<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8" />

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

<title>Weather Now</title>

<style>

\* {

margin: 0;

padding: 0;

box-sizing: border-box;

font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

}

body {

background: linear-gradient(to right, #83a4d4, #b6fbff);

color: #333;

display: flex;

justify-content: center;

align-items: center;

min-height: 100vh;

}

.weather-container {

background: white;

padding: 2rem;

border-radius: 20px;

box-shadow: 0 8px 20px rgba(0, 0, 0, 0.1);

text-align: center;

width: 400px;

position: relative;

}

h1 {

margin-bottom: 1rem;

}

.location {

font-size: 1.5rem;

font-weight: bold;

}

.temperature {

font-size: 3rem;

font-weight: bold;

margin: 0.5rem 0;

}

.condition {

font-size: 1.2rem;

color: #555;

}

.forecast {

margin-top: 1.5rem;

text-align: left;

}

.forecast-day {

margin: 0.5rem 0;

}

.loading {

font-size: 1.2rem;

color: #999;

}

.controls {

margin-top: 1rem;

}

button, input[type="text"] {

padding: 0.5rem 1rem;

margin: 0.5rem;

border: none;

border-radius: 10px;

background-color: #3399ff;

color: white;

cursor: pointer;

font-size: 1rem;

transition: background-color 0.3s;

}

button:hover {

background-color: #267acc;

}

input[type="text"] {

background-color: #f0f8ff;

color: #333;

border: 1px solid #ccc;

}

</style>

</head>

<body>

<div class="weather-container">

<h1>Weather Now</h1>

<div class="loading">Fetching your location and weather...</div>

<div class="location" id="location"></div>

<div class="temperature" id="temperature"></div>

<div class="condition" id="condition"></div>

<img id="icon" alt="Weather Icon" style="width:80px; height:80px; margin-top:10px;" />

<div class="forecast" id="forecast"></div>

<div class="controls">

<input type="text" id="cityInput" placeholder="Enter city name...">

<button onclick="fetchByCity()">Search</button>

<button onclick="getLocation()">Use My Location</button>

</div>

</div>

<script>

const API\_KEY = '44df115749834aefa20890653385c4f1';

let currentUnit = 'C';

let lastCurrentData = null;

let lastForecastData = null;

function convertTemp(temp, toUnit) {

return toUnit === 'F'

? (temp \* 9 / 5) + 32

: (temp - 32) \* 5 / 9;

}

function updateBackground(condition) {

const body = document.body;

const cond = condition.toLowerCase();

if (cond.includes('sunny') || cond.includes('clear')) {

body.style.background = 'linear-gradient(to right, #f9d423, #ff4e50)';

} else if (cond.includes('cloud')) {

body.style.background = 'linear-gradient(to right, #bdc3c7, #2c3e50)';

} else if (cond.includes('rain') || cond.includes('drizzle')) {

body.style.background = 'linear-gradient(to right, #4b79a1, #283e51)';

} else if (cond.includes('snow')) {

body.style.background = 'linear-gradient(to right, #e6dada, #274046)';

} else if (cond.includes('thunder') || cond.includes('storm')) {

body.style.background = 'linear-gradient(to right, #373B44, #4286f4)';

} else if (cond.includes('mist') || cond.includes('fog') || cond.includes('haze')) {

body.style.background = 'linear-gradient(to right, #757F9A, #D7DDE8)';

} else {

body.style.background = 'linear-gradient(to right, #83a4d4, #b6fbff)';

}

}

function updateUI(current, forecast) {

lastCurrentData = current;

lastForecastData = forecast;

document.querySelector('.loading').style.display = 'none';

const isF = currentUnit === 'F';

const temp = isF ? convertTemp(current.temp, 'F').toFixed(1) : current.temp.toFixed(1);

document.getElementById('location').textContent = `${current.city\_name}, ${current.country\_code}`;

document.getElementById('temperature').textContent = `${temp}°${currentUnit}`;

document.getElementById('condition').textContent = current.weather.description;

updateBackground(current.weather.description);

const iconCode = current.weather.icon;

document.getElementById('icon').src = `https://www.weatherbit.io/static/img/icons/${iconCode}.png`;

const forecastContainer = document.getElementById('forecast');

forecastContainer.innerHTML = '<h3>Next Days:</h3>';

for (let i = 1; i < 6; i++) {

const day = forecast[i];

const date = day.valid\_date;

const avgTemp = isF ? convertTemp(day.temp, 'F').toFixed(1) : day.temp.toFixed(1);

const desc = day.weather.description;

const icon = day.weather.icon;

forecastContainer.innerHTML += `

<div class="forecast-day">

<strong>${date}</strong>: ${avgTemp}°${currentUnit} - ${desc}

<img src="https://www.weatherbit.io/static/img/icons/${icon}.png" style="width:30px; vertical-align:middle;" />

</div>

`;

}

}

function fetchWeather(query) {

const currentURL = `https://api.weatherbit.io/v2.0/current?key=${API\_KEY}&city=${query}`;

const forecastURL = `https://api.weatherbit.io/v2.0/forecast/daily?key=${API\_KEY}&city=${query}&days=6`;

Promise.all([

fetch(currentURL).then(res => res.json()),

fetch(forecastURL).then(res => res.json())

]).then(([currentRes, forecastRes]) => {

if (!currentRes.data || !forecastRes.data) {

document.querySelector('.loading').textContent = 'Invalid location or API error.';

} else {

updateUI(currentRes.data[0], forecastRes.data.data);

}

}).catch(() => {

document.querySelector('.loading').textContent = 'Failed to fetch weather data.';

});

}

function fetchByCity() {

const city = document.getElementById('cityInput').value;

if (!city) return;

document.querySelector('.loading').style.display = 'block';

document.getElementById('location').textContent = '';

document.getElementById('temperature').textContent = '';

document.getElementById('condition').textContent = '';

document.getElementById('forecast').innerHTML = '';

fetchWeather(city);

}

function getLocation() {

if (navigator.geolocation) {

navigator.geolocation.getCurrentPosition(position => {

const { latitude, longitude } = position.coords;

const currentURL = `https://api.weatherbit.io/v2.0/current?key=${API\_KEY}&lat=${latitude}&lon=${longitude}`;

const forecastURL = `https://api.weatherbit.io/v2.0/forecast/daily?key=${API\_KEY}&lat=${latitude}&lon=${longitude}&days=6`;

Promise.all([

fetch(currentURL).then(res => res.json()),

fetch(forecastURL).then(res => res.json())

]).then(([currentRes, forecastRes]) => {

if (!currentRes.data || !forecastRes.data) {

document.querySelector('.loading').textContent = 'Location error or API failure.';

} else {

updateUI(currentRes.data[0], forecastRes.data.data);

}

}).catch(() => {

document.querySelector('.loading').textContent = 'Failed to fetch weather data.';

});

}, () => {

document.querySelector('.loading').textContent = 'Location access denied or failed.';

});

} else {

document.querySelector('.loading').textContent = 'Geolocation not supported.';

}

}

window.addEventListener('load', getLocation);

</script>

</body>

</html>