САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО

# Отчет по домашней работе №4 по курсу “Бэкенд разработка”

Выполнили:
Бахарева М. А., К3342
Привалов К.А., К3342

Проверил:
Добряков Д.И.

Санкт-Петербург
2025 г.

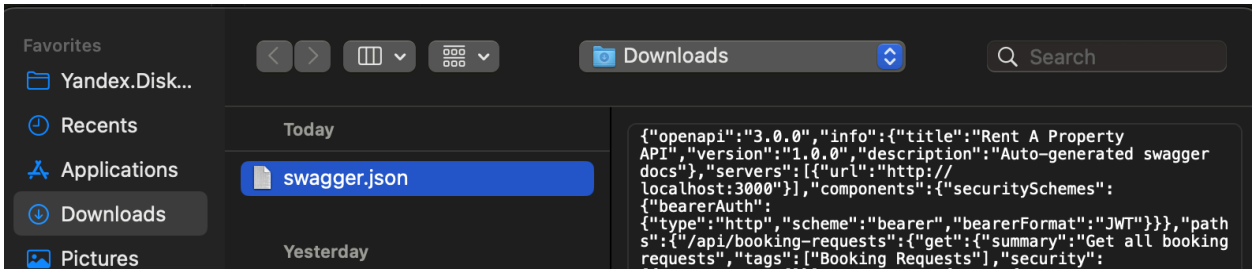
# 1. Задание:

- реализовать тестирование API средствами Postman;
- написать тесты внутри Postman.

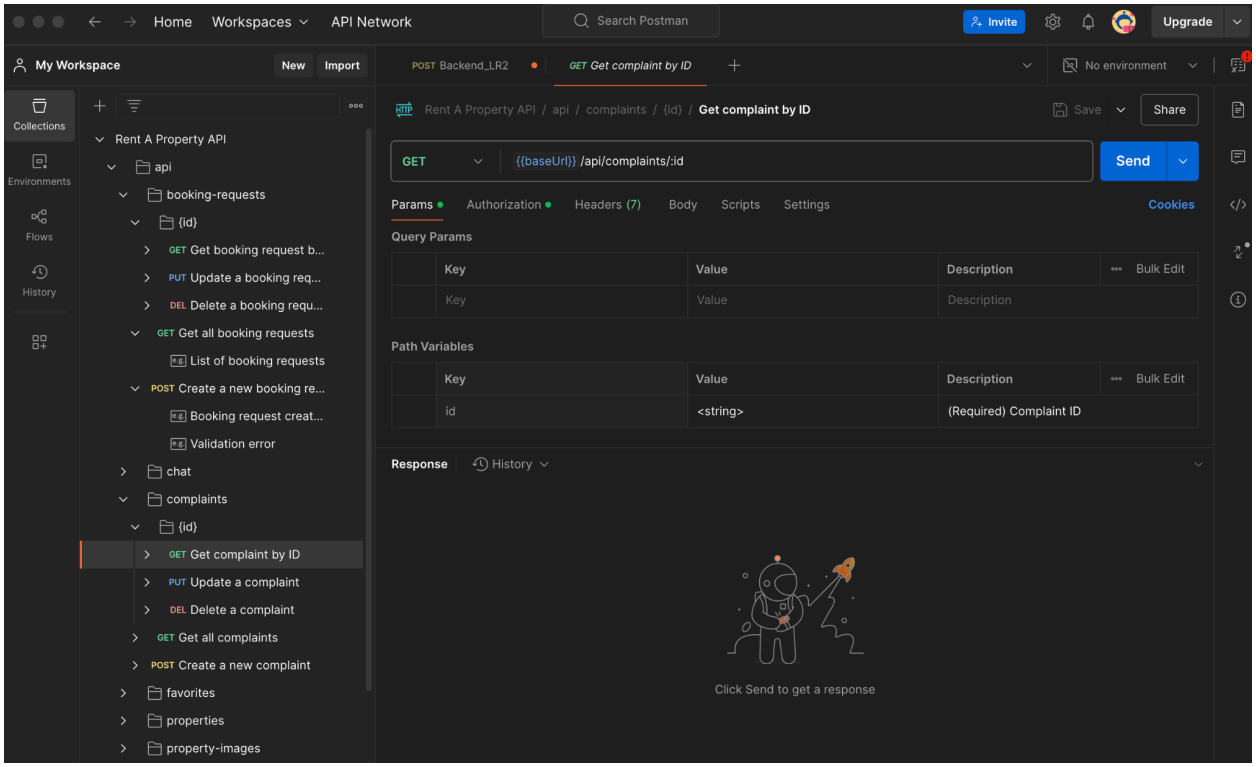
# 2. Ход работы

В предыдущем домашнем задании мы задокументировали наше приложение и энд-поинты в Postman. Вспомним, как это было и что мы получили: (*фрагменты из прошлого ДЗ выделены курсивом*)

*Создаем новую коллекцию в Postman и импортируем в нее этот файл*

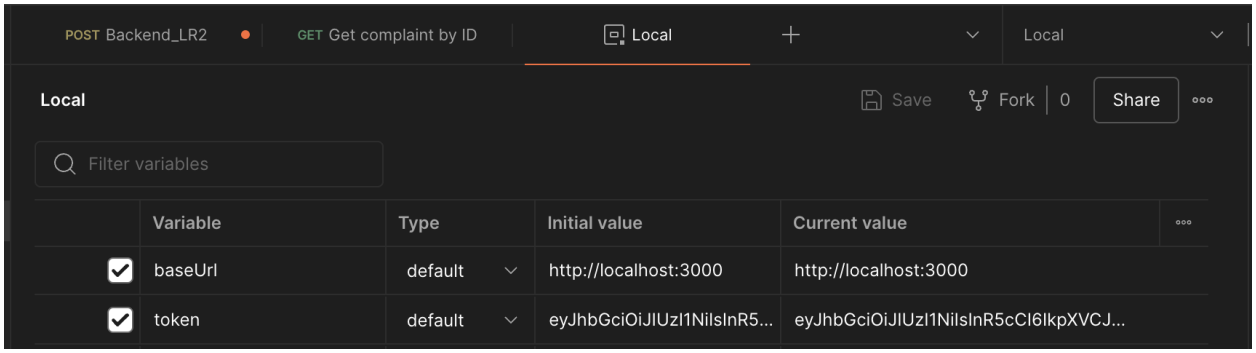



*Получаем Collection с набором энд-пойнтов*

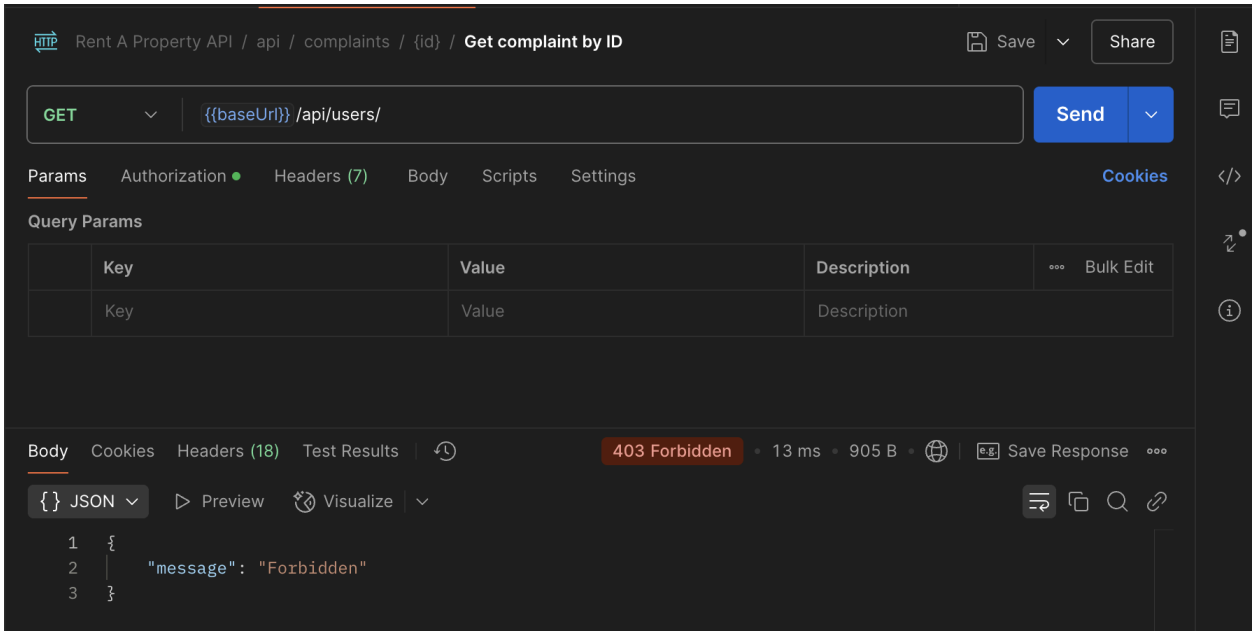



*Настроим окружение. Добавим дефолтный URL, а также пропишем токен конкретного юзера (созданного с помощью Swagger выше), чтобы проверить работоспособность*

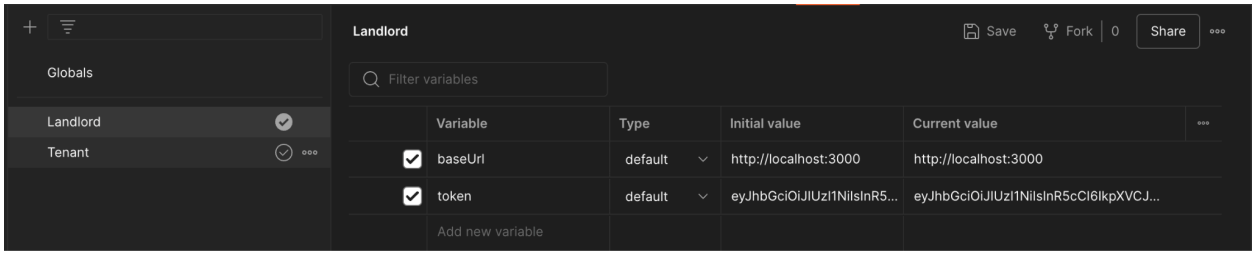
*запросов.*

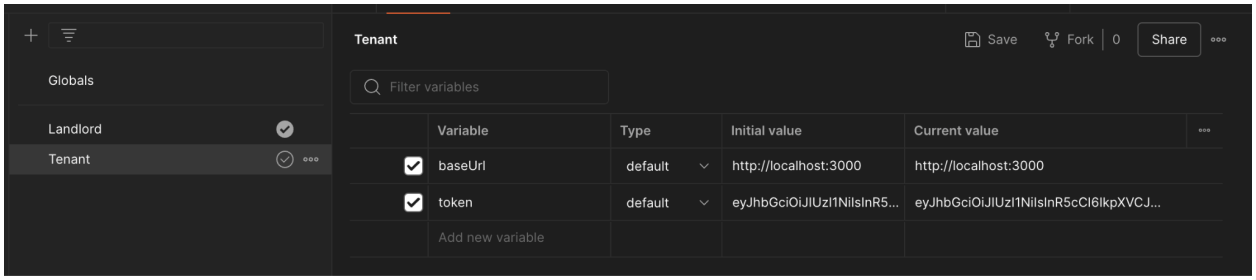



*Проверим работоспособность. Запрос прошел с корректным респонсом в виде запрета видеть всех юзеров системы, так как пользователь не обладает полным набором прав (у него роль - tenant)*



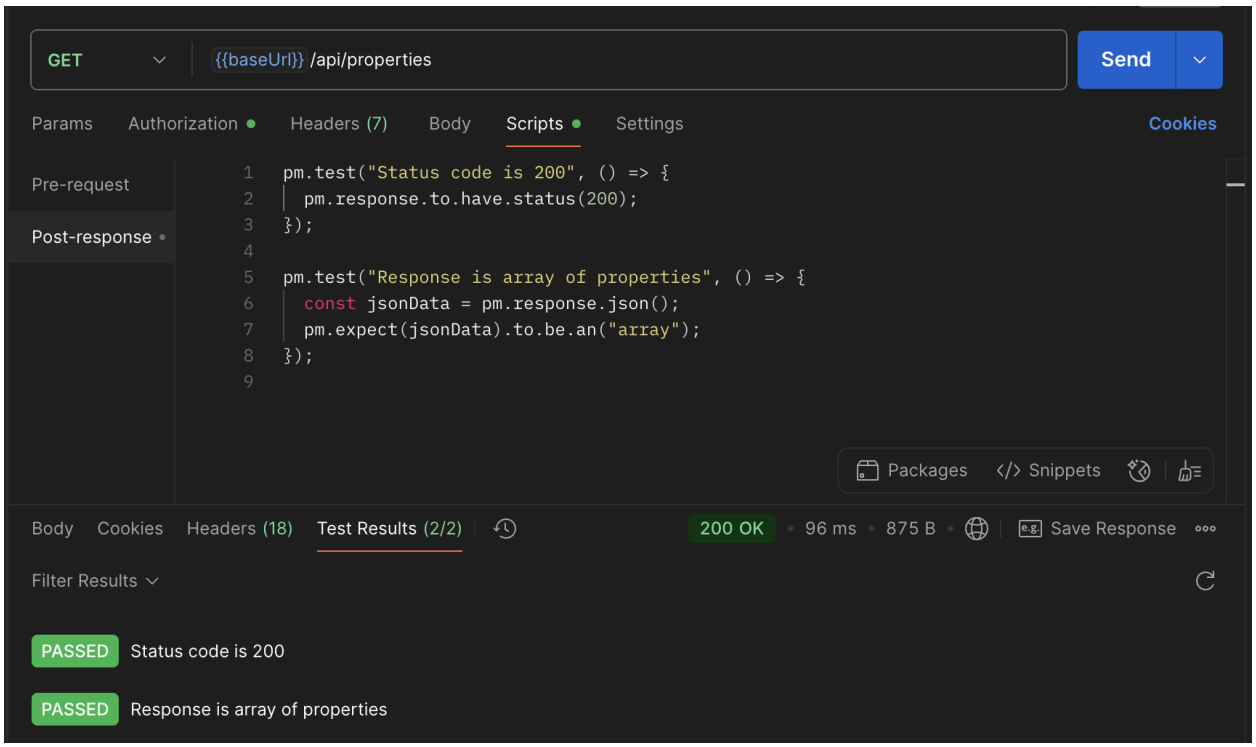

Используем окружения, которые мы создали ранее. Немного модифицируем их. Так как у нас две роли: landlord/tenant, создадим два окружения с соответствующими токенами:



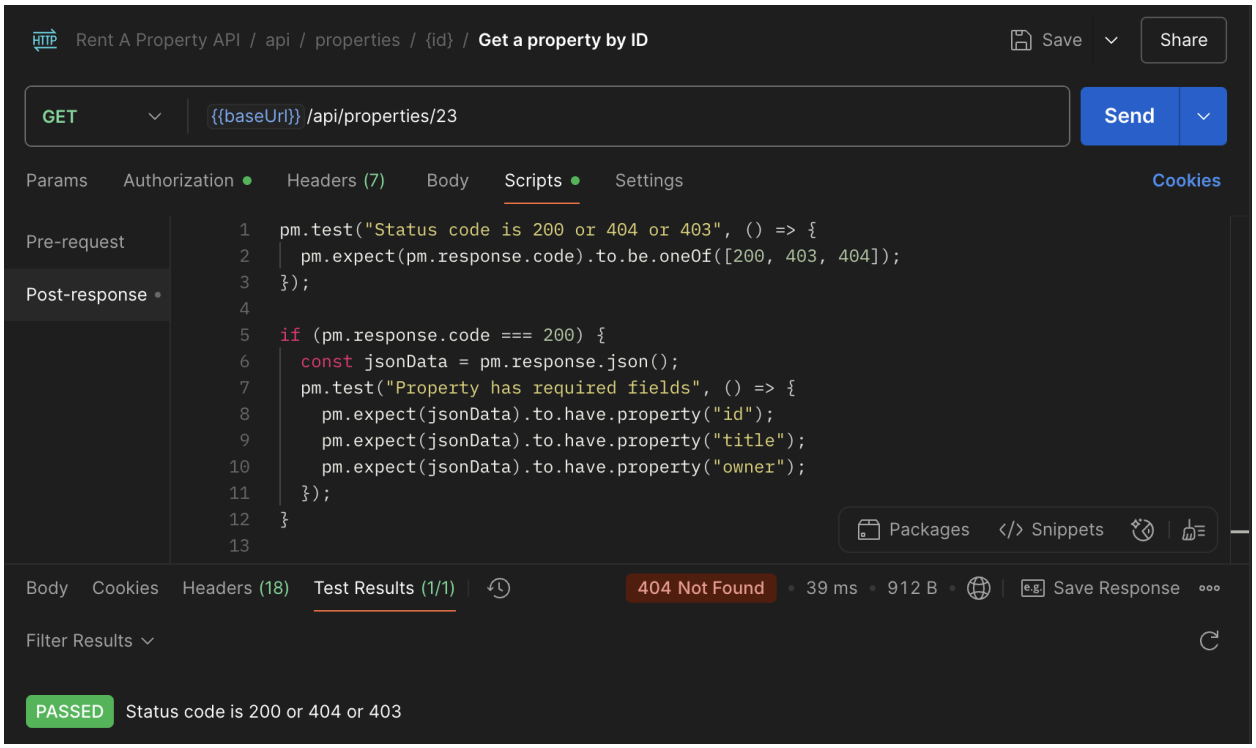



Напишем тесты для проверки работы энд-пойнтов для Properties. Учтем в них работу с ролевой моделью.

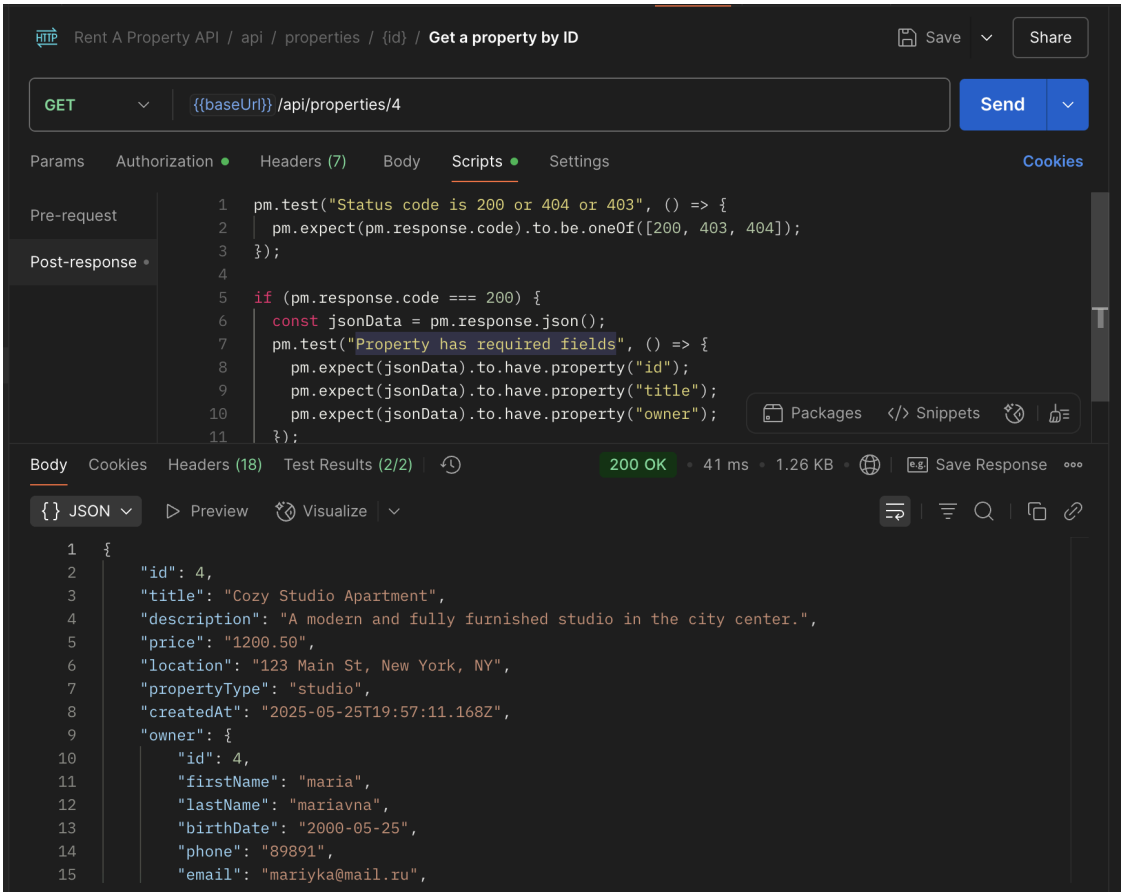
Обеим ролям доступен просмотр всей недвижимости.

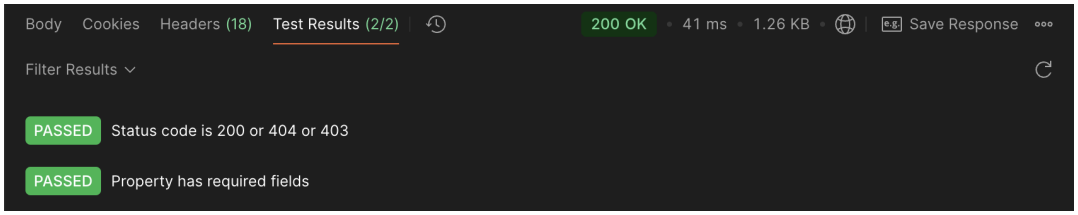



Владелец может просматривать по id только свою недвижимость, арендатор все.

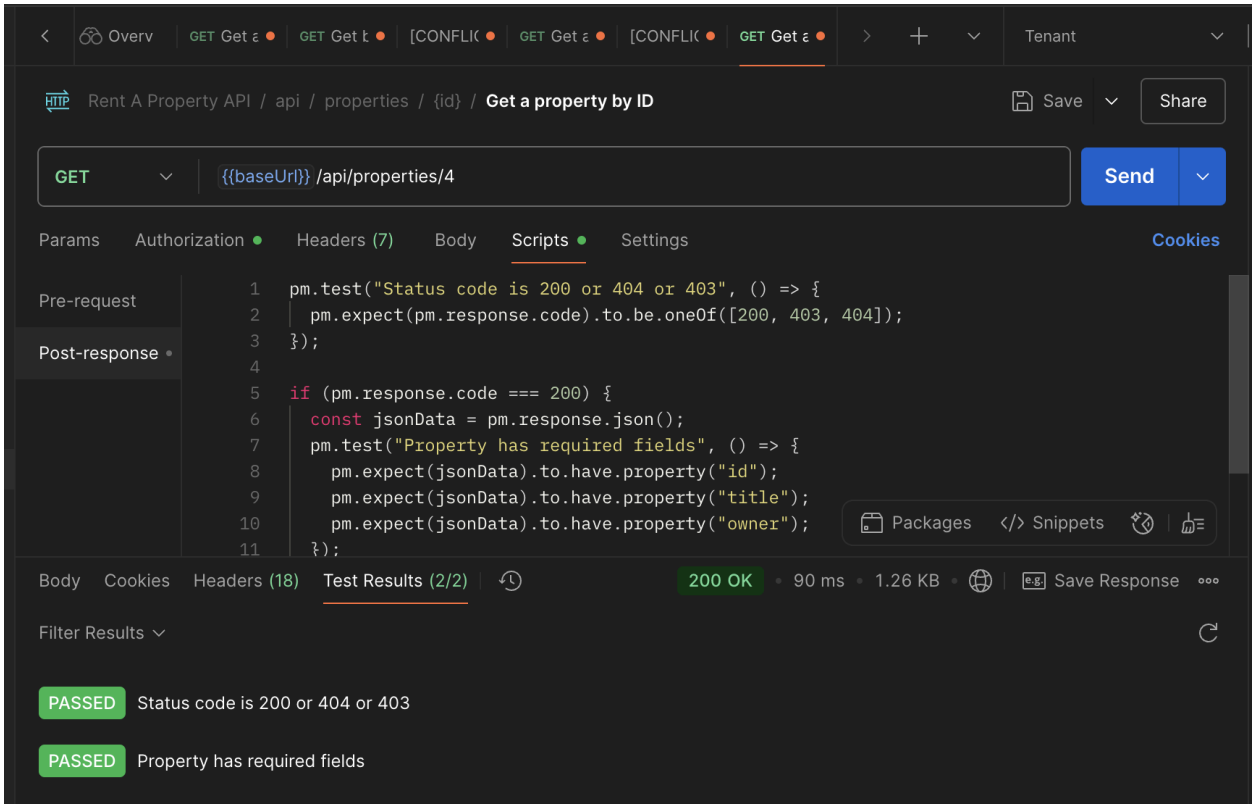



Создадим тестово нашему арендодателю объект недвижимости. Ему оно доступно. Тесты также прошли успешно.



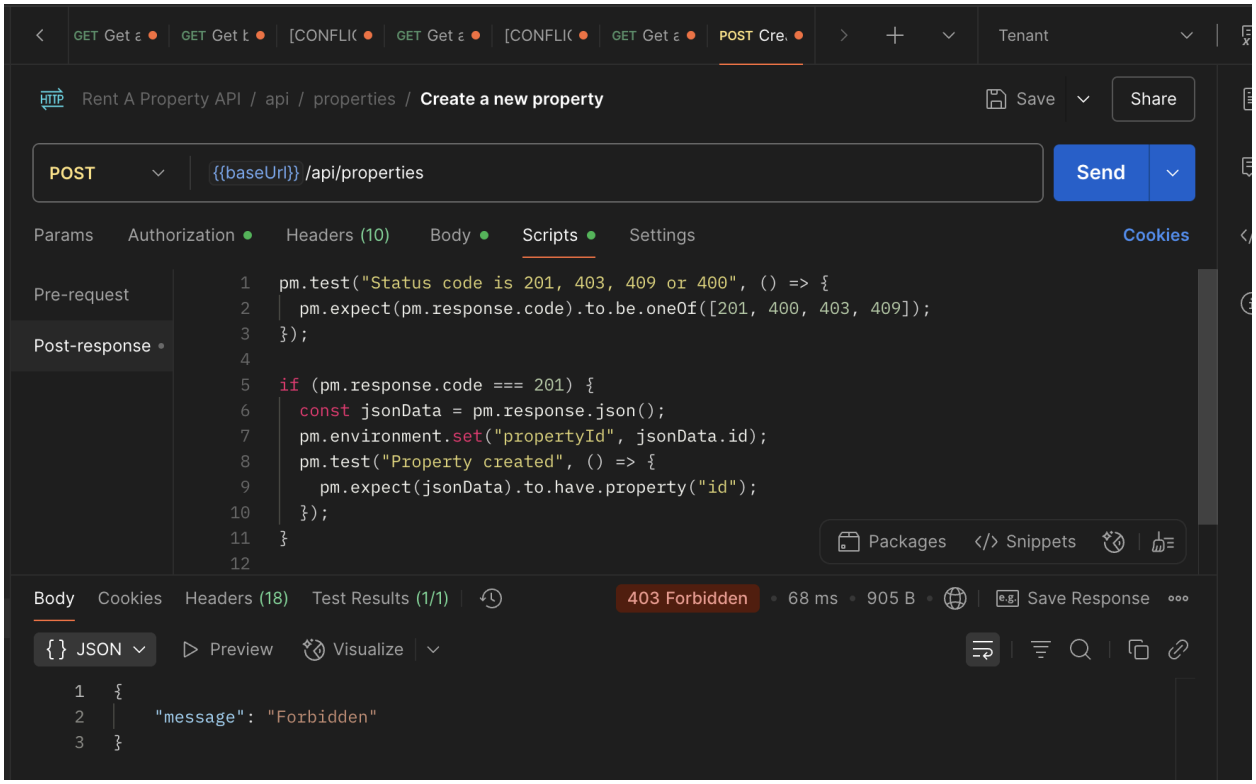

Арендатору также доступна данная недвижимость.



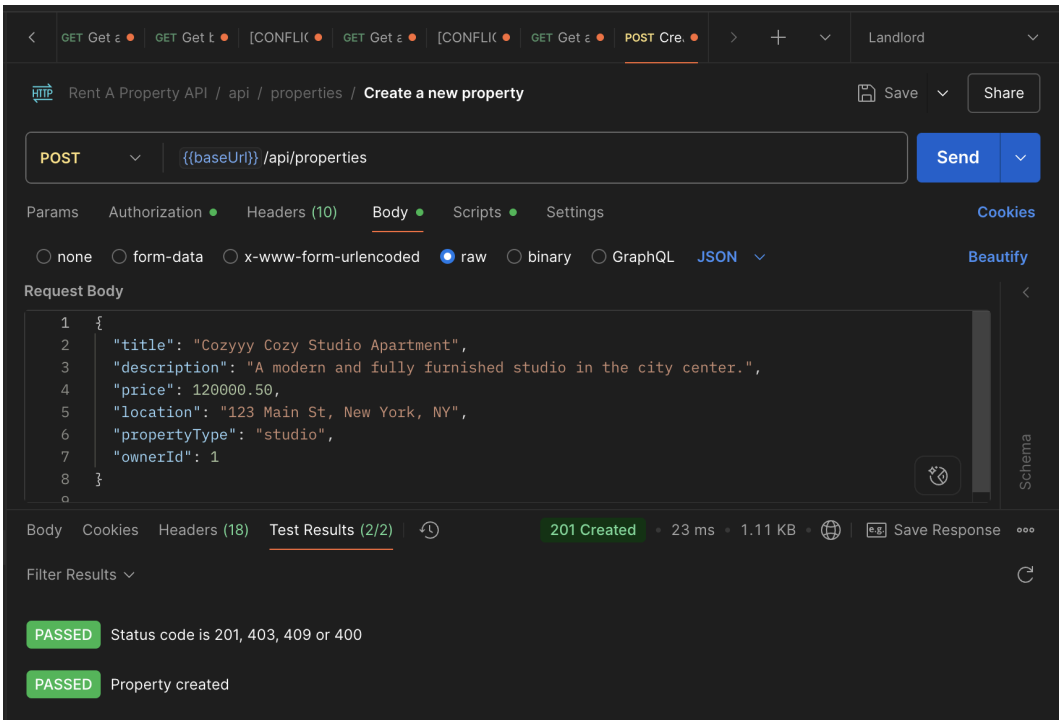

Теперь арендатор пробуется создать недвижимость. Передадим в body следующий json

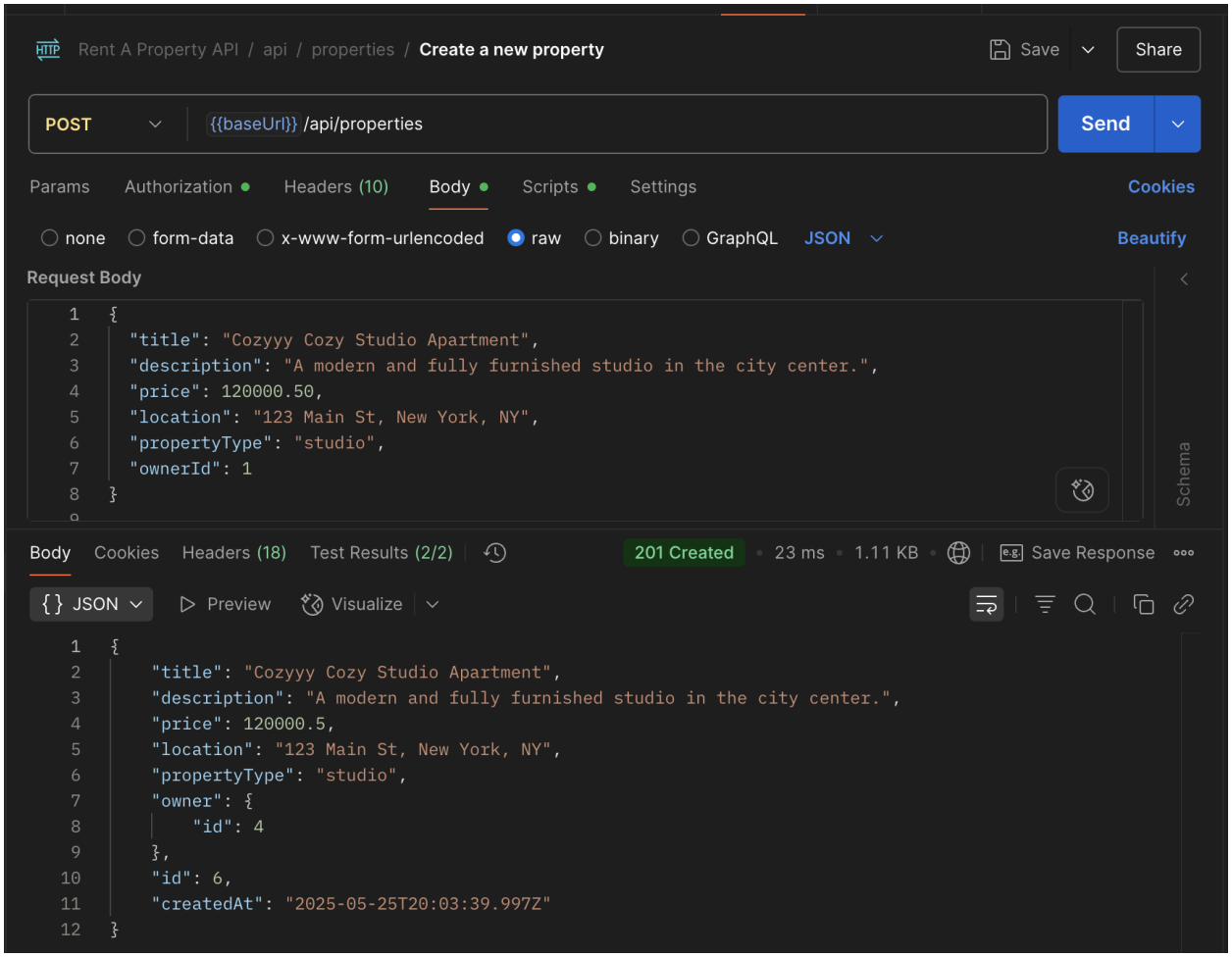
```json
{
  "title": "Test Apartment",
  "description": "Bright and clean",
  "address": "123 Test St",
  "price": 1500,
  "type": "apartment",
  "ownerId": 1
}
```

Получаем Forbidden. Так как у арендатора не может быть прав на создание недвижимости.




Пройдем аналогичный тест, но на окружении владельца недвижимости.






**В итоге можно выделить основные принципы построения тест-кейсов в Postman:**

1. Разделение по типам запросов

Каждому HTTP-методу соответствует свой набор тестов:

- GET: проверка получения данных, проверка статуса 200, соответствие структуры.
- POST: создание ресурса, проверка 201 Created, валидация тела.
- PUT: обновление ресурса, проверка 200, контроль доступа.
- DELETE: удаление, проверка 204, повторная попытка — 404.

2. Проверка ролей и доступа

Проверяется поведение API при запросе с разными токенами (landlord, tenant, неавторизованный пользователь). В тестах симулируются попытки доступа к чужим данным → ожидается 403 Forbidden.

# 3. Вывод

В ходе выполнения лабораторной работы были в качестве примера реализованы и протестированы API-эндпоинты для управления объектами недвижимости (Properties) . Были учтены особенности ролевой модели (LANDLORD, TENANT), а также реализованы проверки прав доступа, коды ошибок и поведение API при разных сценариях.