**Wheather Forecast App**

Code of HTML, CSS, and JavaScript

HTML file (index.html):

<!DOCTYPE html>

<html lang="en">

  <head>

    <meta charset="utf-8">

    <title>Weather Forecast App </title>

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

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

    <script src="script.js" defer></script>

  </head>

  <body>

    <h1>Weather Forecast</h1>

    <div class="container">

      <div class="weather-input">

        <h3>Enter a City Name</h3>

        <input class="city-input" type="text" placeholder="E.g., Delhi, Hyderabad, Mumbai" value="Delhi">

        <button class="search-btn">Search</button>

        <div class="separator"></div>

        <button class="location-btn">Use Current Location</button>

      </div>

      <div class="weather-data">

        <div class="current-weather">

          <div class="details">

            <h2>\_\_\_\_\_\_\_ ( \_\_\_\_\_\_ )</h2>

            <h6>Temperature: \_\_°C</h6>

            <h6>Wind: \_\_ M/S</h6>

            <h6>Humidity: \_\_%</h6>

          </div>

        </div>

        <div class="days-forecast">

          <h2>5-Days Forecast</h2>

          <ul class="weather-cards">

            <li class="card">

              <h3>( \_\_\_\_\_\_ )</h3>

              <h6>Temp: \_\_C</h6>

              <h6>Wind: \_\_ M/S</h6>

              <h6>Humidity: \_\_%</h6>

            </li>

            <li class="card">

              <h3>( \_\_\_\_\_\_ )</h3>

              <h6>Temp: \_\_C</h6>

              <h6>Wind: \_\_ M/S</h6>

              <h6>Humidity: \_\_%</h6>

            </li>

            <li class="card">

              <h3>( \_\_\_\_\_\_ )</h3>

              <h6>Temp: \_\_C</h6>

              <h6>Wind: \_\_ M/S</h6>

              <h6>Humidity: \_\_%</h6>

            </li>

            <li class="card">

              <h3>( \_\_\_\_\_\_ )</h3>

              <h6>Temp: \_\_C</h6>

              <h6>Wind: \_\_ M/S</h6>

              <h6>Humidity: \_\_%</h6>

            </li>

            <li class="card">

              <h3>( \_\_\_\_\_\_ )</h3>

              <h6>Temp: \_\_C</h6>

              <h6>Wind: \_\_ M/S</h6>

              <h6>Humidity: \_\_%</h6>

            </li>

          </ul>

        </div>

      </div>

    </div>

    <footer>

      <div>

          <h1>BY Seemala Indu</h1>

      </div>

  </footer>

  </body>

</html>

CSS File (style.css):

@import url('https://fonts.googleapis.com/css2?family=Open+Sans:wght@400;500;600;700&display=swap');

\* {

  margin: 0;

  padding: 0;

  box-sizing: border-box;

  font-family: 'Open Sans', sans-serif;

}

body {

  background: #FAF0E6;

}

h1 {

  background: #352F44;

  font-size: 1.75rem;

  text-align: center;

  padding: 18px 0;

  color: #fff;

}

.container {

  display: flex;

  gap: 35px;

  padding: 30px;

}

.weather-input {

  width: 550px;

}

.weather-input input {

  height: 46px;

  width: 100%;

  outline: none;

  font-size: 1.07rem;

  padding: 0 17px;

  margin: 10px 0 20px 0;

  border-radius: 4px;

  border: 1px solid #ccc;

}

.weather-input input:focus {

  padding: 0 16px;

  border: 2px solid #352F44;

}

.weather-input .separator {

  height: 1px;

  width: 100%;

  margin: 25px 0;

  background: #BBBBBB;

  display: flex;

  align-items: center;

  justify-content: center;

}

.weather-input .separator::before{

  content: "or";

  color: #6C757D;

  font-size: 1.18rem;

  padding: 0 15px;

  margin-top: -4px;

  background: #E3F2FD;

}

.weather-input button {

  width: 100%;

  padding: 10px 0;

  cursor: pointer;

  outline: none;

  border: none;

  border-radius: 4px;

  font-size: 1rem;

  color: #fff;

  background: #352F44;

  transition: 0.2s ease;

}

.weather-input .search-btn:hover {

  background: #584e72;

}

.weather-input .location-btn {

  background: #766b92;

}

.weather-input .location-btn:hover {

  background: #9b8cc1;

}

.weather-data {

  width: 100%;

}

.weather-data .current-weather {

  color: #fff;

  background: #352F44 ;

  border-radius: 5px;

  padding: 20px 70px 20px 20px;

  display: flex;

  justify-content: space-between;

}

.current-weather h2 {

  font-weight: 700;

  font-size: 1.7rem;

}

.weather-data h6 {

  margin-top: 12px;

  font-size: 1rem;

  font-weight: 500;

}

.current-weather .icon {

  text-align: center;

}

.current-weather .icon img {

  max-width: 120px;

  margin-top: -15px;

}

.current-weather .icon h6 {

  margin-top: -10px;

  text-transform: capitalize;

}

.days-forecast h2 {

  margin: 20px 0;

  font-size: 1.5rem;

}

.days-forecast .weather-cards {

  display: flex;

  gap: 20px;

}

.weather-cards .card {

  color: #fff;

  padding: 18px 16px;

  list-style: none;

  width: calc(100% / 5);

  background: #766b92;

  border-radius: 5px;

}

.weather-cards .card h3 {

  font-size: 1.3rem;

  font-weight: 600;

}

.weather-cards .card img {

  max-width: 70px;

  margin: 5px 0 -12px 0;

}

@media (max-width: 1400px) {

  .weather-data .current-weather {

    padding: 20px;

  }

  .weather-cards {

    flex-wrap: wrap;

  }

  .weather-cards .card {

    width: calc(100% / 4 - 15px);

  }

}

@media (max-width: 1200px) {

  .weather-cards .card {

    width: calc(100% / 3 - 15px);

  }

}

@media (max-width: 950px) {

  .weather-input {

    width: 450px;

  }

  .weather-cards .card {

    width: calc(100% / 2 - 10px);

  }

}

@media (max-width: 750px) {

  h1 {

    font-size: 1.45rem;

    padding: 16px 0;

  }

  .container {

    flex-wrap: wrap;

    padding: 15px;

  }

  .weather-input {

    width: 100%;

  }

  .weather-data h2 {

    font-size: 1.35rem;

  }

}

Javascript File (script.js) :

const cityInput = document.querySelector(".city-input");

const searchButton = document.querySelector(".search-btn");

const locationButton = document.querySelector(".location-btn");

const currentWeatherDiv = document.querySelector(".current-weather");

const weatherCardsDiv = document.querySelector(".weather-cards");

const API\_KEY = "6e45a8992f2ee6cf518d266b23503421"; // API key for OpenWeatherMap API

const createWeatherCard = (cityName, weatherItem, index) => {

    if(index === 0) { // HTML for the main weather card

        return `<div class="details">

                    <h2>${cityName} (${weatherItem.dt\_txt.split(" ")[0]})</h2>

                    <h6>Temperature: ${(weatherItem.main.temp - 273.15).toFixed(2)}°C</h6>

                    <h6>Wind: ${weatherItem.wind.speed} M/S</h6>

                    <h6>Humidity: ${weatherItem.main.humidity}%</h6>

                </div>

                <div class="icon">

                    <img src="https://openweathermap.org/img/wn/${weatherItem.weather[0].icon}@4x.png" alt="weather-icon">

                    <h6>${weatherItem.weather[0].description}</h6>

                </div>`;

    } else { // HTML for the other five day forecast card

        return `<li class="card">

                    <h3>(${weatherItem.dt\_txt.split(" ")[0]})</h3>

                    <img src="https://openweathermap.org/img/wn/${weatherItem.weather[0].icon}@4x.png" alt="weather-icon">

                    <h6>Temp: ${(weatherItem.main.temp - 273.15).toFixed(2)}°C</h6>

                    <h6>Wind: ${weatherItem.wind.speed} M/S</h6>

                    <h6>Humidity: ${weatherItem.main.humidity}%</h6>

                </li>`;

    }

}

const getWeatherDetails = (cityName, latitude, longitude) => {

    const WEATHER\_API\_URL = `https://api.openweathermap.org/data/2.5/forecast?lat=${latitude}&lon=${longitude}&appid=${API\_KEY}`;

    fetch(WEATHER\_API\_URL).then(response => response.json()).then(data => {

        // Filter the forecasts to get only one forecast per day

        const uniqueForecastDays = [];

        const fiveDaysForecast = data.list.filter(forecast => {

            const forecastDate = new Date(forecast.dt\_txt).getDate();

            if (!uniqueForecastDays.includes(forecastDate)) {

                return uniqueForecastDays.push(forecastDate);

            }

        });

        // Clearing previous weather data

        cityInput.value = "";

        currentWeatherDiv.innerHTML = "";

        weatherCardsDiv.innerHTML = "";

        // Creating weather cards and adding them to the DOM

        fiveDaysForecast.forEach((weatherItem, index) => {

            const html = createWeatherCard(cityName, weatherItem, index);

            if (index === 0) {

                currentWeatherDiv.insertAdjacentHTML("beforeend", html);

            } else {

                weatherCardsDiv.insertAdjacentHTML("beforeend", html);

            }

        });

    }).catch(() => {

        alert("An error occurred while fetching the weather forecast!");

    });

}

const getCityCoordinates = () => {

    const cityName = cityInput.value.trim();

    if (cityName === "") return;

    const API\_URL = `https://api.openweathermap.org/geo/1.0/direct?q=${cityName}&limit=1&appid=${API\_KEY}`;

    // Get entered city coordinates (latitude, longitude, and name) from the API response

    fetch(API\_URL).then(response => response.json()).then(data => {

        if (!data.length) return alert(`No coordinates found for ${cityName}`);

        const { lat, lon, name } = data[0];

        getWeatherDetails(name, lat, lon);

    }).catch(() => {

        alert("An error occurred while fetching the coordinates!");

    });

}

const getUserCoordinates = () => {

    navigator.geolocation.getCurrentPosition(

        position => {

            const { latitude, longitude } = position.coords; // Get coordinates of user location

            // Get city name from coordinates using reverse geocoding API

            const API\_URL = `https://api.openweathermap.org/geo/1.0/reverse?lat=${latitude}&lon=${longitude}&limit=1&appid=${API\_KEY}`;

            fetch(API\_URL).then(response => response.json()).then(data => {

                const { name } = data[0];

                getWeatherDetails(name, latitude, longitude);

            }).catch(() => {

                alert("An error occurred while fetching the city name!");

            });

        },

        error => { // Show alert if user denied the location permission

            if (error.code === error.PERMISSION\_DENIED) {

                alert("Geolocation request denied. Please reset location permission to grant access again.");

            } else {

                alert("Geolocation request error. Please reset location permission.");

            }

        });

}

locationButton.addEventListener("click", getUserCoordinates);

searchButton.addEventListener("click", getCityCoordinates);

cityInput.addEventListener("keyup", e => e.key === "Enter" && getCityCoordinates());

Results:



