Міністерство освіти і науки України
Національний технічний університет України
“Київський політехнічний інститут імені Ігоря Сікорського”
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Лабораторна робота №3

**Технології розроблення програмного забезпечення**

«ДІАГРАМА РОЗГОРТАННЯ. ДІАГРАМА КОМПОНЕНТІВ. ДІАГРАМА ВЗАЄМОДІЙ ТА ПОСЛІДОВНОСТЕЙ.»

Виконав:

студент групи ІА-24

Чайка А. П.

Перевірив:

Мягкий М. Ю.

Київ 2024

**Тема лабораторних робіт:**

IRC client (singleton, builder, abstract factory, template method, composite, client-server)

Клієнт для IRC-чатів з можливістю вказівки порту і адреси з'єднання, підтримка базових команд (підключення до чату, створення чату, установка імені, реєстрація, допомога і т.д.), отримання метаданих про канал.

**Завдання:**

1. Ознайомитися з короткими теоретичними відомостями.
2. Розробити діаграму розгортання для проектованої системи.
3. Розробити діаграму компонентів для проектованої системи.
4. Розробити діаграму послідовностей для проектованої системи.
5. Скласти звіт про виконану роботу.

**Зміст**



**Хід роботи:**

**Крок 1.** Діаграма розгортання

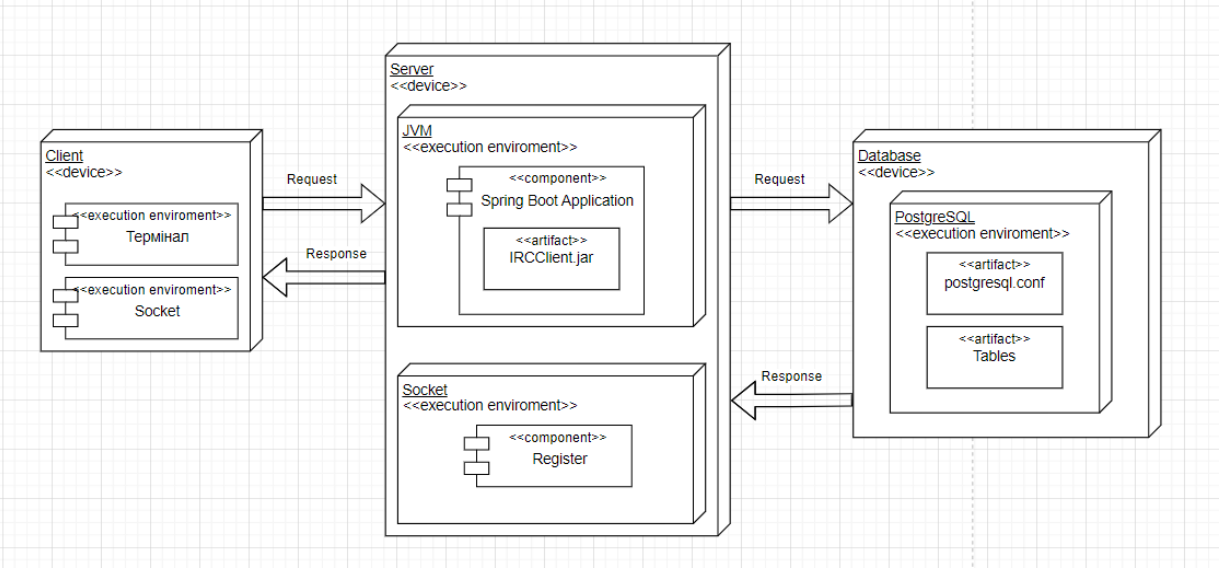


Рис. 1 – Діаграма розгортання

Ця діаграма є компонентною діаграмою, яка описує архітектуру системи з трьома основними компонентами: клієнтом, сервером і базою даних (DB).

1. Client (Клієнт):

- Представлений як окремий пристрій.

- Має два середовища виконання:

- Термінал (Terminal), який може використовуватись для взаємодії користувача із системою.

- Socket — для підключення до сервера та обміну даними через сокети.

- Відправляє запити на сервер і отримує від нього відповіді.

2. Server (Сервер):

- Представлений як пристрій із кількома середовищами виконання:

- JVM (Java Virtual Machine) — середовище, де працює додаток Spring Boot:

- Spring Boot Application — основний компонент серверної частини.

- IRCClient.jar — артефакт (додаток), що працює в рамках цього компоненту.

- Socket — окреме середовище виконання, яке містить компонент Register для обробки запитів клієнта через сокети.

- Сервер приймає запити від клієнта, обробляє їх і повертає відповіді.

3. DB (База даних):

- Представлена як окремий пристрій.

- Використовує PostgreSQL як середовище виконання:

- postgresql.conf — конфігураційний файл бази даних.

- Tables — таблиці бази даних, де зберігаються дані.

- Сервер відправляє запити до бази даних для збереження або отримання даних і отримує відповідь у вигляді результатів запиту.

Основні взаємодії:

- Client-Server: Клієнт надсилає запити на сервер і отримує відповіді через сокет-з’єднання.

- Server-DB: Сервер надсилає запити до бази даних для обробки та зберігання даних, отримуючи відповіді на ці запити.

Ця діаграма демонструє загальну структуру системи, в якій клієнт взаємодіє із сервером, а сервер у свою чергу — з базою даних для обробки запитів.

**Крок 2.** Діаграма компонентів

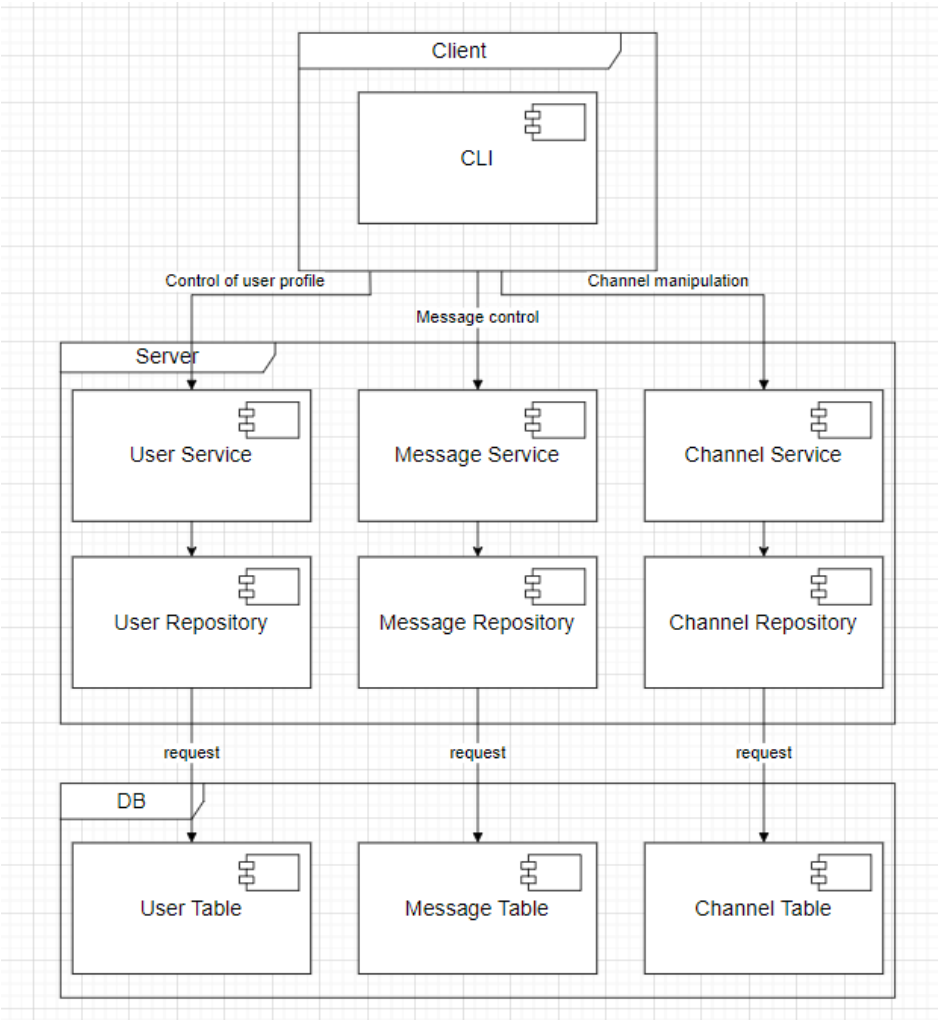


Рис. 2 – Діаграма компонентів

Ця діаграма показує архітектуру системи для керування користувачами, повідомленнями та каналами. Вона складається з трьох основних частин: клієнтської частини, серверної частини та бази даних (DB).

1. Client (Клієнт):

- Містить компонент CLI (Command Line Interface), який дозволяє користувачам взаємодіяти із системою через командний рядок.

- Виконує три основні дії:

  - Control of user profile — управління профілем користувача.

  - Message control — управління повідомленнями.

  - Channel manipulation — управління каналами.

2. Server (Сервер):

- Складається з трьох служб та трьох репозиторіїв:

  - User Service — обробляє запити, пов'язані з профілем користувача, і взаємодіє з User Repository.

  - Message Service — обробляє повідомлення, використовуючи Message Repository для зберігання та отримання даних.

  - Channel Service — керує каналами, спільно працюючи з Channel Repository для доступу до даних про канали.

- User Repository, Message Repository, і Channel Repository служать для взаємодії із базою даних.

3. DB (База даних):

- Містить три таблиці:

  - User Table — зберігає дані користувачів, які обробляються через User Repository.

  - Message Table — зберігає повідомлення, що доступні через Message Repository.

  - Channel Table — зберігає дані про канали, до яких звертається Channel Repository.

Основні взаємодії:

- Клієнт взаємодіє з сервером, надсилаючи запити для управління профілем, повідомленнями та каналами.

- User Service, Message Service, і Channel Service на сервері відповідають за обробку відповідних запитів клієнта.

- Служби використовують відповідні Repositories для звернення до бази даних, щоб отримати або зберегти інформацію у відповідних таблицях.

Ця структура дозволяє клієнтам виконувати дії з профілями користувачів, повідомленнями та каналами, зберігаючи дані в базі даних за допомогою сервісів і репозиторіїв на сервері.

**Крок 3.** Діаграма послідовностей

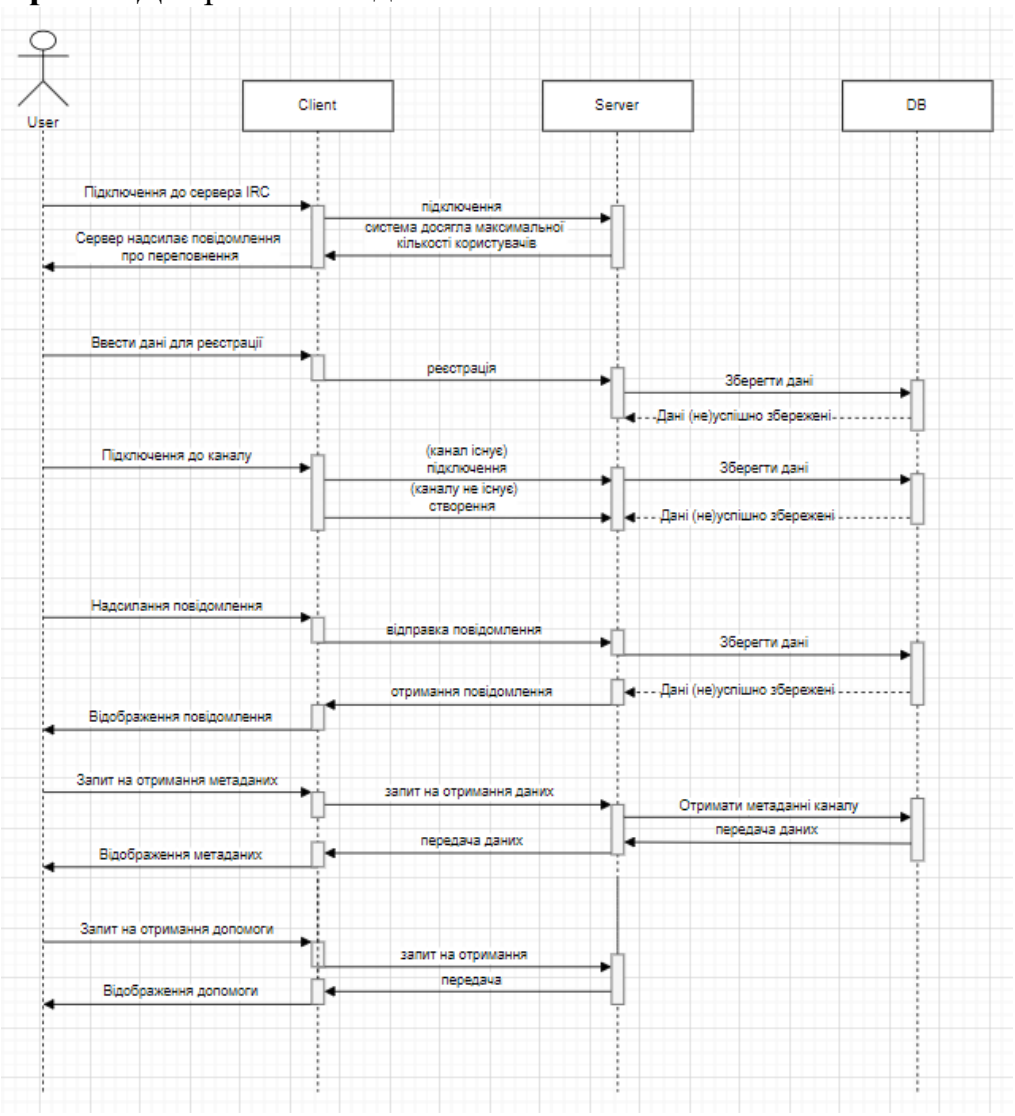


Рис. 3 – Діаграма послідовностей

Ця діаграма є діаграмою послідовностей, яка показує взаємодію між користувачем, клієнтом, сервером та базою даних у процесі роботи з IRC (Internet Relay Chat). Діаграма містить наступні кроки:

1. Підключення до сервера IRC:
   - Користувач підключається до сервера IRC через клієнт.
   - Сервер надсилає повідомлення про максимальну кількість користувачів.
2. Введення даних для реєстрації:
   - Користувач вводить дані для реєстрації через клієнт.
   - Клієнт надсилає дані на сервер для реєстрації.
   - Сервер зберігає дані в базі даних.
3. Підключення до каналу:
   - Користувач підключається до каналу через клієнт.
   - Якщо канал існує, сервер підтверджує підключення.
   - Якщо канал не існує, сервер створює новий канал і зберігає дані в базі даних.
4. Надсилання повідомлення:
   - Користувач надсилає повідомлення через клієнт.
   - Клієнт надсилає повідомлення на сервер.
   - Сервер зберігає дані в базі даних.
5. Відображення повідомлення:
   - Клієнт отримує повідомлення від сервера і відображає його користувачу.
6. Запит на отримання метаданих:
   - Користувач запитує метадані через клієнт.
   - Клієнт надсилає запит на сервер.
   - Сервер отримує метадані з бази даних і передає їх клієнту.
   - Клієнт відображає метадані користувачу.
7. Запит на отримання допомоги:
   - Користувач запитує допомогу через клієнт.
   - Клієнт надсилає запит на сервер.

- Сервер передає відповідь клієнту.
- Клієнт відображає допомогу користувачу.

Ця діаграма є важливою для розуміння процесу взаємодії між різними компонентами системи IRC, що допомагає виявити можливі проблеми та оптимізувати роботу системи.

**Висновок:** У ході виконання лабораторної роботи було розроблено клієнт IRC-чату, що використовує архітектуру клієнт-сервер і патерни проектування (singleton, builder, abstract factory, template method, composite). Додаток підтримує базові команди, такі як підключення до чату, створення чату, установка імені, реєстрація, запит допомоги та отримання метаданих про канал. Реалізована функціональність дозволяє користувачам встановлювати порт та адресу для з'єднання, що підвищує гнучкість використання клієнта в різних мережах.

Виконані завдання включали ознайомлення з теоретичними основами патернів проектування та їх застосування в контексті розробки клієнта для IRC, а також створення діаграм, що описують структуру та взаємодію компонентів системи. Зокрема:

1. Діаграма розгортання демонструє, як компоненти системи розташовані в фізичному середовищі, відображаючи взаємодію клієнта, сервера та бази даних.
2. Діаграма компонентів показує зв’язки між різними сервісами та репозиторіями, що відповідають за управління користувачами, повідомленнями та каналами.
3. Діаграма послідовностей описує процеси реєстрації, підключення до каналів, відправлення та отримання повідомлень, а також запити на метадані і допомогу, що дозволяє чітко уявити основні етапи взаємодії між компонентами.

Розроблений клієнт IRC-чату демонструє використання обраних патернів проектування для організації коду, підвищення його структурованості та зручності підтримки. Завдяки розробленим діаграмам було забезпечено

детальне документування системи, що спрощує її подальшу розробку та підтримку.