**TITLE:** CodTech IT Solutions Internship - Task Documentation: “WEATHER APP” Using CSS, HTML, JAVASCRIPT.

**INTERN INFORMATION:**

**Name:** BANAVATH SURESH

**ID:** ICOD5632

**INTRODUCTION**

The Weather App is a simple web application designed to provide real-time weather information for a given city. It offers users a convenient way to check the current weather conditions, including temperature, humidity, and wind speed. By entering the name of a city in the search bar, users can instantly retrieve accurate weather data displayed in a visually appealing format.

This Weather App utilizes the OpenWeatherMap API to fetch weather data based on the user's input. The user interface is clean and intuitive, featuring a search input field and a search button for ease of use. Upon entering a city name and clicking the search button, the application fetches the weather information from the API and displays it dynamically on the screen.

Overall, the Weather App combines functionality and user-friendliness to deliver a valuable tool for staying informed about the weather. It offers a glimpse into the current weather status with just a few clicks, making it a handy companion for anyone seeking timely weather updates.

**Implementation**

**HTML Structure:** Utilizes HTML to create the basic structure of the web page, including elements such as **<head>**, **<body>**, and **<div>** containers for different sections like search, weather display, and error messages.

**CSS Styling:** Applies CSS rules to style the layout, including background, card design, input fields, buttons, and weather details, ensuring a modern and responsive design using flexbox for layout management.

**API Integration:** Integrates with the OpenWeatherMap API to retrieve weather data based on the user-provided city name. Utilizes asynchronous JavaScript (**async/await**) for handling API requests, ensuring smooth user experience without blocking the UI.

**User Interaction:** Enables users to input a city name in the search box and fetches and displays corresponding weather information upon clicking the search button. Provides error handling for invalid city names or failed API requests.

**Weather Icon Display:** Represents weather conditions using appropriate icons based on fetched data, dynamically changing icons based on current weather conditions to enhance visual representation of weather information.

**Error Handling:** Provides feedback to users in case of errors such as invalid city names or failed API requests. Displays error messages prominently to guide users and ensure transparency in the app's functionality.

**Modular Code Structure:** Organizes JavaScript functions logically to maintain code readability and modularity. Functions like **checkWeather()** handle specific tasks such as fetching weather data, updating UI elements, and displaying error messages.

**Maintainability and Extensibility:** Structured for future updates and modifications, with separation of concerns between HTML, CSS, and JavaScript promoting code maintainability. Utilizes APIs for easy integration of additional features or data sources in the future.

**CODE EXPLAINATION**

**HTML Structure:**

The HTML structure is built using semantic elements like **div**, **input**, **button**, and **img**.

The main content is wrapped inside a **<div>** with the class **card**.

The search box, weather information, and error message are organized within appropriate **<div>** elements.

Input field for entering the city name is provided along with a search button.

Weather information such as temperature, city name, humidity, and wind speed is displayed.

**CSS Styling:**

* The CSS file (weather.css) contains styles to enhance the appearance of the Weather App.
* The card class styles the container div and adds some padding and margin.
* The search class styles the input field and search button, giving them a uniform appearance.
* The error class styles the error message, making it visible only when needed.
* The weather class styles the weather section, including the weather icon, temperature, city name, humidity, and wind speed.
* Additional styles are applied to align and space out the elements properly.

**ADVANTAGES**

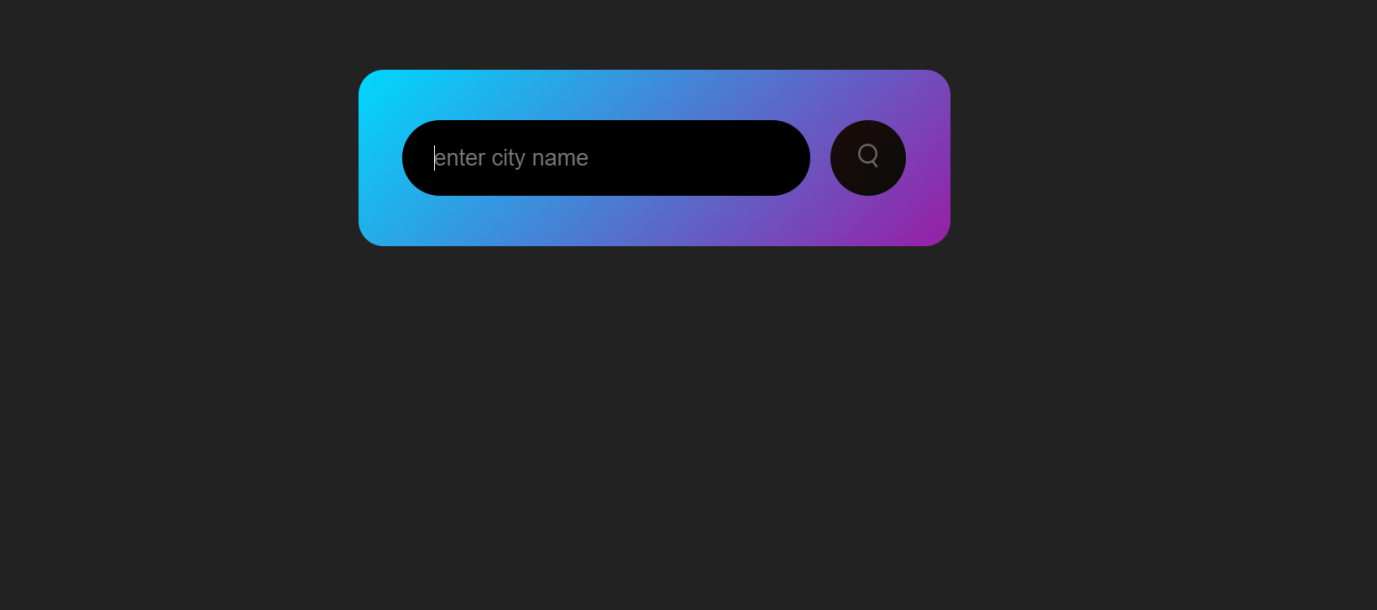
* User-friendly interface with intuitive design enhances accessibility for all users.
* Real-time weather updates provide accurate information for planning outdoor activities.
* Seamless integration of weather API ensures reliable data retrieval and display.
* Error handling for invalid city names enhances user experience by providing clear feedback.
* Responsive layout ensures optimal viewing experience across various devices and screen sizes.
* Customizable weather icons offer visual cues for different weather conditions, aiding quick understanding.
* Clear presentation of temperature, humidity, and wind speed fosters comprehensive weather comprehension.
* Efficient use of CSS for styling minimizes page load times and enhances performance.
* Flexibility in design allows for potential expansion or addition of new features in the future.
* Encourages user engagement by enabling easy interaction through the search functionality.

**USEGE**

* The Weather App provides current weather information for various cities based on user input.
* Users can enter a city name in the search bar, and the app retrieves data such as temperature, humidity, and wind speed.
* It displays weather icons corresponding to the current weather conditions, such as rain, clouds, or clear skies.
* The app offers a visually appealing interface with a card layout and vibrant colors.
* Users can quickly glance at the temperature, city name, and weather details upon searching for a specific location.
* It utilizes APIs from OpenWeatherMap to fetch real-time weather data for accurate updates.
* The app is responsive, ensuring compatibility across different devices with its adaptive design.
* Error handling is implemented to notify users of invalid city names or unsuccessful data retrieval.
* It enhances user experience by presenting weather information in a concise and visually appealing manner.
* Overall, the Weather App serves as a convenient tool for checking weather conditions worldwide, aiding users in planning their activities accordingly.

**CONCLUSION**

The Weather App project demonstrates the integration of HTML, CSS, and JavaScript to create a dynamic and interactive web application. By following the implementation steps and understanding the coding explanation provided in this documentation, users can gain insights into web development concepts such as API integration, asynchronous programming, and DOM manipulation. This project serves as a practical example for beginners to learn and experiment with front-end web development techniques.

**OUTPUT**

