**Как создать проект в Django и использовать стороннее API**

1) API для погоды: [https://openweathermap.org/](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqa0xINHJBc3hwdld0M3o0RmpKaFNnQUJ6LWw2Z3xBQ3Jtc0tsVWtmVjI1TEdEeE40SDhVTVR1U1J3MWRBaHRxcnlYc0tHdVJDcGhycjIyMk1OWGRvcnYtX3hSMGZ2QUZmVXc5Q1ptRFJpWmM5X0ZmM01Dc2txampCYVlqaGZwVmo1OWNlLU1nSG04QjFwVU9yTm5JWQ&q=https%3A%2F%2Fopenweathermap.org%2F&v=lsAbq2RcWlQ)

2) Официальный сайт Django: [https://www.djangoproject.com/](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqblZMdl9hQXdWSDZMZTVrR0NUVGZubDhKcHpRZ3xBQ3Jtc0tuZkZTTWxsSUN5Mk1yUjJIQmw2YXJGbkZnQU5RTlFuRkhzZjdYNXBaTEZmNXNWR1JHVWh6QnJaWV9LVDFQQl9WNkVObFFBQS1fN09GWTJHSkFtR2VPOExmajZ3aG9Yd0ducDlrNnpya0w2eTRHd001SQ&q=https%3A%2F%2Fwww.djangoproject.com%2F&v=lsAbq2RcWlQ)

3) Модуль requests: [https://pypi.org/project/requests/](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbERqZ0ttVlhIUUo2MGR3OXhOT05WWVRGMnJwd3xBQ3Jtc0ttSnV0dUVPbi1ET0Y3a0xyZmctM3ZHUmVoRVY3TVFEcHd6cy0tdUhsNWpCTXdWX2UzYVEtUGVyUmhrb2tBaXJSM2VaSEZLaTdfU0NPME56ejVjR2M2UnM0WWFGX1FrQzRrWnAxdk9iemlqVXhZNEJ3OA&q=https%3A%2F%2Fpypi.org%2Fproject%2Frequests%2F&v=lsAbq2RcWlQ" \t "_blank)

# Django учебники с подробным описанием разработки проекта по шагам (можно использовать примеры оттуда):

# <https://developer.mozilla.org/ru/docs/Learn/Server-side/Django/skeleton_website> , <https://developer.mozilla.org/ru/docs/Learn/Server-side/Django/Models> -

1. <https://metanit.com/python/django/1.3.php>
2. <https://docs.djangoproject.com/en/2.2/intro/tutorial01/> - Документация Джанго от разработчика
3. <https://www.youtube.com/watch?v=lsAbq2RcWlQ&list=PL0lO_mIqDDFXr3RB0UDbUyZixUnIM_7LY&index=25> – видео с описанием как использовать сторонние API
4. <https://pythonru.com/uroki/django-rest-api#Сериализатор-Comment> – создание API блога с аутентификацией и многими распространенными паттернами API (дополнительно о сериализаторах - <https://www.django-rest-framework.org/tutorial/1-serialization/> )
5. <https://www.django-rest-framework.org/tutorial/quickstart/>, <https://www.django-rest-framework.org/api-guide/requests/> - [официальная документация Django REST Framework](https://www.django-rest-framework.org/)
6. Курс по Python: [https://itproger.com/course/python-full](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqbjhMTUdvRDBYOU1iZHd3MXZUWEthRDM4SVREQXxBQ3Jtc0trUlZWNExfMExnTzJudHVjOUotV2d6VW9xRWNyX1dhdkRwVW9qOHg0c213bnVEOG1MQzZFSlZueVRnX2tickRfN3NDV1AwYmpBWXpvTTdSdDZBWjZSWFdtWlU0dXQ2UXE1aEZzRW1CclBoYm5sMGpNYw&q=https%3A%2F%2Fitproger.com%2Fcourse%2Fpython-full&v=lsAbq2RcWlQ)

Установить питон и джанго (в терминале)

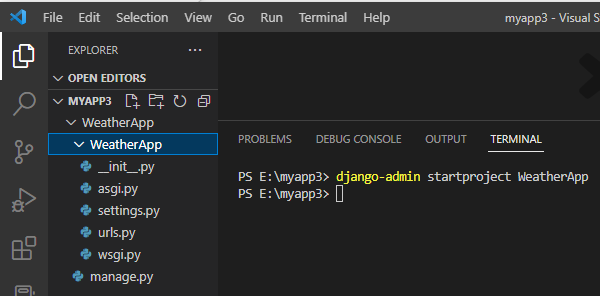
pip install Django

Создать папку E:\myapp3 и открыть ее в visual Studio Code

Открыть терминал и ввести команду

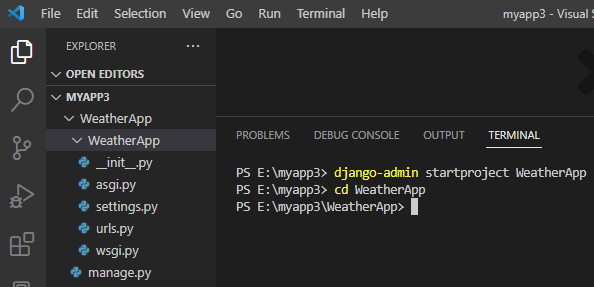
PS E:\myapp3> django-admin startproject WeatherApp

Появится папка WeatherApp с проектом (содержит различные настройки во вложенной одноименной папке и файл manage.py – главный файл с помощью которого можно запускать сервер и работать с разл. приложениями) в visual Studio Code



Перейти в созданную папку в окне терминала

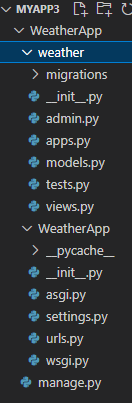
сd WeatherApp



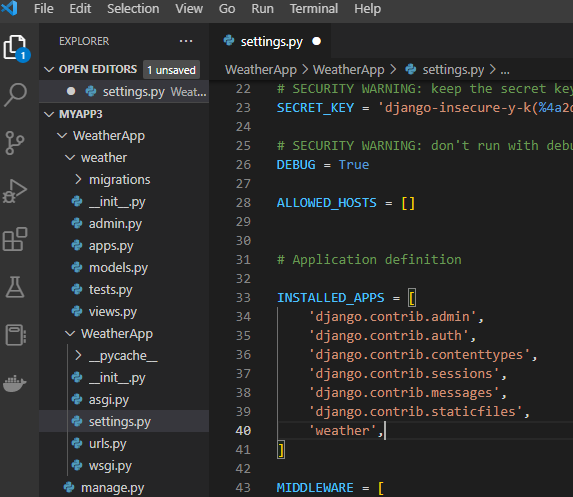
Создать приложение Погода (weather) внутри Django проекта (в одном проекте может быть несколько приложений: форум, магазин, новостной блог)

PS E:\myapp3\WeatherApp> python manage.py startapp weather

Создалась папка с приложением



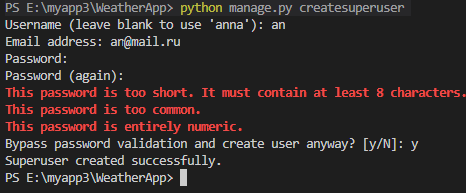
Зарегистрировать добавленное приложение в файле settings.py (регистрация необходима для всех новых приложений)



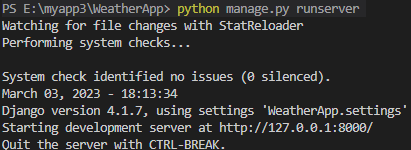
Создать пользователя для админпанели:

1)выполнить подготовительные миграции - PS E:\myapp3\WeatherApp> python manage.py migrate

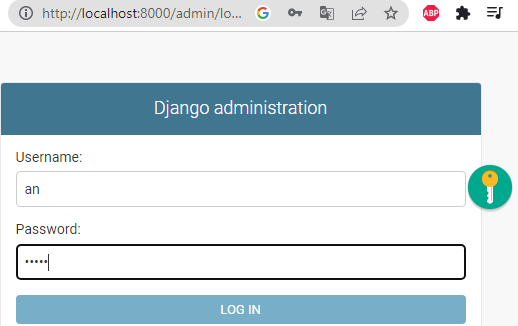
2)создать суперюзера - PS E:\myapp3\WeatherApp> python manage.py createsuperuser



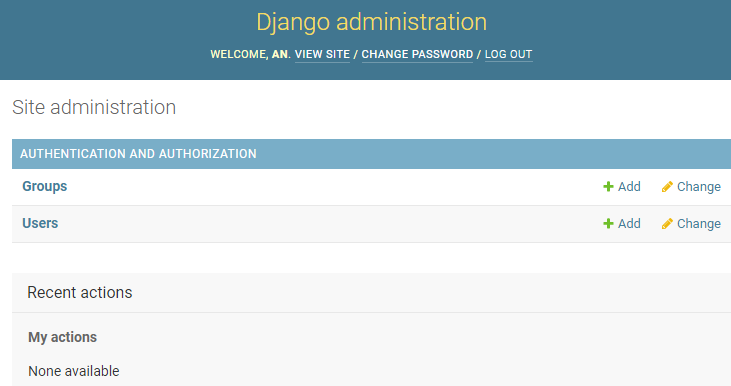
Запустить сервер: PS E:\myapp3\WeatherApp> python manage.py runserver

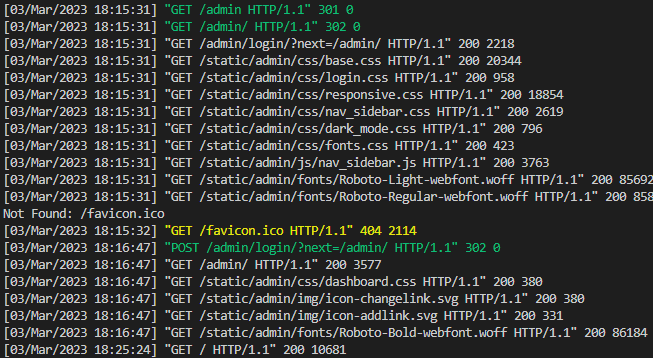


Проверить в браузере – ввести в строку браузера: localhost:8000/admin



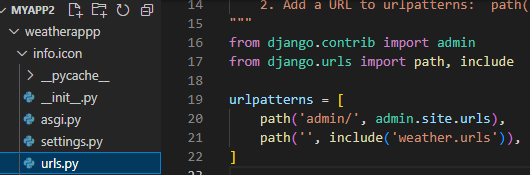
Ввести логин/пароль для созданного суперюзера – и получить доступ к админ панели



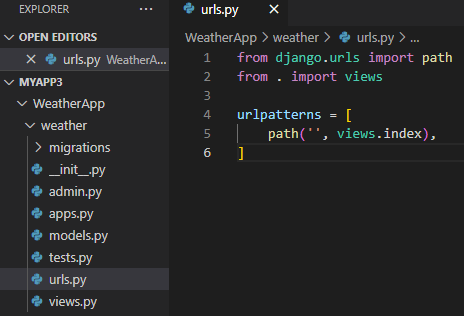


Выйти из локального сервера (Ctrl+C) и создать главную страницу приложения:

1)изменить файл urls.py в папке WeatherApp, добавив новый path к файлу urls.py приложения weather

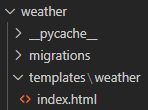


2)создать файл urls.py в приложении weather, в котором из текущей директории импортируем файл views, обращаемся к нему в path и используем функцию index

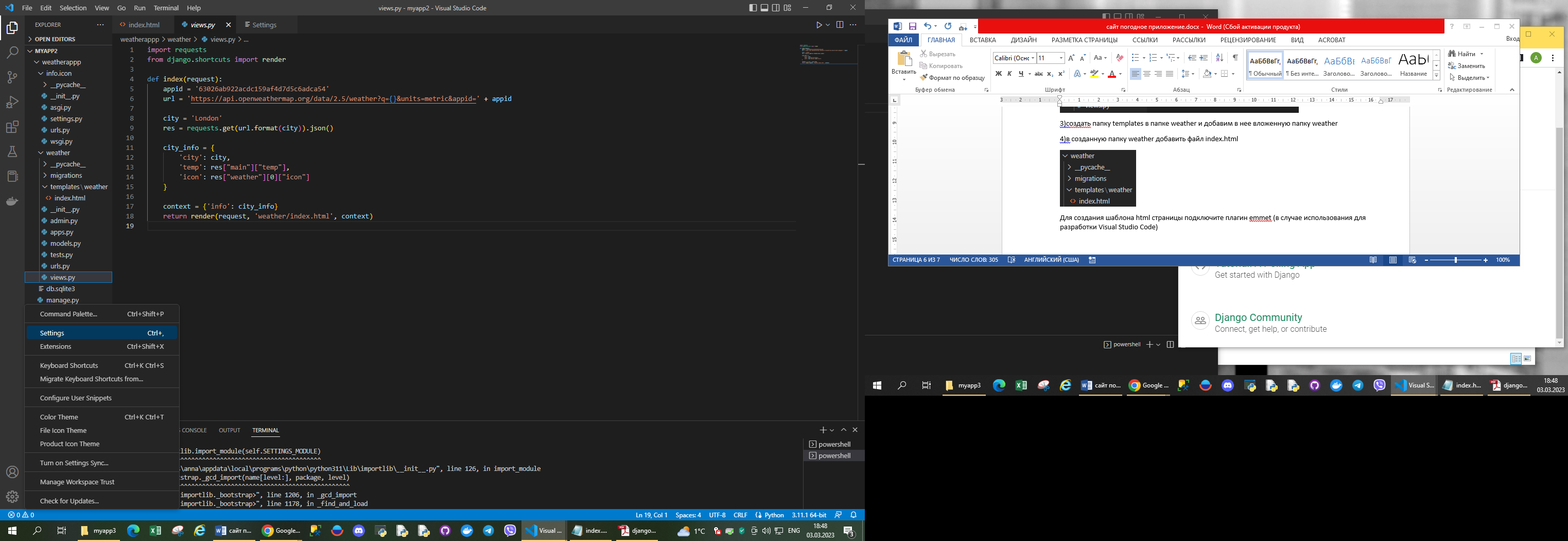
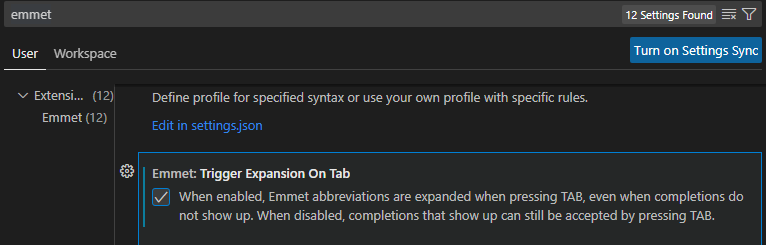


3)cоздать папку templates в папке приложения weather и добавим в нее вложенную папку weather

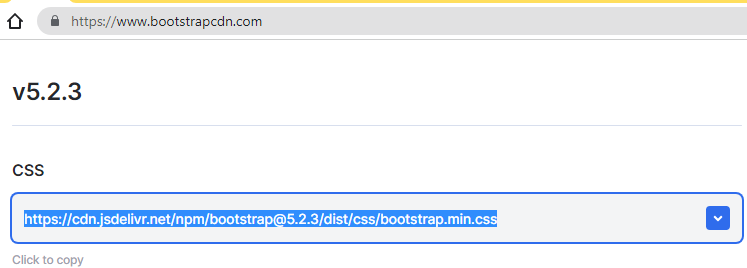
4)в созданную папку weather добавить файл index.html



4.1)Для создания шаблона html страницы подключите плагин emmet (в случае использования для разработки Visual Studio Code – открыть Settings, ввести в строке поиска emmet и установить флажок “Emmet: Trigger Expansion On Tab”)

4.2)Открыть файл index.html , ввести !, нажать Enter – cформируется готовый шаблон, который нужно изменить, добавив необходимые для вывода информации о погоде элементы внутрь тега body и подключив bootstrap для быстрой настройки стилей (<https://www.bootstrapcdn.com/> , <https://getbootstrap.com/> )



<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

</body>

</html>

<!DOCTYPE html>

<html lang="ru">

<head>

    <meta charset="UTF-8">

    <meta http-equiv="X-UA-Compatible" content="IE=edge">

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

    <title>Document</title>

</head>

<body>

     <h1>Погодное приложение</h1>

    <div class="row">

  <form action="">

    <label for="city">Город</label>

    <input type="text" id="city" class="form-control" name="city" placeholder="Введите город">

    <input type="submit" name="send" value="Узнать" class="mt-2 btn btn-danger">

  </form>

  <div class="col-4 offset-4">

    <h1>информация</h1>

    <div class="alert alert-danger">

      погода в

      городе: {{ info.city }} <br>

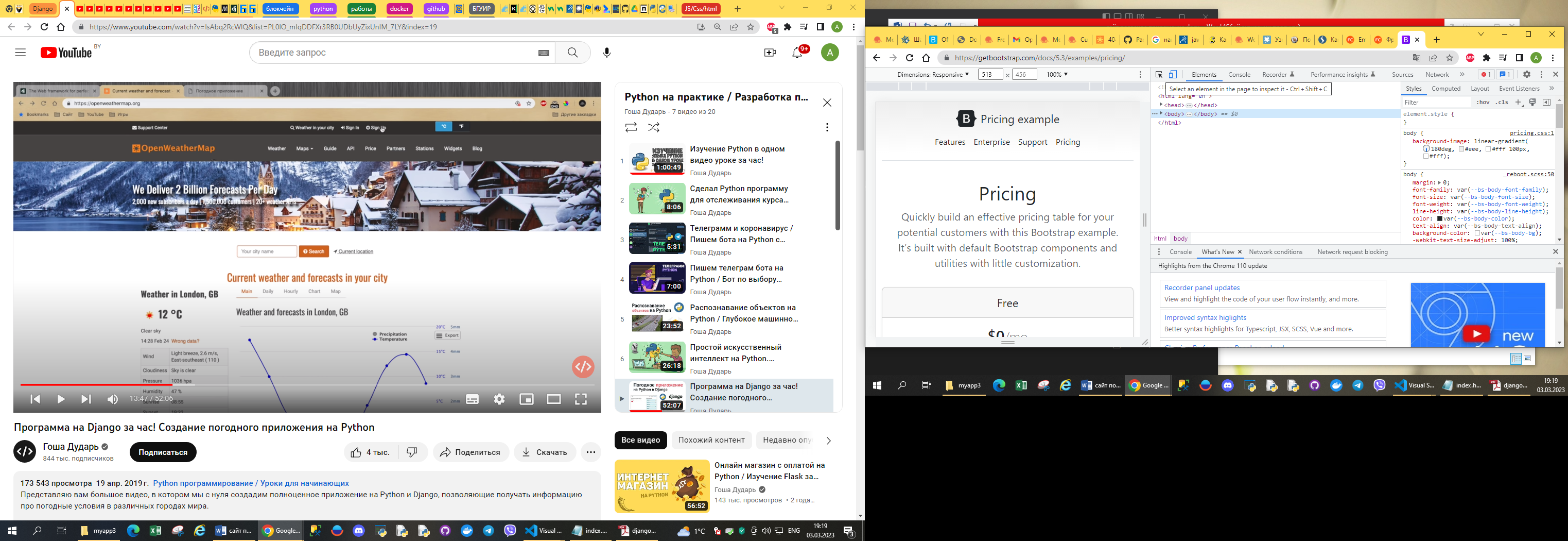
      температура: {{ info.temp }}<sup>0</sup> <br>

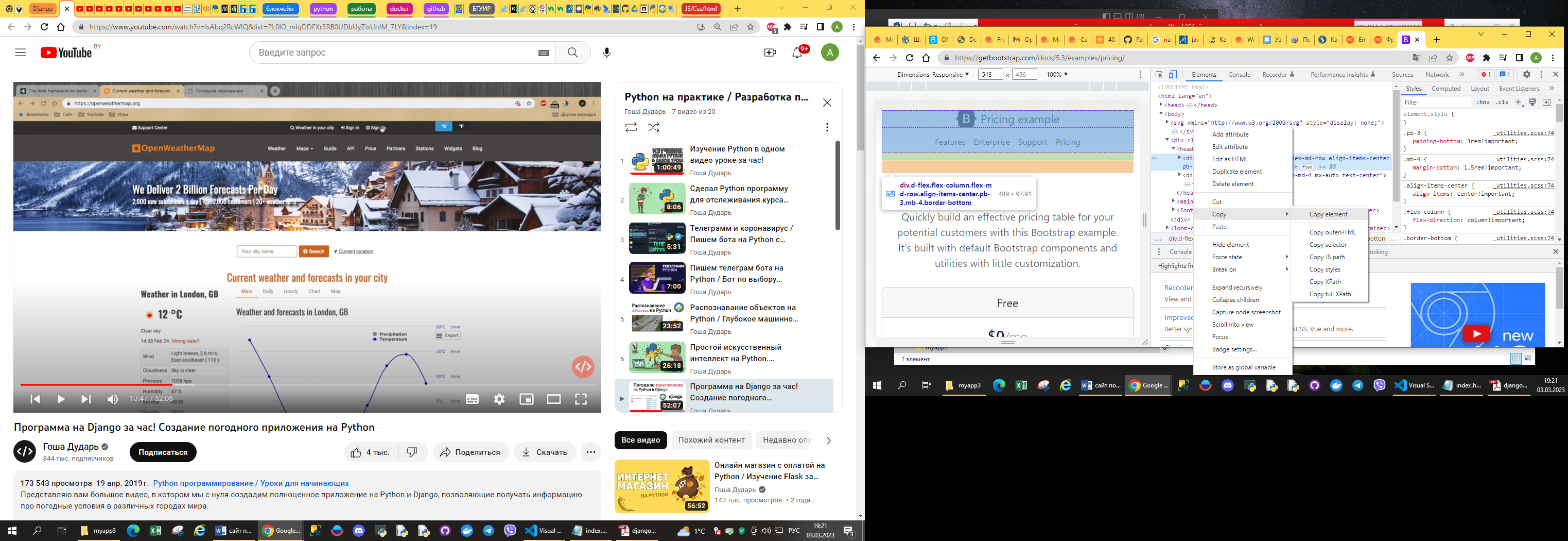
           <img src="http://openweathermap.org/img/wn/{{ info.icon }}@2x.png" alt="фото" class="img-thumbnail">

</body>

</html>

4.3)Можно добавить шапки страницы из любого примера bootstrap(выбрать пример из <https://getbootstrap.com/docs/5.3/examples/pricing/>, нажать в контекстном меню пункт Inspect и затем кнопку выбора нужного элемента )

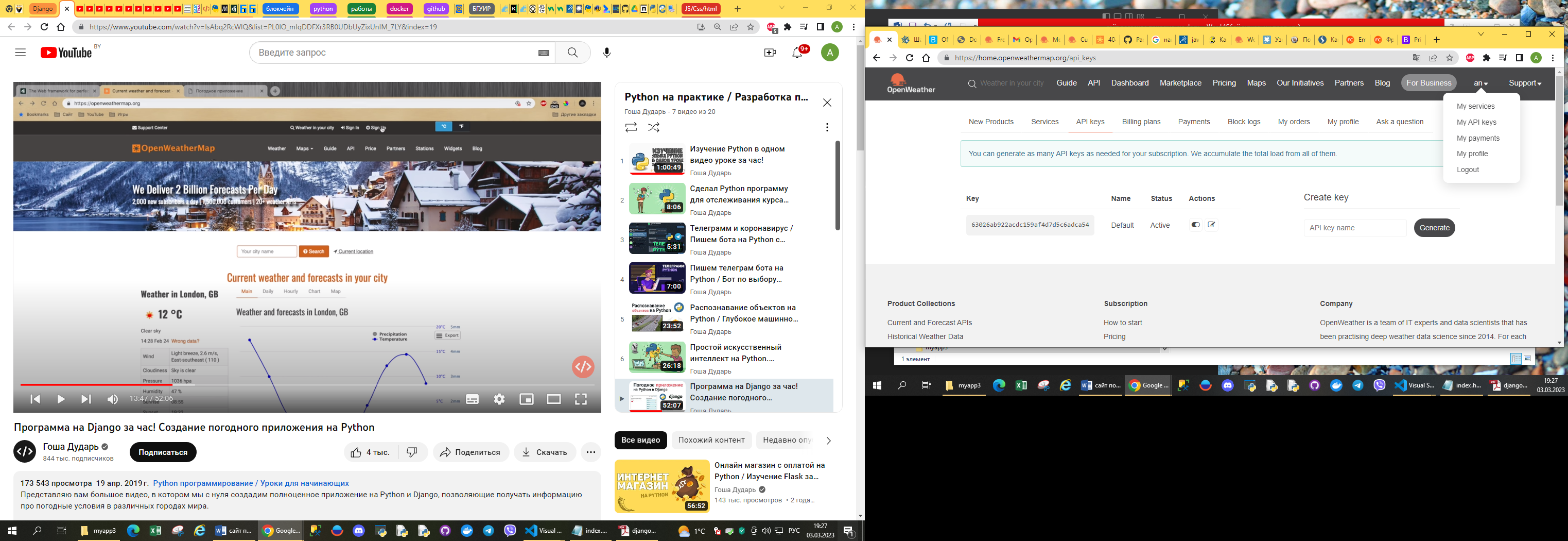




4.4)С помощью пункта меню Copy element скопировать нужный код, вставить его и изменить по своему вкусу.

5)cоздать в файле **views.py** приложения weather функцию index(request) для получения данных

5.1) зарегистрироваться на сайте [https://openweathermap.org/](https://www.youtube.com/redirect?event=video_description&redir_token=QUFFLUhqa0xINHJBc3hwdld0M3o0RmpKaFNnQUJ6LWw2Z3xBQ3Jtc0tsVWtmVjI1TEdEeE40SDhVTVR1U1J3MWRBaHRxcnlYc0tHdVJDcGhycjIyMk1OWGRvcnYtX3hSMGZ2QUZmVXc5Q1ptRFJpWmM5X0ZmM01Dc2txampCYVlqaGZwVmo1OWNlLU1nSG04QjFwVU9yTm5JWQ&q=https%3A%2F%2Fopenweathermap.org%2F&v=lsAbq2RcWlQ) и получить свой личный уникальный ключ для отправки API запросов, который присвоить переменной appid в функции index(request).

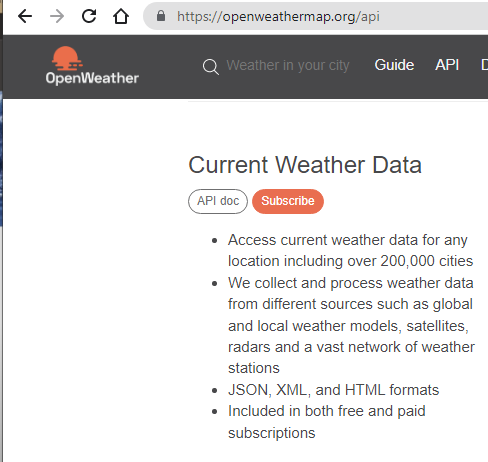


url скопировать из API doc – изменить как в примере ниже,

там же посмотреть расшифровку всех параметров (добавить к тому что есть в примере функции сведения о других пяти параметрах температуры)

при помощи Django переходим на указанный url адрес получаем инфо в json формате.

Для отправки запроса и получения данных по нему нужно установить модуль requests и подключить его в файле views.py (import requests) (в терминале выполнить команду – pip install requests) – инфо о всех модулях находится на <https://pypi.org> (<https://pypi.org/project/requests/> )



import requests

from django.shortcuts import render

def index(request):

    appid = '63026ab922acdc159af4d7d5c6adca54'

    url = 'https://api.openweathermap.org/data/2.5/weather?q={}&units=metric&appid=' + appid

    city = 'London'

    res = requests.get(url.format(city)).json()

# получили конвертирование json формата в словарь

    city\_info = {

        'city': city,

        'temp': res["main"]["temp"],

        'icon': res["weather"][0]["icon"]

    }

    context = {'info': city\_info}

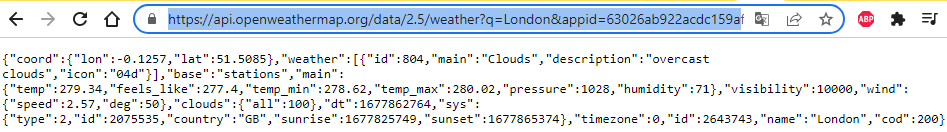
    return render(request, 'weather/index.html', context)

переменная res – ответ на запрос по значению города (city)

units=metric – для преобразования температуры в цельсии (по умолчанию Фаренгейт)

(для вывода в терминал всего ответа используйте print(res.text), но до преобразования в json)

Или можно url <https://api.openweathermap.org/data/2.5/weather?q=London&appid=63026ab9>... (используйте свой ключ для значения appid) ввести в строку браузера



city\_info – параметры согласно API doc

context – словарь, предается в шаблон index.html:

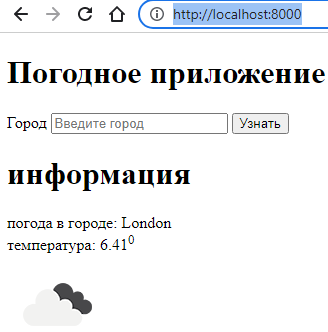
      городе: {{ info.city }} <br>

      температура: {{ info.temp }}<sup>0</sup> <br>

           <img src="http://openweathermap.org/img/wn/{{ info.icon }}@2x.png"

Дополнительно о передаче данных в шаблон смотри тут: **Передача данных в шаблоны** https://metanit.com/python/django/2.3.php

Запустить сервер в терминале и проверить работу приложения в браузере



6) Добавить возможность просмотра погоды для разных городов, выбирая их из списка

6.1)создать таблицу из городов в файле models.py приложения weather:

- открыть добавленный при создании приложения файл **models.py**, который используется для определения моделей. Модель представляет класс, унаследованный от django.db.models.Model.

Класс City – наследник models.Model

Name – тип данных CharField

Добавить в файл (имя класса – имя добавляемой таблицы, атрибуты – добавляемые поля таблицы):

from django.db import models

class City(models.Model):

    name = models.charField(max\_length=30)

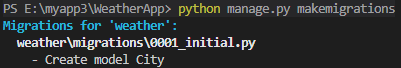
    def \_\_str\_\_(self):

        return self.name

6.2) создать миграции:

Каждая модель сопоставляется с определенной таблицей в базе данных. Для создания таблицы в бд, которая хранит объекты модели Person, надо создать и выполнить миграцию. Миграция преобразует базу данных в соответствии с определением моделей:

PS E:\myapp3\WeatherApp> python manage.py makemigrations



После этого в приложении в папку migrations добавляется файл (у меня 0001\_initial.py) с содержимым – это и есть миграция. Поле id, которое будет представлять первичный ключ, добавляется по умолчанию.

Другие типы полей моделей можно посмотреть тут: https://metanit.com/python/django/5.2.php

# Generated by Django 4.1.7 on 2023-03-03 17:34

from django.db import migrations, models

class Migration(migrations.Migration):

    initial = True

    dependencies = [

    ]

    operations = [

        migrations.CreateModel(

            name='City',

            fields=[

                ('id', models.BigAutoField(auto\_created=True, primary\_key=True, serialize=False, verbose\_name='ID')),

                ('name', models.CharField(max\_length=30)),

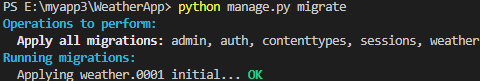
            ],

        ),

    ]

Выполнить данную миграцию:

PS E:\myapp3\WeatherApp> python manage.py migrate



6.3) добавить таблицу в файл admin.py приложения weather, чтобы она отображалась в панели администратора (импортируем таблицу City из текущей директории и регистрируем ее в панели админа)

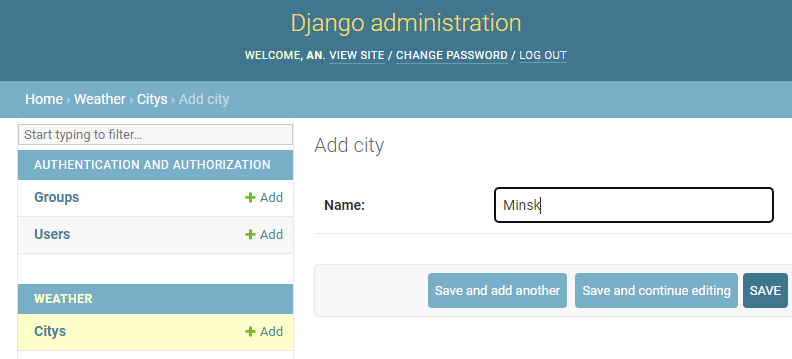
from django.contrib import admin

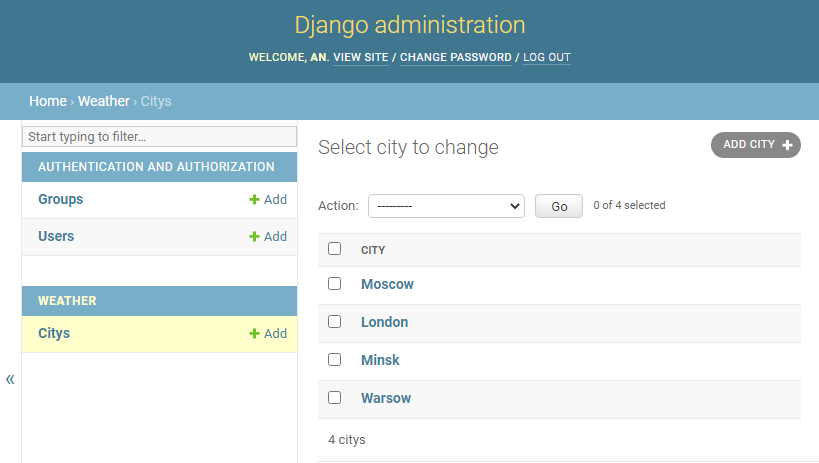
from .models import City

# Register your models here.

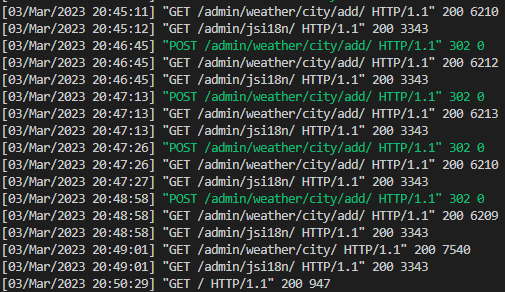
admin.site.register(City)

запустить сервер (PS E:\myapp3\WeatherApp> python manage.py runserver), перейти в панель админа, добавить города в таблицу





В терминале увидеть как выполняются соответствующие запросы:



Также можно добавить новые объекты города в файле **models.py** с помощью одного из методов, описанных здесь: https://metanit.com/python/django/5.12.php

from django.db import models

class City(models.Model):

    name = models.CharField(max\_length=30)

    def \_\_str\_\_(self):

        return self.name

Vitebsk=City.objects.create(name="Vitebsk")

6.4)реализовать добавление городов в таблицу на главной странице

Файл index.html – добавить цикл для прохода по словарю all\_info

