**Міністерство освіти і науки України**

**Черкаський державний технологічний університет**

**Факультет інформаційних технологій і систем**

**Кафедра інформаційних технологій проектування**

Дисципліна: Backend development

**З В І Т**

**з лабораторної роботи № 1**

студента 3-го курсу спеціальності 126 «Інформаційні системи і технології»

***Завгороднього Нікіти Ігоровича***

Виконав   \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

         (підпис)

Оцінка  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Перевірено  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

                 (дата)

Перевірила \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /                              Викладач Катаєв Д.С. /

            (підпис)                                              (прізвище та ініціали)

 Черкаси – 2024 року

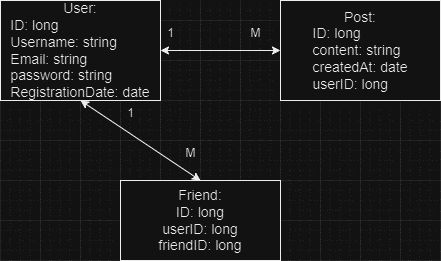
Лабораторна робота №2

Розробка бізнес-логіки та архітектури веб-додатку.

Завдання: визначити бізнес-логіку веб-додатка за індивідуальним варіантом завдання та відділити бізнес-логіку за MVC схемою розподілу керуючої логіки та даних веб-додатка. Виконати проектування архітектури додатка використовуючи відповідний шаблон та обрати паттерн проектування для реалізації додатка. Реалізувати один з патернів у Spring Boot.

Хід роботи:

1. Діаграма класів



1. Патерн проектування архітектури проекту

project

│

├── src

│ ├── main

│ │ ├── java

│ │ │ ├── com.example.project

│ │ │ │ ├── entities

│ │ │ │ │ ├── ProjectApplication

│ │ ├── resources

│ │ │ ├── templates

│ │ │ │ ├── project.mustache

│ │ │ │ └── profile.mustache

│ │ │ │ ├── User.java

│ │ │ │ ├── Post.java

│ │ │ │ ├── PostRepository.java

│ │ │ │ ├── UserController.java

│ │ │ │ ├── UserRepository.java

│ │ │ └── application.properties

│ └── test

│ ├── java

│ │ ├── com.example.project

│ │ │ └── ProjectApplicationTest

├── .gitignore

├── build.gradle

├── gradlew

├── gradlew.bat

├── HELP.md

└── settings.gradle

1. Розробив файли для створення та наповнення таблиць.

Код скрипта створення таблиць:

-- Створення таблиці користувачів  
CREATE TABLE users (  
 id SERIAL PRIMARY KEY,  
 username VARCHAR(100) NOT NULL,  
 email VARCHAR(100) UNIQUE NOT NULL,  
 password VARCHAR(255) NOT NULL,  
 created\_at TIMESTAMP DEFAULT *CURRENT\_TIMESTAMP*);  
  
-- Створення таблиці друзів  
CREATE TABLE friends (  
 id SERIAL PRIMARY KEY,  
 user\_id INT NOT NULL,  
 friend\_id INT NOT NULL,  
 status VARCHAR(20) NOT NULL DEFAULT 'pending', -- pending, accepted, declined  
 created\_at TIMESTAMP DEFAULT *CURRENT\_TIMESTAMP*,  
 FOREIGN KEY (user\_id) REFERENCES users(id),  
 FOREIGN KEY (friend\_id) REFERENCES users(id)  
);  
  
-- Створення таблиці постів  
CREATE TABLE posts (  
 id SERIAL PRIMARY KEY,  
 user\_id INT NOT NULL,  
 content TEXT NOT NULL,  
 created\_at TIMESTAMP DEFAULT *CURRENT\_TIMESTAMP*,  
 FOREIGN KEY (user\_id) REFERENCES users(id)  
);  
  
-- Створення таблиці коментарів  
CREATE TABLE comments (  
 id SERIAL PRIMARY KEY,  
 post\_id INT NOT NULL,  
 user\_id INT NOT NULL,  
 content TEXT NOT NULL,  
 created\_at TIMESTAMP DEFAULT *CURRENT\_TIMESTAMP*,  
 FOREIGN KEY (post\_id) REFERENCES posts(id),  
 FOREIGN KEY (user\_id) REFERENCES users(id)  
);

Код скрипта наповнення таблиць:

-- Додавання тестових користувачів  
INSERT INTO users (username, email, password) VALUES  
 ('user1', 'user1@example.com', 'password1'),  
 ('user2', 'user2@example.com', 'password2'),  
 ('user3', 'user3@example.com', 'password3');  
  
-- Додавання друзів  
INSERT INTO friends (user\_id, friend\_id, status) VALUES  
 (1, 2, 'accepted'),  
 (1, 3, 'pending'),  
 (2, 3, 'accepted');  
  
-- Додавання тестових постів  
INSERT INTO posts (user\_id, content) VALUES  
 (1, 'Hello, world!'),  
 (2, 'This is my first post!');  
  
-- Додавання коментарів  
INSERT INTO comments (post\_id, user\_id, content) VALUES  
 (1, 2, 'Great post!'),  
 (1, 3, 'Nice to see you here!');

1. API проекту.

POST /api/register - реєстрація користувача

POST /api/login - аутентифікація користувача

GET /api/profile/{user\_id} - перегляд профілю користувача

PUT /api/profile - редагування профілю користувача

POST /api/friend-request - надсилання запиту на додавання в друзі

GET /api/friend-request - перегляд отриманих запитів на дружбу

PUT /api/friend-request/{request\_id} - прийняття або відхилення запиту

GET /api/friends - перегляд списку друзів

POST /api/posts - створення поста

GET /api/posts - перегляд постів

DELETE /api/posts/{post\_id} - видалення поста

1. –
2. –
3. –

Висновок - визначив бізнес-логіку веб-додатка за індивідуальним варіантом завдання та відділив бізнес-логіку за MVC схемою розподілу керуючої логіки та даних веб-додатка. Виконав проектування архітектури додатка використовуючи відповідний шаблон та обрав паттерн проектування для реалізації додатка.