Министерство науки и высшего образования Российской Федерации
федеральное государственное автономное образовательное учреждение высшего образования
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Отчет

по лабораторной работе № 3 «Реализация серверной части на django rest. Документирование API»

по дисциплине «**Web-программирование**»

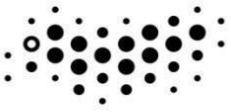
Автор: Власов М. И.

Факультет: ИКТ

Группа: K33402

Преподаватель: Говоров А. И.

Дата: 11.12.21

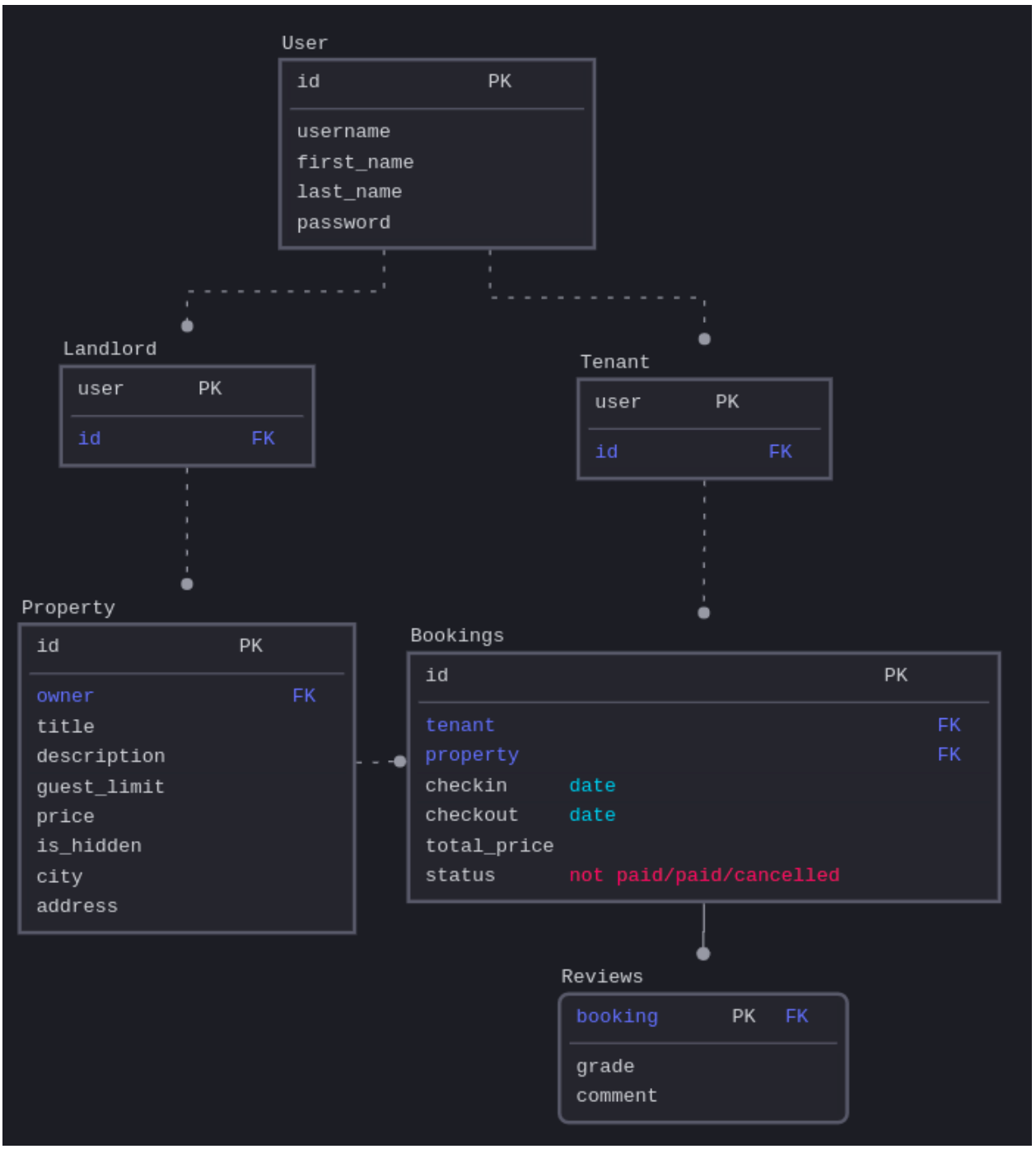


УНИВЕРСИТЕТ ИТМО

Санкт-Петербург, 2021

Цель работы: овладеть практическими навыками реализации серверной части (backend) приложений средствами Django REST framework.

Модель базы данных



Реализация API

Мы создали API для всех нашим таблиц. Рассмотрим для примера таблицу *User*

Для начала получим список пользователей (*ListView*)

```
GET /user/list/
```

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "id": 1,
        "username": "admin",
        "first_name": "",
        "last_name": ""
    },
    {
        "id": 3,
        "username": "passtest",
        "first_name": "Pass",
        "last_name": "Test"
    },
    {
        "id": 4,
        "username": "tenant",
        "first_name": "Te",
        "last_name": "Nant"
    },
    {
        "id": 5,
        "username": "nate",
        "first_name": "Te",
        "last_name": "Na"
    },
    {
        "id": 6,
        "username": "onemore",
        "first_name": "One",
        "last_name": "More"
    },
    {
        "id": 7,
        "username": "todelete",
```

Теперь посмотрим информацию о конкретном пользователе (*RetrieveView*)

```
GET /user/6
```

```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 6,
    "username": "onemore",
    "first_name": "One",
    "last_name": "More"
}
```

Так выглядит окно для создания нового пользователя (*CreateView*)

```
GET /user/create/
```

```
HTTP 405 Method Not Allowed
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "detail": "Method \"GET\" not allowed."
}
```

Raw data | HTML form

Username

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

First name

Last name

Password

POST

Давайте обновим пользователя с номером 3 (*UpdateView*)

```
GET /user/update/3
```

```
HTTP 200 OK
Allow: GET, PUT, PATCH, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 3,
    "username": "passtest",
    "first_name": "Pass",
    "last_name": "Test",
    "password": "pbkdf2_sha256$320000$W44AOrc5gzb8jF7oiDoOvX$jrvSlkcbE+LPRT/UTFDDOhg7RcJu5KSr2XuoowV0lIs="
}
```

Raw data | HTML form

Username: passtest
Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

First name: Pass

Last name: Test

Password: newpass

PUT

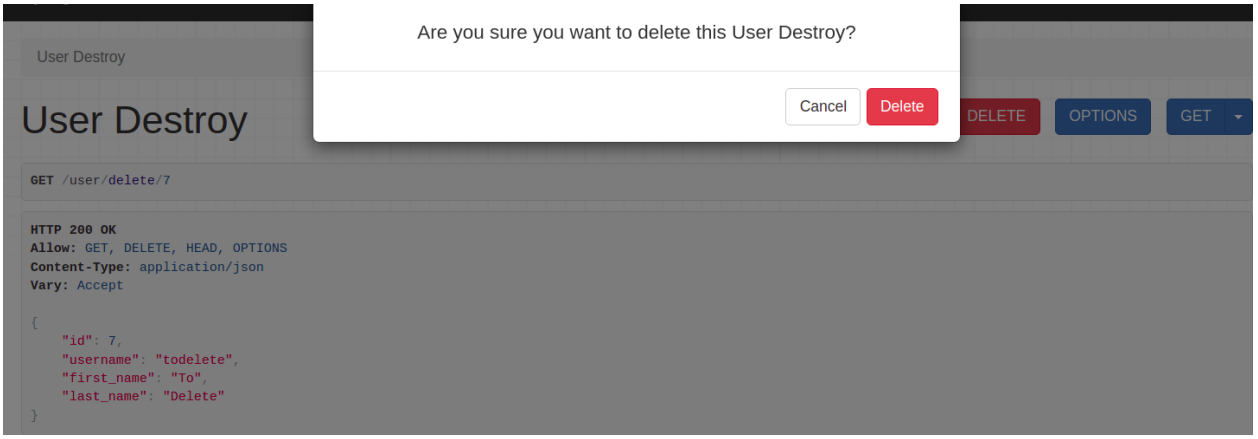
Посмотрим на результат (видим, что изменился шифр пароля)

```
PUT /user/update/3
```

```
HTTP 200 OK
Allow: GET, PUT, PATCH, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "id": 3,
    "username": "passtest",
    "first_name": "Pass",
    "last_name": "Test",
    "password": "pbkdf2_sha256$320000$ub2HsC1LafC9urFSECdqjb$St3ZxmsKjl+kmDNSWU/YVT0fNFsSA9hLxDMa55h/RkQ="
}
```

Удалим пользователя с ником todelete (*DeleteView*)



Убедимся, что пользователь успешно удалён

```
GET /user/7
```

```
HTTP 404 Not Found
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "detail": "Not found."
}
```

Перейдём к таблице *Property*. Обратим внимание, что некоторые поля имеют вложенность больше единицы

```
GET /property/list/
```


```
HTTP 200 OK
Allow: GET, HEAD, OPTIONS
Content-Type: application/json
Vary: Accept

[
    {
        "id": 1,
        "owner": {
            "user": {
                "id": 3,
                "username": "passtest",
                "first_name": "Pass",
                "last_name": "Test"
            }
        },
        "city": "Saint Petersburg",
        "address": "Belorusskaya street, 6",
        "title": "Dorm",
        "description": "No",
        "guest_limit": 3,
        "price": 10,
        "is_hidden": false
    },
    {
        "id": 2,
        "owner": {
            "user": {
                "id": 1,
                "username": "admin",
                "first_name": "Admin",
                "last_name": "Adminov"
            }
        },
        "city": "Moscow",
        "address": "Lenina street, 1",
        "title": "Apartment",
        "description": "Very good",
        "guest_limit": 5,
        "price": 111,
        "is_hidden": false
    }
```

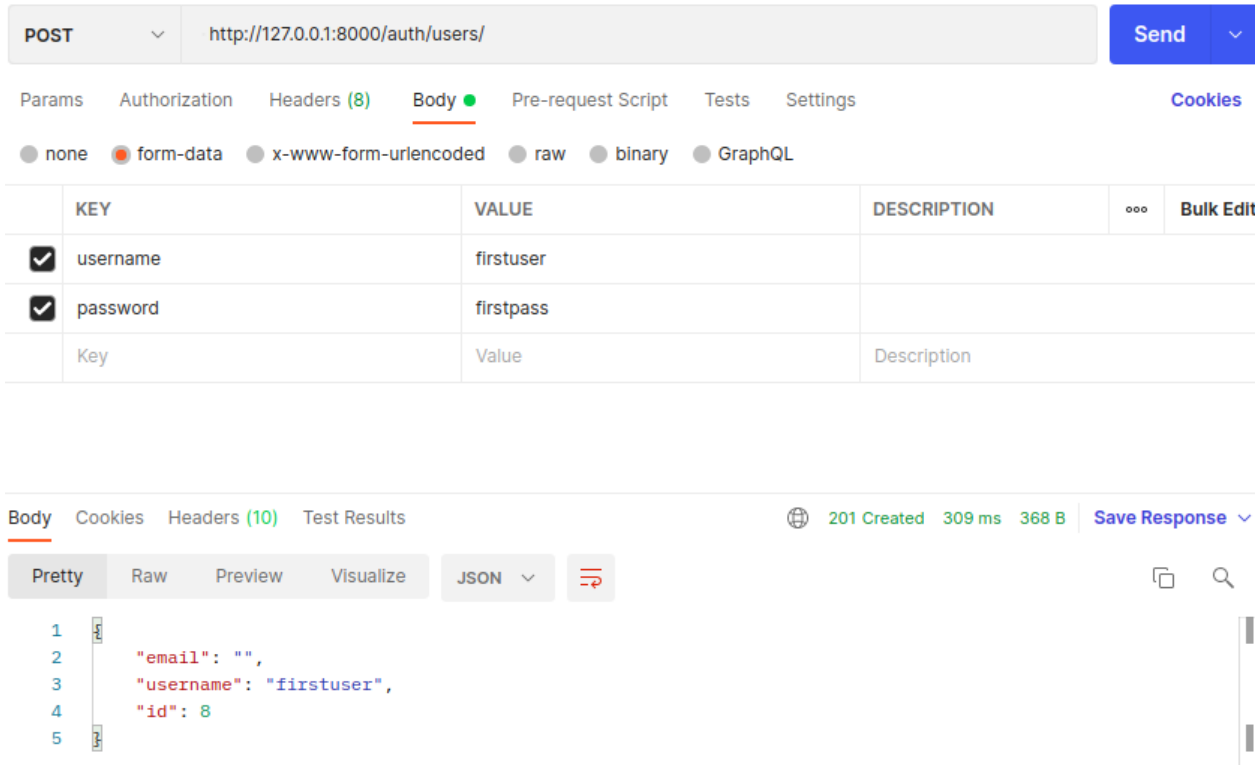
Забронируем жильё

Убедимся, что появилась соответствующая запись. Заметим, что автоматически была подсчитана общая стоимость на основе даты заезда, выезда и цены за ночь

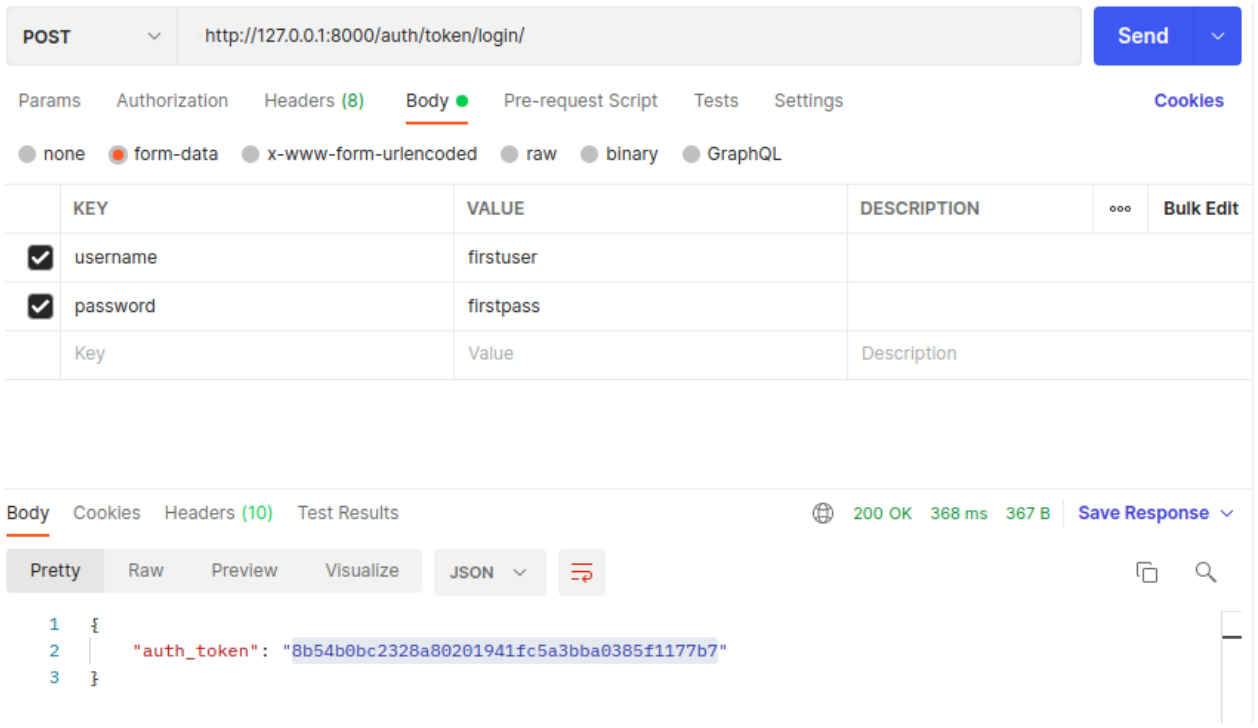

```
{
    "tenant": {
        "user": {
            "id": 6,
            "username": "onemore",
            "first_name": "One",
            "last_name": "More"
        }
    },
    "property": {
        "id": 2,
        "owner": {
            "user": {
                "id": 1,
                "username": "admin",
                "first_name": "Admin",
                "last_name": "Adminov"
            }
        },
        "city": "Moscow",
        "address": "Lenina street, 1",
        "title": "Apartment",
        "description": "Very good",
        "guest_limit": 5,
        "price": 111,
        "is_hidden": false
    },
    "checkin": "2021-12-30",
    "checkout": "2022-01-03",
    "total_price": 444,
    "status": 1
}
```

Подключение регистрации и авторизации по токенам

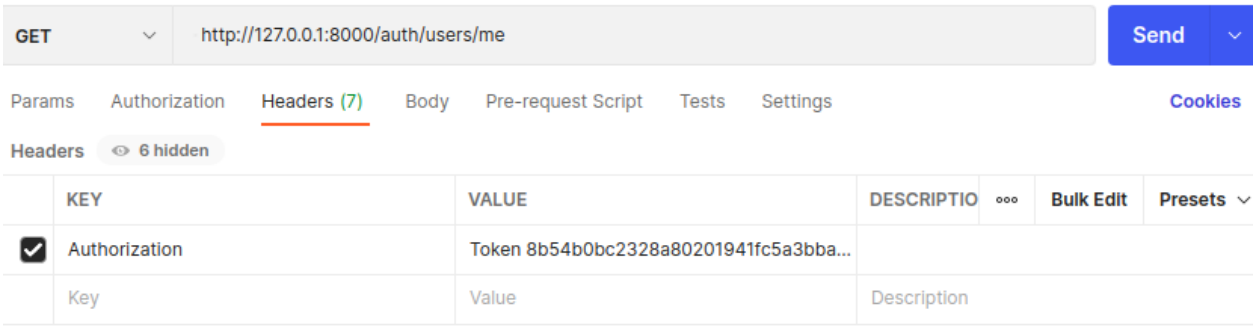
Создадим нового пользователя

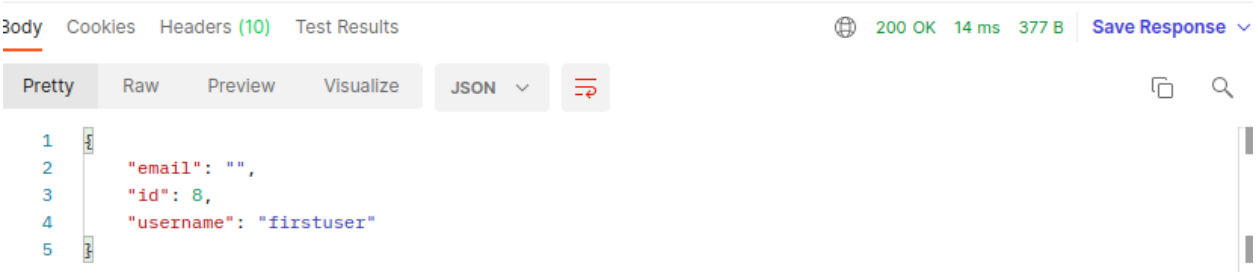


Авторизуемся и получим токен



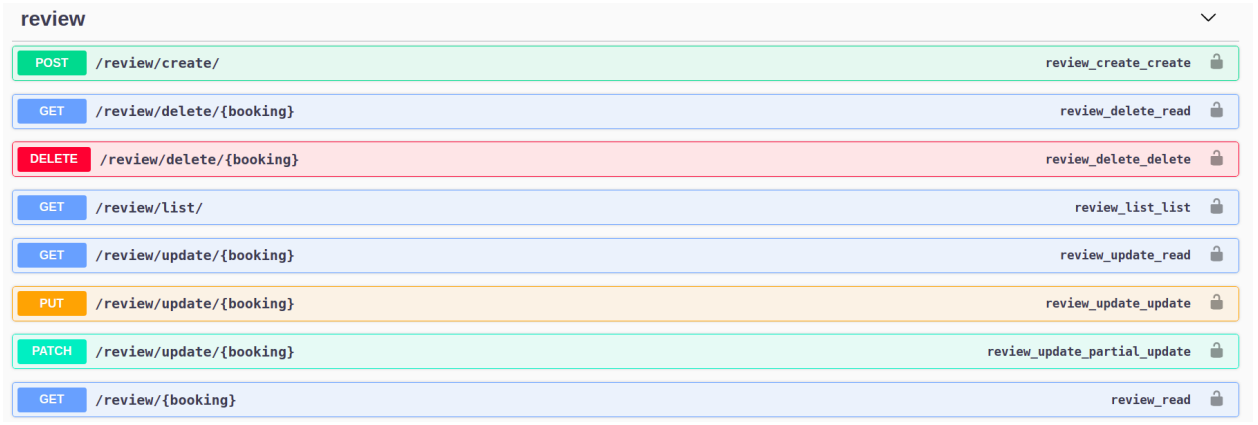
Получим информацию о текущем пользователе с помощью токена



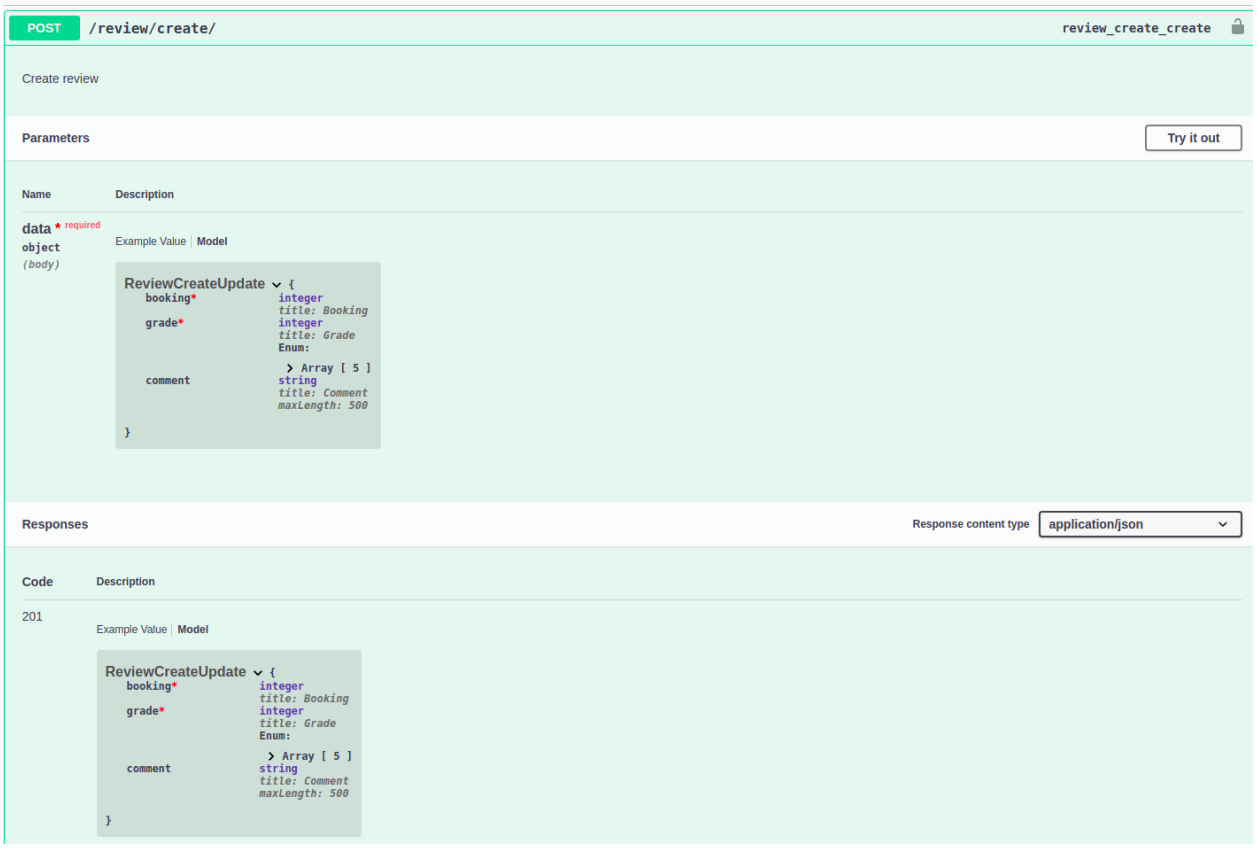


Документация

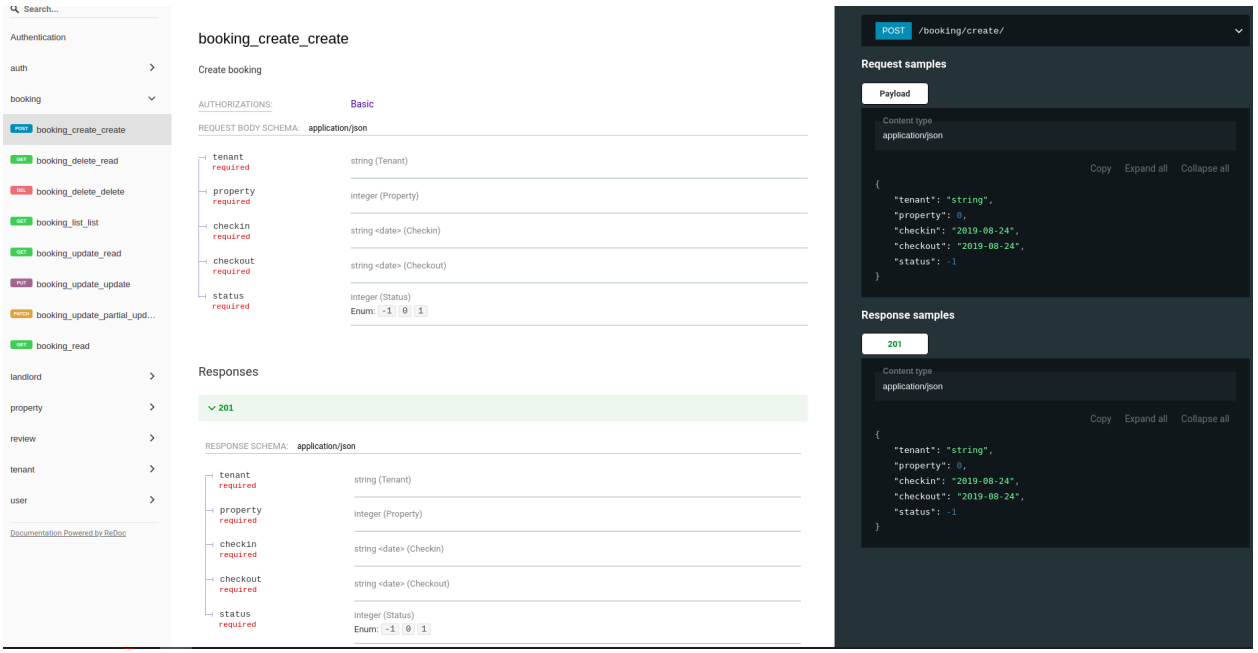
В Swagger документация выглядит следующим образом



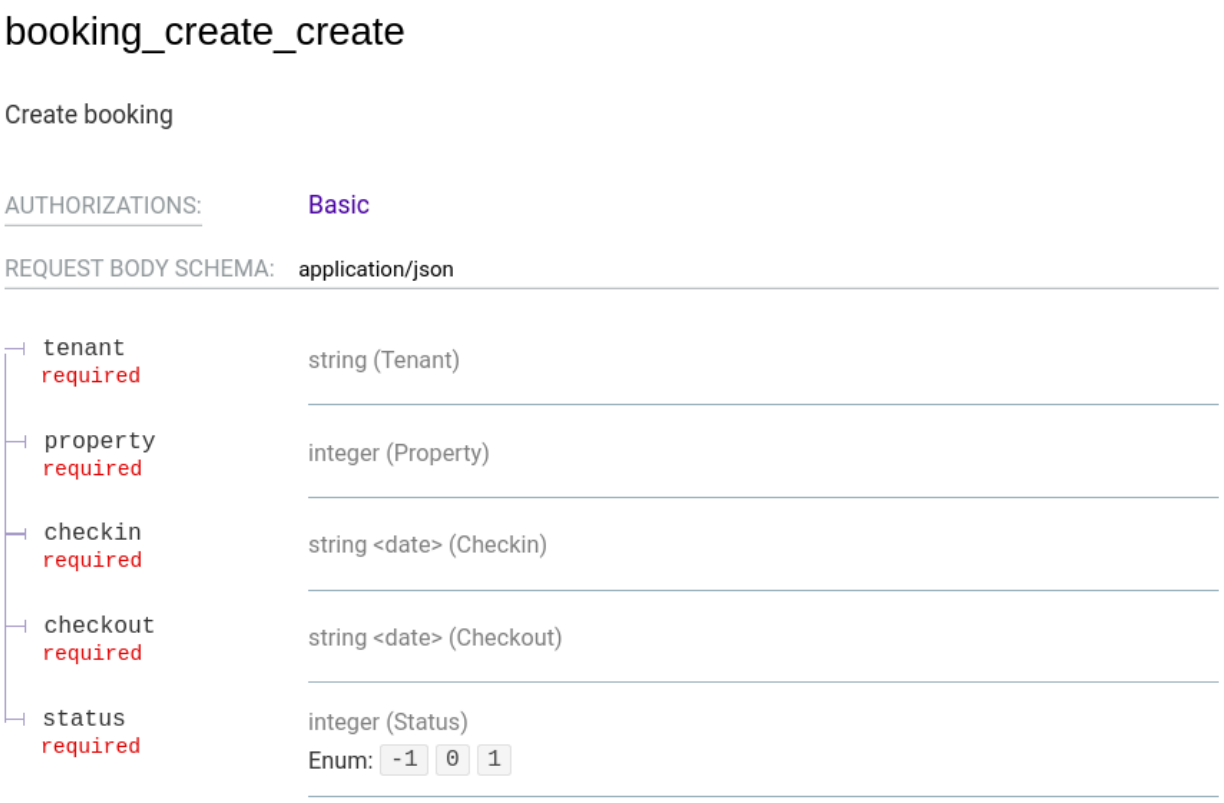
Рассмотрим метод создания отзыва подробнее



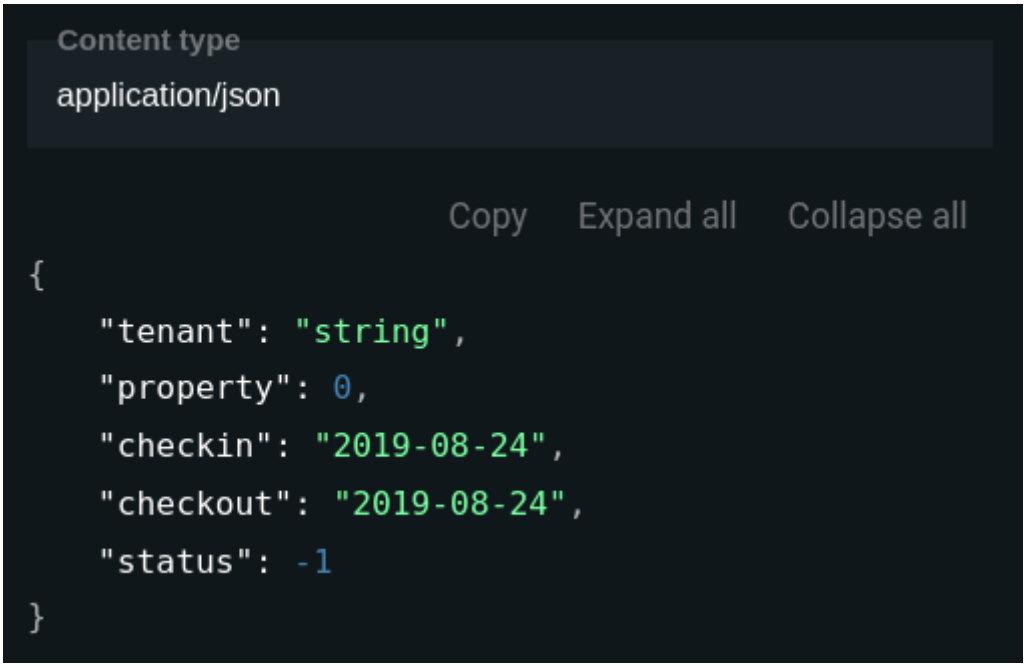
Перейдём к redoc



Рассмотрим параметры создания бронирования подробнее



Эти же параметры в формате *json*



Вывод: в ходе лабораторной работы мы овладели практическими навыками реализации серверной части (backend) приложений средствами Django REST framework.