TITLE: CodTech IT Solutions Internship - Task Documentation: “Weather Forecast app” Using CSS, HTML, JAVASCRIPT.

INTERN INFORMATION:

Name: Makili Lakshmi Rani

ID: C0D6252

INTRODUCTION:

My Weather app project is a dynamic web application built using HTML, CSS, and JavaScript. It provides users with real-time weather information for any city worldwide. With a clean and intuitive interface, users can easily sccccsearch for a city, view current weather conditions, and access detailed weather data including temperature, humidity, and wind speed. The app fetches data from the OpenWeatherMap API, ensuring accurate and up-to date forecasts. Stylish design elements and responsive layout enhance user experience across devices. This project showcases the seamless integration of front-end technologies to deliver a practical and user-friendly weather forecasting solution.

Implementation:

- JavaScript Framework: Utilize a modern JavaScript framework for building the frontend application.
- HTML/CSS: Use HTML5 and CSS3 for structuring and styling the user interface, ensuring compatibility with various web browsers.
- Responsive Design: Implement responsive design principles to ensure optimal viewing experience across desktop and mobile devices.
- User Interface Components: Utilize UI libraries for designing interactive and visually appealing components

CODE EXPLAINATION

HTML Structure:

This HTML document represents a weather forecast web application with a simple and intuitive user interface. It consists of a card layout containing sections for search input, error message 2 display, weather information display, and weather details. The user can input a city name in the search bar and click the search button to retrieve the weather data for that city. The weather data includes temperature, city name, humidity, and wind speed, which are dynamically updated on the card. Additionally, weather icons change based on the weather

condition. The application fetches weather data from the OpenWeatherMap API using JavaScript's fetch() method, and it handles error messages for invalid city names. The CSS stylesheet provides styling for layout, fonts, colors, and responsiveness, enhancing the visual appeal and user experience of the application.

CSS Styling:

1. Universal Reset : Sets margin, padding, and font-family to zero for all elements and applies box-sizing: border-box to ensure consistent box models.
2. Body Background : Applies a background image to the body with cover size and no-repeat, creating a visually appealing background for the entire page.
3. Card Styling : Styles the `.card` class to create a card-like container with rounded corners, gradient background, and white text color.
4. Search Bar : Styles the search bar with a flexible input field and a circular button for searching. Aligns items and justifies space between them within the `.search` class.
5. Weather Icon : Specifies the width and margin-top for the weather icon, ensuring it's visually prominent and positioned correctly within the card.
6. Weather Information : Defines font sizes and margin adjustments for the temperature and city name within the `.weather h1` and `.weather h2` classes.
7. Details Section : Styles the details section containing humidity and wind speed information, arranging items in a flex container with space between them.
8. Column Styling : Defines the appearance of each column within the details section, including the icon width, margin, and font size for humidity and wind speed.
9. Hidden Elements : Hides the weather information and error message by default using `display: none` to only reveal them when necessary.
10. Error Message : Styles the error message with left margin, font size, and color, ensuring it's legible and visually separated from other content.

JavaScript Functionality:

In this weather forecast app, the JavaScript functionality is responsible for fetching weather data from the OpenWeatherMap API based on the user's input, updating the DOM elements

with the retrieved data, and handling errors. 1. API Integration : It integrates with the OpenWeatherMap API using the provided API key to fetch weather data for the specified city. 2. Event Listener : It adds an event listener to the search button, so when the button is clicked, it triggers a function to fetch weather data based on the city entered by the user. 3. Fetch Data : It uses the Fetch API to make a GET request to the OpenWeatherMap API endpoint with the specified city and API key. 4. Response Handling : It checks the response status. If the status is 404 (Not Found), it displays an error message indicating an invalid city name. Otherwise, it parses the JSON response and updates the DOM with the weather data. 5. Updating DOM : It updates the DOM elements with the retrieved weather data, including city name, temperature, humidity, wind speed, and weather icon. 6. Weather Icon : It dynamically changes the weather icon based on the weather condition retrieved from the API response. Overall, the JavaScript functionality in this app facilitates dynamic interaction with the API, updates the user interface based on the retrieved data, and provides feedback to the user in case of errors.

USAGE:

Check Current Weather: Users can open the app to instantly see the current weather conditions for their location or any other location they specify.

2. View Detailed Forecast: The app typically provides a detailed forecast for the upcoming hours or days, including temperature, humidity, wind speed, and precipitation.

3. Search for Locations: Users can search for specific locations to check the weather there, whether it's their hometown, a travel destination, or any other place of interest.

4

4. Save Favorite Locations: Many weather apps allow users to save their favorite locations for quick access, especially for frequent travel destinations or places of interest.

5. Receive Weather Alerts: Some apps offer weather alerts for severe weather conditions like storms, hurricanes, or heavy rainfall, helping users stay informed and prepared.

6. Plan Activities: By checking the weather forecast, users can plan outdoor activities accordingly, whether it's a picnic, hike, or sports event, ensuring they choose the right

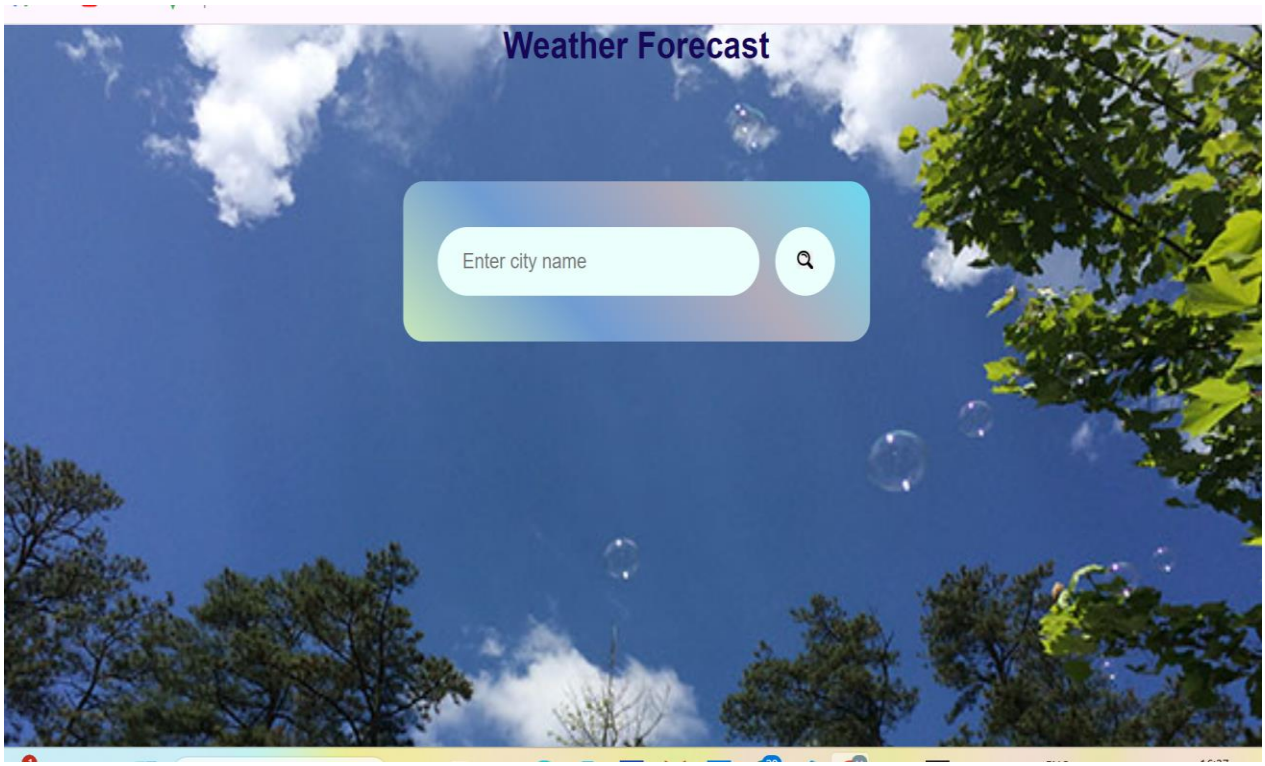
time and place.

7. Track Weather Trends: Users can track weather trends over time, observing changes in temperature, precipitation, or other parameters, which can be useful for various purposes like gardening or climate research.

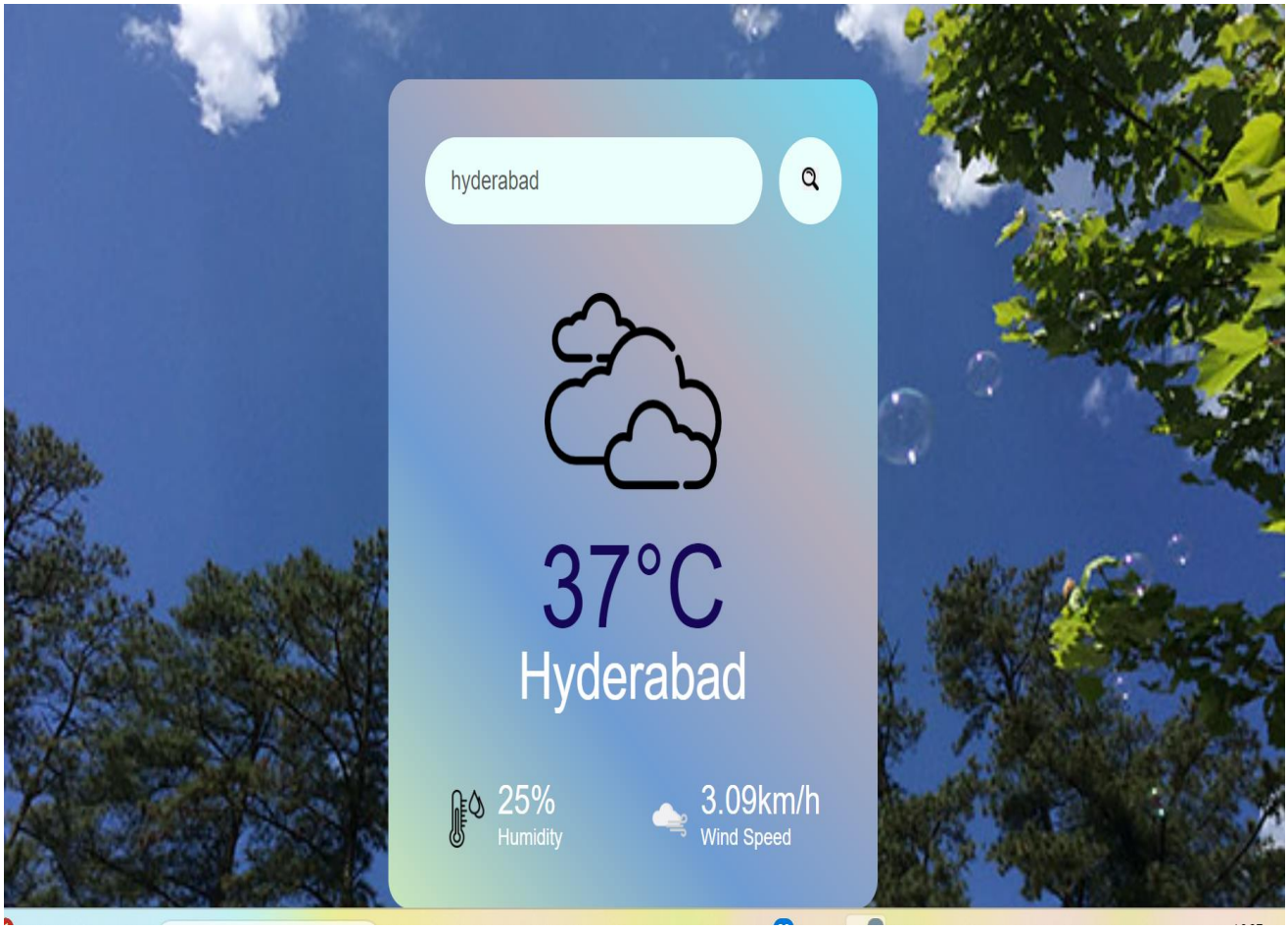
CONCLUSION:

In conclusion, this weather app project demonstrates the integration of HTML, CSS, and JavaScript to create a functional and visually appealing application. By leveraging APIs like OpenWeatherMap, we can fetch real-time weather data and dynamically update the user interface. Through this project, I've honed my skills in asynchronous programming, DOM manipulation, error handling, and responsive design. Moving forward, this project serves as a solid foundation for further exploration and refinement of web development skills.

OUTPUT:



Output : Figure(1)



Output : Figure (2)