Portfolio Website Project

**Course: UI/UX Design Fundamentals**

Instructor Name: Dhiraj Sir

Date of Submission: 13/08/2025

Submitted By:  
GOKULAKRISHNAN - 2462075  
GEORGE THOMAS - 2462072  
ABHISHEK KS - 2462013

# Abstract

This project involves the design and development of a weather application using HTML, CSS, and JavaScript, integrated with the OpenWeatherMap API. The key goal was to create a user-friendly, responsive, and visually appealing interface that allows users to search for real-time weather information by city name. The app utilizes asynchronous API calls to retrieve temperature, weather descriptions, and icons dynamically. The final outcome is a functional, accessible, and adaptable weather app that can be expanded with advanced features in the future.

# Objectives

• Design a modern, user-friendly weather application interface.

• Implement API integration to fetch live weather data.

• Ensure cross-device responsiveness using CSS media queries.

• Apply clean code practices and maintainable structure.

• Present data in an intuitive and visually appealing manner.

# Scope of the Project

The project focuses on the development of a front-end weather application using HTML, CSS, and JavaScript. It integrates the OpenWeatherMap API to fetch real-time data based on user input. The scope includes responsive design for desktop and mobile devices, semantic HTML structure, and CSS styling for branding and readability. Server-side integration or persistent storage is outside the scope.

# Tools & Technologies Used

HTML5 - For structuring the application content.

CSS3 - For styling and layout management.

JavaScript (ES6) - For API integration and dynamic content updates.

OpenWeatherMap API - To fetch real-time weather data.

VS Code - Code editing environment.

Chrome DevTools - For debugging and testing.

# HTML Structure Overview

• Semantic tags such as <header>, <main>, <section>, and <footer> are used.

• Search input and button for city-based queries.

• Weather display section including city name, temperature, description, and icon.

# CSS Styling Strategy

• External CSS file for styling.

• Flexbox for centering and layout alignment.

• Media queries for mobile responsiveness.

• Smooth hover effects and transitions.

• Color gradients for background aesthetics.

# Key Features

Responsive Design - Works seamlessly across devices.

Live Weather Data - Real-time updates from OpenWeatherMap API.

Weather Icons - Visually represent weather conditions.

User-Friendly Interface - Simple input and clean layout.

# Challenges Faced & Solutions

Challenge: Handling API errors for invalid city names.  
Solution: Added error handling with user-friendly alerts.

Challenge: Responsive layout alignment.  
Solution: Used CSS Flexbox and media queries to adjust elements dynamically.

# Outcome

The weather app successfully retrieves and displays live weather data based on user input. The responsive design ensures usability across devices, and the integration with OpenWeatherMap API provides accurate and up-to-date information. The project improved understanding of API handling, asynchronous JavaScript, and responsive design techniques.

# Future Enhancements

• Add geolocation-based automatic weather detection.

• Implement dark mode theme toggle.

• Add extended forecasts and historical data.

• Improve accessibility with ARIA labels.

# Sample Code

<!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>

<style>

body {

font-family: Arial, sans-serif;

background: linear-gradient(to bottom, #4facfe, #00f2fe);

color: white;

text-align: center;

margin: 0;

padding: 0;

}

.container {

margin-top: 100px;

}

input {

padding: 10px;

border: none;

border-radius: 5px;

width: 250px;

margin-right: 10px;

}

button {

padding: 10px 15px;

border: none;

border-radius: 5px;

background: #0078ff;

color: white;

cursor: pointer;

}

button:hover {

background: #005fcc;

}

.weather {

margin-top: 30px;

display: none;

}

.weather img {

width: 100px;

}

</style>

</head>

<body>

<div class="container">

<h1>🌦 Weather App</h1>

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

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

<div class="weather" id="weatherBox">

<h2 id="cityName"></h2>

<img id="weatherIcon" alt="Weather Icon">

<p id="temperature"></p>

<p id="description"></p>

</div>

</div>

<script>

async function getWeather() {

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

const apiKey = "c3391a7ef3a165e68accf3a26d78ce56";

const url = `https://api.openweathermap.org/data/2.5/weather?q=${city}&appid=${apiKey}&units=metric`;

try {

const response = await fetch(url);

if (!response.ok) throw new Error("City not found");

const data = await response.json();

document.getElementById('cityName').textContent = `${data.name}, ${data.sys.country}`;

document.getElementById('temperature').textContent = `🌡 ${data.main.temp}°C`;

document.getElementById('description').textContent = `📝 ${data.weather[0].description}`;

document.getElementById('weatherIcon').src = `https://openweathermap.org/img/wn/${data.weather[0].icon}@2x.png`;

document.getElementById('weatherBox').style.display = "block";

} catch (error) {

alert(error.message);

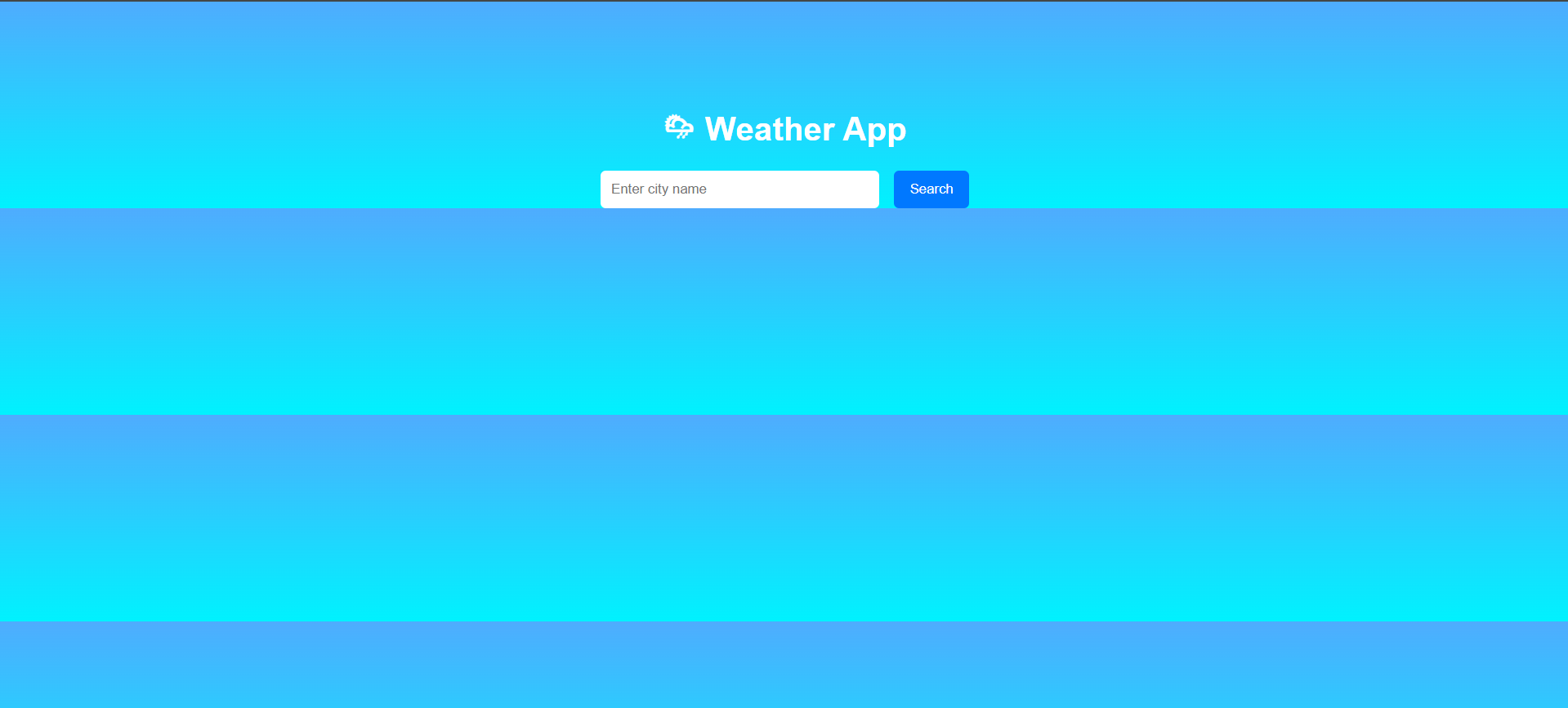
}

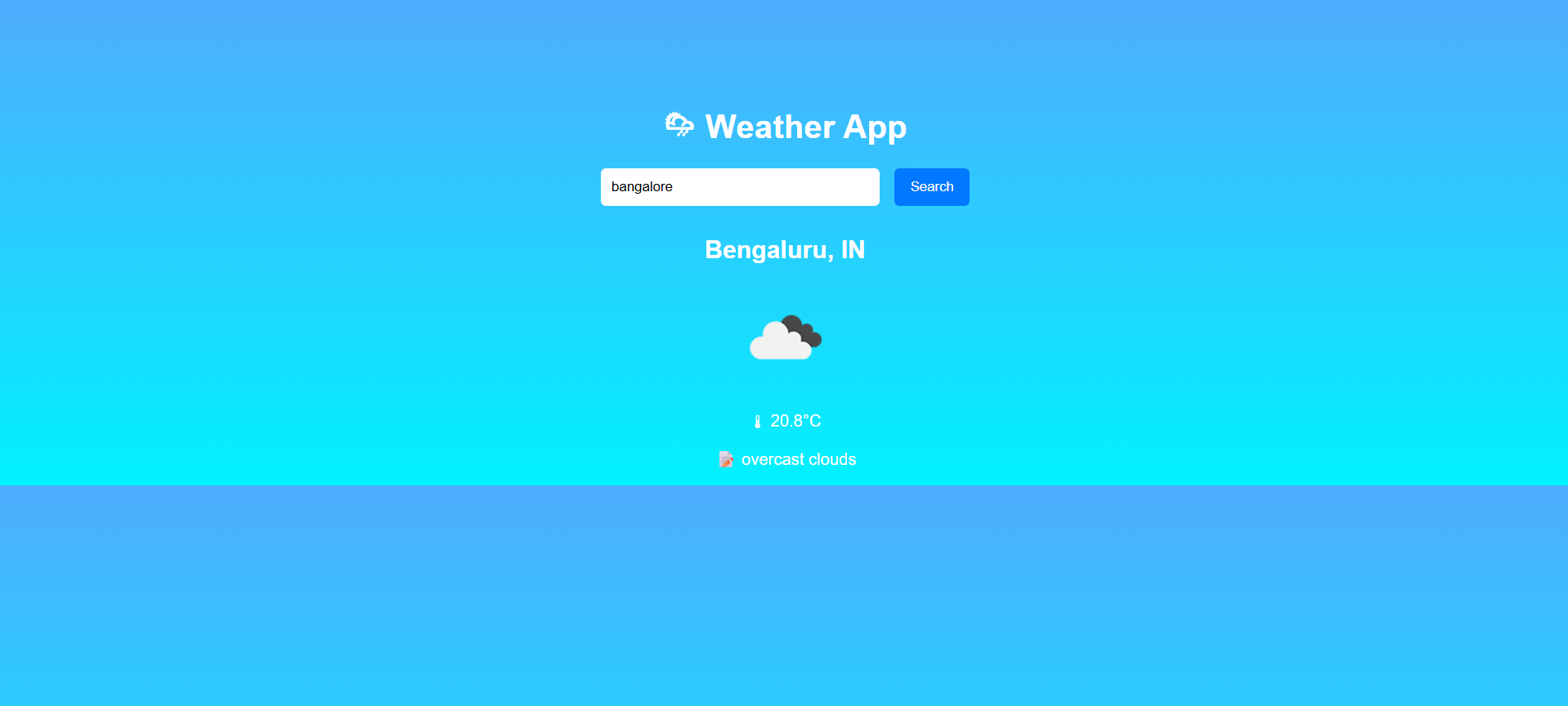
}

</script>

</body></html>

# Screenshots of Final Output





# Conclusion

The Weather App project served as a comprehensive exercise in front-end development, API integration, and responsive UI design. It demonstrated how HTML, CSS, and JavaScript can work together to produce a practical, interactive, and visually appealing application. The knowledge gained will be valuable for future projects involving dynamic data fetching and presentation.

# References

OpenWeatherMap API: https://openweathermap.org/api

MDN Web Docs: https://developer.mozilla.org/

W3Schools: https://www.w3schools.com/