2024

Internet Programming Project

WEATHER APPLICATION

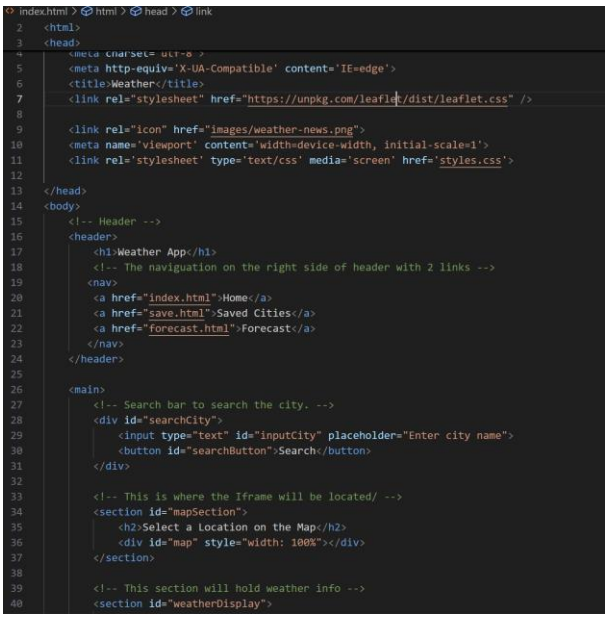
PARTH PATEL

Weather App

Git Repository: https://github.com/ParthPatelA/Weather.git

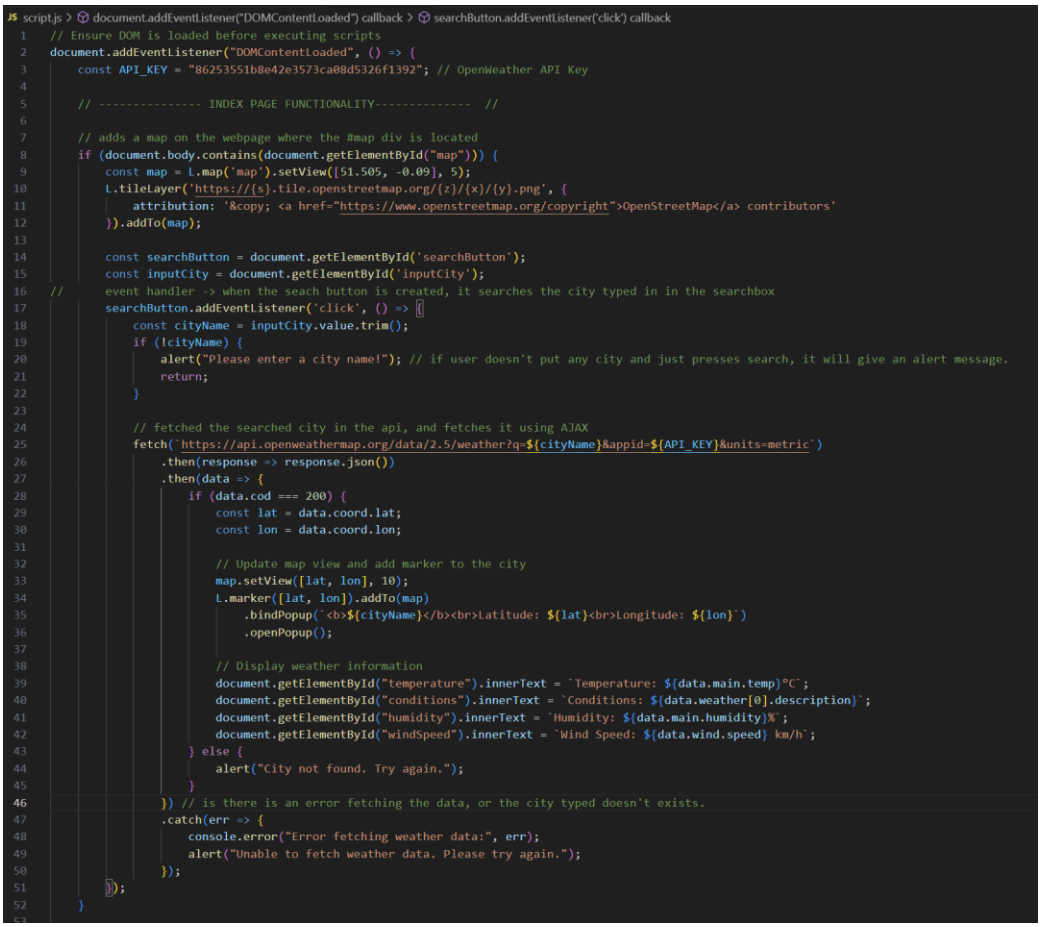
- Started by making the index page (main page), which has a header, with 3 links to the side.

The 3 links that lead to the home page and lead to the two other pages of this website. Then, the home page also has a search bar that allows you to search any city, and have it appeared on the map. The map was taken by the openweather API, and the weather information is also taken from the API. Lastly the footer also has the the 3 links for each webpage.



The leaflet styles and leaflet script is for the map put on the webpage. Makes the map interactive.

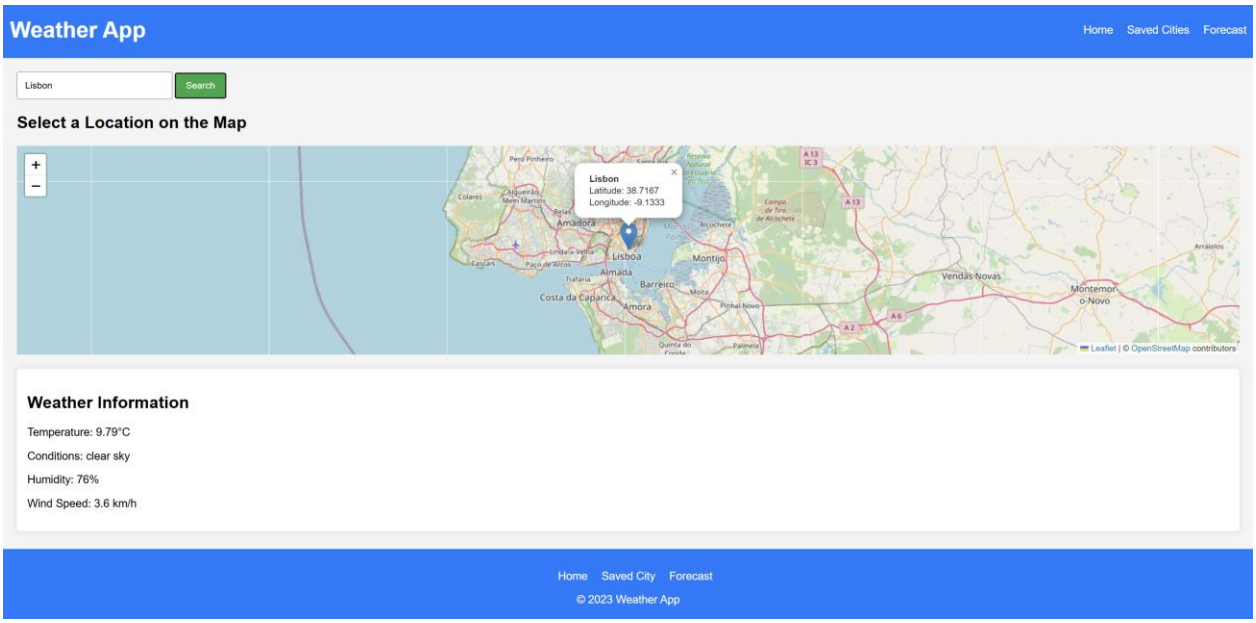
This is the javascript for the index page:



```
script.js > document.addEventListener("DOMContentLoaded") callback > searchButton.addEventListener('click') callback
// Ensure DOM is loaded before executing scripts
document.addEventListener("DOMContentLoaded", () => {
    const API_KEY = "86253551b8e42e3573ca08d5326f1392"; // OpenWeather API Key

    // ---------- INDEX PAGE FUNCTIONALITY---------- //

    // adds a map on the webpage where the #map div is located
    if (document.body.contains(document.getElementById("map"))) {
        const map = L.map('map').setView([51.505, -0.09], 5);
        L.tileLayer('https://{s}.tile.openstreetmap.org/{z}/{x}/{y}.png', {
            attribution: '&copy; <a href="https://www.openstreetmap.org/copyright">OpenStreetMap</a> contributors'
        }).addTo(map);

        const searchButton = document.getElementById('searchButton');
        const inputCity = document.getElementById('inputCity');
//      event handler -> when the seach button is created, it searches the city typed in in the searchbox
        searchButton.addEventListener('click', () => {
            const cityName = inputCity.value.trim();
            if (!cityName) {
                alert("Please enter a city name!"); // if user doesn't put any city and just presses search, it will give an alert message.
                return;
            }

            // fetched the searched city in the api, and fetches it using AJAX
            fetch(`https://api.openweathermap.org/data/2.5/weather?q=${cityName}&appid=${API_KEY}&units=metric`)
                .then(response => response.json())
                .then(data => {
                    if (data.cod === 200) {
                        const lat = data.coord.lat;
                        const lon = data.coord.lon;

                        // Update map view and add marker to the city
                        map.setView([lat, lon], 10);
                        L.marker([lat, lon]).addTo(map)
                            .bindPopup(`<b>${cityName}</b><br>Latitude: ${lat}<br>Longitude: ${lon}`)
                            .openPopup();

                        // Display weather information
                        document.getElementById("temperature").innerText = `Temperature: ${data.main.temp}°C`;
                        document.getElementById("conditions").innerText = `Conditions: ${data.weather[0].description}`;
                        document.getElementById("humidity").innerText = `Humidity: ${data.main.humidity}%`;
                        document.getElementById("windSpeed").innerText = `Wind Speed: ${data.wind.speed} km/h`;
                    } else {
                        alert("City not found. Try again.");
                    }
                }) // is there is an error fetching the data, or the city typed doesn't exists.
                .catch(err => {
                    console.error("Error fetching weather data:", err);
                    alert("Unable to fetch weather data. Please try again.");
                });
        });
    }
```

Event listener for “DOMContentLoaded” makes sure that the entire DOM HTML is loaded. Then, it puts the map from the api on the page on the div “map” where the world map will be located. There is also a search feature, where when the user searches for a city, it will fetch the city weather information in an API, using AJAX. Once the data is fetched, it will apear on the weather information div and will display each information. Also, if no city was searched or city searched doesn’t exists, it will give an alert message.

First Page Result:

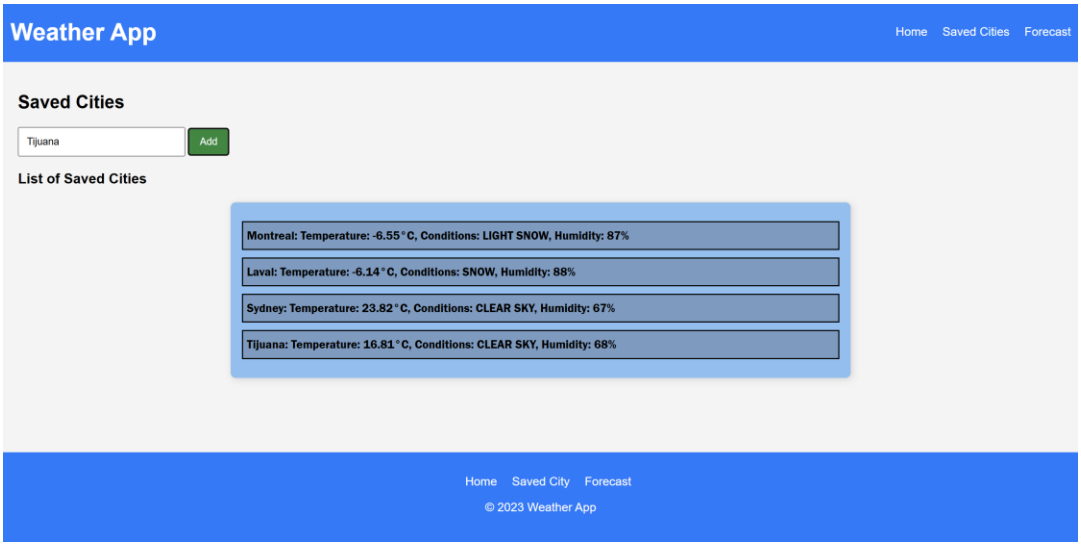


- The Second page is the saved city page, where the user can enter a city and save it in a list of cities, where the weather is displayed for each city. This is implemented so the user can see the weather of the city he likes to see every time.

```
JS script.js > document.addEventListener("DOMContentLoaded") callback > updateForecastCards
2   document.addEventListener("DOMContentLoaded", () => {
54      // ---------- SAVE PAGE FUNCTIONALITY ---------- //
55
56      if (document.body.contains(document.getElementById("savedCities"))) {
57          const addButton = document.getElementById('addButton');
58          const inputSavedCity = document.getElementById('inputSavedCity');
59          const savedCitiesList = document.getElementById('savedCities');
60
61          // event listener for save button, trigger the function where it will save the city's weather info in a list.
62          addButton.addEventListener('click', () => {
63              const cityName = inputSavedCity.value.trim();
64              if (!cityName) { // if user doesn't enter anything.
65                  alert("Please enter a city name!");
66                  return;
67              }
68
69              // fetches from the API using AJAX
70              fetch(`https://api.openweathermap.org/data/2.5/weather?q=${cityName}&appid=${API_KEY}&units=metric`)
71                  .then(response => response.json())
72                  .then(data => {
73                      if (data.cod === 200) {
74                          const lat = data.coord.lat;
75                          const lon = data.coord.lon;
76
77                          // Create list item for the saved city
78                          const li = document.createElement("li");
79                          // in the list it will have all the weather info.
80                          li.innerHTML = `
81                              <strong>${cityName}</strong>:
82                              Temperature: ${data.main.temp}°C,
83                              Conditions: ${data.weather[0].description.toUpperCase()},
84                              Humidity: ${data.main.humidity}%
85                          `;
86                          savedCitiesList.appendChild(li); // it will keep appending it for each city saved in a list.
87                      } else {
88                          alert("City not found. Try again."); // if the city is not found.
89                      }
90                  })
91                  // If there are any error fetching the data.
92                  .catch(err => {
93                      console.error("Error fetching weather data:", err);
94                      alert("Unable to fetch weather data. Please try again.");
95                  });
96      });
97  }
```

The search featuresis the same as the previous page. The only difference is that once you search a city, it will fetch the weather information from the API and make a <li> element where it displays the weather information. Then you can add multiple cities to a list and see all the weather information for each city. This allows the user to see all the cities it wants to see on a regular basis.

The Second webpage:



```html
<!-- save.html > ... -->
<html>
<head>
</head>
<body>
    <header>
        <h1>Weather App</h1>
        <!-- The naviguation on the right side of header with 2 links -->
        <nav>
        <a href="index.html">Home</a>
        <a href="save.html">Saved Cities</a>
        <a href="forecast.html">Forecast</a>
        </nav>
    </header>
<main>
        <!-- Saved Cities Title -->
        <section id="savedCitiesSection">
            <h2>Saved Cities</h2>
            <!-- Search bar for adding new cities -->
            <div id="addCity">
                <input type="text" id="inputSavedCity" placeholder="Enter city name to add"> <!--The search bar-->
                <button id="addButton">Add</button> <!--The search button-->
            </div>

            <!-- List of saved cities -->
            <div id="cityList">
                <h3>List of Saved Cities</h3>
                <ul id="savedCities">
                    <!-- Dynamically populated saved cities -->
                </ul>
            </div>
        </section>
    </main>

    <!-- Footer -->
    <footer>
        <nav> <!--Links-->
            <a href="index.html">Home</a>
            <a href="save.html">Saved City</a>
            <a href="forecast.html">Forecast</a>
        </nav>
        <p>&copy; 2023 Weather App</p>
    </footer>

    <script src='script.js'></script>

</body>
</html>
```

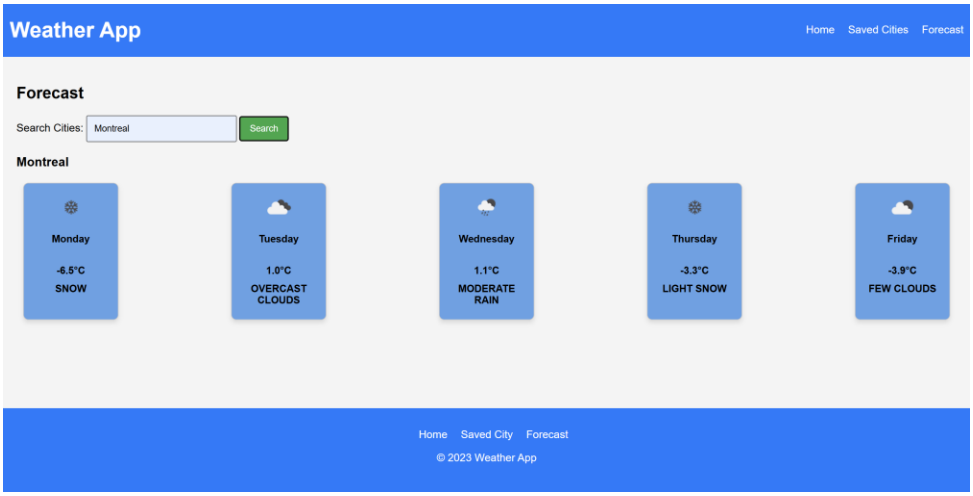
- The third webpage is the webpage that shows the forecast for the next 5 days:

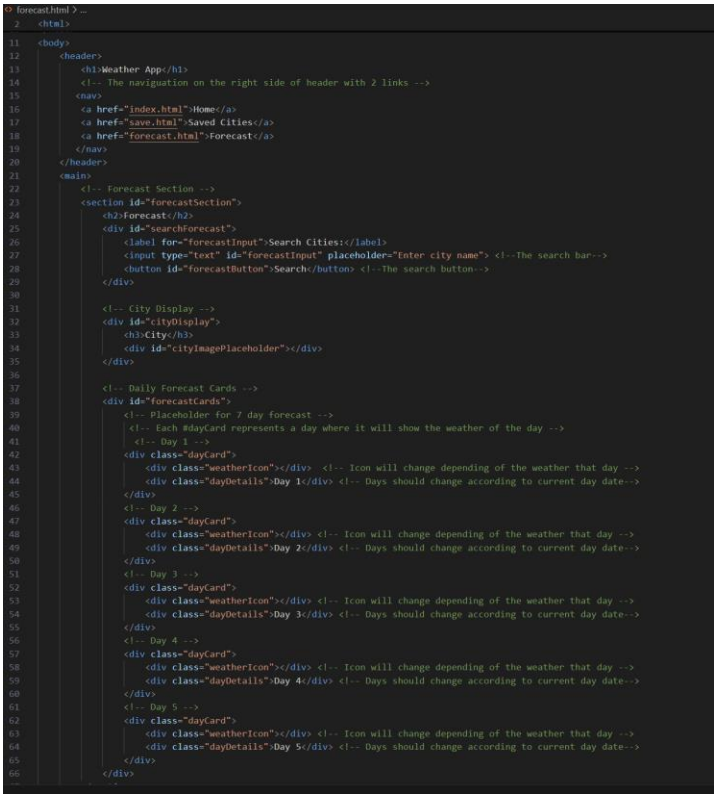
```javascript
// JS script.js > document.addEventListener("DOMContentLoaded") callback > searchButton.addEventListener('click') callback > then() callback
document.addEventListener("DOMContentLoaded", () => {

    // ---------------- FORECAST PAGE FUNCTIONALITY ------------------ //

    if (document.body.contains(document.getElementById("forecastSection"))) {
        // gets all the info about the where the divs are.
    const forecastButton = document.getElementById("forecastButton");
    const forecastInput = document.getElementById("forecastInput");
    const forecastCards = document.getElementById("forecastCards");
    const cityDisplay = document.getElementById("cityDisplay");

    // When searching a city
    forecastButton.addEventListener("click", () => {
        const cityName = forecastInput.value.trim();
        if (!cityName) {
            alert("Please enter a city name!");
            return;
        }

        // Fetch forecast data
        fetch(`https://api.openweathermap.org/data/2.5/forecast?q=${cityName}&appid=${API_KEY}&units=metric`)
            .then(response => response.json())
            .then(data => {
                if (data.cod === "200") {
                    updateCityDisplay(cityName); //updates the displays
                    updateForecastCards(data.list); //upates the card
                } else {
                    alert("City not found. Please try again.");
                }
            })
            .catch(err => {
                console.error("Error fetching data:", err);
                alert("Failed to fetch forecast data. Please try again later.");
            });
    });

    // Update the city display section
    function updateCityDisplay(cityName) {
        cityDisplay.innerHTML = `
            <h3>${cityName}</h3>
        `;
    }
```

```
139
140     // It will show the forecast of 5 days
141     // Update the forecast cards
142     function updateForecastCards(forecastList) {
143         const days = ["Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"]; // depending
144         let dayIndex = new Date().getDay();                                                          // So if use
145
146         // Clear previous forecast cards
147         const dayCards = forecastCards.querySelectorAll(".dayCard");
148         dayCards.forEach(card => card.innerHTML = "");
149
150         // Populate new forecast data
151         for (let i = 0; i < 7; i++) {
152             const forecast = forecastList[i * 8]; // Every 8th index gives roughly a day apart
153             if (forecast) {
154                 // the information on each card
155                 const card = dayCards[i];
156                 const iconCode = forecast.weather[0].icon; // the weather icon based on the forecast.
157                 const temperature = forecast.main.temp.toFixed(1).toUpperCase(1);
158                 const description = forecast.weather[0].description.toUpperCase();
159
160                 card.innerHTML =
161                 // the icon for the forecast is also taken from openweather, where I got my API
162                 `
163                     <div class="weatherIcon">
164                         <img src="https://openweathermap.org/img/wn/${iconCode}.png" alt="${description}">
165                     </div>
166                     <div class="dayDetails">
167                         <h4>${days[(dayIndex + i) % 7]}</h4>
168                         <p>${temperature}°C</p>
169                         <p>${description}</p>
170                     </div>
171                 `;
172             }
173         }
174     }
175     }
176 });
```

The search feature is the same as the previous two pages. When the user presses the button search, it fetches the data from the API and will display the forecast for the next 5 days in the cards. It will show the forecast for the next 5 days based on the day you searched. For example, if you search on Wednesday, it will show the forecast for Wednesday, Thursday, Friday, Saturday and Sunday. Each card shows an icon of the weather based on the forecast that day, so if it is snowing, it will have a snowing icon. It will also show the Temperature of that day, along side the weather conditions of that day. The icon was taken from OpenWeather, the same website I got the API for the weather information.

The result of the third webpage:





```
forecast.html > ...
<html>

<body>
    <header>
        <h1>Weather App</h1>
        <!-- The navigation on the right side of header with 2 links -->
        <nav>
        <a href="index.html">Home</a>
        <a href="save.html">Saved Cities</a>
        <a href="forecast.html">Forecast</a>
        </nav>
    </header>
    <main>
        <!-- Forecast Section -->
        <section id="forecastSection">
            <h2>Forecast</h2>
            <div id="searchForecast">
                <label for="forecastInput">Search Cities:</label>
                <input type="text" id="forecastInput" placeholder="Enter city name"> <!--The search bar-->
                <button id="forecastButton">Search</button> <!--The search button-->
            </div>

            <!-- City Display -->
            <div id="cityDisplay">
                <h3>City</h3>
                <div id="cityImagePlaceholder"></div>
            </div>

            <!-- Daily Forecast Cards -->
            <div id="forecastCards">
                <!-- Placeholder for 7 day forecast -->
                <!-- Each #dayCard represents a day where it will show the weather of the day -->
                <!-- Day 1 -->
                <div class="dayCard">
                    <div class="weatherIcon"></div>  <!-- Icon will change depending of the weather that day -->
                    <div class="dayDetails">Day 1</div> <!-- Days should change according to current day date-->
                </div>
                <!-- Day 2 -->
                <div class="dayCard">
                    <div class="weatherIcon"></div> <!-- Icon will change depending of the weather that day -->
                    <div class="dayDetails">Day 2</div> <!-- Days should change according to current day date-->
                </div>
                <!-- Day 3 -->
                <div class="dayCard">
                    <div class="weatherIcon"></div> <!-- Icon will change depending of the weather that day -->
                    <div class="dayDetails">Day 3</div> <!-- Days should change according to current day date-->
                </div>
                <!-- Day 4 -->
                <div class="dayCard">
                    <div class="weatherIcon"></div> <!-- Icon will change depending of the weather that day -->
                    <div class="dayDetails">Day 4</div> <!-- Days should change according to current day date-->
                </div>
                <!-- Day 5 -->
                <div class="dayCard">
                    <div class="weatherIcon"></div> <!-- Icon will change depending of the weather that day -->
                    <div class="dayDetails">Day 5</div> <!-- Days should change according to current day date-->
                </div>
            </div>
```

The CSS for the webpage looks like this:

styles.css > ...

```css
/* Main page / Index */
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
    background-color: #f4f4f4;
    display: flex;
    flex-direction: column;
    height: 100%;
}

header {
    display: flex; /* Use flexbox */
    justify-content: space-between; /* Space out the items */
    align-items: center; /* Center items vertically */
    background: #007BFF;
    color: #fff;
    padding-left: 10px;
    padding-right: 10px;
}

nav {
    display: flex; /* Use flexbox for navigation links */
}

nav a {
    color: #fff; /* Set link color */
    text-decoration: none;
    margin-left: 20px;
}

/* On hover it will change color and underline it. */
nav a:hover {
    text-decoration: underline;
    color: #00008b;
}

/*  The seacrch bar */
input[type="text"] {
    padding: 10px;
    width: 200px;
}
```

```css
/* search button */
button {
    padding: 10px 15px;
    background: #28A745;
    color: #fff;
    border: none;
    cursor: pointer;
}

button:hover {
    background: #218838;
}
/* Main area with content */
main {
    padding: 20px;
    flex: 1;
}

/* The Iframe Map section */
#mapSection {
    margin-bottom: 20px;
}
/* map size */
#map {
    width: 100%;
    height: 300px;
}

/* Display with weather informations */
#weatherDisplay {
    background: #fff;
    padding: 15px;
    border-radius: 5px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
}

html {
    height: 100%;
}
```

```css
/* Footer */
footer {
    background-color: #007BFF;
    padding: 10px;
    text-align: center;
    color: #fff;
    padding: 30px;
    display: flex;
    flex-direction: column;
    align-items: center;
}


/* Saved Cities */
/* ------------------------------------------------------------------ */


/* Makes like a box around the main content */
#savedCities {
    width: 90%;
    max-width: 800px;
    margin: 20px auto;
    padding: 15px;
    font-size: 16px;
    background-color: #86c0f2;
    border: 1px solid #dddddd;
    border-radius: 8px;
    box-shadow: 2px 2px 8px rgba(0, 0, 0, 0.1);
    list-style-type: none;
}

#savedCities li {
    background-color: #769bc2;
    font-family : 'Franklin Gothic Medium', 'Arial Narrow', Arial, sans-serif;
    padding: 5px;
    margin: 10px 0; /* Add space between individual city entries */
    line-height: 1.6;
    border: #000000 solid ;
}


/* Forecast */
/* ---------------------------------- */

/* Forecast Section Styling */
/* Puts the info in some sort of box */
.forecast-container {
    display: flex;
    flex-wrap: wrap;
    gap: 1rem;
}
```

styles.css > ...

```css
.forecast-card {
    background-color: #f0f8ff;
    border: 1px solid #ccc;
    padding: 1rem;
    border-radius: 8px;
    width: 200px;
    text-align: center;
    box-shadow: 2px 2px 5px rgba(0, 0, 0, 0.1);
}

.forecast-card h4 {
    font-size: 1.2rem;
    margin-bottom: 0.5rem;
}

#forecastCards {
    display: flex;
    justify-content: space-between;
    flex-wrap: wrap;
    gap: 10px;
    margin-top: 20px;
    margin-left: 10px;
    margin-right: 10px;
}

.dayCard {
    background: #61a2e7;
    border: 1px solid #ccc;
    border-radius: 8px;
    padding: 10px;
    text-align: center;
    width: 120px;
    box-shadow: 0 4px 6px rgba(0, 0, 0, 0.1);
}

.dayCard img {
    width: 50px;
    height: 50px;
}

.weatherIcon {
    margin-bottom: 10px;
}

.dayDetails h4 {
    font-size: 15px;
    margin: 10px 0;
    padding-bottom: 20px;
}

.dayDetails p {
    margin: 5px;
    font-size: 15px;
    padding-bottom: 5px;
    font-weight: bold;
}
```