МІНІСТЕРСТВО ФІНАНСІВ УКРАЇНИ
ДЕРЖАВНИЙ ПОДАТКОВИЙ УНІВЕРСИТЕТ

ФАКУЛЬТЕТ ФІНАНСІВ ТА ЦИФРОВИХ ТЕХНОЛОГІЙ
КАФЕДРА КОМП'ЮТЕРНИХ ТА ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ І СИСТЕМ

КВАЛІФІКАЦІЙНА РОБОТА

на тему:

«Розробка Telegram-боту мовою Java»

Здобувача вищої освіти першого (бакалаврського) рівня,
IV курсу, групи КБ-20-1, спеціальності 122 «Комп'ютерні науки»,
освітня програма «Інформаційні управляючі системи і технології (за галузями)»

_____ Тутик Олександр Вікторович
(підпис) (прізвище, ім'я, по батькові)

Науковий керівник _____ Горбовий Артур Юліанович д.т.н, професор
(підпис) (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Керівник освітньо-професійної програми _____ Омельчук Антон Анатолійович, к.т.н.
(підпис) (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Допущено до захисту за рішенням кафедри комп'ютерних та інформаційних технологій і систем протокол від «22» травня 2024 № 15

Завідувач кафедри _____ Філоненко Михайло Миколайович, к.ф.-м.н., доцент
(підпис) (прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

Ірпінь – 2024

Реєстрація № 37 27.05.2024 _____
(дата) (підпис відповідальної особи)

Результати перевірки до захисту 22.05.2024 _____
(до захисту, на доопрацювання) (дата) (підпис)

Результат захисту 3 (задовільно) 12.06.2024 _____
(оцінка) (дата) (підпис)

Голова комісії: _____ В.М. Франчук
(підпис)

Члени комісії: _____ М.М. Філоненко
(підпис)

_____ С.М. Грищенко
(підпис)

_____ В.О. Ніжегородцев
(підпис)

БЛАНК ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ

Об'єкти контролю	Діапазон оцінки в балах	Фактична кількість балів	Примітки	Дата, П.І.Б.: керівника, голови комісії членів комісії,секретаря, (підписи)
ОЦІНЮВАННЯ КВАЛІФІКАЦІЙНОЇ РОБОТИ КОМІСІЄЮ ПО ЗАХИСТУ				
1. Зміст випускної роботи (якість теоретичного матеріалу, його зв'язок з практичним, вміння представляти матеріал в логічній послідовності, наявність висновків та пропозицій)	від 0 до 30	19		12.06.2024 Франчук В.М. Філоненко М.М. Грищенко С.М. Ніжегородцев В.О. Суховецька Л.П.
2. Дотримання вимог щодо оформлення	від 0 до 10	8		
3. Вчасне виконання роботи на всіх етапах у встановлені кафедрою терміни	від 0 до 5	3		
4. Рецензент	від 0 до 5	3		
РАЗОМ БАЛІВ	від 0 до 50	33		

Оцінювання кваліфікаційної роботи науковим керівником - 30 балів.

ПЕРЕВЕДЕННЯ ОЦІНКИ ЗА 100-БАЛЬНОЮ ШКАЛОЮ ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалоюЄКТС	Оцінка за національною шкалою	Дата	П.І.Б.: голови, членів комісії, секретаря, (підпис)
63	D	3(задовільно)	12.06.2024	Франчук В.М. Філоненко М.М. Грищенко С.М. Ніжегородцев В.О. Суховецька Л.П.

ПРАВИЛА ПЕРЕВЕДЕННЯ ОЦІНОК ДО НАЦІОНАЛЬНОЇ ШКАЛИ ТА ШКАЛИ ЄКТС

Оцінка за 100-бальною шкалою	Оцінка за шкалою ЄКТС	Оцінка за національною шкалою
90-100	A	5 (відмінно)
80-89	B	4 (добре)
70-79	C	
60-69	D	3 (задовільно)
50-59	E	
35-49	FX	2 (незадовільно)

Факультет фінансів та цифрових технологій
Кафедра комп'ютерних та інформаційних технологій і систем
Галузь знань: 12 «Інформаційні технології»
Спеціальність: 122 «Комп'ютерні науки»
Освітня програма: «Інформаційні управляючі системи і технології (за галузями)»

Затверджую
Завідувач кафедри _____ Філоненко М.М.
«23» жовтня 2023

Завдання на кваліфікаційну роботу
здобувачу вищої освіти
Тутик Олександр Вікторович
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи розробка Telegram-боту мовою Java.
   Затверджена наказом ректора від «20» жовтня 2023 № 1411
2. Строк здачі закінченої роботи «06» травня 2024
3. Мета роботи розробити діючого телеграм бота на мові програмування Java.
   Об'єкт дослідження розробка телеграм-боту.
   Предмет дослідження процес створення телеграм-бота.
   Орієнтовна база дослідження мова програмування Java та сервіс створення чат-ботів телеграм.

4. Вихідні дані до роботи методи та підходи до розробки та створення телеграм ботів на мові програмування Java.

5. Зміст кваліфікаційної роботи (перелік питань за кожним розділом)
   ВСТУП;
   РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ РОЗРОБКИ ТЕЛЕГРАМ- БОТІВ; 1.1. Розглянуті існуючі роботи та проекти з розробки Telegram-боту мовою Java; 1.2. Основні аспекти, які було вже вивчено та розроблено в галузі; 1.3. Функціональні та нефункціональні вимоги до Telegram-боту; 1.4. Основні можливості та функціональність; 1.5. Структуру Telegram-боту; РОЗДІЛ 2. РОЗРОБКА ТА РЕАЛІЗАЦІЯ; 2.1.Використані технології та інструменти для розробки Telegram-бота на Java; 2.2. Етапи розробки телеграм-боту; РОЗДІЛ 3. ЕТАПИ ТЕСТУВАННЯ; 3.1. Підготовка боту до роботи; 3.2. Тестування телеграм-боту. 3.3. Пропозиції що до удосконалення телеграм-боту
   ВИСНОВКИ;
   СПИСОК ВИКОРИСТАНИХ ДЖЕРЕ
   ДОДАТКИ.

6. Календарний план виконання роботи

№ п/п	Назва етапів випускної кваліфікаційної роботи	Строк виконання етапів роботи	
		За планом	Фактично
1	2	3	4
1	Вибір теми, підбір літературних джерел, формування структури кваліфікаційної роботи	До 15.10.2023	До 15.10.2023
2	**І розділ** (розкрити сутність і стан дослідження наукової проблематики роботи, після чого навести її загальну характеристику; огляд літератури за темою і вибір конкретних напрямів досліджень)	29.12.2023	29.12.2023
3	**ІІ розділ** (аналіз експериментальної частини про методики досліджень)	15.02.2024	15.02.2024
4	**ІІІ розділ** (проведені теоретичні і (або) експериментальні дослідження та проектні рішення)	15.04.2024	15.04.2024
5	**Висновки**	01.05.2024	01.05.2024
6	Доопрацювання роботи з урахуванням зауважень наукового керівника	06.05.2024	06.05.2024
7	Отримання відгуку керівника	28.05.2024	28.05.2024
8	Отримання зовнішньої рецензії	30.05.2024	30.05.2024

7. Дата видачі завдання «23» жовтня 2023

8. Керівник випускної кваліфікаційної роботи _____ Горбовий А
(підпис) (прізвище, ініціал

9. Керівник освітньо-професійної програми _____ Омельчук А
(підпис) (прізвище, ініціал

10. Завдання до виконання отримав здобувач вищої освіти _____ Тутик О.В.
(підпис) (прізвище, ініціал

АНОТАЦІЯ

Розробка телеграм-боту на мові Java

Мета роботи полягає в створені телеграм-боту на мові програмування Java

Завданням роботи є:

- Аналіз методів розробки телеграм - бота
- Розробка та реалізація телеграм – бота
- Етап тестування телеграм – бота

Предметом дослідження є комплекс теоретичних та практичних аспектів в створення телеграм-боту.

Об'єкт дослідження є розробка телеграм-боту.

База дослідження виступає мова програмування Java та сервіс створення чат-ботів телеграм.

За результатами дослідження сформульовані методика та етапи реалізації телеграм-боту.

Одержані результати можуть бути використані при розробці телеграм боту.

Кваліфікаційна робота містить 52 сторінка, 1 таблиця, 21 рисунка, список

використаних джерел 44 найменувань, 1 додаток.

Ключові слова: розробка телеграм боту на мові програмування java.

Рік виконання кваліфікаційної роботи 2024

Рік захисту роботи 2024

ABSTRACT

Development of a Telegram bot in Java

The goal of the work is to create a Telegram bot in the Java programming language

The tasks of the work are:

- Analysis of methods of developing telegram bots

- Development and implementation of telegrams - a bot

- The stage of testing telegrams is a bot

The subject of the research is the creation of a Telegram bot in the Java programming language.

The object of research is the development of a Telegram bot.

The basis of the research is the Java programming language and the service for creating Telegram chatbots.

According to the results of the research, the methodology and stages of implementation of the Telegram bot were formulated.

The obtained results can be used in the development of bot telegrams.

The qualification work contains 52 pages, 1 table, 21 figures, a list

44 names of used sources, 1 appendix.

Keywords: development of bot telegrams in java programming language.

The year of performance of qualification work is 2024

The year of job protection is 2024

ЗМІСТ

ВСТУП……9

РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ РОЗРОБКИ ТЕЛЕГРАМ- БОТІВ……12

1.1. Розглянуті існуючі роботи та проекти з розробки Telegram-ботів мовою Java……...12

1.2. Основні аспекти, які було вже вивчено та розроблено в галузі…….14

1.3. Функціональні та нефункціональні вимоги до Telegram-боту……..15

1.4. Основні можливості та функціональність……....16

1.5. Структуру Telegram-боту……..17

РОЗДІЛ 2. РОЗРОБКА ТА РЕАЛІЗАЦІЯ…….……..24

2.1. Використані технології та інструменти для розробки Telegram-бота на Java...……24

2.2. Етапи розробки телеграм-боту.……...25

РОЗДІЛ 3. ЕТАПИ ТЕСТУВАННЯ……..33

3.1. Підготовка боту до роботи…….……33

3.2. Тестування телеграм-боту……...34

3.3. Пропозиції що до удосконалення телеграм-боту……..36

ВИСНОВКИ……..42

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ……44

ДОДАТКИ………48

ВСТУП

Сучасні інформаційні технології стрімко розвиваються, і одним з найбільш популярних засобів комунікації сьогодні є месенджери. Серед них Telegram виділяється завдяки своїй багатофункціональності, безпеці та відкритості для розробників. Однією з важливих складових Telegram є боти — автоматизовані програми, що взаємодіють з користувачами через повідомлення, виконуючи різноманітні завдання від надання інформації до автоматизації процесів.

Актуальність теми дипломної роботи обумовлена швидким зростанням популярності телеграм ботів і необхідністю розробки якісних, надійних та функціональних рішень. Розробка телеграм бота на мові Java дозволяє використовувати потужність і гнучкість цієї мови програмування, що робить можливим створення складних і ефективних програмних продуктів.

Метою даної роботи є розробка телеграм боту на мові Java, який би відповідав сучасним вимогам до функціональності та надійності. Для досягнення цієї мети необхідно вирішити ряд завдань:

Аналіз методів розробки телеграм бота

Розробка телеграм боту стала однією з популярних задач у сфері програмування завдяки широким можливостям і простоті інтеграції з іншими сервісами. Аналіз методів розробки телеграм ботів включає в себе вивчення різних підходів, інструментів та технологій, які дозволяють ефективно створювати та підтримувати такі системи. У цьому розділі ми розглянемо основні методи розробки телеграм ботів, зокрема використання різних мов програмування, бібліотек та фреймворків.

Вибір мови програмування для розробки телеграм бота залежить від різних факторів, таких як вимоги до продуктивності, масштабованість,

доступність бібліотек та фреймворків, а також досвід розробників. Найбільш популярними мовами для розробки телеграм ботів є Python, Java, JavaScript, PHP та Go.

Розробка архітектури телеграм бота є важливим етапом, який визначає основні компоненти системи та їх взаємодію. Правильно спроектована архітектура забезпечує гнучкість, масштабованість і легкість в обслуговуванні програмного продукту. У цьому розділі ми розглянемо основні етапи розробки архітектури телеграм бота на мові Java, а також реалізацію ключових компонентів.

Проведення етапів тестування для забезпечення якості та надійності роботи бота

Тестування є ключовим етапом розробки програмного забезпечення, що дозволяє забезпечити його якість і надійність. У випадку розробки телеграм бота, тестування включає перевірку коректності обробки команд, інтеграції з базою даних і зовнішніми API, а також перевірку продуктивності системи. У цьому розділі ми розглянемо основні етапи тестування телеграм бота на мові Java.

Об'єктом дослідження є процес розробки телеграм бота, а предметом — методи та засоби створення телеграм бота на мові Java.

Методи дослідження включають вивчення наукової та технічної літератури з питань розробки та тестування телеграм ботів, аналіз існуючих підходів та інструментів, практичну реалізацію розробленого рішення та його тестування.

Структура роботи складається з кількох розділів. У першому розділі проведено аналіз методів розробки телеграм ботів, розглянуто основні підходи та інструменти. Другий розділ присвячений розробці архітектури та реалізації телеграм бота на мові Java, включаючи опис використаних технологій та алгоритмів. Третій розділ висвітлює етапи тестування бота, описує застосовані

методи та результати тестування.

Таким чином, дана робота виконана щоб зробити свій внесок у розвиток практики розробки телеграм ботів, пропонуючи ефективні методи та інструменти для створення якісних програмних рішень на мові Java.

РОЗДІЛ 1. АНАЛІЗ МЕТОДІВ РОЗРОБКИ ТЕЛЕГРАМ- БОТУ

1.1. Розглянуті існуючі роботи та проекти з розробки Telegram-ботів мовою Java

Розробка Telegram-ботів мовою Java є популярною завдяки стабільності та багатофункціональності самої мови. Давайте розглянемо кілька існуючих бібліотек та проектів, які допомагають створювати Telegram-ботів на Java. [1]

TelegramBots - Це одна з найпопулярніших Java-бібліотек для створення Telegram-ботів. Вона пропонує простий та інтуїтивно зрозумілий API для інтеграції з Telegram Bot API.

Основні можливості:

- Підтримка всіх типів повідомлень і оновлень.
- Простий механізм для обробки команд.
- Можливість роботи з веб-хуками або довгим опитуванням.

```java
import org.telegram.telegrambots.bots.TelegramLongPollingBot;
import org.telegram.telegrambots.meta.api.objects.Update;

public class MyBot extends TelegramLongPollingBot {

    @Override
    public void onUpdateReceived(Update update) {
        // Ваш код для обробки оновлень
    }

    @Override
    public String getBotUsername() {
        return "YOUR_BOT_USERNAME";
    }

    @Override
    public String getBotToken() {
        return "YOUR_BOT_TOKEN";
    }
}
```

Рис.1.1. Приклад коду чат-боту

Java Telegram Bot API - Ще одна популярна бібліотека для розробки Telegram-ботів на Java. Вона також забезпечує повну підтримку Telegram Bot API.

Основні можливості:

- Підтримка всіх типів повідомлень і оновлень.
- Можливість налаштування вебхуків.
- Проста інтеграція з іншими Java-проектами.

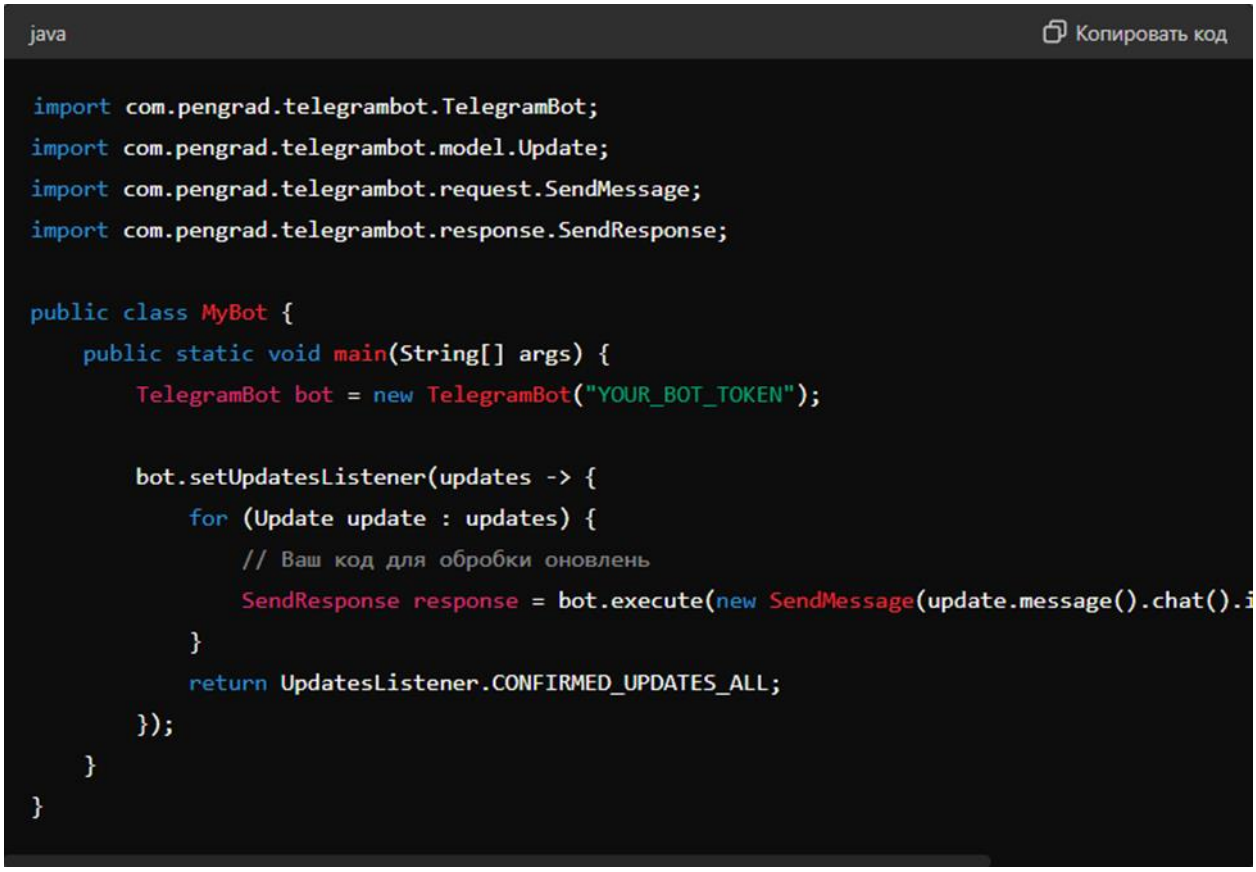


Рис. 1.2. Приклад використання коду на мові Java

Існує кілька підходів до розробки Telegram-ботів на Java, і вибір залежить від специфіки проекту який планується виконувати. TelegramBots і Java Telegram Bot API є двома з найбільш широко використовуваних бібліотек, які надають потужні можливості для інтеграції з Telegram.

1.2. Основні аспекти, які було вже вивчено та розроблено в галузі

Розробка Telegram-ботів на Java є активною галуззю, в якій було досягнуто значного прогресу. Основні аспекти, які було вже вивчено та розроблено, включають наступні напрями в даній сфері.

Бібліотеки та API – на даний час, існує декілька відомих бібліотек, які значно спрощують створення та управління Telegram-ботами.

- TelegramBots - Надає зручний API для взаємодії з Telegram Bot API, підтримує як вебхуки, так і довге опитування (long polling).
- Java Telegram Bot API - Ще одна популярна бібліотека, що дозволяє працювати з усіма типами повідомлень та оновлень Telegram.
- JTelegramBot - Менш відома, але корисна бібліотека для розробки ботів з можливістю обробки команд та інтеграції з іншими Java додатками.

Шаблони та архітектури - Існує ряд розроблених архітектур та шаблонів для створення ботів, до прикладу:

- Шаблон команд - Організація коду навколо команд, які бот може обробляти, з чіткою структурою для додавання нових команд.
- Архітектура на основі подій - Використання підходу на основі подій для обробки оновлень від Telegram, що дозволяє більш гнучке управління логікою бота.
- Інтеграція зі Spring Boot - Багато проектів використовують Spring Boot для створення більш комплексних і легко розгортаваних ботів.

Інтеграція з іншими сервісами - боти часто інтегруються з різними сторонніми сервісами для розширення своїх можливостей в залежності від

звдання самого боту та вимог замовника, абож побажанням самого споживасча. [23].

1.3. Функціональні та нефункціональні вимоги до Telegram-боту

Функціональні та нефункціональні вимоги є ключовими аспектами в розробці будь-якого програмного забезпечення, включаючи Telegram-ботів. Вони визначають, що саме повинно бути реалізовано та які характеристики повинно мати програмне забезпечення для задоволення потреб користувачів та інших зацікавлених сторін.

Функціональні вимоги

Функціонал боту невеликий, а саме повернення надісланих йому повідомлень назад користувачу. Як це повинно відбуватися, користувач надсилає до прикладу ботові повідомлення «привіт», бот в свою чергу повертає дане повідомлення назад користувачу. Ось такий простенький функціонал буде в нашого телеграм-бота.

Нефункціональні вимоги

Нефункціональні вимоги визначають, як повинно працювати програмне забезпечення. Вони включають характеристики системи, які не стосуються конкретних функцій, але є критичними для забезпечення задовільної роботи програми.

До прикладу в моєму випадку мій бот повинен мати нефункціональні вимоги наступні, а саме:

1.Продуктивність - бот повинен обробляти запити користувачів у режимі реального часу з максимальною затримкою не більше 1 секунди.

2.Надійність - Бот повинен бути доступним для користувачів 99.9% часу, включаючи автоматичне відновлення після збоїв.

3. Безпека - бот повинен забезпечувати захист даних користувачів за допомогою шифрування та безпечних методів аутентифікації.

Функціональні вимоги визначають конкретні можливості та функції, які Telegram-бот повинен виконувати для досягнення своїх цілей. Нефункціональні вимоги, навпаки, описують загальні характеристики системи, які важливі для забезпечення якісної та надійної роботи бота. Визначення та дотримання обох типів вимог є критично важливим для успішної розробки та впровадження Telegram-бота.

1.4. Основні можливості та функціональність

Основні можливості та функціональність Telegram-боту залежать від його призначення та цілей, але існує ряд загальних функцій, які часто реалізуються в таких ботах, як правило це ряд базових функцій для початка роботи бота та користувача, так би мовити взаємодії між ними. Розглянемо основні можливості та функціональність, які можуть бути включені до Telegram-бота, зазвичай за стандартом переважно в більшості випадках створення ботів.

Обробка команд - стартові команди**:** Наприклад, /start для початку взаємодії з ботом, /help для отримання допомоги.

Користувацькі команди**:** Специфічні команди для виконання певних дій (наприклад, /news для отримання останніх новин).

Відправка повідомлень **:**

- Текстові повідомлення: Надсилання користувачам текстових повідомлень.
- Медіафайли: Надсилання зображень, відео, аудіо та документів.

- Швидкі відповіді: Використання клавіатури з попередньо визначеними варіантами відповідей для швидкої взаємодії.

Інтеграція з іншими сервісами - API: Взаємодія з зовнішніми API для отримання або надсилання даних (наприклад, інтеграція з API погодних сервісів або платіжних систем). Бази даних**:** Зберігання та отримання даних з баз даних.

Взаємодія з користувачами - Чат-бот**:** Автоматична відповідь на повідомлення користувачів з використанням попередньо налаштованих сценаріїв.

Натуральна мова**:** Використання технологій обробки природної мови (NLP) для розуміння та відповіді на запити користувачів.

Функціональність Telegram-боту – В моєму ботові функціонал не великий. Відразу зайшовши в чат з ботом потрібно буде виконати стандартну команду для всіх телеграм-ботів /start/ ,після чого наш бот розпочинає працювати та виконувати наданий йому функціонал. В моєму боті цей функціонал приставлений наступним чином, це повертання надісланих повідомлень боту користувачем даного чат-боту. Зазвичай в більшості ботів стандартний функціонал по типу включають обробку команд, відправку повідомлень, збір та обробку даних, інтеграцію з іншими сервісами, забезпечення безпеки, автоматизацію завдань, взаємодію з користувачами та інші функції, які дозволяють забезпечити ефективну та зручну взаємодію з користувачами. Реалізація цих можливостей залежить від конкретних потреб та цілей проекту. [20].

1.5. Структуру Telegram-боту

Структура Telegram-боту на мові Java може бути організована різними методами, залежно від складності та вимог до проекту. Однак, за звичай

структура саме Telegram-боту включає кілька основних компонентів, які відповідають за різні аспекти його роботи. До прикладу наведено структури Telegram-боту на Java з використанням бібліотеки TelegramBots які є структурою побудови телеграм боту.

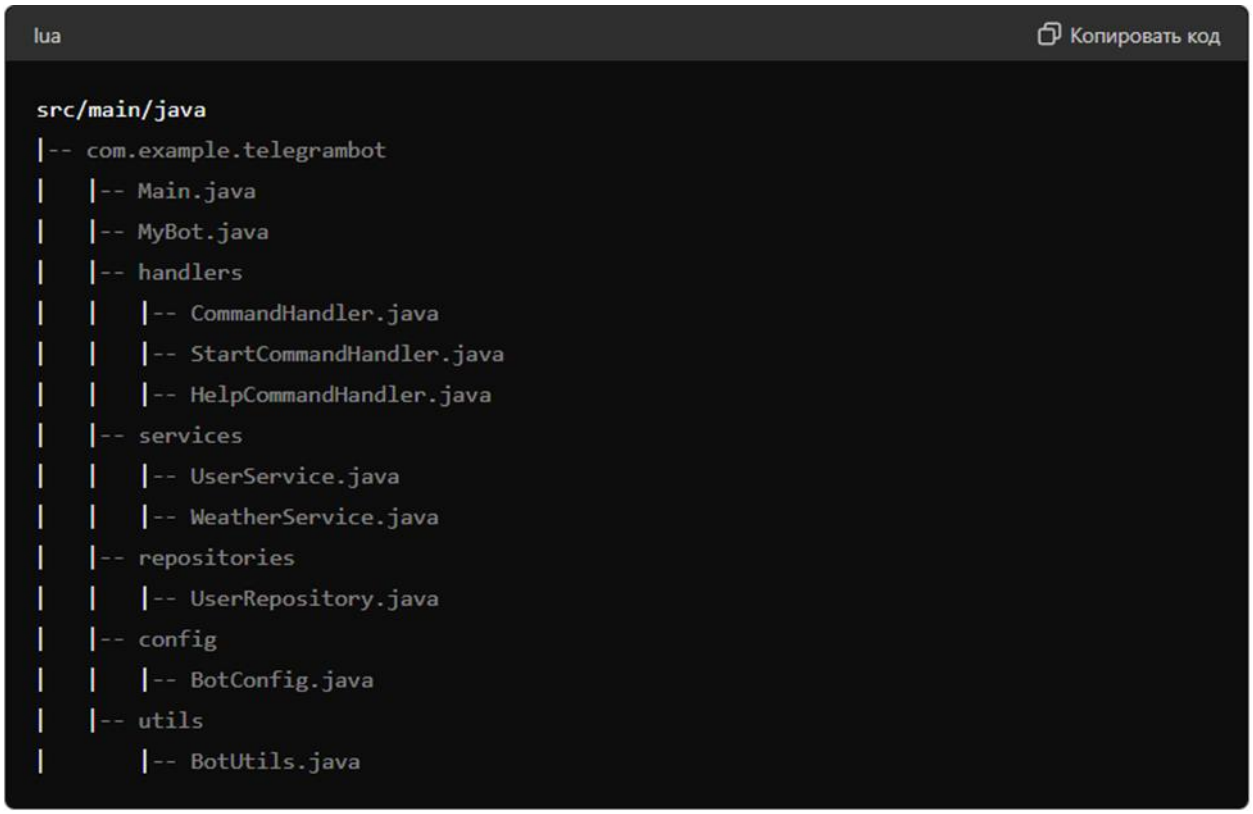


Рис. 1.3. Структура проекту

Main Class (Головний клас) - Цей клас ініціалізує бібліотеку Telegram API та реєструє бота.[6].

```java
package com.example.telegrambot;

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

public class Main {
    public static void main(String[] args) {
        // Ініціалізація контексту Telegram API
        ApiContextInitializer.init();

        // Створення екземпляру TelegramBotsApi
        TelegramBotsApi botsApi = new TelegramBotsApi();
        try {
            // Реєстрація бота
            botsApi.registerBot(new MyBot());
        } catch (TelegramApiException e) {
            e.printStackTrace();
        }
    }
}
```

Рис. 1.4. Main Class (Головний клас)

Bot Class (Клас бота) - Цей клас обробляє отримані оновлення та визначає логіку взаємодії з користувачами.

```java
package com.example.telegrambot;

import org.telegram.telegrambots.bots.TelegramLongPollingBot;
import org.telegram.telegrambots.meta.api.objects.Update;

public class MyBot extends TelegramLongPollingBot {
    @Override
    public void onUpdateReceived(Update update) {
        // Обробка отриманих оновлень від користувачів
        if (update.hasMessage() && update.getMessage().hasText()) {
            String messageText = update.getMessage().getText();
            long chatId = update.getMessage().getChatId();

            if (messageText.equals("/start")) {
                // Обробка команди /start
                new StartCommandHandler().handle(update);
            } else if (messageText.equals("/help")) {
                // Обробка команди /help
                new HelpCommandHandler().handle(update);
            } else {
                // Інші команди або повідомлення
            }
        }
    }

    @Override
    public String getBotUsername() {
        return "YOUR_BOT_USERNAME"; // Вкажіть ім'я вашого бота
    }

    @Override
    public String getBotToken() {
        return "YOUR_BOT_TOKEN"; // Вкажіть токен вашого бота
    }
}
```

Рис. 1.4. Bot Class (Клас бота)

Command Handlers (Обробники команд) - Обробники команд в Telegram-боті на Java відповідають за виконання певних дій у відповідь на команди користувачів. Вони допомагають структурувати код, зробити його більш організованим та легким для підтримки і розширення. Розглянемо детальніше, як можна реалізувати обробники команд.

java Копировать код

```java
package com.example.telegrambot.handlers;

import org.telegram.telegrambots.meta.api.objects.Update;

public interface CommandHandler {
    void handle(Update update);
}
```

Рис. 1.5. Command Handlers (Обробники команд)

Start Command Handler.java - це клас, який реалізує обробку команди /start в Telegram-боті. Ця команда зазвичай використовується для привітання нового користувача та надання основної інформації про бота.

java Копировать код

```java
package com.example.telegrambot.handlers;

import org.telegram.telegrambots.meta.api.methods.send.SendMessage;
import org.telegram.telegrambots.meta.api.objects.Update;
import org.telegram.telegrambots.meta.exceptions.TelegramApiException;
import com.example.telegrambot.MyBot;

public class StartCommandHandler implements CommandHandler {
    @Override
    public void handle(Update update) {
        // Отримання chatId для відправки повідомлення
        long chatId = update.getMessage().getChatId();

        // Створення об'єкта повідомлення з текстом привітання
        SendMessage message = new SendMessage()
            .setChatId(chatId)
            .setText("Welcome to the bot! Here you can find various functionalities. Use /hel

        try {
            // Виконання надсилання повідомлення
            new MyBot().execute(message);
        } catch (TelegramApiException e) {
            e.printStackTrace();
        }
    }
}
```

Рис. 1.6. Реалізація класа Start Command Handler

HelpCommandHandler.java - це клас, який реалізує обробку команди /help в Telegram-боті. Ця команда зазвичай використовується для надання користувачам інформації про доступні команди та можливості бота.

java Копировать код

```java
package com.example.telegrambot.handlers;

import org.telegram.telegrambots.meta.api.methods.send.SendMessage;
import org.telegram.telegrambots.meta.api.objects.Update;
import org.telegram.telegrambots.meta.exceptions.TelegramApiException;
import com.example.telegrambot.MyBot;

public class HelpCommandHandler implements CommandHandler {
    @Override
    public void handle(Update update) {
        // Отримання chatId для відправки повідомлення
        long chatId = update.getMessage().getChatId();

        // Створення об'єкта повідомлення з текстом допомоги
        String helpText = "Here are the available commands:\n"
                        + "/start - Start the bot\n"
                        + "/help - Show this help message\n"
                        + "/weather - Get the current weather\n"
                        + "/info - Get information about the bot";

        SendMessage message = new SendMessage()
            .setChatId(chatId)
            .setText(helpText);

        try {
            // Виконання надсилання повідомлення
            new MyBot().execute(message);
        } catch (TelegramApiException e) {
            e.printStackTrace();
        }
    }
}
```

Рис. 1.7. Реалізація HelpCommandHandler

Ці класи обробляють специфічні команди, які надходять від користувачів.

Data Access Layer (Шар доступу до даних) - це частина архітектури додатка, яка відповідає за взаємодію з базою даних або іншими джерелами даних. DAL забезпечує абстракцію та інкапсуляцію доступу до даних, дозволяючи іншим частинам додатка взаємодіяти з даними через визначений API, не турбуючись про деталі реалізації зберігання даних.

```java
package com.example.telegrambot.repositories;

public class UserRepository {
    // Логіка для доступу до бази даних користувачів, наприклад, методи збереження та отриман
}
```

Рис. 1.8. Data Access Layer (Шар доступу до даних)

Configuration (Конфігурація) в контексті розробки програмного забезпечення - це процес налаштування параметрів та властивостей програми для забезпечення її правильної роботи в різних середовищах. Конфігурація включає налаштування підключення до баз даних, параметри безпеки, логування, налаштування залежностей та інші критичні для функціонування додатка параметри.

```java
package com.example.telegrambot.config;

public class BotConfig {
    public static final String BOT_TOKEN = "YOUR_BOT_TOKEN";
    public static final String BOT_USERNAME = "YOUR_BOT_USERNAME";
    // Інші конфігураційні параметри
}
```

Рис. 1.9. - Configuration (Конфігурація)

Utility Classes (Утилітарні класи) - це класи, які містять статичні методи і використовуються для виконання загальних, часто використовуваних завдань або для спільного використання в різних частинах програми. Вони не містять стану (не мають полів) і не потребують створення екземплярів.[20].

```java
package com.example.telegrambot.utils;

public class BotUtils {
    // Корисні утиліти, наприклад, методи для форматування тексту або обробки даних
}
```

Рис. 1.10. Utility Classes (Утилітарні класи)

Ця структура забезпечує чітке розділення відповідальностей та полегшує подальше розширення функціональності бота. Кожен компонент має своє місце та чітко визначену роль, що сприяє підтримуваності та масштабованості даного проекту проекту, та надає можливості для його розвитку. .[37].

РОЗДІЛ 2. РОЗРОБКА ТА РЕАЛІЗАЦІЯ

2.1. Використані технології та інструменти для розробки Telegram-бота на Java

Для розробки Telegram-бота на Java використовувати різні технології та інструменти, які допомагають забезпечити ефективну розробку боту. Основні технології та інструменти які використовував при розробці це ж звичайно мова програмування Java. Є популярною мовою програмування, яка широко використовується для розробки серверних додатків, включаючи Telegram-ботів. Вона може запропонувати багатий набір бібліотек і фреймворків, що спрощує розробку телеграм ботів вцілому. Також звичайно використовував бібліотеку «Telegram Bot API» - дозволяє розробникам створювати боти, які взаємодіють з користувачами на платформі Telegram. Це RESTful API, що дозволяє отримувати оновлення, повідомлення та події від користувачів і відправляти відповіді на запити користувачів.

Основні можливості Telegram Bot API

1. Отримання оновлень - бот може отримувати повідомлення від користувачів через методи getUpdates або за допомогою вебхуків.
2. Відправка повідомлень - бот може відправляти текстові повідомлення, зображення, файли, відео та інші медіафайли.
3. Відправка повідомлень - бот може відправляти текстові повідомлення, зображення, файли, відео та інші медіафайли.
4. Відправка повідомлень - бот може відправляти текстові повідомлення, зображення, файли, відео та інші медіафайли.
5. Відправка повідомлень: Бот може відправляти текстові повідомлення, зображення, файли, відео та інші медіафайли.

6. Callback Queries: Обробка подій з кнопок, які користувач натискає в інтерактивних повідомленнях.[33].

2.2. Етапи розробки телеграм-боту

Першочерговим етапом розробки щоб створить бота потрібно звернутися до сервісу телеграма а саме «BotFather». Що взагалі це за такий сервіс і що він виконує. Це офіційний бот від Telegram, призначений для створення та управління іншими ботами на платформі Telegram. Він надає інтерфейс для взаємодії з Telegram Bot API, дозволяючи розробникам легко створювати, налаштовувати і керувати своїми ботами. Призначення BotFather це бути основним інструментом для розробників, які хочуть створити нових ботів у Telegram.

Основні функції BotFather включають.

1. Створення нового бота - BotFather генерує унікальний токен доступу, який використовується для автентифікації вашого бота в Telegram API.

2. Налаштування бота - ви можете змінювати основні параметри бота, такі як ім'я, опис, фотографію профілю та інші.

3. Управління командами - додавання, видалення та налаштування команд, які може виконувати бот.

4. Забезпечення безпеки - BotFather дозволяє встановлювати права доступу та обмеження для бота.

Як працювати з BotFather, першочергово знайти цього бота в телеграмі та виконати команду /start/, за тім перейти до створення бота натиснувши на команду /newbot/ , наступним етапом дати назву боту після надання назви бота, отримати токен боту, оскільки він необхідний для взаємодії вашого бота з Telegram Bot API зберегти його.[32].

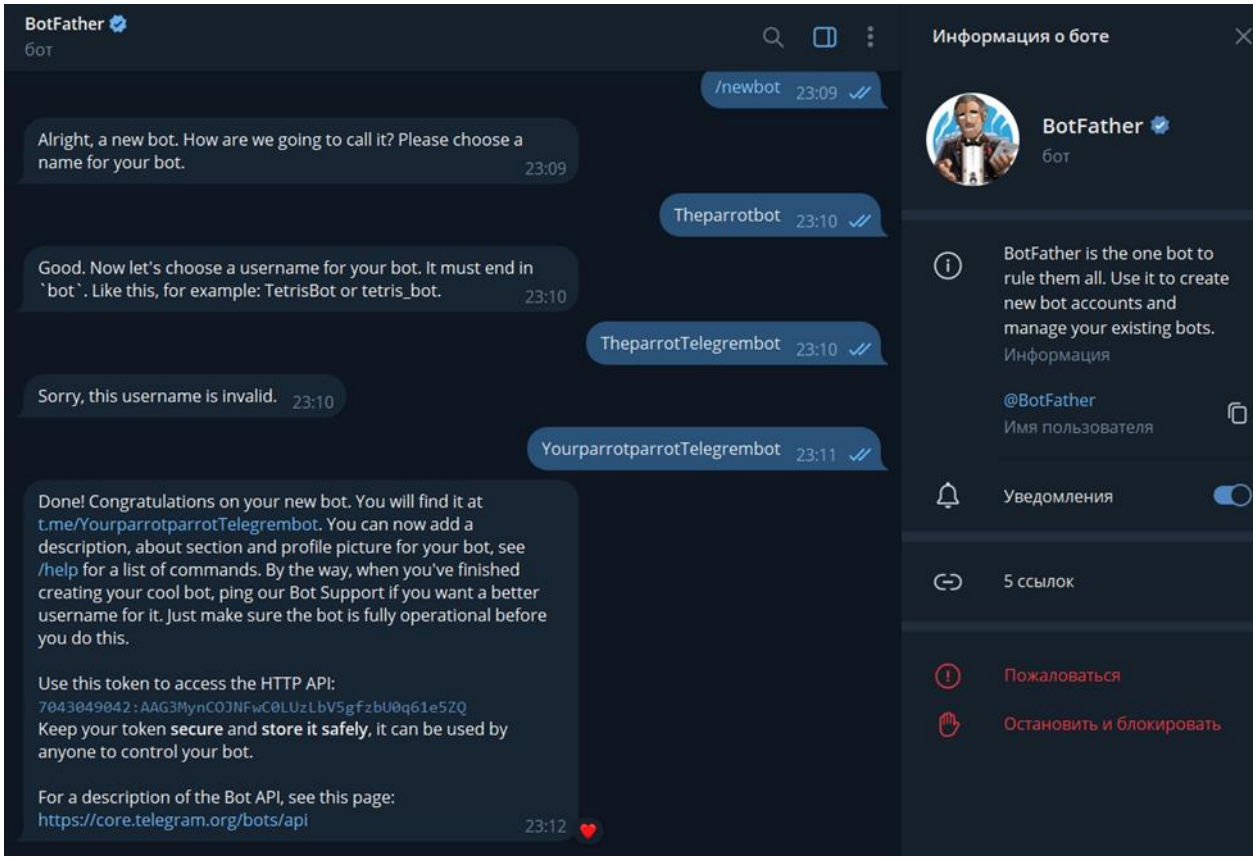


Рис. 2.1. Створення боту за допомогою BotFather

На даному зображенні , наданий діалог створення мого боту за допомогою телеграм сервісу тобто «BotFather». При надані боту імені в моєму випадку мій бот має ім’я «YourparrotparrotTelegrembot». Після надання імені надається токен боту: «7043049042:AAG3MynCOJNFwC0LUzLbV5gfzbU0q61e5ZQ».

Після створення телеграм боту в самому просторі телеграму можна перейти до створення самого програмного забезпечення на мові Jawa. В цьому нам допоможе середовище розробки IntelliJ IDEA.

IntelliJ IDEA - комерційне інтегроване середовище розробки для різних мов програмування (Java, Python, Scala, PHP) від компанії JetBrains. Система поставляється у вигляді урізаної по функціональності безкоштовної версії «Community Edition» і повнофункціональної комерційної версії «Ultimate

Edition», для якої активні розробники відкритих проектів мають можливість отримати безкоштовну ліцензію.

Для початку потрібно створити проект в середовищі розробки як ми вже раніше згадували, в нашому випадку це IntelliJ IDEA.

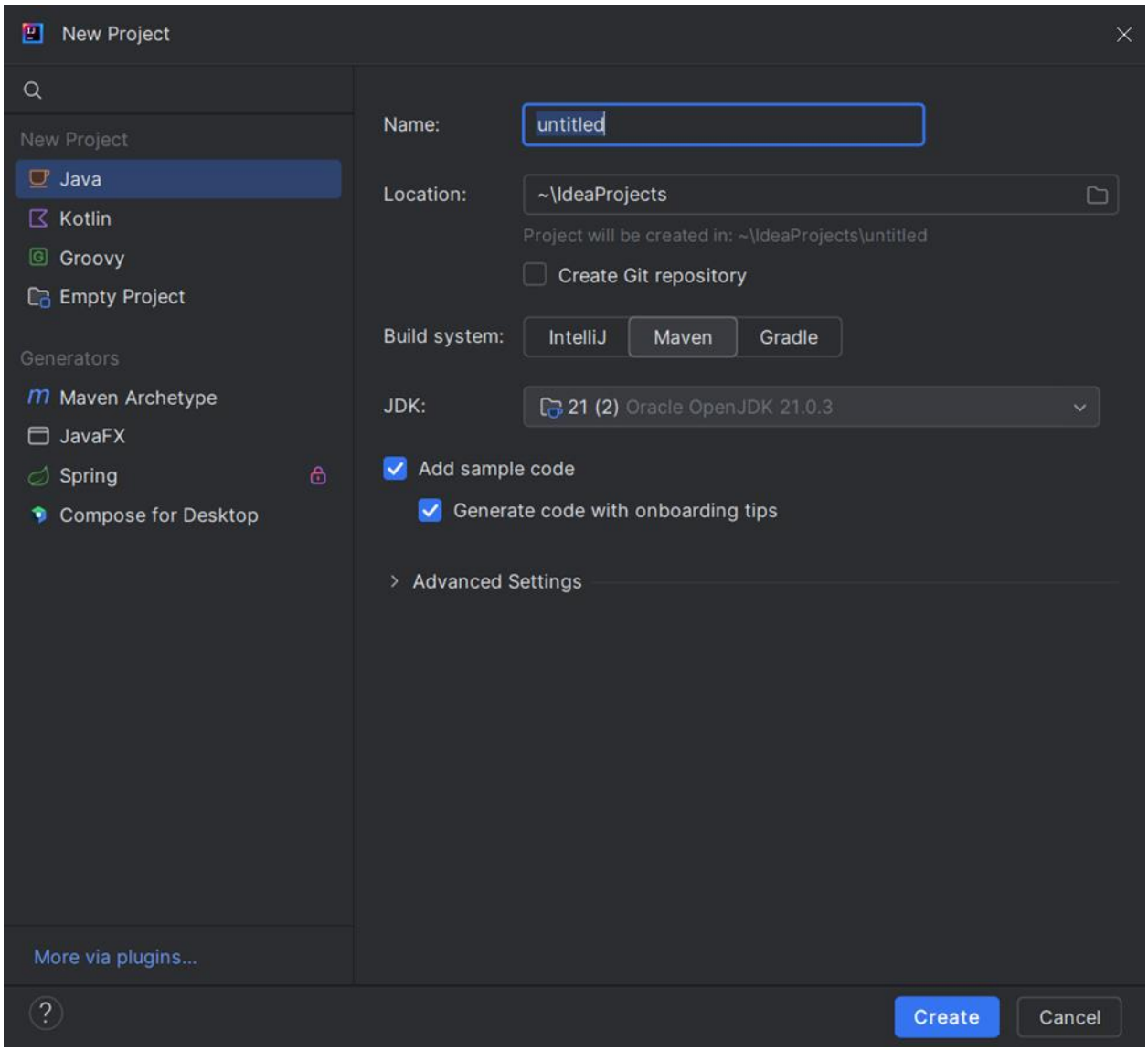


Рис. 2.2. Створення проекту в середовищі розробки IntelliJ IDEA

Наступним етапом буде створення залежності в методі «M pom.xml», для об’єднання проекту з телеграм ботом, щоб код програми міг працювати в нашому телеграм боті та виконувати коректно свій функціонал роботу.

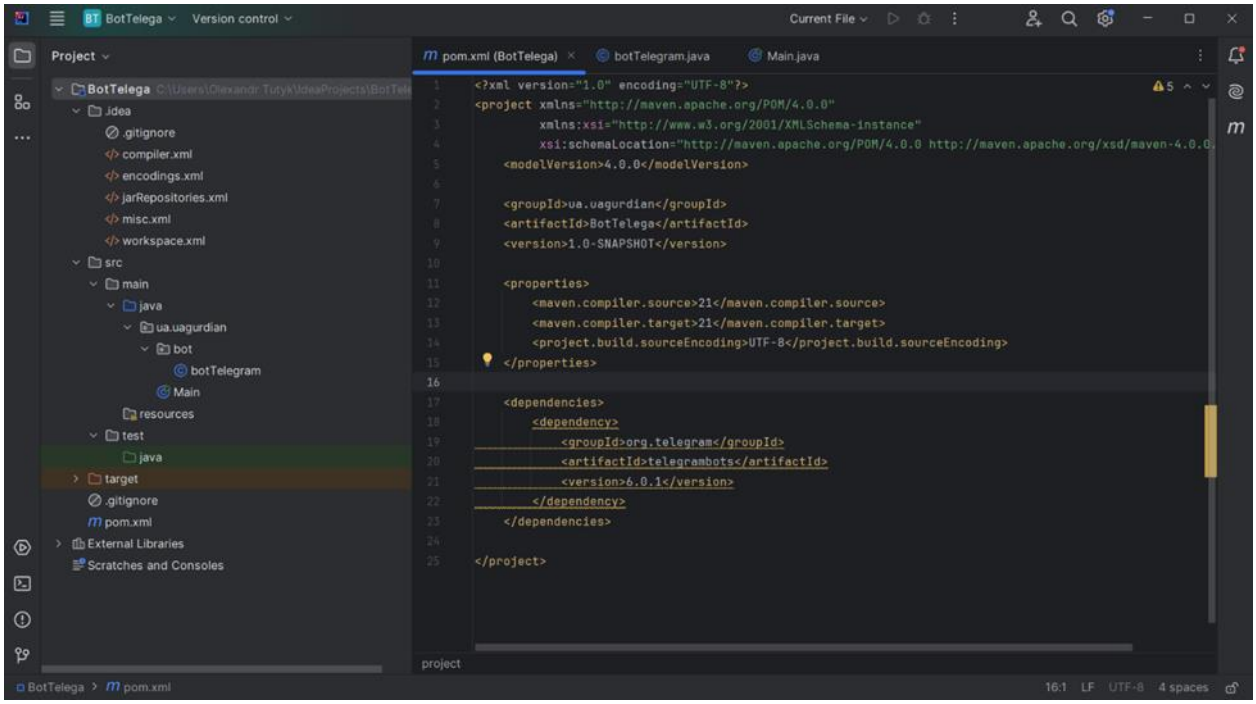


Рис. 2.3. Створення залежності в методі «M pom.xml»

Безпосередньо сама залежність яка була внесена в код програми:

```
<dependencies>
    <dependency>
        <groupId>org.telegram</groupId>
        <artifactId>telegrambots</artifactId>
        <version>6.0.1</version>
    </dependency>
</dependencies>
```

Перейдемо до створення логіки нашого боту, тобто надання функціоналу , та об’єднаємо наш клас botTelegram з створеним ботом з ім’ям «YourparrotparrotTelegrembot».[41].

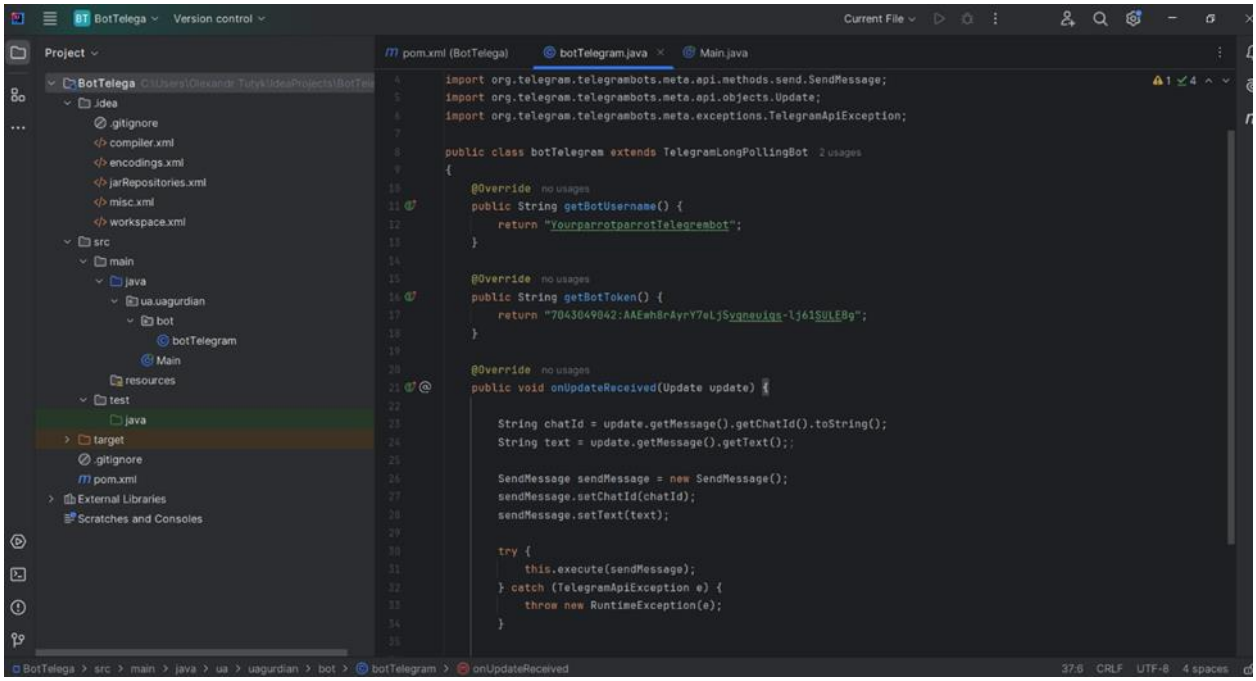


Рис. 2.4. Створення зв’язку між телеграм ботом та програмним забезпеченням

Вкажемо відповідно в код ім’я нашого боту та токен, щоб дане програмне забезпечення працювало як одне ціле з нашим телеграм ботом.

```
@Override
    public String getBotUsername() {
        return "YourparrotparrotTelegrembot";
    }


    @Override
    public String getBotToken() {
        return "7043049042:AAEwh8rAyrY7eLj5vqneuiqs-lj61SULEBg";
    }
```

Далі реалізуємо саму логіку нашого боту, функціонал буде доволі не великим. Завданням нашого боту буде повернення повідомлень відправлених користувачем назад в чат так би мовити віддзеркалювати надіслані повідомлення цьому боту. Ось де саме реалізована ця логіка в коді в класі.

«botTelegram».

```
@Override
   public void onUpdateReceived(Update update) {


      String chatId = update.getMessage().getChatId().toString();
      String text = update.getMessage().getText();;


      SendMessage sendMessage = new SendMessage();
      sendMessage.setChatId(chatId);
      sendMessage.setText(text);


      try {
         this.execute(sendMessage);
      } catch (TelegramApiException e) {
         throw new RuntimeException(e);
      }



   }
```

На далі об’єднаємо клас «botTelegram» та метод «Mail», для передачі інформації між класами, для правильної роботи нашого боту.

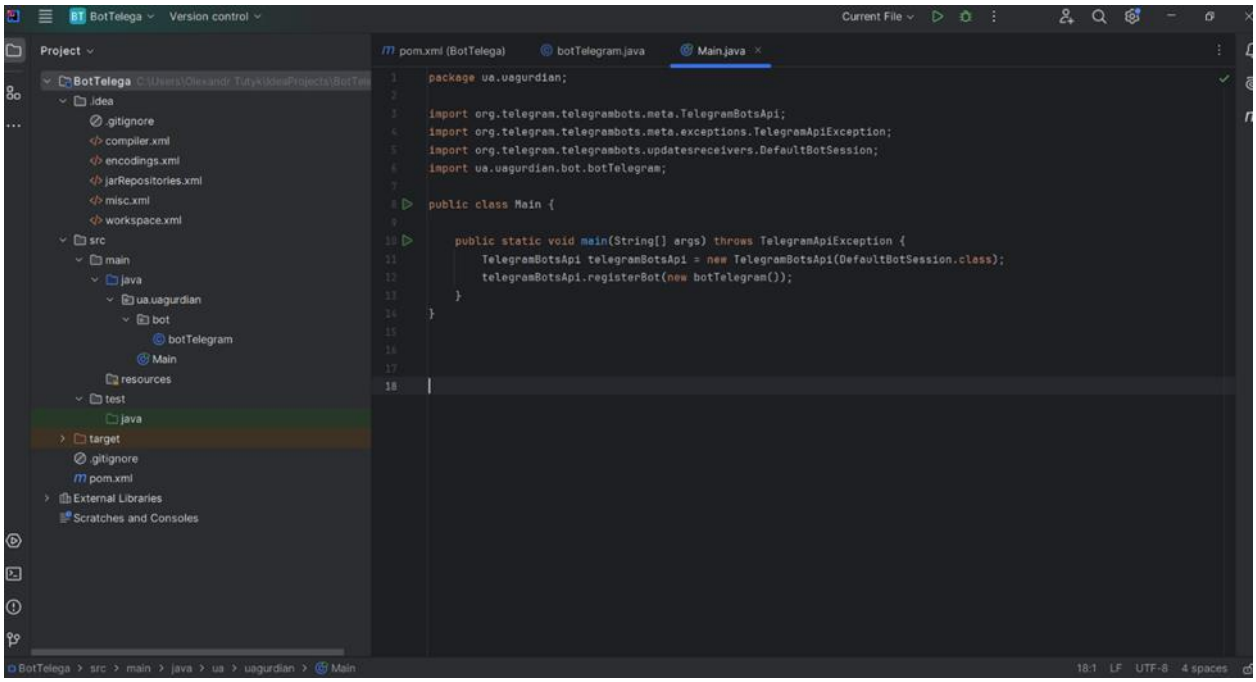


Рис. 2.5. Об’єднання двох класів

Об’єднання було виконано саме в цій ділянці коду public class Main {

```
public static void main(String[] args) throws TelegramApiException {
    TelegramBotsApi telegramBotsApi = new
TelegramBotsApi(DefaultBotSession.class);
    telegramBotsApi.registerBot(new botTelegram());
}
}
```

На етапі розроблення виникла доволі велика проблема, в роботі даного боту, проблема не в самому коді програми, а в запускі даного боту на даному

пристрої. Тобто суть проблеми є в тому що додаток не буде працювати якщо код не буде запущений на комп'ютері тобто на віртуальній машині. Вирішення даної проблеми є, це розміщення її на сервері в цьому може допомогти сервіс під назвою hyperhost.ua , це наш український сервіс з оренди серверів. Запустивши нашого бота на сервері він буде працювати весь час поки буде ввімкнений сервер.[37].

РОЗДІЛ 3. ЕТАПИ ТЕСТУВАННЯ

3.1 Підготовка боту до роботи

Тестування програмного забезпечення є невід'ємною частиною процесу розробки, яка дозволяє забезпечити якість, надійність та стабільність продукту. У випадку з телеграм ботами, важливо забезпечити коректну роботу всіх функцій та інтеграцій, а також відповідність бота очікуванням користувачів. Тестування даного боту виконувалося в телеграмі. Для початку тестування потрібно розробленого чат-бота знайти в телеграмі за даним ім’ям : «YourparrotparrotTelegrembot».

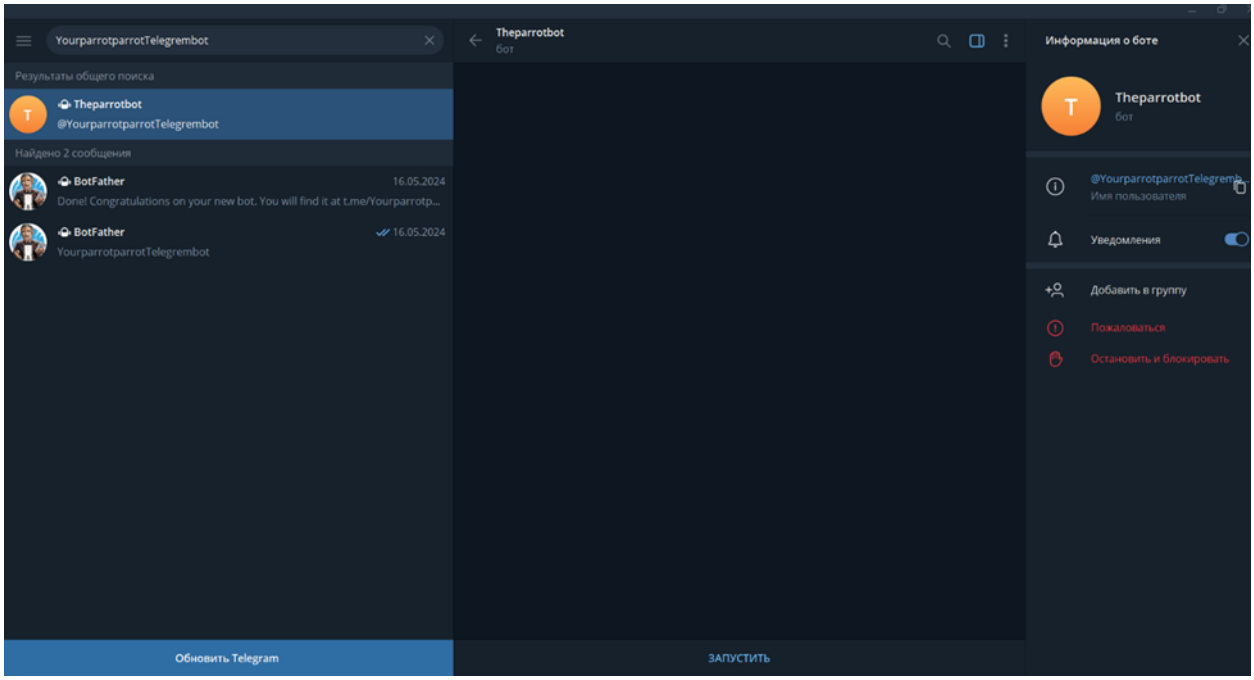


Рис. 2.6. Пошук бота в телеграмі

Бот знайдений, з цього етапу можливо розпочинати тестування даного чат-бота, першочергово запустимо бота за допомогою відповідної кнопки в даному інтерфейсі який зображено на "рис.2.6 пошук бота в телеграмі" .

Позатім коли наш бот успішно запущений, потрібно запустити наш проект на Java за допомогою середовища розробки IntelliJ IDEA. Запуск проекту в середовищі розробки здійснюється для активації коду щоб чат-бот міг працювати та виконувати запрограмований йому функціонал. Без даної

процедури чат-бот не буде функціонувати. Для цього відкриємо наш проект та запустимо його. Запуск проекту здійснюється за допомогою кнопки «Run», та чикаємо на успішний запуск програмного забезпечення.

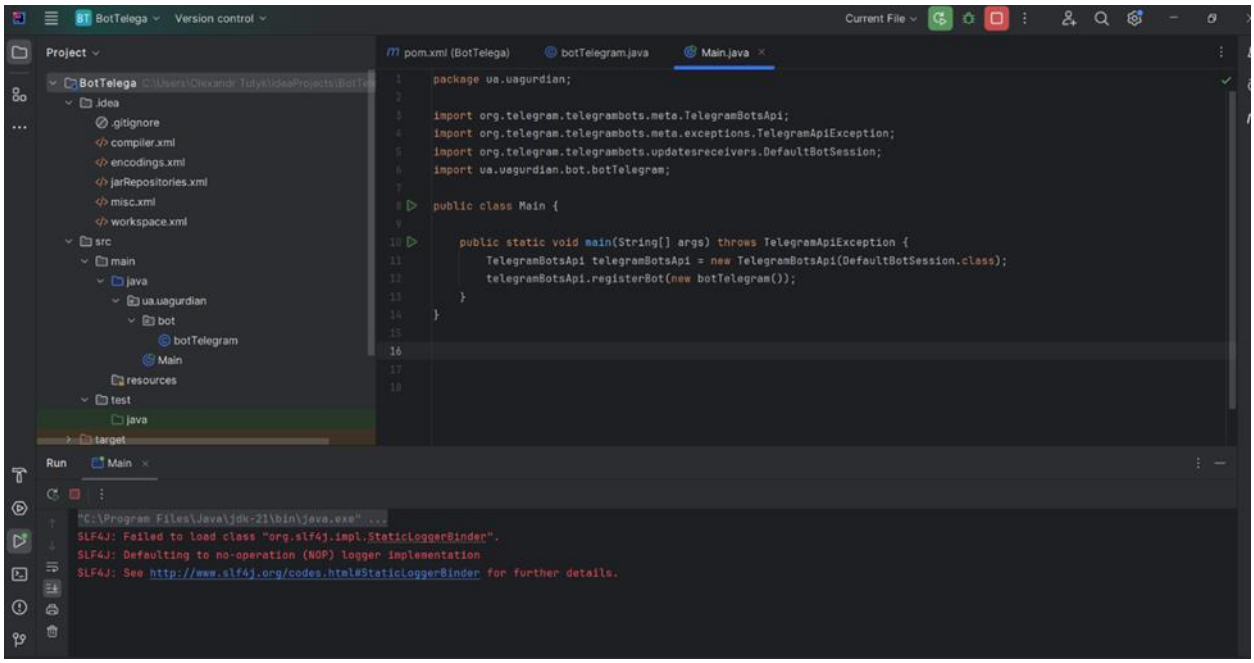


Рис. 2.8. Запуск проекту в середовищі програмування

Коли пересвідчились в коректному запуску коду на віртуальній машині, наступним етапом буде , безпосереднє тестування телеграм-боту.

3.2. Тестування телеграм-боту

На даному етапі, розпочнемо саме випробування чат-боту, на його робото-здібність виконання програмованих задач та роботі інтерфейсу даного чату.

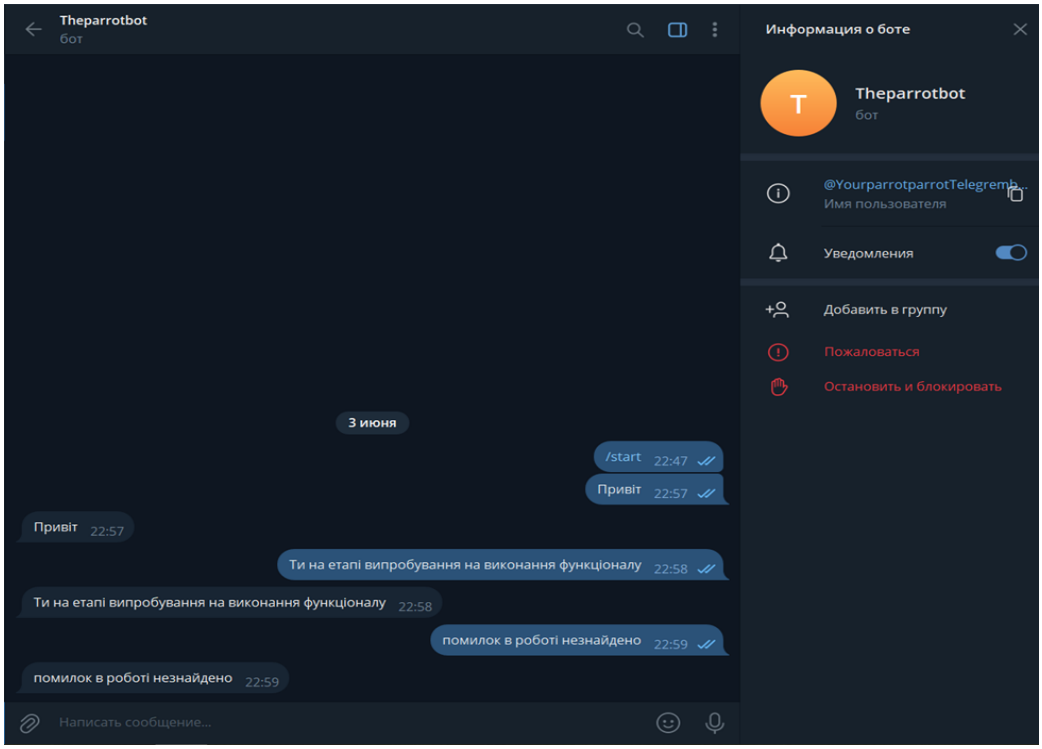


Рис. 2.9. Перевірка робото-здібність нашого боту

Бот працює та виконує свій невеликий функціонал, в цьому можливо переконатися переглянувши даний «Рис.2.10. діалогу і з телеграм ботом» .

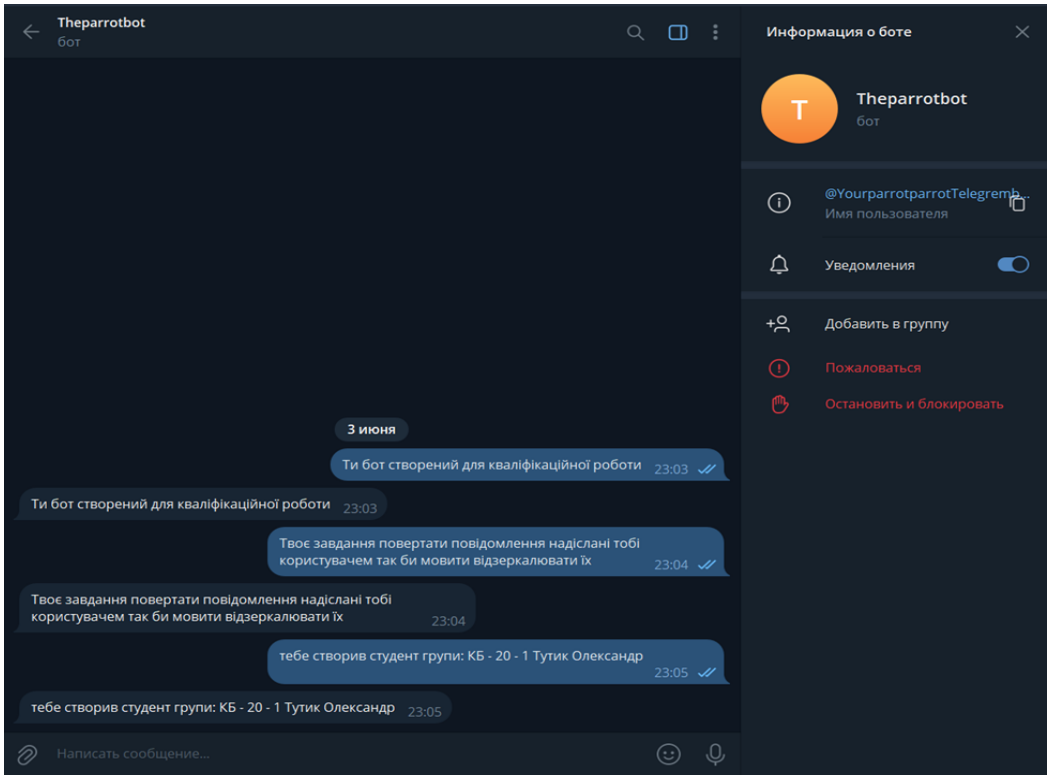


Рис. 2.10. Діалог і з телеграм ботом

Випробування чат-боту були проведені. Чат-бот працює коректно помилок не було помічено, зауваження відсутні.

3.3. Пропозиції що до удосконалення телеграм-боту

Пропозиції що до того, щоб зробити цей бот не тільки розважальним, але й корисним інструментом, є можливість додати декілька удосконалень. Є кілька ідей, які допоможуть покращити функціональність та інтерфейс бота, зробивши його більш інтерактивним, розумним та привабливим для користувачів.

Одним з головних вдосконалень может бути фільтрація введення додавши фільтрацію, щоб бот міг не повторював образливі чи неприйнятні слова.

Використовувати бібліотеку або власний список "поганих" слів для фільтрації.

Другим пунктом , має виступати аналіз та обробка тексту, яка може додати можливість розпізнавання певних команд, наприклад, help, info, щоб бот міг відповідати на них певним чином. Використовувати бібліотеки для аналізу тексту, наприклад, NLP (Natural Language Processing).

Варіативність відповідей

Додати випадковий фактор, щоб бот іноді змінював формулювання відповіді, наприклад, додавання випадкових емодзі або варіацій фрази.

Використовувати шаблони для відповідей, щоб уникнути повного повторення тексту.

Логування та статистика

Додавання логування всіх повідомлень та відповідей для аналізу популярних фраз та вдосконалення роботи бота.

Збір статистики про те, які повідомлення найчастіше надсилаються та які відповіді генеруються.

Підтримка декількох мов:

Додати підтримку декількох мов, щоб бот міг відповідати на різних мовах залежно від введення користувача.

Використовувати API для перекладу тексту, наприклад, Google Translate API.

Відповідь на запити користувача:

Інтеграція з веб-сервісами для отримання актуальної інформації, наприклад, погода, новини, відповіді на прості запити.

Інтеграція з іншими платформами:

Розширення функціоналу бота для роботи на різних платформах, наприклад, Telegram, Slack, Discord.

Таблиця1.1.

Пропозиції що до удосконалення телеграм-боту

Пропозиція	Опис	Приклади реалізації
Фільтрація введення	Додавання системи фільтрації для уникнення повторення образливих чи неприйнятних слів.	Використання списку "поганих" слів для перевірки введеного тексту перед його повторенням.
Аналіз та обробка тексту	Розпізнавання певних команд та аналіз введеного тексту для більш розумних відповідей.	Реалізація команд типу `help`, `info` та використання бібліотек NLP для аналізу тексту.
Варіативність відповідей	Розробка системи випадкових відповідей для уникнення одноманітності та покращення взаємодії.	Використання шаблонів відповідей та додавання випадкових емодзі або варіацій фраз.
Логування та статистика	Збір та аналіз даних про взаємодію з ботом для подальшого вдосконалення його роботи.	Логування всіх повідомлень та відповідей, збір статистики про популярні фрази та частоту їх використання.
Інтерактивність	Додавання функцій, які роблять взаємодію з ботом більш захопливою та корисною для користувача.	Реалізація міні-ігор, вікторин та використання таймерів для більш цікавих відповідей.
Підтримка декількох мов	Розширення можливостей бота для роботи з різними мовами.	Використання API для перекладу тексту, наприклад, Google Translate API.
Відповідь на запити користувача	Інтеграція з веб-сервісами для надання актуальної інформації на запити користувача.	Інтеграція з веб-сервісами для отримання погоди, новин або іншої актуальної інформації.
Інтеграція з іншими платформами	Розширення функціоналу бота для роботи на різних платформах.	Інтеграція з Telegram, Slack, Discord та іншими популярними платформами для спілкування.

Для удосконалення чат-боту , можна використати код, додавши кілька нових функцій, таких як фільтрація введення, обробка команд, варіативність відповідей, та інтеграція з іншими сервісами. Дані пропозиції що до покращення функціоналу боту представлені в таблиці 1.1. Ось приклад розширеного коду.

Приклад коду для удосконалення чат-бота : import java.util.*;

```
public class ParrotBot {
    private static final String[] BAD_WORDS = {"badword1", "badword2"};
    private static final String[] RESPONSES = {"Так, звісно!", "Я згоден!", "Можливо...", "Це цікаво!"};
    private static final Random random = new Random();
    private static final Map<String, String> COMMANDS = new HashMap<>();

    public static void main(String[] args) {
        // Заповнюємо команди
        COMMANDS.put("help", "Я можу повторювати твої слова або відповідати на команди: help, info, game.");
        COMMANDS.put("info", "Я бот-попуга, створений для повторення твоїх слів і допомоги з деякими командами.");
        COMMANDS.put("game", "Давай зіграємо у гру! Введи будь-яке слово, і я спробую відгадати, що воно означає.");

        Scanner scanner = new Scanner(System.in);
        System.out.println("Привіт! Я - бот-попуга. Що ти хочеш мені сказати?");

        while (true) {
            String input = scanner.nextLine();

            if (containsBadWords(input)) {
                System.out.println("Будь ласка, не використовуйте образливі слова.");
            } else if (COMMANDS.containsKey(input.toLowerCase())) {
                System.out.println(COMMANDS.get(input.toLowerCase()));
```

```
            } else {
                String response = getRandomResponse(input);
                System.out.println(response);
            }
        }
    }

    private static boolean containsBadWords(String input) {
        for (String badWord : BAD_WORDS) {
            if (input.toLowerCase().contains(badWord)) {
                return true;
            }
        }
        return false;
    }

    private static String getRandomResponse(String input) {
        return RESPONSES[random.nextInt(RESPONSES.length)] + " Ти сказав: "
+ input;
    }
}
```

Опис коду по удосконаленню чат-бота

Фільтрація введення:

Метод containsBadWords(String input) перевіряє, чи містить введений текст образливі слова, використовуючи список BAD_WORDS.

Обробка команд:

Додано Map COMMANDS, що містить можливі команди та їх відповіді. Бот перевіряє введення користувача на наявність команд і відповідає відповідним чином.

Варіативність відповідей:

Метод getRandomResponse(String input) вибирає випадкову відповідь із масиву RESPONSES і додає її до тексту, введеного користувачем, що робить відповіді менш одноманітними.

Інтерактивність:

Додано команду game як приклад інтерактивної функції. В майбутньому можна розширити цей функціонал для додавання міні-ігор або інших інтерактивних функцій.

Подальші покращення

Логування та статистика:

Можна додати логування всіх повідомлень та відповідей для подальшого аналізу та покращення роботи бота. Це може бути реалізовано за допомогою простого запису в файл або використання бібліотек для логування.

Підтримка декількох мов:

Використовувати API для перекладу тексту, щоб бот міг підтримувати декілька мов. Це потребуватиме інтеграції з сервісом, таким як Google Translate API.

Інтеграція з іншими платформами:

Розширення функціоналу бота для роботи на різних платформах, таких як Telegram, Slack, Discord. Це вимагатиме використання відповідних API для кожної платформи.

ВИСНОВКИ

Під час виконання даної кваліфікаційної роботи я ознайомився з розробкою телеграм-ботів на мові Java та середовищем розробки IntelliJ IDEA. В процесі роботи було створено телеграм-бота з базовим функціоналом: він відповідає на повідомлення, які надсилає користувач.

Цей проект вимагав вивчення нових класів та бібліотек, необхідних для розробки бота. Важливим етапом було ознайомлення з існуючими бібліотеками, такими як TelegramBots, які забезпечують інтеграцію з Telegram API. Було проаналізовано переваги різних підходів до створення архітектури бота, включаючи обробку оновлень (updates), команд і запитів користувачів. Було розглянуто різні способи забезпечення надійності та масштабованості бота, що включало використання багатопоточності та ефективного управління ресурсами.

Розробка телеграм-бота включала кілька ключових етапів. Спочатку було створено основний клас бота, де здійснювалася ініціалізація та налаштування основних параметрів. Далі було реалізовано класи для обробки повідомлень та взаємодії з користувачами. Особлива увага приділялася забезпеченню відповідності коду стандартам Java, а також використанню об'єктно-орієнтованих принципів програмування. Було реалізовано функції для відправки та отримання повідомлень, обробки команд, а також взаємодії з зовнішніми сервісами та базами даних.

Тестування бота проходило в кілька етапів. Перший етап включав юніт-тестування основних методів для перевірки коректності роботи окремих частин коду. Наступним етапом було інтеграційне тестування, яке перевіряло взаємодію бота з Telegram API та іншими сервісами. На завершальному етапі було проведено функціональне тестування, під час якого перевірялися реальні сценарії використання бота, включаючи обробку різних типів повідомлень та

команд. Крім того, було проведено навантажувальне тестування для оцінки витривалості бота під великим навантаженням.

Загалом, ця тема є актуальною, оскільки розробка ботів є корисною та затребуваною сферою діяльності для програмістів. Різноманітність та функціональність ботів можуть бути адаптовані під будь-які потреби, залежно від побажань замовника та сфери використання.

Сфери застосування ботів дуже широкі. Наприклад, у сфері онлайн-торгівлі бот може замінити менеджера на деяких етапах, зменшуючи навантаження на працівників та скорочуючи час відповіді на повідомлення.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Ратушняк Т. В. Програмування мовою JAVA: практикум: навчальний посібник. Державна фіскальна служба України, Університет державної фіскальної служби України. – Ірпінь, 2017. – 212 с.
2. Bloch J. Effective Java: 3rd Edition, Addison Wesley, 2017, 412 p.
3. Schildt H. Java: A Beginner's Guide: 8th Edition, McGraw-Hill Education, 2018, 684 p.
4. Schildt H. Java: The Complete Reference: 11th Edition, McGraw-Hill Education, 2018, 1208 p.
5. Horstmann C. S. Core Java Volume I – Fundamentals: 11th Edition, Prentice Hall 2018, 889 p.
6. Horstmann C. S. Core Java SE 9 for the Impatient: 2nd Edition Addison-Wesley Professional, 2017, 576 p.
7. Cheng F. Exploring Java 9. Build Modularized Applications in Java. -
8. Berkeley: Apress, 2018. - 174 p
9. Varanasi B. Introducing Maven: A Build Tool for Today's Java Developers. -
10. Salt Lake City: Apress Berkeley, 2019. 140 p.
11. Eckel B. Thinking in Java 4th Edition: Pearson, 2006, 1150 p.
12. Deitel P., Deitel H. Java How to Program, Early Objects: 11th Edition, Pearson, 2017, 1296 p.
13. Deitel P., Deitel H. Java How To Program, Late Objects: 11th Edition, Pearson, 2017, 1248 p.
14. Steele J. The Android Developer’s Cookbook Building Applications with the Android SDK / James Steele, Nelson To. – United States, Indiana, Crawfordsville: AddisonWesley, RR Donnelley, 2011. – 339 p.
15. Gosling J. The Java® Language Specification – Java SE 8 Edition / James Gosling, Bill Joy, Steele, Gilad Bracha, Alex Buckley. – 500 Oracle Parkway, Redwood City, California 94065, U.S.A. – 2018. – 788 p.

16. Sierra K. Head First Java, 2nd Edition / Kathy Sierra, Bert Bates. O'Reilly Media; 2nd edition. – 2018. – 688 p.
17. Walls C. Spring in Action, Fourth Edition. Covers Spring 4 / Craig Walls. – Manning Publications Co., 2019. – 624 p.
18. Копитко М.Ф., Іванків К.С. Основи програмування мовою Java: Тексти лекцій. – Львів: Видавничий центр ЛНУ ім. Івана Франка, 2002. – 83 с.
19. Spring Framework Documentation [Electronic resource]. – Access mode: https://docs.spring.io/spring/docs/current/spring-framework-reference/.
20. Хорстманн С. Java. Бібліотека професіонала, том 1. Основи 11-е. / С. Хорстманн. – Діалектика-Вільямс, 2020. – 864 с.
21. Шилдт Г. Java. Повне керівництво , том 1. / Г. Шилдт. – Діалектика, 2020. – 730 с. 5.2.
22. Лонг Д. Java в хмарі. Spring Boot, Spring Cloud, Cloud Foundry / Д. Лонг, К. Бастані
23. Брнакевич І.Є., Вагін П.П. Програмування мовою Java: використання фундаментальних класів: Тексти лекцій. – Львів: Видавничий центр ЛНУ імені Івана Франка, 2002. – 75 с.
24. «Java Persistence API і Hibernate» Гевін Кінг , Гері Грегорі , Кристіан Бауер
25. Gosling J. The Java® Language Specification – Java SE 8 Edition / James Gosling, Bill Joy, Steele, Gilad Bracha, Alex Buckley. – 500 Oracle Parkway, Redwood City, California 94065, U.S.A. – 2018. – 788 p.
26. Sanghera P. SCJP Exam for J2SE 5. A Concise and Comprehensive Study Guide for The Sun Certified Java Programmer Exam. – New York: Springer- Verlag, Inc., 2019. – 423 p.
27. JSON проти XML [Електронний ресурс]/ Рамай Шанкар // hackr.io – 2021. - Режим доступу до ресурсу: https://hackr.io/blog/json-vs-xml
28. http://www.scala-lang.org/
29. http://www.scala-sbt.org/release/tutorial/
30. http://akka.io

31. Онлайн-курс Java Basic: [Електронний ресурс]. – Режим доступу https: https://online.ithillel.ua/courses/javabasic?gclid=CjwKCAiArY2fBhB9EiwAWqHK6k9Le8CQXS5mU4G1ChALbCh4ukk7qMsBpHdKwu9G6_i3pPVDM8U6XRoCDZkQAvD_BwE.
32. Галкін О.В., Катеринич Л.О., Шкільняк О.С. Програмування на Java 8: Навчальний посібник для студентів факультету комп'ютерних наук та кібернетики. – Київ: ЛОГОС, 2017. – 186 с.
33. Кадомський К.К., Ніколюк П.К. Java. Теорія і практика : Навч. посіб. - Вінниця: Донну, 2019. 197 с.
34. Васильєв О.М. Програмування мовою Java. — Тернопіль: Богдан, 2019. — 696 с.
35. Бегун А.В. Алгоритмізація і програмування: Об'єктно-орієнтовне програмування – К.КНЕУ, 2005-176 с
36. The Java Language Specification, Second Edition. James Gosling, Bill Joy,Guy Steele, Gilad Bracha
37. Зв'язовик М.О. Програмування на Java [Курс лекцій] (2003)
38. Горбань А.Г. Програмування в Java. 2008 – 310 с
39. Комп'ютерні мережі: [навчальний посібник] / А. Г. Микитишин, М. М. Митник, П. Д. Стухляк, В. В. Пасічник. — Львів: «Магнолія 2006», 2013. — 17 с.
40. SQL та NoSQL підходи [Електронний ресурс]/ Бенджамін Андерсен// hackr.io – 2020. - Режим доступу до ресурсу: https://www.ibm.com/cloud/blog/sql-vs-nosql
41. Об'єктно-орієнтоване програмування [Електронний ресурс] – Режим доступу до ресурсу: https://uk.wikipedia.org/wiki/%D0%9E%D0%B1%27%D1%94%D0%BA%D1%82%D0%BD%D0%BE-%D0%BE%D1%80%D1%96%D1%94%D0%BD%D1%82%D0%BE%D0%B2%D0%B0%D0%BD%D0%B5_%D0%BF%D1

%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D1%83%D0%B2%D0%B0%D0%BD%D0%BD%D1%8F

42. Що таке Java Java [Електронний ресурс] – Режим доступу до ресурсу: https://uk.myservername.com/what-is-java-vector-java-vector-class-tutorial-with-examples#:~:text=Q%20%23%201)%20%D0%A9%D0%BE%20%D1%82%D0%B0%D0%BA%D0%B5%20%D0%B2%D0%B5%D0%BA%D1%82%D0%BE%D1%80,%D0%BE%D1%82%D1%80%D0%B8%D0%BC%D0%B0%D1%82%D0%B8%20%D0%B4%D0%BE%D1%81%D1%82%D1%83%D0%BF%20%D0%B7%D0%B0%20%D0%B4%D0%BE%D0%BF%D0%BE%D0%BC%D0%BE%D0%B3%D0%BE%D1%8E%20%D1%96%D0%BD%D0%B4%D0%B5%D0%BA%D1%81%D1%96%D0%B2

43. Java BitSet [Електронний ресурс] – Режим доступу до ресурсу: https://docs.oracle.com/javase/7/docs/api/java/util/BitSet.html

44. .BitVector [Електронний ресурс] – Режим доступу до ресурсу: https://flylib.com/books/en/2.300.1/bit_vectors.html

ДОДАТКИ

Додаток А

Перелік додаткової інформації

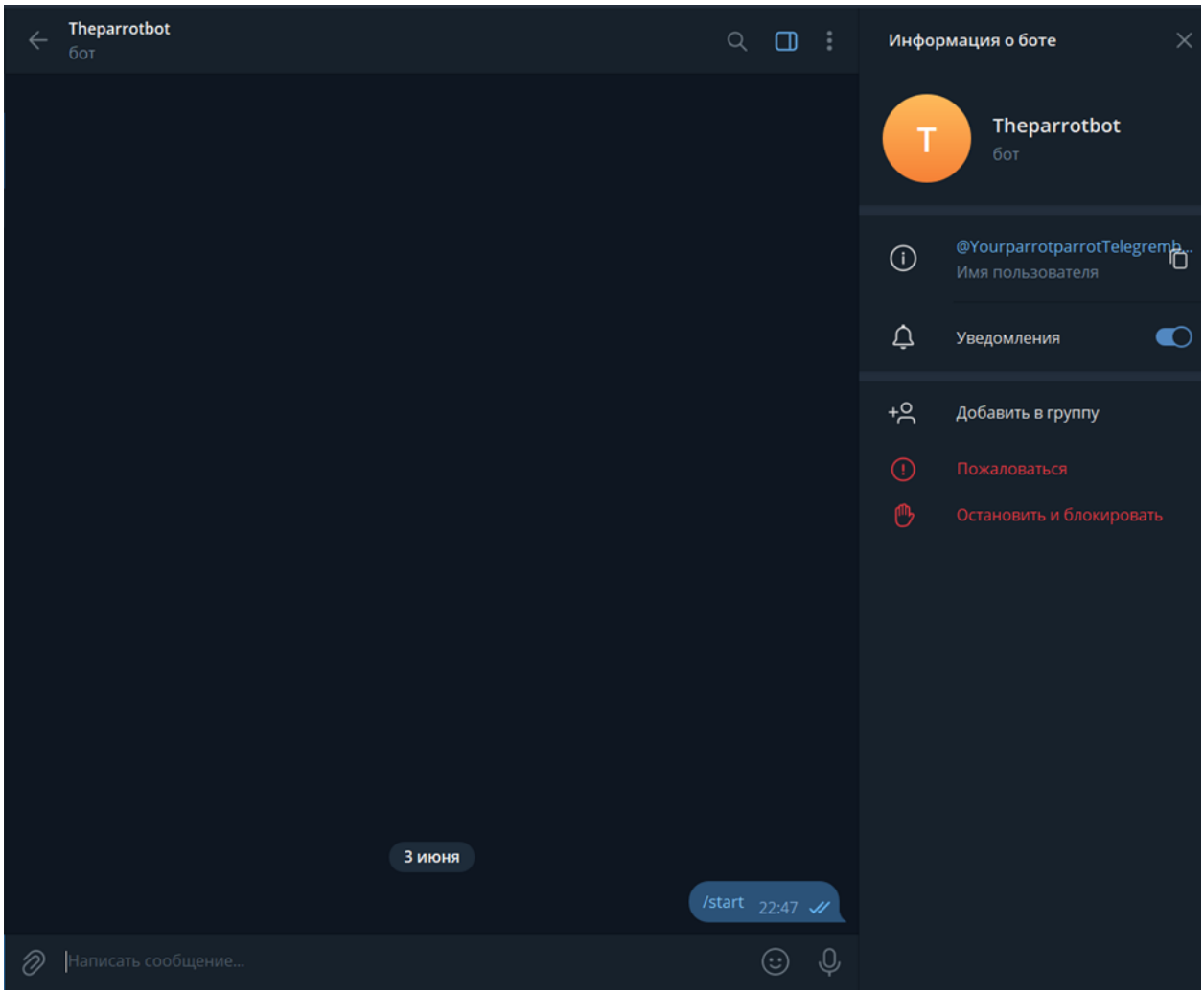


Рис. А. 1. Запуск телеграм боту

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
```

<groupId>ua.uagurdian</groupId>Продовження додатку А
<artifactId>BotTelega</artifactId>
<version>1.0-SNAPSHOT</version>

<properties>
<maven.compiler.source>21</maven.compiler.source>
<maven.compiler.target>21</maven.compiler.target>
<project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
</properties>

<dependencies>
<dependency>
<groupId>org.telegram</groupId>
<artifactId>telegrambots</artifactId>
<version>6.0.1</version>
</dependency>
</dependencies>

</project>

```
package ua.uagurdian.bot;

import org.telegram.telegrambots.bots.TelegramLongPollingBot;
import org.telegram.telegrambots.meta.api.methods.send.SendMessage;
import org.telegram.telegrambots.meta.api.objects.Update;
import org.telegram.telegrambots.meta.exceptions.TelegramApiException;

public class botTelegram extends TelegramLongPollingBot
```

Продовження додатку А

```
{
    @Override
    public String getBotUsername() {
        return "YourparrotparrotTelegrembot";
    }

    @Override
    public String getBotToken() {
        return "7043049042:AAEwh8rAyrY7eLj5vqneuiqs-lj61SULEBg";
    }

    @Override
    public void onUpdateReceived(Update update) {

        String chatId = update.getMessage().getChatId().toString();
        String text = update.getMessage().getText();;

        SendMessage sendMessage = new SendMessage();
        sendMessage.setChatId(chatId);
        sendMessage.setText(text);

        try {
            this.execute(sendMessage);
        } catch (TelegramApiException e) {
            throw new RuntimeException(e);
```

```
        }
    }
```

Продовження додатку А

```
package ua.uagurdian;

import org.telegram.telegrambots.meta.TelegramBotsApi;
import org.telegram.telegrambots.meta.exceptions.TelegramApiException;
import org.telegram.telegrambots.updatesreceivers.DefaultBotSession;
import ua.uagurdian.bot.botTelegram;

public class Main {

    public static void main(String[] args) throws TelegramApiException {
        TelegramBotsApi telegramBotsApi = new
TelegramBotsApi(DefaultBotSession.class);
        telegramBotsApi.registerBot(new botTelegram());
    }
}
```