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.

Задание

Написать свой boilerplate на express + TypeORM + typescript. В структуре проекта должно быть явное разделение на:

- модели (сущности)
- контроллеры
- роуты

Реализация

1. Структура проекта

Для выполнения требования о явном разделении логики была создана следующая структура директорий:

```
/src
├── /controllers   # Логика обработки запросов
├── /entities      # Модели данных для TypeORM
├── /middleware    # Для аутентификации
├── /routes        # Файлы маршрутизации
└── index.ts       # Основной файл приложения
```

- **/entities**: Здесь определяются классы, которые TypeORM использует для создания таблиц в базе данных и для представления данных в приложении. Каждая сущность (например, User, Recipe) находится в своем файле.
- **/controllers**: Содержат функции-обработчики, которые вызываются при поступлении запроса на определенный эндпоинт. Контроллеры инкапсулируют бизнес-логику: валидацию входных данных, взаимодействие с моделями (через TypeORM) и формирование ответа.
- **/routes**: В этой директории определяются эндпоинты API. Каждый файл отвечает за определенную группу маршрутов (например, userRoutes.ts для пользователей, recipeRoutes.ts для рецептов). Они связывают HTTP-методы и пути с соответствующими функциями в контроллерах.
- **/middleware**: Содержит промежуточные функции, которые выполняются

перед основными обработчиками запросов. Реализован authMiddleware.ts с функцией protect для защиты роутов, требующих аутентификации пользователя.

2. Описание реализованных маршрутов (Routes)

На основе предоставленных файлов были реализованы следующие группы эндпоинтов:

Пользователи (userRoutes.ts)

- POST /api/users - Создание нового пользователя.
- POST /api/users/login - Аутентификация пользователя.
- GET /api/users - Получение списка всех пользователей (защищено).
- GET /api/users/search/by - Поиск пользователей (защищено).
- GET /api/users/:id - Получение пользователя по ID (защищено).
- PUT /api/users/:id - Обновление данных пользователя (защищено).
- DELETE /api/users/:id - Удаление пользователя (защищено).

Рецепты (recipeRoutes.ts)

- POST /api/recipes - Создание нового рецепта.
- GET /api/recipes - Получение списка рецептов.
- GET /api/recipes/:id - Получение рецепта по ID.
- PUT /api/recipes/:id - Обновление рецепта.
- DELETE /api/recipes/:id - Удаление рецепта.

Ингредиенты (ingredientRoutes.ts)

- POST /api/ingredients - Создание ингредиента.
- GET /api/ingredients - Получение всех ингредиентов.
- GET /api/ingredients/:id - Получение ингредиента по ID.
- PUT /api/ingredients/:id - Обновление ингредиента.
- DELETE /api/ingredients/:id - Удаление ингредиента.

Комментарии (commentRoutes.ts)

- POST /api/comments - Создание комментария (защищено).
- GET /api/comments - Получение комментариев для рецепта.

- DELETE /api/comments/:id - Удаление комментария (защищено).

Прочие (likeRoutes.ts, favoriteRoutes.ts, followRoutes.ts)

- Реализованы эндпоинты для создания и удаления лайков, добавлений в избранное и подписок.

Заключение

В ходе выполнения лабораторной работы был создан boilerplate для веб-приложения с использованием Express, TypeScript и TypeORM.