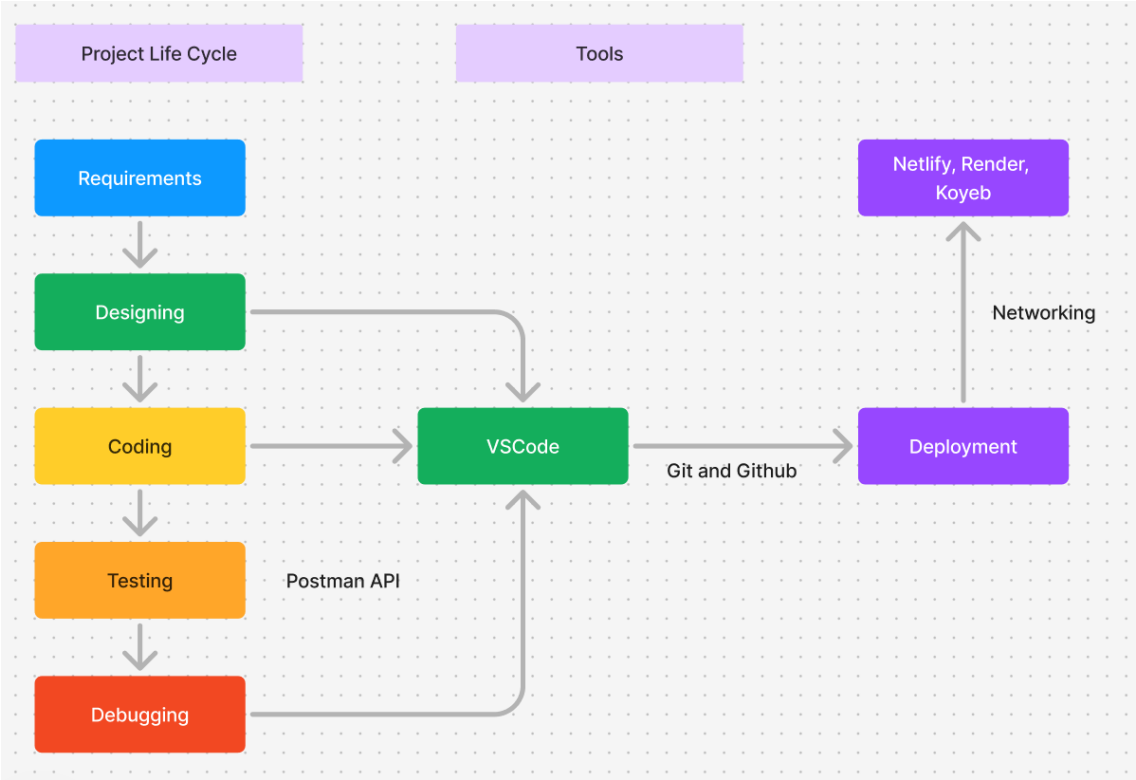
Day 1: Project and Product

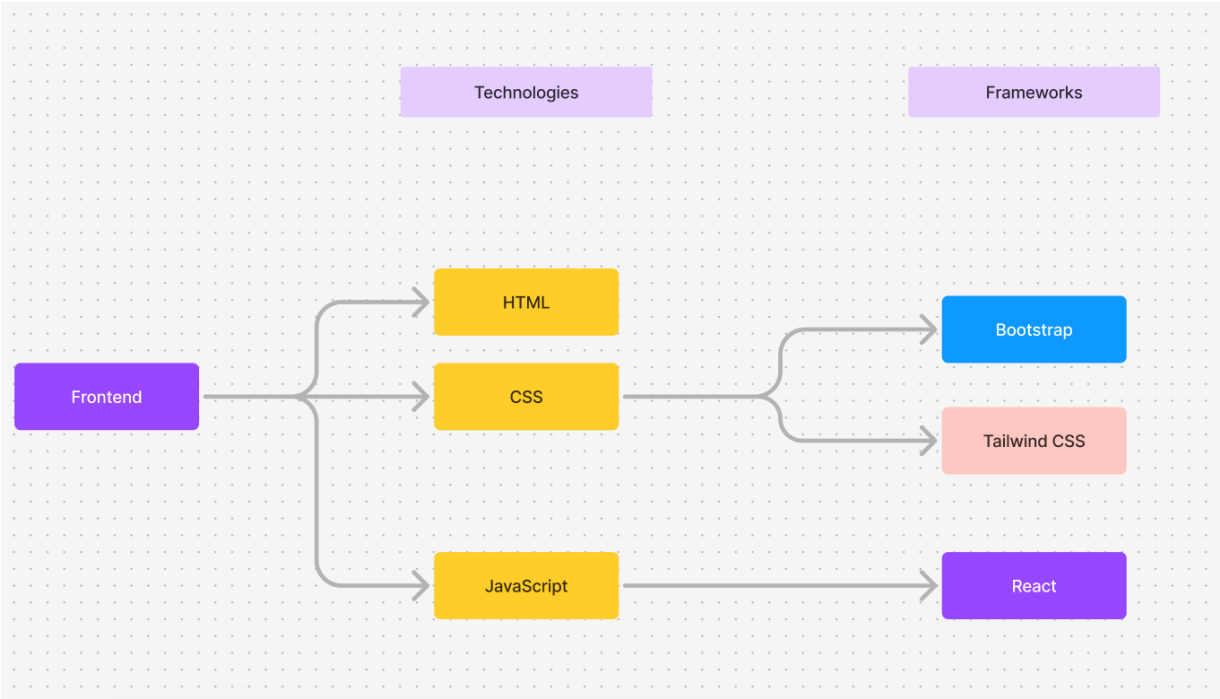
Project is a software under development and will become product once if it is completely developed.

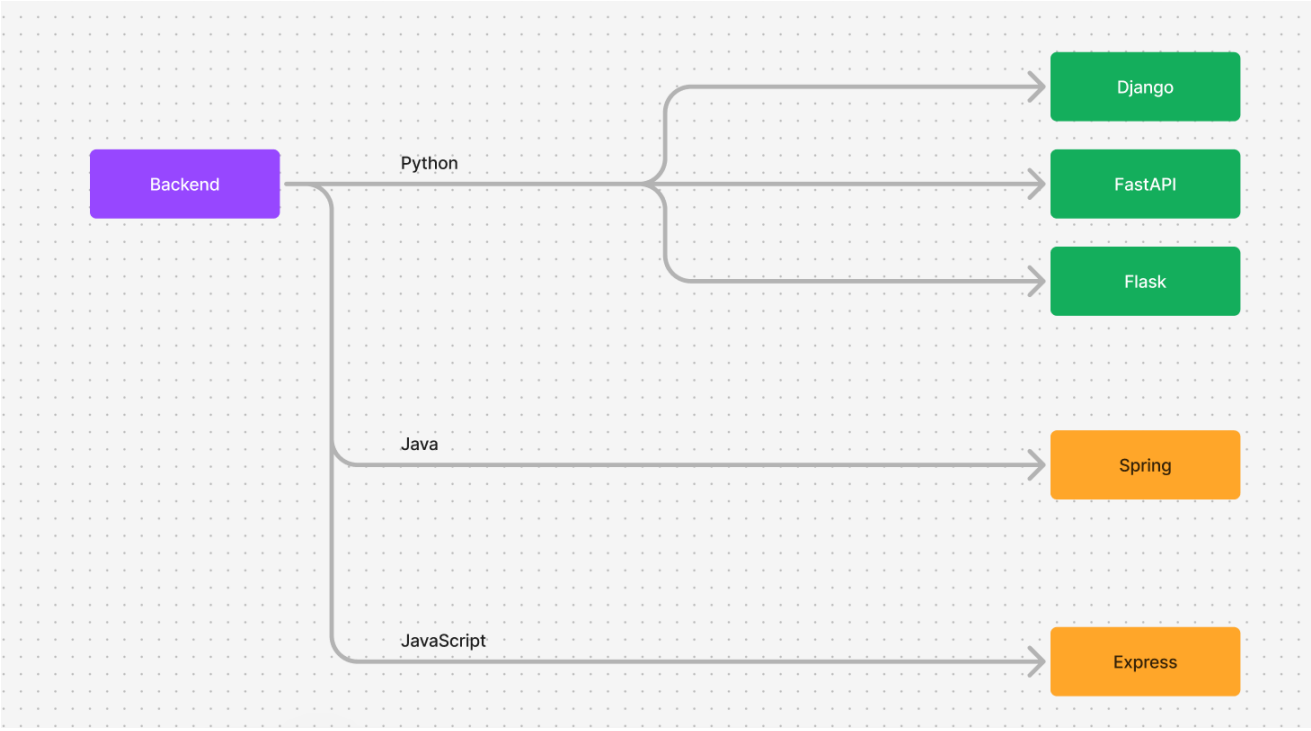
E.g.: Instagram – Product, its clone is a Project

Project Life Cycle

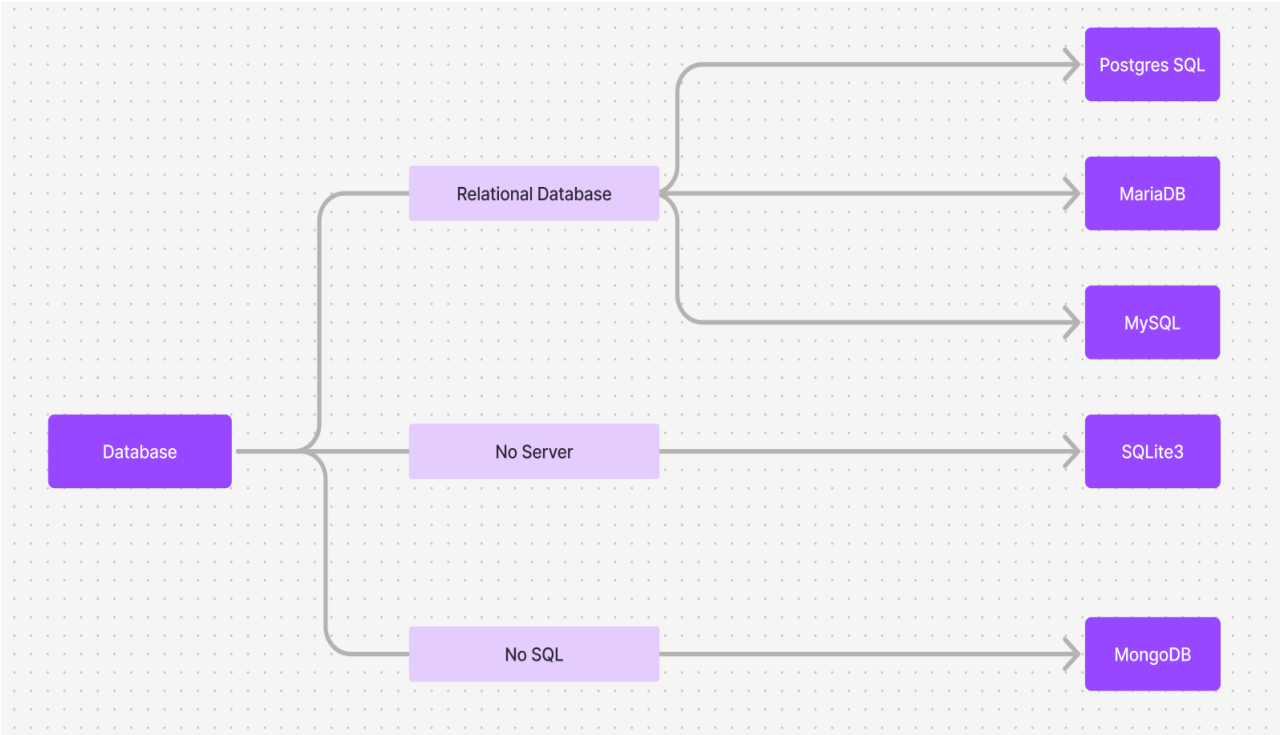


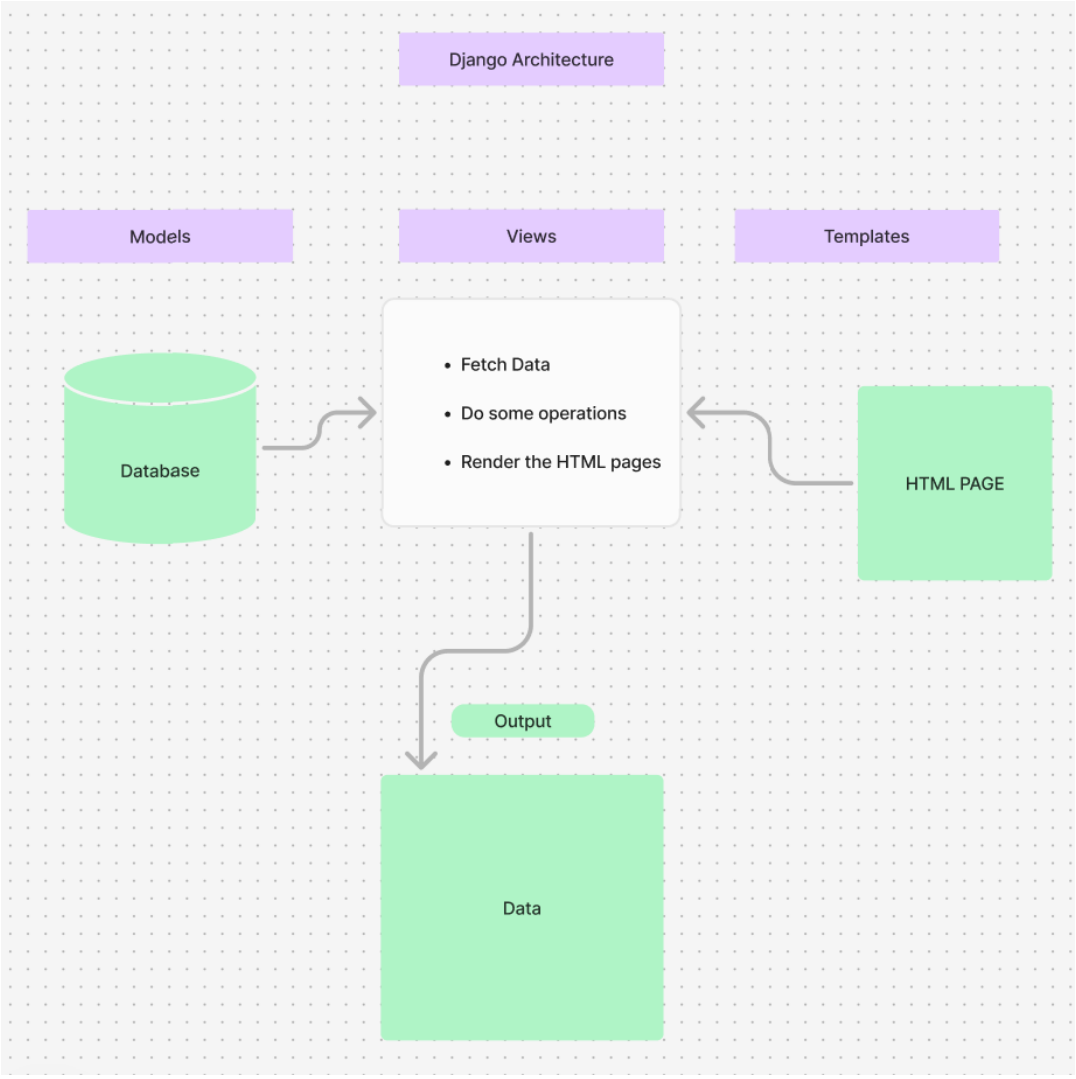
John@pentagonspace.in





John@pentagonspace.in





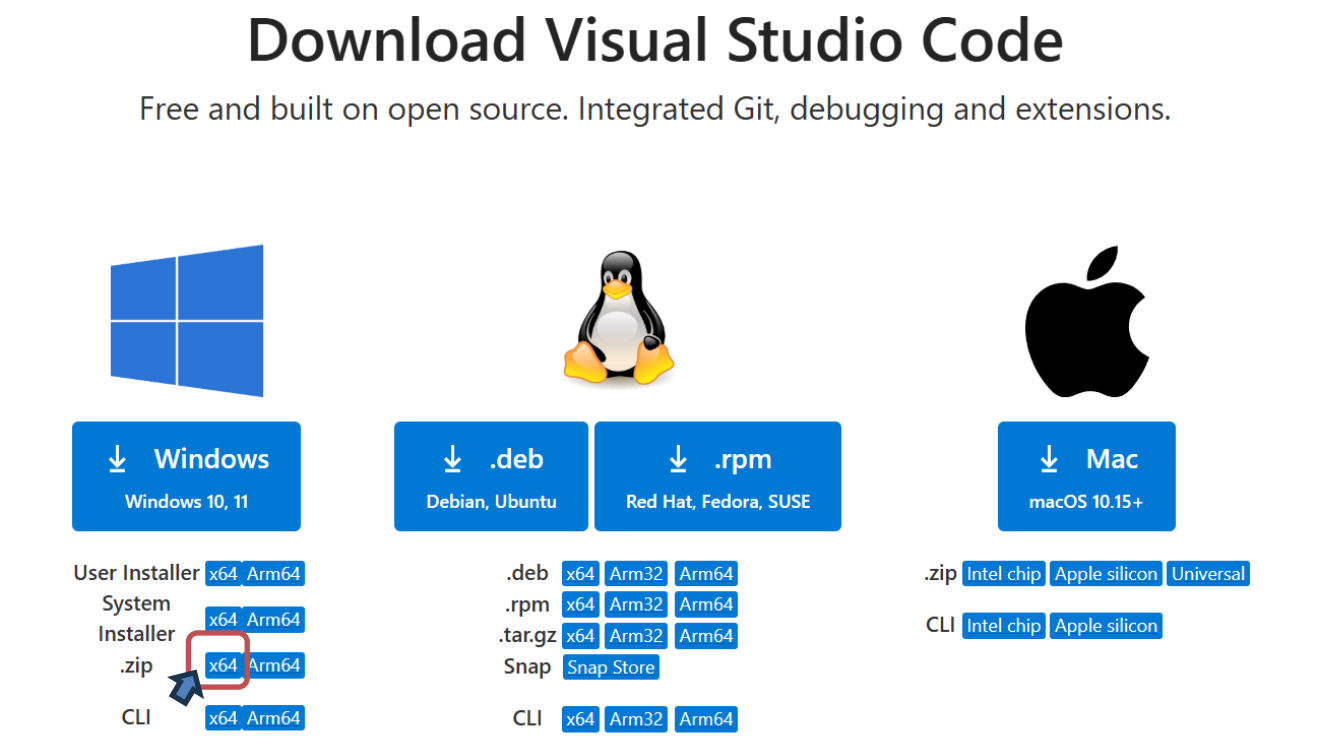
John@pentagonspace.in

Project 1 Weather App

Installation

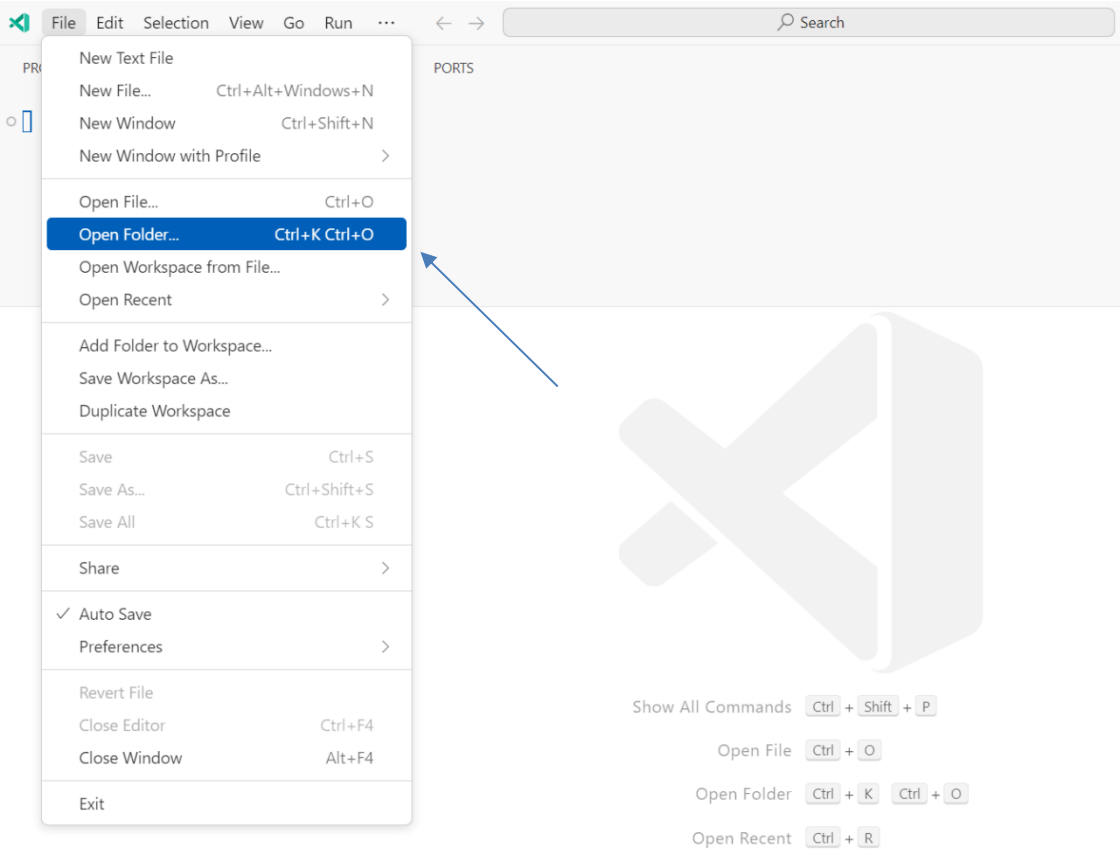
Tailwind CLI | Using PostCSS | Framework Guides | Play CDN

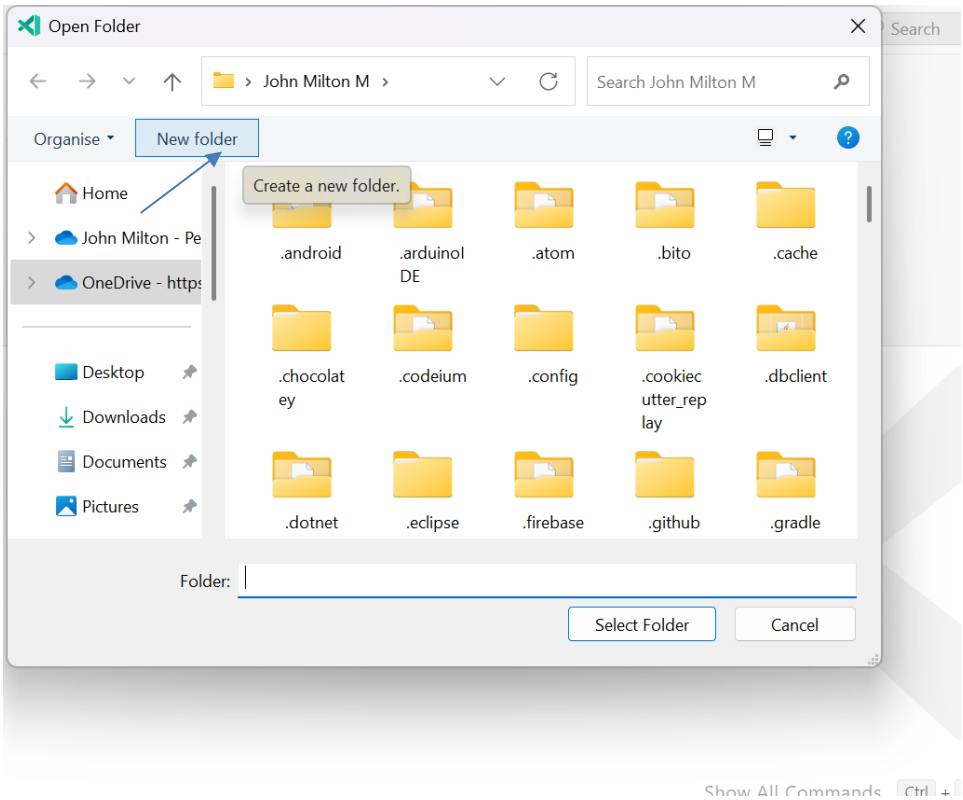
The simplest and fastest way to get up and running with Tailwind CSS from scratch is with the Tailwind CLI tool. The CLI is also available as a **standalone executable** if you want to use it without installing Node.js.



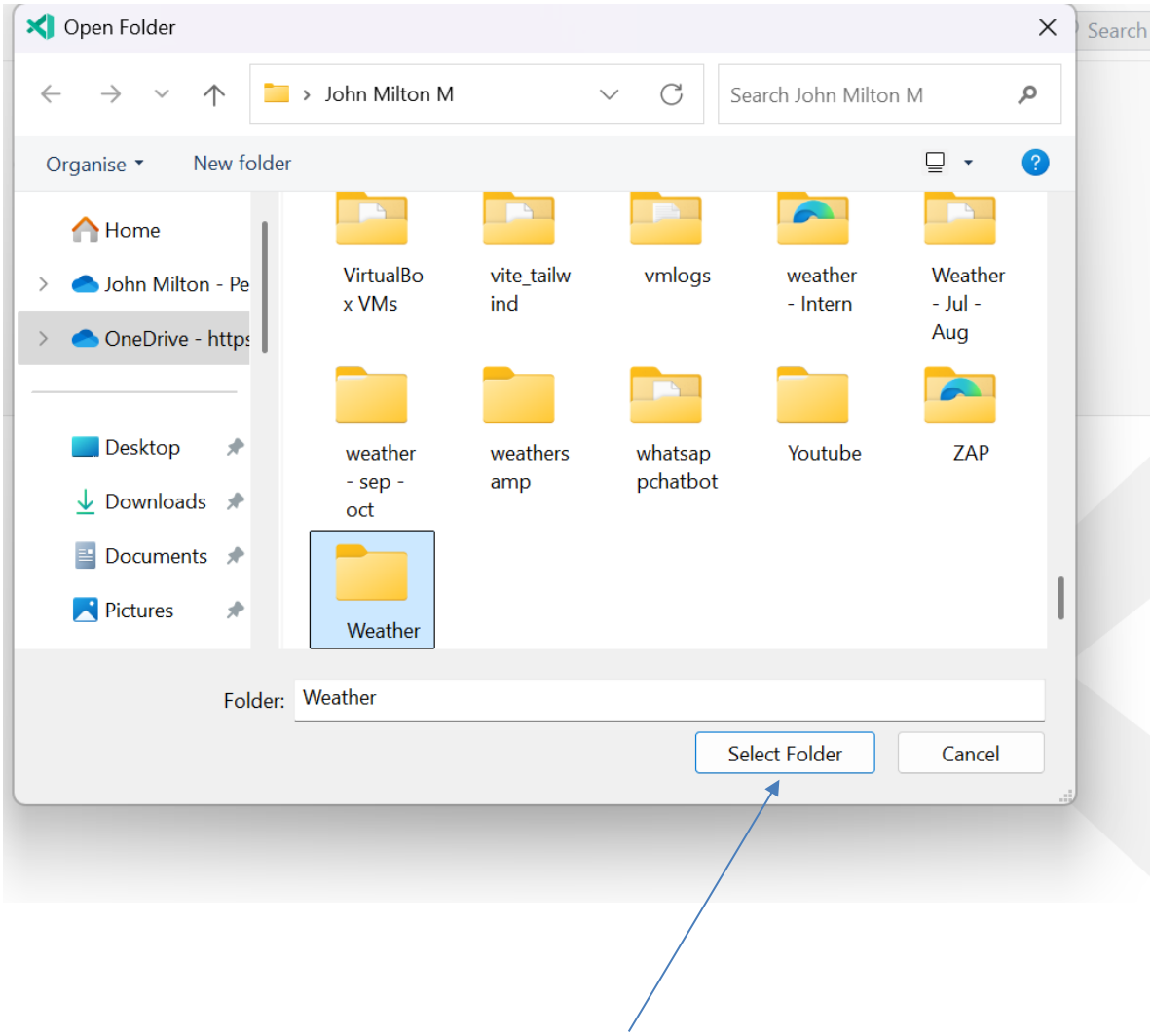
1. Install VsCode
2. Open New window (Ctrl + Shift + N)
3. File -> Open Folder

John@pentagonspace.in



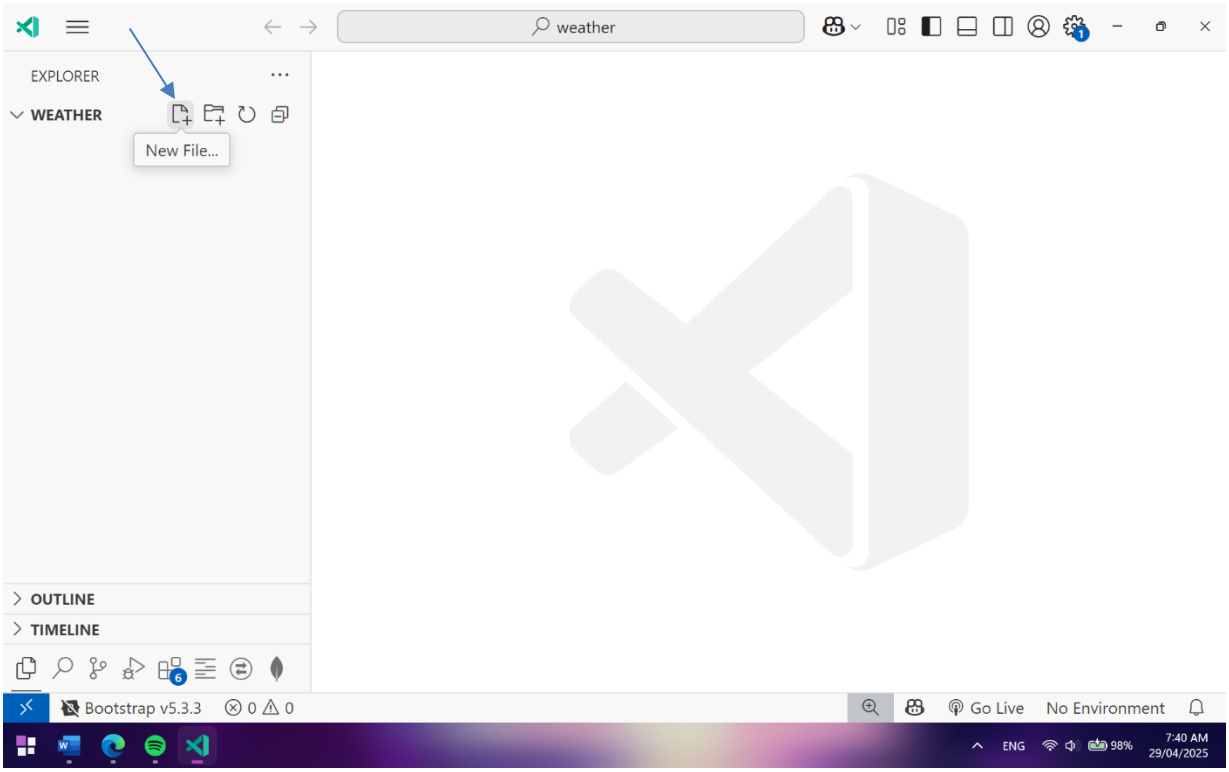


John@pentagonspace.in

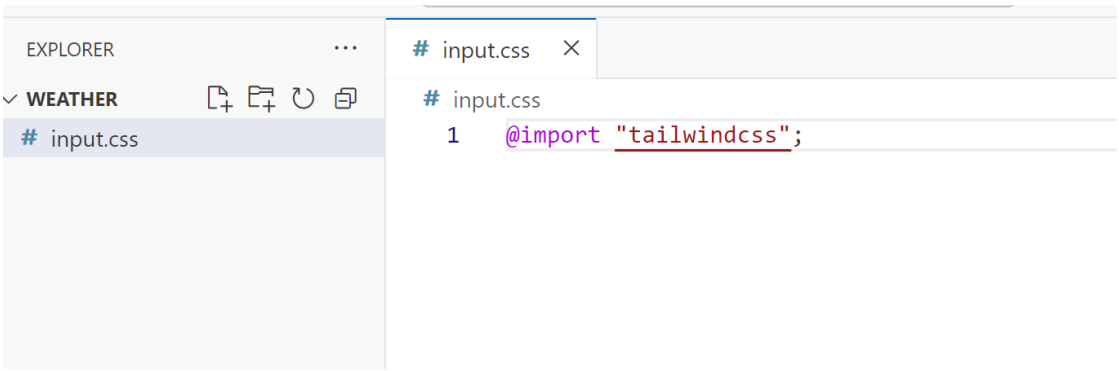


4. Click on `New Folder,` give name as `weather`
5. Click `Select Folder`

John@pentagonspace.in



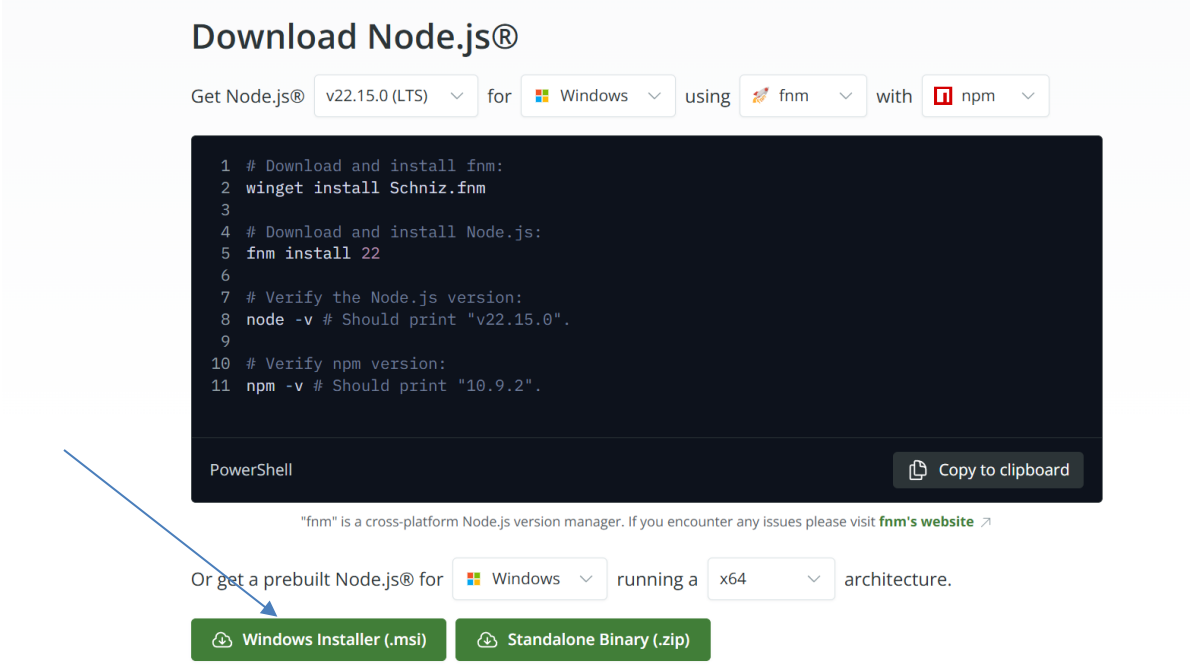
6. Click on `New File`, give name as input.css




```
@import "tailwindcss";
```


John@pentagonspace.in

7. Make Sure you have installed with [Node.js]



8. Go to Vscode and open Integrated Terminal (Ctrl + J)

Type in the terminal and press enter:

```
npm install tailwindcss @tailwindcss/cli
```

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

npm install tailwindcss

up to date, audited 335 packages in 12s

53 packages are looking for funding
  run `npm fund` for details

12 moderate severity vulnerabilities

To address all issues, run:
  npm audit fix

Run `npm audit` for details.
```

9. Run the Tailwindcss Watch mode in background (Necessary to run whenever you are reopening the vscode again)

```
PROBLEMS    OUTPUT    DEBUG CONSOLE    TERMINAL    PORTS

npx @tailwindcss/cli -i input.css -o style.css --watch
```

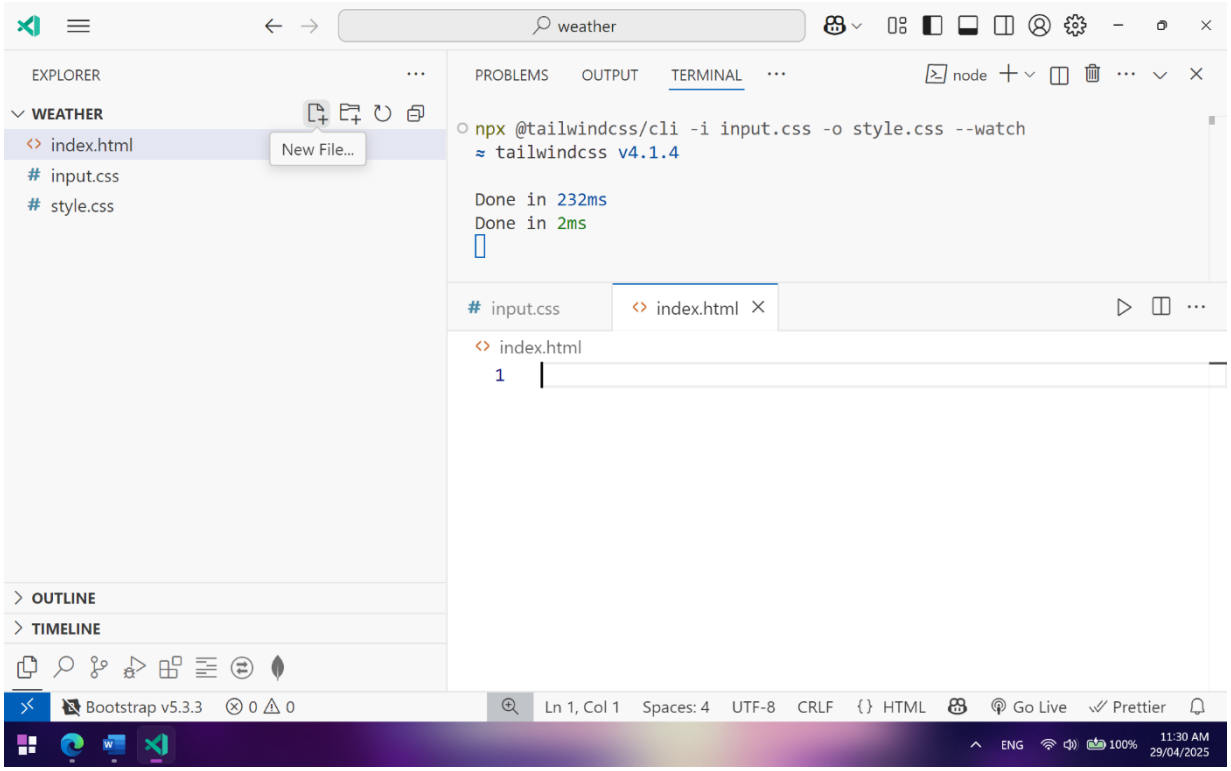
Type in the terminal and press enter:

```
npx @tailwindcss/cli -i input.css -o style.css --watch
```

10. Now we need to create index.html and link style.css

John@pentagonspace.in

Steps to create boiler plate code and linking `style.css` :



Let the `npx tailwindcss -i input.css -o style.css -w` be running in the background and it will write the style.css for you by watching the index.html if any predefined utility class name is there

1. Press shift key and 1 (shift + 1) to get exclamatory symbol (!) and press enter, this will generate the boiler plate code.
2. Change the title from `Document` to `weather App`
3. Inside the `head` block, type `link:css` and press enter

John@pentagonspace.in

```
npx @tailwindcss/cli -i input.css -o style.css -w
Done in 1ms
Done in 1ms
Done in 1ms
Done in 419µs
Done in 795µs
```

```
<!
```

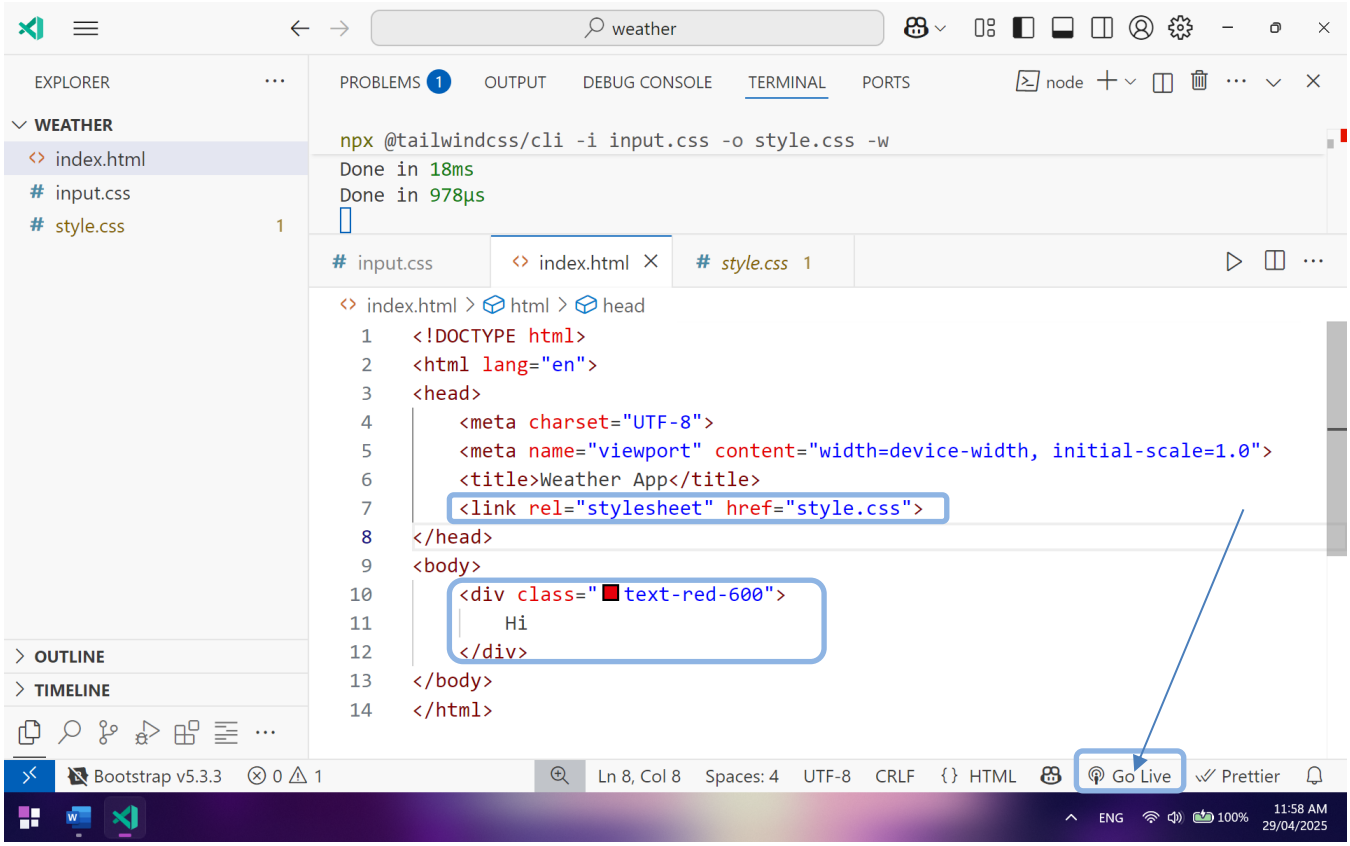
```
Emmet Abbreviation

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    |
</body>
```

```
npx @tailwindcss/cli -i input.css -o style.css -w
Done in 795µs
Done in 30ms
Done in 1ms
Done in 13ms
Done in 1ms
```

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Weather App</title>
</head>
<body>

</body>
</html>
```

Now for testing the tailwindcss, inside the `body` block, we need to create a `div` block with the class name `text-red-600` with some content

1. Inside `body` block type `.text-red-600` and press enter
2. Now it has been created one `div` block with class name `text-red-600`
3. Give Content `Hi` and turn on the live server (Need to install Extension)



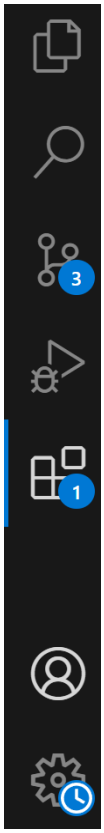
4. Click on `Go Live`

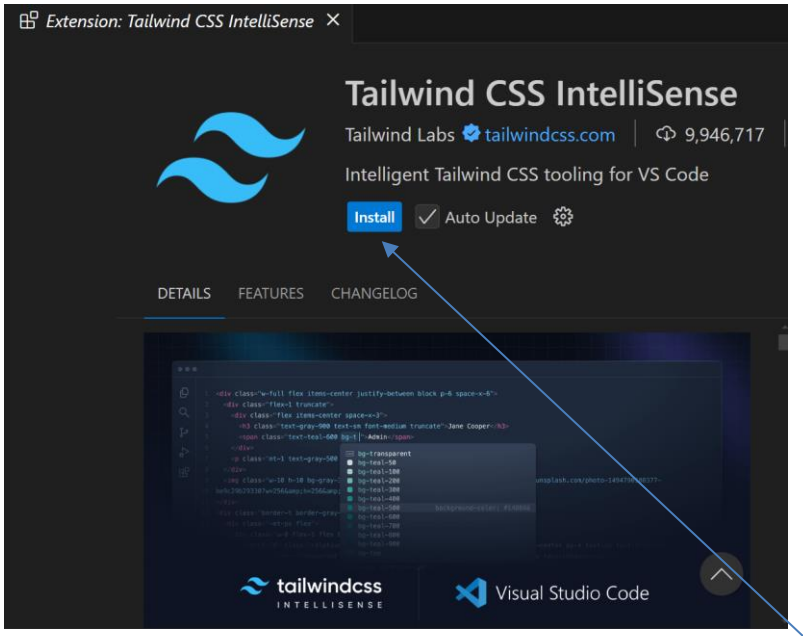
For color and suggestions, we need to install `tailwindcss intellisense` extension.

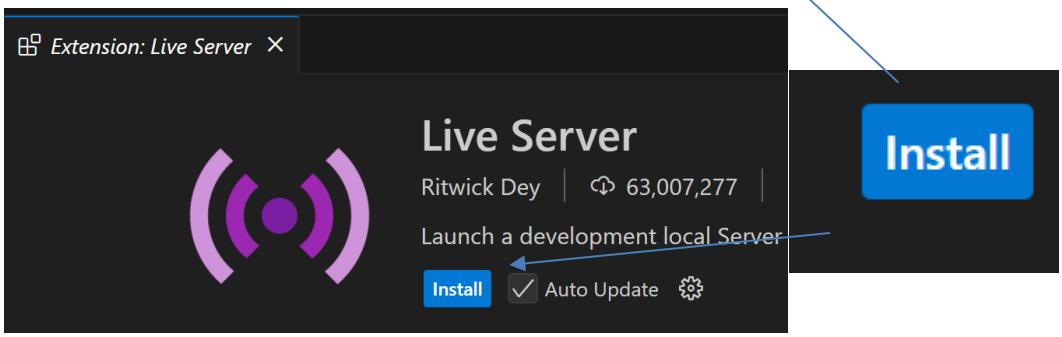
John@pentagonspace.in

Extension Installation Guide:

1. Click on Extension Market Place  (It will be in this row ->)
2. Search for
   - Tailwindcss intellisense
   - Live server
3. Click on install.

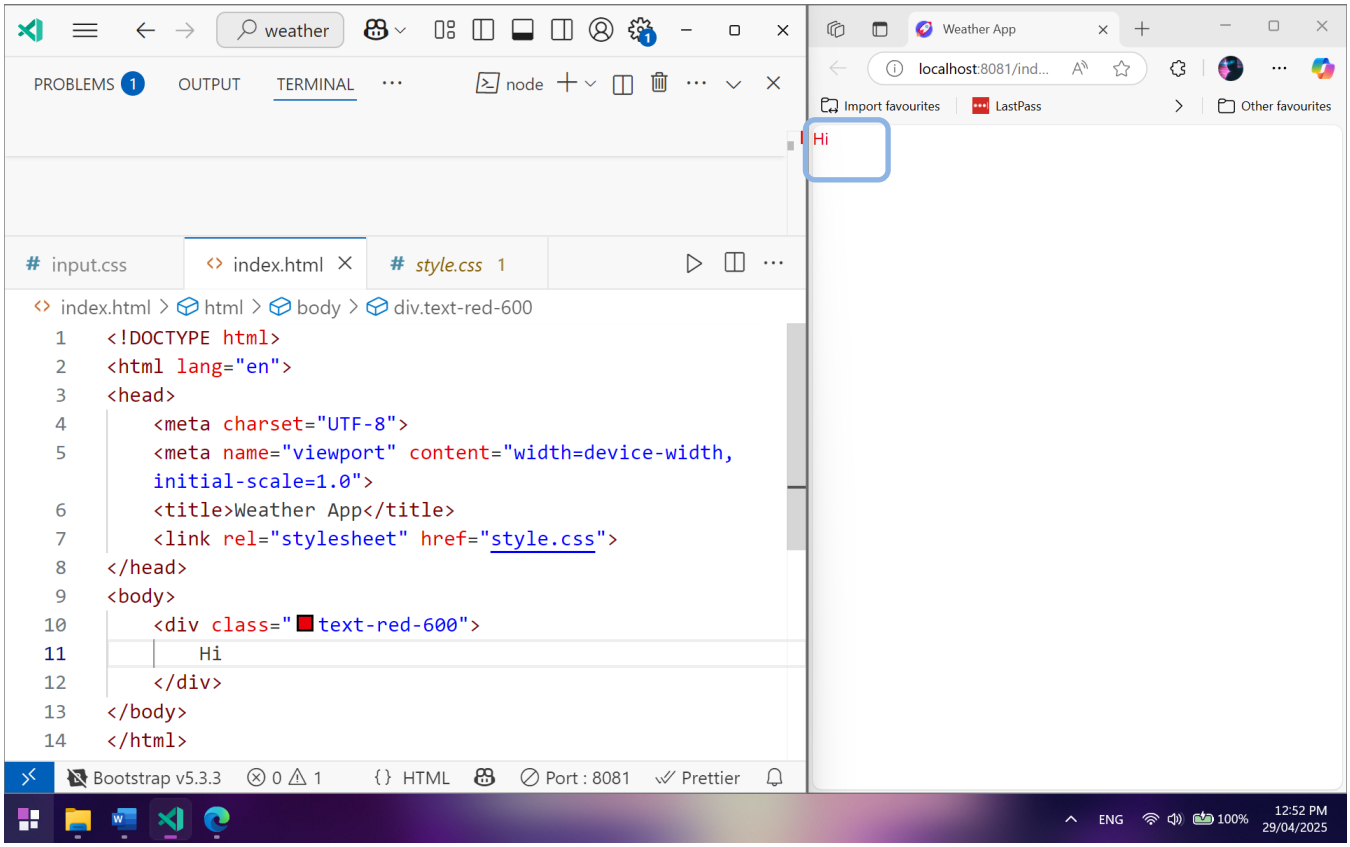






John@pentagonspace.in

Output:



If you can able to see the text content in red color, Voila, you have successfully configured frontend for you project !!!

John@pentagonspace.in

Day 2: Weather App – Frontend Part -1, 2, 3

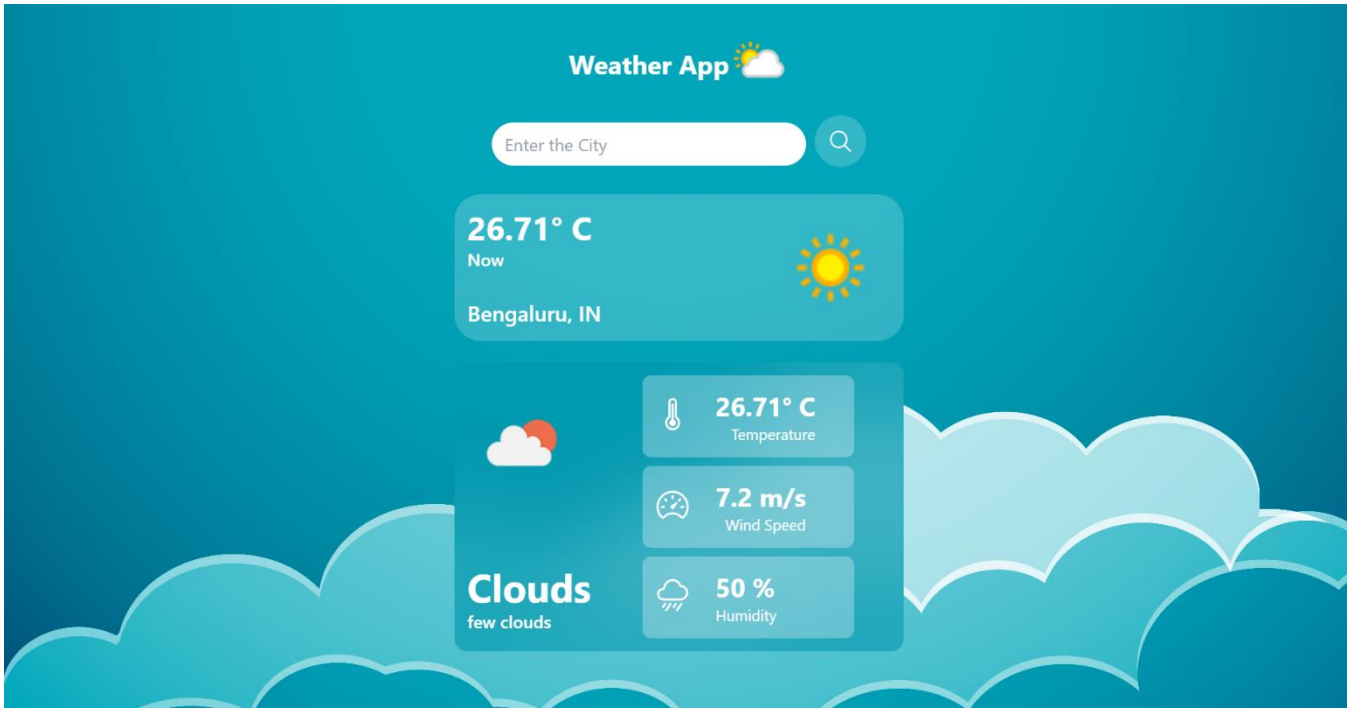
Now, we just need to Design the Frontend part.

Here we are going to learn CSS flex model easily with the help of CSS Framework (TailwindCSS)

As per our project life cycle, initially we need to gather the requirements.

Requirement:

We want to design the page like this,



Now, lets divide the design part by part. And we have 4 blocks.

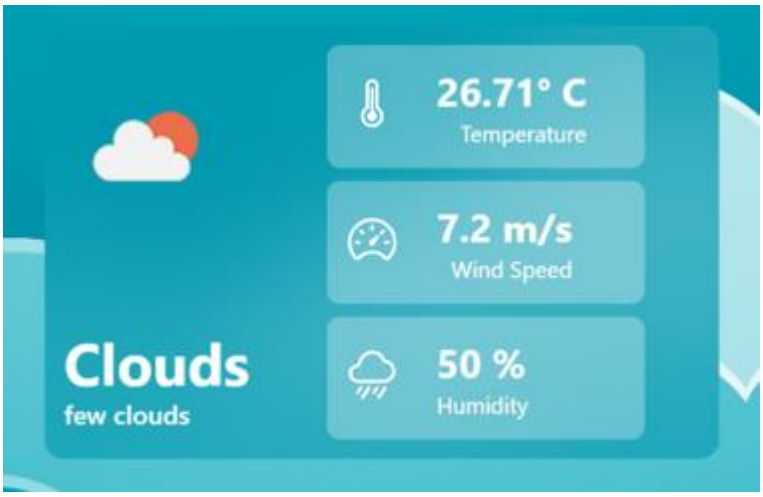
John@pentagonspace.in

Part 1

Part 2

Part 3

Part 4



John@pentagonspace.in

Initially we need to apply background, wait I'll share all the required images In compressed format. Click here to Dowload. Extract into `images` folder

Appling background:

----------Step 1----------

```
<body class="bg-[url('images/bg_image.jpg')]">

</body>
```

We need to cover the background(`bg-cover`), avoid repetition(`bg-no-repeat`), height should be screen height (`min-h-screen`)

----------Step 2----------

```
<body class="bg-[url('images/bg_image.jpg')] bg-cover bg-no-repeat min-h-screen">

</body>
```

We have 4 blocks inside the body and we need to give gap(`gap-5`) between each and every block and it is in column direction (`flex-col`), space around content (`p-10`)

----------Step 3----------

```
<body class="bg-[url('images/bg_image.jpg')] bg-cover bg-no-repeat min-h-screen items-center flex flex-col gap-5 p-10">

</body>
```

Part 1


```
<div class="flex items-center w-fit">

        <h1 class="text-white text-2xl font-bold">Weather App</h1>
    <img class="h-14 w-14" src="images/116.png" alt="" >

    </div>
```

Part 2


```
<form action="" method="get" class="flex gap-2 justify-center items-center w-4/5 md:w-
3/5 lg:w-[70%] ">

        <input type="text" name="city" class="rounded-full py-2 px-3 focus:bg-white/20
outline-white/70 bg-white focus:placeholder:text-white text-white w-4/5 md:w-3/5 lg:w-
[35%] " placeholder="Enter the City">

        <button class="search" type="submit">
            <ion-icon class="text-white text-2xl bg-white/20 rounded-full p-3"
name="search-outline"></ion-icon>
        </button>
    </form>
```

Part 3

John@pentagonspace.in

```
<div class="flex bg-white/20 p-3 rounded-3xl backdrop-blur-sm text-white w-4/5 md:w-3/5
lg:w-[35%] cursor-pointer hover:scale-[105%] ease-in-out duration-500 justify-between">

<div id='left' class="flex flex-col">
    <div id="top" class="mb-6 ">
        <p class="font-bold text-3xl">26.71&deg; C</p>
        <p class="font-semibold">Now</p>
    </div>
    <div id="bottom" class="font-semibold text-xl">
       {{name}}, {{country}}
    </div>
</div>
<div id="right" class="justify-center items-center">
    <img src="images/113.png" alt="" class="w-28 h-28">
</div>
    </div>
```

Day 3: Weather App – Frontend Part -4

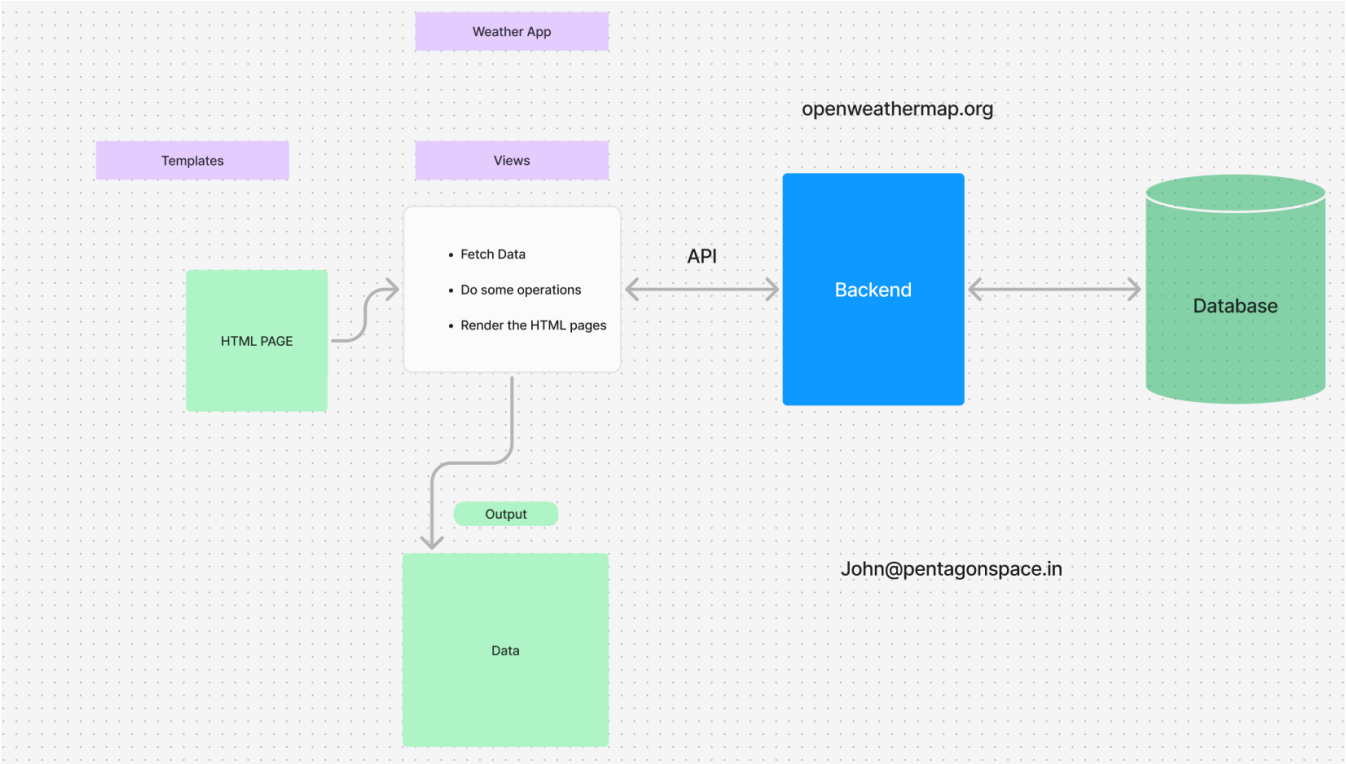
Part 4


```
<div class="flex p-3 bg-[#0198afa9] rounded-xl backdrop-blur-2xl md:w-3/5 lg:w-
[35%] w-[80%] gap-12" >

  <div id="left" class="text-white">
      <div class="top mb-16 mt-3 ">
          <img src="images/116.png" alt="">
      </div>
      <div id="bottom">
          <p class="flex font-bold text-4xl"> Clouds </p>
          <p class="font-semibold">few clouds</p></div></div>
```


John@pentagonspace.in

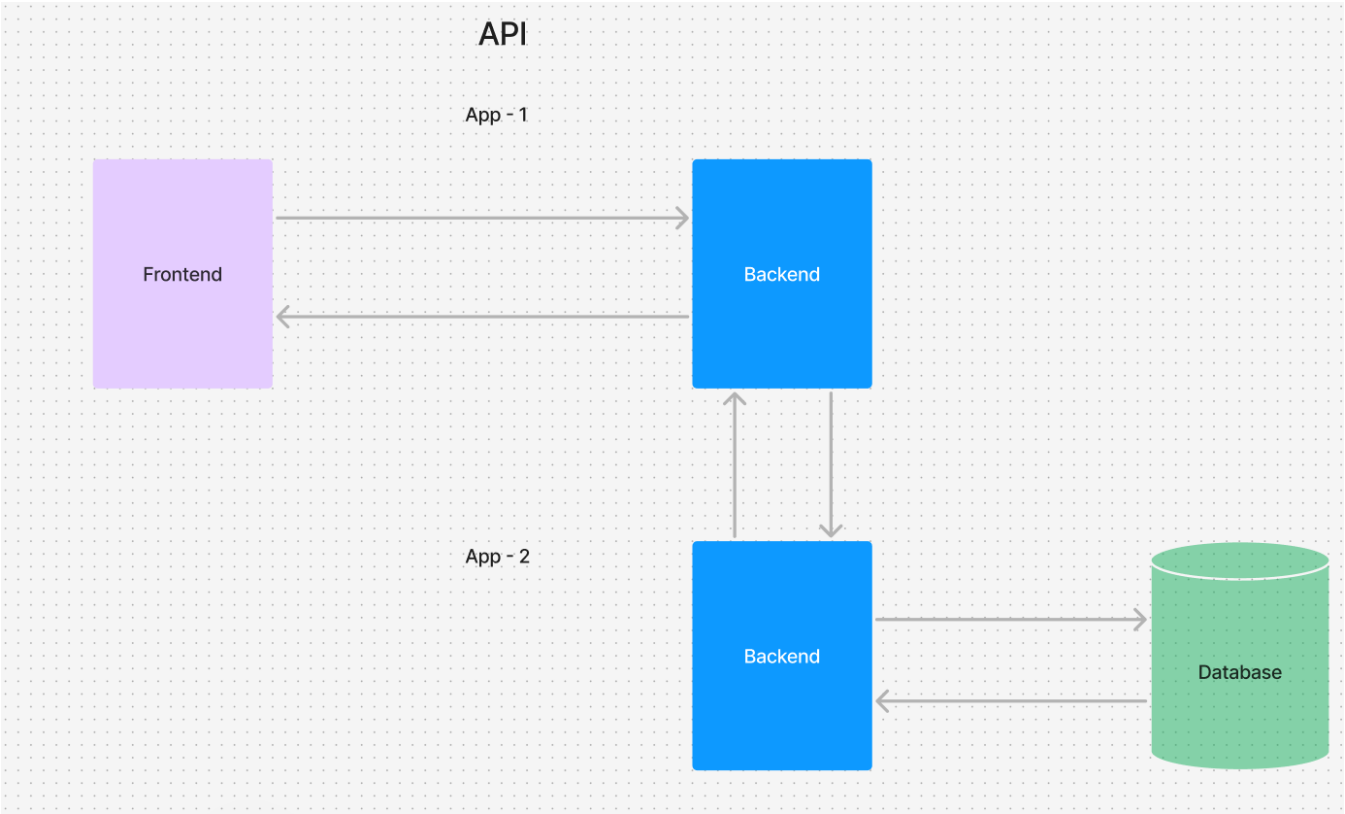
```
<div class="flex w-1/2 flex-col space-y-2">


<div id="right" class="flex p-3 bg-white/30 rounded-lg gap-6">
            <div id="left"><ion-icon class="text-white w-8 h-8 mt-2"
name="thermometer-outline"></ion-icon></div>

            <div id="right" class="text-white">

                <div class="font-bold text-2xl">26.71&deg; C</div>
                <div class="text-sm text-right">Temperature</div>

            </div>


        </div>
        <div id="right" class="flex p-3 bg-white/30 rounded-lg gap-6">
            <div id="left"><ion-icon class="text-white w-8 h-8 mt-2"
name="speedometer-outline"></ion-icon></div>

            <div id="right" class="text-white">

                <div class="font-bold text-2xl">7.2 m/s</div>
                <div class="text-sm text-right">Wind Speed</div>
            </div>
        </div>
        <div id="right" class="flex p-3 bg-white/30 rounded-lg gap-6">
            <div id="left"><ion-icon class="text-white w-8 h-8 mt-2" name="rainy-
outline"></ion-icon></div>

            <div id="right" class="text-white">

                <div class="font-bold text-2xl">50 %</div>
                <div class="text-sm text-right">Humidity</div> </div>
            </div></div></div>
```

Day 4: Weather App – API + Backend




```
from django.shortcuts import render
import requests

def home(request):
    city = request.GET.get('city', 'bangalore')
    api_key = 'b87770e0c526050cc7426cd23fa9cf28'
    url =
f'https://api.openweathermap.org/data/2.5/weather?q={city}&appid={api_key}&units=metric
'

    print(url)

    api = requests.get(url).json()

    temperature = api['main']['temp']
    wind_speed = api['wind']['speed']
    humidity = api['main']['humidity']
    name = api['name']
    country = api['sys']['country']
    icon = api['weather'][0]['icon']
    weather = api["weather"][0]['main']
    description = api['weather'][0]['description']
```

```
    img_url = f"https://openweathermap.org/img/wn/{icon}@2x.png"


    #
https://api.openweathermap.org/data/2.5/weather?q=bangalore&appid=b87770e0c526050cc7426
cd23fa9cf28&units=metric

    return render(request, 'index.html', {'temperature':temperature,
'wind_speed':wind_speed, 'humidity':humidity, 'name':name,
'country':country,'img_url':img_url, 'weather':weather, 'description':description})
```



Note:

You have to install module `requests` in order to handle API !

```
pip install requests
```


John@pentagonspace.in

Hint: How to Setup Backend?

Step 1: Ensure that you have installed with python (Install the latest stable version 3.11.9) and check in command prompt

```
py -V
```

(or)

```
python --version
```

Step 2: Install Django (if first command not working try others)

```
pip install django
```

(or)

```
py -m pip install django
```

(or)

```
python -m pip install django
```

Step 3: Create Django project

```
django-admin startproject project
```

(or)

```
py -m django startproject project
```

Rename the project folder to backend and get into that (change directory)

```
cd backend
```

Create app,

```
py manage.py startapp app
```

Create `templates`, `static` folder (only inside the `app` folder so that no need of extra configuration in `settings.py`)

John@pentagonspace.in

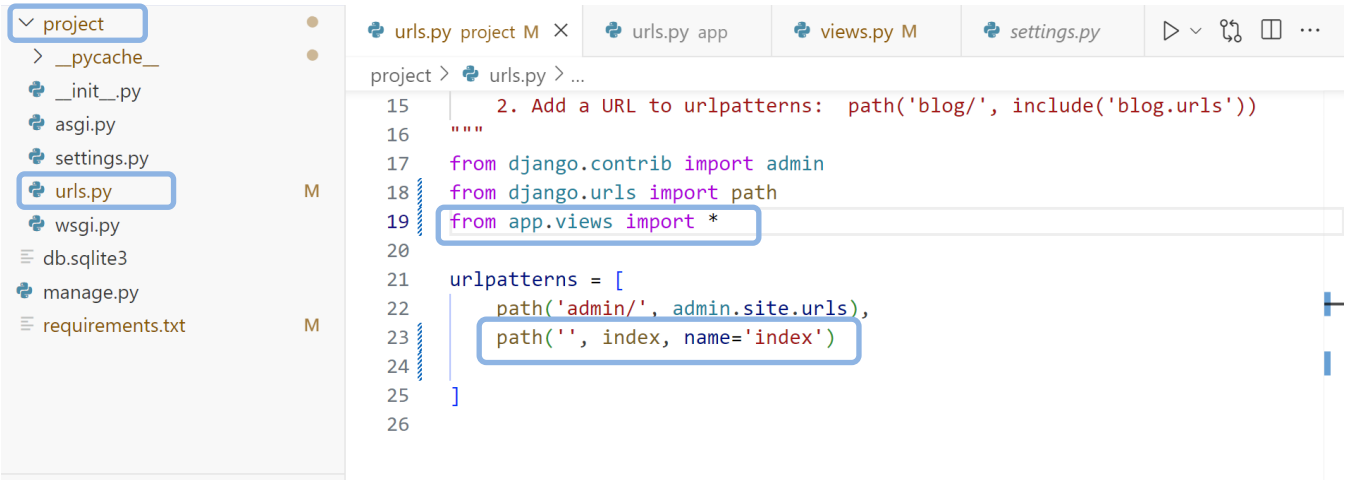
Day 5: Weather App – Frontend + Backend

This is the crucial day where you will learn how to connect frontend and the backend.

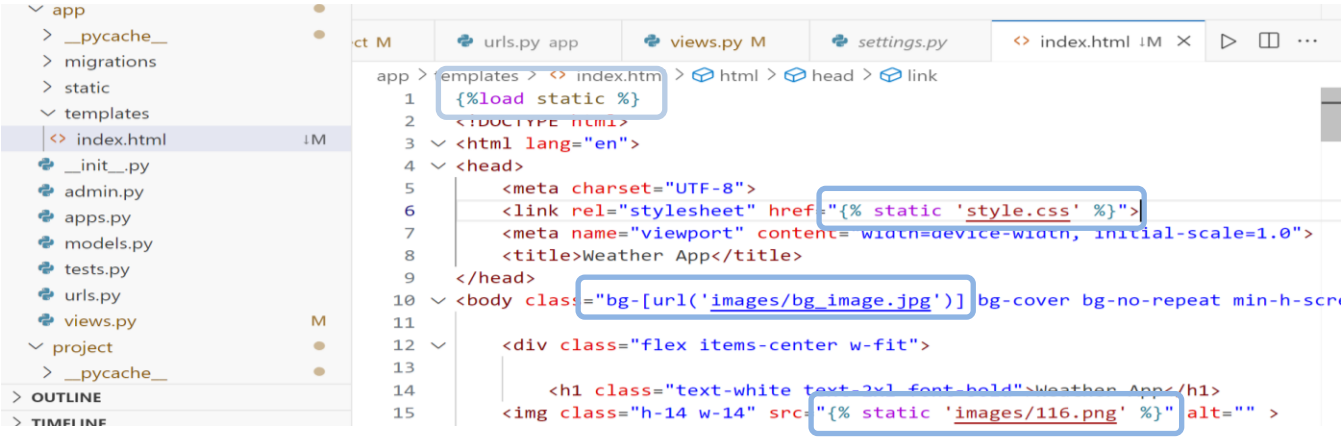
Step 1: Move the `index.html` to `templates` folder and move the `images` folder, `style.css` into `static` folder.

Step 2: Configure the `urls.py` of project folder (refer the screenshot below)

Step 3: Create one view - `index` function (refer API + Backend) inside the `views.py` (inside the app folder)



Step 4: Load the `static` inside the `index.html` and link the `style.css` all the images (except background image) using static url



Step 5: Finally run the server and follow the URL http://127.0.0.1:8000/

```
py manage.py runserver
```

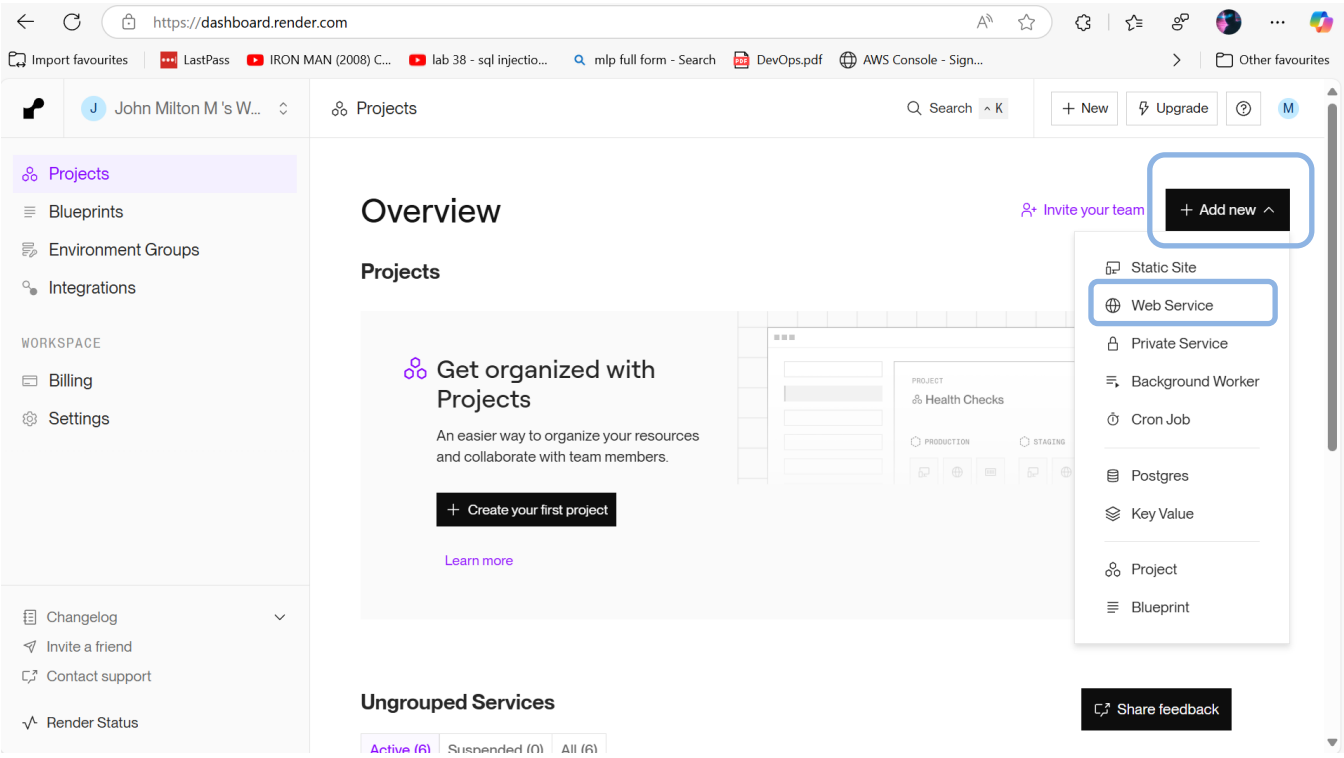

John@pentagonspace.in

Day 6: Deployment (Production Server)

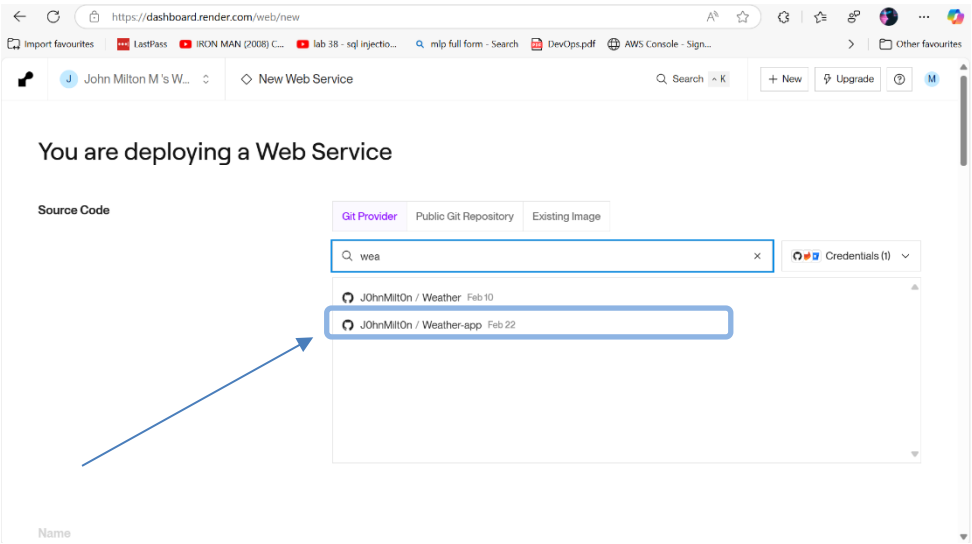
This is the last and foremost step involved in hosting your own public website

Step 1: Login to render.com and go to dashboard.

Step 2: Create new web service.



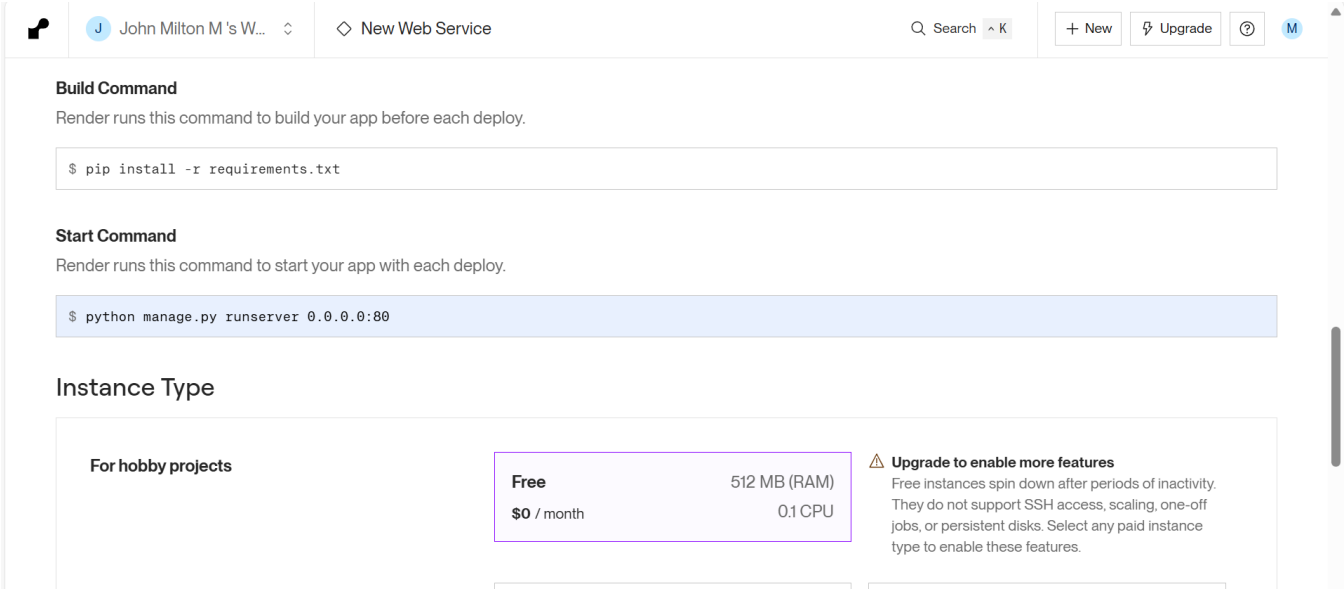
Step 3: Upload the project to github Repository and connect to render.



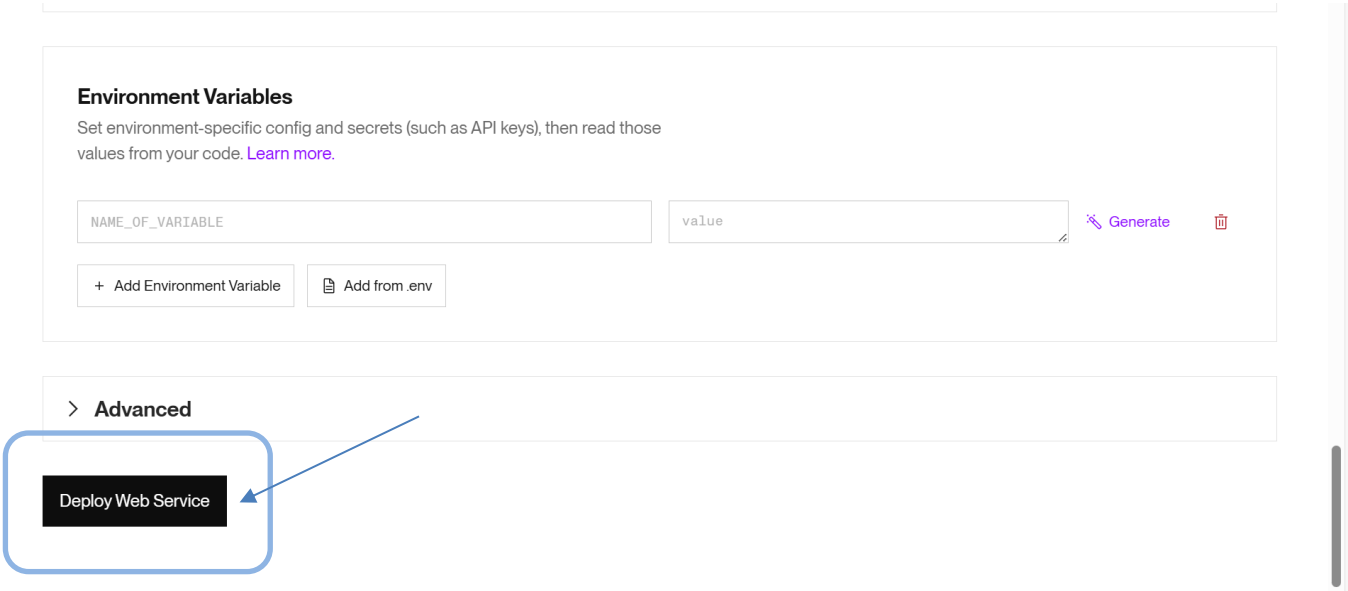
John@pentagonspace.in

Step 4: Add start command and select Instance type as free

```
py manage.py runserver 0.0.0.0:80
```



Step 5: Click on Deploy and wait until you get green signal.



Now click on the link. Voila, Your website is live !!!

John@pentagonspace.in