САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Домашняя работа №2

Выполнил:

Попов Никита

К3344

Проверил:
Добряков Д. И.

Санкт-Петербург

2026 г.

Задача

1. Опишите все функциональные и нефункциональные требования к вашему бэкенд-приложению
2. Выберите формат реализации API, которого вы будете придерживаться: REST API или Backend For Frontend
3. Спроектируйте все эндпоинты вашего API в формате OpenAPI-схемы с учётом реализованной диаграммы БД
4. Опишите тело каждого запроса и тело каждого ответа, продумайте возможные ошибки, возвращаемые из API

Ход работы

1. Общая характеристика проекта

Разработано бэкенд-приложение для сервиса бронирования ресторанов с возможностью:

- регистрации и аутентификации пользователей
- управления профилем
- создания и просмотра ресторанов
- управления меню ресторана
- добавления и просмотра отзывов
- создания и управления бронированиями

Система реализована в архитектурном стиле REST API.

2. Функциональные требования

2.1 Пользователи (Users)

Система должна обеспечивать:

1. Регистрацию пользователя
2. Аутентификацию (получение JWT)
3. Получение данных текущего пользователя
4. Обновление профиля пользователя

Роли:

- USER — обычный пользователь

- OWNER — владелец ресторана
- ADMIN — администратор системы

2.2 Рестораны (Restaurants)

Система должна позволять:

1. Просматривать список ресторанов
2. Просматривать конкретный ресторан
3. Создавать ресторан (только OWNER)
4. Получать отзывы ресторана
5. Добавлять отзыв к ресторану

2.3 Меню (Menu)

Система должна обеспечивать:

1. Получение меню ресторана
2. Добавление позиции в меню (только владелец ресторана)

2.4 Бронирования (Bookings)

Система должна обеспечивать:

1. Создание бронирования
2. Получение своих бронирований
3. Изменение статуса бронирования (подтверждение/отмена)

3. Нефункциональные требования

3.1 Производительность

- Время ответа API ≤ 300 мс при нормальной нагрузке
- Поддержка минимум 100 одновременных пользователей

3.2 Безопасность

- JWT-аутентификация
- Хеширование паролей (bcrypt)
- Ролевая модель доступа (RBAC)
- Валидация входных данных
- Защита от SQL-инъекций (ORM)

3.3 Надежность

- Обработка всех ошибок с корректными HTTP-кодами
- Логирование ошибок
- Транзакционность при создании бронирований

3.4 Масштабируемость

- Возможность горизонтального масштабирования
- Stateless API

3.5 Документирование

- Полная OpenAPI-документация
- Swagger UI

4. Выбранный формат API

Выбран формат: REST API

Причины выбора:

- Простота архитектуры
- Четкое соответствие CRUD-операциям
- Легкая интеграция с frontend
- Универсальность и стандартизация

Вывод

В рамках проекта реализован полнофункциональный REST API для сервиса бронирования ресторанов.

- Проработаны все CRUD-операции
- Реализована ролевая модель доступа
- Спроектирована OpenAPI-схема
- Учтены возможные ошибки
- Обеспечена безопасность и масштабируемость