The Weather App

This app is used to detect the weather of any city across the world.

Pre-requisite:

Basic HTML, CSS and JavaScript

The HTML Code:

The <head> tag and the opening <body> tag:

<!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>Weather App - e86 </title>

    <link rel="stylesheet" href="style.css">

</head>

<body>

Within the body tag, there’s a main-container wrapping everything:

<div class="main-container">

Then there is another div inside called container-top

<div class="container top">

This very div called container-top houses **h1, form and section**

<h1>Weather - App</h1>

**The Form**

<form action="" onsubmit="return false">

          <input type="text" class="search" placeholder="Enter city name">

          <button type="submit" class="btn">Check</button>

          <p class="error"></p>

    </form>

*On submit “return false” allows the browser not to refresh after the user clicks the submit button.*

*P class error tells the user to enter a correct location*

**The Section**

Contains 4 divs – container that wraps icon, location and current

<section>

                <div class="container">

                    <div class="icon">

                        <img src="" alt="" class="weather-icon">

                    </div>

                    <div class="location">

                        <div class="city">Your Location</div>

                        <div class="date">Date Placeholder</div>

                    </div>

                    <div class="current">

                        <div class="temp">Temperature <span>°C</span></div>

                        <div class="weather">Weather: Condtion</div>

                        <div class="temp-range">Temp Range: °C / °C</div>

                    </div>

                </div>

            </section>

Link the JavaScript tag

<script src="app.js"></script>

The CSS Code:

Doing a default CSS setting for margin, padding and border with \*

\*{

    margin: 0;

    padding: 0;

    box-sizing: border-box;

}

The most essential CSS Property a web should have: *background, font-family, width, height, display, justify-content, align-items.*

body {

    background: linear-gradient(to right bottom, #852cea, #001678);

    font-family: Arial, sans-serif;

    width: 100%;

    height: 100vh; /\*height: 100vh; means the height of this element is equal to 100% of the viewport height. \*/

    display: flex;

    justify-content: center;

    align-items: center;

}

Styling the *container class*

.container {

    width: 80%;

    margin: 0 auto;

}

Styling the main-*container class*

.main-container {

    display: flex;

    justify-content: center;

    align-items: center;

    flex-direction: column;

    width: 100%;

    max-width: 320px;

    height: 80vh;

    background: url(./images/background.jpg) no-repeat center;

    background-size: cover;

    border-radius: 10px;

    overflow: hidden;

    position: relative;

}

*The ::before selector inserts something before the content of each selected element(s)*

.main-container::before {

    content: "";

    position: absolute;

    left: 0;

    top: 0;

    width: 100%;

    height: 100%;

    background: linear-gradient(to botttom,

        rgba(0,0,0,0),

        rgba(0,0,0,.6));

    z-index: 0;

}

*Styling the container-top class excluding everything in the <section> tag:*

.top {

    text-align: center;

    z-index: 999;

}

.top h1 {

    color: white;

    line-height: 2;

}

.top input,

.top button {

    display: block;

    width: 80%;

    margin: 0 auto;

    padding: 10px;

    margin-bottom: 5px;

    outline: none;

    border: none;

}

.top input {

    background-color: rgba(0,0,0,.5);

    border-bottom: solid 2px rgb(255, 255, 254);

    border-radius: 10px 0 10px 0;

    text-align: center;

    font-size: 14px;

    color: white;

}

.top input:focus {

    color: white;

    font-size: 15px;

}

.top input::placeholder {

    color: rgba(255,255,255.5);

}

.top button {

    background-color: #db0000;

    color: white;

    font-size: 14px;

    font-weight: bold;

    text-transform: uppercase;

    border-radius: 3px;

    cursor: pointer;

    transition: all .3s;

}

.top button:hover {

    background-color: #1f1e1e;

}

.top .error {

    color: white;

    font-size: 14px;

}

*Styling the icon, location and current class*

.icon {

    text-align: center;

    padding-top: 15px;

}

.location {

    padding-bottom: 15px;

    text-align: center;

    color: white;

    line-height: 1.6;

}

.location .city {

    font-size: 20px;

    font-weight: bold;

}

.current {

    width: 100%;

    position: relative;

    padding-top: 10px;

    color: white;

    text-align: left;

}

.current::before {

    content: "";

    position: absolute;

    top: 0;

    left: 0;

    width: 100%;

    height: 1px;

    background-color: white;

}

.current > \* {

    padding-bottom: 10px;

}

*That means the selector div > p.some\_class only matches paragraphs of .some\_class that are nested* ***directly inside*** *a div, and not any paragraphs that are nested further within. This implies that every element matching div > p.some\_class necessarily also matches div p.some\_class, with the* [*descendant combinator*](https://www.w3.org/TR/selectors/#descendant-combinators) *(space), so the two are understandably often confused. – from stackoverflow.* Meaning it’s same as .current{}

The JavaScript Code:

First, we need to declare the *api* as a constant:

const api = {

    key:  "2c931dacd44278418a80b71cf093ac78",

    base: "https://api.openweathermap.org/data/2.5/"

}

Then, we need to declare the *search and button*: