САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Практическая/Лабораторная работа

Выполнил:

Габов Михаил

К3340

Проверил:
Добряков Д. И.

Санкт-Петербург

2022 г.

1. Цель работы

Целью данной лабораторной работы являлась разработка серверной части приложения (RESTful API) с использованием фреймворка Express.js, языка TypeScript и ORM TypeORM на основе boilerplate

2. Описание предметной области

Платформа позволяет пользователям:

- Регистрироваться и аутентифицироваться в системе.
- Публиковать собственные рецепты, включая пошаговую инструкцию и список ингредиентов.
- Просматривать рецепты других пользователей.
- Оставлять комментарии к рецептам.
- Добавлять рецепты в "Избранное".
- "Лайкать" рецепты.
- Подписываться на других авторов.

Эта функциональность требует создания сложной структуры данных и набора эндпоинтов для управления всеми сущностями.

3. Проектирование API

3.1. Модели данных (Сущности)

Для представления данных предметной области в базе данных были спроектированы следующие сущности с помощью TypeORM. Ключевые модели и их связи описаны ниже:

- **User**: Основная сущность, представляющая пользователя. Содержит информацию для входа (логин, хешированный пароль), а также связи с рецептами, комментариями, лайками и т.д.
- **Recipe**: Сущность рецепта. Содержит название, описание, сложность, время приготовления и связь с автором (User).
- **Ingredient**: Сущность, хранящая название ингредиента (например, "Мука", "Сахар").
- **RecipeIngredient**: Связующая таблица между Recipe и Ingredient,

указывающая количество конкретного ингредиента в рецепте (например, "Мука - 200г").
- **RecipeStep**: Шаг приготовления в рецепте с описанием и порядковым номером.
- **Comment**: Комментарий пользователя к определенному рецепту.
- **Like**: Сущность, фиксирующая факт лайка от пользователя к рецепту.
- **Favorite**: Сущность, фиксирующая добавление рецепта в избранное пользователем.
- **Follow**: Сущность, представляющая подписку одного пользователя на другого.

Пример описания модели Recipe.ts:

src/models/Recipe.ts

```
import { Entity, PrimaryGeneratedColumn, Column, ManyToOne, OneToMany }
from 'typeorm';
import { User } from './User';
import { RecipeIngredient } from './RecipeIngredient';
import { RecipeStep } from './RecipeStep';
// ... другие импорты

@Entity()
export class Recipe {
  @PrimaryGeneratedColumn()
  id: number;

  @Column()
  title: string;

  @Column('text')
  description: string;

  // ... другие поля
```

```
  @ManyToOne(() => User, user => user.recipes)
  author: User;

  @OneToMany(() => RecipeIngredient, recipeIngredient =>
recipeIngredient.recipe)
  ingredients: RecipeIngredient[];

  @OneToMany(() => RecipeStep, step => step.recipe)
  steps: RecipeStep[];

  // ... другие связи
}
```

3.2. Спецификация эндпоинтов (Маршруты)

API предоставляет следующий набор эндпоинтов для взаимодействия с системой, сгруппированных по контроллерам:

- **User Controller (/users)**
  - POST /register: Регистрация нового пользователя.
  - POST /login: Аутентификация пользователя, получение JWT.
  - GET /me: Получение информации о текущем аутентифицированном пользователе.
- **Recipe Controller (/recipes)**
  - POST /: Создание нового рецепта (требует аутентификации).
  - GET /: Получение списка всех рецептов.
  - GET /:id: Получение детальной информации о конкретном рецепте.
  - PUT /:id: Обновление рецепта (доступно только автору).
  - DELETE /:id: Удаление рецепта (доступно только автору).
- **Comment Controller (/comments)**
  - POST /: Добавление комментария к рецепту (требует аутентификации).
  - GET /recipe/:recipeId: Получение всех комментариев для рецепта.
- **Like, Favorite, Follow Controllers**

- Реализуют CRUD-операции для соответствующих действий (поставить/убрать лайк, добавить/удалить из избранного, подписаться/отписаться). Все действия требуют аутентификации.

4. Ключевые аспекты реализации

4.1. Структура проекта

- src/controllers: Обработка HTTP-запросов, вызов сервисного слоя (в данном случае логика в контроллерах).
- src/models: Описание сущностей базы данных для TypeORM.
- src/dtos: Data Transfer Objects — объекты для передачи данных между клиентом и сервером, обеспечивают валидацию и четкий контракт API.
- src/middleware: Промежуточное ПО, в частности authMiddleware.ts для проверки JWT.
- src/data-source.ts: Конфигурация подключения к базе данных.
- src/index.ts: Точка входа в приложение, инициализация Express и middleware.

4.2. Аутентификация и авторизация

Безопасность API обеспечивается с помощью JWT.

1. При успешном логине (POST /users/login) пользователю выдается токен.
2. Для доступа к защищенным ресурсам клиент должен передавать этот токен в заголовке x-access-token.
3. Middleware authMiddleware.ts перехватывает запрос, проверяет валидность токена и, в случае успеха, добавляет информацию о пользователе в объект запроса.

6. Выводы

В результате выполнения лабораторной работы был реализован RESTful API для кулинарного приложения.