**МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ**

**ДЕРЖАВНИЙ УНІВЕРСИТЕТ ІНФОРМАЦІЙНО-КОМУНІКАЦІЙНИХ ТЕХНОЛОГІЙ**

**КАФЕДРА ІНЖЕНЕРІЇ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ**

КУРСОВА РОБОТА

на тему:

«Командний менеджер задач / Застосунок для спільної роботи над проектами та управління задачами в команді»

з предмету «Конструювання програмного забезпечення (Java)»

студента 3 курсу групи ПД-34

кафедри інженерії програмного забезпечення

Стаценка Володимира Костянтиновича

Викладач:

Калинюк Богдан

оцінка \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Київ 2024

ЗМІСТ

[ВСТУП 3](#_Toc167909234)

[1. Проектування інформаційної системи 4](#_Toc167909235)

[1.1 опис User stories 4](#_Toc167909236)

[1.2 Проектування бази даних 6](#_Toc167909237)

[1.3 Проектування діаграми класів 8](#_Toc167909238)

[1.4 Написання тестових сценаріїв 10](#_Toc167909239)

[2. Реалізація API 11](#_Toc167909240)

[2.1 Підготовка бази даних 11](#_Toc167909241)

[2.2 Створення Spring boot Application 11](#_Toc167909242)

[2.3 Написання контролерів 13](#_Toc167909243)

[2.4 Написання сервісів 15](#_Toc167909244)

[2.5 Написання Unit тестів 19](#_Toc167909245)

[3. Підключення Swagger, системи авторизації з ролями 21](#_Toc167909246)

[3.1 Підключення системи авторизації OAuth2 21](#_Toc167909247)

[3.2 Підключення Swagger UI з детальним описом REST API 22](#_Toc167909248)

[4 Інтеграційне тестування системи 25](#_Toc167909249)

[Висновок 26](#_Toc167909250)

# ВСТУП

У сучасному світі інформаційні технології є важливим аспектом усіх частин життя людини. Спеціалісти із сфери інформаційних технологій кожен день працюють над різноманітними проєктами які призначені для автоматизації та оптимізації різноманітних речей у людському житті тим самим підштовхуючи людство до розвитку.

Й для зручної роботи над сучасними проєктами програмісти не працюють по одинці – вони працюють у команді. Й для того щоб під час роботи над проєктом не виникало труднощів й не було не до розумінь, роботу над проєктом організовують у різних виглядах, починаючи від дошки у офісі завершуючи онлайн сервісами, але не усі такі сервіси є безкоштовними.

Темою курсового проєкту є «Командний менеджер задач» мета якого є створення програмного продукту для організації командної роботи під час роботи над різноманітними проєктами, шлях видання завдань конкретним членам команди.

# 1. Проектування інформаційної системи

## 1.1 опис User stories

Створення команди.

Опис: я, як голова команди, хочу мати можливість створити команду на сайті для подальшої роботи із нею.

Критерії прийняття:

* голова команди має можливість створювати команду на сайті;
* голова команди має можливість переглянути створену команду;
* голова команди має доступ до керування процесами команди.

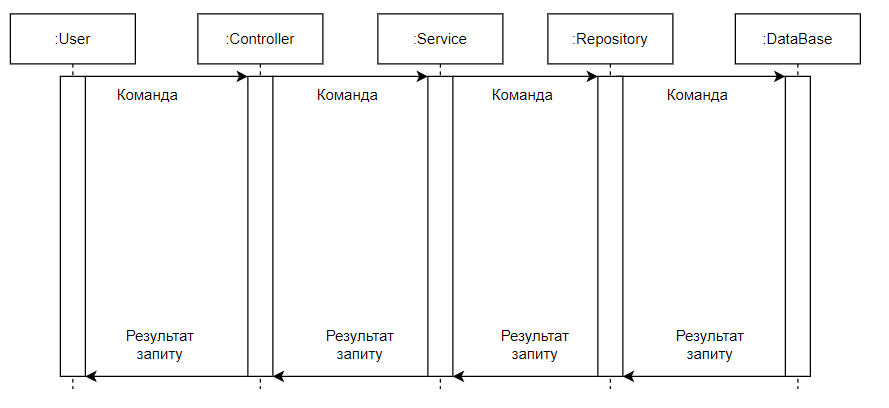


Рисунок 1.1 – Діаграма послідовностей створення команди

Видання завдання працівнику.

Опис: я, як голова команди, хочу мати можливість видавати завдання учасникам створеної команди.

Критерії прийняття:

* голова команди може створювати завдання;
* голова команди може видавати завдання учасникам команди;
* голова команди може переглянути статус виконання завдання.

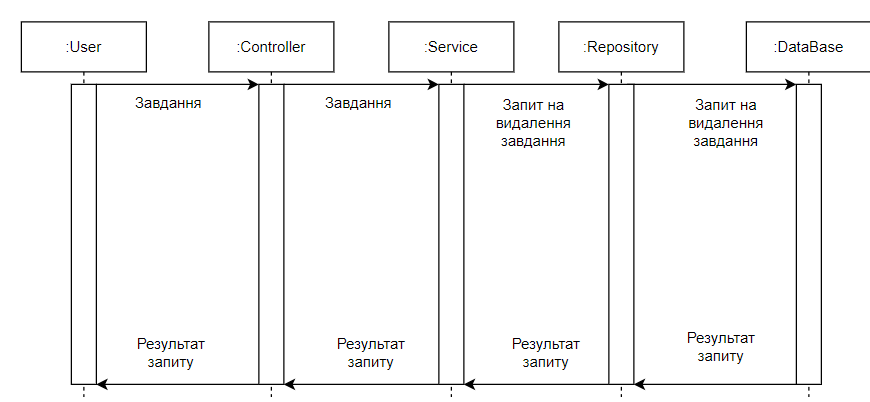


Рисунок 1.2 – Діаграма послідовності видалення завдання

Перегляд користувача по його електронній адресі.

Опис: Я, як голова команди, хочу мати можливість знайти потенційного учасника команди за допомогою його електронної адреси.

Критерії прийняття:

* голова команди має можливість знайти користувача за допомогої електронної адреси;
* головка команди має можливість переглянути інформацію про знайденого користувача.

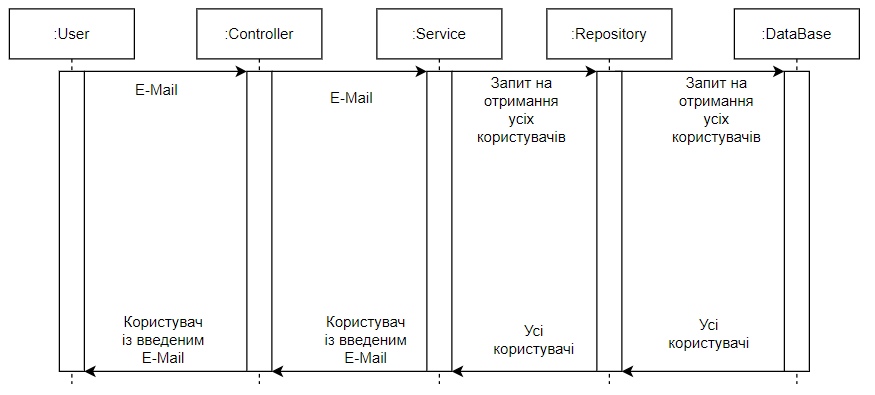


Рисунок 1.3 – Діаграма послідовності пошуку користувача за e-mail

Завершення завдання у випадку його виконання.

Опис: я, як учасник команди, хочу мати можливість завершити завдання на сайті у випадку мого виконання поставленої задачі.

Критерії прийняття:

* учасник команди може переглядати усі поставлені задачі;
* учасник команди може завершити завдання на сайті.

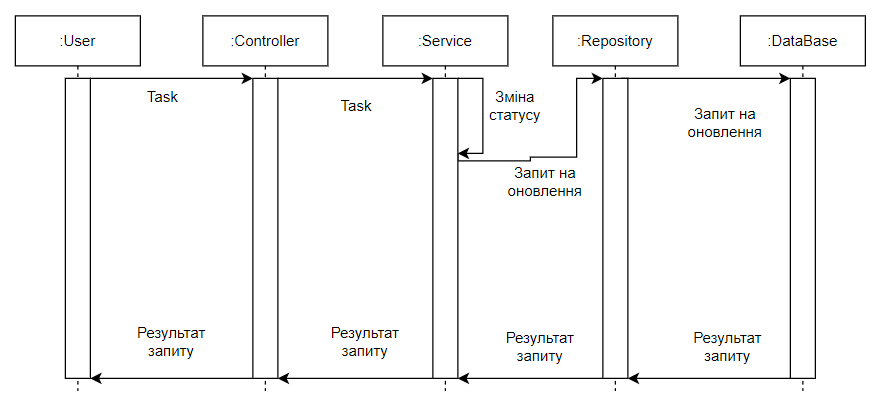


Рисунок 1.4 – Діаграма послідовності завершення завдання

## 1.2 Проектування бази даних

Проаналізувавши предмету області було виділено 4 таблиці які є необхідними для роботи майбутнього програмного продукту:

* таблиця із даними про користувача;
* таблиця із даними про команду;
* таблиця із даними про проєкт;
* таблиця із даними про завдання.

На основі цих чотирьох таблиць було спроектовано діаграму колекцій для майбутньої бази даних проекту. Діаграма колекцій зображена на рисунку 1.6.

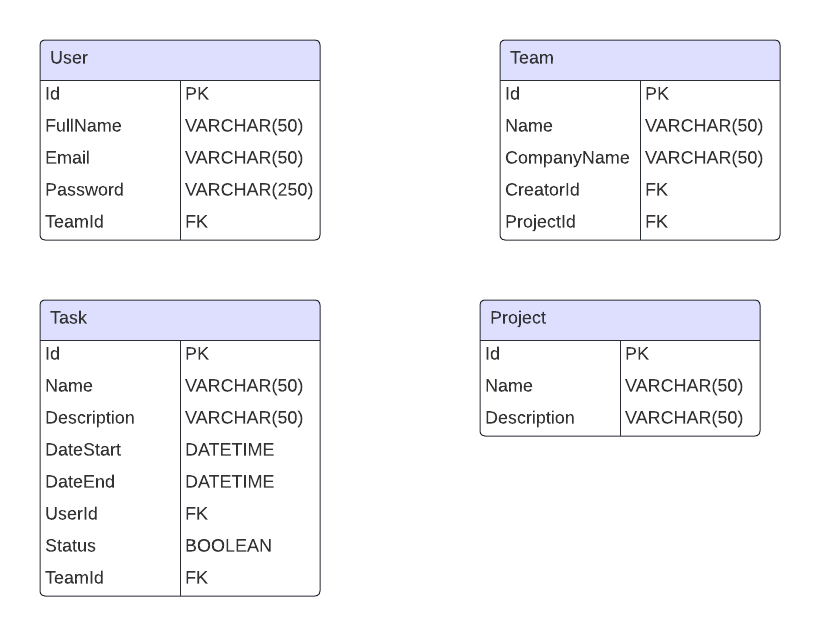


Рисунок 1.5 – Діаграма колекцій бази даних

Таблиця User відповідає за збереження користувачів й містить такі поля:

* id (первинний ключ);
* FullName (повне ім’я);
* Email (електронна адреса);
* Password (пароль);
* TeamId (зовнішній ключ команди користувача).

Таблиця Team відповідає за збереження команд й містить такі поля:

* id (первинний ключ);
* Name (назва команди);
* CompanyName (назва компанії);
* ProjectId (зовнішній ключ проекту);
* CreatorId (зовнішній ключ творця команди).

Таблиця Project відповідає за збереження даних про проекти й містить такі поля:

* id;
* Name;
* Description.

Таблиця Task відповідає за збереження задач й містить такі поля:

* id (первинний ключ);
* Name (назва);
* Description (опис);
* DateStart (дата початку);
* DateEnd (дата кінця);
* UserId (зовнішній ключ виконавця);
* Status (статус);
* TeamId (зовнішній ключ команди).

## 1.3 Проектування діаграми класів

На основі ER діаграми бази даних було розроблено діаграму класів програмного продукту. Діаграма класів зображена на рисунку 1.7.

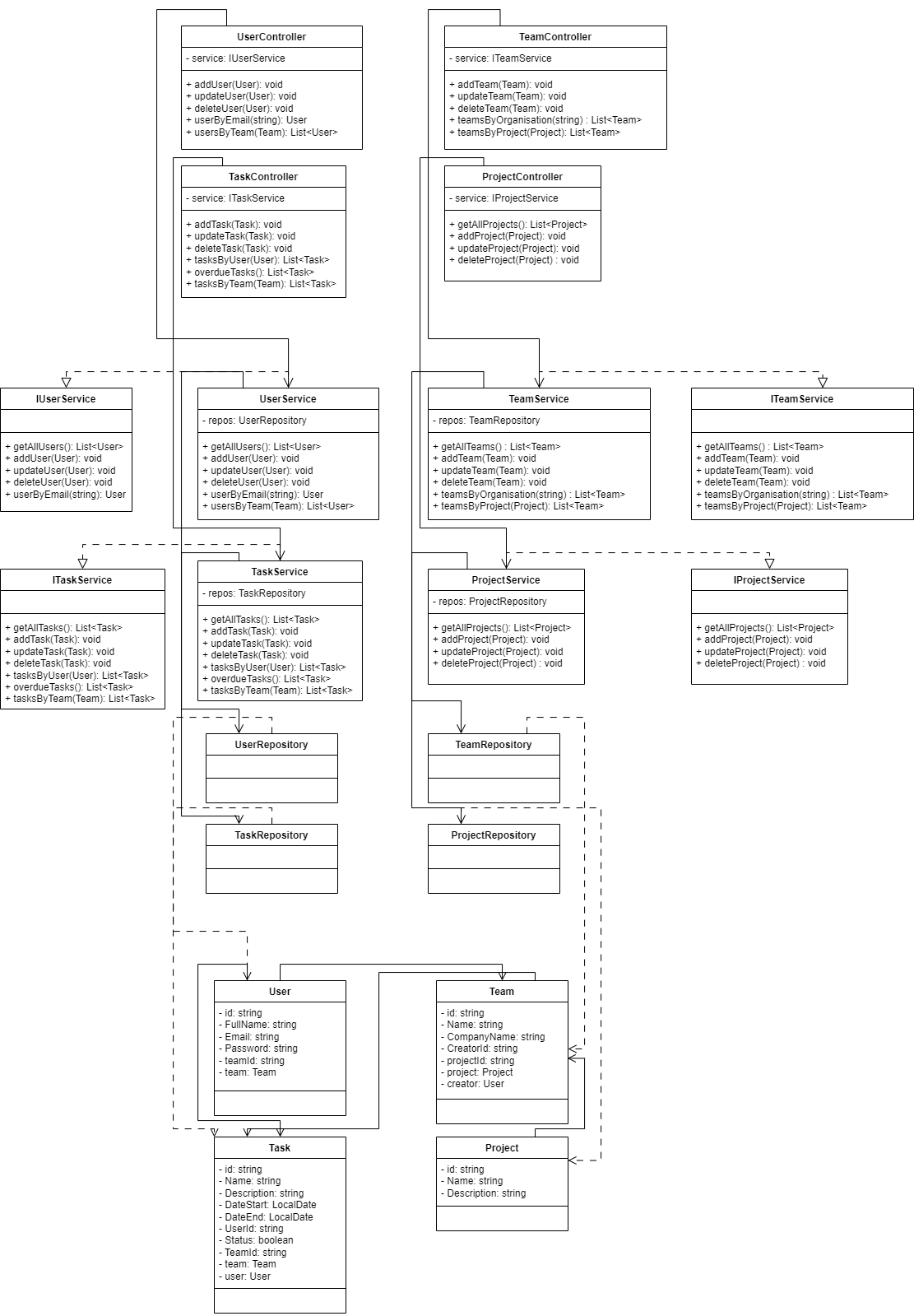


Рисунок 1.6 – Діаграма класів програмного продукту

Класи Task, User, Team, Project це класи моделі які являються уявленням даних, із таблиць у базі даних, у коді програмного продукту.

Класи UserRepository, TaskRepository, TeamRepository, ProjectRepository це класи репозиторії які напряму працюють із даними із бази даних, тобто ці класи роблять запити до бази даних та повертають, або вносять зміни у дані.

Класи UserService, TaskService, TeamService, ProjectService це класи сервіси у яких реалізується уся бізнес логіка програми. Сервіси отримують дані із репозиторіїв, оброблюють їх та передають у контролери.

Класи UserController, TaskController, TeamController, ProjectController це класи контролери які оброблюють запити зі сторони клієнта, потім роблять запити у сервіси за отриманням даних й повертають клієнту отримані із сервісів дані.

## 1.4 Написання тестових сценаріїв

Розробимо тестові сценарії для тестування функціоналу та стабільності програмного продукту.

Таблиця 1.1 – Тестові сценарії

|  |  |  |  |
| --- | --- | --- | --- |
| Id | Назва | Кроки | Статус |
| 1 | Створення команди | 1. Внесення даних потрібних для створення команди 2. Створення команди | Пройдено |
| 2 | Створення користувача | 1. Внесення даних потрібних для створення користувача 2. Створення користувача | Пройдено |
| 3 | Видалення користувача | 1. Внесення користувача для видалення 2. Видалення користувача | Пройдено |
| 4 | Редагування користувача | 1. Внесення нових даних про користувача 2. Внесення змін | Пройдено |
| 5 | Створення проєкту | 1. Внесення потрібних даних для проекту 2. Додавання проекту | Пройдено |

# 2. Реалізація API

## 2.1 Підготовка бази даних

Для використання бази даних потрібно зареєструватися на сайті MongoDB Atlas та створити базу даних. У базі даних було створено 4 колекції:

* User;
* Team;
* Project;
* Task.

Створена база даних зображена на рисунку 2.1.

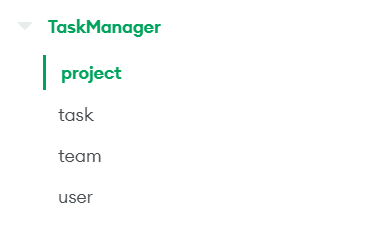


Рисунок 2.1. – Розроблена база даних та колекції до неї

## 2.2 Створення Spring boot Application

Для створення Spring boot Application потрібно зайти на сайт spring initialize, ввести потрібні дані та додати усі необхідні залежності до проекту. Процес створення Spring boot Application зображено на рисунку 2.2.

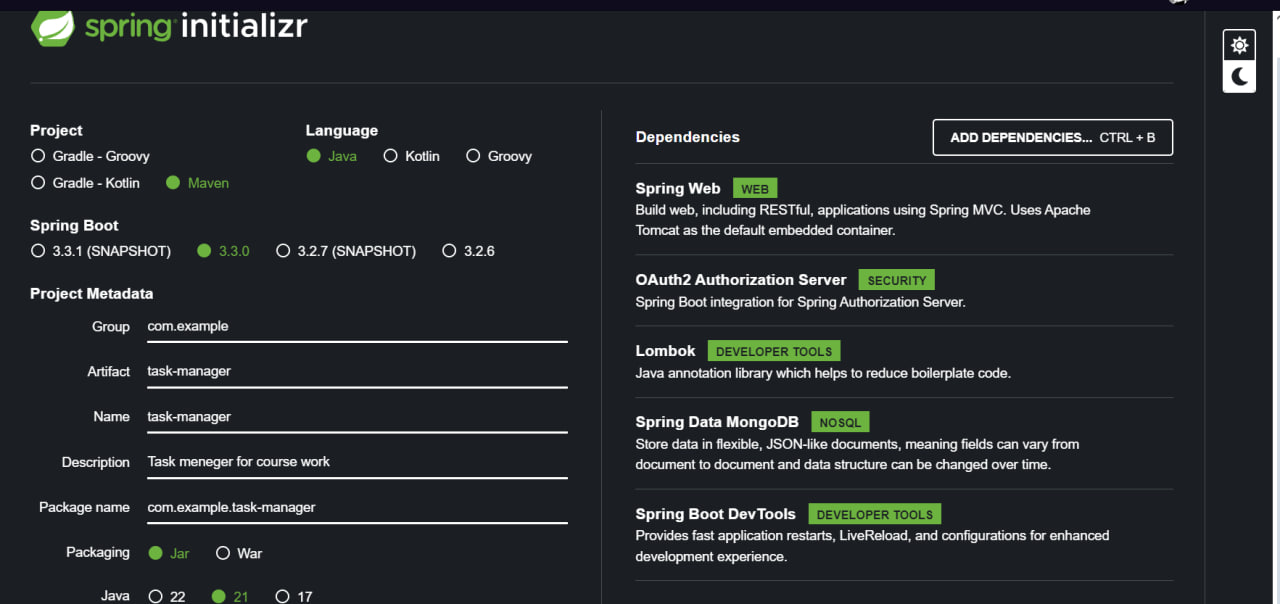


Рисунок 2.2 – Процес створення Spring boot Application

Після встановлення усіх необхідних налаштувань потрібно натиснути кнопку «Сгенерувати». На комп’ютер завантажиться архів проекту Spring boot Application. Каталог проекту який було завантажено зображено на рисунку 2.3.

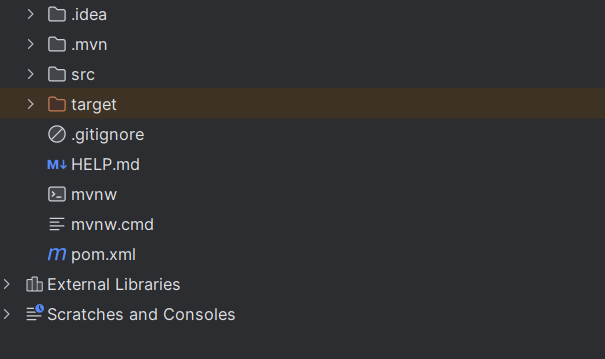


Рисунок 2.3 – Каталог проекту

## 2.3 Написання контролерів

У програмі було реалізовано 5 контролери:

* ProjectController (відповідає за запити пов’язані із проєктами);
* TaskController (відповідає за запити пов’язані із завданнями);
* TeamController (відповідає за запити пов’язані із командами);
* UserController (відповідає за запитом пов’язаними із користувачами);
* AuthController (відповідає за унтентифікацію).

Нижче наведено скріншоти реалізованих контролерів.

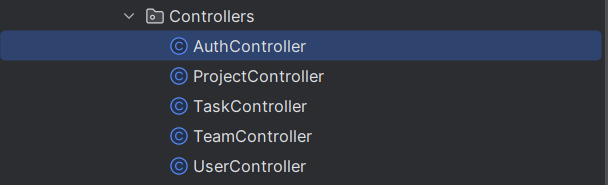


Рисунок 2.4 – Список усіх контролерів створених у програмі

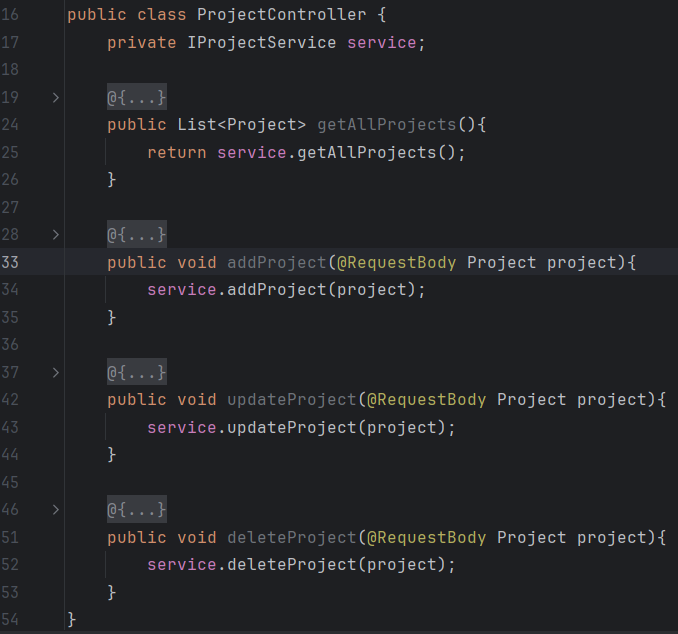


Рисунок 2.5 – Створений контролер ProjectController

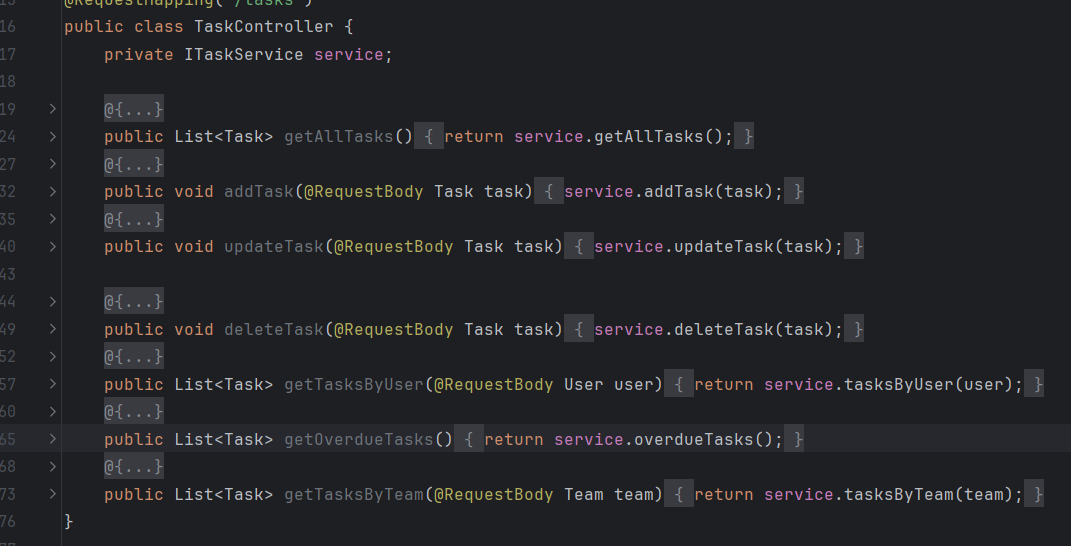


Рисунок 2.6 – Створений контролер TaskController



Рисунок 2.7 – Створений контролер TeamController

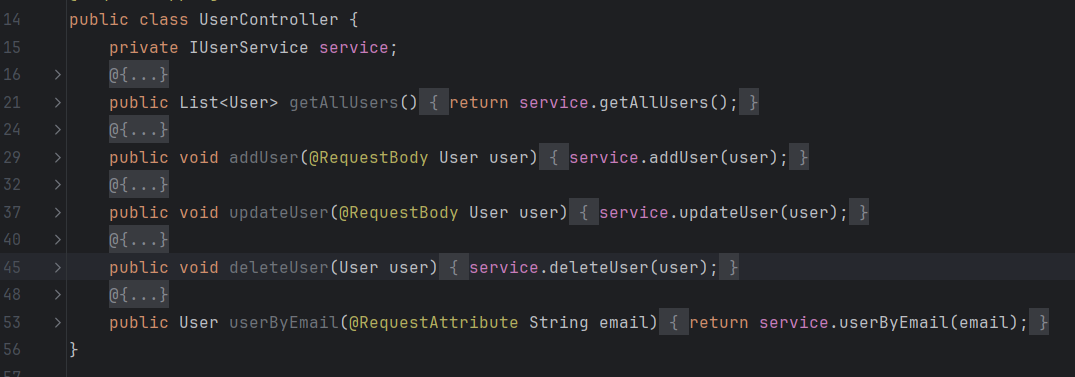


Рисунок 2.8 – Створений контролер UserController

## 2.4 Написання сервісів

У програмі було створено 4 сервіси:

* ProjectService (відповідає за проєкти);
* TaskService (відповідає за завдання);
* TeamService (відповідає за команди);
* UserService (відповідає за користувачів).

Нижче наведено скріншоти реалізованих сервісів.

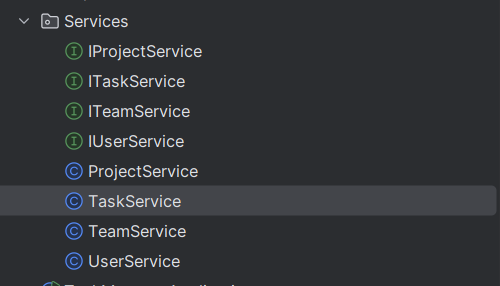


Рисунок 2.9 – Список усіх створених сервісів

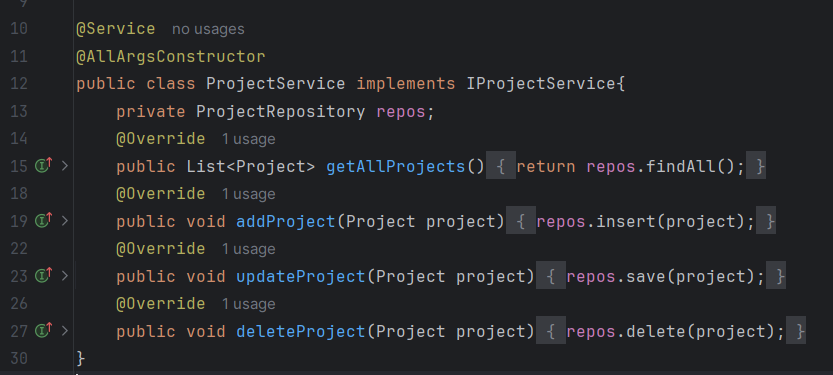


Рисунок 2.9 – Створений сервіс ProjectService

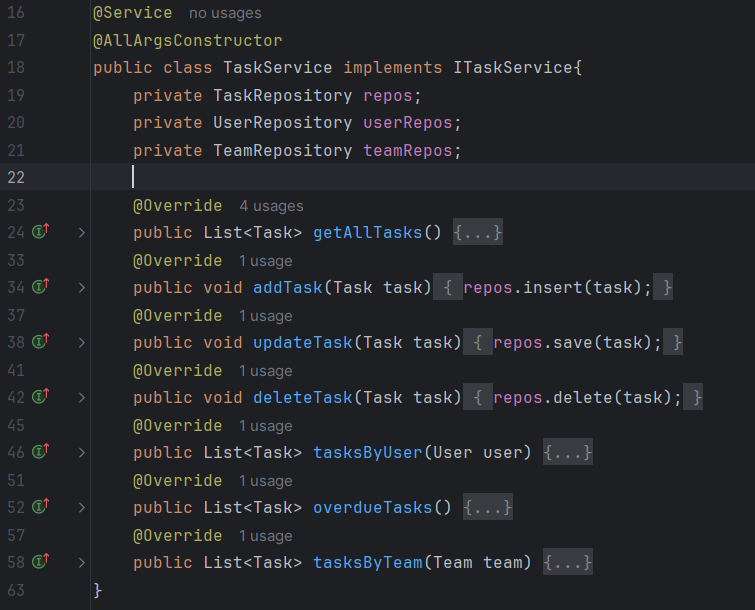


Рисунок 2.10 – Створений сервіс TaskService

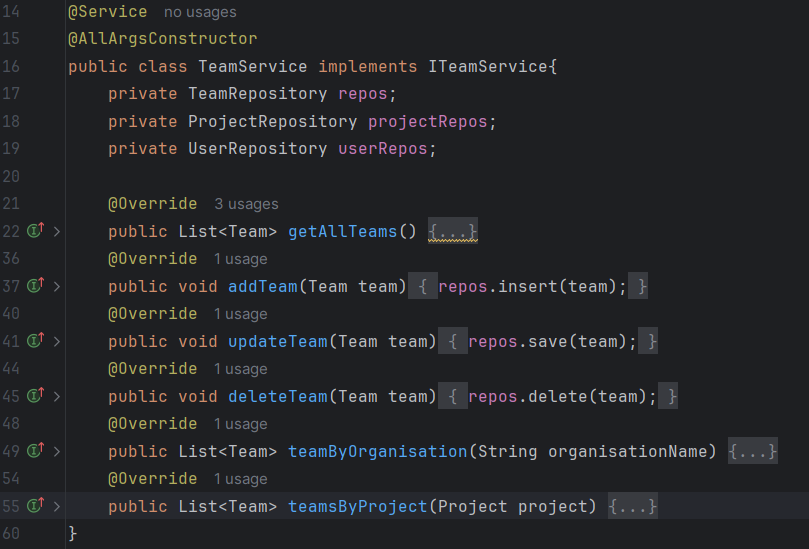


Рисунок 2.11 – Створений сервіс TeamService

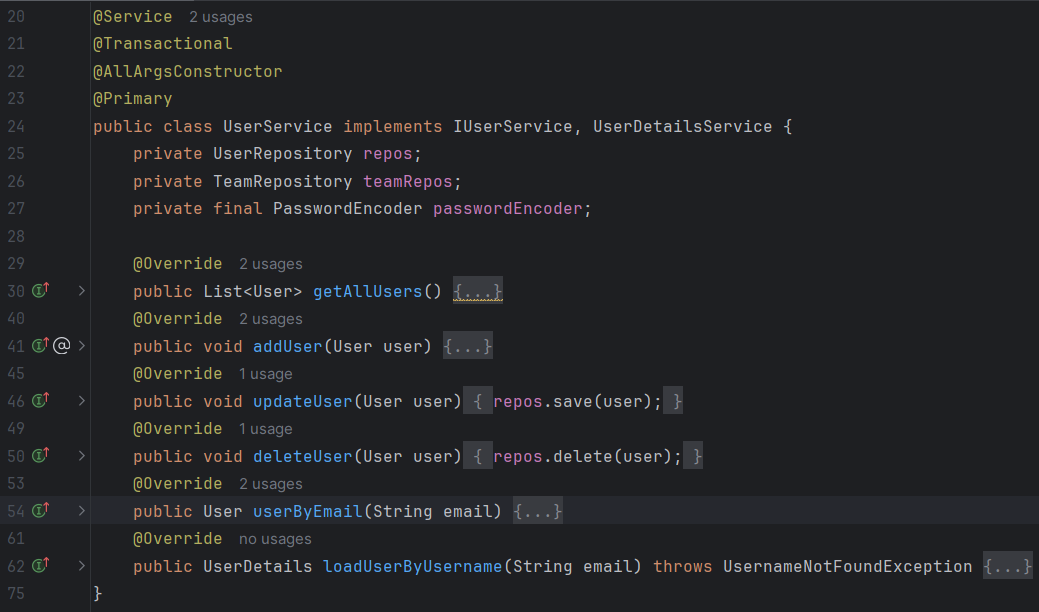


Рисунок 2.12 – Створений сервіс UserService

## 2.5 Написання Unit тестів

Створюванні Unit тести будуть розроблені лише для Services через недоречність створення Unit тестів для репозиторіїв та контролерів.

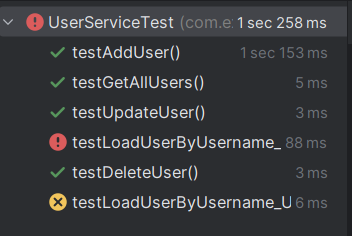


Рисунок 2.13 – Результат тестування UserService

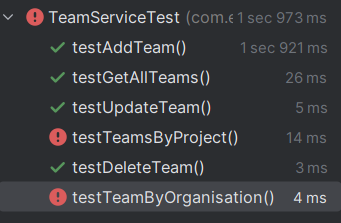


Рисунок 2.14 – Результат тестування TeamService



Рисунок 2.15 – Результат тестування TaskService

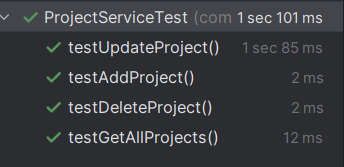


Рисунок 2.16 – Результат тестування ProjectService

Проаналізувавши помилки було зроблено висновок, що тести не пройдені через не правильно задані початкові дані. Протестувавши цей функціонал із використанням postman було отримано позитивний результат.

# 3. Підключення Swagger, системи авторизації з ролями

## 3.1 Підключення системи авторизації OAuth2

Для повноцінної роботи системи авторизації OAuth2 потрібно мати у програмі такі 3 компоненти:

* SecurityConfig;
* AuthController;
* UserDetailsService.

SecurityConfig та AuthController розроблені як нові повноцінні класи. UserDetailsService виконано у якості UserService який імплементує у собі інтерфейс UserDetailsService. Нижче наведено скріншоти створених файлів.

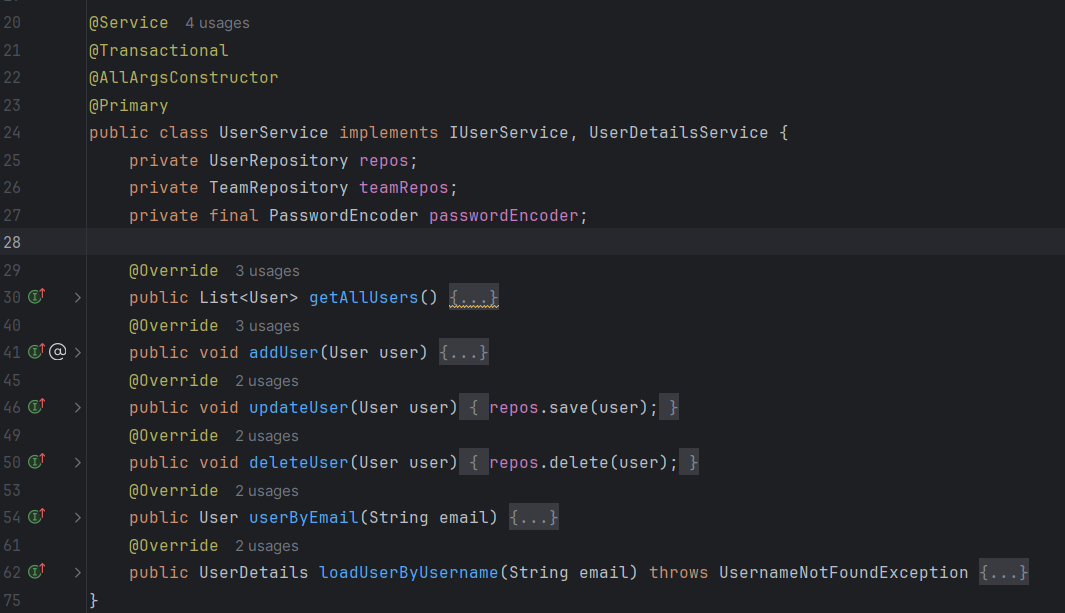


Рисунок 3.1 – Клас який імплементує у собі інтерфейс UserDetailsService

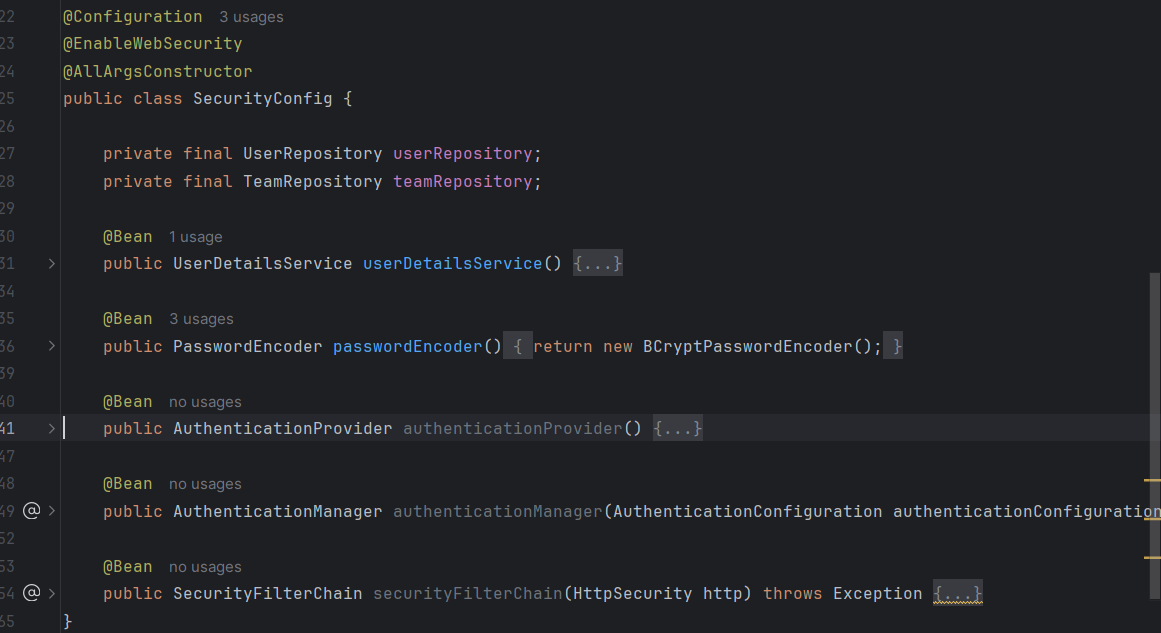


Рисунок 3.2 – Розроблений клас SecurityConfig

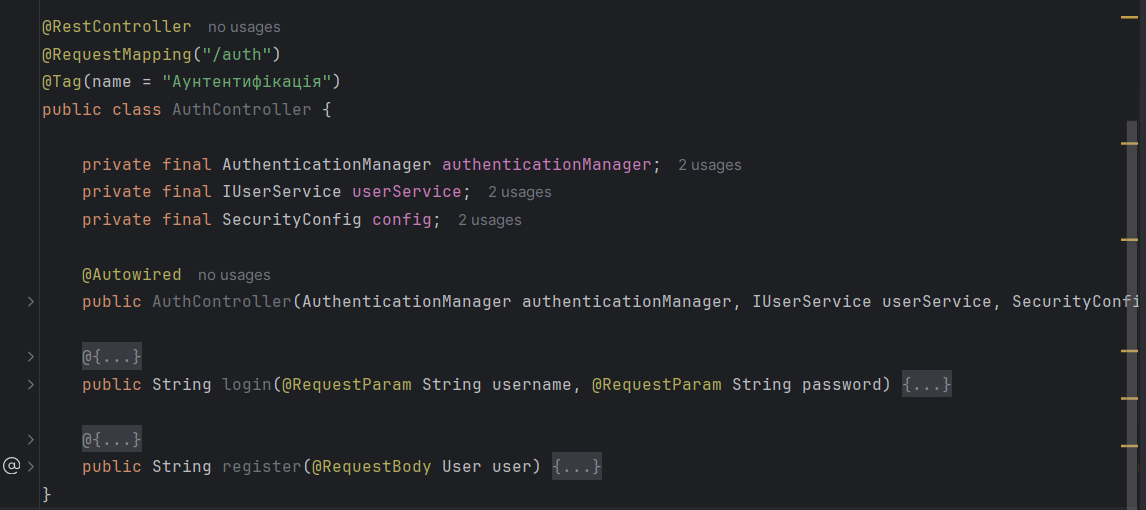


Рисунок 3.3 – Розроблений контролер AuthController

## 3.2 Підключення Swagger UI з детальним описом REST API

Підключення swagger UI відбувається за допомогою залежності open API, а опис за допомогою анотацій у самому коді. Нижче зображено скріншоти підключеного swagger UI та опис rest API.

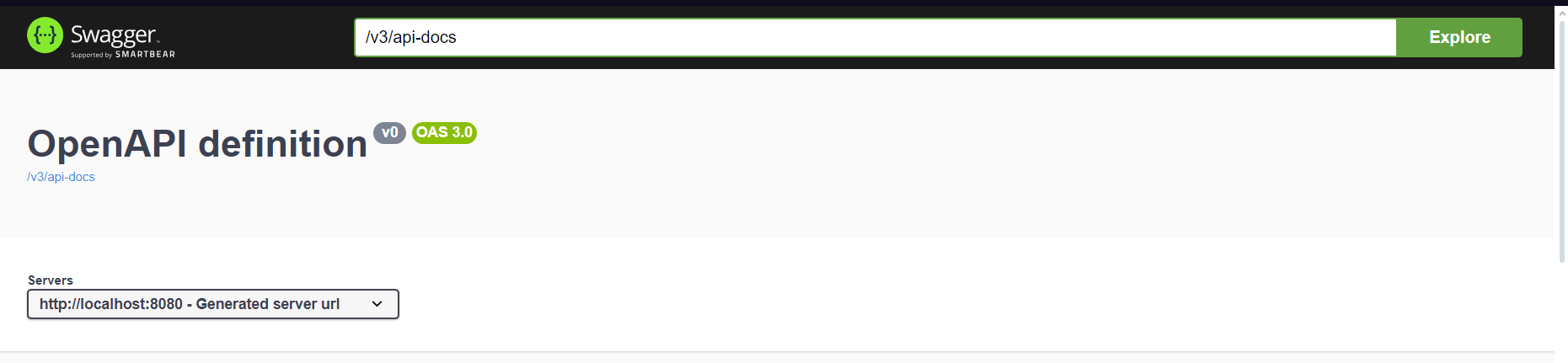


Рисунок 3.4 – Сторінка swagger UI

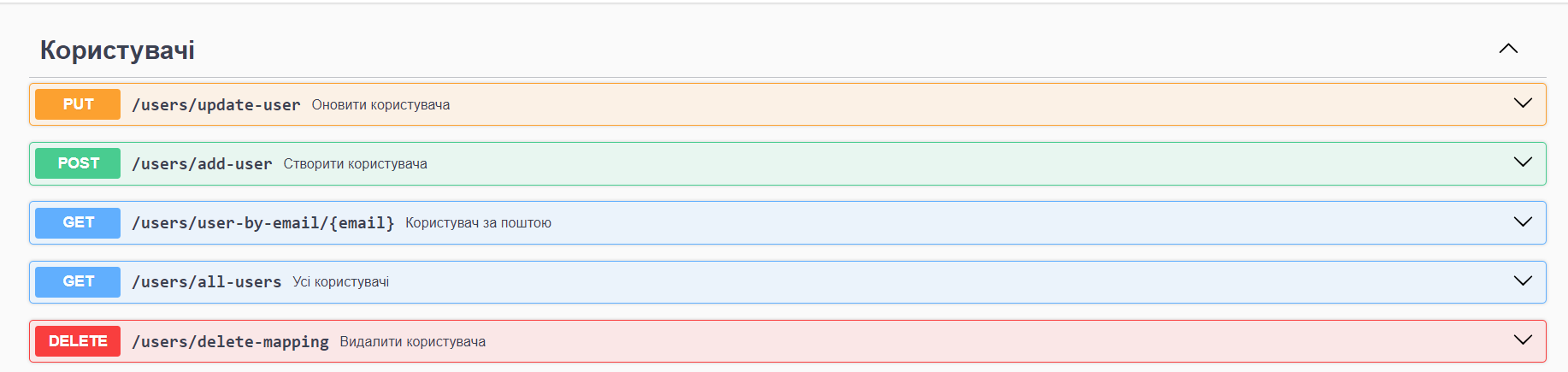


Рисунок 3.5 – Rest контролер який відповідає за користувачів

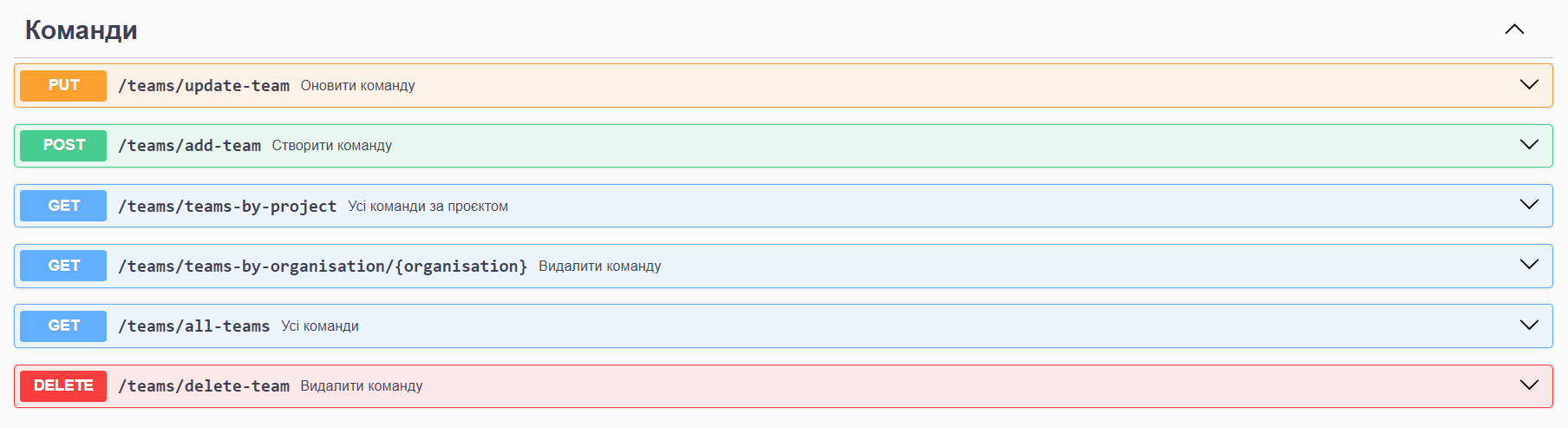


Рисунок 3.6 - Rest контролер який відповідає за команди

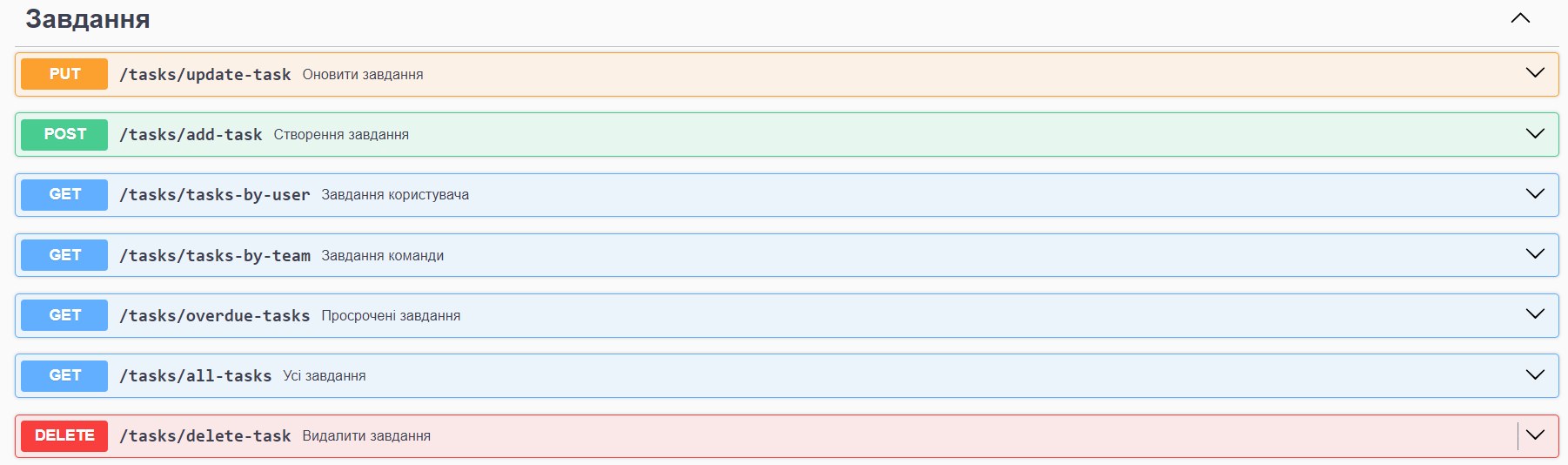


Рисунок 3.7 - Rest контролер який відповідає за завдання

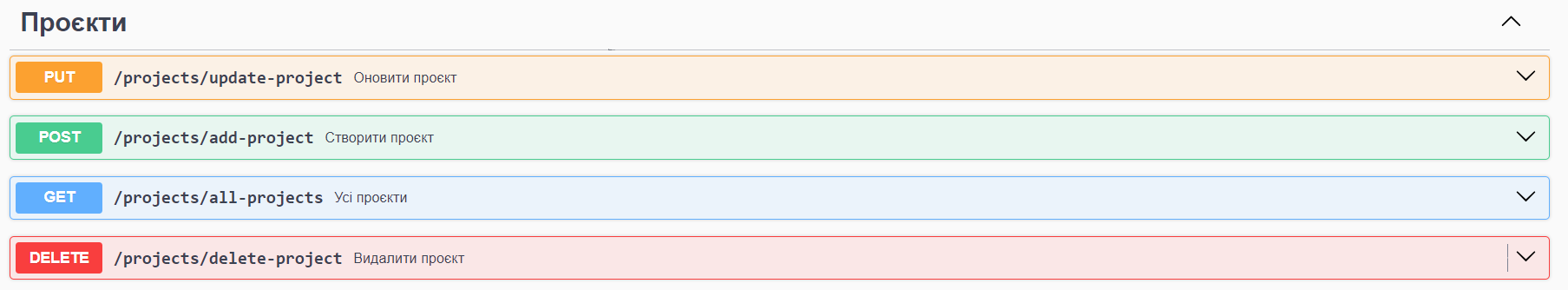


Рисунок 3.8 - Rest контролер який відповідає за проєкти

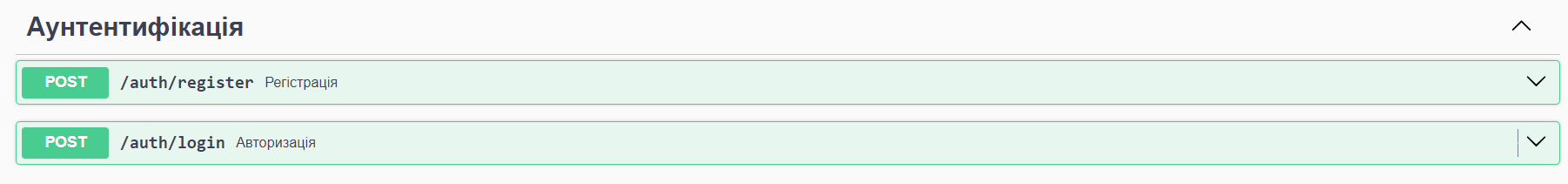


Рисунок 3.9 - Rest контролер який відповідає за аунтентифікацію

# 4 Інтеграційне тестування системи

Використовуючі user stories із розділу 1.1 було розроблено інтеграційні тести. Інтеграційні тести знаходяться у таблиці 4.1.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Id | Назва | Кроки | Очікувальний результат | Фактичний результат | Результат |
| 1 | Створення команди | 1. Введення усіх потрібних даних для створення команди 2. Відправка запиту на створення команди | Повідомлення про статус запиту 200 | Повідомлення про статус запиту 200 | Пройдено |
| 2 | Видання завдання працівнику | 1. Внесення усіх потрібних даних 2. Видання завдання | Повідомлення про статус запиту 200 | Повідомлення про статус запиту 200 | Пройдено |
| 3 | Перегляд користувача за його електронною адресою | 1. Ввід електронної адреси користувача 2. Відправка запиту на пошук | JSON файл із потрібним користувачем | JSON файл із потрібним користувачем | Пройдено |
| 4 | Завершення завдання |  | Повідомлення про завершення завдання | Повідомлення про завершення завдання | Пройдено |

# Висновок

В ході виконання курсового проекту було розроблено програмний застосунок з використанням фреймворку Spring Boot та мові програмування Java. В програмі було реалізовано RestAPI

Розроблений застосунок призначений для використання у системі командного менеджера задач. Програмний продукт підтримує роботу багатьох команд над одним проектом та видавання завдань учасникам команди та моніторинг виданих задач.

Для зручної роботи із RestAPI до програми було підключено swagger UI. Для користувачів не знайомих із API застосунку було описано методи та що вони виконують.