**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ  
ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В.Г.ШУХОВА»  
(БГТУ им. В.Г.Шухова)**

Лабораторная работа по теме

«Серверное программирование. Django»  
дисциплина:

Технологии web-программирования

Выполнил: ст. группы ВТ-41   
Фаракшин Н. Р.  
Проверил: ст. пр. Картамышев С.В.

**Белгород 2020**

**Лабораторная работа № 3**

**Серверное программирование. Django**

**Цель работы:** познакомиться с основами backend разработки web-приложений. Научится писать скрипты на языке Python. Познакомиться с основами работы docker. Познакомиться с фреймворком Django и научиться разворачивать проект, производить его настройку. Научится работать с API в приложении Postman.

**Задание к лабораторной работе:**

1. Развернуть базовое приложение Django.
2. Настроить конфигурацию работы приложения с docker.
3. Добавить модуль для работы с API.
4. Добавить несколько контроллеров со статическими данными.
5. Продемонстрировать работу API в Postman.

**Демонстрация работы приложения**

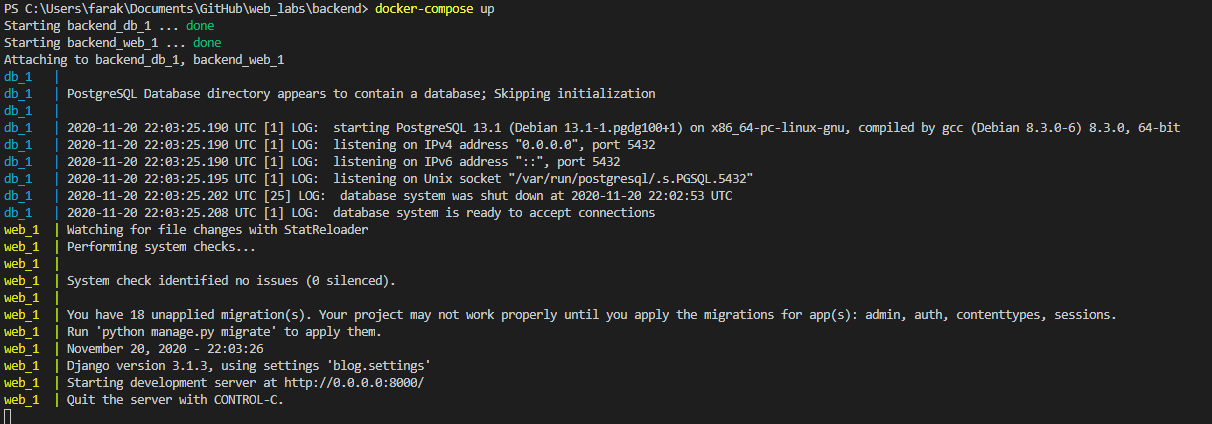


Рис. Запущено два контейнера Docker с помощью Docker-Compose

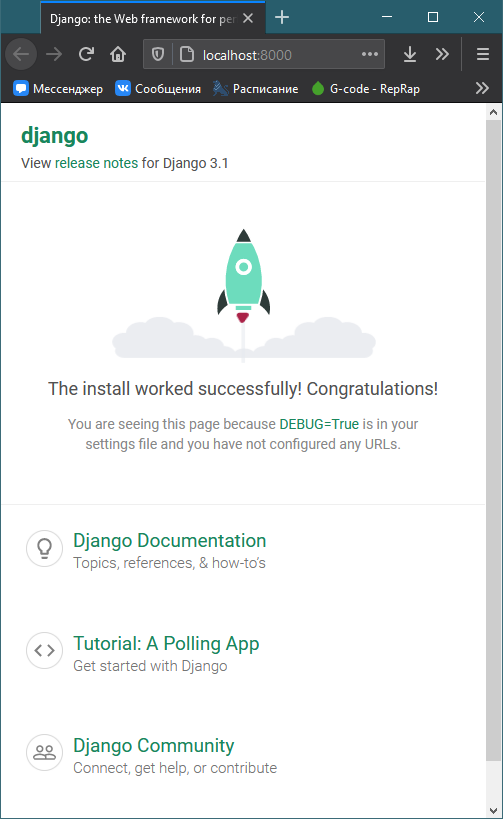


Рис. Приложение Django корректно работает

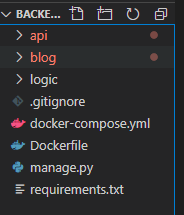


Рис. Структура проекта

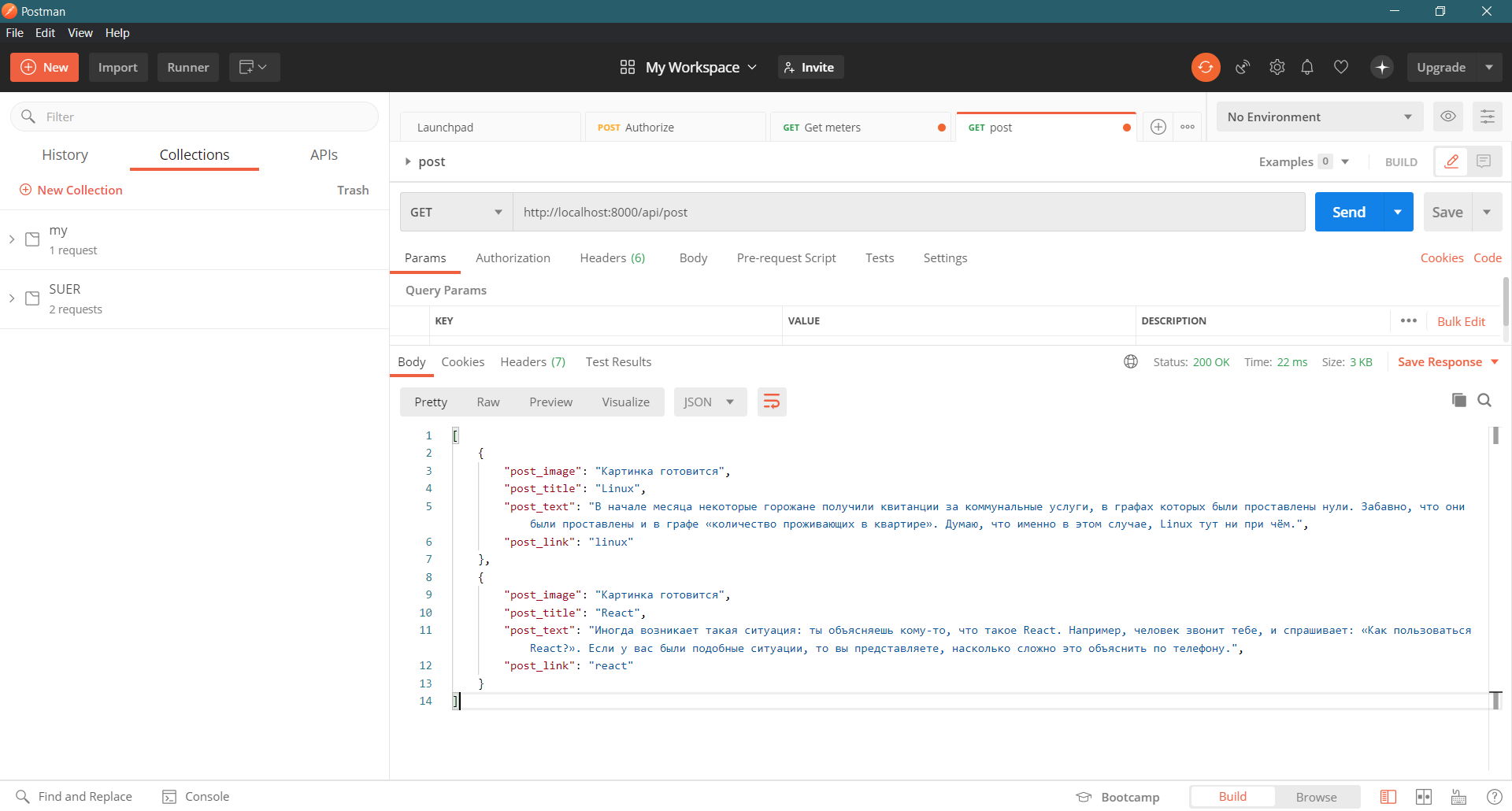


Рис. Запрос к API в приложении Postman успешно получает данные

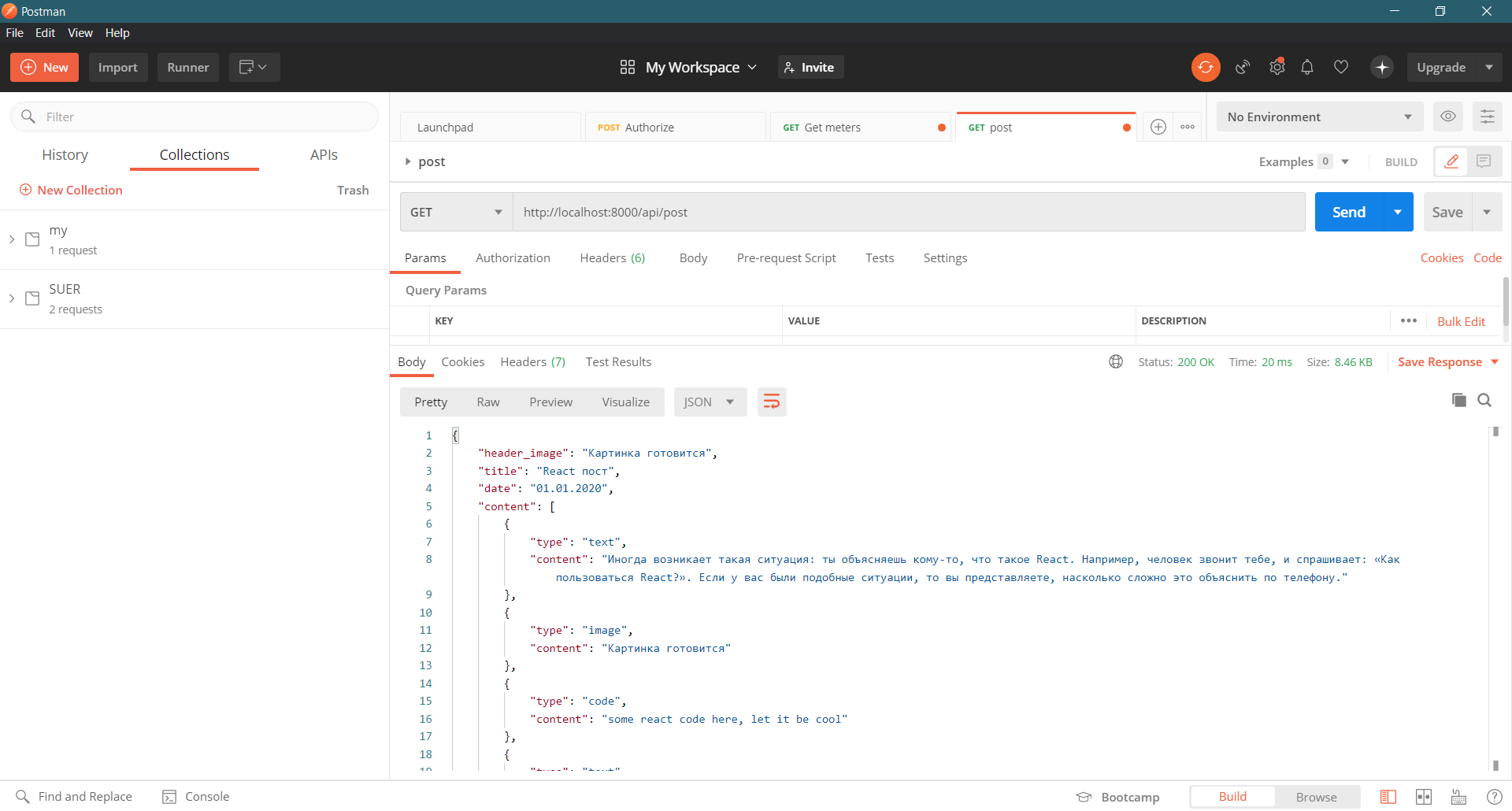


Рис. Запрос к другой конечной точке API

**Приложение**

Исходный код файлов Dockerfile и docker-compose.yml. Код отдающего данные контроллера.

**docker-compose.yml**

version: "3.8"

services:

  db:

    image: postgres

    environment:

      - POSTGRES\_DB=postgres

      - POSTGRES\_USER=postgres

      - POSTGRES\_PASSWORD=postgres

  web:

    build: .

    command: python manage.py runserver 0.0.0.0:8000

    volumes:

      - .:/code

    ports:

      - "8000:8000"

    depends\_on:

      - db

**Dockerfile**

FROM python:3

ENV PYTHONUNBUFFERED=1

WORKDIR /code

COPY requirements.txt /code/

RUN pip install -r requirements.txt

COPY . /code/

**views.py**

def static\_post\_list(request):

    response = json.dumps(

        [

            {

                "post\_image": "Картинка готовится",

                "post\_title": "Linux",

                "post\_text": "В начале месяца некоторые горожане получили…”

                "post\_link": "linux"

            },

            {

                "post\_image": "Картинка готовится",

                "post\_title": "React",

                "post\_text": "Иногда возникает такая ситуация: … ",

                "post\_link": "react"

            },

        ]

    )

    return HttpResponse(response)

def static\_post(request):

    response = json.dumps(

        {

            "header\_image": "Картинка готовится",

            "title": "React пост",

            "date": "01.01.2020",

            "content": [

                {

                    "type": "text",

                    "content": "Иногда возникает такая ситуация: ..."

                },

                {

                    "type": "image",

                    "content": "Картинка готовится"

                },

                {

                    "type": "code",

                    "content": "some react code here, let it be cool"

                },

                {

                    "type": "text",

                    "content": "Нелегка в России судьба React: ..."

                }

            ]

        }

    )

    return HttpResponse(response)