

**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО**

Дисциплина: Бэк-энд разработка

Отчет

Домашняя работа №1

Выполнил:

Шалунов Андрей

Группа К3440

**Проверил:
Добряков Д. И.**

Санкт-Петербург

2025 г.

Задача

Необходимо спроектировать набор следующих диаграмм:

- общая архитектура решения (сервисы и их взаимосвязи, клиент-серверное взаимодействие).
- диаграмма компонентов.
- диаграммы БД по каждому сервису.
- диаграммы основных пользовательских сценариев (те сценарии, которые позволяют вашим приложением полноценно воспользоваться, пройти весь путь).

Ход работы

Я выбрал вариант №3 с прошлого семестра.

Сервис для аренды недвижимости:

- Вход
- Регистрация
- Личный кабинет пользователя (список арендованных и арендуемых объектов)
- Поиск недвижимости с фильтрацией по типу, цене, расположению
- Страница объекта недвижимости с фото, описанием и условиями аренды
- История сообщений и сделок пользователя

С помощью [draw.io](#) получилось спроектировать общую архитектуру сервиса для аренды недвижимости:

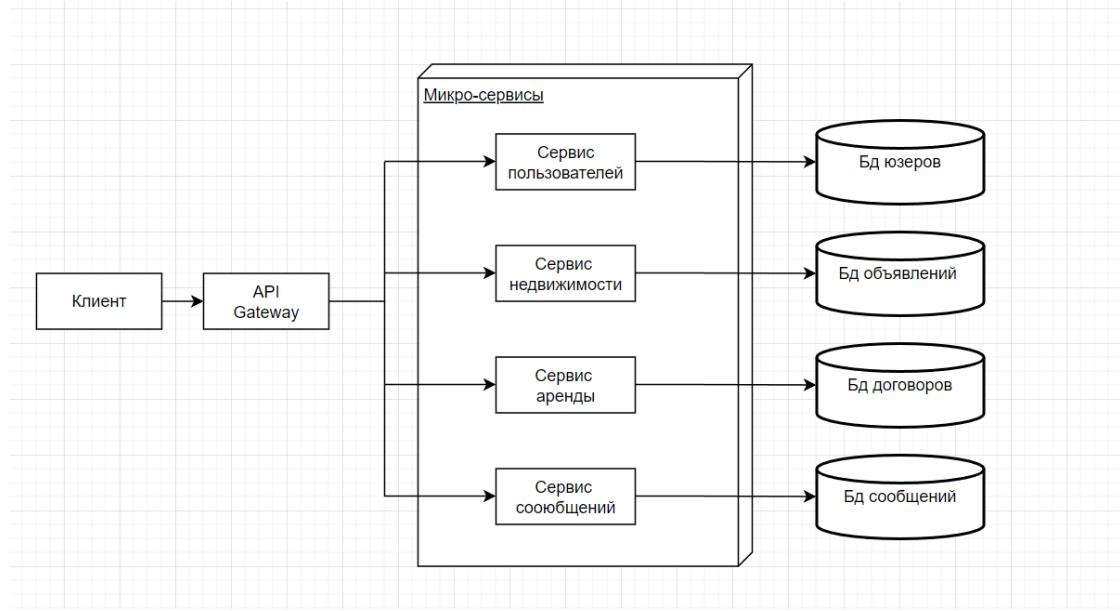


Рисунок 1 - Общая архитектура решения

На схеме показаны четыре микросервиса:

Пользователи - регистрация, авторизация, профили.

Недвижимость - объявления.

Аренда - договоры и статусы.

Сообщения - диалоги между пользователями.

Все внешние запросы идут от клиента к API Gateway, который маршрутизирует вызовы к нужному сервису. У каждого сервиса своя БД, что обеспечивает изоляцию доменов и независимое масштабирование.

Диаграмма БД

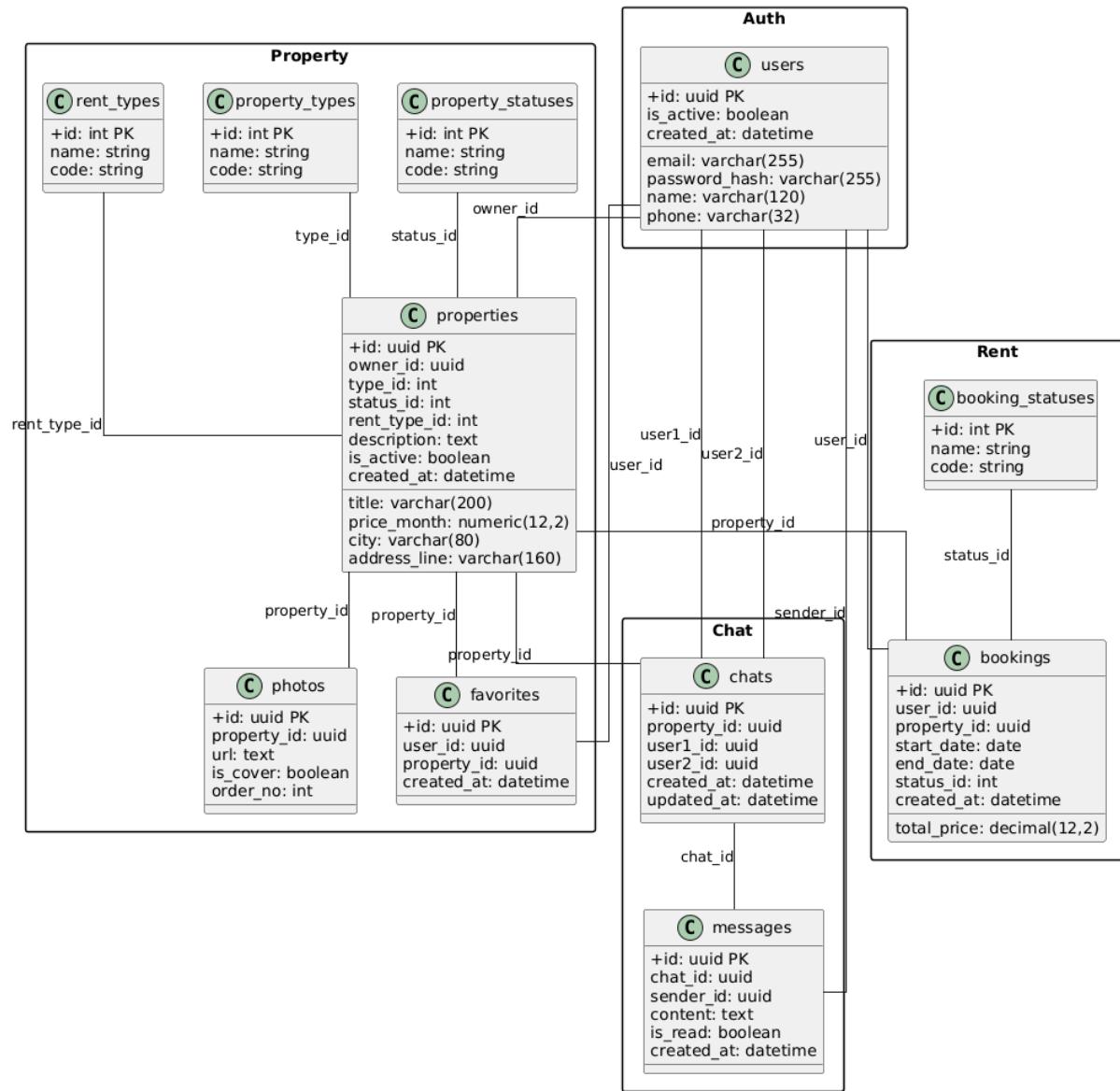


Рисунок 2 - Диаграмма бд

Диаграмма базы данных показывает четыре домена микросервисов и их связи. В Auth хранится таблица пользователей, которую используют остальные сервисы через идентификаторы. В Property находятся справочники типов, статусов и видов аренды, основная таблица объявлений, а также фото объектов и избранное. В Chat описаны чаты между двумя пользователями по конкретному объявлению и сообщения внутри этих чатов. В Rent хранится информация о бронированиях с датами, суммой и статусом на основе справочника статусов. Связи проходят по полям `user_id`, `owner_id` и `property_id`, что разделяет

ответственность сервисов и поддерживает все ключевые сценарии от регистрации и просмотра карточек с фото до переписки и истории бронирований.

Диаграмма компонентов

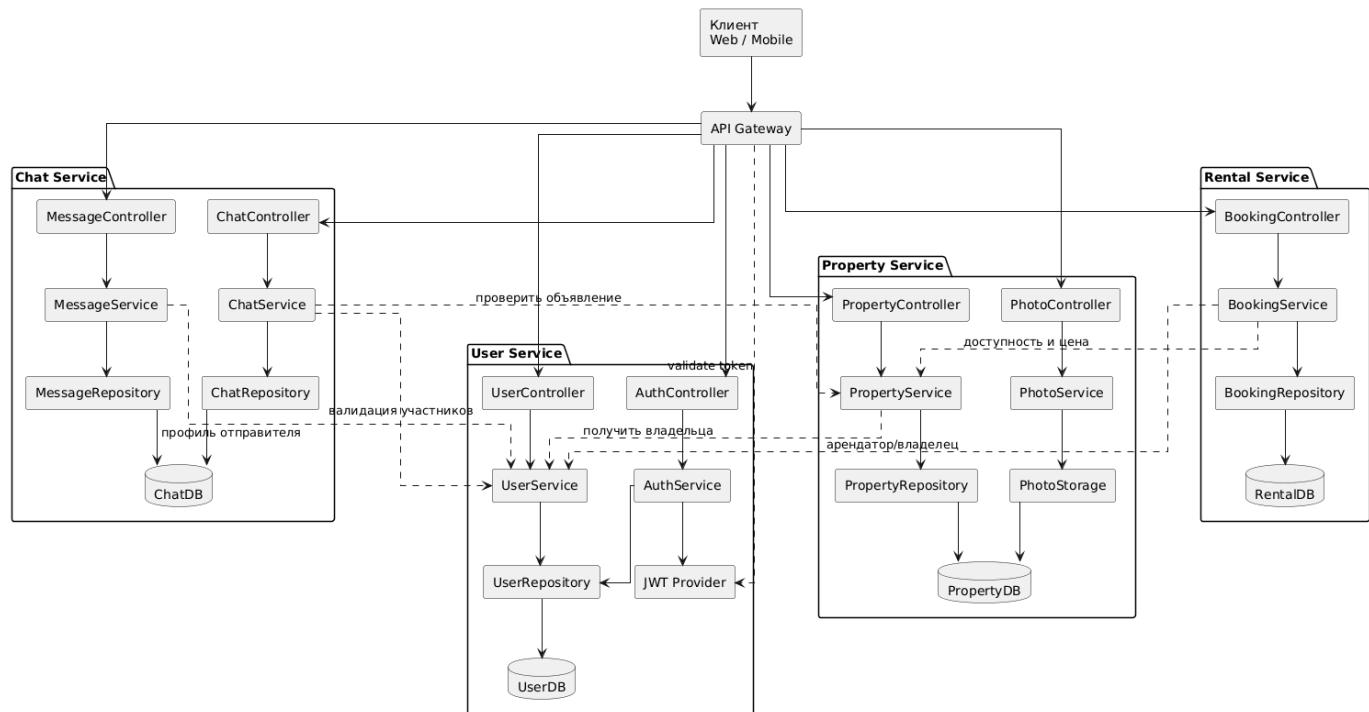


Рисунок 3 - Диаграмма компонентов

Диаграмма компонентов показывает, как клиентские запросы проходят через API Gateway к контроллерам каждого микросервиса, где логика разделена на уровни контроллеров, сервис и репозиторий. У каждого сервиса своя база данных, что обеспечивает изоляцию и независимое масштабирование. User Service отвечает за регистрацию, аутентификацию и профили, Property Service за объявления и фото, Chat Service за чаты и сообщения, Rental Service за бронирования. Межсервисные вызовы выполняются по REST только на уровне сервисов например бронирование запрашивает данные о владельце и объекте у User Service и Property Service.

Диаграммы пользовательских сценариев

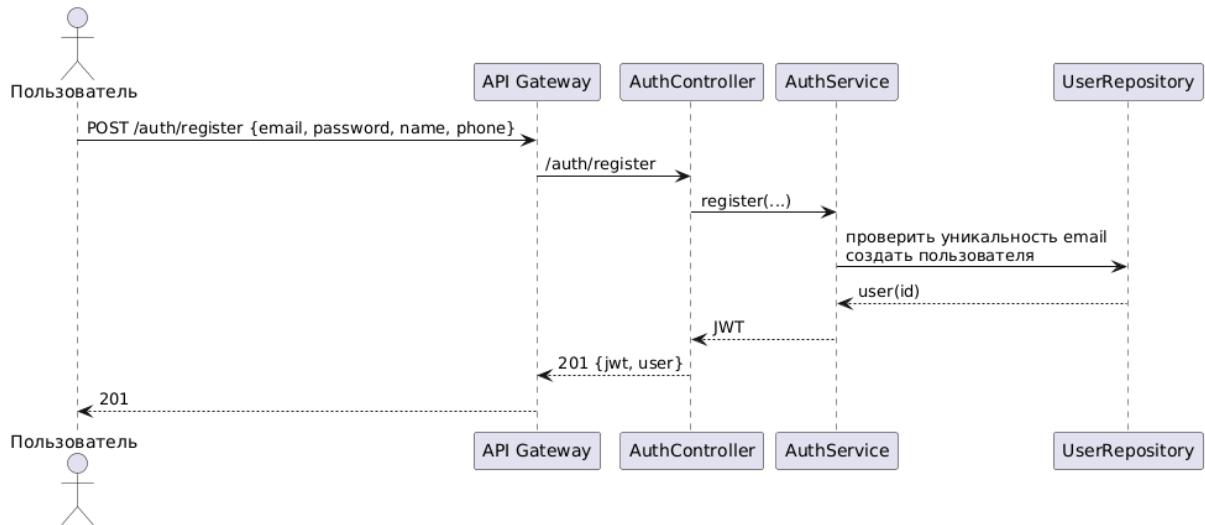


Рисунок 4 - Регистрация пользователя

Сервис создает пользователя и сразу возвращает токен авторизации.

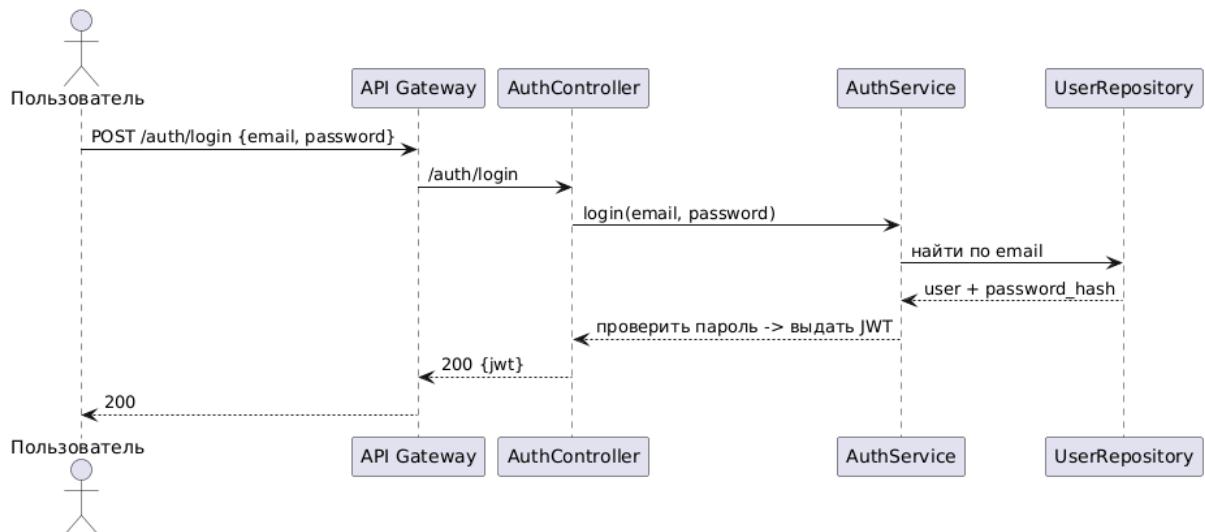


Рисунок 5 - Логин

Проверка пары логин/пароль и выдача JWT.

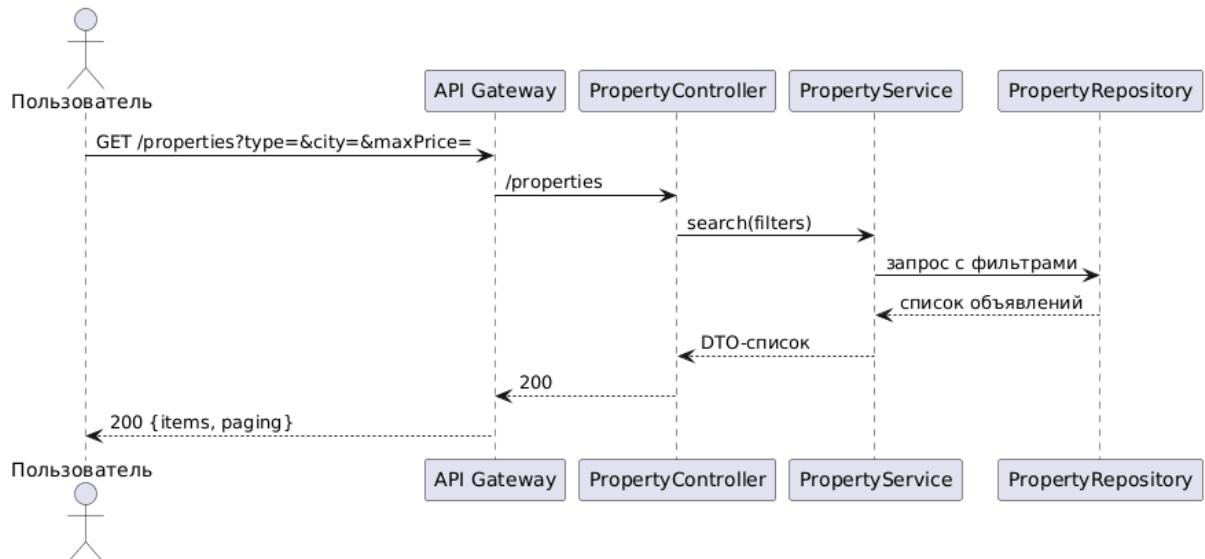


Рисунок 6 - Поиск недвижимости с фильтрами

Контроллер получает фильтры, сервис формирует запрос в БД и возвращает страницу результатов.

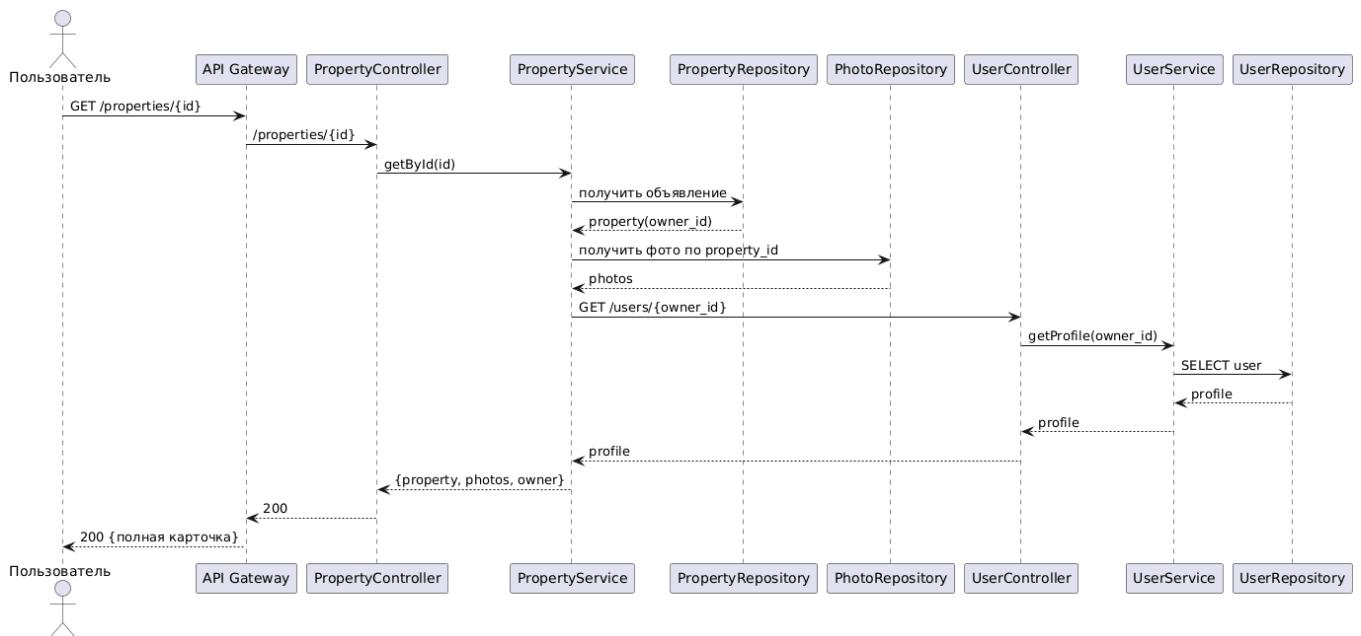


Рисунок 7 - Просмотр карточки объекта

Агрегируется объект, его фото и профиль владельца через межсервисный вызов.

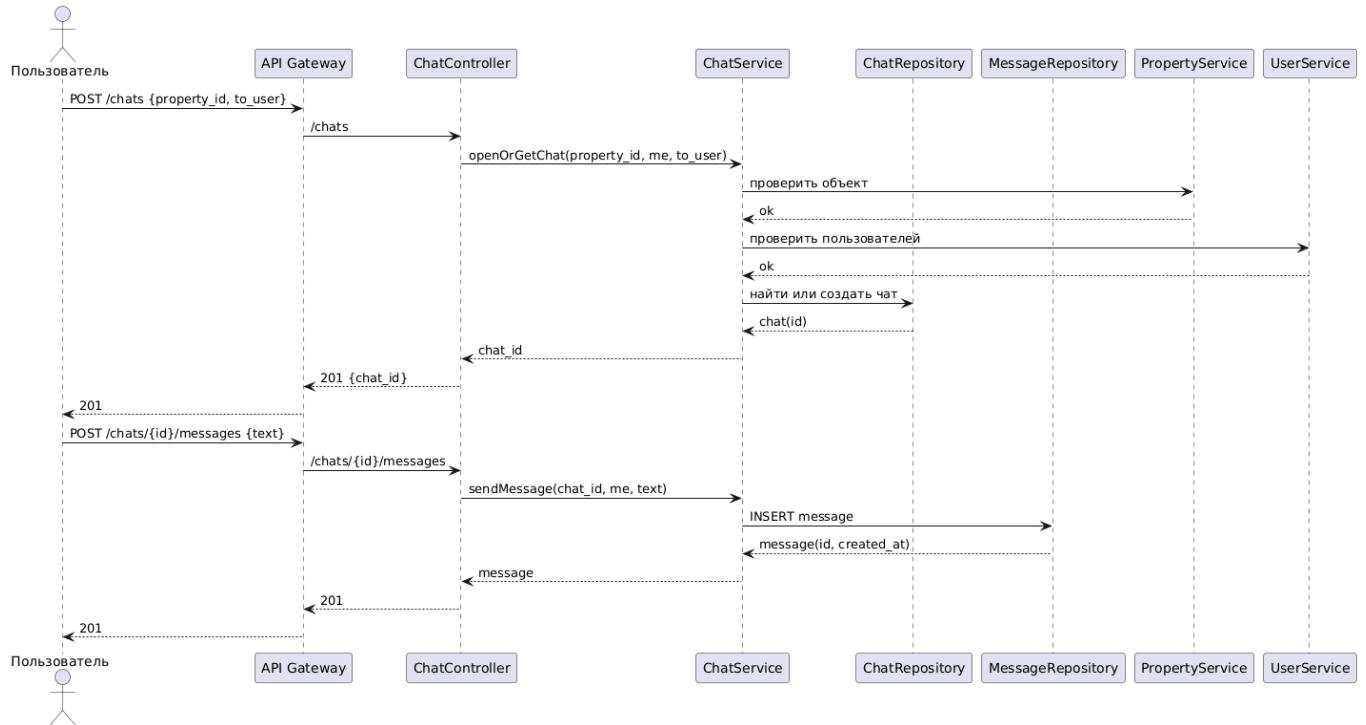


Рисунок 8 - Открыть чат и отправить сообщение

При первом запросе создается чат по объекту, затем сохраняется сообщение.

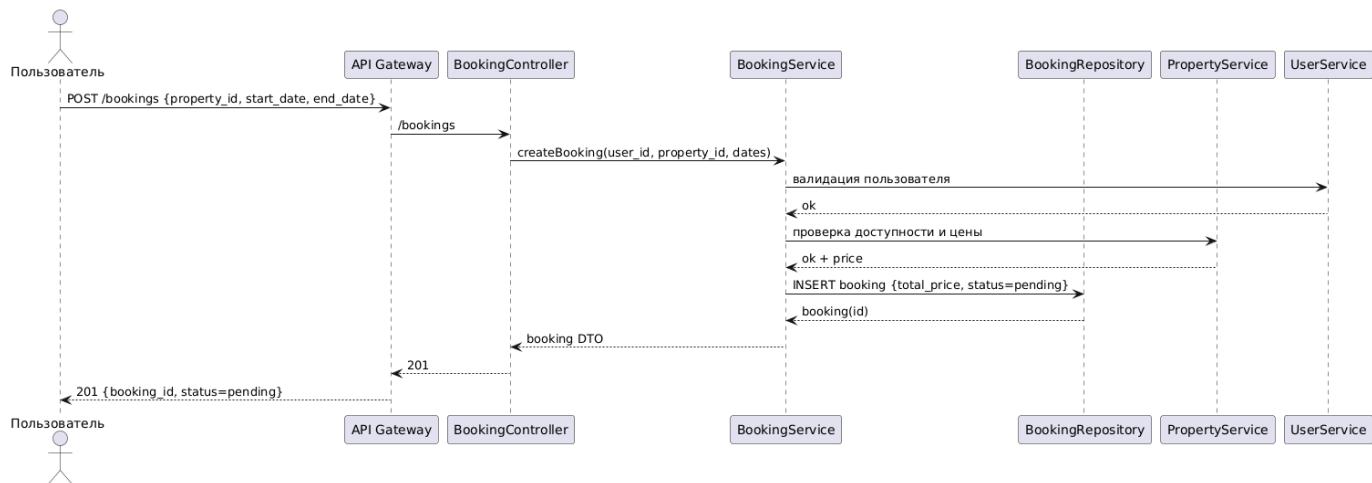


Рисунок 8 - Создать бронь

Сервис проверяет пользователя и объект, рассчитывает стоимость и создает бронь.

Вывод

В ходе выполнения домашней работы получилось спроектировать архитектуру сервиса аренды из четырёх микросервисов с API Gateway и независимыми БД. Сделаны диаграммы компонентов, ER модель и последовательности основных сценариев от регистрации и поиска объектов до чатов и бронирования.