META FRONT-END DEVELOPER

(Professional Certification)

REACT BASED PROJECT REPORT

OF

WEATHERWISE – WEATHER APPLICATION

Payal SamarKumar Sahu
(Submitted by)

Table Of Content

INTRODUCTION

Problem Statement

Weather information is crucial for daily planning, whether it's for commuting, scheduling events, or simply dressing appropriately. However, many weather applications offer cluttered interfaces, slow performance, or unreliable data, which can lead to user frustration. There is a need for a streamlined, user-friendly weather application that provides accurate and timely weather updates in a visually appealing and intuitive manner.

Objective

The primary objective of WeatherWise is to deliver a seamless user experience by providing real-time weather data and forecasts through a clean and responsive interface. The application aims to be accessible to users of all technical proficiencies, ensuring that weather information is readily available and easy to understand.

Project Overview

WeatherWise is a web-based weather application built using HTML, CSS, and React. It leverages the OpenWeatherMap API to fetch and display current weather conditions and forecasts for any searched location. Key features include a search functionality to find weather by city, automatic geolocation to fetch weather based on the user's current location, and a responsive design to ensure usability across various devices. The project emphasizes simplicity, speed, and reliability, aiming to become the go-to weather app for users seeking straightforward weather information.

TECHNOLOGIES USED

Frontend

1) **HTML**: Used to create the structure of the WeatherWise application. HTML elements form the backbone of the application, providing a skeleton that defines the layout and organization of content.
2) **CSS**: Applied for styling and layout, CSS ensures that the application is visually appealing and responsive. It includes custom styles as well as CSS frameworks to create a modern and user-friendly interface.
3) **React**: The core technology for building the user interface. React's component-based architecture allows for modular, reusable components, enhancing the maintainability and scalability of the application.

APIs

OpenWeatherMap API: Integrated to fetch real-time weather data. This API provides comprehensive weather information, including current conditions, forecasts, and more, enabling WeatherWise to offer accurate and up-to-date weather updates.

Deployment

Netlify: Chosen for hosting the WeatherWise application. Netlify provides a reliable and efficient platform for deploying web applications, with features like continuous deployment, custom domain support, and SSL certificates, ensuring that WeatherWise is always accessible and secure.

FEATURES

Current Weather

1) **Real-Time Updates**: Displays the current weather conditions for any searched location, including temperature, humidity, wind speed, and weather description.
2) **Weather Icons**: Utilizes visually intuitive icons to represent different weather conditions, enhancing the user experience.

Forecast

1) **5-Day Forecast**: Provides a detailed weather forecast for the next five days, allowing users to plan ahead.
2) **Daily Breakdown**: Includes information on daily highs and lows, weather conditions, and precipitation probabilities.

Search Functionality

1) **City Search**: Users can search for weather information by entering the name of any city worldwide. The application retrieves and displays the relevant weather data quickly and accurately.
2) **Autocomplete**: Implements an autocomplete feature to suggest city names as users type, making the search process faster and more convenient.

Geolocation

1) **Automatic Location Detection**: Automatically fetches and displays weather information based on the user's current geographical location using the browser's geolocation feature.
2) **Manual Override**: Allows users to manually enter a location if they prefer not to use geolocation or if geolocation data is inaccurate.

Responsive Design

1) **Cross-Device Compatibility**: Ensures that the application is fully responsive and functions seamlessly across various devices, including desktops, tablets, and smartphones.
2) **Adaptive Layouts**: Uses CSS media queries to adjust the layout and design based on the screen size, ensuring optimal user experience on all devices.

Error Handling

1) **User-Friendly Messages**: Displays clear and concise error messages for issues such as invalid city names, failed API requests, or network problems.
2) **Fallback Options**: Provides fallback weather information or prompts users to try again later in case of persistent errors.

CONCLUSION

Project Summary

WeatherWise successfully achieves its objective of providing a seamless and user-friendly platform for accessing real-time weather information. The application leverages modern web technologies, including HTML, CSS, and React, to deliver a clean and responsive interface. By integrating the OpenWeatherMap API, WeatherWise offers accurate current weather conditions and forecasts, catering to users' daily planning needs.

Acknowledgments

Special thanks to OpenWeatherMap for providing the weather data API, which is central to the functionality of WeatherWise. Gratitude is also extended to Netlify for offering a reliable hosting platform that ensures the application is always accessible to users.

WeatherWise stands as a testament to the power of modern web development tools and practices, delivering a reliable, efficient, and visually appealing weather application. Future improvements and additional features will continue to enhance its value, making WeatherWise an indispensable tool for users seeking accurate weather information.

APPENDIX

Code Snippets (API Call)

api.js

```
const api_key = "0e5f53991ec3117f01e97a07b672cc32";

export const fetchData = function (URL, callback) {
  fetch(`${URL}&appid=${api_key}`)
    .then((res) => res.json())
    .then((data) => callback(data));
};

export const url = {
  currentWeather(lat, lon) {
    return `https://api.openweathermap.org/data/2.5/weather?${lat}&${lon}&units=metric`;
  },
  forecast(lat, lon) {
    return `https://api.openweathermap.org/data/2.5/forecast?${lat}&${lon}&units=metric`;
  },
  airPollution(lat, lon) {
    return `https://api.openweathermap.org/data/2.5/air_pollution?${lat}&${lon}`;
  },
  reverseGeo(lat, lon) {
    return `https://api.openweathermap.org/geo/1.0/reverse?${lat}&${lon}&limit=5`;
  },

  geo(query) {
    return `https://api.openweathermap.org/geo/1.0/direct?q=${query}&limit=5`;
  }
};
```

Website ScreenShot