**Отчет по лабораторной работе № 4**

**Цели работы:**

* получить навыки проектирования и документирования API
* получить навыки создания окружения на основе технологии контейнеризации Docker.

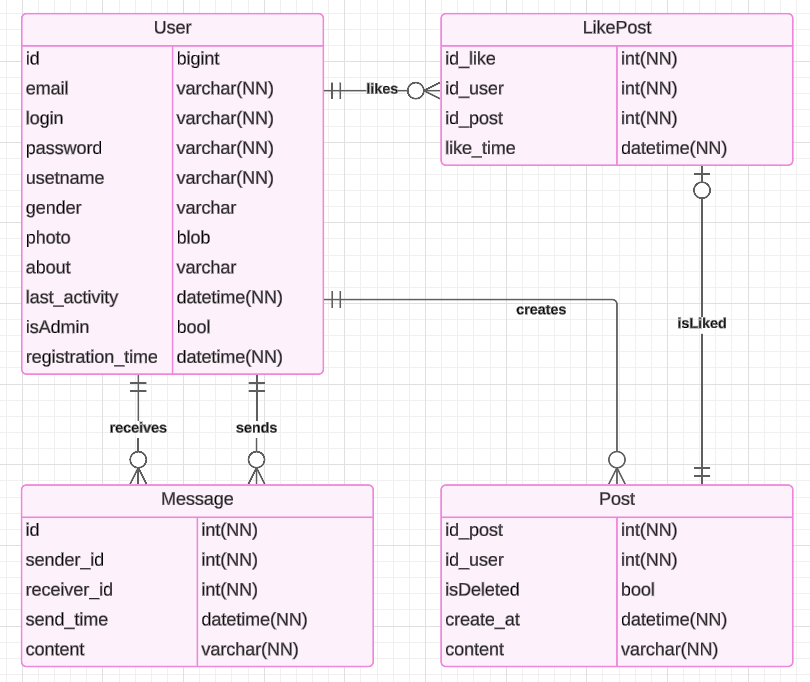
**Задание 1. Проектирование модели требований и реестра вариантов использования**

| **Актор** | **User Story** | **Use Case** |
| --- | --- | --- |
| Пользователь | US-1.1 Как пользователь социальной сети, я хочу зарегистрироваться, чтобы получить доступ к функционалу платформы. | UC-1.1 Зарегистрироваться на сайте. |
| Пользователь | US-1.2 Как пользователь социальной сети, я хочу авторизоваться, чтобы получить доступ к своему профилю. | UC-1.2 Войти в свой аккаунт. |
| Авторизованный пользователь | US-1.3 Как пользователь социальной сети, я хочу создавать посты, чтобы делиться своими мыслями. | UC-1.3 Создать новый пост. |
| Авторизованный пользователь | US-1.4 Как пользователь социальной сети, я хочу отправлять личные сообщения другим пользователям, чтобы общаться. | UC-1.4 Отправить сообщение. |
| Авторизованный пользователь | US-1.5 Как пользователь социальной сети, я хочу лайкать посты, чтобы выражать свою реакцию. | UC-1.5 Поставить лайк посту. |
| Администратор | US-2.1 Как администратор, я хочу просматривать список пользователей, чтобы управлять доступом к системе. | UC-2.1 Просмотреть список пользователей. |
| Администратор | US-2.2 Как администратор, я хочу удалять посты, чтобы поддерживать порядок в системе. | UC-2.2 Удалить пост пользователя. |
| Администратор | US-2.3 Как администратор, я хочу отслеживать активность пользователей, чтобы предотвращать нарушения. | UC-2.3 Посмотреть активность пользователя |

**Задание 2. Проектирование физической модели базы данных**

1. Логическая модель БД

| **Сущность** | **Таблица БД** | **Поле** | **Смысл поля** | **Тип данных** |
| --- | --- | --- | --- | --- |
| Пользователь | users | user\_id | Уникальный идентификатор пользователя | BIGINT |
| login | Логин пользователя | VARCHAR |
| email | Почта пользователя | VARCHAR |
| user\_password | Пароль пользователя | VARCHAR |
| username | Никнейм пользователя | VARCHAR |
| gender | Гендер пользователя | VARCHAR |
| about | Данные о пользователе | VARCHAR |
| photo | Фото пользователя | BLOB |
| last\_activity | Дата последней активности пользователя | TIMESTAMP |
| is\_admin | Является ли пользователь админом | BOOLEAN |
| created\_at | Дата создания аккаунта пользователя | TIMESTAMP |
| Сообщение | messages | message\_id | Уникальный идентификатор сообщения | BIGINT |
| user\_id | Уникальный идентификатор отправителя сообщения | BIGINT |
| user\_id | Уникальный идентификатор отправителя сообщения | BIGINT |
| content | Содержимое сообщения | VARCHAR |
| created\_at | Дата создания сообщения | TIMESTAMP |
| Пост | post | post\_id | Уникальный идентификатор поста | BIGINT |
| user\_id | Уникальный идентификатор пользователя | BIGINT |
| content | Содержимое поста | VARCHAR |
| created\_at | Дата создания поста | TIMESTAMP |
| Пост нравиться | likedPost | like\_id | Уникальный идентификатор лайка пользователя | BIGINT |
| user\_id | Уникальный идентификатор пользователя | BIGINT |
| post\_id | Уникальный идентификатор поста | BIGINT |
| created\_at | Дата поставить нравиться | TIMESTAMP |

2. ER-диаграмма базы данных:

3. сикуэль скрипт:

-- Создание базы данных

CREATE DATABASE SocialNetworkDB;

USE SocialNetworkDB;

-- Таблица User

CREATE TABLE User (

id BIGINT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

email VARCHAR(255) NOT NULL UNIQUE,

login VARCHAR(255) NOT NULL UNIQUE,

password CHAR(60) NOT NULL,

usename VARCHAR(255) NOT NULL,

gender VARCHAR(50),

photo BLOB,

about VARCHAR(500),

last\_activity DATETIME NOT NULL,

isAdmin BOOLEAN NOT NULL DEFAULT FALSE,

registration\_time DATETIME NOT NULL

);

-- Создание индекса для поля last\_activity

CREATE INDEX idx\_last\_activity ON User (last\_activity);

-- Таблица Post

CREATE TABLE Post (

id\_post INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

id\_user BIGINT NOT NULL,

isDeleted BOOLEAN NOT NULL DEFAULT FALSE,

create\_at DATETIME NOT NULL,

content VARCHAR(4000) NOT NULL,

FOREIGN KEY (id\_user) REFERENCES User(id)

);

-- Таблица LikePost

CREATE TABLE LikePost (

id\_like INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

id\_user BIGINT NOT NULL,

id\_post INT NOT NULL,

like\_time DATETIME NOT NULL,

FOREIGN KEY (id\_user) REFERENCES User(id),

FOREIGN KEY (id\_post) REFERENCES Post(id\_post)

);

-- Таблица Message

CREATE TABLE Message (

id INT NOT NULL AUTO\_INCREMENT PRIMARY KEY,

sender\_id BIGINT NOT NULL,

receiver\_id BIGINT NOT NULL,

send\_time DATETIME NOT NULL,

content VARCHAR(4000) NOT NULL,

FOREIGN KEY (sender\_id) REFERENCES User(id),

FOREIGN KEY (receiver\_id) REFERENCES User(id)

);

**Задание 3. Проектирование маршрутов и конечных точек в виде спецификации API**

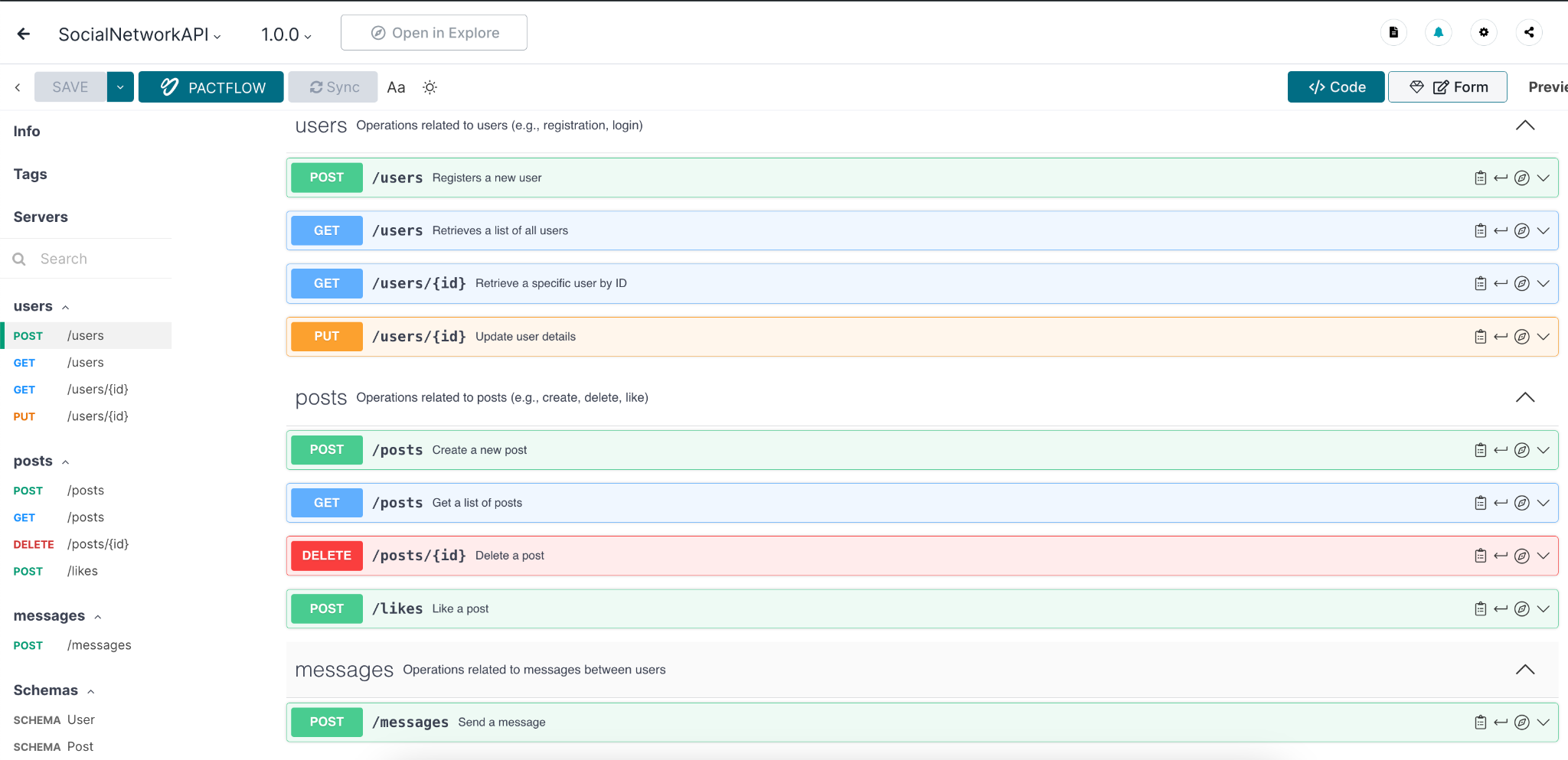
1. Маршруты и конечные точки

| **Сущность** | **Таблица БД** | **Маршрут** |
| --- | --- | --- |
| Пользователь | User | /users/ |
| Пользователь | User | /users/{id}/ |
| Пост | Post | /posts/ |
| Пост | Post | /posts/{id}/ |
| Лайк на пост | LikePost | /likes/ |
| Лайк на пост | LikePost | /likes/{id}/ |
| Сообщение | Message | /messages/ |
| Сообщение | Message | /messages/{id}/ |

2. Сопоставление маршрутов и конечных точек с вариантами использования

| **Актор** | **Use Case** | **Маршрут** | **HTTP-запрос** | **Аутентификация** |
| --- | --- | --- | --- | --- |
| Пользователь | UC-1.1 Зарегистрироваться на сайте | /users/ | POST | Нет |
| Пользователь | UC-1.2 Войти в свой аккаунт | /users/login/ | POST | Нет |
| Авторизованный пользователь | UC-1.3 Создать новый пост | /posts/ | POST | Да |
| Авторизованный пользователь | UC-1.4 Отправить сообщение | /messages/ | POST | Да |
| Авторизованный пользователь | UC-1.5 Поставить лайк посту | /likes/ | POST | Да |
| Администратор | UC-2.1 Просмотреть список пользователей | /users/ | GET | Да |
| Администратор | UC-2.2 Удалить пост пользователя | /posts/{id}/ | DELETE | Да |
| Администратор | UC-2.3 Посмотреть активность пользователя | /users/{id}/ | GET | Да |

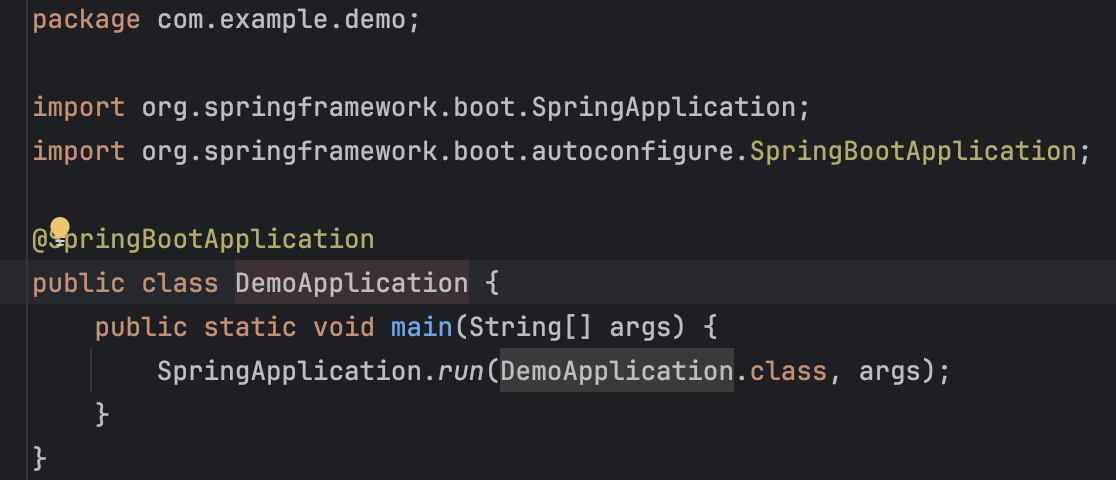
**Задание 4. Проектирование и тестирование API**

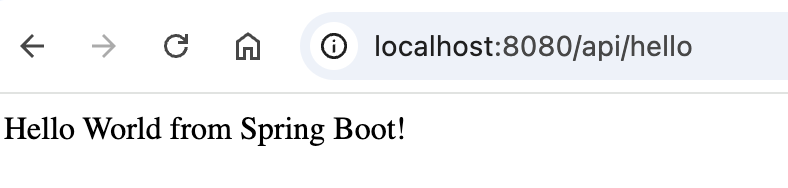


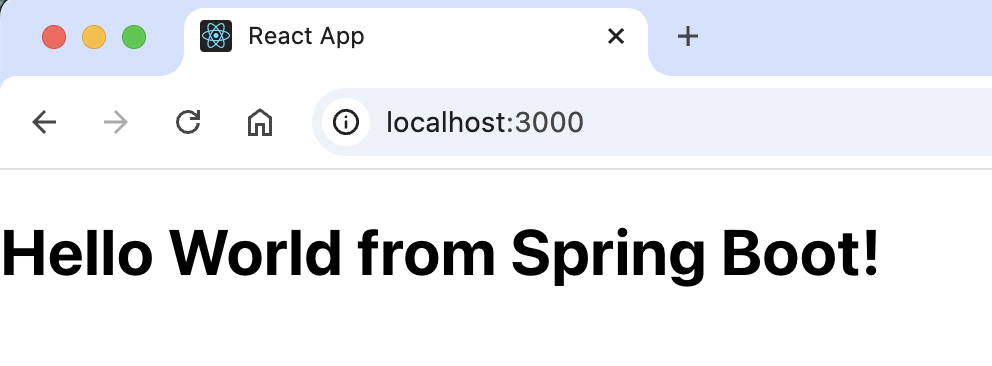
**Задание 5. Спроектировать инфраструктуру проекта и развернуть ее**

1) Создано простое приложение HelloWorld



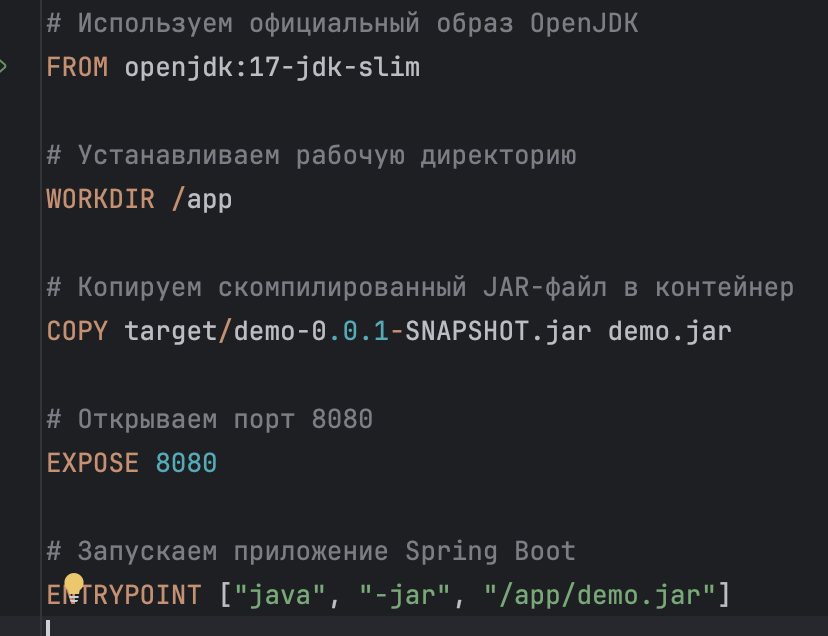


Результат:

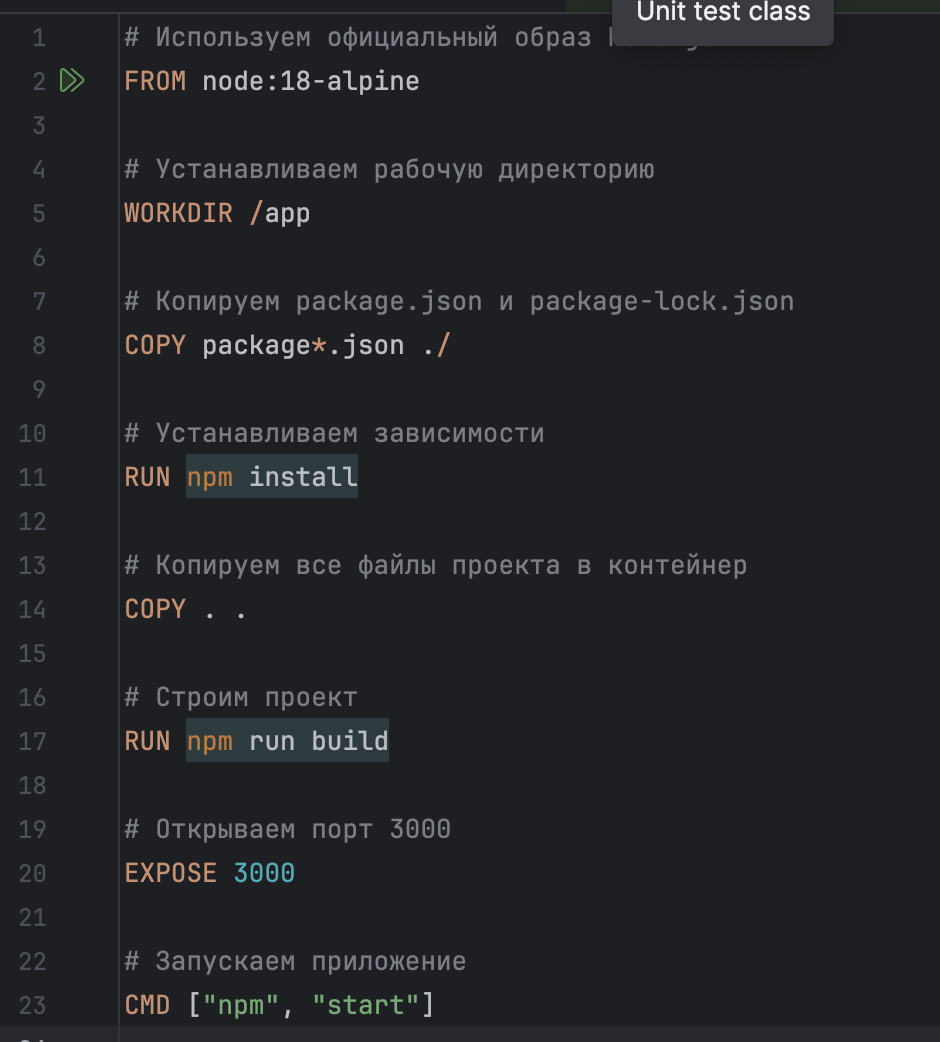


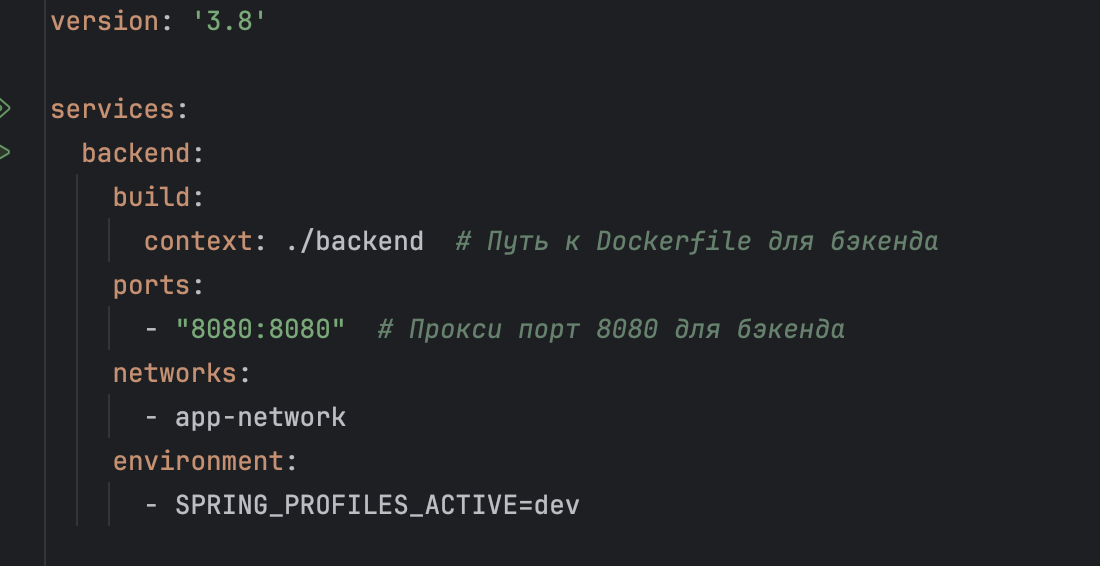
4) Docker

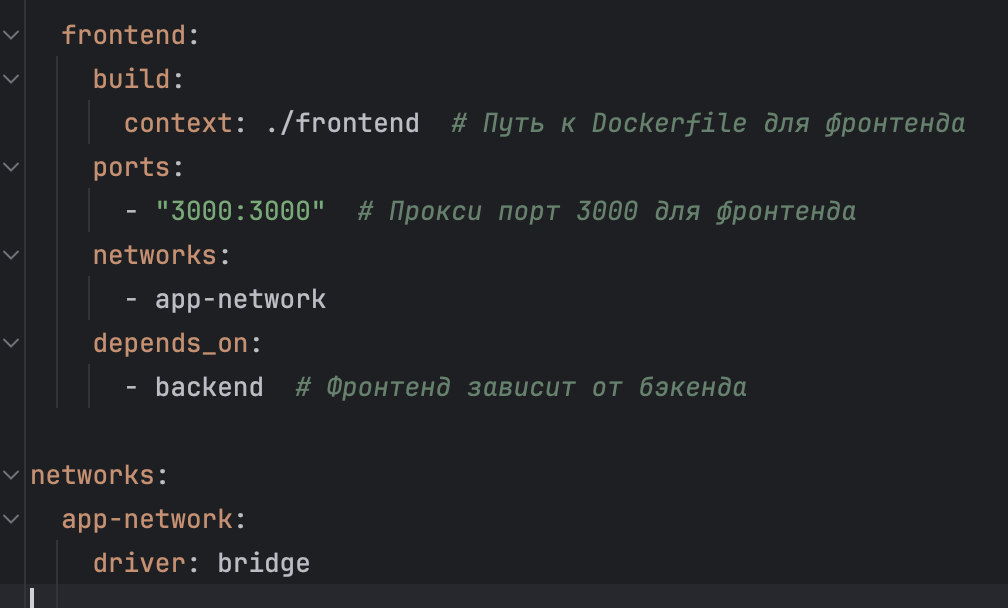
Dockerfile back



Dockerfile front

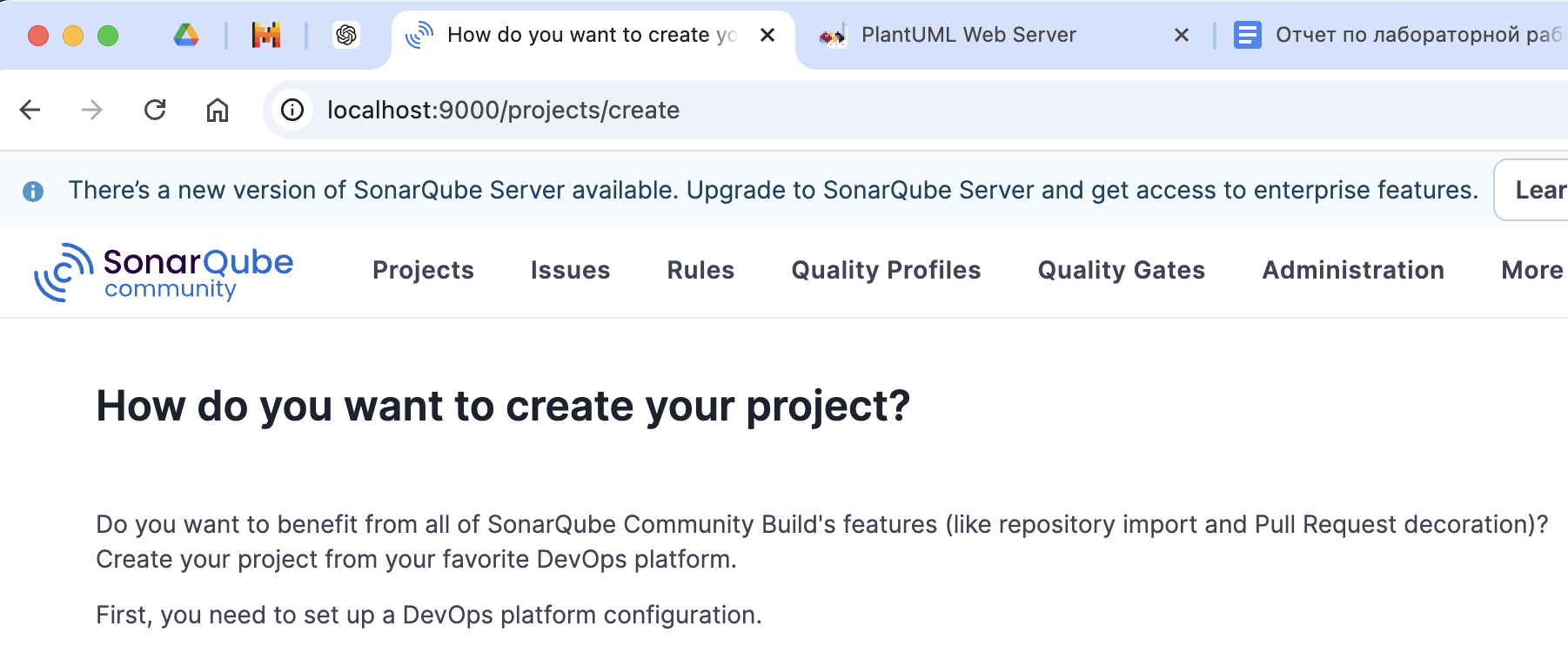


Docker-compose.yml backend

Docker-compose.yml frontend

5) CI/CD  


6) SonarQube



7) Диаграмма PlantUML: инфраструктура бекенд и веб-приложения на основе системы контейнеризации Docker инфраструктуры

