Titles

Project Name – Weather Forecast

Build by – Shashikant Sahu Batch –

1st March 2025

Table of Contents

Introduction

The Weather Dashboard Application is a web-based project designed to provide current weather conditions and a 5-day forecast for any searched city. Additionally, users can fetch weather data based on their geolocation and interact with a dropdown menu containing their recently searched cities. The application utilizes the OpenWeatherMap API to gather accurate weather data and presents it through a clean and responsive UI.

This project is an ideal blend of practicality and design for developers and end-users. It's built with web technologies like HTML, CSS, and JavaScript, making it suitable for developers of all skill levels to understand and customize.

Organized File Structure

Creating a well-organized file structure is crucial for readability, maintainability, and collaboration in any web project. Your project is already structured well, and here’s a breakdown of each component:

Description of Files:

1. index.html

- This is the **main entry point** of the application.
- It includes:
  - HTML structure for header, search bar, current weather, and forecast cards.
  - Links to **Tailwind CSS** and **Font Awesome** for styling and icons.
  - References the external JavaScript file (index.js).
  - Uses Tailwind utility classes for **responsive layout**, **spacing**, and **styling**.
  - <meta http-equiv="refresh" content="30"> — auto-refreshes the page every 30 seconds to keep weather data fresh.

2. index.js

- Contains all the **core logic** of the application.
- Responsibilities:
  - Fetch current weather and 5-day forecast using OpenWeatherMap API.
  - Handle geolocation for fetching local weather.
  - Display current weather and forecast dynamically in the UI.
  - Store and retrieve recent city searches using localStorage.
  - Manage user interactions (search, dropdown, location).

3. output.css

- This file contains the **compiled Tailwind CSS** (after running the Tailwind build process).
- Tailwind is a utility-first CSS framework, and this file holds all the utility classes used in your HTML.
- If you're using a Tailwind CLI or PostCSS setup, you'd typically generate this using:

    - npx tailwindcss -i ./src/input.css -o ./output.css –watch

4. README.md

- Markdown file used for **documentation**.
- Should include:
    - Project description
    - Features
    - Technologies used
    - clone links

.

Creativity and Presentation

This section highlights the creative aspects, technologies used, and includes an example of the code implementation along with a project screenshot.
Technologies Used

- **HTML5**: For structuring the UI of the weather dashboard.
- **CSS3**: To style the webpage and improve its responsiveness and visual appearance.
- **JavaScript**: To handle the functionality, API integration, and dynamic updates to the user interface.
- **OpenWeatherMap API**: Used to fetch real-time weather data, including current conditions and 5-day forecasts.
- **Local Storage**: To maintain a history of recently searched cities.

---

1. Technologies Used

Technology	Purpose
HTML5	Webpage structure and layout
Tailwind CSS	Utility-first CSS for responsive and clean design
JavaScript (ES6)	DOM manipulation, API integration, logic
OpenWeatherMap	Source for real-time weather and forecast data
Font Awesome	Weather icons and visual cues
localStorage	Save and load recent city searches
Geolocation API	Fetch user's current location

2. Code

.html code :-

```
<!DOCTYPE html>
<html lang="en">

<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <meta http-equiv="refresh" content="30">
    <title>Weather Forecast</title>
    <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.7.2/css/all.min.css"
        integrity="sha512-
Evv84Mr4kqVGRNSgIGL/F/aIDqQb7xQ2vcrdIwxfjThSH8CSR7PBEakCr51Ck+w+/U6swU2Im1vVX0SVk9ABhg=
="
        crossorigin="anonymous" referrerpolicy="no-referrer" />
    <link rel="stylesheet" href="output.css">
</head>

<body class="bg-gray-100 ">
    <!-- Header Section -->
    <header class="text-center bg-blue-800 p-3 md:p-4 lg:mb-2">
        <h1 class="text-xl md:text-3xl font-bold text-white">Weather Dashboard</h1>
    </header>
    <!-- Main container for the Weather Dashboard -->
    <main class="max-w-5xl mx-auto lg:mt-5 mb-7 p-5 bg-blue-300 shadow-lg lg:rounded-lg">
        <!-- Search Section -->
        <div class="search-list mb-10 mt-10 lg:mt-0">
            <div class="flex flex-col space-y-5 md:space-y-5 lg:space-y-0 md:space-x-3">
                <label for="cityName" class="text-xl lg:text-2xl font-bold">Enter a City Name</label>
                <input type="text" placeholder="E.g., New Delhi, Ranchi, Mumbai"
                    class="city-input w-full p-2 border rounded-lg shadow-sm focus:outline-none focus:ring-2 focus:ring-blue-
500" />
                <div class="flex space-x-5 justify-between lg:mt-3">
                    <button
                        class="search-btn bg-blue-600 text-white px-6 py-2 rounded-lg shadow-lg hover:bg-blue-700 transition
duration-300">
                        Search
                </button>
                <button
                    class="use-current-location-btn bg-gray-500 text-white px-4 py-2 rounded-lg shadow-lg hover:bg-gray-
300 transition duration-300">
                    Use Current Location
                </button>
            </div>
        </div>
    </div>

    <!-- Current Weather Card Section -->
    <div class="current-weather-cards bg-gray-500 text-white p-5 rounded-lg shadow-md mb-8">
        <div class="flex justify-around items-center">
            <div class="mb-6">
                <h2 class="text-2xl font-semibold">London (2024-04-23)</h2>
                <p class="text-lg">Temperature: 7.06°C</p>
                <p>Wind: 2.85 M/S</p>
                <p>Humidity: 88%</p>
            </div>
            <div class="text-5xl text-blue-300">
                <i class="fa-solid fa-cloud-sun-rain"></i>
            </div>
        </div>
    </div>

    <!-- 5-Day Forecast Section -->
    <h3 class="text-center m-4 font-bold text-2xl text-black">5-Day Forecast</h3>
    <div class="weather-cards grid grid-cols-1 md:grid-cols-2 lg:grid-cols-5 gap-4">

        <div class="bg-gray-500 text-white p-4 rounded-lg shadow-sm text-center">
            <p class="text-lg font-semibold">(2024-04-24)</p>
            <i class="fa-solid fa-cloud-moon-rain text-2xl text-blue-300"></i>
            <p>Temp: 5.67°C</p>
            <p>Wind: 2.77 M/S</p>
            <p>Humidity: 76%</p>
        </div>

        <div class="bg-gray-500 text-white p-4 rounded-lg shadow-sm text-center">
```

```
            <p class="text-lg font-semibold">(2024-04-25)</p>
            <i class="fa-solid fa-cloud-moon-rain text-2xl text-blue-300"></i>
            <p>Temp: 5.23°C</p>
            <p>Wind: 1.46 M/S</p>
            <p>Humidity: 71%</p>
          </div>

          <div class="bg-gray-500 text-white p-4 rounded-lg shadow-sm text-center">
            <p class="text-lg font-semibold">(2024-04-26)</p>
            <i class="fa-solid fa-cloud-moon-rain text-2xl text-blue-300"></i>
            <p>Temp: 7.66°C</p>
            <p>Wind: 1.87 M/S</p>
            <p>Humidity: 91%</p>
          </div>

          <div class="bg-gray-500 text-white p-4 rounded-lg shadow-sm text-center">
            <p class="text-lg font-semibold">(2024-04-27)</p>
            <i class="fa-solid fa-cloud-moon-rain text-2xl text-blue-300"></i>
            <p>Temp: 9.09°C</p>
            <p>Wind: 3.77 M/S</p>
            <p>Humidity: 95%</p>
          </div>

          <div class="bg-gray-500 text-white p-4 rounded-lg shadow-sm text-center mb-5 lg:mb-0">
            <p class="text-lg font-semibold">(2024-04-28)</p>
            <i class="fa-solid fa-cloud-showers-heavy text-2xl text-blue-300"></i>
            <p>Temp: 8.46°C</p>
            <p>Wind: 4.23 M/S</p>
            <p>Humidity: 94%</p>
          </div>

        </div>

    </main>
    <!-- Linking the custom JavaScript file -->
    <script src="index.js"></script>
</body>
</html>
```

.js code :-

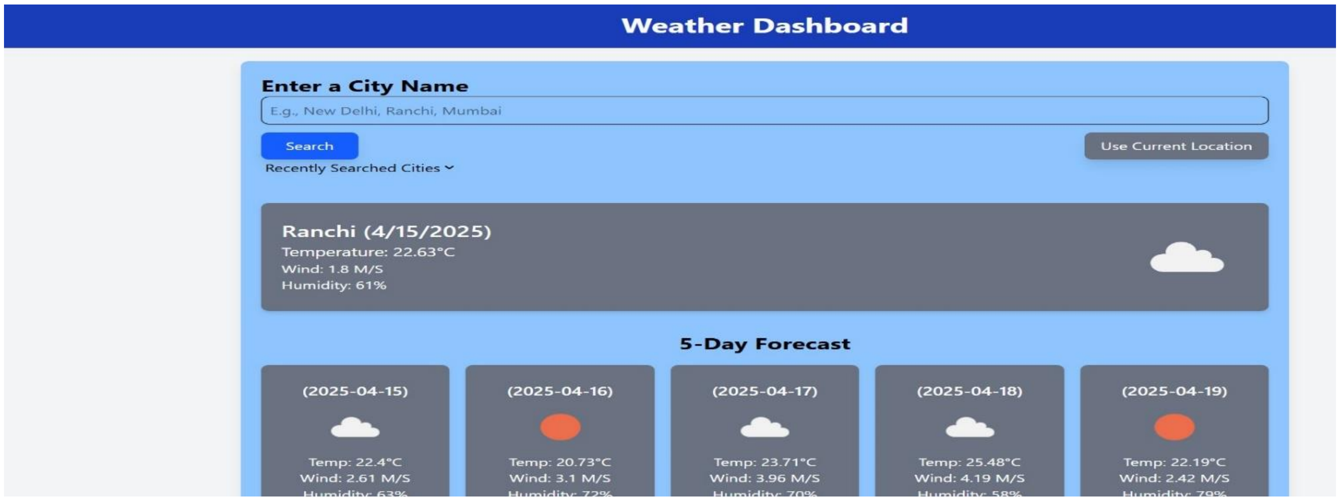
```
const API_KEY = "72e3ad9ab4c8272e73f8fe0223483b84";
const searchBtn = document.querySelector(".search-btn");
const cityInput = document.querySelector(".city-input");
const currentWeatherCard = document.querySelector(".current-weather-cards");
const forecastCards = document.querySelector(".weather-cards");
const useCurrentLocationBtn = document.querySelector(
  ".use-current-location-btn"
);
const recentCitiesDropdown = document.createElement("select");
recentCitiesDropdown.classList.add("recent-cities-dropdown");

// Append the dropdown to the search section
const searchSection = document.querySelector(".search-list");
searchSection.appendChild(recentCitiesDropdown);

// Initialize recent cities list (from local storage)
let recentCities = JSON.parse(localStorage.getItem("recentCities")) || [];
updateDropdown();

// Function to fetch weather data for a city
async function fetchWeatherData(city) {
  try {
    const response = await fetch(
      `https://api.openweathermap.org/data/2.5/weather?q=${city}&appid=${API_KEY}&units=metric`
    );
    const data = await response.json();

    if (response.ok) {
      displayCurrentWeather(data);
      fetchForecastData(data.coord.lat, data.coord.lon);
      updateRecentCities(city); // Update the recent cities list
    } else {
      alert(data.message);
    }
  } catch (error) {
    console.error("Error fetching weather data:", error);
    alert("Failed to fetch weather data. Please try again.");
  }
}

// Function to fetch forecast data using coordinates
async function fetchForecastData(lat, lon) {
  try {
    const response = await fetch(
      `https://api.openweathermap.org/data/2.5/forecast?lat=${lat}&lon=${lon}&appid=${API_KEY}&units=metric`
    );
    const data = await response.json();

    if (response.ok) {
      displayForecast(data);
    } else {
      alert(data.message);
    }
  } catch (error) {
    console.error("Error fetching forecast data:", error);
    alert("Failed to fetch forecast data. Please try again.");
  }
}

// Display current weather
function displayCurrentWeather(data) {
  const { name, main, wind, weather, dt } = data;
  const date = new Date(dt * 1000).toLocaleDateString();

  currentWeatherCard.innerHTML = `
        <div class="flex justify-between items-center">
            <div>
                <h2 class="text-2xl font-semibold">${name} (${date})</h2>
                <p class="text-lg">Temperature: ${main.temp}°C</p>
                <p>Wind: ${wind.speed} M/S</p>
                <p>Humidity: ${main.humidity}%</p>
            </div>
```

```
          <div>
              <img class='h-30' src="https://openweathermap.org/img/wn/${weather[0].icon}@4x.png"
alt="${weather[0].description}" />
          </div>
        </div>
    `;
}

// Display 5-day forecast
function displayForecast(data) {
  forecastCards.innerHTML = "";
  const dailyData = {};

  data.list.forEach((item) => {
    const date = item.dt_txt.split(" ")[0];
    if (!dailyData[date]) {
      dailyData[date] = [];
    }
    dailyData[date].push(item);
  });

  const days = Object.keys(dailyData).slice(0, 5);
  days.forEach((date) => {
    const dayData = dailyData[date][0];
    const { main, wind, weather } = dayData;

    forecastCards.innerHTML += `
          <div class="bg-gray-500 text-white rounded-lg shadow-sm text-center pt-6 pb-6">
              <p class="text-lg font-semibold">(${date})</p>
              <img class=" h-20  inline-block" src="https://openweathermap.org/img/wn/${weather[0].icon}@2x.png"
alt="${weather[0].description}" />
              <p>Temp: ${main.temp}°C</p>
              <p>Wind: ${wind.speed} M/S</p>
              <p>Humidity: ${main.humidity}%</p>
          </div> `;
  });
}

// Add a city to the recent cities list
```

```
function updateRecentCities(city) {
  if (!recentCities.includes(city)) {
    recentCities.unshift(city);
    if (recentCities.length > 5) {
      recentCities.pop(); // Limit to 5 cities
    }
    localStorage.setItem("recentCities", JSON.stringify(recentCities));
    updateDropdown();
  }
}
// Update the dropdown menu with recent cities
function updateDropdown() {
  recentCitiesDropdown.innerHTML =
    '<option value="" disabled selected>Recently Searched Cities</option>';
  recentCities.forEach((city) => {
    const option = document.createElement("option");
    option.value = city;
    option.textContent = city;
    recentCitiesDropdown.appendChild(option);
  });
}
// Event listener for dropdown selection
recentCitiesDropdown.addEventListener("change", (event) => {
  const city = event.target.value;
  if (city) {
    fetchWeatherData(city);
  }
});
// Event listener for the "Search" button
searchBtn.addEventListener("click", () => {
  const city = cityInput.value.trim();
  if (city) {
    fetchWeatherData(city);
    cityInput.value = "";
  } else {
    alert("Please enter a city name.");
  }
});
```

```
// Event listener for "Use Current Location" button
useCurrentLocationBtn.addEventListener("click", () => {
  if (navigator.geolocation) {
    navigator.geolocation.getCurrentPosition(
      (position) => {
        const { latitude, longitude } = position.coords;
        fetchForecastData(latitude, longitude);
        fetchWeatherDataByCoordinates(latitude, longitude);
      },
      () => {
        alert(
          "Unable to retrieve your location. Please enable location services."
        );
      }
    );
  } else {
    alert("Geolocation is not supported by this browser.");
  }
});
// Fetch weather data using coordinates
async function fetchWeatherDataByCoordinates(lat, lon) {
  try {
    const response = await fetch(
      `https://api.openweathermap.org/data/2.5/weather?lat=${lat}&lon=${lon}&appid=${API_KEY}&units=metric`
    );
    const data = await response.json();
    if (response.ok) {
      displayCurrentWeather(data);
    } else {
      alert(data.message);
    }
  } catch (error) {
    console.error("Error fetching weather data by coordinates:", error);
    alert("Failed to fetch weather data by coordinates. Please try again.");
  }
}
```

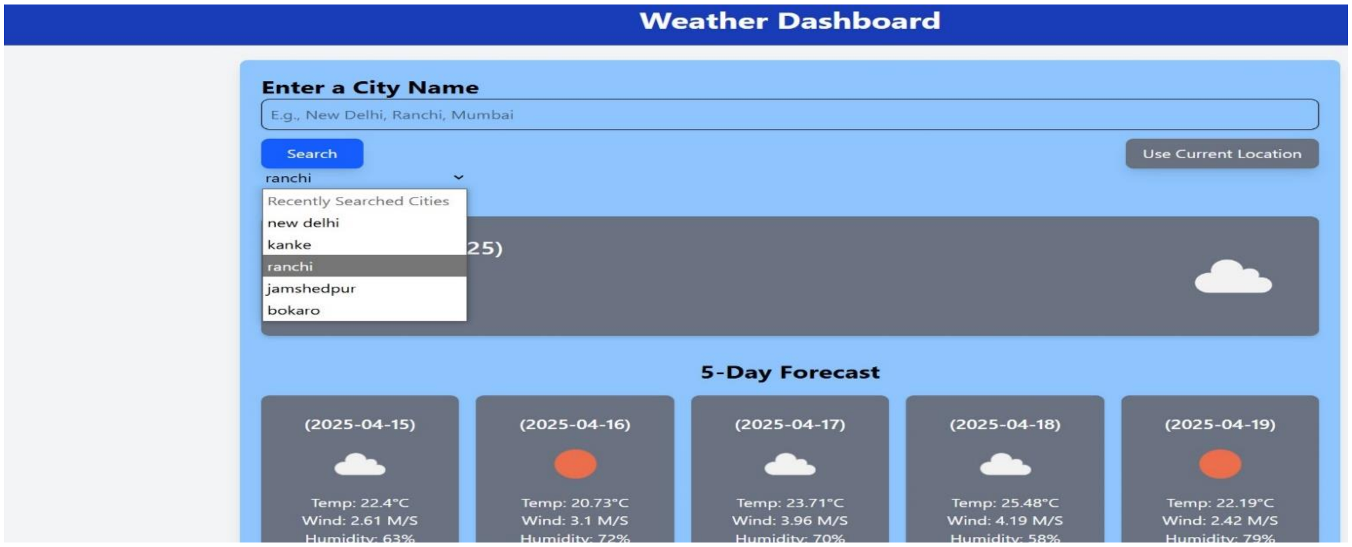
3. **Screenshot :-**



Ranchi (4/15/2025)
Temperature: 22.63°C
Wind: 1.8 M/S
Humidity: 61%

5-Day Forecast

(2025-04-15)	(2025-04-16)	(2025-04-17)	(2025-04-18)	(2025-04-19)
Temp: 22.4°C	Temp: 20.73°C	Temp: 23.71°C	Temp: 25.48°C	Temp: 22.19°C
Wind: 2.61 M/S	Wind: 3.1 M/S	Wind: 3.96 M/S	Wind: 4.19 M/S	Wind: 2.42 M/S
Humidity: 63%	Humidity: 72%	Humidity: 70%	Humidity: 58%	Humidity: 79%



5-Day Forecast

(2025-04-15)	(2025-04-16)	(2025-04-17)	(2025-04-18)	(2025-04-19)
Temp: 22.4°C	Temp: 20.73°C	Temp: 23.71°C	Temp: 25.48°C	Temp: 22.19°C
Wind: 2.61 M/S	Wind: 3.1 M/S	Wind: 3.96 M/S	Wind: 4.19 M/S	Wind: 2.42 M/S
Humidity: 63%	Humidity: 72%	Humidity: 70%	Humidity: 58%	Humidity: 79%

- **GitHub Link:- https://github.com/shashikant021/WeatherForecast**