EXPERIMENT NO. - 01

Roll No.	24
Name	Chaitanya Dinesh Dhayarkar
Class	D15B
Subject	Full Stack Development
Lab Outcome	**Build responsive and interactive UIs using Tailwind CSS**
Date of Performance/ Submission	28/07/2025 11/08/2025
Signature & Grades	

AIM: Build responsive and interactive UIs using Tailwind CSS

THEORY:

1.Introduction

Tailwind CSS is a **utility-first CSS framework** that allows developers to create custom, responsive designs directly in HTML using predefined classes. Instead of writing traditional CSS or using prebuilt UI components (like in Bootstrap), Tailwind gives you low-level utility classes like - **p-4, text-center, bg-blue-500**, etc., which you can combine to build any design.

2.Key Concepts:

i)Utility-First: You build designs using utility classes instead of writing custom CSS rules.

ii)Mobile-First: Tailwind uses mobile-first responsive breakpoints (sm, md, lg).

iii)Customizable: You can override the default design system by editing the configuration file. **iv)No CSS Naming Conflicts**: Avoids issues with BEM or naming classes.

v)Fast Prototyping: Quickly mock up ideas directly in your markup.

2. Tailwind CSS – Configuration Information

Basic Configuration Steps:

1) **Initialize the config file**:

```
npx tailwindcss init
```

2) **Edit tailwind.config.js to customize**:

```
 module.exports = {
content: [
" ./index.html",
"./src/**/*.{js,ts,jsx,tsx}",  // adjust depending on   your project
 ],
Theme: {
 extend: {
 colors: {
 primary: "#1E3A8A",
 secondary: "#9333EA",
 },
 },
 },
 plugins: [],
}
```

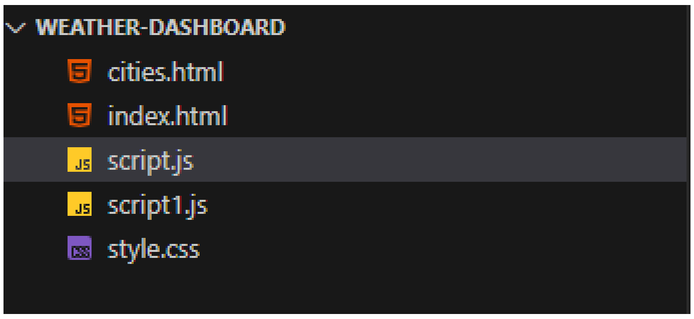
3) **PostCSS Configuration (Required in most setups):**

```
// postcss.config.js
module.exports = {
 plugins: {
tailwindcss: {},
autoprefixer: {},
},
}
```

4) **Main CSS file** (where Tailwind directives are imported):

```
 @tailwind base;
@tailwind components;
@tailwind utilities;
```

3. Project Directory Structure



```
WEATHER-DASHBOARD
    cities.html
    index.html
    script.js
    script1.js
    style.css
```

CODE:

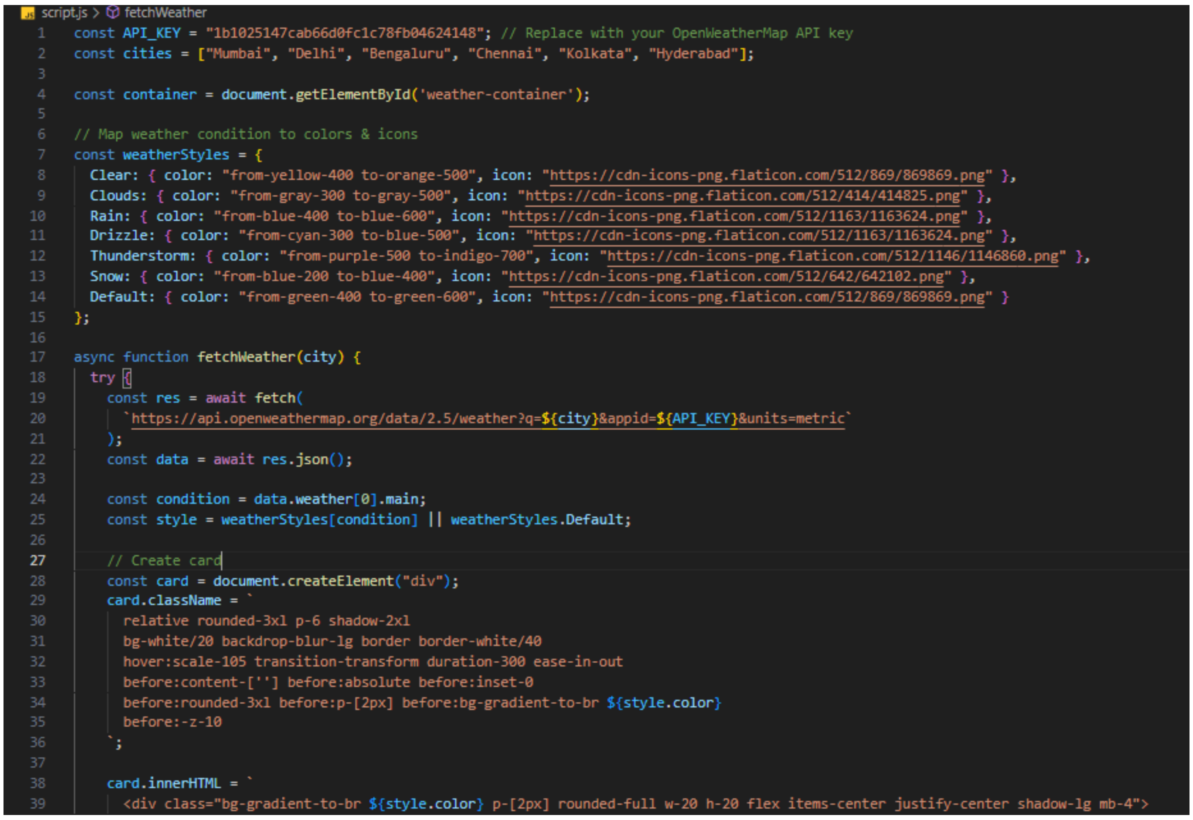
Index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Stylish Weather Dashboard</title>
  <script src="https://cdn.tailwindcss.com"></script>
</head>
<body class="bg-gradient-to-br from-blue-100 via-sky-200 to-blue-300 min-h-screen flex flex-col">

  <!-- Navbar -->
  <nav class="bg-gradient-to-r from-sky-500 to-blue-600 text-white shadow-lg">
    <div class="max-w-7xl mx-auto px-6 py-4 flex justify-between items-center">
      <!-- Logo & Title -->
      <div class="flex items-center gap-3">
        <img src="https://cdn-icons-png.flaticon.com/512/869/869869.png" class="w-8 h-8" alt="logo"
/>
        <span class="text-xl font-bold tracking-wide">Weather Dashboard</span>
      </div>

      <!-- Nav Links -->
      <div class="flex gap-6">
        <a href="index.html" class="relative group">
          Home
          <span class="absolute left-0 bottom-0 w-full h-[2px] bg-white"></span>
        </a>
        <a href="cities.html" class="relative group">
          Cities
          <span class="absolute left-0 bottom-0 w-0 h-[2px] bg-white transition-all group-hover:w-
full"></span>
        </a>
      </div>
    </div>
  </nav>
  <!-- Main Content -->
  <main class="flex flex-col items-center py-12 flex-grow">
    <h1 class="text-4xl font-extrabold text-sky-900 drop-shadow-lg mb-10">
      ⛅ Real-Time Weather Dashboard
    </h1>

    <!-- Weather Container -->
    <div id="weather-container" class="grid grid-cols-1 sm:grid-cols-2 lg:grid-cols-3 gap-10 px-6 w-full
max-w-6xl">
      <!-- Cards will be added here -->
    </div>
  </main>
  <script src="script.js">   </script></body></html>
```

Cities.html

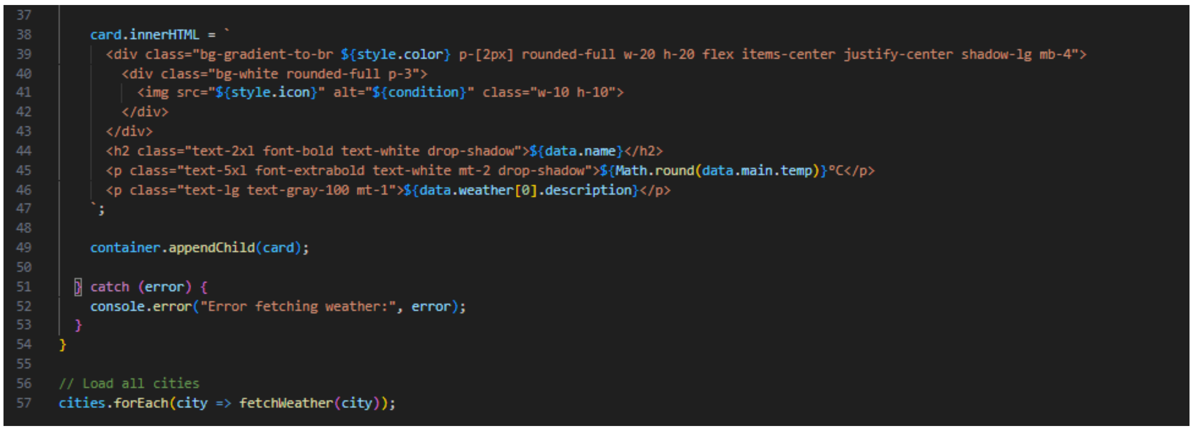
```
<!DOCTYPE html><html lang="en"><head>

  <meta charset="UTF-8" />

  <meta name="viewport" content="width=device-width, initial-scale=1.0" />

  <title>Search City</title>

  <script src="https://cdn.tailwindcss.com"></script>

</head>

<body class="bg-gradient-to-br from-blue-100 via-white to-blue-200 min-h-screen flex flex-col">

  <!-- Navbar -->

  <nav class="bg-blue-600 text-white p-4 shadow-lg flex justify-between items-center">

    <h1 class="text-2xl font-bold">⛅ Weather App</h1>

    <div class="space-x-6">

      <a href="index.html" class="hover:underline">Dashboard</a>

      <a href="cities.html" class="hover:underline">Search City</a>

    </div>

  </nav>

  <div class="flex flex-col items-center mt-10 px-4">

    <h2 class="text-2xl font-bold text-blue-800 mb-4">Search Weather by City</h2>

    <input id="city-input" type="text" placeholder="Enter city name..."

      class="px-4 py-2 border rounded-lg shadow-md focus:outline-none focus:ring-2 focus:ring-blue-
400 w-80 text-center">

    <button onclick="searchCity()"

      class="mt-4 px-6 py-2 bg-blue-600 text-white rounded-lg hover:bg-blue-700 shadow-lg">
      Search
    </button>
  </div>
  <div id="result" class="mt-8 flex justify-center"></div>
  <script src="script1.js"> </script></body></html>
```

Script.js



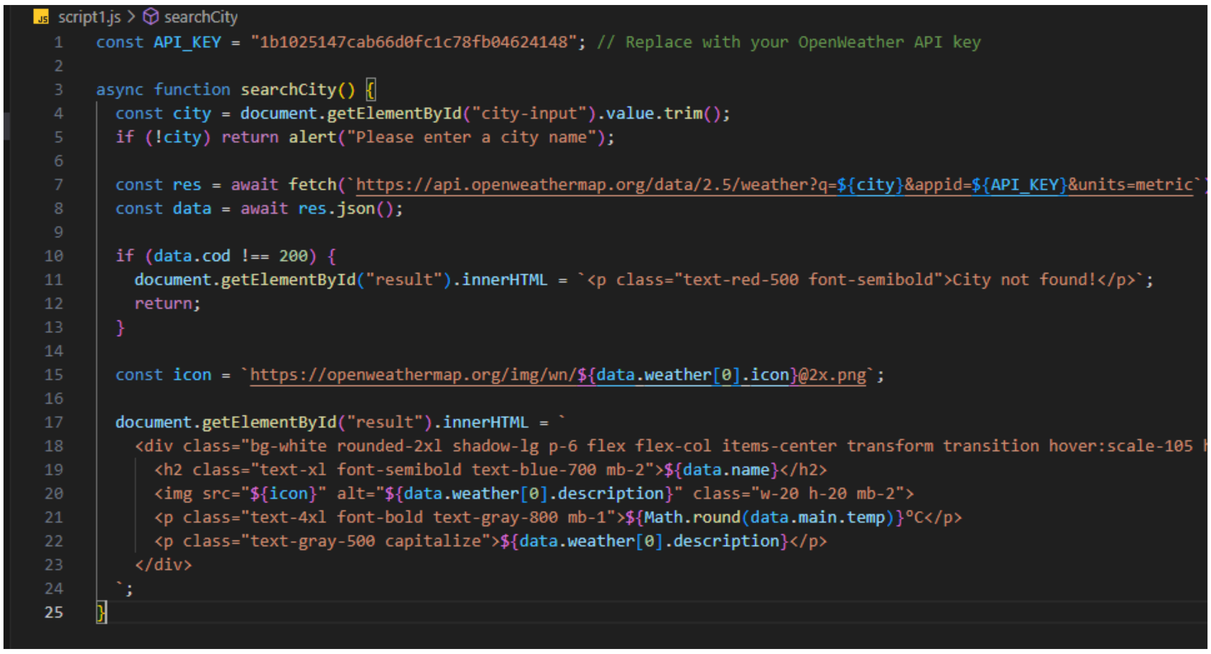
script.js > fetchWeather

```javascript
const API_KEY = "1b1025147cab66d0fc1c78fb04624148"; // Replace with your OpenWeatherMap API key
const cities = ["Mumbai", "Delhi", "Bengaluru", "Chennai", "Kolkata", "Hyderabad"];

const container = document.getElementById('weather-container');

// Map weather condition to colors & icons
const weatherStyles = {
  Clear: { color: "from-yellow-400 to-orange-500", icon: "https://cdn-icons-png.flaticon.com/512/869/869869.png" },
  Clouds: { color: "from-gray-300 to-gray-500", icon: "https://cdn-icons-png.flaticon.com/512/414/414825.png" },
  Rain: { color: "from-blue-400 to-blue-600", icon: "https://cdn-icons-png.flaticon.com/512/1163/1163624.png" },
  Drizzle: { color: "from-cyan-300 to-blue-500", icon: "https://cdn-icons-png.flaticon.com/512/1163/1163624.png" },
  Thunderstorm: { color: "from-purple-500 to-indigo-700", icon: "https://cdn-icons-png.flaticon.com/512/1146/1146860.png" },
  Snow: { color: "from-blue-200 to-blue-400", icon: "https://cdn-icons-png.flaticon.com/512/642/642102.png" },
  Default: { color: "from-green-400 to-green-600", icon: "https://cdn-icons-png.flaticon.com/512/869/869869.png" }
};

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

    const condition = data.weather[0].main;
    const style = weatherStyles[condition] || weatherStyles.Default;

    // Create card
    const card = document.createElement("div");
    card.className = `
      relative rounded-3xl p-6 shadow-2xl
      bg-white/20 backdrop-blur-lg border border-white/40
      hover:scale-105 transition-transform duration-300 ease-in-out
      before:content-[''] before:absolute before:inset-0
      before:rounded-3xl before:p-[2px] before:bg-gradient-to-br ${style.color}
      before:-z-10
    `;

    card.innerHTML = `
      <div class="bg-gradient-to-br ${style.color} p-[2px] rounded-full w-20 h-20 flex items-center justify-center shadow-lg mb-4">
```



```javascript

    card.innerHTML = `
      <div class="bg-gradient-to-br ${style.color} p-[2px] rounded-full w-20 h-20 flex items-center justify-center shadow-lg mb-4">
        <div class="bg-white rounded-full p-3">
          <img src="${style.icon}" alt="${condition}" class="w-10 h-10">
        </div>
      </div>
      <h2 class="text-2xl font-bold text-white drop-shadow">${data.name}</h2>
      <p class="text-5xl font-extrabold text-white mt-2 drop-shadow">${Math.round(data.main.temp)}°C</p>
      <p class="text-lg text-gray-100 mt-1">${data.weather[0].description}</p>
    `;

    container.appendChild(card);

  } catch (error) {
    console.error("Error fetching weather:", error);
  }
}

// Load all cities
cities.forEach(city => fetchWeather(city));
```

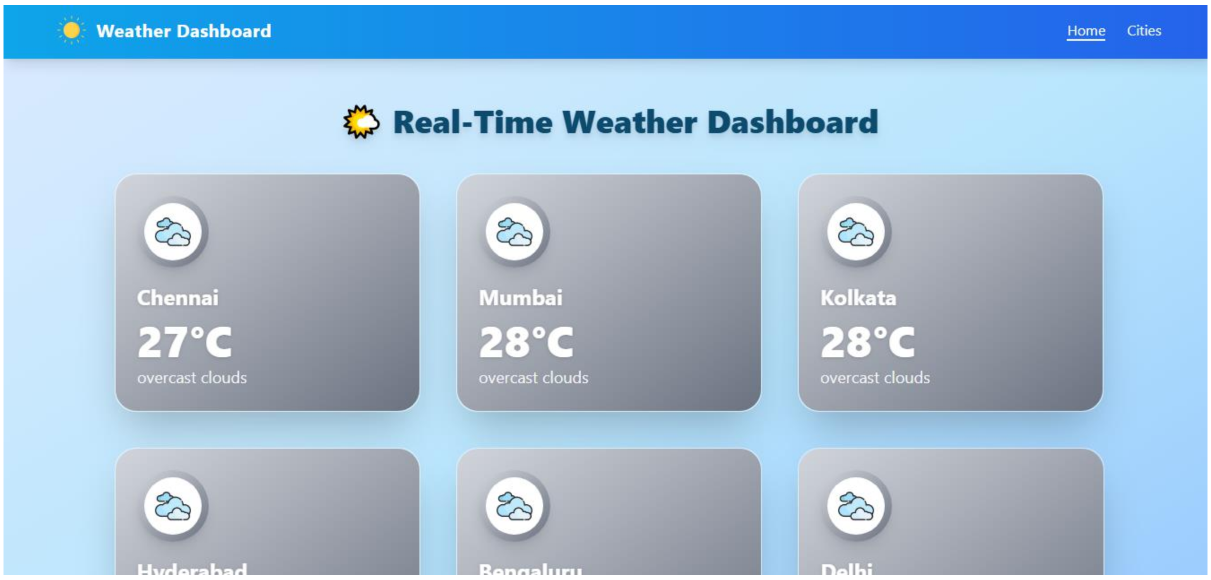
Script1.js

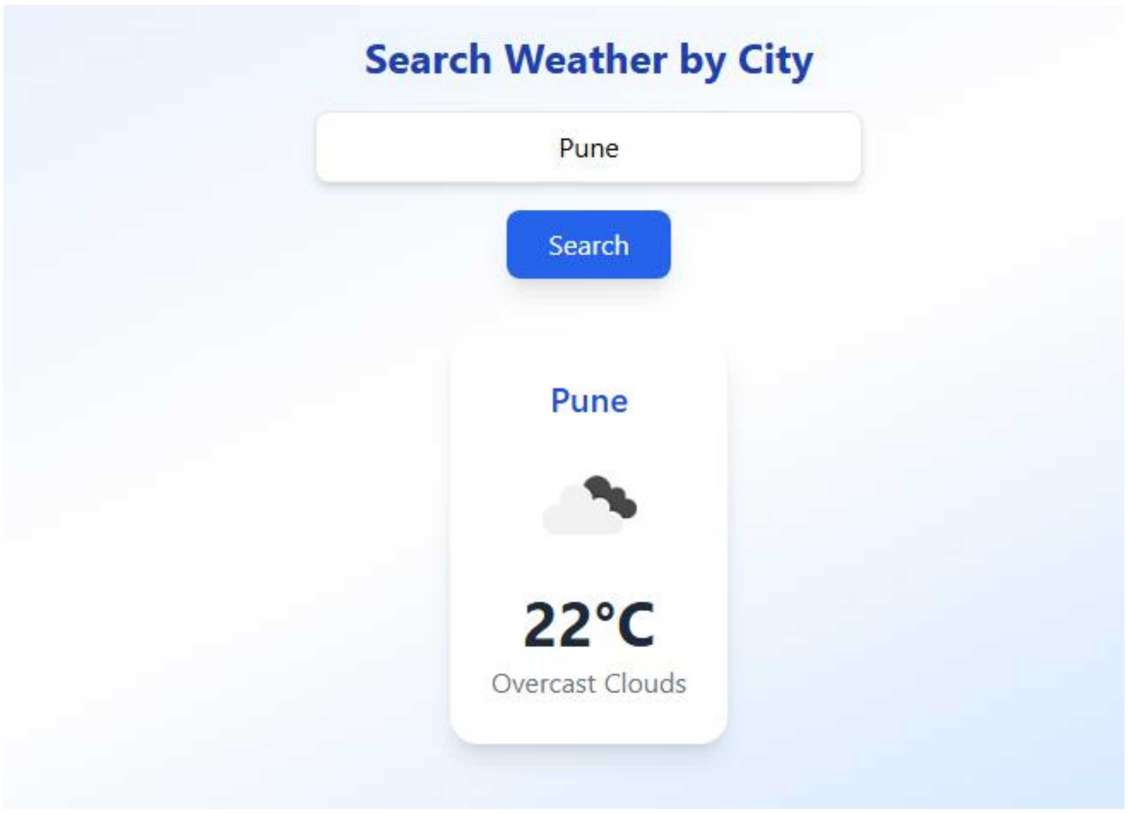


script1.js > searchCity

```javascript
const API_KEY = "1b1025147cab66d0fc1c78fb04624148"; // Replace with your OpenWeather API key

async function searchCity() {
  const city = document.getElementById("city-input").value.trim();
  if (!city) return alert("Please enter a city name");

  const res = await fetch(`https://api.openweathermap.org/data/2.5/weather?q=${city}&appid=${API_KEY}&units=metric`
  const data = await res.json();

  if (data.cod !== 200) {
    document.getElementById("result").innerHTML = `<p class="text-red-500 font-semibold">City not found!</p>`;
    return;
  }

  const icon = `https://openweathermap.org/img/wn/${data.weather[0].icon}@2x.png`;

  document.getElementById("result").innerHTML = `
    <div class="bg-white rounded-2xl shadow-lg p-6 flex flex-col items-center transform transition hover:scale-105
      <h2 class="text-xl font-semibold text-blue-700 mb-2">${data.name}</h2>
      <img src="${icon}" alt="${data.weather[0].description}" class="w-20 h-20 mb-2">
      <p class="text-4xl font-bold text-gray-800 mb-1">${Math.round(data.main.temp)}°C</p>
      <p class="text-gray-500 capitalize">${data.weather[0].description}</p>
    </div>
  `;
}
```

Output:





Conclusion:

In this experiment, we successfully built a responsive and interactive real-time weather dashboard using Tailwind CSS and the OpenWeatherMap API. Tailwind CSS's utility-first approach allowed us to rapidly design a clean, modern, and mobile-friendly user interface without writing custom CSS. The integration of OpenWeatherMap enabled live weather data retrieval, which was dynamically displayed in visually appealing gradient cards. This project demonstrated how combining Tailwind CSS with an external API can produce a professional, data-driven web application that is both functional and aesthetically engaging.