МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»

(БГТУ им. В.Г.Шухова)

**Лабораторная работа №5**

дисциплина «Технологии web-программирования»

тема: «REST API»

Выполнил: студент группы ВТ-41

Ковалёв И. Д.

Проверил:

Картамышев С. В.

Белгород 2020

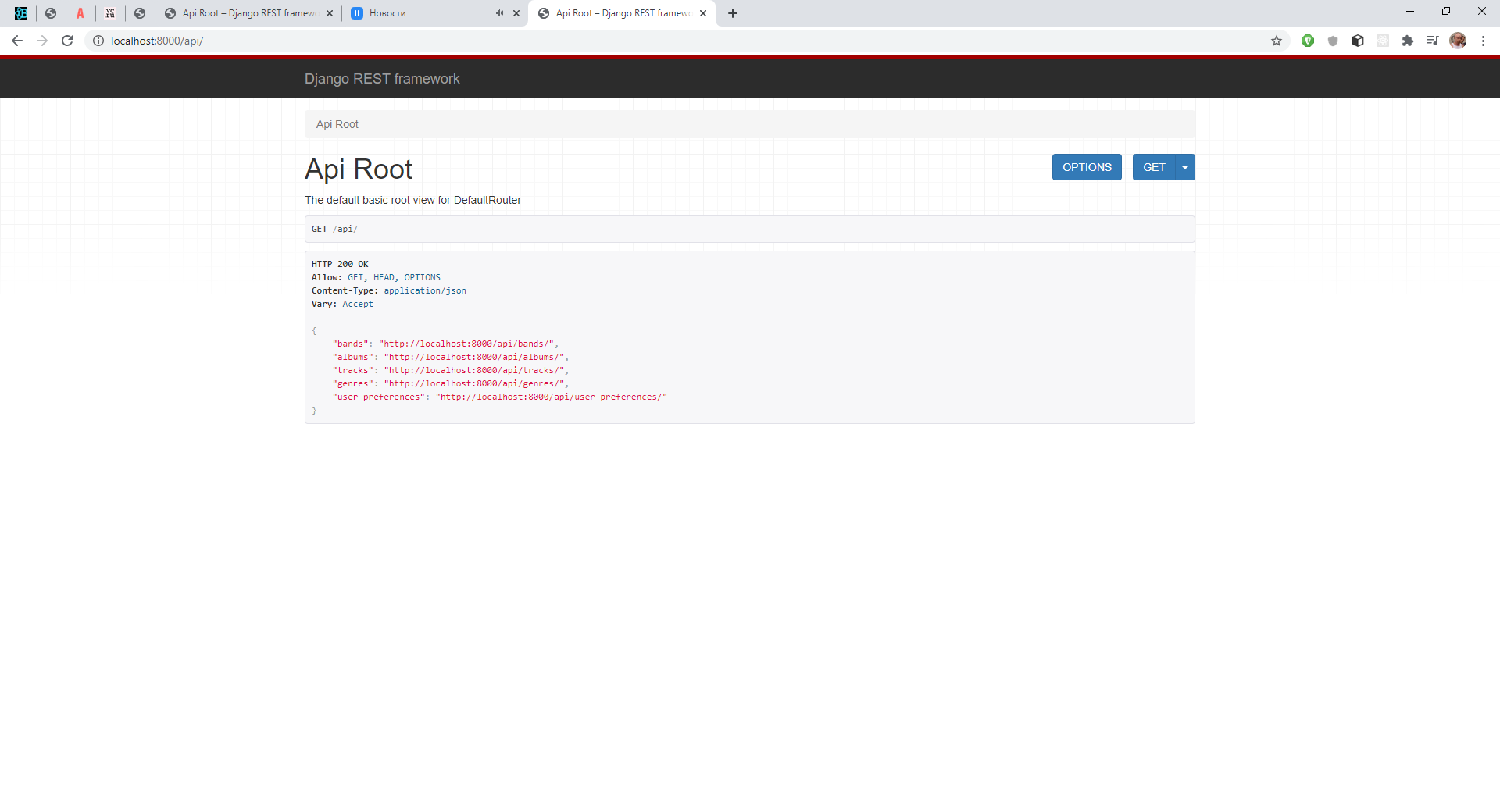
**Цель работы**: изучить основы разработки API для web-приложений. Разработать REST API для своего проекта.

**Задание к лабораторной работе:**

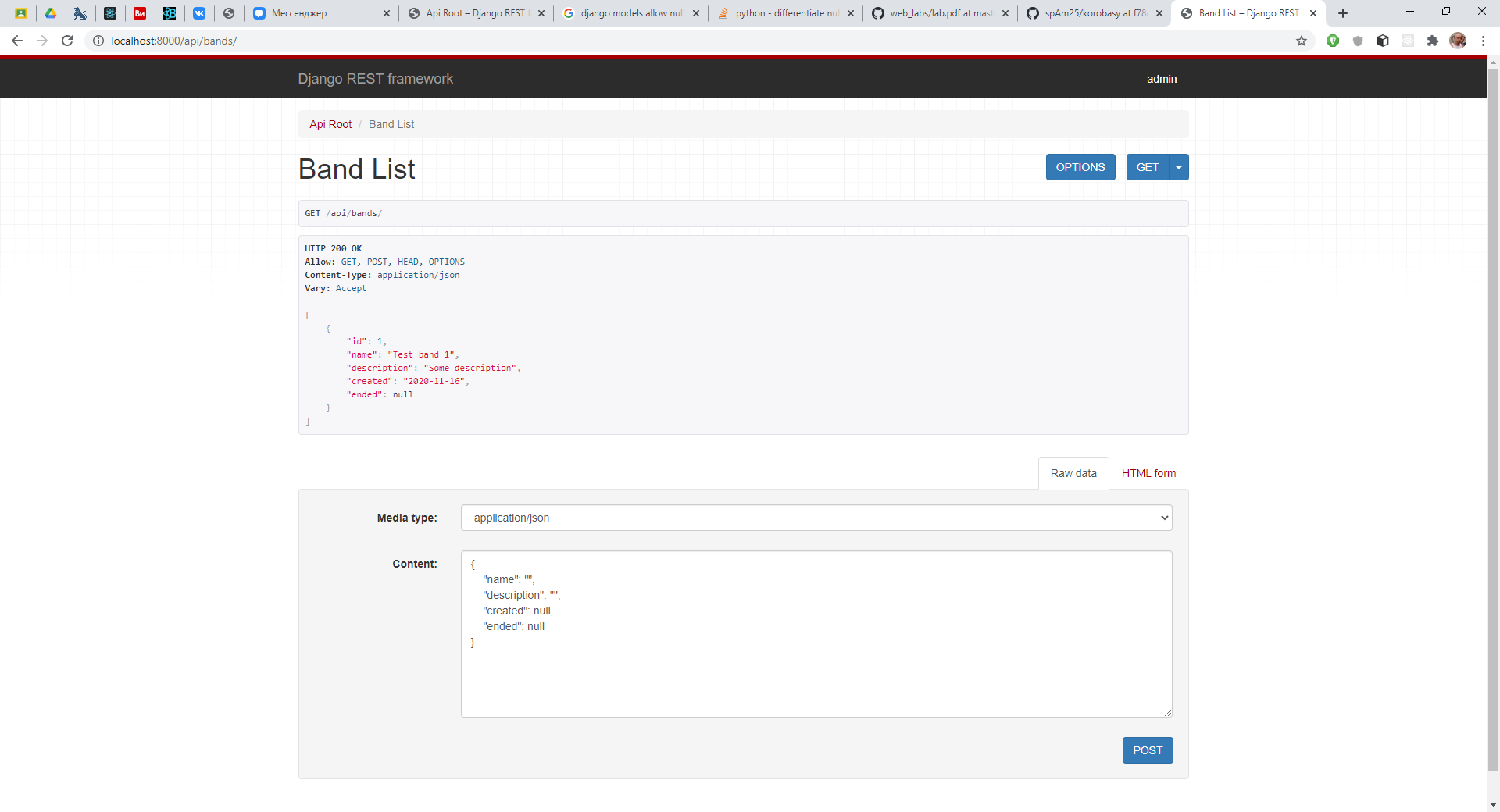
1. Изучить структуру формата представления данных JSON.
2. Изучить типы запросов к API: HEAD, GET, POST, PUT, DELETE.
3. Спроектировать и реализовать собственное REST API (Получение, создание, изменение и удаление каких-либо объектов).
4. В отчёт необходимо предоставить документацию к использованию методов. (Либо словесным описание, либо через Swagger)

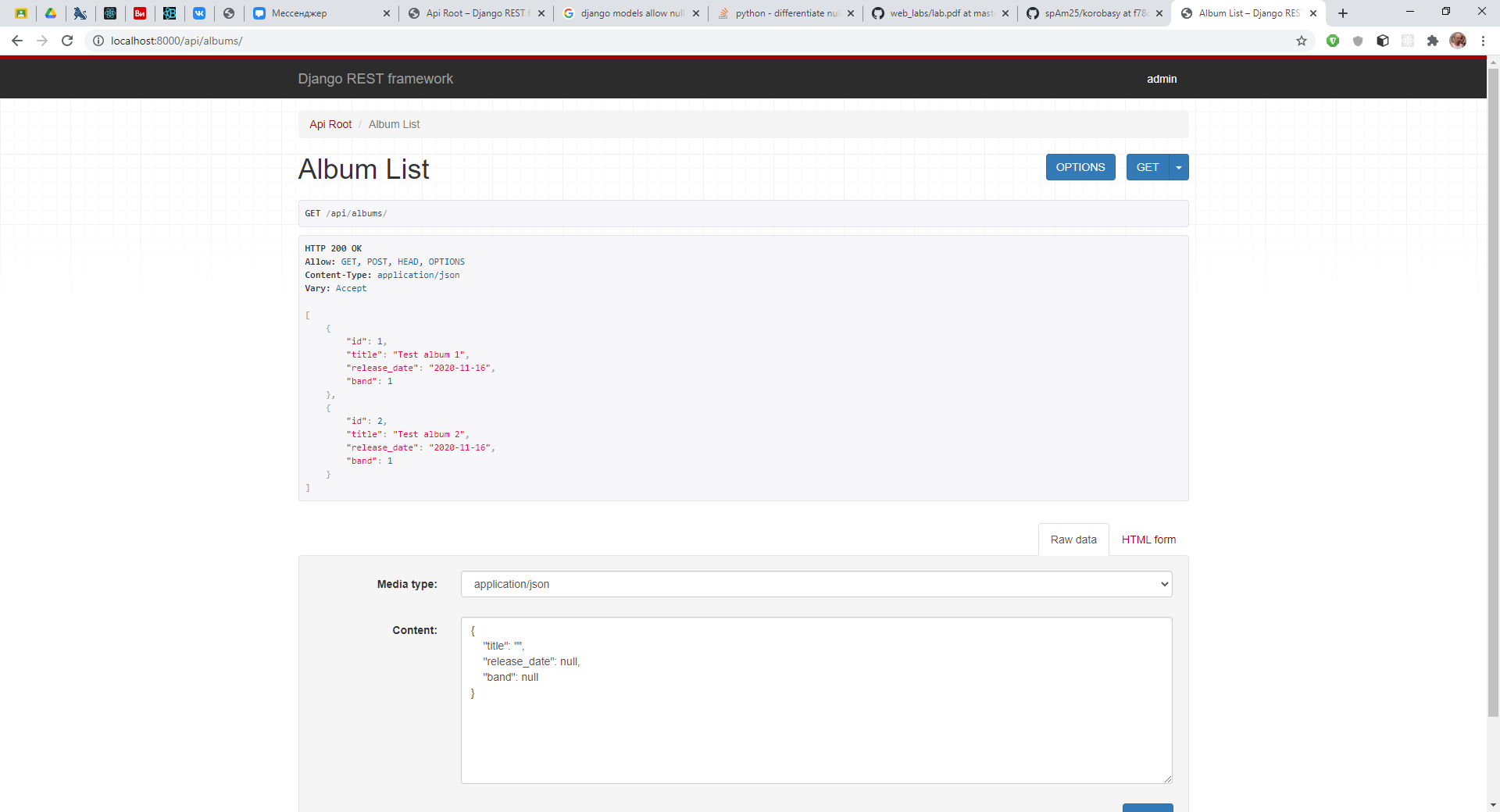
**Ход выполнения работы:**

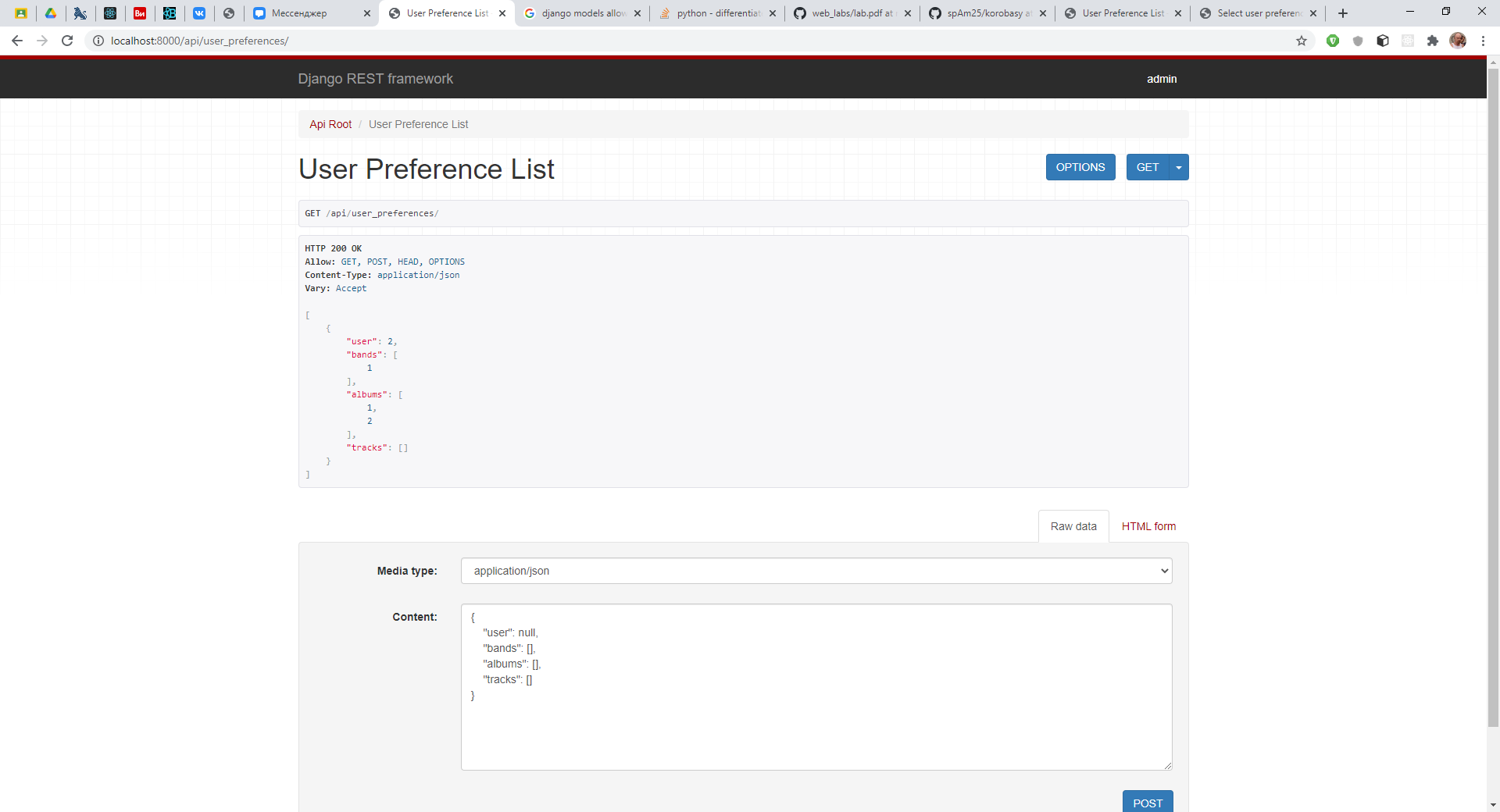
Развернем сервер и проверим функционирование API. Были созданы такие URL:

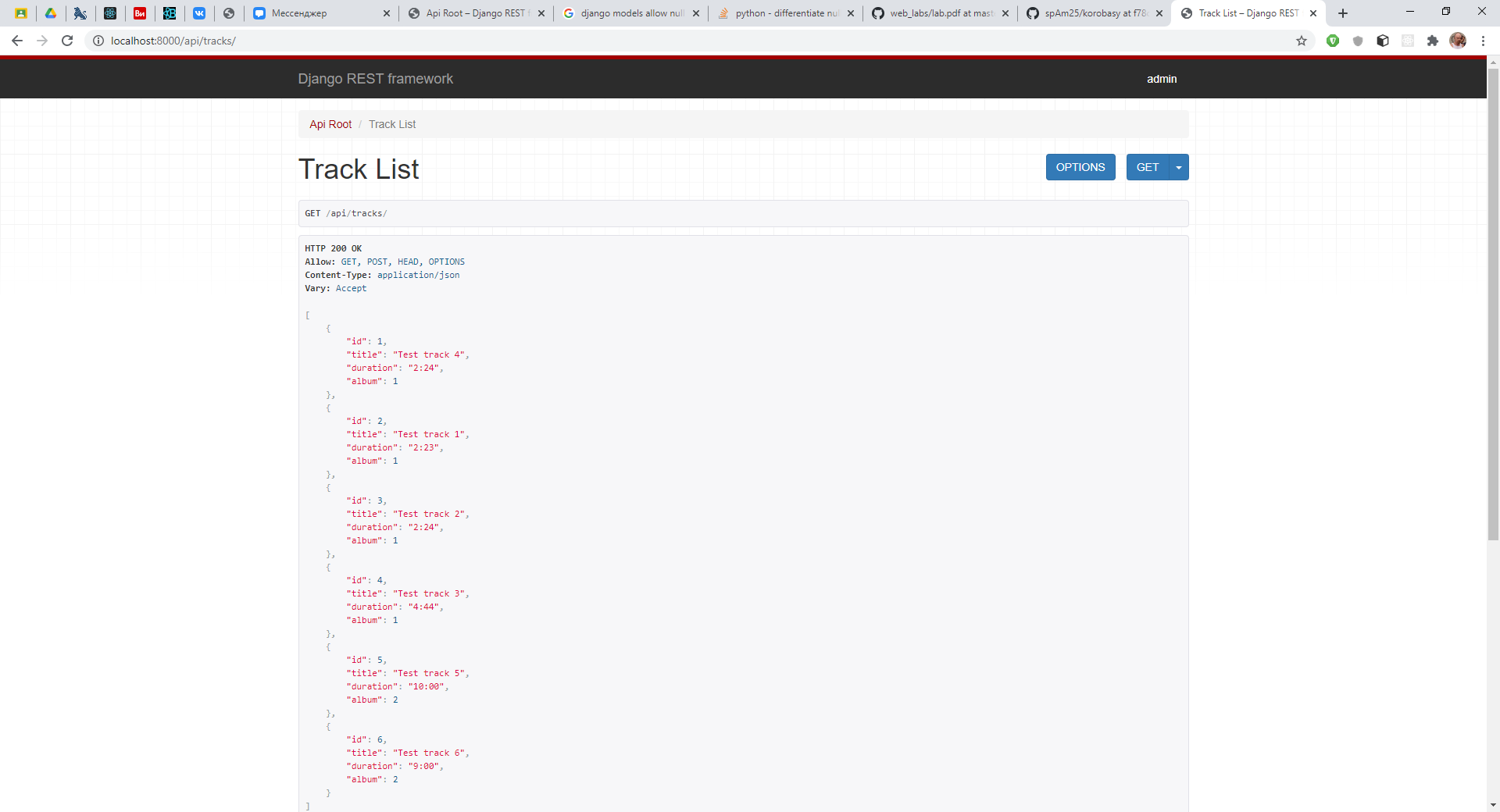


Добавив в нее некоторые тестовые данные и отправим GET запросы в каждый из URL:









Была создана документация по API с помощью drf-yasg, которая доступна по <http://localhost:8000/api/swagger/> или http://localhost:8000/api/redoc/

**Код разработанного приложения**

**Содержимое файла serializers.py:**

from rest\_framework import serializers

from .models import \*

class BandSerializer(serializers.ModelSerializer):

    class Meta:

        model = Band

        fields = '\_\_all\_\_'

class TrackSerializer(serializers.ModelSerializer):

    class Meta:

        model = Track

        fields = '\_\_all\_\_'

class AlbumSerializer(serializers.ModelSerializer):

    class Meta:

        model = Album

        fields = '\_\_all\_\_'

class UserPreferenceSerializer(serializers.ModelSerializer):

    # Отдавать данные пользователя поприкольнее?

    class Meta:

        model = UserPreference

        fields = [

            'user',

            'artists',

            'albums',

            'tracks'

        ]

class GenreSerializer(serializers.ModelSerializer):

    class Meta:

        model = Genre

        fields = '\_\_all\_\_'

class PersonSerializer(serializers.ModelSerializer):

    class Meta:

        model = Person

        fields = '\_\_all\_\_'

**Содержимое файла views.py:**

from django\_filters.rest\_framework import DjangoFilterBackend

from rest\_framework import viewsets

from rest\_framework.permissions import IsAuthenticated

from .serializers import \*

from .models import \*

class BandViewSet(viewsets.ModelViewSet):

    queryset = Band.objects.all()

    serializer\_class = BandSerializer

class TrackViewSet(viewsets.ModelViewSet):

    queryset = Track.objects.all()

    serializer\_class = TrackSerializer

class AlbumViewSet(viewsets.ModelViewSet):

    queryset = Album.objects.all()

    serializer\_class = AlbumSerializer

class UserPreferenceViewSet(viewsets.ModelViewSet):

    #queryset = UserPreference.objects.all()

    serializer\_class = UserPreferenceSerializer

    # Отдаем только аутентифицированному пользователю

    permission\_classes = [IsAuthenticated]

    # Все, что его интересует

    def get\_queryset(self):

        return UserPreference.objects.filter(user=self.request.user)

class GenreViewSet(viewsets.ModelViewSet):

    queryset = Genre.objects.all()

    serializer\_class = GenreSerializer

class PersonViewSet(viewsets.ModelViewSet):

    queryset = Person.objects.all()

    serializer\_class = PersonSerializer