# Bansilal Ramnath Agarwal Charitable Trust’s

Vishwakarma Institute of Technology,Pune-37

*(An Autonomous Institute of Savitribai Phule Pune University)*

 **Department of Artificial Intelligence and Data Science**

|  |  |
| --- | --- |
| **Division** | A |
| **Batch** | 1 |
| **Rollno** | 26 |
| **Name** | Jineshwari Bagul |

**Create a Weather Forecasting App, in React JS. Configure weather api, which contains country wise information. UI contains a search-bar, where use enter the name of the country, results should display today's date, time and timestamped weather. Also it should show forecast of next 5 days.**

**Code:**

WeatherApp.jsx :

// export default WeatherApp;

import React, { useState } from "react";

import axios from "axios";

import { format } from "date-fns";

import "./WeatherApp.css"; // Import CSS

const WeatherApp = () => {

  const [country, setCountry] = useState("");

  const [weatherData, setWeatherData] = useState(null);

  const [forecast, setForecast] = useState([]);

  const [loading, setLoading] = useState(false);

  const API\_KEY = "f4b870cdf2e9c1b663abfcae6f60b23a";

  const fetchWeather = async () => {

    if (!country) return;

    setLoading(true);

    try {

      const response = await axios.get(

        `https://api.openweathermap.org/data/2.5/weather?q=${country}&units=metric&appid=${API\_KEY}`

      );

      const { lon, lat } = response.data.coord;

      const forecastRes = await axios.get(

        `https://api.openweathermap.org/data/2.5/forecast?lat=${lat}&lon=${lon}&units=metric&appid=${API\_KEY}`

      );

      setWeatherData(response.data);

      setForecast(forecastRes.data.list.slice(0, 5));

    } catch (error) {

      alert("Invalid country name or API limit reached. Try again!");

    }

    setLoading(false);

  };

  // Combine time and weather emojis dynamically

  const getWeatherEmoji = () => {

    if (!weatherData) return "🌍";

    const weatherDesc = weatherData.weather[0].description.toLowerCase();

    const temp = weatherData.main.temp;

    const timezoneOffset = weatherData.timezone;

    const currentTime = new Date(new Date().getTime() + timezoneOffset \* 1000);

    const currentHour = currentTime.getUTCHours();

    let emoji = "";

    // 🌙 Time-based emojis

    if (currentHour >= 5 && currentHour < 12) {

      emoji = "🌞"; // Morning

    } else if (currentHour >= 12 && currentHour < 18) {

      emoji = "🌇"; // Afternoon

    } else {

      emoji = "🌙"; // Night

    }

    // 🌡️ Temperature-based emojis

    if (temp >= 30) {

      emoji += "🔥"; // Hot weather

    } else if (temp < 10) {

      emoji += "❄️"; // Cold weather

    } else {

      emoji += "🌤️"; // Mild weather

    }

    // ☁️ Weather condition emojis

    if (weatherDesc.includes("rain")) emoji = "🌧️💨";

    if (weatherDesc.includes("cloud")) emoji = "☁️";

    if (weatherDesc.includes("clear")) emoji = emoji; // Keep time + temp combo

    return emoji;

  };

  const getBackgroundClass = () => {

    if (!weatherData) return "cloudy";

    const timezoneOffset = weatherData.timezone;

    const currentTime = new Date(new Date().getTime() + timezoneOffset \* 1000);

    const currentHour = currentTime.getUTCHours();

    if (currentHour >= 5 && currentHour < 12) return "morning";

    if (currentHour >= 12 && currentHour < 18) return "afternoon";

    return "night";

  };

  return (

    <div className={`app-container ${getBackgroundClass()}`}>

      <div className="content">

        <h1 className="app-title">🌦️ Weather App</h1>

        {/\* Search Bar \*/}

        <div className="search-bar">

          <input

            type="text"

            placeholder="Enter Country"

            value={country}

            onChange={(e) => setCountry(e.target.value)}

          />

          <button onClick={fetchWeather}>Get Weather</button>

        </div>

        {loading && <p>Loading...</p>}

        {/\* Current Weather Display \*/}

        {weatherData && (

          <div className="weather-card">

            <div className="weather-emoji">{getWeatherEmoji()}</div>

            <h2>{weatherData.name}, {weatherData.sys.country}</h2>

            <div className="weather-info">

              <p>🌡️ {weatherData.main.temp}°C</p>

              <p>☁️ {weatherData.weather[0].description}</p>

              <p>💨 {weatherData.wind.speed} m/s</p>

              <p>🕒 {format(new Date(), "EEEE, MMMM d, yyyy")}</p>

            </div>

          </div>

        )}

        {/\* 5-Day Forecast Grid \*/}

        {forecast.length > 0 && (

          <div>

            <h3 className="forecast-title">Next 5 Days Forecast:</h3>

            <div className="forecast-container">

              {forecast.map((item, index) => (

                <div key={index} className="forecast-card">

                  <p>📅 {format(new Date(item.dt \* 1000), "EEEE")}</p>

                  <p>🌡️ {item.main.temp}°C</p>

                  <p>💨 {item.wind.speed} m/s</p>

                  <p>⏱️ {format(new Date(item.dt \* 1000), "p")}</p>

                </div>

              ))}

            </div>

          </div>

        )}

      </div>

    </div>

  );

};

export default WeatherApp;

main.jsx:

import { StrictMode } from 'react'

import { createRoot } from 'react-dom/client'

import './index.css'

import App from './App.jsx'

createRoot(document.getElementById('root')).render(

  <StrictMode>

    <App />

  </StrictMode>,

)

WeatherApp.css:  
/\* 🟢 Full-screen layout with center alignment \*/

.app-container {

    min-height: 100vh;

    width: 100vw;

    display: flex;

    justify-content: center;

    align-items: center;

    transition: background 0.5s ease;

    box-sizing: border-box;

    padding: 2rem;

    margin: 0;

}

/\* Dynamic background colors \*/

.morning {

    background: linear-gradient(to right, #ffde7d, #ffcd59);

}

.afternoon {

    background: linear-gradient(to right, #ff9a8b, #ff6a88);

}

.night {

    background: linear-gradient(to right, #1e3c72, #2a5298);

}

.cloudy {

    background: linear-gradient(to right, #a1c4fd, #c2e9fb);

}

/\* Main content container \*/

.content {

    display: flex;

    flex-direction: column;

    justify-content: center;

    align-items: center;

    max-width: 1100px;

    width: 90%;

    padding: 3rem;

    background: white;

    box-shadow: 0 20px 40px rgba(0, 0, 0, 0.1);

    border-radius: 16px;

    text-align: center;

    animation: fadeIn 1s ease;

}

/\* App title \*/

.app-title {

    font-size: 2.5rem;

    font-weight: bold;

    color: #333;

    margin-bottom: 2rem;

}

/\* Search bar styling \*/

.search-bar {

    display: flex;

    justify-content: center;

    align-items: center;

    gap: 1rem;

    width: 100%;

    max-width: 800px;

    margin-bottom: 2rem;

}

.search-bar input {

    flex: 1;

    padding: 1rem;

    border-radius: 10px;

    border: 1px solid #ccc;

    outline: none;

    font-size: 1rem;

    color: #333;

    box-shadow: 0 4px 10px rgba(0, 0, 0, 0.1);

}

.search-bar button {

    padding: 1rem 2rem;

    border: none;

    background-color: #007bff;

    color: white;

    font-size: 1rem;

    cursor: pointer;

    border-radius: 10px;

    transition: all 0.3s;

}

.search-bar button:hover {

    background-color: #0056b3;

    transform: scale(1.05);

}

/\* Weather emoji \*/

.weather-emoji {

    font-size: 6rem;  /\* Smaller emoji for better scaling \*/

    margin-bottom: 1rem;

    animation: popIn 0.5s ease-in;

}

/\* Weather information \*/

.weather-card {

    display: flex;

    flex-direction: column;

    align-items: center;

    max-width: 800px;

    width: 100%;

    padding: 3rem;

    border-radius: 12px;

    box-shadow: 0 15px 30px rgba(0, 0, 0, 0.1);

    background: white;

    color: #333;

    transition: transform 0.3s;

}

.weather-card:hover {

    transform: scale(1.02);

}

.weather-info p {

    font-size: 1.4rem;

    margin: 0.5rem 0;

}

/\* Forecast section \*/

.forecast-container {

    display: grid;

    grid-template-columns: repeat(5, 1fr); /\* 🌟 2 columns layout \*/

    gap: 1.5rem;

    width: 100%;

    margin-top: 3rem;

}

.forecast-card {

    background: #f0f0f0;

    color: #333;

    padding: 1.5rem;

    border-radius: 12px;

    box-shadow: 0 8px 20px rgba(0, 0, 0, 0.1);

    transition: transform 0.3s;

    text-align: center;

}

.forecast-card:hover {

    transform: scale(1.05);

}

/\* 🌟 Responsive Design \*/

@media (max-width: 768px) {

    .forecast-container {

        grid-template-columns: 1fr; /\* 1 column for small screens \*/

    }

}

/\* Animations \*/

@keyframes fadeIn {

    from {

        opacity: 0;

    }

    to {

        opacity: 1;

    }

}

@keyframes popIn {

    0% {

        transform: scale(0);

    }

    100% {

        transform: scale(1);

    }

}

/\* 🌟 Responsive Design \*/

@media (max-width: 1024px) {

    .content {

        padding: 2rem;

    }

    .search-bar {

        flex-direction: column;

        gap: 1rem;

    }

    .weather-card {

        padding: 2rem;

    }

    .weather-emoji {

        font-size: 5rem;

    }

    .forecast-container {

        grid-template-columns: repeat(2, 1fr); /\* 2 columns for medium screens \*/

    }

}

@media (max-width: 768px) {

    .content {

        padding: 1.5rem;

    }

    .weather-emoji {

        font-size: 4rem;

    }

    .forecast-container {

        grid-template-columns: 1fr; /\* Single column for small screens \*/

    }

}

@media (max-width: 480px) {

    .app-title {

        font-size: 2rem;

    }

    .weather-card {

        padding: 1.5rem;

    }

    .weather-emoji {

        font-size: 3rem;

    }

}

App.jsx:

import React from "react";

import WeatherApp from "./WeatherApp";

const App = () => {

  return (

    <div>

      <WeatherApp />

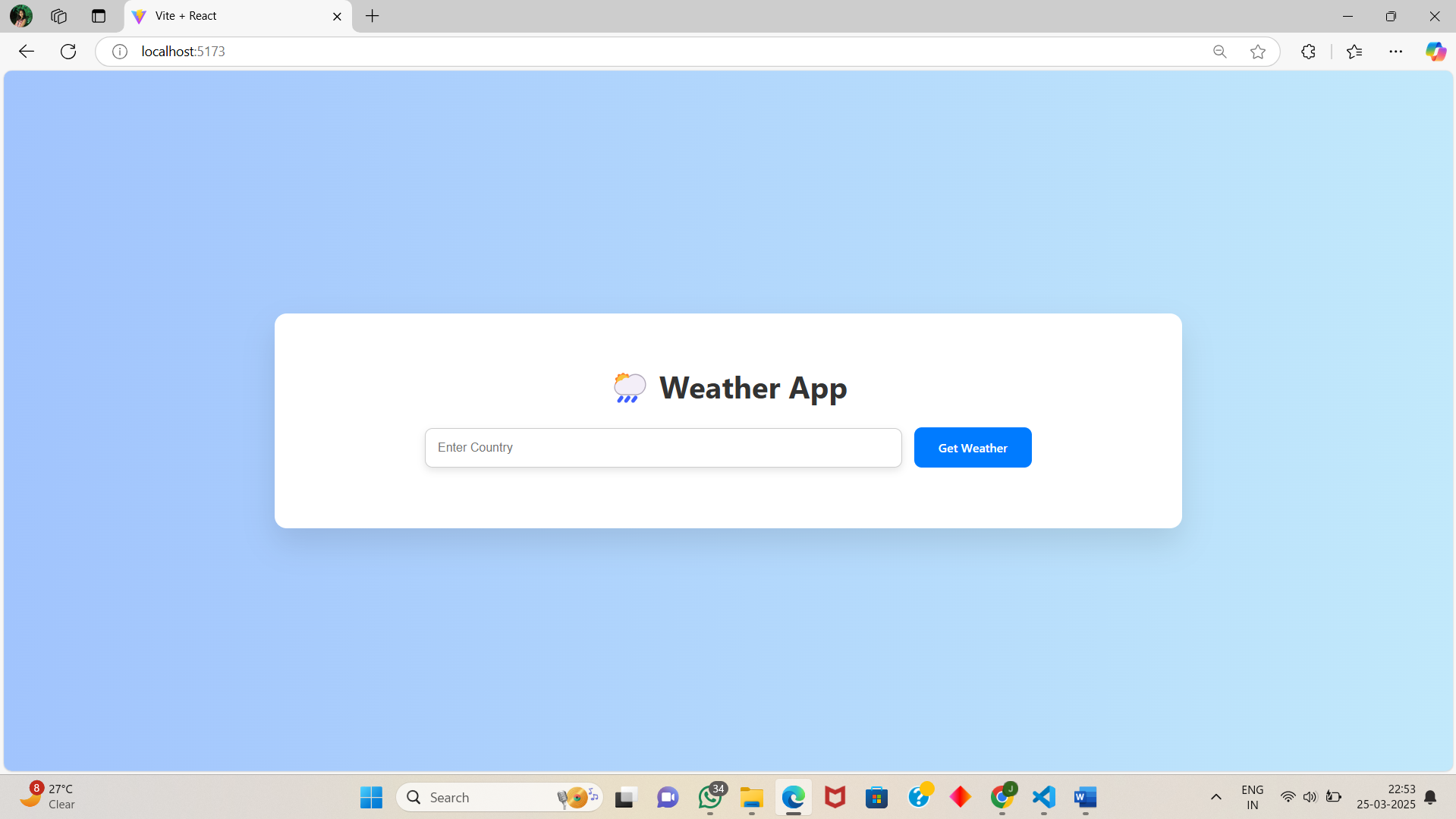
    </div>

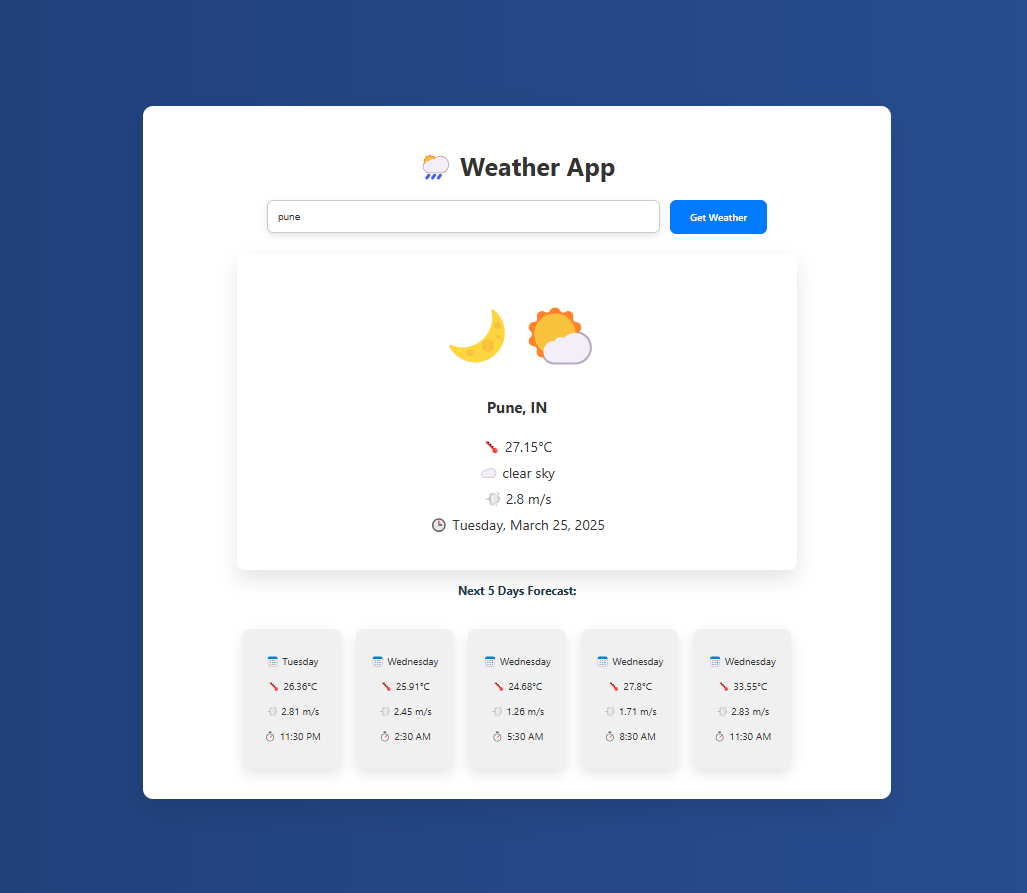
  );

};

export default App;

**OUTPUT :**

****

****