо структуру, функціональність і значення FTP.

FTP є протоколом прикладного рівня, який працює за моделлю клієнт-сервер. Клієнт FTP встановлює з'єднання з сервером FTP, після чого користувач може завантажувати файли на сервер або завантажувати їх з сервера на свій комп'ютер. FTP використовує два порти для роботи: порт 21 для управління з'єднанням і порт 20 для передачі даних. Ця розподільча структура дозволяє одночасно керувати кількома файлами та підтримувати кілька сесій передачі.

Є два режими передачі:

- Активний режим: Сервер ініціює з'єднання для передачі даних до клієнта.
- Пасивний режим: Клієнт ініціює з'єднання для передачі даних, що дозволяє уникнути проблем із брандмауерами та NAT.

Для ідентифікації користувачів FTP використовує автентифікацію за допомогою імені користувача та пароля. Однак існує також анонімний FTP, де користувачі можуть входити на сервер без ідентифікації, використовуючи загальний обліковий запис "anonymous".

Основний принцип роботи FTP полягає у встановленні двох з'єднань між клієнтом і сервером: з'єднання для керування та з'єднання для передачі даних.

Спочатку клієнт підключається до сервера через порт 21 і ініціює управління з'єднанням, відправляючи команди FTP, такі як USER (ім'я користувача) і PASS (пароль). Після успішної автентифікації клієнт може використовувати команди для навігації файловою системою сервера (наприклад, LIST для отримання списку файлів і директорій) та для передачі файлів (наприклад, STOR для завантаження файлу на сервер і RETR для завантаження файлу з сервера).

FTP залишається важливим інструментом для адміністраторів мереж і розробників, забезпечуючи ефективний і надійний спосіб управління файлами на віддалених серверах. Хоча новіші протоколи, такі як SFTP (SSH File Transfer Protocol) і FTPS (FTP Secure), пропонують підвищену безпеку завдяки шифруванню, FTP все ще використовується в багатьох середовищах, де безпека не є головним пріоритетом або де інші методи захисту вже впроваджені на рівні мережі.

FTP також має важливе значення у сценаріях автоматизованої передачі файлів, де потрібно регулярно передавати великі обсяги даних між системами. Багато підприємств використовують FTP для резервного копіювання, обміну великими наборами даних та інтеграції з іншими системами.

FTP (File Transfer Protocol) є фундаментальним інструментом для передачі файлів у мережевих середовищах. Завдяки своїй простоті та ефективності, FTP залишається важливим для адміністраторів мереж і розробників, забезпечуючи надійний спосіб управління файлами на віддалених серверах. Незважаючи на наявність більш безпечних альтернатив, FTP продовжує грати важливу роль у багатьох мережевих інфраструктурах завдяки своїй функціональності та широкій підтримці.

- DNS (domain name system)

DNS (Domain Name System) — протокол, що перетворює зрозумілі для людини адреси в складні IP-адреси і навпаки. Завдяки DNS ми можемо отримувати доступ до інтернет-ресурсів за їхніми доменними іменами.

DNS (Domain Name System) є одним з найважливіших компонентів Інтернету, забезпечуючи перетворення зручних для людини доменних імен на IP-адреси, необхідні для маршрутизації та доступу до ресурсів в мережі. Створений у 1983 році Полом Мокапетрісом, DNS став основою для зручного користування Інтернетом. У цій доповіді ми розглянемо структуру, функціональність та значення DNS у сучасних мережах.

DNS є ієрархічною дистрибуційною системою баз даних, яка використовує клієнт-серверну модель для зберігання та обробки доменних імен.

Основні компоненти DNS включають:

- Доменні імена: Зручні для людини адреси, такі як www.example.com.
- IP-адреси: Числові адреси, такі як 192.0.2.1, які використовуються для ідентифікації пристроїв у мережі.
- DNS-сервери: Сервери, які зберігають інформацію про доменні імена та IP-адреси, і відповідають на запити DNS-клієнтів (рекурсивних резолверів).

Є кілька рівнів ієрархії DNS:

- Кореневі сервери: Найвищий рівень DNS, який управляє кореневою зоною і направляє запити до відповідних TLD-серверів.
- TLD-сервери: Сервери, які управляють доменами верхнього рівня, такими як .com, .org, .net.

- Авторитетні DNS-сервери: Сервери, які зберігають інформацію про конкретні домени та їхні IP-адреси.

Коли користувач вводить доменне ім'я в браузері, відбувається кілька кроків для отримання відповідної IP-адреси:

1. Запит клієнта: Клієнт (рекурсивний резолвер) надсилає запит на локальний DNS-сервер.
2. Запит до кореневих серверів: Якщо локальний DNS-сервер не має відповідної інформації, він надсилає запит до кореневих серверів.
3. Запит до TLD-серверів: Кореневі сервери направляють запит до відповідних TLD-серверів.
4. Запит до авторитетних серверів: TLD-сервери направляють запит до авторитетних DNS-серверів для конкретного домену.
5. Отримання IP-адреси: Авторитетний сервер відповідає з відповідною IP-адресою, яка повертається клієнту.

Цей процес, відомий як рекурсивне резолювання, зазвичай відбувається за лічені мілісекунди, забезпечуючи швидкий доступ до веб-сайтів.

DNS є критично важливим для функціонування Інтернету, забезпечуючи легкий доступ до веб-сайтів та інших ресурсів. Без DNS користувачам довелося б запам'ятовувати числові IP-адреси для кожного ресурсу, що було б непрактичним і незручним. DNS також підтримує масштабованість Інтернету, дозволяючи додавати нові домени та IP-адреси без переривання роботи мережі. Крім того, DNS забезпечує різні додаткові функції, такі як балансування навантаження, де кілька IP-адрес можуть відповідати за один домен для розподілу трафіку, та захист від DDoS-атак через використання кешування та розподілу запитів.

DNS (Domain Name System) є фундаментальним компонентом Інтернету, забезпечуючи перетворення доменних імен на IP-адреси та забезпечуючи

зручний доступ до мережевих ресурсів. Завдяки своїй ієрархічній структурі та ефективному механізму резолювання, DNS забезпечує швидкий і надійний доступ до Інтернету для мільярдів користувачів по всьому світу. Розуміння DNS є важливим для адміністраторів мереж, розробників та всіх, хто працює з Інтернет-технологіями, оскільки цей протокол забезпечує основні можливості, які роблять Інтернет зручним і ефективним.

- SSH

SSH (Secure Shell) є критично важливим протоколом у світі комп'ютерних мереж, забезпечуючи безпечний і надійний доступ до віддалених серверів і пристроїв. Створений у 1995 році Тату Ілоненом, SSH швидко став стандартом для захищеного адміністрування систем і передачі даних. У цій доповіді ми розглянемо структуру, функціональність та значення SSH у сучасних мережевих технологіях.

SSH є протоколом прикладного рівня, який використовує криптографічні методи для захисту даних, що передаються між клієнтом і сервером.

Основні компоненти SSH включають:

- Клієнт SSH: Програма, що ініціює з'єднання з віддаленим сервером.
- Сервер SSH: Програма, яка приймає з'єднання від клієнтів і забезпечує доступ до системи.
- Шифрування: Використовується для захисту даних під час передачі. Популярні методи шифрування включають AES, Blowfish та RSA.

- Аутентифікація: Забезпечує ідентифікацію користувачів за допомогою паролів або криптографічних ключів (наприклад, RSA або DSA ключі).

SSH-сесія починається з ініціювання клієнтом з'єднання до сервера на стандартному порту 22. Після встановлення з'єднання сервер надсилає свій публічний ключ клієнту для шифрування подальшого обміну даними. Клієнт і сервер обмінюються криптографічними ключами для встановлення захищеного каналу зв'язку.

Основний принцип роботи SSH полягає у забезпеченні захищеного каналу для адміністрування систем і передачі даних. Після встановлення з'єднання користувач може виконувати різноманітні операції, такі як запуск команд на віддаленому сервері, передача файлів за допомогою протоколів SCP або SFTP, а також налаштування тунелювання для захисту інших видів трафіку (наприклад, HTTP або FTP).

SSH також забезпечує переваги над іншими протоколами завдяки своїм захисним механізмам. Шифрування захищає дані від перехоплення і несанкціонованого доступу, а аутентифікація забезпечує, що тільки авторизовані користувачі можуть отримати доступ до системи. Ключова автентифікація зменшує ризик атак методом перебору паролів.

SSH відіграє критичну роль у сучасних мережах, забезпечуючи безпечний доступ до віддалених систем і захищену передачу даних. Для адміністраторів систем SSH є незамінним інструментом для керування серверами, виконання резервного копіювання, оновлення програмного забезпечення та моніторингу систем. Розробники використовують SSH для доступу до віддалених репозиторіїв коду, розгортання додатків і управління хмарними інфраструктурами.

Крім того, SSH забезпечує захист від багатьох загроз безпеки, таких як перехоплення даних, атаки "людина посередині" (MITM) і несанкціонований доступ. Широке використання SSH у корпоративних та індивідуальних середовищах робить його важливим компонентом для забезпечення безпеки мережевої інфраструктури.

SSH (Secure Shell) є важливим протоколом для безпечного доступу до віддалених систем і передачі даних у мережах. Завдяки своїм сильним захисним механізмам, таким як шифрування та аутентифікація, SSH забезпечує надійний і безпечний спосіб адміністрування серверів і управління мережами. Розуміння та ефективне використання SSH є критично важливим для адміністраторів мереж, розробників і всіх, хто працює з віддаленими системами, оскільки цей протокол забезпечує захист та ефективність, необхідні для сучасних мережевих технологій.

- POP3

POP3 (Post Office Protocol) - це стандартний протокол, який використовується для отримання електронних листів. Протокол поштового з'єднання призначено для обробки запитів на отримання пошти від клієнтських програм. POP3 розроблений для отримання пошти з будь-якої точки доступу до Інтернету. POP3 є найпоширенішим типом облікового запису електронної пошти. POP3 дозволяє завантажувати листи на пристрій, а потім видаляти їх із сервера. Недоліки використання POP3: листи на сервері не зберігаються, включаючи надіслані. Перегляд надісланих листів можливий лише з пристрою, з якого їх надіслали. Усі параметри зберігання листів налаштовуються окремо в поштовому клієнті користувача.

Структура POP3 складається з команд для автентифікації, запитів і обробки повідомлень.

- SMTP

SMTP (Simple Mail Transfer Protocol) - це протокол передачі пошти. Основне завдання сервера SMTP: підтвердження прийому, повернення, оповіщення про помилку або запит на додаткові дані. За допомогою протоколу SMTP перевіряється коректність системи, в яку надсилається вихідне повідомлення. Протокол використовує два порти: 25 та 587. Другий варіант - з захищеним SSL, хоча порт 25 використовується набагато частіше на практиці.

Використання особистого SMTP сервера дуже легко, в мережі є багато доступних інструкцій і фахівців з цього протоколу. Особистий SMTP сервер переважно використовують для організації масових розсилок, особистого листування або транзакційних листів. Всі протоколи забезпечують налагоджену роботу Інтернету, яким ми користуємося щодня. Розуміння роботи мереж на базовому рівні дуже важливе для кожного адміністратора сервера або вебмайстра. Це необхідно для правильного налаштування сервісів в інтернеті, а також для легкого виявлення проблем і вирішення неполадок. А які мережеві протоколи ви знаєте? Додайте їх у коментарях під статтею. Структура SMTP включає команди для ініціалізації з'єднання, передачі даних і завершення сеансу.

На основі порівняльного аналізу мною було обрано протокол HTTP, бо технологія SSL захищає будь-яких користувачів і підвищує довіру до ресурсу, а незалежний орган перевіряє особу власника сертифіката. і некритичні недоліки для мого проєкту.

Таблиця 1.3 -Порівняльний аналіз протоколів передачі даних

Протокол	Опис	Рівень	Переваги	Недоліки	Приклади використання
MAC	Унікальний ідентифікатор пристрою, використовується для ідентифікації пристроїв у локальній мережі	Рівень з'єднання	Унікальність для кожного пристрою, фізична адреса обладнання	Не забезпечує передачу даних між мережами	Локальні мережі
IP	Унікальна адреса для кожного пристрою в мережі, використовується для маршрутизації	Мережевий рівень	Можливість маршрутизації, підтримка IPv4 та IPv6	Залежність від мережевої інфраструктури	Інтернет, хостинг, VPN
ICMP	Протокол для обміну повідомленнями між пристроями	Мережевий рівень	Діагностика мережевих помилок, інформування	Не передає дані	Ping, Traceroute
TCP	Протокол керування передачею даних, забезпечує надійну доставку	Транспортний рівень	Надійність, корекція помилок, впорядкування	Низька швидкість через перевірки	SSH, WWW, FTP
UDP	Протокол для ненадійної передачі даних	Транспортний рівень	Висока швидкість, простота	Втрата пакетів, відсутність перевірки	IP-телефонія, стрімінг відео
HTTP	Основний протокол для передачі гіпертексту	Рівень додатків	Простота, універсальність	Відсутність захисту	Веб-сайти, API

HTTPS	Захищений протокол передачі гіпертексту, використовуючи SSL/TLS	Рівень додатків	Захист даних, підвищена довіра	Додаткові витрати на сертифікати	Веб-сайти з підвищеною безпекою
FTP	Протокол для передачі файлів	Рівень додатків	Простота, універсальність	Небезпека передачі особистих даних	Завантаження файлів на сервер
DNS	Система доменних імен, перетворення імен у IP-адреси	Рівень додатків	Спрощення навігації в мережі	Залежність від серверів DNS	Веб-сайти, домени
SSH	Протокол для безпечного віддаленого керування системою	Рівень додатків	Високий рівень безпеки	Складність налаштування для новачків	Віддалене керування серверами, передача даних
POP3	Протокол для отримання повідомлень електронної пошти	Рівень додатків	Простота, доступність	Відсутність зберігання листів на сервері	Клієнтські поштові програми
SMTP	Протокол передачі електронної пошти	Рівень додатків	Надійність, широке використання	Потреба у налаштуванні для масових розсилок	Електронна пошта, транзакційні листи

На основі проведеного порівняльного аналізу, протокол HTTP (Hypertext Transfer Protocol Secure) є оптимальним вибором для проекту, оскільки забезпечує шифрування переданих даних за допомогою технології SSL/TLS, захищаючи користувачів від прослуховування та маніпуляцій. Крім того, використання сертифікатів SSL підтверджує автентичність сайту, підвищуючи довіру користувачів до ресурсу. Підтримка незалежними органами, які

перевіряють особу власника сертифіката, додає додатковий рівень безпеки.. Таким чином, HTTP забезпечує необхідний рівень безпеки та довіри, що робить його найкращим вибором для реалізації проекту.

2 ТЕХНІЧНЕ ОБГРУНТУВАННЯ ОПТИМАЛЬНОГО ВАРІАНТА ВИРІШЕННЯ ОСНОВНОЇ ЗАДАЧІ З АНАЛІЗОМ СУЧАСНОГО СТАНУ РОЗРОБОК

2.1 Аналіз та структурування телеграм боту

Одним з найголовніших аспектів у створенні телеграм-ботів є організація їх функціональності та контенту. При першому запуску бота користувач повинен мати змогу легко знайти всі необхідні йому функції, оскільки його основною метою є отримання інформації або виконання певних завдань, а не пошук необхідної опції.

Якщо телеграм-бот призначений, наприклад, для отримання інформації про погоду, його структура повинна бути такою, щоб користувач міг отримати доступ до будь-якої функції з головного меню бота. Користувачам не потрібні зайві опції або розділи, коли вони використовують бот для отримання певної інформації. Також у таких ботах функції можуть бути груповані відповідно до їхньої тематики для зручності користувачів [2].

Важливим аспектом у структуруванні телеграм-бота є його навігація. Вона повинна бути легко доступною та інтуїтивно зрозумілою, з кількома основними розділами бота, з яких користувач може перейти до інших. Під час навігації користувач повинен завжди знати, де знаходиться в даний момент, і мати можливість легко переходити до інших розділів. Для цього можна підсвічувати поточний розділ у навігаційному меню.

Найкращим способом показати структуру телеграм-бота користувачеві є використання команд або меню бота. Тут присутні всі доступні функції та опції, і користувач може легко переходити між ними. Це особливо зручно для користувачів, які шукають конкретні функції або опції бота.

До структури телеграм-бота також можна віднести організацію його розділів та функцій. Якщо існують розділи з аналогічними функціями, вони мають мати схожий зовнішній вигляд та організацію контенту. Наприклад, якщо користувач викликає функцію для отримання погоди, структура відповіді може бути подібною до інших функцій, таких як отримання новин або виконання обчислень.

Головною ознакою гарної структури телеграм-бота є можливість користувача легко навігуватися по його функціях та опціях [4]. Для виявлення

популярних функцій можна аналізувати активність користувачів за допомогою різних інструментів, щоб оптимізувати навігацію та зробити її ще зручнішою для користувачів.

2.2 Розробка структури інтерфейсу телеграм бота для вивчення мови програмування Java

Розробка структури телеграм-бота - це процес планування та реалізації навігації між його функціями та можливостями. Основною метою цього процесу є спрощення використання бота користувачем і забезпечення зручного переміщення між його функціями.

Головними вимогами до структури телеграм-бота є:

1. Назви команд та опцій повинні відповідати їхнім функціям.
2. Команди та опції повинні бути розташовані в логічній послідовності.
3. Зв'язок між різними функціями бота має бути зрозумілим.
4. Доступ до головних функцій бота має бути легким та швидким.

Якщо структура бота є логічною та зрозумілою, користувач з більшою вірогідністю буде продовжувати використовувати його, а якщо ні, то можливо буде шукати альтернативні рішення. Тому важливо створити структуру, яка буде зрозумілою та зручною для користувача.

Структура телеграм-бота може бути реалізована у формі деревоподібної або довільної структури, залежно від потреб користувачів та специфіки бота [1]. Наприклад, у випадку бота для вивчення мови програмування Java, може бути застосована деревоподібна структура, де доступ до різних функцій може здійснюватися через "гілки" тематичних розділів. Враховуючи специфіку телеграм-бота для вивчення Java, було вирішено використовувати деревоподібну структуру. З головного меню бота користувач зможе отримати доступ до різних тематичних розділів, таких як основи Java, робота з класами, робота з файлами тощо. В кожному з цих розділів будуть доступні відповідні функції та матеріали для вивчення.

Таким чином, створюючи структуру інтерфейсу телеграм-бота для вивчення Java, важливо врахувати потреби користувачів та забезпечити зручну та логічну навігацію між його функціями.

Було розроблено загальну структуру мого додатку і відображено її у вигляді схеми (див. рис. 2.1).

```
- JavaHelper
  |- HELP.md
  |- mvnw
  |- mvnw.cmd
  |- pom.xml
  |- README.md
  |- src
  |  |- main
  |  |  |- java
  |  |  |  |- ua
  |  |  |     |- JavaHelper
  |  |  |        |- config
  |  |  |        |  |- BotConfig.java
  |  |  |        |  |- BotInitializer.java
  |  |  |        |- service
  |  |  |           |- TelegramBot.java
  |  |  |        |- JavaHelperApplication.java
  |  |  |- resources
  |  |     |- application.properties
  |  |     |- logback.xml
  |- test
  |  |- java
  |     |- ua
  |        |- JavaHelper
  |           |- JavaHelperApplicationTests.java
  |- target
  |- Зовнішні бібліотеки
  |- Scratches and Consoles
```

Рисунок 2.1 – Загальна структура додатку

Нижче наведено роз’яснення до рисунку 2.1

Кореневий каталог:

- JavaHelper: Містить усі файли та каталоги проекту.
- HELP.md: Файл README з інструкціями щодо використання проекту.
- mvnw: Пакетний файл для запуску Maven.
- mvnw.cmd: Пакетний файл для запуску Maven у Windows.
- pom.xml: Файл Maven POM, який описує проект та його залежності.

- README.md: Файл README з загальною інформацією про проект.

Каталог src/main**:**

- java: Містить усі Java-файли проекту.
  - ua.JavaHelper: Містить усі класи та інтерфейси проекту.
    - config: Містить класи конфігурації проекту.
      - BotConfig.java: Клас конфігурації Telegram-бота.
      - BotInitializer.java: Клас ініціалізації Telegram-бота.
    - service: Містить сервісні класи проекту.
      - TelegramBot.java: Клас Telegram-бота.
    - JavaHelperApplication.java: Клас Spring Boot, який є точкою входу в додаток.
- resources: Містить ресурси проекту.
  - application.properties: Файл конфігурації Spring Boot.
  - logback.xml: Файл конфігурації Logback.

Каталог test**:**

- java: Містить тестові файли проекту.
  - ua.JavaHelper: Містить тестові класи проекту.
    - JavaHelperApplicationTests.java: Клас тестового набору JavaHelperApplication.

Каталог target**:**

- Містить файли, створені Maven під час складання проекту.

Зовнішні бібліотеки:

- Містить сторонні бібліотеки, які використовуються в проекті.

Scratches and Consoles:

- Містить тимчасові файли та журнали.

3 РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ КЛІЄНТСЬКОЇ ЧАСТИНИ ДЛЯ КЕРУВАННЯ ПАРАМЕТРАМИ ПРОГРАМНОГО СЕРЕДОВИЩА ВИВЧЕННЯ МОВИ ПРОГРАМУВАННЯ ТЕЛЕГРАМ БОТА

Клієнтська частина телеграм бота це та частина, яку бачить користувач під час взаємодії з ботом. Вона відіграє ключову роль, оскільки саме з неї складається перше враження про бота, а потім користувач починає оцінювати його функціонал та зручність використання.

При розробці клієнтської частини телеграм бота важливо дотримуватися певних стандартів, що є загальноприйнятими у веб-розробці. Наприклад, бот також повинен мати свого роду "дах" та "підвал", де можуть бути розміщені основні команди, інформація про бота, посилання на додаткові ресурси тощо. Кожен функціональний блок бота повинен мати в собі контент, який відповідає його призначенню. Наприклад, якщо бот має функцію пошуку та відображення навчальних матеріалів по Java, то відразу після введення користувачем запиту має бути відображений відповідний контент. Функціональність бота також може включати можливість надсилання запитів на отримання додаткової інформації, взаємодію з користувачем за допомогою кнопок чи команд, а також можливість збереження прогресу навчання та історії запитів.

На рисунку 3.1 наведена схема організації зв’язку, а на рисунку 3.2 блок-схема алгоритму взаємодії бота і користувача.

Рисунок 3.1 – Загальна схема організації зв’язку

Рисунок 3.2 – Блок-схема алгоритму в взаємодії користувача з ботом

Візуальна частина бота і його реакція з боку клієнтського рівня наведені на рисунках нижче.

На рисунку 3.3 зображено введення команди /start яку повинні підтримувати усі існуючі телеграм боти. Також привітальний коментар боту в якому він розповідає для чого він потрібен.

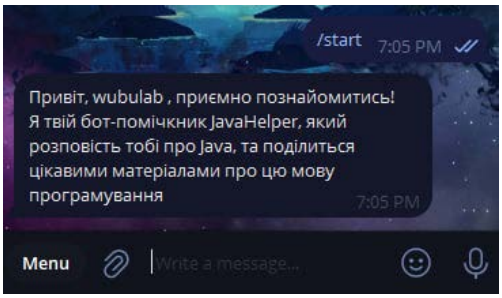


Рисунок 3.3 – Реакція боту на команду start

На рисунку 3.4 зображено меню боту, що випливає коли натискається кнопка меню в лівій частині стрічки вводу повідомлення.

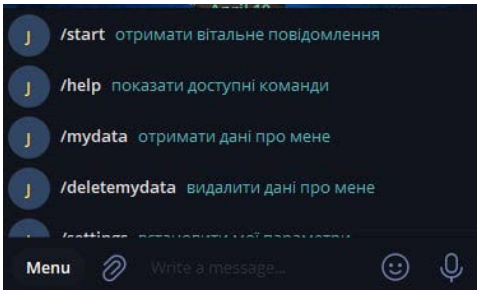


Рисунок 3.4 – Зовнішній вигляд меню телеграм боту

На рисунку 3.5 зображено введення команди /help, після введення якої бот наддасть список доступних команд з їх детальним описом.

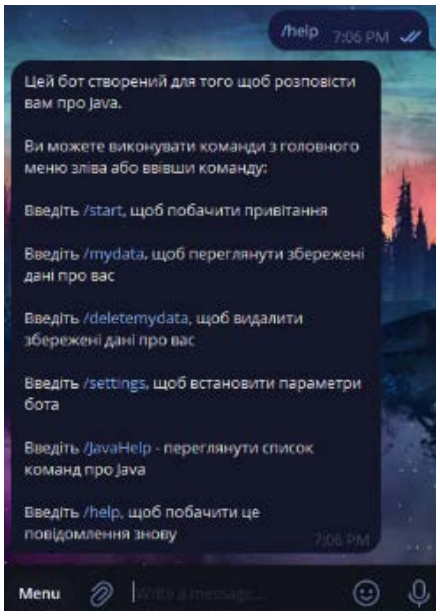


Рисунок 3.5 - Реакція боту на команду help

Одна з найважливіших команд мого боту це команда JavaHelp яка при введені покаже список деяких команд що пов’язані саме з вивченням мови програмування.

На рисунку 3.6 зображено введення команди /JavaHelp, після введення якої бот наддасть список доступних команд з цієї категорії і їх опис.

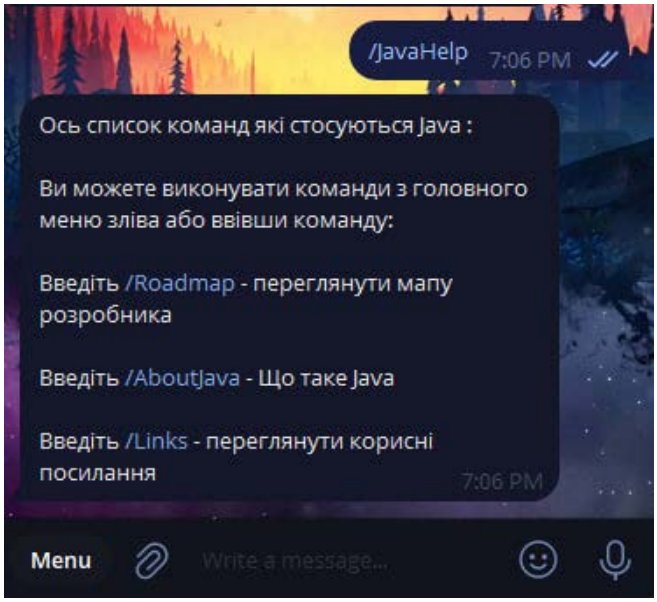


Рисунок 3.6 – Реакція бота на команду JavaHelp

Для цієї команди я зробив окремі три підкоманди, які з’являються лише тоді коли вводиться ця. В повідомленні що виводить бот, також є опис цих команд. Перша з них це команда /Roadmap. При введені цієї команди, бот напише мапу технологій java розробника. Ця мапа представляє собою велику велику схему на якій розписані усі гілки і технології що потрібні для розробника.

Рисунок з повідомленням від боту та власне сама мана наведені нижче.

Рисунок 3.7 – Реакція боту на команду Roadmap

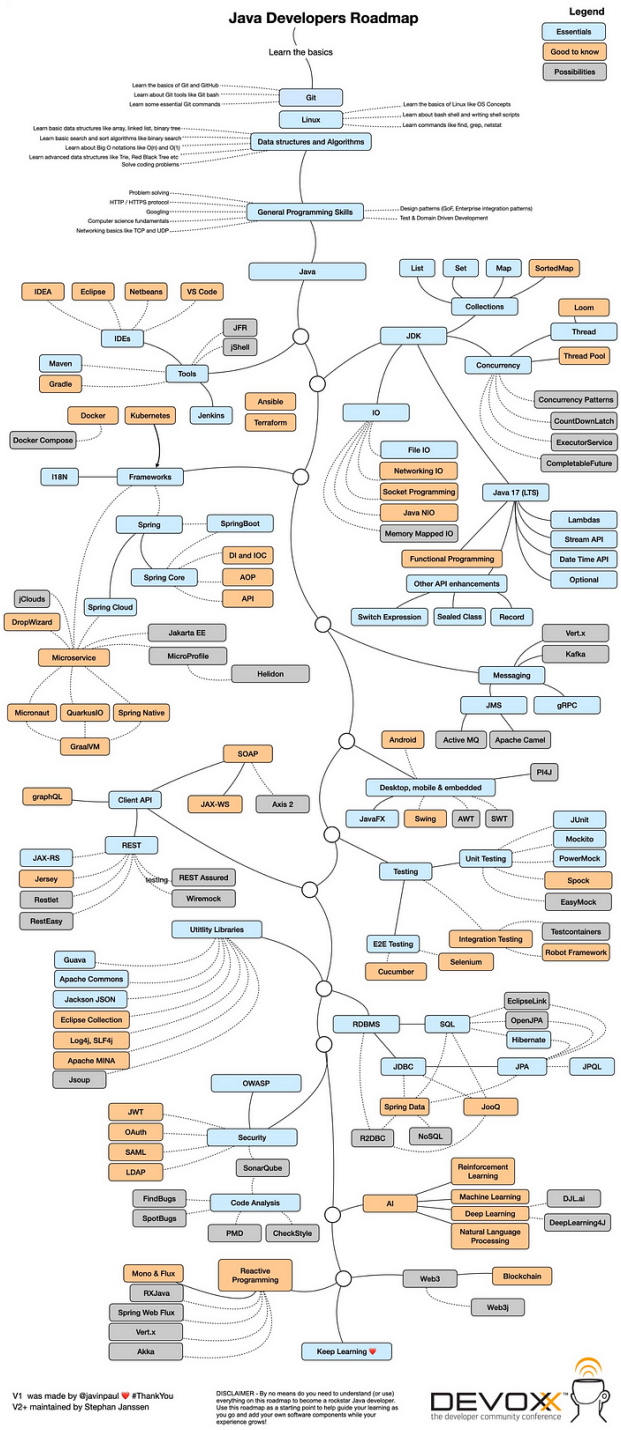


Рисунок 3.8 – Мапа Java розробника

Друга команда це /AboutJava. При введені її бот розповість, що ж таке java та для чого потрібна. І поділиться посиланням яке розкаже про цю мову програмування більш детально.

Реакцію бота наведено на рисунку 3.9

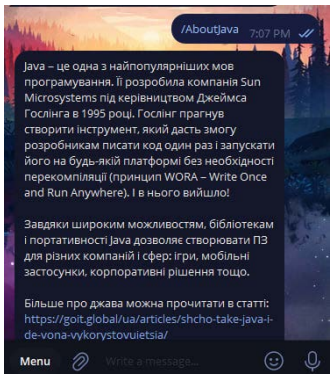


Рисунок 3.9 – Реакція бота на команду AboutJava

Третя і остання ж, команда - це /Links. Як і зазначено в її описі після введення цієї команди боту він надасть перелік корисних посилань які точно знадобляться саме новачку.

Реакцію бота наведено на рисунку 3.10

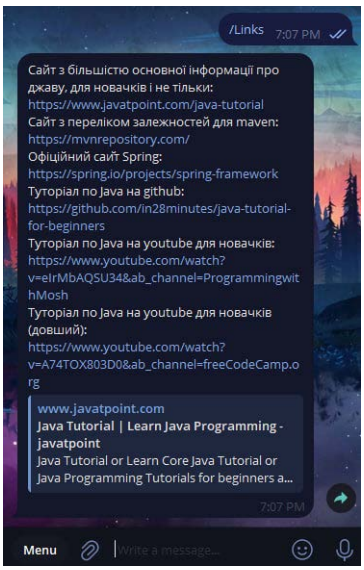


Рисунок 3.10 – Вигляд повідомлення в якому бот надає посилання.

Отже, розробка клієнтської частини телеграм бота передбачає створення зручного та інтуїтивно зрозумілого інтерфейсу, що дозволяє користувачеві легко та ефективно взаємодіяти з ботом та отримувати необхідну інформацію.

4 РОЗРОБЛЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ СЕРВЕРНОЇ ЧАСТИНИ ТЕЛЕГРАМ БОТА

Розробка серверної частини телеграм бота для вивчення мови програмування Java відіграє критичну роль у забезпеченні функціональності та ефективної взаємодії з користувачами. Серверна частина відповідає за обробку запитів від клієнтів, взаємодію з базою даних, а також забезпечує відправку відповідей та виконання основної логіки бота.

Кожний веб-сайт чи бот повинен мати сервер для забезпечення його функціональності та доступності. Сервер дозволяє обробляти запити від користувачів, здійснювати необхідні обчислення, доступ до бази даних, а також забезпечує відповіді на ці запити. У випадку телеграм бота, серверна частина відповідає за обробку повідомлень від користувачів та відправку відповідей.

Для забезпечення зв'язку між клієнтською та серверною частинами телеграм бота було обрано використання протоколу Telegram Bot API. Цей API дозволяє взаємодіяти з телеграм ботом за допомогою HTTP-запитів, що надає можливість відправляти та отримувати повідомлення, обробляти команди користувачів та керувати різними аспектами роботи бота.

Головний файл серверної частини містить ініціалізацію бота та обробку вхідних запитів. У ньому відбувається підключення до Telegram Bot API, ініціалізація необхідних компонентів та запуск сервера. Для забезпечення безперервної роботи бота на сервері використовується постійне відслідковування вхідних повідомлень та команд від користувачів.

Для обробки реєстрації нових користувачів та збереження їхніх даних бот може взаємодіяти з базою даних. Після отримання від користувача необхідної інформації, такої як логін та пароль, серверна частина перевіряє їхню правильність та створює новий запис у базі даних для збереження даних користувача.

Крім того, серверна частина може включати функціональність для надання користувачам доступу до навчального матеріалу з Java, відправлення відповідей на запити користувачів та виконання різних завдань відповідно до їхніх потреб та запитів.

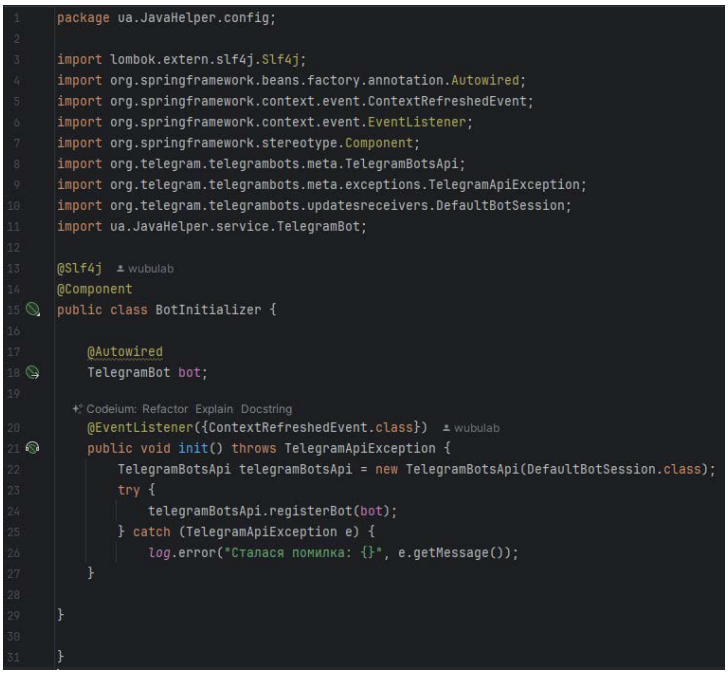
Нижче будуть наведені скріншоти реалізації методів та класів та їх реакції на команди з точки зору бекенду. Про структуру цих класів і мого додатку, в цілому, було згадано в другому розділі.

```java
package ua.JavaHelper;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication  wubulab
public class JavaHelperApplication {

    public static void main(String[] args) {  wubulab

        SpringApplication.run(JavaHelperApplication.class, args);

    }

}
```

Рисунок 4.1 – Точка входу в програму, головний метод

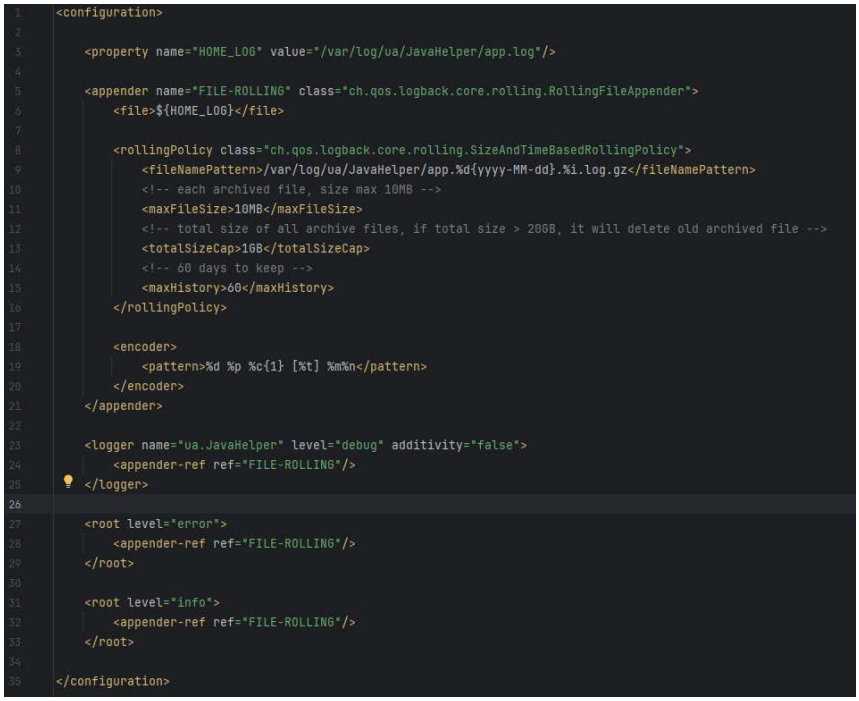
```java
package ua.JavaHelper.config;

import lombok.Data;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.PropertySource;

@Configuration  wubulab
@Data
@PropertySource("/application.properties")
public class BotConfig {

    @Value("JavaHelperBot")
    String botName;

    @Value("6473259672:AAFArOm-wdA69a2qoxMs3ATdMlmYy2Sjcbw")
    String token;
}
```

Рисунок 4.2– Клас що визначає сутності програми та конфігурує бота



```java
package ua.JavaHelper.config;

import lombok.extern.slf4j.Slf4j;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.event.ContextRefreshedEvent;
import org.springframework.context.event.EventListener;
import org.springframework.stereotype.Component;
import org.telegram.telegrambots.meta.TelegramBotsApi;
import org.telegram.telegrambots.meta.exceptions.TelegramApiException;
import org.telegram.telegrambots.updatesreceivers.DefaultBotSession;
import ua.JavaHelper.service.TelegramBot;

@Slf4j  wubulab
@Component
public class BotInitializer {

    @Autowired
    TelegramBot bot;

    Codeium: Refactor  Explain  Docstring
    @EventListener({ContextRefreshedEvent.class})  wubulab
    public void init() throws TelegramApiException {
        TelegramBotsApi telegramBotsApi = new TelegramBotsApi(DefaultBotSession.class);
        try {
            telegramBotsApi.registerBot(bot);
        } catch (TelegramApiException e) {
            log.error("Сталася помилка: {}", e.getMessage());
        }

}

}
```

Рисунок 4.3– Клас що відповідає за ініціалізацію телеграм боту, включає також обробку виключень



```xml
<configuration>

    <property name="HOME_LOG" value="/var/log/ua/JavaHelper/app.log"/>

    <appender name="FILE-ROLLING" class="ch.qos.logback.core.rolling.RollingFileAppender">
        <file>${HOME_LOG}</file>

        <rollingPolicy class="ch.qos.logback.core.rolling.SizeAndTimeBasedRollingPolicy">
            <fileNamePattern>/var/log/ua/JavaHelper/app.%d{yyyy-MM-dd}.%i.log.gz</fileNamePattern>
            <!-- each archived file, size max 10MB -->
            <maxFileSize>10MB</maxFileSize>
            <!-- total size of all archive files, if total size > 20GB, it will delete old archived file -->
            <totalSizeCap>1GB</totalSizeCap>
            <!-- 60 days to keep -->
            <maxHistory>60</maxHistory>
        </rollingPolicy>

        <encoder>
            <pattern>%d %p %c{1} [%t] %m%n</pattern>
        </encoder>
    </appender>

    <logger name="ua.JavaHelper" level="debug" additivity="false">
        <appender-ref ref="FILE-ROLLING"/>
    </logger>

    <root level="error">
        <appender-ref ref="FILE-ROLLING"/>
    </root>

    <root level="info">
        <appender-ref ref="FILE-ROLLING"/>
    </root>

</configuration>
```

Рисунок 4.4- Спеціальний файл обробки виключень та логування програми

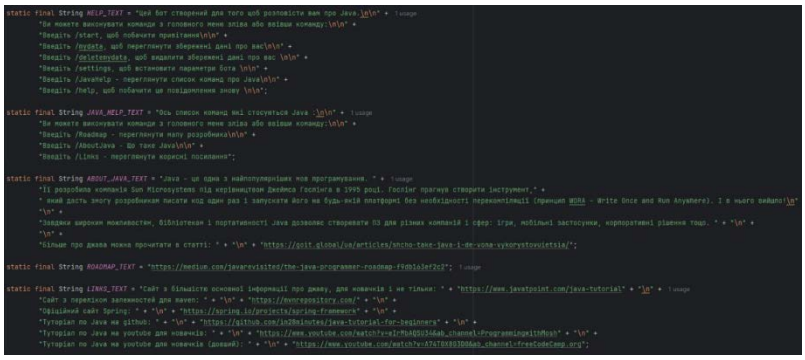


Рисунок 4.5 - Реалізація опису команд в бекенд частині

```
switch (messageText) {
    case "/start":
        startCommandReceived(chatId, update.getMessage().getChat().getFirstName());
        break;

    case "/help":
        sendMessage(chatId, HELP_TEXT);
        break;

    case "/JavaHelp":
        sendMessage(chatId, JAVA_HELP_TEXT);
        break;
```

Рисунок 4.6 – Приклад використання команд з точки зору бекенду

З рисунку 4.6 видно що команда викликає метод відправки повідомлення з параметром який є списком команд що наводились на рисунку 4.5
В програмі реалізована функція розпізнавання недійсної команди. Як це виглядає зображено на рисунку 4.6

```
    default:
        sendMessage(chatId, textToSend: "Вибачте, команда не знайдена");
        break;
}
```

Рисунок 4.7 – Обробка виключення при введені недійсної команди

Отже, розробка серверної частини телеграм бота для вивчення Java передбачає створення надійного та ефективного механізму обробки запитів користувачів, забезпечення взаємодії з базою даних, а також забезпечення безперебійної роботи бота на сервері.

Зв'язок між розробкою веб-системи або телеграм бота та телекомунікаціями може бути відображений через різні аспекти:

Канали зв'язку: Телекомунікаційні мережі забезпечують інфраструктуру для передачі даних між серверами, клієнтськими пристроями та іншими системами. Наприклад, веб-система може використовувати HTTP-запити для взаємодії між клієнтом та сервером через телекомунікаційний канал Інтернету.

Протоколи передачі даних: Технології телекомунікацій визначають протоколи передачі даних, такі як HTTP, які використовуються для комунікації між клієнтами та серверами у веб-системах або телеграм ботах. Ці протоколи визначають правила передачі, формати даних та стандарти взаємодії.

Мережева безпека: Телекомунікаційні технології також відіграють ключову роль у забезпеченні безпеки передачі даних між клієнтами та серверами. Захист мережі, шифрування даних та ідентифікація користувачів - це важливі аспекти, які враховуються при розробці та експлуатації веб-систем або ботів.

Передача медіаданих: У випадку веб-систем, що включають відео- або аудіоматеріали, або телеграм ботів, що передають медіафайли, телекомунікаційні технології забезпечують канали передачі цих даних. Швидкість передачі, якість зв'язку та оптимізація мережі грають критичну роль у задоволенні потреб користувачів.

Взаємодія з користувачем: Технології телекомунікацій також використовуються для взаємодії з користувачами. Наприклад, телеграм бот може використовувати SMS-повідомлення для надсилання сповіщень або підтвердження операцій.

Отже, розробка веб-систем або телеграм ботів безпосередньо пов'язана з технологіями телекомунікацій через використання мережевих протоколів, каналів зв'язку та інфраструктури, що забезпечує передачу даних між системами та користувачами.

5.ОХОРОНА ПРАЦІ

Робота над Телеграм ботом для вивчення об'єктно орієнтованої мови програмування на базі протоколу http відбувалася в приміщені, яке обладнане комп'ютеризованими робочими місцями. На дослідника мали вплив такі небезпечні та шкідливі виробничі фактори:

1. Фізичні: підвищена запиленість та загазованість повітря робочої зони; підвищений рівень шуму на робочому місці; підвищена чи понижена вологість повітря; підвищений рівень статичної електрики; підвищений рівень електромагнітного випромінювання; недостатня освітленість робочої зони.

2. Психофізіологічні: розумове перевантаження; перенапруга аналізаторів; статичне перевантаження.

Відповідно до визначених факторів формуємо рішення щодо безпечного виконання роботи.

5.1. Технічні рішення щодо безпечного виконання роботи

5.1.1. Обладнання робочого місця

Організація робочого місця проектувальника, який користується ПК, повинна відповідати ергономічним вимогам, які стосуються усіх його елементів та їхнього розташування. Робота виконується у сидячому положенні. При цьому необхідно враховувати загальні ергономічні стандарти, а також специфічні особливості трудової діяльності. [2].

Площа, виділена для одного робочого місця з ПК, повинна становити не менш 6 $м^2$, а об'єм – не менше 20 $м^3$ [3].

Природне освітлення має здійснюватись через світлові прорізи, орієнтовані переважно на північ чи північний схід, і забезпечувати коефіцієнт природної освітленості (КПО) не нижче, ніж 1,5%.

Виробничі приміщення повинні обладнуватись шафами для зберігання документів, стелажами, тумбами тощо, з урахуванням вимог до площі приміщень.

У приміщеннях з ПК слід щоденно робити вологе прибирання.

Приміщення із ПК мають бути оснащені аптечками першої медичної допомоги.

При приміщеннях із ПК мають бути обладнані побутові приміщення для відпочинку під час роботи, кімната психологічного розвантаження. В кімнаті психологічного розвантаження слід передбачити встановлення пристроїв для приготування й роздачі тонізуючих напоїв, а також місця для занять фізичною культурою.

Конструкція робочого місця користувача ПК повинна забезпечувати підтримку оптимальної робочої пози з такими ергономічними характеристиками: стопи ніг – на підлозі або на підставці для ніг; стегна – у горизонтальній площині; передпліччя – вертикальні; лікті – під кутом 70-90° до вертикальної площини; зап’ястя зігнуті під кутом не більше 20° щодо горизонтальної площини, нахил голови – 15-20° щодо вертикальної площини.

Висота робочої поверхні стола для відеотерміналу повинна перебувати в межах 680-800 мм, а ширина – забезпечувати можливість виконання операцій у зоні досяжності моторного поля.

Робочий стіл для ПК повинен мати простір для ніг висотою не менш 600 мм, шириною не менш 500 мм, глибиною на рівні колін не менш 450 мм, на рівні витягнутої ноги - не менш 650 мм.

Робоче сидіння (стілець, крісло) користувача ПК повинен мати наступні основні елементи: сидіння, спинку й стаціонарні або знімні підлокітники.

Екран монітора й клавіатура повинні розташовуватися на оптимальній відстані від очей користувача, але не ближче 600 мм, з урахуванням розміру алфавітно-цифрових знаків і символів.

Клавіатуру варто розміщати на поверхні стола або на спеціальної, регульованої по висоті, робочій поверхні окремо від стола на відстані 100-300 мм від краю, найближчого до працівника. Кут нахилу клавіатури повинен бути в межах 5-15°.

При організації праці, пов’язаної з використанням ПК, для збереження здоров’я працюючих, запобігання професійним захворюванням І підтримки працездатності передбачаються внутрішньо змінні регламентовані перерви для відпочинку.

Внутрішньозмінні режими праці й відпочинку містять додаткові нетривалі перерви в періоди, що передують появі об’єктивних і суб’єктивних ознак стомлення й зниження працездатності.

Працюючі з ПК підлягають обов'язковим медичним оглядам: попереднім – при влаштуванні на роботу і періодичним – протягом трудової діяльності. Основними критеріями оцінки придатності до роботи з ПК мають бути показники стану органів зору: гострота зору, показники рефракції, акомодації, стану бінокулярного апарату ока тощо. При цьому необхідно враховувати також стан організму в цілому.

Лінія електромережі для живлення ПК, периферійних пристроїв ПК й устаткування для обслуговування, ремонту й налагодження ПК в приміщенні виконана як окрема групова трипровідна мережа, шляхом прокладання фазових, нульових робочих і нульового захисного провідників. Нульовий захисний провідник використається для заземлення електроприладів.

Металеві труби й гнучкі металеві рукави заземлені. Заземлення відповідає вимогам Правила безпечної експлуатації електроустановок споживачів [4].

Неприпустимим є:

- Використання кабелів і проводів з пошкодженою ізоляцією під час експлуатації, яка втратила захисні властивості.
- Залишення кабелів і проводів з неізольованими провідниками під напругою.
- Використання саморобних подовжувачів, які не відповідають вимогам Правил устрою електроустановок до переносних електропроводів.
- Застосування нестандартного (саморобного) обладнання для опалення приміщень, наприклад, електронагрівального устаткування або ламп накалювання.
- Використання пошкоджених розеток, вимикачів та інших електроприладів, а також ламп зі слідами затемнення або здуття скла.
- Підвішування світильників безпосередньо на струмоведучих проводах, обгортання електроламп і світильників папером, тканиною та іншими горючими матеріалами, а також експлуатація їх без ковпаків (розсіювачів).

- Використання електроапаратури та приладів в умовах, що не відповідають вказівкам або рекомендаціям виробників.

5.2. Технічні рішення з гігієни праці та виробничої санітарії

5.2.1. Мікроклімат

Параметри мікроклімату нормуються в залежності від: періоду року; категорії робіт; технологічного процесу.

Для нормування параметрів мікроклімату календарний рік поділяється на два періоди:

– холодний період – період року, коли середньодобова температура зовні приміщення нижча за +10ºС;

– теплий – коли середньодобова температура зовні приміщення становить +10ºС і вище.

Робота над Телеграм ботом для вивчення об'єктно орієнтованої мови програмування на базі протоколу http за енерговитратами відноситься до категорії 1а [5]. Допустимі параметри мікроклімату для категорії 1а наведені в табл.5.2.1 (відповідно до ДСН 3.3.6.042-99 [6]).

Таблиця 5.2.1 – Параметри мікроклімату

Період року	Допустимі		
	t,°C	W, %	V, м/с
Теплий	22-28	55	0,1-0,2
Холодний	21-25	75	0,1

Для підтримки оптимального рівня мікроклімату в приміщенні передбачено систему опалення та вентиляції повітря. Виміри показників мікроклімату повинні проводитись на початку, в середині і в кінці холодного і теплого періодів року, не менше трьох разів за робочу зміну. При коливаннях показників мікроклімату, пов'язаних з технологічними процесами та іншими причинами, виміри необхідно проводити також при найменших і найбільших значеннях термічних навантажень на працюючих, що мають місце протягом робочої зміни.

5.2.2. Склад повітря робочої зони

В приміщенні, де здійснюється розробка, можливими забруднювачами повітря може бути офісна техніка та пил, який потрапляє ззовні. ГДК шкідливих речовин, які знаходяться в досліджуваному приміщені, наведені в таблиці 5.2.2.

Таблиця 5.2.2 – ГДК шкідливих речовин у повітрі

Назва речовини	ГДК, мг/м^3		Клас небезпечності
	Максимально разова	Середньо добова	
Фенол	0,01	0,01	3
Формальдегід	0,035	0,003	2
Пил нетоксичний	0,5	0,15	4
Озон	0,16	0,03	4

В повітрі зовнішнього природного середовища, як і в повітряному середовищі приміщень завжди є наявною певна кількість заряджених частинок – іонів. Так в 1 $см^3$ чистого зовнішнього повітря міститься близько 1000 негативних іонів і понад 1200 позитивних. Параметри іонного складу повітря на робочому місці, що обладнане ПК, повинні відповідати допустимим нормам (табл.5.2.3).

Таблиця 5.2.3 – Рівні іонізації повітря приміщень при роботі на ПК

Рівні	Кількість іонів в 1 $см^3$	
	n+	n-
Мінімально необхідні	400	600
Оптимальні	1500-3000	3000-5000
Максимально необхідні	50000	50000

Для дотримання нормального складу повітря робочої зони в приміщенні використовують припливно-витяжну вентиляцію. Систематично здійснюють провітрювання через віконні отвори та вологе прибирання. Планується встановлення системи кондиціювання.

5.2.3. Виробниче освітлення

Світло впливає не лише на функцію органів зору, а й на діяльність організму в цілому. При поганому освітленні людина швидко втомлюється, працює менш продуктивно, зростає потенційна небезпека помилкових дій і нещасних випадків. Згідно з статистичними даними, до 5% травм можна

пояснити недостатнім або нераціональним освітленням, а в 20% воно сприяло виникненню травм. Врешті, погане освітлення може призвести до професійних захворювань, наприклад, таких як робоча мнопія (короткозорість), спазм акомодації.

При надмірній яскравості джерел світла та оточуючих предметів може відбутись засліплення працівника. Нерівномірність освітлення та неоднакова яскравість оточуючих предметів призводять до частої переадаптації очей під час виконання роботи і, як наслідок цього – до швидкого втомлення органів зору

Норми освітленості при штучному освітленні та КПО (для III пояса світлового клімату) при природному та сумісному освітленні для виконання роботи зазначені у таблиці 5.2.4 (відповідно до ДБН В.2.5-28-2018 [7]):

Таблиця 5.2.4 - Норми освітленості в приміщенні

Характеристика зорової роботи	Найменший розмір об'єкта розрізнювання	Розряд зорової роботи	Підрозряд зорової роботи	Контраст об'єкта розрізнення з фоном	Характеристика фона	Освітленість, лк		КПО, $e_н$,%			
						Штучне освітлення		Природнє освітлення		Сумісне освітлення	
						Комбіноване	Загальне	Верхнє або верхнє	Бокове	Верхнє або верхнє	Бокове
Дуже високої точності	Від 0,15 до 0,3	II	г	великий	світлий	1000	300	7	2,5	4,2	1,5

Місце праці повинно бути розташоване так, щоб уникнути попадання в очі прямого світла. Щоб уникнути світлових відблисків необхідно використовувати обладнання з матовою поверхнею. Для захисту очей від прямого сонячного світла чи джерел штучного освітлення необхідно застосовувати захисні козирки та жалюзі на вікнах.

Для створення оптимальних умов зорової роботи слід враховувати не лише кількість та якість освітлення, а й кольорове оточення. Так, при світлому пофарбуванні інтер'єру завдяки збільшенню кількості відбитого світла рівень освітленості підвищується на 20 – 40% (при тій же потужності джерел світла), різкість тіней зменшується, покращується рівномірність освітлення.

5.2.4. Виробничий шум

Шум може тимчасово активізувати або постійно пригнічувати психічні процеси організму людини. Фізіологічні та біологічні наслідки можуть проявлятись у формі порушення функцій слуху та інших аналізаторів, зокрема вестибулярного апарату, координуючої функції кори головного мозку, нервової системи, систем травлення і кровообігу.

Індивідуальні особливості людини, пов'язані з різними психологічними реакціями на вплив шуму, суттєво впливають на його сприйняття

Допустимі рівні шуму та вібрації на місцях праці осіб, що працюють з ПК, встановлені санітарними нормами ДсанПіН 3.3.2-007-98 [8], витяг з яких подано в таблиці 5.2.5.

Таблиця 5.2.5 - Допустимі еквівалентні рівні шуму

Вид професійної діяльності, місце праці	Еквівалентні рівні шуму, дБА.
Програмісти	50
Оператори в залах опрацювання інформації на ПК та оператори комп'ютерного набору	65
В приміщеннях для розташування шумних агрегатів	75

Основними заходами боротьби з шумом є усунення або ослаблення причин шуму в самому його джерелі у процесі роботи, використання звукопоглинаючих матеріалів, раціональне планування виробничих приміщень.

5.2.5. Виробничі випромінювання

Оскільки розробка проводилося за допомогою ПК, то на робочому місці працівника можливий підвищений рівень електромагнітного випромінювання.

Основою функціонування організму є дуже слабкі біоелектричні струми, що синхронізують природні біологічні режими. Штучні ЕМП якщо співпадають з частотами біологічних ритмів мозку або біоелектричною активністю серця чи інших органів людини можуть призвести до десинхронізації функціональних процесів в організмі.

Механізм біологічної дії на організм людини полягає як у тепловому, так і нетепловому специфічному ефекті, теплова дія ЕМП проявляються у підвищенні температури тіла, а також локальному, вибірковому нагріванні тканин, органів, клітин унаслідок переходу електромагнітної енергії у теплову.

Допустимі значення параметрів неіонізуючих електромагнітних випромінювань від монітору комп’ютера представлені в табл. 5.2.6.

Таблиця 5.2.6 – Допустимі значення параметрів неіонізуючих електромагнітних випромінювань

Види поля	Допустимі параметри поля		Допустима поверхнева щільність потоку енергії (інтенсивність потоку енергії), Вт/м^2
	за електричною складовою (Е), В/м	за магнітною складовою (Н), А/М	
Напруженість електромагнітного поля, 6 кГц...3 МГц	50	5	
3 МГц...30МГц	2	-	
30 МГц...5 ГГц	-	-	10
Електромагнітне поле оптичного діапазону в ультрафіолетовій частині спектру: УФ-С (220...280 мм)			0,001
УФ-В (280...320 мм)			0,01
УФ-А (320. ..400 мм)			10,0
в інфрачервоній частині спектру: 0,76... 10,0 мкм			35,0.. .70,0
Напруженість електричного поля ВДТ			20 вВ/м

Для зменшення впливу ЕМП від ПК на дослідника, необхідно дотримуватися регламентованих режимів роботи та відпочинку.

5.3. Пожежна безпека

В приміщенні, де здійснювалась робота над Телеграм ботом для вивчення об'єктно орієнтованої мови програмування на базі протоколу http

використовуються тільки негорючі речовини та матеріали у холодному стані, за ступенем вибухопожежної та пожежної небезпеки приміщення відноситься до категорії «Д». За вогнестійкістю будинок відноситься до другої категорії [9]. Робоча зона приміщення віднесена до класу вибухонебезпечності В-IIа та пожежонебезпечності П-IIа, оскільки вибухонебезпечна концентрація пилу і волокон може утворюватися лише внаслідок аварії або несправності.

5.3.1. Технічні рішення системи запобігання пожежі

Система запобігання пожежі включає такі основні напрями:

– запобігання утворенню горючого середовища. Досягається: застосуванням герметичного виробничого устаткування; максимально можливою заміною в технологічних процесах горючих речовин та матеріалів негорючими; обмеженням кількості пожежо- та вибухонебезпечних речовин при використанні та зберіганні, а також правильним їх розміщенням; ізоляцією горючого та вибухонебезпечного середовища; організацією контролю за складом повітря в приміщенні та контролю за станом середовища в апаратах; застосуванням робочої та аварійної вентиляції; відведенням горючого середовища в спеціальні пристрої та безпечні місця; застосуванням в установках з горючими речовинами пристроїв захисту від пошкоджень та аварій; використанням інгібувальних (хімічно активні компоненти, що сприяють припиненню пожежі) та флегматизаційних (інертні компоненти, що роблять середовище негорючим) добавок та ін.

– запобігання виникненню в горючому середовищі (чи внесенню в нього) джерела запалювання. Досягається: використанням устаткування та пристроїв, при роботі яких не виникає джерел запалювання; використанням

електроустаткування, що відповідає за виконанням класу пожежо- та вибухонебезпеки приміщень та зон, груп і категорії вибухонебезпечної суміші; виконанням вимог щодо сумісного зберігання речовин та матеріалів; використанням устаткування, що задовольняє вимоги електростатичної іскробезпеки; улаштуванням блискавкозахисту; організацією автоматичного контролю параметрів, що визначають джерела запалювання; використанням швидкодіючих засобів захисного вимкнення; заземленням устаткування, видовжених металоконструкцій; використанням під час роботи з ЛЗР інструментів, що не допускають іскроутворення; ліквідацією умов для самоспалахування речовин і матеріалів; усуненням контакту з повітрям пірофорних речовин; підтриманням температури нагрівання поверхні устаткування пристроїв, речовин та матеріалів, які можуть контактувати з горючим середовищем нижче гранично допустимої (80 % температури самозаймання).

Можливі причини виникнення пожежі у приміщенні, де проводилося дослідження системи фазового автопідстроювання частоти такі:

– несправна електропроводка (іскріння, перегрів провідників, пересихання електроізоляційних матеріалів);

– залишення без нагляду увімкнутих комп’ютерів, обчислювальної техніки та інших електроприладів.

Для запобігання виникнення пожежі доцільні такі заходи:

– призначення осіб, що відповідальні за пожежну безпеку приміщення;

– систематичне проведення повторних протипожежних інструктажів та занять за програмою пожежно-технічного мінімуму з особами, що відповідальні за пожежну безпеку;

– утримання в справному стані засобів протипожежного захисту.

5.3.2. Технічні рішення системи протипожежного захисту

Протипожежний захист промислових об’єктів забезпечується:

– правильним вибором необхідного ступеня вогнестійкості будівельних конструкцій; правильним об’ємно-планувальним рішенням будівель і споруд; розташуванням приміщень та виробництв з урахуванням вимог пожежної безпеки;

– улаштуванням протипожежних перепон у будівлях, системах вентиляції, опалювальних та кабельних комунікаціях;

– спорудженням протидимного захисту;

– забезпеченням евакуації людей;

– використанням засобів пожежної сигналізації, сповіщування та пожежогасіння;

– організацією пожежної охорони об’єкта;

– засобами, що забезпечують успішне розгортання тактичних дій гасіння пожежі.

Досліджуване приміщення має два переносних (порошкових, водопінних або водяних) вогнегасників з масою заряду вогнегасної речовини 5 кг і більше. Крім того, треба передбачати по одному вуглекислотному вогнегаснику з величиною заряду вогнегасної речовини 3 кг і більше:

– на 20 м^2 площі підлоги в таких приміщеннях: офісні приміщення з ПЕОМ, комори, електрощитові, вентиляційні камери та інші технічні приміщення;

– на 50 м^2 площі підлоги приміщень архівів, машинних залів, бібліотек, музеїв [10].

ВИСНОВКИ

– Проаналізовано сучасний стан галузі вивчення мови програмування Java через телеграм-ботів. Проведено порівняльний аналіз існуючих аналогів, що підтвердив актуальність даної розробки та її потенційну здатність задовольнити потреби користувачів, які цікавляться вивченням Java або бажають поглибити свої знання в цьому напрямку. Обрано метод розробки телеграм-бота власноручно з використанням мови програмування Java через його значні переваги. Визначено основні завдання дослідження, необхідні для успішної розробки теми "Створення телеграм-бота для навчання мови програмування Java".

– Проведено порівняльний аналіз протоколів передачі даних та обрано найкращий з них, для мого проєкту.

– Проведено аналіз структури та функціональних можливостей існуючих телеграм-ботів для навчання програмуванню та розглянуто їхні типи. Розроблено структуру інтерфейсу телеграм-бота для навчання мови програмування Java. Підготовлено навчальну програму та розроблено дизайн з темами для вивчення Java через телеграм-бота. Описано основні алгоритми роботи телеграм-бота та створено блок-схеми з їх описом.

– Проаналізовано та обґрунтовано вибір засобів для реалізації телеграм-бота для навчання мови програмування Java. Для цього обрано наступні інструменти: Java, Telegram Bot API. Розроблено базу даних для телеграм-бота. Оглянуто розроблену клієнтську частину телеграм-бота. Розроблено серверну частину телеграм-бота.

– Проведено тестування розробленого телеграм-бота для навчання Java, під час якого було перевірено коректність його роботи. Зроблено висновок про правильну функціональність програми. Розроблено інструкцію користувача для використання телеграм-бота.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Wikipedia [Електронний ресурс]. https://uk.wikipedia.org/wiki/Java
2. Coursera [Електронний ресурс]. https://www.coursera.org/articles/what-is-java-used-for
3. DOU [Електронний ресурс]. https://dou.ua/lenta/articles/how-to-learn-java/
4. GoIT [Електронний ресурс]. https://goit.global/ua/articles/shcho-take-java-i-de-vona-vykorystovuietsia/
5. М.Ф. Копитко, К.С. Іванків «Основи програмування мовою Java»
6. Cases [Електронний ресурс]. https://cases.media/en/article/vivchennya-movi-programuvannya-java-perevagi-ta-vikliki
7. Брюс Еккель «Філософія Java»
8. Берт Бейтс, Кеті Сьєрра «Вивчаємо Java»
9. Роберт Мартін «Clean Code»
10. ДСТУ OHSAS 18002:2015. Системи управління гігієною та безпекою праці. Основні принципи виконання вимог OHSAS 18001:2007 (OHSAS 18002:2008, IDT). К. : ГП «УкрНИУЦ», 2016. 21 с.
11. ДСТУ 8604:2015 Дизайн і ергономіка. Робоче місце для виконання робіт у положенні сидячи. Загальні ергономічні вимоги. URL: http://online.budstandart. com/ua/catalog/doc-page?id_doc=71028.
12. НПАОП 0.00-7.15-18 Вимоги щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями. URL: http://sop.zp.ua/norm_npaop _0 _ 00-7_ 15-18_01_ua.php.
13. ДБНВ.2.5-27-2006. Захисні заходи електробезпеки в електроустановках будинків і споруд. К. : Мінбуд України, 2006. 154
14. Гігієнічна класифікація праці (за показниками шкідливості і небезпеки факторів виробничого середовища від 12.08.1986 № 4137-86. -

[Електронний ресурс] - Режим доступу: http://zakon4.rada.gov.ua/laws/show/v4137400-86

15. ДСН 3.3.6.042-99 Санітарні норми мікроклімату виробничих приміщень. - [Електронний ресурс] - Режим доступу: http://mozdocs.kiev.ua/view.php?id=1972
16. ДБН В.2.5-28-2018 Природне і штучне освітлення - [Електронний ресурс] - Режим доступу: http://document.ua/prirodne-i-shtuchne-osvitlennja-nor8425.html
17. НПАОП 0.00-7.15-18 Вимоги щодо безпеки та захисту здоров'я працівників під час роботи з екранними пристроями. URL: http://sop.zp.ua/norm_npaop _0 _ 00-7_ 15-18_01_ua.php.
18. ДБН В.1.1-7:2016 Пожежна безпека об`єктів будівництва. Загальні вимоги. URL: http://www.poliplast.ua/doc/dbn _v.1.1-7
19. Наказ Міністерства внутрішніх справ України «Про затвердження Правил експлуатації та типових норм належності вогнегасників». URL: https://zakon.rada.gov.ua/laws/show/z0225-18#Text.

Додаток А

(обов'язковий)

ІЛЮСТРАЦІЙНА ЧАСТИНА

Телеграм бот для вивчення об'єктно орієнтованої мови програмування на базі протоколу https

(назва бакалаврської дипломної роботи)

```
- JavaHelper
  |- HELP.md
  |- mvnw
  |- mvnw.cmd
  |- pom.xml
  |- README.md
  |- src
  |  |- main
  |  |  |- java
  |  |  |  |- ua
  |  |  |     |- JavaHelper
  |  |  |        |- config
  |  |  |        |  |- BotConfig.java
  |  |  |        |  |- BotInitializer.java
  |  |  |        |- service
  |  |  |           |- TelegramBot.java
  |  |  |        |- JavaHelperApplication.java
  |  |  |- resources
  |  |     |- application.properties
  |  |     |- logback.xml
  |- test
  |  |- java
  |     |- ua
  |        |- JavaHelper
  |           |- JavaHelperApplicationTests.java
  |- target
  |- Зовнішні бібліотеки
  |- Scratches and Consoles
```

Рисунок А.1 - Загальна структура додатку

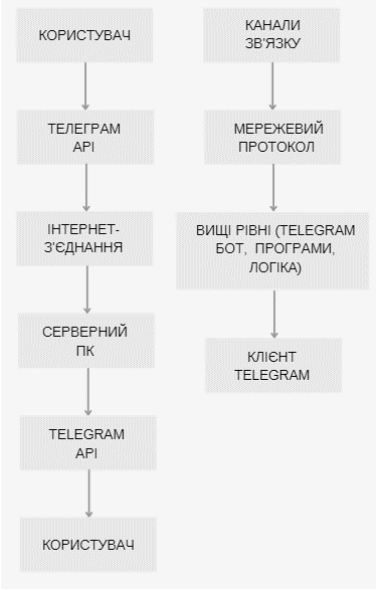


Рисунок А.2 – Загальна схема організації зв’язку

Рисунок А.3 – Блок-схема алгоритму в взаємодії користувача з ботом

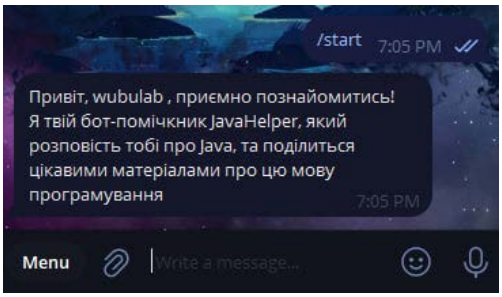


Рисунок А.4 – Реакція боту на команду start

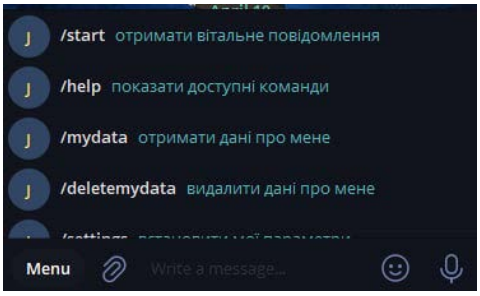


Рисунок А.5 – Зовнішній вигляд меню телеграм боту

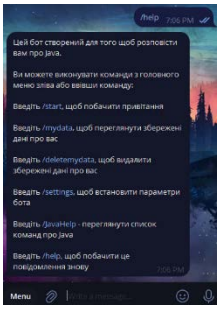


Рисунок А.6 - Реакція боту на команду help

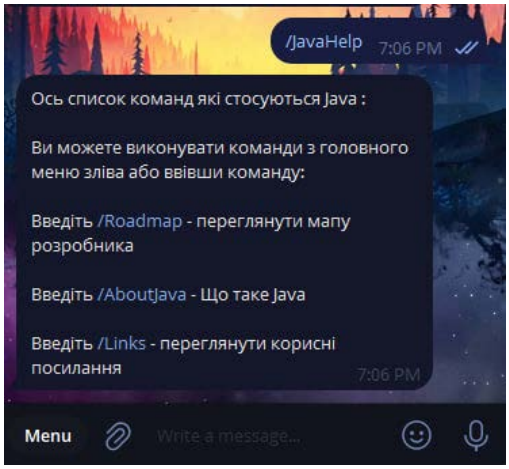


Рисунок А.7 – Реакція бота на команду JavaHelp

Рисунок А.8 – Реакція боту на команду Roadmap

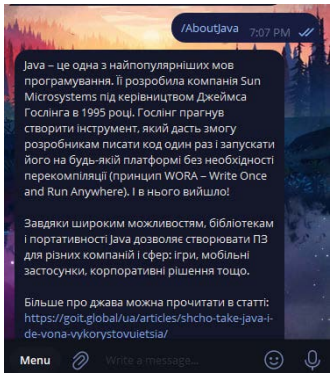


Рисунок А.9 – Реакція бота на команду AboutJava

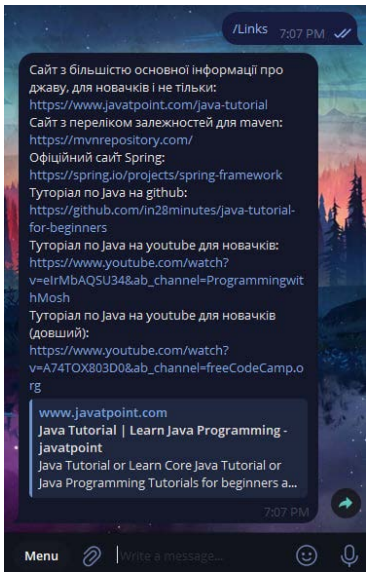


Рисунок А.10 – Вигляд повідомлення в якому бот надає посилання.

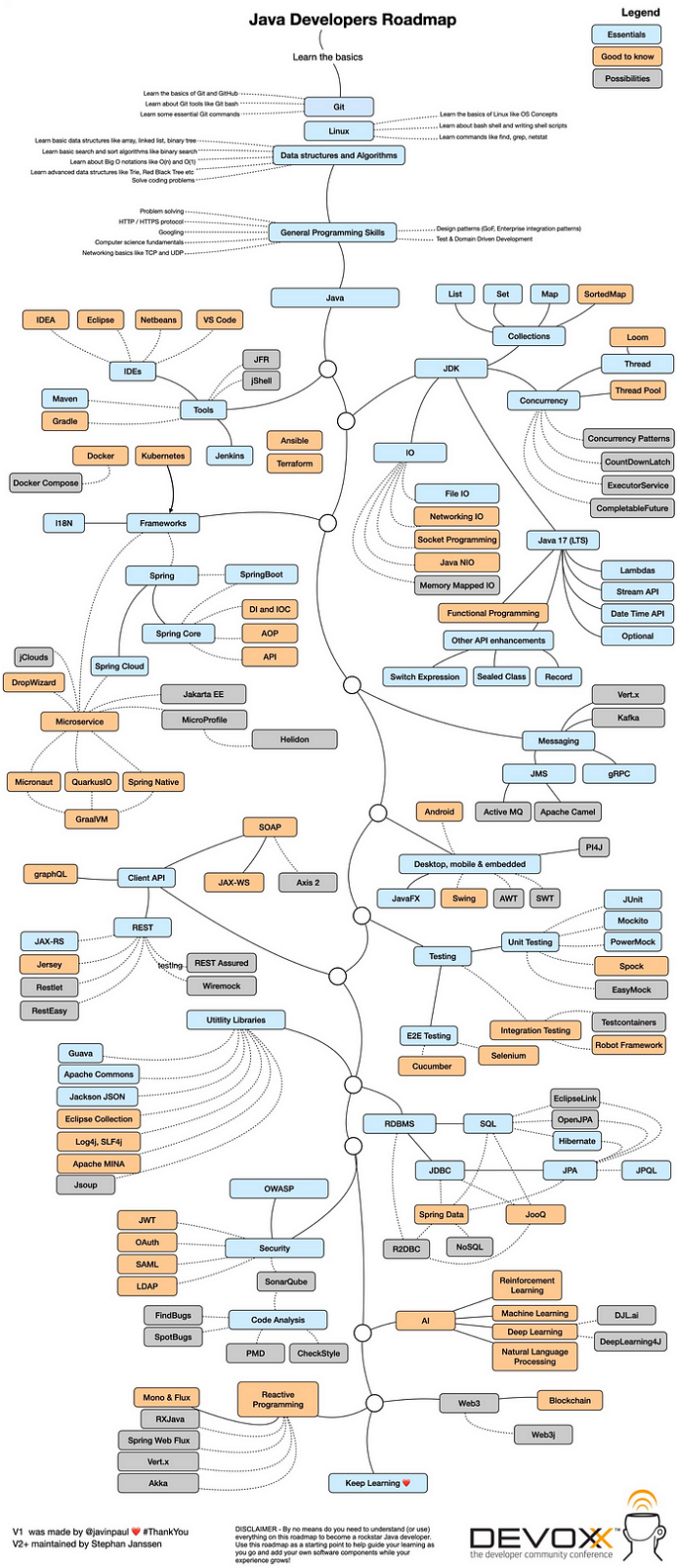


Рисунок А.11 – Мапа Java розробника

```java
package ua.JavaHelper;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication  wubulab
public class JavaHelperApplication {

    public static void main(String[] args) {  wubulab

        SpringApplication.run(JavaHelperApplication.class, args);

    }

}
```

Рисунок А.12 – Точка входу в програму, головний метод

```java
package ua.JavaHelper.config;

import lombok.Data;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.PropertySource;

@Configuration  wubulab
@Data
@PropertySource("/application.properties")
public class BotConfig {

    @Value("JavaHelperBot")
    String botName;

    @Value("6473259672:AAFArOm-wdA69a2qoxMs3ATdMlmYy2Sjcbw")
    String token;
}
```

Рисунок А.13– Клас що визначає сутності програми та конфігурує бота

```java
package ua.JavaHelper.config;

import lombok.extern.slf4j.Slf4j;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.event.ContextRefreshedEvent;
import org.springframework.context.event.EventListener;
import org.springframework.stereotype.Component;
import org.telegram.telegrambots.meta.TelegramBotsApi;
import org.telegram.telegrambots.meta.exceptions.TelegramApiException;
import org.telegram.telegrambots.updatesreceivers.DefaultBotSession;
import ua.JavaHelper.service.TelegramBot;

@Slf4j  wubulab
@Component
public class BotInitializer {

    @Autowired
    TelegramBot bot;

    * Codeium: Refactor | Explain | Docstring
    @EventListener({ContextRefreshedEvent.class})  wubulab
    public void init() throws TelegramApiException {
        TelegramBotsApi telegramBotsApi = new TelegramBotsApi(DefaultBotSession.class);
        try {
            telegramBotsApi.registerBot(bot);
        } catch (TelegramApiException e) {
            log.error("Сталася помилка: {}", e.getMessage());
    }

}

}
```

Рисунок А.14– Клас що відповідає за ініціалізацію телеграм боту, включає також обробку виключень

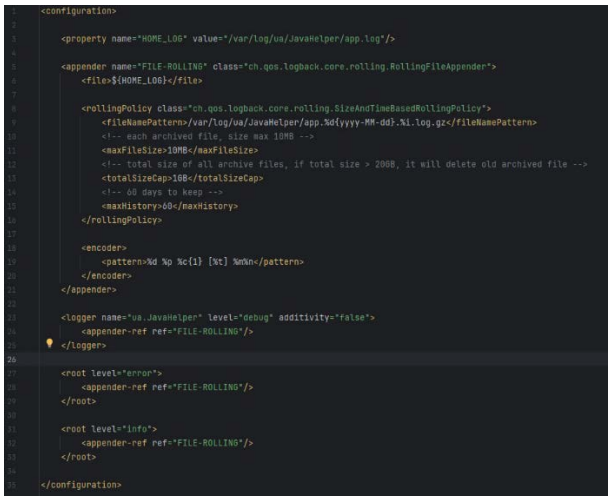


Рисунок А.15- Спеціальний файл обробки виключень та логування програми

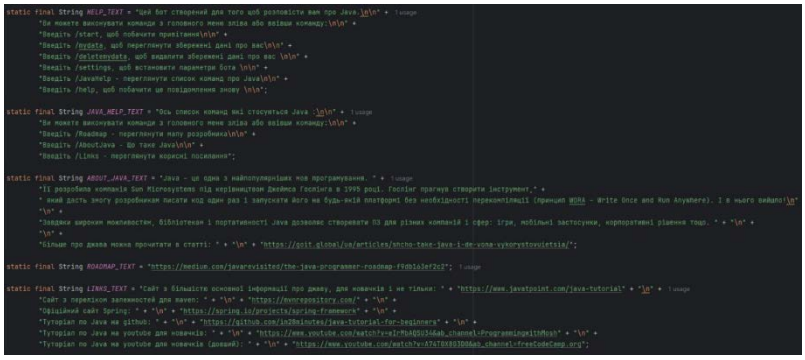


Рисунок А.16 - Реалізація опису команд в бекенд частині

Рисунок А.17 – Приклад використання команд з точки зору бекенду

Рисунок А.18 – Обробка виключення при введені недійсної команди

Додаток Б

ПРОТОКОЛ
ПЕРЕВІРКИ КВАЛІФІКАЦІЙНОЇ РОБОТИ
НА НАЯВНІСТЬ ТЕКСТОВИХ ЗАПОЗИЧЕНЬ

Назва роботи: «Телеграм-бот для вивчення об'єктно-орієнтованої мови програмування на базі протоколу http»

Тип роботи: Бакалаврська дипломна робота
(БДР, МКР)

Підрозділ кафедра інфокомунікаційних систем і технологій, факультет інформаційних електроннихсистем
(кафедра, факультет)

Показники звіту подібності Unicheck

Оригінальність 96,3% Схожість 3,7%

Аналіз звіту подібності (відмітити потрібне):

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

☐ 2. Виявлені у роботі запозичення не мають ознак плагіату, але їх надмірна кількість викликає сумніви щодо цінності роботи і відсутності самостійності її виконання автором. Роботу направити на розгляд експертної комісії кафедри.

☐ 3. 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 _____ Васильківський М.В.
(підпис) (прізвище, ініціали)

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

Автор роботи _____ Канюк Д.В.
(підпис) (прізвище, ініціали)

Керівник роботи _____ Воловик А.Ю.
(підпис) (прізвище, ініціали)

Додаток В
Лістинг застосунку

```
package ua.JavaHelper;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class JavaHelperApplication {

    public static void main(String[] args) {

        SpringApplication.run(JavaHelperApplication.class, args);

    }

}
```

```
package ua.JavaHelper.config;

import lombok.Data;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.Configuration;
import org.springframework.context.annotation.PropertySource;

@Configuration
@Data
@PropertySource("/application.properties")
public class BotConfig {

    @Value("${bot.name}")
    String botName;

    @Value("${bot.token}")
    String token;
}
```

```
package ua.JavaHelper.service;

import lombok.extern.slf4j.Slf4j;
import org.springframework.stereotype.Component;
import org.telegram.telegrambots.bots.TelegramLongPollingBot;
import org.telegram.telegrambots.meta.api.methods.commands.SetMyCommands;
import org.telegram.telegrambots.meta.api.methods.send.SendMessage;
import org.telegram.telegrambots.meta.api.objects.Update;
import org.telegram.telegrambots.meta.api.objects.commands.BotCommand;
import
org.telegram.telegrambots.meta.api.objects.commands.scope.BotCommandScopeD
efault;
import org.telegram.telegrambots.meta.exceptions.TelegramApiException;
import ua.JavaHelper.config.BotConfig;

import java.util.ArrayList;
import java.util.List;

@Slf4j
@Component
public class TelegramBot extends TelegramLongPollingBot {

    final BotConfig config;

    static final String HELP_TEXT = "Цей бот створений для того щоб
розповісти вам про Java.\n\n" +
            "Ви можете виконувати команди з головного меню зліва або ввівши
команду:\n\n" +
            "Введіть /start, щоб побачити привітання\n\n" +
            "Введіть /mydata, щоб переглянути збережені дані про вас\n\n" +
            "Введіть /deletemydata, щоб видалити збережені дані про вас \n\n" +
            "Введіть /settings, щоб встановити параметри бота \n\n" +
            "Введіть /JavaHelp - переглянути список команд про Java\n\n" +
            "Введіть /help, щоб побачити це повідомлення знову \n\n";
```

```
    static final String JAVA_HELP_TEXT = "Ось список команд які стосуються Java :\n\n" +
            "Ви можете виконувати команди з головного меню зліва або ввівши команду:\n\n" +
            "Введіть /Roadmap - переглянути мапу розробника\n\n" +
            "Введіть /AboutJava - Що таке Java\n\n" +
            "Введіть /Links - переглянути корисні посилання";

    static final String ABOUT_JAVA_TEXT = "Java – це одна з найпопулярніших мов програмування. " +
            "Її розробила компанія Sun Microsystems під керівництвом Джеймса Гослінга в 1995 році. Гослінг прагнув створити інструмент," +
            " який дасть змогу розробникам писати код один раз і запускати його на будь-якій платформі без необхідності перекомпіляції (принцип WORA – Write Once and Run Anywhere). І в нього вийшло!\n" +
            "\n" +
            "Завдяки широким можливостям, бібліотекам і портативності Java дозволяє створювати ПЗ для різних компаній і сфер: ігри, мобільні застосунки, корпоративні рішення тощо. " + "\n" +
            "\n" +
            "Більше про джава можна прочитати в статті: " + "\n" +
"https://goit.global/ua/articles/shcho-take-java-i-de-vona-vykorystovuietsia/";

    static final String ROADMAP_TEXT = "https://medium.com/javarevisited/the-java-programmer-roadmap-f9db163ef2c2";

    static final String LINKS_TEXT = "Сайт з більшістю основної інформації про джаву, для новачків і не тільки: " + "https://www.javatpoint.com/java-tutorial" + "\n" +
            "Сайт з переліком залежностей для maven: " + "\n" +
"https://mvnrepository.com/" + "\n" +
            "Офіційний сайт Spring: " + "\n" + "https://spring.io/projects/spring-framework" + "\n" +
            "Туторіал по Java на github: " + "\n" +
"https://github.com/in28minutes/java-tutorial-for-beginners" + "\n" +
            "Туторіал по Java на youtube для новачків: " + "\n" +
"https://www.youtube.com/watch?v=eIrMbAQSU34&ab_channel=Programmingw
```

```
ithMosh" + "\n" +
        "Туторіал по Java на youtube для новачків (довший): " + "\n" +
"https://www.youtube.com/watch?v=A74TOX803D0&ab_channel=freeCodeCamp
.org";
<configuration>

  <property name="HOME_LOG" value="/var/log/ua/JavaHelper/app.log"/>

  <appender name="FILE-ROLLING"
class="ch.qos.logback.core.rolling.RollingFileAppender">
    <file>${HOME_LOG}</file>

    <rollingPolicy
class="ch.qos.logback.core.rolling.SizeAndTimeBasedRollingPolicy">
      <fileNamePattern>/var/log/ua/JavaHelper/app.%d{yyyy-MM-
dd}.%i.log.gz</fileNamePattern>
      <!-- each archived file, size max 10MB -->
      <maxFileSize>10MB</maxFileSize>
      <!-- total size of all archive files, if total size > 20GB, it will delete old
archived file -->
      <totalSizeCap>1GB</totalSizeCap>
      <!-- 60 days to keep -->
      <maxHistory>60</maxHistory>
    </rollingPolicy>

    <encoder>
      <pattern>%d %p %c{1} [%t] %m%n</pattern>
    </encoder>
  </appender>

  <logger name="ua.JavaHelper" level="debug" additivity="false">
    <appender-ref ref="FILE-ROLLING"/>
  </logger>

  <root level="error">
    <appender-ref ref="FILE-ROLLING"/>
  </root>
```

```
      <rollingPolicy
class="ch.qos.logback.core.rolling.SizeAndTimeBasedRollingPolicy">
         <fileNamePattern>/var/log/ua/JavaHelper/app.%d{yyyy-MM-
dd}.%i.log.gz</fileNamePattern>
         <!-- each archived file, size max 10MB -->
         <maxFileSize>10MB</maxFileSize>
         <!-- total size of all archive files, if total size > 20GB, it will delete old
archived file -->
         <totalSizeCap>1GB</totalSizeCap>
         <!-- 60 days to keep -->
         <maxHistory>60</maxHistory>
      </rollingPolicy>

   <root level="info">
      <appender-ref ref="FILE-ROLLING"/>
   </root>
<totalSizeCap>1GB</totalSizeCap>
         <!-- 60 days to keep -->
         <maxHistory>60</maxHistory>
      </rollingPol<rollingPolicy
class="ch.qos.logback.core.rolling.SizeAndTimeBasedRollingPolicy">
         <fileNamePattern>/var/log/ua/JavaHelper/app.%d{yyyy-MM-
dd}.%i.log.gz</fileNamePattern>
         <!-- each archived file, size max 10MB -->
         <maxFileSize>10MB</maxFileSize>
         <!-- total size of all archive files, if total size > 20GB, it will delete old
archived file -->icy>
</configuration>
```

```
package org.telegram.telegrambots.meta;

import java.lang.reflect.InvocationTargetException;

import org.apache.commons.lang3.StringUtils;

import org.telegram.telegrambots.meta.api.methods.updates.SetWebhook;

import org.telegram.telegrambots.meta.exceptions.TelegramApiException;

import org.telegram.telegrambots.meta.generics.BotSession;
```

```
import org.telegram.telegrambots.meta.generics.LongPollingBot;
import org.telegram.telegrambots.meta.generics.TelegramBot;
import org.telegram.telegrambots.meta.generics.Webhook;
import org.telegram.telegrambots.meta.generics.WebhookBot;


public class TelegramBotsApi {
    Class<? extends BotSession> botSessionClass;
    private boolean useWebhook;
    private Webhook webhook;
    public TelegramBotsApi(Class<? extends BotSession> botSessionClass) throws
TelegramApiException {
        if (botSessionClass == null) {
            throw new TelegramApiException("Parameter botSessionClass can not be
null or empty");
        } else {
            this.botSessionClass = botSessionClass;
        }
    }
    public TelegramBotsApi(Class<? extends BotSession> botSessionClass,
Webhook webhook) throws TelegramApiException {
        if (botSessionClass == null) {
            throw new TelegramApiException("Parameter botSessionClass can not be
null or empty");
        } else if (webhook == null) {
            throw new TelegramApiException("Parameter webhook can not be null or
empty");
        } else {
```

```
            this.botSessionClass = botSessionClass;
            this.useWebhook = true;
            this.webhook = webhook;
            this.webhook.startServer();
        }
    }
    public BotSession registerBot(LongPollingBot bot) throws
TelegramApiException {
        if (bot == null) {
            throw new TelegramApiException("Parameter bot can not be null");
        } else if (!this.validateBotUsernameAndToken(bot)) {
            throw new TelegramApiException("Bot token and username can't be
empty");
        } else {
            bot.onRegister();
            bot.clearWebhook();
            BotSession session;
            try {
                session =
(BotSession)this.botSessionClass.getConstructor().newInstance();
            } catch (IllegalAccessException | InvocationTargetException |
InstantiationException | NoSuchMethodException var4) {
                ReflectiveOperationException e = var4;
                throw new TelegramApiException(e);
            }
            session.setToken(bot.getBotToken());
            session.setOptions(bot.getOptions());
```

```
            session.setCallback(bot);
            session.start();
            return session;
        }
    }

    public void registerBot(WebhookBot bot, SetWebhook setWebhook) throws
TelegramApiException {
        if (setWebhook == null) {
            throw new TelegramApiException("Parameter setWebhook can not be
null");
        } else {
            if (this.useWebhook) {
                if (this.webhook == null) {
                    throw new TelegramApiException("This instance doesn't support
Webhook bot, use correct constructor");
                }
                if (!this.validateBotUsernameAndToken(bot)) {
                    throw new TelegramApiException("Bot token and username can't be
empty");
                }
                bot.onRegister();
                this.webhook.registerWebhook(bot);
                bot.setWebhook(setWebhook);
            }
        }
    }
```

```
    private boolean validateBotUsernameAndToken(TelegramBot telegramBot) {
        return StringUtils.isNotEmpty(telegramBot.getBotToken()) &&
StringUtils.isNotEmpty(telegramBot.getBotUsername());
    }
}


    public TelegramBot(BotConfig config) {
        this.config = config;
        List<BotCommand> listOfCommands = new ArrayList<>();
        listOfCommands.add(new BotCommand("/start", "отримати вітальне
повідомлення"));
        listOfCommands.add(new BotCommand("/help", "показати доступні
команди"));
        listOfCommands.add(new BotCommand("/mydata", "отримати дані про
мене"));
        listOfCommands.add(new BotCommand("/deletemydata", "видалити дані
про мене"));
        listOfCommands.add(new BotCommand("/JavaHelp", "переглянути список
команд про java"));
        listOfCommands.add(new BotCommand("/Roadmap", "переглянути мапу
java розробника"));
        listOfCommands.add(new BotCommand("/AboutJava", "Що таке Java"));
        listOfCommands.add(new BotCommand("/Links", "переглянути корисні
посилання"));
        listOfCommands.add(new BotCommand("/settings", "встановити мої
параметри"));

        try {
            this.execute(new SetMyCommands(listOfCommands, new
BotCommandScopeDefault(), null));
        } catch (TelegramApiException e) {
            log.error("Помилка налаштування списку команд бота: {}",
e.getMessage());
        }
    }
```

```
    @Override
    public String getBotUsername() {
        return config.getBotName();
    }

    @Override
    public String getBotToken() {
        return config.getToken();
    }

    @Override
    public void onUpdateReceived(Update update) {
        if (update.hasMessage() && update.getMessage().hasText()) {
            String messageText = update.getMessage().getText();
            long chatId = update.getMessage().getChatId();

            switch (messageText) {
                case "/start":
                    startCommandReceived(chatId,
update.getMessage().getChat().getFirstName());
                    break;

                case "/help":
                    sendMessage(chatId, HELP_TEXT);
                    break;

                case "/JavaHelp":
                    sendMessage(chatId, JAVA_HELP_TEXT);
                    break;

                case "/Links":
                    sendMessage(chatId, LINKS_TEXT);
                    break;

                case "/AboutJava":
```

```
                sendMessage(chatId, ABOUT_JAVA_TEXT);
                break;

            case "/Roadmap":
                sendMessage(chatId, ROADMAP_TEXT);
                break;


            default:
                sendMessage(chatId, "Вибачте, команда не знайдена");
                break;
        }
    }
}

    private void startCommandReceived(long chatId, String userName) {
        String answer = "Привіт, " + userName + " " + ", приємно познайомитись!"
+ "\n" + "Я твій бот-помічкник JavaHelper," +
                " який розповість тобі про Java, та поділиться цікавими матеріалами
про цю мову програмування";
        log.info("Replied to user {}", userName);

        sendMessage(chatId, answer);
    }

    private void sendMessage(long chatId, String textToSend) {

        SendMessage sendMessage = new SendMessage();
        sendMessage.setChatId(String.valueOf(chatId)); //chatId);
        sendMessage.setText(textToSend);
        try {
            execute(sendMessage);
        } catch (TelegramApiException e) {
            log.error("Сталася помилка: {}", e.getMessage());
        }

    }
```

```
}

package ua.JavaHelper.config;

import lombok.extern.slf4j.Slf4j;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.context.event.ContextRefreshedEvent;
import org.springframework.context.event.EventListener;
import org.springframework.stereotype.Component;
import org.telegram.telegrambots.meta.TelegramBotsApi;
import org.telegram.telegrambots.meta.exceptions.TelegramApiException;
import org.telegram.telegrambots.updatesreceivers.DefaultBotSession;
import ua.JavaHelper.service.TelegramBot;

@Slf4j
@Component
public class BotInitializer {

    @Autowired
    TelegramBot bot;

    @EventListener({ContextRefreshedEvent.class})
    public void init() throws TelegramApiException {
        TelegramBotsApi telegramBotsApi = new
TelegramBotsApi(DefaultBotSession.class);
        try {
            telegramBotsApi.registerBot(bot);
        } catch (TelegramApiException e) {
            log.error("Сталася помилка: {}", e.getMessage());
    }

}

}
<configuration>
```

```
   <property name="HOME_LOG" value="/var/log/ua/JavaHelper/app.log"/>

   <appender name="FILE-ROLLING"
class="ch.qos.logback.core.rolling.RollingFileAppender">
      <file>${HOME_LOG}</file>

      <rollingPolicy
class="ch.qos.logback.core.rolling.SizeAndTimeBasedRollingPolicy">
         <fileNamePattern>/var/log/ua/JavaHelper/app.%d{yyyy-MM-
dd}.%i.log.gz</fileNamePattern>
         <!-- each archived file, size max 10MB -->
         <maxFileSize>10MB</maxFileSize>
         <!-- total size of all archive files, if total size > 20GB, it will delete old
archived file -->
         <totalSizeCap>1GB</totalSizeCap>
         <!-- 60 days to keep -->
         <maxHistory>60</maxHistory>
      </rollingPolicy>

      <encoder>
         <pattern>%d %p %c{1} [%t] %m%n</pattern>
      </encoder>
   </appender>

   <logger name="ua.JavaHelper" level="debug" additivity="false">
      <appender-ref ref="FILE-ROLLING"/>
   </logger>

   <root level="error">
      <appender-ref ref="FILE-ROLLING"/>
   </root>
      <rollingPolicy
class="ch.qos.logback.core.rolling.SizeAndTimeBasedRollingPolicy">
         <fileNamePattern>/var/log/ua/JavaHelper/app.%d{yyyy-MM-
dd}.%i.log.gz</fileNamePattern>
         <!-- each archived file, size max 10MB -->
         <maxFileSize>10MB</maxFileSize>
```

```
        <!-- total size of all archive files, if total size > 20GB, it will delete old
archived file -->
        <totalSizeCap>1GB</totalSizeCap>
        <!-- 60 days to keep -->
        <maxHistory>60</maxHistory>
      </rollingPolicy>

   <root level="info">
      <appender-ref ref="FILE-ROLLING"/>
   </root>
<totalSizeCap>1GB</totalSizeCap>
        <!-- 60 days to keep -->
        <maxHistory>60</maxHistory>
      </rollingPol<rollingPolicy
class="ch.qos.logback.core.rolling.SizeAndTimeBasedRollingPolicy">
        <fileNamePattern>/var/log/ua/JavaHelper/app.%d{yyyy-MM-
dd}.%i.log.gz</fileNamePattern>
        <!-- each archived file, size max 10MB -->
        <maxFileSize>10MB</maxFileSize>
        <!-- total size of all archive files, if total size > 20GB, it will delete old
archived file -->icy>
</configuration>
```