Capstone Project

**Course code:** CSA4385

**Course:** Internet Programming for Green App

**Team:** 03

**Name:**

1. P. Radha Devi

2. D. Siddartha Reddy

**Registration no:**

192211652

192211651

**Slot:** B

**Title:** “Weather App in HTML CSS and JavaScript”

**Guided by:** Venkatesan sir

**Department:** CSE

**College:** Saveetha School of Engineering



**Objectives**

**1. Develop a User-Friendly Interface**

* **HTML & CSS:**
  + Create a responsive layout using HTML5 structure with semantic elements (e.g., <header>, <main>, <footer>).
  + Use CSS for styling, ensuring readability and accessibility of weather information.
  + Implement responsive design principles (e.g., media queries) to adapt the layout for different screen sizes.

**2. Implement Dynamic Functionality with JavaScript**

* **JavaScript:**
  + Use JavaScript to fetch weather data from a weather API (e.g., OpenWeatherMap, Weatherstack).
  + Display weather information dynamically, such as current weather conditions, forecasts, and temperature trends.
  + Enhance user interaction with features like toggling between Fahrenheit and Celsius, dynamic weather icons, and interactive elements for user preferences (e.g., favorite locations).

**3. Build a Robust Back-End with PHP (optional for a basic weather app)**

* **PHP:**
  + Although a simple weather app typically doesn't require PHP for backend operations (as it can directly interact with APIs), you might use PHP for server-side caching or proxying API requests if necessary.

**4. Database Management with MySQL (not typically needed for a basic weather app)**

* **MySQL:**
  + Usually not required for a weather app unless you plan to store historical weather data or user preferences over time.

**5. Ensure Data Security and Validation**

* **Security Measures:**
  + Validate user inputs on the client-side (JavaScript) to prevent basic errors.
  + Ensure secure API usage (HTTPS) and handle sensitive data (if any) securely.

**6. Facilitate Admin and User Roles (usually not applicable for a basic weather app)**

* **Admin/User Roles:**
  + Typically not necessary for a weather app unless it includes user accounts or advanced features like saved locations.

**7. Provide Responsive and Accessible Design**

* **Responsive Design:**
  + Ensure the weather app is usable across devices (desktops, tablets, smartphones) by testing and adjusting layouts as needed.
* **Accessibility:**
  + Follow accessibility standards (WCAG) for color contrast, keyboard navigation, and screen reader compatibility.

**8. Integrate External APIs and Services**

* **Weather API Integration:**
  + Choose a reliable weather API and integrate it into your JavaScript code for fetching weather data.
  + Consider additional APIs for features like geolocation (to automatically detect user location) or historical weather data (if needed).

**Example Workflow**

1. **Front-End Development:**
   * Set up your HTML structure with placeholders for weather data.
   * Style the interface using CSS, focusing on responsiveness and readability.
2. **Dynamic Functionality:**
   * Use JavaScript to fetch weather data from the chosen API.
   * Display current weather conditions and forecasts dynamically based on API responses.
3. **Security and Validation:**
   * Implement basic client-side validation for user inputs (if any).
4. **API Integration:**
   * Integrate a weather API by sending requests and handling responses (JSON format).
5. **Accessibility and Responsive Design:**
   * Test the app across devices and ensure it meets accessibility standards.
   * Use media queries and flexible layouts for responsiveness.
6. **Deployment:**
   * Host your application on a web server (e.g., GitHub Pages, Netlify) for public access.

By following this approach, you can create a weather application that is user-friendly, dynamic, and integrates external data sources effectively using HTML, CSS, and JavaScript. Adjust the complexity based on additional features like user accounts or historical data storage if required.

Top of Form

Bottom of Form

**Keywords**

* Pet Adoption
* Web Development
* HTML
* JavaScript
* PHP
* MySQL
* User Interface (UI)
* User Experience (UX)
* Responsive Design
* Form Handling
* Database Management
* Server-Side Scripting
* Client-Side Scripting
* API Integration
* Web Application
* **Introduction**

Imagine a weather application that not only keeps you informed but does so with elegance and efficiency. Our weather application, meticulously crafted using HTML, CSS, and JavaScript, brings you the latest weather updates with ease and style. Whether you're a frequent traveler, a weather enthusiast, or simply someone who likes to stay prepared, our application caters to your needs seamlessly. Through a thoughtfully designed user interface, you can explore current weather conditions and detailed forecasts for any location worldwide.

Powered by reliable APIs, our application ensures you receive real-time data that you can trust. The interface is not just functional but also aesthetically pleasing, with responsive design principles ensuring it looks great and operates flawlessly across devices of all sizes—from desktops to tablets and smartphones. Interactive features, driven by JavaScript, allow you to customize your experience, whether it's toggling between Celsius and Fahrenheit, setting favorite locations for quick access, or viewing weather trends over time. This application isn't just about checking the weather; it's about enhancing your understanding and enjoyment of weather information through intuitive design and powerful technology.

Join us in exploring weather in a whole new way—simple, insightful, and always accessible.HTML (Hypertext Markup Language) will form the backbone of the website, providing structure and content. JavaScript will enhance the user experience by adding dynamic interactions and real-time features, such as instant form validation and interactive pet galleries. PHP (Hypertext Preprocessor) will be employed for server-side scripting, managing data interactions, processing user inputs, and communicating with a MySQL database to handle the backend operations.

Our platform aims to modernize the pet adoption process by bridging the gap between prospective pet owners and animals in need of homes. With a dedicated backend for managing pet data and adoption applications, administrators can efficiently oversee listings and applications. Meanwhile, the user-centric design will empower adopters to find their ideal pets with ease. By integrating best practices in web development and focusing on usability, we aspire to enhance the adoption experience, promote animal welfare, and connect more pets with loving homes.

* **Weather App (JavaScript):**

const url =

'https://api.openweathermap.org/data/2.5/weather';

const apiKey =

'f00c38e0279b7bc85480c3fe775d518c';

$(document).ready(function () {

weatherFn('Pune');

});

async function weatherFn(cName) {

const temp =

`${url}?q=${cName}&appid=${apiKey}&units=metric`;

try {

const res = await fetch(temp);

const data = await res.json();

if (res.ok) {

weatherShowFn(data);

} else {

alert('City not found. Please try again.');

}

} catch (error) {

console.error('Error fetching weather data:', error);

}

}

function weatherShowFn(data) {

$('#city-name').text(data.name);

$('#date').text(moment().

format('MMMM Do YYYY, h:mm:ss a'));

$('#temperature').

html(`${data.main.temp}°C`);

$('#description').

text(data.weather[0].description);

$('#wind-speed').

html(`Wind Speed: ${data.wind.speed} m/s`);

$('#weather-icon').

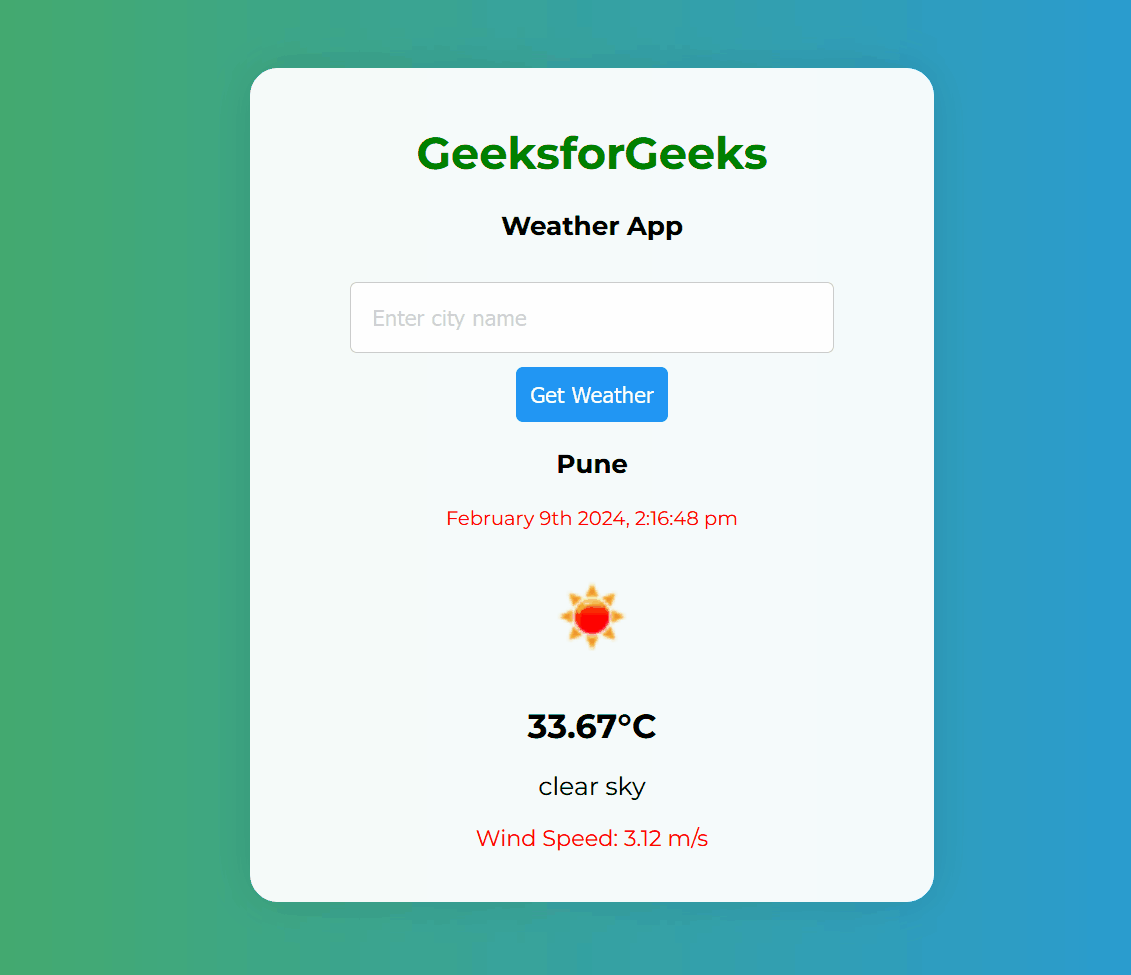
attr('src',

`...`);

$('#weather-info').fadeIn();

}

* **OUTPUT:**



* **Weather App (HTML)**

<!DOCTYPE html>

<head>

<link rel="stylesheet" href="style2.css">

<link rel="stylesheet" href=

"https://cdnjs.cloudflare.com/ajax/libs/animate.css/4.1.1/animate.min.css">

<link rel="stylesheet" href=

"https://cdnjs.cloudflare.com/ajax/libs/font-awesome/5.15.1/css/all.min.css">

<link rel="stylesheet" href=

"https://fonts.googleapis.com/css2?family=Montserrat:wght@400;700&display=swap">

<title>GFG App</title>

</head>

<body>

<div class="container">

<div class="weather-card">

<h1 style="color: green;">

GeeksforGeeks

</h1>

<h3>

Weather App

</h3>

<input type="text" id="city-input"

placeholder="Enter city name">

<button id="city-input-btn"

onclick="weatherFn($('#city-input').val())">

Get Weather

</button>

<div id="weather-info"

class="animate\_\_animated animate\_\_fadeIn">

<h3 id="city-name"></h3>

<p id="date"></p>

<img id="weather-icon" src="" alt="Weather Icon">

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

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

<p id="wind-speed"></p>

</div>

</div>

</div>

<script src=

"https://code.jquery.com/jquery-3.6.0.min.js">

</script>

<script src=

"https://momentjs.com/downloads/moment.min.js">

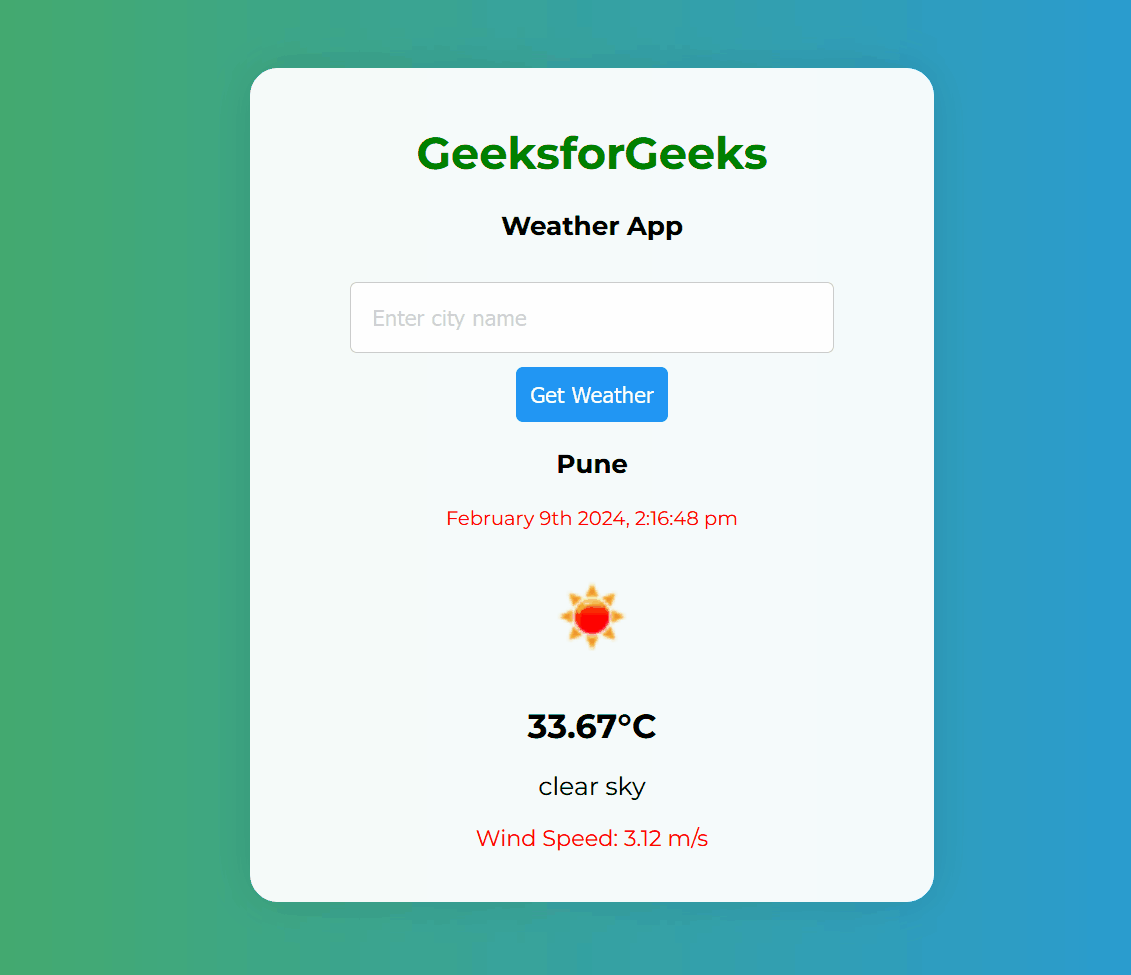
</script>

<script src="script2.js"></script>

</body>

</html>

* **OUTPUT:**



* **Weather App (CSS):**

body {

margin: 0;

font-family: 'Montserrat', sans-serif;

display: flex;

justify-content: center;

align-items: center;

height: 100vh;

background: linear-gradient(to right, #4CAF50, #2196F3);

}

.container {

text-align: center;

}

.weather-card {

background-color: rgba(255, 255, 255, 0.95);

border-radius: 20px;

padding: 20px;

box-shadow: 0 0 30px rgba(0, 0, 0, 0.1);

transition: transform 0.3s ease-in-out;

width: 450px;

}

.weather-card:hover {

transform: scale(1.05);

}

#city-input {

padding: 15px;

margin: 10px 0;

width: 70%;

border: 1px solid #ccc;

border-radius: 5px;

font-size: 16px;

}

#city-input:focus {

outline: none;

border-color: #2196F3;

}

#city-input::placeholder {

color: #aaa;

}

#city-input-btn {

padding: 10px;

background-color: #2196F3;

color: #fff;

border: none;

border-radius: 5px;

font-size: 16px;

cursor: pointer;

}

#city-input-btn:hover {

background-color: #1565C0;

}

#weather-info {

display: none;

}

#weather-icon {

width: 100px;

height: 100px;

}

#temperature {

font-size: 24px;

font-weight: bold;

margin: 8px 0;

}

#description {

font-size: 18px;

margin-bottom: 10px;

}

#wind-speed {

font-size: 16px;

color: rgb(255, 0, 0);

}

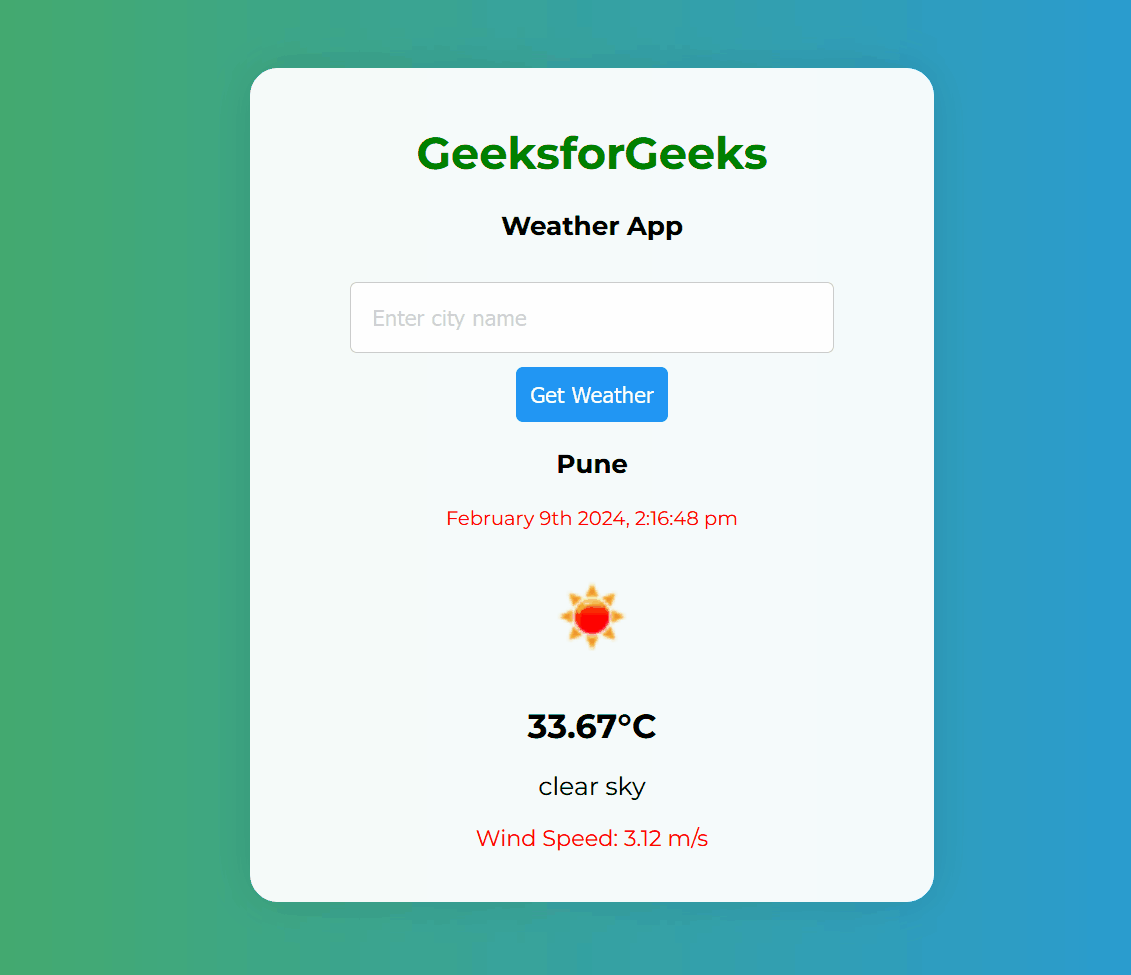
#date {

font-size: 14px;

color: rgb(255, 0, 0);

}

* **OUTPUT:**

****

* **CONCLUSION:**

In conclusion, our weather application represents the synergy of modern web technologies—HTML, CSS, and JavaScript—to deliver a streamlined and insightful weather experience. Whether you're checking the forecast for a weekend getaway, planning your daily activities, or simply staying informed, our application ensures you have access to accurate and up-to-date weather information at your fingertips. With a user-friendly interface designed for responsiveness across devices, you can rely on our application to provide not only convenience but also aesthetic appeal.

* Top of Form
* Bottom of Form