# PROFESSIONAL TRAINING REPORT

**at**

# Sathyabama Institute of Science and Technology (Deemed to be University)

Submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering Degree in Computer Science and Engineering

By

**DEVIREDDYSRAVANI LAKSHMI DURGA (Reg.No: - 40110312)**



# DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING SCHOOL OF COMPUTING

**SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY JEPPIAAR NAGAR, RAJIV GANDHI SALAI,**

# CHENNAI – 600119, TAMILNADU

**OCT 2022**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY**

**(DEEMED TO BE UNIVERSITY)**

## Accredited with Grade “A” by NAAC

**(Established under Section 3 of UGC Act, 1956)**

**JEPPIAAR NAGAR, RAJIV GANDHI SALAI, CHENNAI– 600119**

[**www.sathyabamauniversity.ac.in**](http://www.sathyabamauniversity.ac.in/)

## DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

**BONAFIDE CERTIFICATE**

This is to certify that this Project Report is the bonafide work of **D.Sravani Lakshmi Durga (40110312)** who carried out the project entitled “**Weather Report Python Django**” under my supervision from Aug 2022 to Oct 2022.

**Internal Guide**

**Dr. S.L. Jany Shabu M. Tech ,Ph . D**

## Head of the Department

**Dr. L. Lakshmanan M.E., Ph.D.,**



**Submitted for Viva voce Examination held on**

**Internal Examiner External Examiner**

# DECLARATION

I, **D. Sravani Lakshmi Durga** here by declare that the Project Report entitled “ **Weather Report**

**Python Django**” done by me under the guidance of **Dr. S.L. Jany Shabu M. Tech , Ph. D** is submitted in partial fulfillment of the requirements for the award of Bachelor of Engineering degree in Computer Science and Engineering.

**DATE:**

**PLACE: SIGNATURE OF THECANDIDATE**

# ACKNOWLEDGEMENT

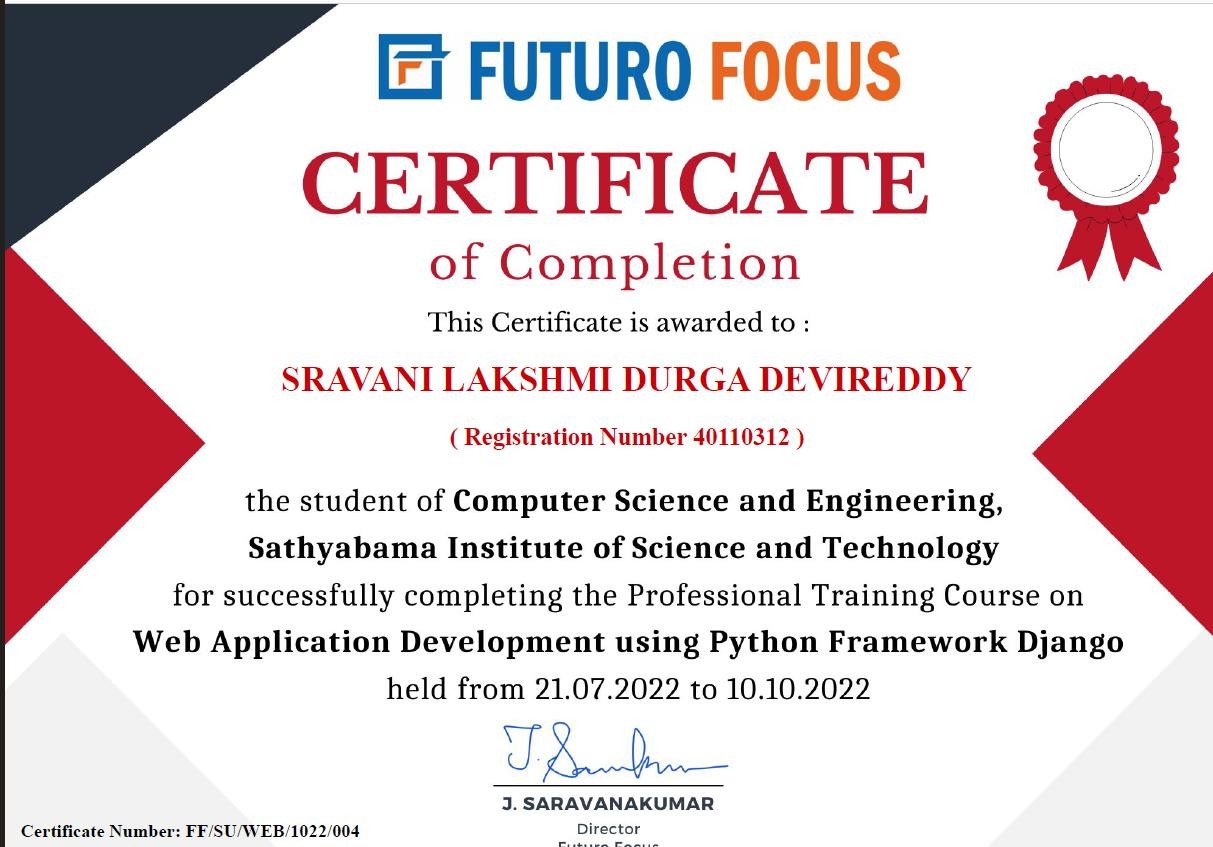
I am pleased to acknowledge my sincere thanks to **Board of Management** of **SATHYABAMA** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D.**, **Dean**, School of Computing, and **Dr. S. Vigneshwari M.E., Ph.D., and Dr. L. Lakshmanan M.E., Ph.D., Heads of the Department of Computer Science and Engineering** for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide, **Dr. S.L. Jany Shabu M. Tech , Ph. D** for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Computer Science and Engineering** who were helpful in many ways for the completion of the project.

# TRAINING CERTIFICATE



# ABSTRACT

Weather forecasting is the application of science and technology to predict the state of the atmosphere for a given location. Ancient weather forecasting methods usually relied on observed patterns of events, also termed pattern recognition. For example, it might be observed that if the sunset was particularly red, the following day often brought fair weather. However, not all of these predictions prove reliable.

Here this system will predict weather based on parameters such as temperature, humidity and wind. User will enter current temperature; humidity and wind, System will take this parameter and will predict weather (rainfall in inches) from previous data in database(dataset). The role of the admin is to add previous weather data in database, so that system will calculate weather (estimated rainfall in inches) based on these data. Weather forecasting system takes parameters such as temperature, humidity, and wind and will forecast weather based on previous record therefore this prediction will prove reliable. This system can be used in Air Traffic, Marine, Agriculture, Forestry, Military, and Navy etc.

# TABLE OF CONTENTS

**CHAPTER TITLE PAGE NO**

[ABSTRACT 06](#_TOC_250020)

1. INTRODUCTION 09
2. AIM AND SCOPE OF THE

PRESENT INVESTIGATION 10

* 1. [Aim 10](#_TOC_250019)
  2. [Scope of investigation 10](#_TOC_250018)
     1. [Configuration 10](#_TOC_250017)
     2. [Hardware Requirement 11](#_TOC_250016)
     3. [Software Requirement 11](#_TOC_250015)

1. USED TECHNOLOGIES 12
   1. [HTML 12](#_TOC_250014)
   2. [CSS 12](#_TOC_250013)
   3. [Java Script 13](#_TOC_250012)
   4. [Python 14](#_TOC_250011)
   5. [Django 14](#_TOC_250010)
      1. [Features Of Django 15](#_TOC_250009)
   6. [Django Frame Work 15](#_TOC_250008)
   7. [Pillow 16](#_TOC_250007)
2. SYSTEM ANALYSIS AND IMPLEMENTATION 17

OF THE APPLICATION

* 1. [Requirements 17](#_TOC_250006)
  2. [Installation 17](#_TOC_250005)
     1. [Django 17](#_TOC_250004)
     2. [Pillow 17](#_TOC_250003)
  3. [Creation 18](#_TOC_250002)
  4. [Execution 18](#_TOC_250001)

1. RESULT AND CONCLUSIONS 19
   1. [Result 20](#_TOC_250000)
   2. Conclusions 21
2. APPENDIX
   1. Reference 22
   2. Screenshots 22
   3. Source code 25
   4. SYSTEM ARCHITECTURE 36



**CHAPTER-1 INTRODUCTION**

The “Weather Webapp Using Python Django And Weather API” is very users friendly it is design to help user to access Weather of different place all over the world on the screen. This webapp is made using React js and Weather API because it runs on local host and it is very user friendly and it does not consume more memory. This webapp can be run on limited data and it can be run many old device it does not take lot of time to load. To make this application, we need an API (*Application Programming Interface*) which will provide the data you need to create a weather web application. For the current project, we will use an API to retrieve weather data in the world. Some APIs can be used for free, while some APIs have to be paid for a certain price.

To create a weather web application, we will use the API from OpenWeatherMap.org which you can use for free or paid. To use this API, you need an API key. To get the API key, you must register with the Open Weather Map web first. So please log in/sign up to https://home.openweathermap.org/users/sign\_in.then later on we will use react js and deploy this web application.

1. **RELATED WORK**

There are many big Weather forecasting app which have many things to offer so we have made simple and user friendly Webapp which can be used by any user without any difficult and it will show the real time weather forecast of the place which user has input in the search bar. As it is simple program the runtime is very good weather get load very fast.

Benefit of Weather webapp

* 1. As it is running on local host it is safe
  2. Load time is very fast
  3. Less memory
  4. It can be used on small device

# CHAPTER-2

**AIM AND SPECIFICATION OF WEB APPLICATION**

## AIM

Aim of this project is to develop a weather application in python django.

Weather apps enable users to get instant alerts regarding weather conditions. Weather apps are the simplest method to know about the updates of the upcoming weather. The apps make a forecast of the climate based on the location services. The standard features of the app offer daily, weekly and monthly forecast data. User can view the weather in his/her current location. Weather icon or background image will change depending on weather conditions. User can toggle the temperature unit in Celsius. User can search for weather information of other places.

## SCOPE OF INVESTIGATION

Weather forecasting is the prediction of the state of the atmosphere for a given location using the application of science and technology. This includes temperature, rain, cloudiness, wind speed, and humidity. Weather warnings are a special kind of short-range forecast carried out for the protection of human life. Weather warnings are issued by the governments throughout the world for all kinds of threatening weather events including tropical storms and tropical cyclones depending upon the location. The forecast may be short-range or Long-range. It is a very interesting and challenging task. This report provides a basic understanding of the purpose and scope of weather forecasts, the basic principles and the general models developed for forecasting. 1.0 Introduction: Weather forecasting means the prediction of the weather through the application of the principles of physics, supplemented by a variety of statistical and empirical techniques. In addition to predictions of atmospheric phenomena themselves, weather forecasting includes predictions of changes on the Earth's surface climate. These changes are caused by atmospheric conditions like snow and ice cover, storm tides, and floods.

The basis for weather prediction started with the theories of

the ancient Greek philosophers and continued with Renaissance scientists. It was followed by the scientific revolution of the 17th and 18th centuries. The theoretical models of 20th-and 21st-century atmospheric scientists and meteorologists helped for the betterment in applications. The so-called synoptic weather map came to be the principal tool of 19th-century meteorologists. This is used today in weather stations and on television weather reports all over the world. All can happen only through a comprehensive weather forecast.

## CONFIGURATION

We can configure this project on following Operating System:

***Windows*:** This project can easily be configured on the windows operating system. For running this project on Windows system, you will have to install Python, PIP, Django. ***Linux*:** We can run this project also on all versions of Linux Operating system.

***Mac:*** We can also easily configure this project on Mac Operating system.

## HARDWARE REQUIREMENT

Minimum RAM: - 1GB Minimum Hard Disk: - 128GB

Processor: - Intel Pentium 4(1.50 GHz) or above

## SOFTWARE REQUIREMENT

Operating System: - Support for both LINUX and WINDOWS users

Back End: - Python,Django,Django framework Front End Language: - CSS & Java

IDE :- Visual Studio Code

# CHAPTER-3 USED TECHNOLOGIES

## HTML

HTML is an acronym which stands for Hyper Text Markup Language which is used for creating web pages and web applications. Let's see what is meant by Hypertext Markup Language, and Web page.

**Hyper Text:** HyperText simply means "Text within Text." A text has a link within it, is a hypertext. Whenever you click on a link which brings you to a new webpage, you have clicked on a hypertext. HyperText is a way to link two or more web pages (HTML documents) with each other.

**Markup language**: A markup language is a computer language that is used to apply layout and formatting conventions to a text document. Markup language makes text more interactive and dynamic. It can turn text into images, tables, links, etc.

**Web Page:** A web page is a document which is commonly written in HTML and translated by a web browser. A web page can be identified by entering an URL. A Web page can be of the static or dynamic type. With the help of HTML only, we can create static web pages.

Hence, HTML is a markup language which is used for creating attractive web pages with the help of styling, and which looks in a nice format on a web browser. An HTML document is made of many HTML tags and each HTML tag contains different content.

## CSS

**C**ascading **S**tyle **S**heets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.

CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs,variations in display for different devices and screen sizes as well as a variety of other effects.

CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTML.

Advantages of CSS

* + - **CSS saves time** − You can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
    - **Pages load faster** − If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
    - **Easy maintenance** − To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
    - **Superior styles to HTML** − CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
    - **Multiple Device Compatibility** − Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.
    - **Global web standards** − Now HTML attributes are being deprecated and it is being recommended to use CSS. So its a good idea to start using CSS in all the HTML pages to make them compatible to future browsers.

## JAVA SCRIPT

JavaScript is a dynamic programming language that's used for web development, in web applications, for game development, and lots more. It allows you to implement dynamic features on web pages that cannot be done with only HTML and CSS.Many browsers use JavaScript as a scripting language for doing dynamic things

on the web. Any time you see a click-to-show dropdown menu, extra content added to a page, and dynamically changing element colors on a page, to name a few features, you're seeing the effects of JavaScript.

## PYTHON

Python is the language used to build the Django framework. It is a dynamic scripting language similar to Perl and Ruby. The principal author of Python is Guido van Rossum. Python supports dynamic typing and has a garbage collector for automatic memory management. Another important feature of Python is dynamic name solution which binds the names of functions and variables during execution. All the business logic has been used as database for the project.

## DJANGO

* + - Django is a high-level Python web framework that enables rapid development of secure and maintainable websites. Built by experienced developers, Django takes care of much of the hassle of web development, so you can focus on writing your app without needing to reinvent the wheel. It is free and open source, has a thriving and active community, great documentation, and many options for free and paid-for support.
    - Django can be (and has been) used to build almost any type of website — from content management systems and wikis, through to social networks and news sites. It can work with any client-side framework, and can deliver content in almost any format (including HTML, RSS feeds, JSON, and XML).
    - Internally, while it provides choices for almost any functionality you might want (e.g. several popular databases, templating engines, etc.), it can also be extended to use other components if needed.

## 3.5.1 FEATURES OF DJANGO:

Rapid Development. Secure.

Scalable.

Fully loaded.

Versatile.

Open Source.

Vast and Supported Community.

## DJANGO FRAMEWORK

* 1.Django is an open source web application frame work written in Python. The primary goal of Django is to make the development of complex, data-based websites easier. Thus Django emphasizes the

reusability and pluggability of components to ensure rapid developments.

* 2.Django consists of three major parts: Model, View and Template.

## PILLOW

* PIL is an abbreviation of Python Imaging Library and it adds image processing to Python.
* The Pillow library contains all the basic image processing functionality. You can do image resizing, rotation and transformation.
* Pillow module allows you to pull some statistics data out of image using histogram method, which later can be used for statistical analysis and automatic contrast enhancement.
* Pillow module gives more functionalities, runs on all major operating system and support for python
* You can do almost anything on digital images using pillow module. Apart from basic image processing functionality, including point operations, filtering images using built-in convolution kernels, and color space conversions.

## DATA MANAGEMENT

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a [database management system (DBMS)](https://www.oracle.com/in/database/what-is-database/#WhatIsDBMS). Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just database.

Data within the most common types of databases in operation today is typically modeled in rows and columns in a series of tables to make processing and data querying efficient. The data can then be easily accessed, managed, modified, updated, controlled, and organized. Most databases use structured query language (SQL) for writing and querying data

## 3.8.1 SQLite3

SQLite3 can be integrated with Python using sqlite3 module, which was written by Gerhard Haring. It provides an SQL interface compliant with the DB-API 2.0 specification described by PEP 249. You do not need to install this module separately because it is shipped by default along with Python version 2.5.x onwards. To use sqlite3 module, you must first create a connection object that represents the database and then optionally you can create a cursor object, which will help you in executing all the SQL statements.

# CHAPTER-4

**SYSTEM ANALYSIS AND IMPLEMENTATIONOF THE**

# APPLICATION

## REQUIREMENTS

* + - 8GB Ram
    - SSD (HDD - replace) : 400mb/s - 4000mb/s
    - Windows, MacOS, Linux
    - Python
    - Django Framework
    - Pillow

## INSTALLATION

We need to install the requirements for the project like Django, pillow

,Django Framework, SQLite 3.

## DJANGO

* + - 1. Django can be installed easily using pip .
      2. In the command prompt, execute the following command: **pip install django** . 3.This will download and install Django.

4.After the installation has completed, you can verify your Django installation by executing django-admin --version in the command prompt.

## PILLOW

Open Terminal (Applications/Terminal) and run:

* + - 1. xcode-select -install (You will be prompted to install the Xcode Command Line Tools)
      2. sudo easy\_install pip.
      3. sudo pip install pillow. 4.pip install pillow.

## CONNECTING SQLITE 3 WITH PYTHON

To create a connection between the MySQL database and Python, the connect() method of mysql.connector module is used.

* + - 1. Download and Install the free MySQL database
      2. After installing the MySQL database, open your Command prompt.
      3. Navigate your Command prompt to the location of PIP.
      4. Now run the commands given below to download and install “MySQL Connector”. 5.Test MySQL Connector

6.Create Connection.

## CREATION

Step 1: Open file in “Visual Studio Code” after that click “new project“.

Step 2: Choose Django and click.

Step 3: Select file location, wherever you want. Step 4: Create application name.

Step 5: Click create. Finish creating project by clicking “create” button. Step 6: Start of coding.

## EXECUTION

Step 1: Extract/unzip the file

Step 2: Go inside the project folder, open cmd and type the following commands to install Django Framework and run the webserver

## pip install -r requirements.txt

1. **python manage.py runserver**

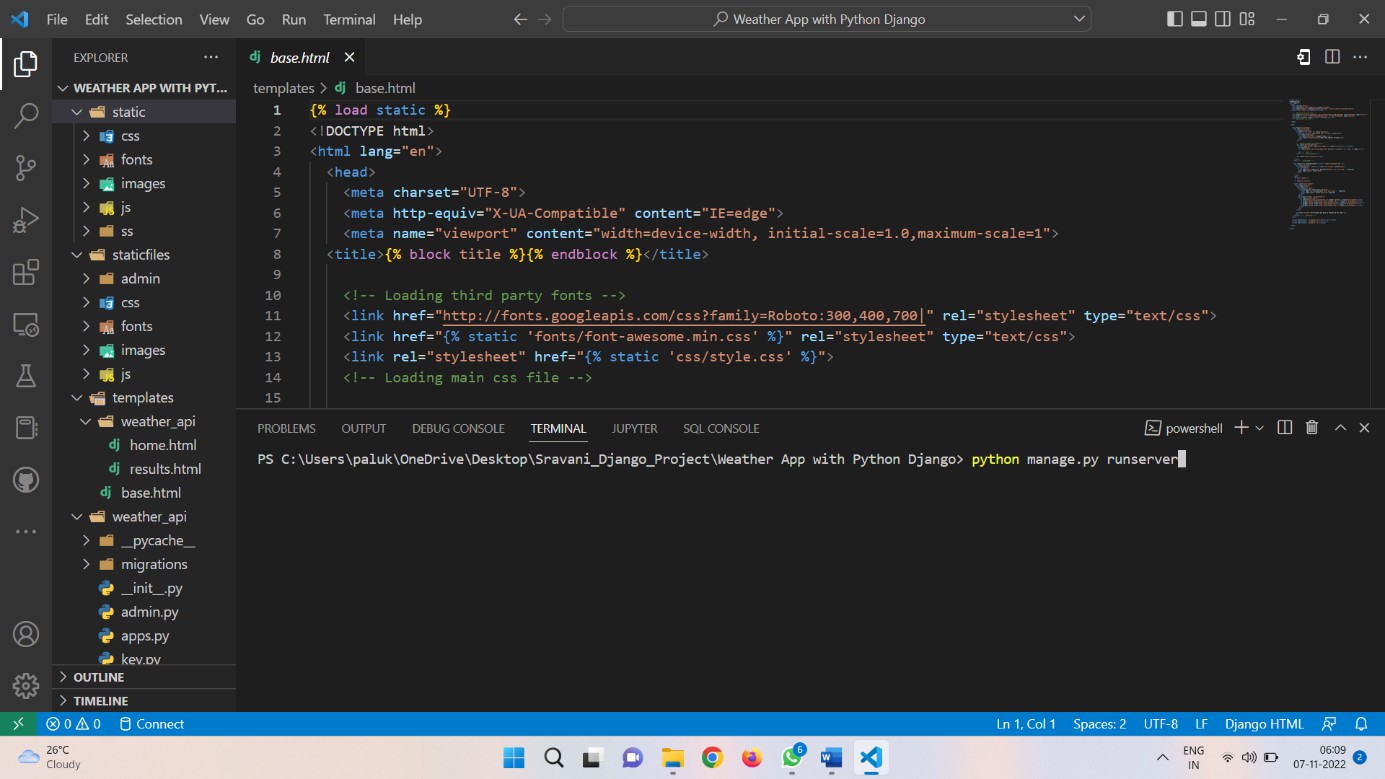
Step 3: Finally, follow the link executed in the Local Host Step 4: It redirects to the web application.

# CHAPTER-5 RESULTS AND CONCLUSIONS

# RESULT

The sathyabama hotel was successfully developed using Django and python this app allows us to book hotel for users and admin can add the new hotel room , manage the prices and can allow users to book there room. This app will work in both desktop and mobile phone. responsive. After completing the entire code we need to open the terminal and run the following command as given below.

# python manage.py runserver



**Fig-5.1**

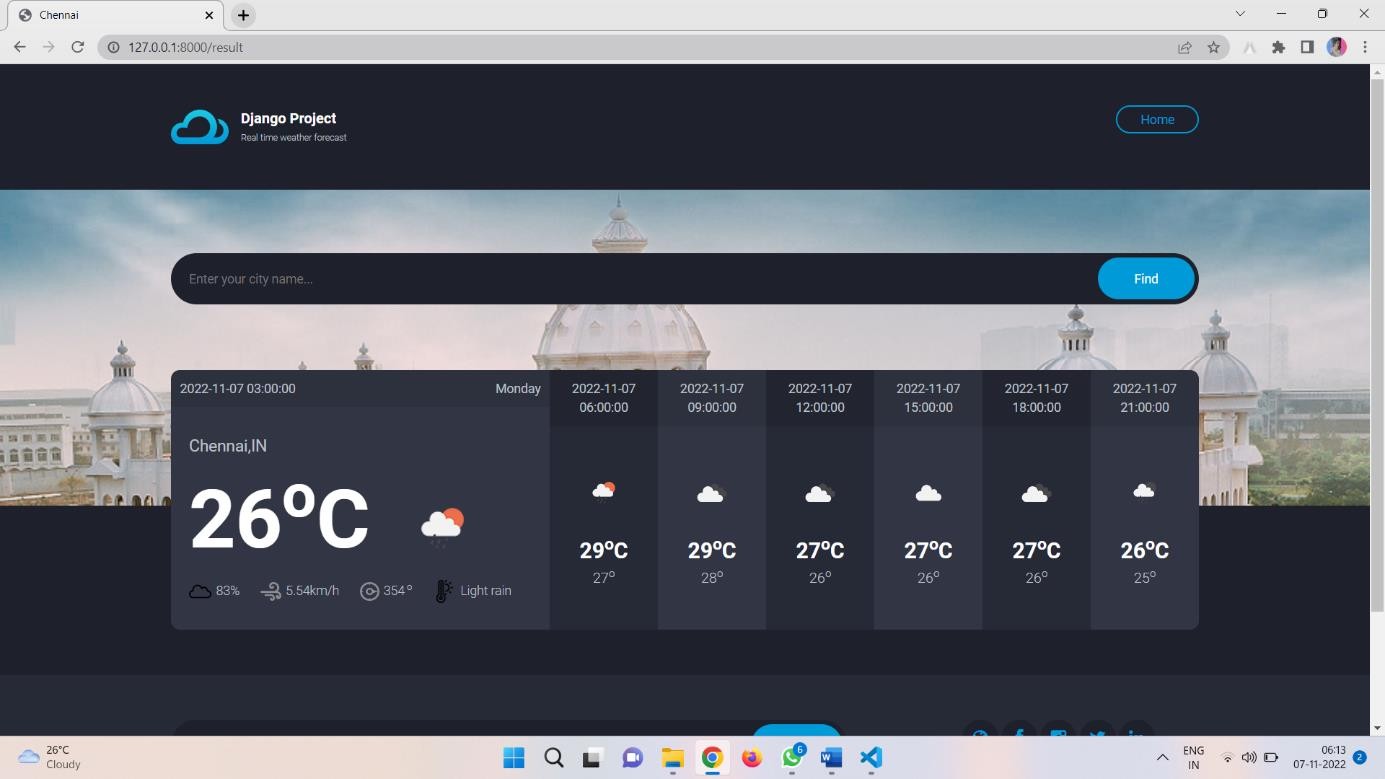
## now the program will be executed and a local host link will be given

**Fig-5.2**

## By opening the link it will direct to the web page we had created

**Fig-5.3**

**Then in the search box we can enter any country/city name to know the temperature of that location (Example:-chennai)**



# CONCLUSION

Fig 5.4

Weather plays a major role in our day to day life so weather app also plays a major role. On an everyday basis, people use weather app to determine what to wear on a given day. Since outdoor activities are severely curtailed by heavy rain, snow and wind chill, forecasts can be used to plan activities around these events, and to plan ahead and survive them. Weather app is something that is very useful in our day to day life.

Weather apps are the most popular way to instantly access forecasts, warnings and other useful weather information on our mobile phones and pc. They provide the public with unprecedented flexibility. Users choose the information they want, how they get it and how it looks. Weather apps enable users to get instant alerts regarding weather conditions. Weather apps are the simplest method to know about the updates of the upcoming weather.

# CHAPTER-6 APPENDIX

**REFERENCES**

# https://[www.w3schools.com/html](http://www.w3schools.com/html)

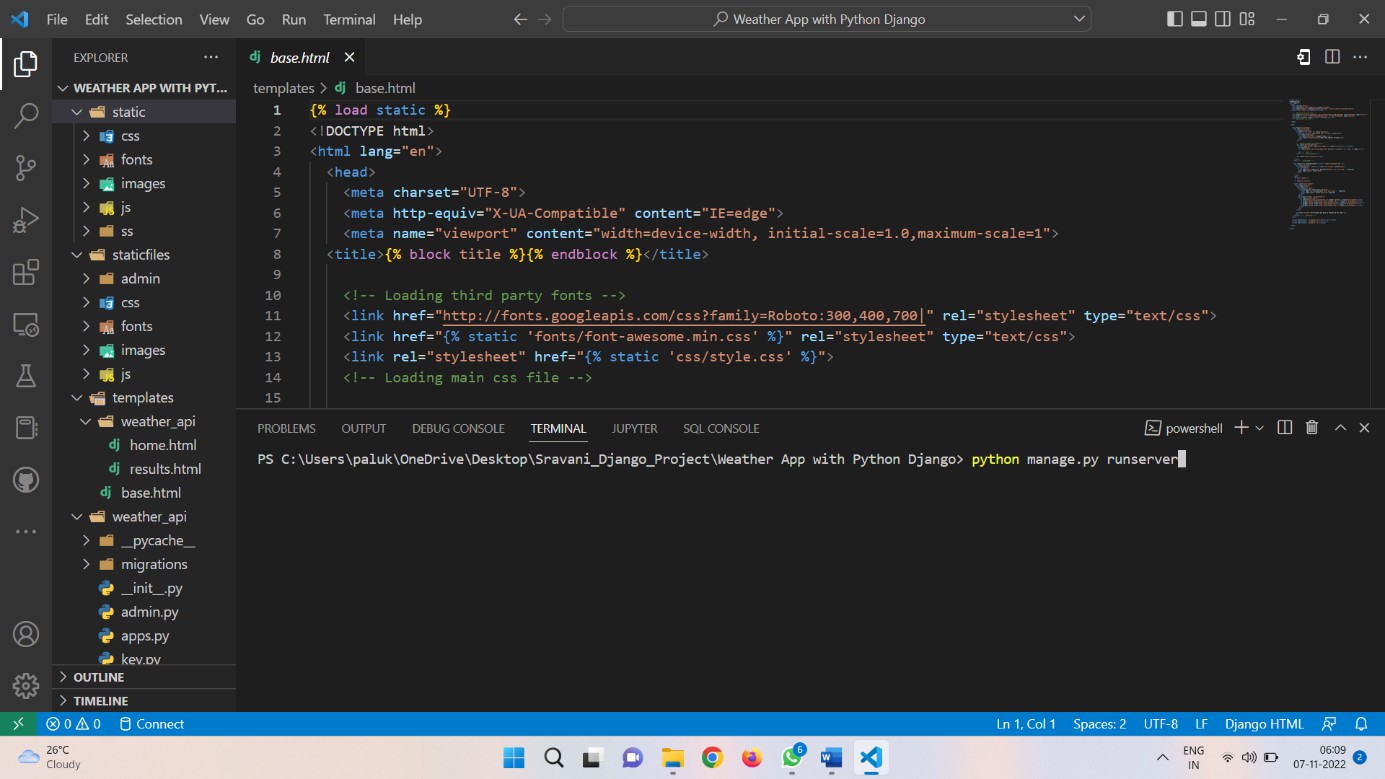
* + - [**https://www.w3.org/style/css**](https://www.w3.org/style/css)

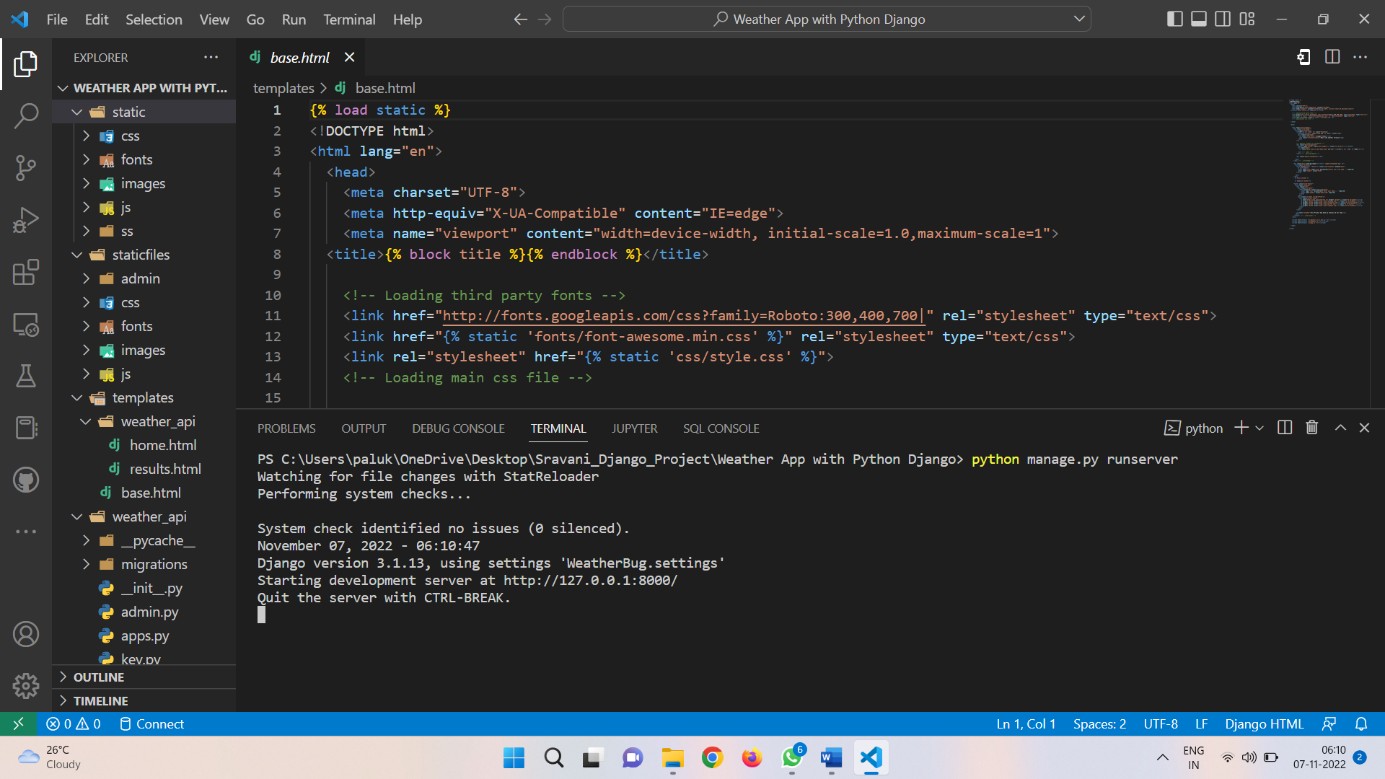
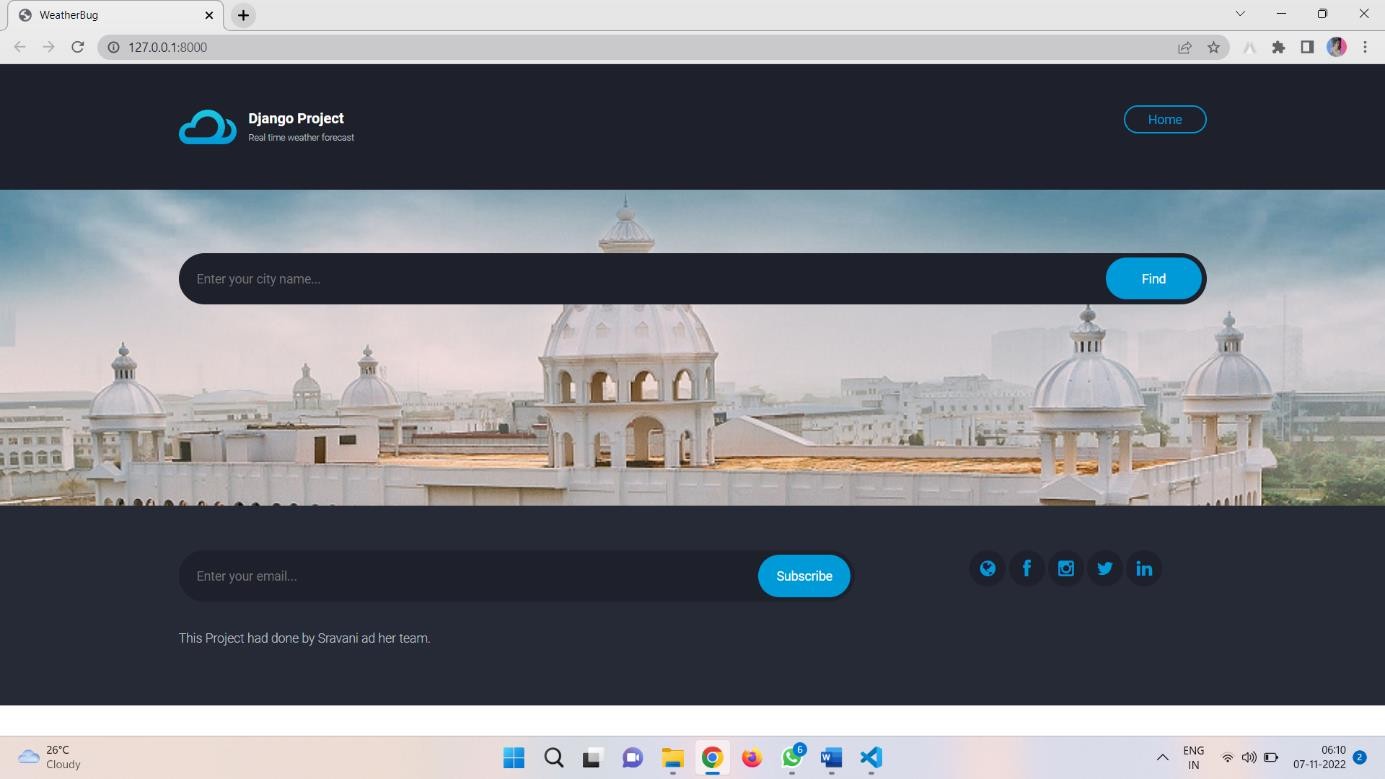
# https://wikepedia.or/wiki/MySQL

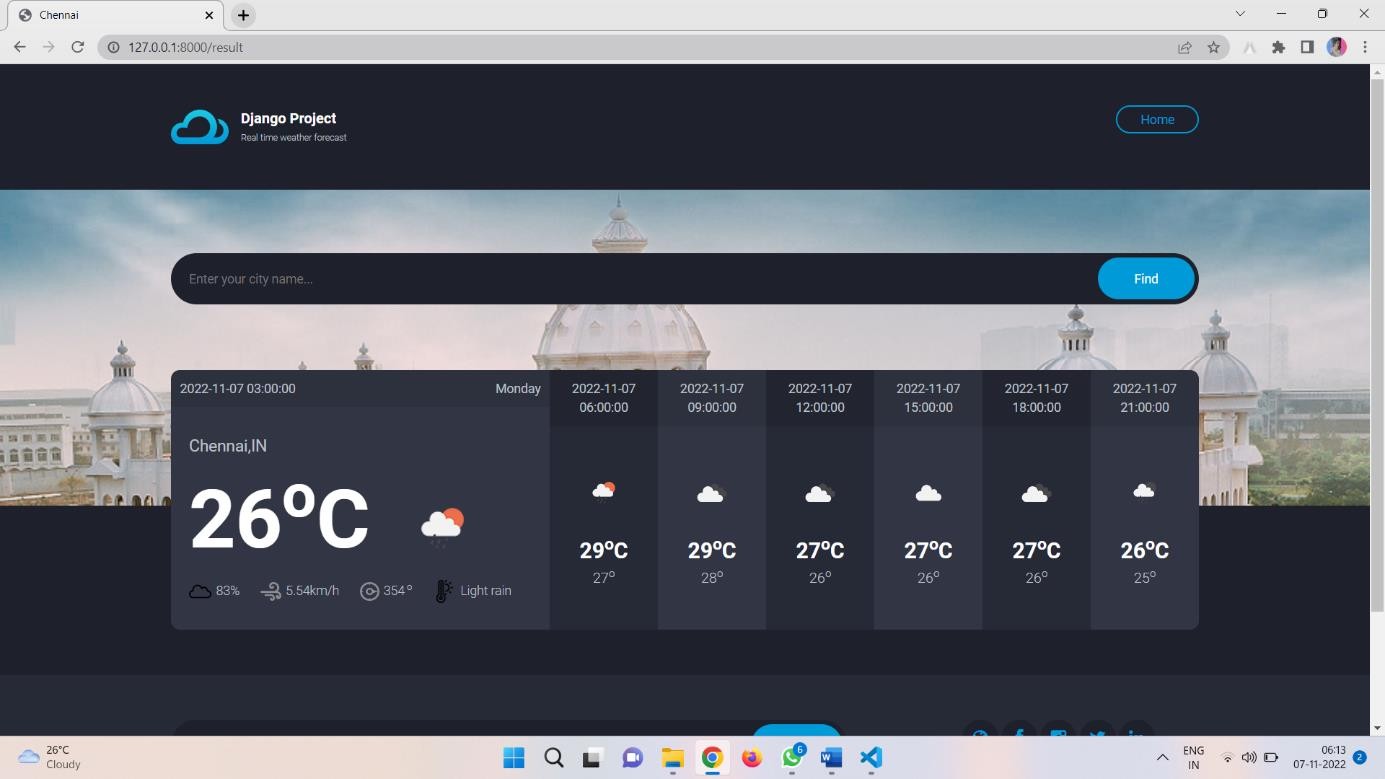
* + - [**https://wikipedia.org/wiki/javascript**](https://wikipedia.org/wiki/javascript)
    - [**https://httpd.apache.org**](https://httpd.apache.org/)
    - [**https://w3schools.com/php**](https://w3schools.com/php)

# Learning PHP, MySQL & JavaScript: With jQuery, CSS & HTML

**SCREENSHOTS**







# COMPLETE SOURCE CODE

1. **MAIN CODE**

from django.shortcuts import render, redirect from weather\_api.key import api\_key

import requests import math

# from .models import Social

# Create your views here.

def index(request):

return render(request, "weather\_api/home.html")

def result(request):

if request.method == "POST":

city\_name = request.POST["city"].lower() url =

f["http://api.openweathermap.org/data/2.5/forecast?q=](http://api.openweathermap.org/data/2.5/forecast?q){city\_name}&appid={api\_key

}"

w\_dataset = requests.get(url).json() try:

context = { ####

"city\_name":w\_dataset["city"]["name"], "city\_country":w\_dataset["city"]["country"], "wind":w\_dataset['list'][0]['wind']['speed'],

"degree":w\_dataset['list'][0]['wind']['deg'],

"status":w\_dataset['list'][0]['weather'][0]['description'],

"cloud":w\_dataset['list'][0]['clouds']['all'],

'date':w\_dataset['list'][0]["dt\_txt"],

'date1':w\_dataset['list'][1]["dt\_txt"],

'date2':w\_dataset['list'][2]["dt\_txt"],

'date3':w\_dataset['list'][3]["dt\_txt"],

'date4':w\_dataset['list'][4]["dt\_txt"],

'date5':w\_dataset['list'][5]["dt\_txt"],

'date6':w\_dataset['list'][6]["dt\_txt"],

] -273.0),

"temp": round(w\_dataset["list"][0]["main"]["temp"] -273.0), "temp\_min1":math.floor(w\_dataset["list"][1]["main"]["temp\_min"

"temp\_max1": math.ceil(w\_dataset["list"][1]["main"]["temp\_max"] -273.0),

"temp\_min2":math.floor(w\_dataset["list"][2]["main"]["temp\_min"

] -273.0),

"temp\_max2":

math.ceil(w\_dataset["list"][2]["main"]["temp\_max"] -273.0), "temp\_min3":math.floor(w\_dataset["list"][3]["main"]["temp\_min"

] -273.0),

"temp\_max3":

math.ceil(w\_dataset["list"][3]["main"]["temp\_max"] -273.0), "temp\_min4":math.floor(w\_dataset["list"][4]["main"]["temp\_min"

] -273.0),

"temp\_max4":

math.ceil(w\_dataset["list"][4]["main"]["temp\_max"] -273.0), "temp\_min5":math.floor(w\_dataset["list"][5]["main"]["temp\_min"

] -273.0),

"temp\_max5":

math.ceil(w\_dataset["list"][5]["main"]["temp\_max"] -273.0), "temp\_min6":math.floor(w\_dataset["list"][6]["main"]["temp\_min"

] -273.0),

"temp\_max6":

math.ceil(w\_dataset["list"][6]["main"]["temp\_max"] -273.0),

"pressure":w\_dataset["list"][0]["main"]["pressure"],

"humidity":w\_dataset["list"][0]["main"]["humidity"],

"sea\_level":w\_dataset["list"][0]["main"]["sea\_level"],

"weather":w\_dataset["list"][1]["weather"][0]["main"], "description":w\_dataset["list"][1]["weather"][0]["description"

],

"icon":w\_dataset["list"][0]["weather"][0]["icon"],

"icon1":w\_dataset["list"][1]["weather"][0]["icon"],

"icon2":w\_dataset["list"][2]["weather"][0]["icon"],

"icon3":w\_dataset["list"][3]["weather"][0]["icon"],

"icon4":w\_dataset["list"][4]["weather"][0]["icon"],

"icon5":w\_dataset["list"][5]["weather"][0]["icon"],

"icon6":w\_dataset["list"][6]["weather"][0]["icon"],

}

except:

context = {

"city\_name":"Not Found, Check your spelling..."

}

return render(request, "weather\_api/results.html", context) else:

return redirect('home')

# def social\_links(request):

# sl = Social.objects.all() # context = {

# 'sl': sl # }

# return render(request, 'weather\_api/base.html', context)

"""

Django settings for WeatherBug project.

Generated by 'django-admin startproject' using Django 3.0.7.

For more information on this file, see https://docs.djangoproject.com/en/3.0/topics/settings/

For the full list of settings and their values, see https://docs.djangoproject.com/en/3.0/ref/settings/ """

import os

# Build paths inside the project like this: os.path.join(BASE\_DIR, ...) BASE\_DIR = os.path.dirname(os.path.dirname(os.path.abspath( file )))

# Quick-start development settings - unsuitable for production

# See https://docs.djangoproject.com/en/3.0/howto/deployment/checklist/

# SECURITY WARNING: keep the secret key used in production secret! SECRET\_KEY = '5esgtomarm#@0!uik@%d\_44^!cd42faruk437w)ef+@g@3m+ek-4l@g\_l6o'

# SECURITY WARNING: don't run with debug turned on in production! DEBUG = True

ALLOWED\_HOSTS = []

# Application definition

INSTALLED\_APPS = [

'django.contrib.admin', 'django.contrib.auth', 'django.contrib.contenttypes', 'django.contrib.sessions', 'django.contrib.messages', 'django.contrib.staticfiles', 'weather\_api'

]

MIDDLEWARE = [

'django.middleware.security.SecurityMiddleware', 'django.contrib.sessions.middleware.SessionMiddleware', 'django.middleware.common.CommonMiddleware', 'django.middleware.csrf.CsrfViewMiddleware', 'django.contrib.auth.middleware.AuthenticationMiddleware', 'django.contrib.messages.middleware.MessageMiddleware', 'django.middleware.clickjacking.XFrameOptionsMiddleware',

]

ROOT\_URLCONF = 'WeatherBug.urls'

TEMPLATES = [

{

'BACKEND': 'django.template.backends.django.DjangoTemplates', 'DIRS': [os.path.join(BASE\_DIR, 'templates')],

'APP\_DIRS': True, 'OPTIONS': {

'context\_processors': [ 'django.template.context\_processors.debug', 'django.template.context\_processors.request', 'django.contrib.auth.context\_processors.auth', 'django.contrib.messages.context\_processors.messages',

],

},

},

]

WSGI\_APPLICATION = 'WeatherBug.wsgi.application'

# Database

# https://docs.djangoproject.com/en/3.0/ref/settings/#databases

DATABASES = {

'default': {

'ENGINE': 'django.db.backends.sqlite3',

'NAME': os.path.join(BASE\_DIR, 'db.sqlite3'),

}

}

# Password validation

# https://docs.djangoproject.com/en/3.0/ref/settings/#auth-password-validators

AUTH\_PASSWORD\_VALIDATORS = [

{

'NAME':

'django.contrib.auth.password\_validation.UserAttributeSimilarityValidator',

},

{

'NAME':

'django.contrib.auth.password\_validation.MinimumLengthValidator',

},

{

'NAME':

'django.contrib.auth.password\_validation.CommonPasswordValidator',

},

{

'NAME':

'django.contrib.auth.password\_validation.NumericPasswordValidator',

},

]

# Internationalization

# https://docs.djangoproject.com/en/3.0/topics/i18n/

LANGUAGE\_CODE = 'en-us'

TIME\_ZONE = 'Asia/Dhaka'

USE\_I18N = True

USE\_L10N = True

USE\_TZ = True

# Static files (CSS, JavaScript, Images)

# https://docs.djangoproject.com/en/3.0/howto/static-files/

STATIC\_ROOT = os.path.join(BASE\_DIR, "staticfiles")

STATIC\_URL = '/static/'

STATICFILES\_DIRS = [os.path.join(BASE\_DIR, "static")]

## TEMPLATE CODE

* 1. **Home html**

{% extends 'base.html' %}

{% block title %} WeatherBug {% endblock title %}

{% block content %}

{% endblock content %}

## Base html

{% load static %}

<!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,maximum-scale=1">

<title>{% block title %}{% endblock %}</title>

<!-- Loading third party fonts -->

<link href="[http://fonts.googleapis.com/css?family=Roboto:300,400,700](http://fonts.googleapis.com/css?family=Roboto%3A300%2C400%2C700)|" rel="stylesheet" type="text/css">

<link href="{% static 'fonts/font-awesome.min.css' %}" rel="stylesheet" type="text/css">

<link rel="stylesheet" href="{% static 'css/style.css' %}">

<!-- Loading main css file -->

</head>

<body>

<div class="site-content">

<div class="site-header">

<div class="container">

<a href="{% url 'home' %}" class="branding">

<img src="{%static 'images/logo.png' %}" alt="" class="logo">

<div class="logo-type">

<h1 class="site-title"> Django Project </h1>

<big class="site-description">Real time weather forecast</big>

</div>

</a>

<!-- Default snippet for navigation -->

<div class="main-navigation">

<button type="button" class="menu-toggle"><i class="fa fa- bars"></i></button>

<ul class="menu">

<li class="menu-item current-menu-item" id="omar"><a href="{% url 'home' %}">Home</a></li>

</ul> <!-- .menu -->

</div> <!-- .main-navigation -->

<div class="mobile-navigation"></div>

</div>

</div> <!-- .site-header -->

<div class="hero" data-bg-image="{% static 'images/sathyabama.png' %}">

<div class="container">

<form action="{%url 'result'%}" class="find-location" method="post">

{% csrf\_token%}

<input type="text" name="city" placeholder="Enter your city name..." required>

<input type="submit" value="Find">

</form>

</div>

</div>

{% block content %}

{% endblock content %}

<footer class="site-footer">

<div class="container">

<div class="row">

<div class="col-md-8">

<form action="#" class="subscribe-form">

<input type="text" placeholder="Enter your email..." required>

<input type="submit" value="Subscribe" required>

</form>

</div>

<div class="col-md-3 col-md-offset-1">

<div class="social-links">

<a href=["https://w](http://www.projectworlds.in/)ww.proje[ctworlds.in"](http://www.projectworlds.in/) target="\_blank"><i class="fa fa-globe"></i></a>

<a target="\_blank" href="https://[www.facebook.com/](http://www.facebook.com/)"><i class="fa fa-facebook"></i></a>

<a target="\_blank" href="https://[www.instagram.com/](http://www.instagram.com/)"><i class="fa fa-instagram"></i></a>

<a target="\_blank" href="https://[www.twitter.com/](http://www.twitter.com/)"><i class="fa fa-twitter"></i></a>

<a target="\_blank" href="https://[www.linkedin.com/](http://www.linkedin.com/)"><i class="fa fa-linkedin"></i></a>

</div>

</div>

</div>

<p class="colophon">This Project had done by Sravani ad her

team.</p>

</div>

</footer> <!-- .site-footer -->

</div>

<script src="{%static 'js/jquery-1.11.1.min.js' %}"></script>

<script src="{%static 'js/plugins.js' %}"></script>

<script src="{%static 'js/app.js' %}"></script>

</body>

</html>

## Result html

{% extends 'base.html' %}

{% load static %}

{% block title %}{{city\_name}} {% endblock title %}

{% block content %}

<div class="forecast-table">

<div class="container">

<div class="forecast-container">

<div class="today forecast">

<div class="forecast-header">

<div class="day">{{date}}</div>

<div class="date">{% now "l" %}</div>

</div> <!-- .forecast-header -->

<div class="forecast-content">

<div class="location">{{city\_name}},{{city\_country}}</div>

<div class="degree">

<div class="num">{{temp}}<sup>o</sup>C</div>

<div class="forecast-icon">

<img src=["http://openweathermap.org/img/wn/](http://openweathermap.org/img/wn/){{icon}}@2x.png" alt="" width=90>

</div>

</div>

<span><img style="height: 25px; width: 25px" src="{%static 'images/cloud.png' %}"

alt="">{{cloud}}%</span>

<span><img src="{%static 'images/icon-wind.png' %}" alt="">{{wind}}km/h</span>

<span><img src="{%static 'images/icon-compass.png' %}" alt="degree">{{degree}}<sup> o</sup></span>

<span><img style="height: 25px; width: 25px;" src="{%static 'images/hot.png' %}"

alt="status">{{status|capfirst }}</span>

</div>

</div>

<div class="forecast">

<div class="forecast-header">

<div class="day">{{date1}}</div>

</div> <!-- .forecast-header -->

<div class="forecast-content">

<div class="forecast-icon">

<img src=["http://openweathermap.org/img/wn/](http://openweathermap.org/img/wn/){{icon1}}@2x.png" alt="" width=48>

</div>

<div class="degree">{{temp\_max1}}<sup>o</sup>C</div>

<small>{{temp\_min1}}<sup>o</sup></small>

</div>

</div>

<div class="forecast">

<div class="forecast-header">

<div class="day">{{date2}}</div>

</div> <!-- .forecast-header -->

<div class="forecast-content">

<div class="forecast-icon">

<img src=["http://openweathermap.org/img/wn/](http://openweathermap.org/img/wn/){{icon2}}@2x.png" alt="" width=48>

</div>

<div class="degree">{{temp\_max2}}<sup>o</sup>C</div>

<small>{{temp\_min2}}<sup>o</sup></small>

</div>

</div>

<div class="forecast">

<div class="forecast-header">

<div class="day">{{date3}}</div>

</div> <!-- .forecast-header -->

<div class="forecast-content">

<div class="forecast-icon">

<img src=["http://openweathermap.org/img/wn/](http://openweathermap.org/img/wn/){{icon3}}@2x.png" alt="" width=48>

</div>

<div class="degree">{{temp\_max3}}<sup>o</sup>C</div>

<small>{{temp\_min3}}<sup>o</sup></small>

</div>

</div>

<div class="forecast">

<div class="forecast-header">

<div class="day">{{date4}}</div>

</div> <!-- .forecast-header -->

<div class="forecast-content">

<div class="forecast-icon">

<img src=["http://openweathermap.org/img/wn/](http://openweathermap.org/img/wn/){{icon4}}@2x.png" alt="" width=48>

</div>

<div class="degree">{{temp\_max4}}<sup>o</sup>C</div>

<small>{{temp\_min4}}<sup>o</sup></small>

</div>

</div>

<div class="forecast">

<div class="forecast-header">

<div class="day">{{date5}}</div>

</div> <!-- .forecast-header -->

<div class="forecast-content">

<div class="forecast-icon">

<img src=["http://openweathermap.org/img/wn/](http://openweathermap.org/img/wn/){{icon5}}@2x.png" alt="" width=48>

</div>

<div class="degree">{{temp\_max5}}<sup>o</sup>C</div>

<small>{{temp\_min5}}<sup>o</sup></small>

</div>

</div>

<div class="forecast">

<div class="forecast-header">

<div class="day">{{date6}}</div>

</div> <!-- .forecast-header -->

<div class="forecast-content"SYSTEM

SYSTEM ARCHITECTURE :

