САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

Дисциплина: Бэк-энд разработка

Отчет

Практическая работа 2

Выполнил:

Сокович Степан

К3344

Проверил:
Добряков Д. И.

Санкт-Петербург

2022 г.

Задача

Опишите все функциональные и нефункциональные требования к вашему бэкенд-приложению

Выберите формат реализации API, которого вы будете придерживаться: REST API или Backend For Frontend

Спроектируйте все эндпоинты вашего API в формате OpenAPI-схемы с учётом реализованной диаграммы БД

Опишите тело каждого запроса и тело каждого ответа, продумайте возможные ошибки, возвращаемые из API

Ход работы

Опишем требования системы:

1.1 Функциональные требования (FR)

Пользователи и доступ

Регистрация пользователя (email/phone + пароль).

Логин / логаут.

Просмотр и редактирование профиля.

Роли: TENANT (арендатор), LANDLORD (арендодатель),

Объявления (объекты недвижимости)

Создание объявления арендодателем.

Редактирование/деактивация объявления владельцем.

Просмотр карточки объявления.

Список объявлений с фильтрами и сортировками.

Загрузка/удаление фото.

Поиск

Фильтры: город/район, цена min/max, кол-во комнат, тип жилья, площадь, дата доступности.

Сортировки: по цене, по дате публикации.

Бронирования

Создание бронирования арендатором на даты.

Статусы бронирования: PENDING, APPROVED, REJECTED, CANCELLED.

Арендодатель подтверждает/отклоняет.

Арендатор может отменить по правилам.

Создание “инвойса” на оплату бронирования.

Отметка статуса платежа (в реальном мире — через платежный провайдер и webhooks).

Статусы

Отзывы

Арендатор оставляет отзыв по завершённому бронированию.

Просмотр отзывов по объявлению.

1.2 Нефункциональные требования (NFR)

Способность к масштабированию

Низкий 95-й перцентиль ответа list/search: ≤ 300–500 мс (без учёта CDN для картинок).

Пагинация обязательна для списков.

Безопасность

JWT access + refresh.

Хеширование паролей.

Валидация входных данных.

Надёжность

Идемпотентность для критичных операций

Транзакции при смене статусов брони/платежей.

Сопровождаемость

Версионирование API: /api/v1.

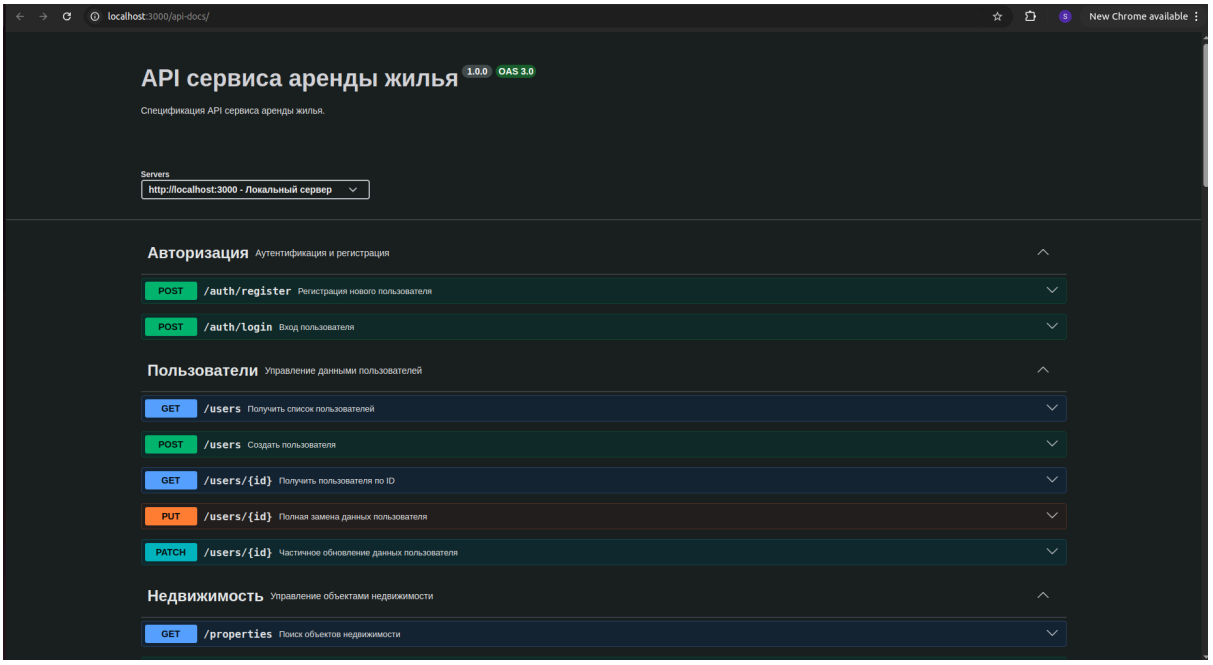
Единый формат ошибок.

Наблюдаемость

Логи запросов (без чувствительных данных).

Метрики (RPS, latency, errors).

Спроектируем контракт будущего сервиса и сгенерируем swagger-документацию по нему



Вывод

Мы составили список функциональных и нефункциональных требований к системе, а также разработали и структурировали его контракт