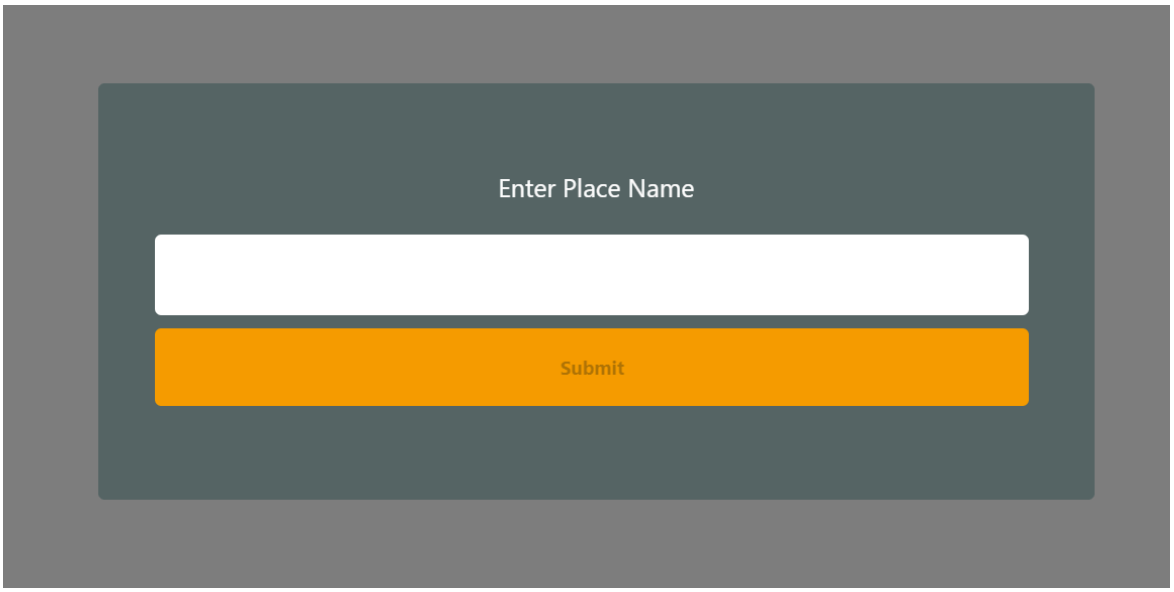
Weather Application (GitHub Repo)

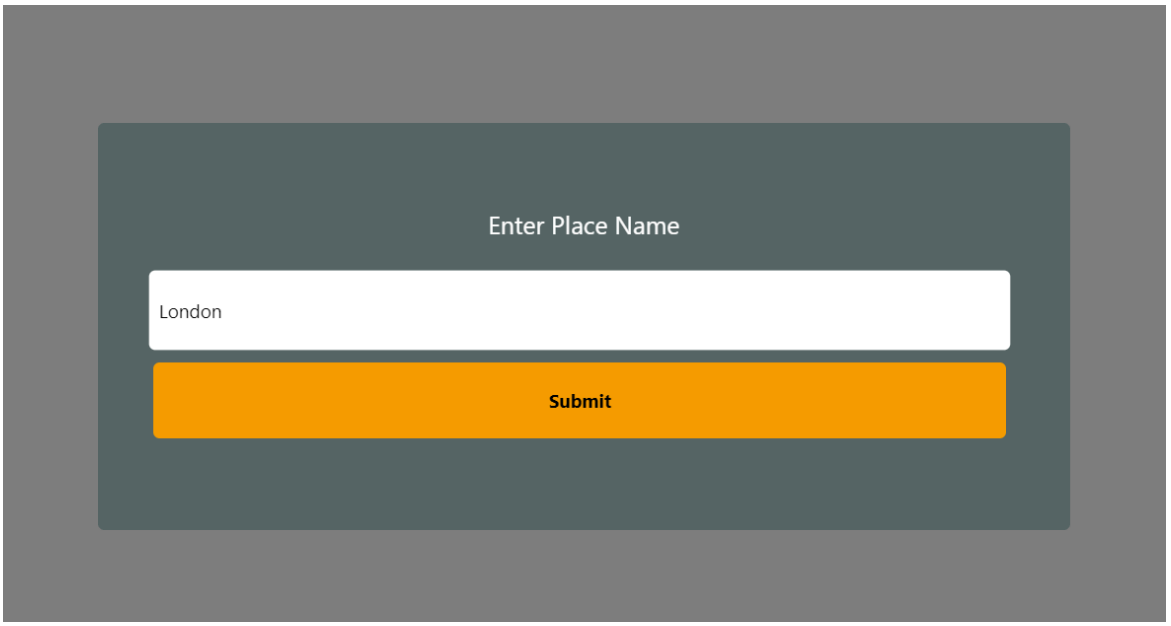
Project Overview

This project is an **interactive web application** designed to allow users to **enter a location** and **retrieve the current weather conditions** for that location. The app displays details such as **temperature, precipitation, wind speed, weather description, an icon, and more**. The application uses OpenWeatherMap's free API calls for the weather data (specifically the geocoding and weather APIs).

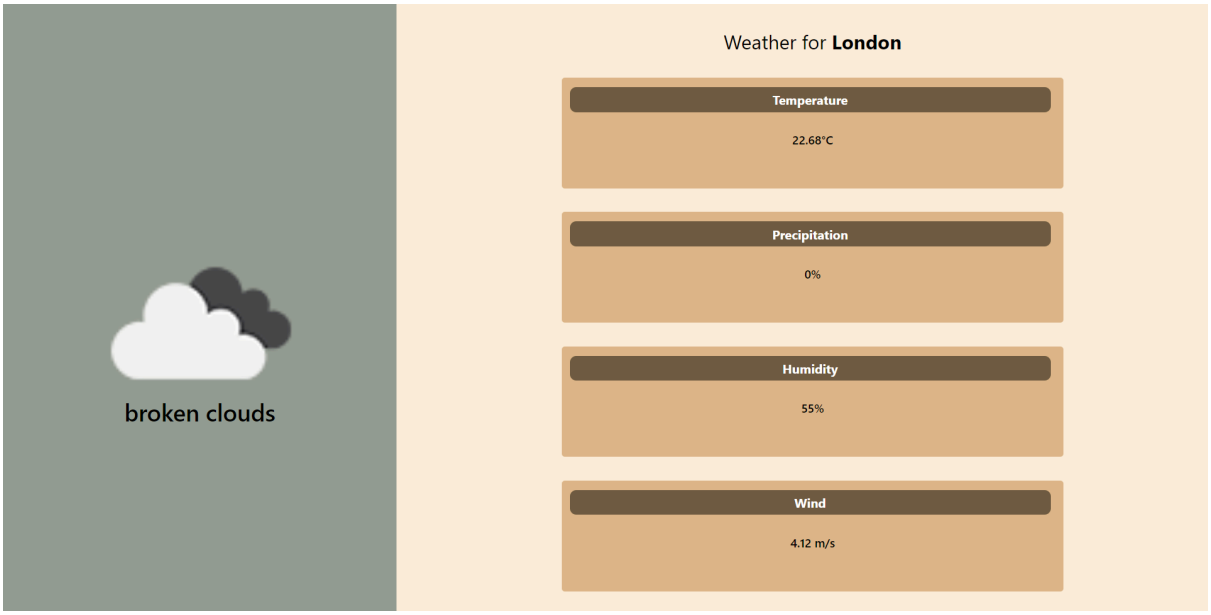
Key Features

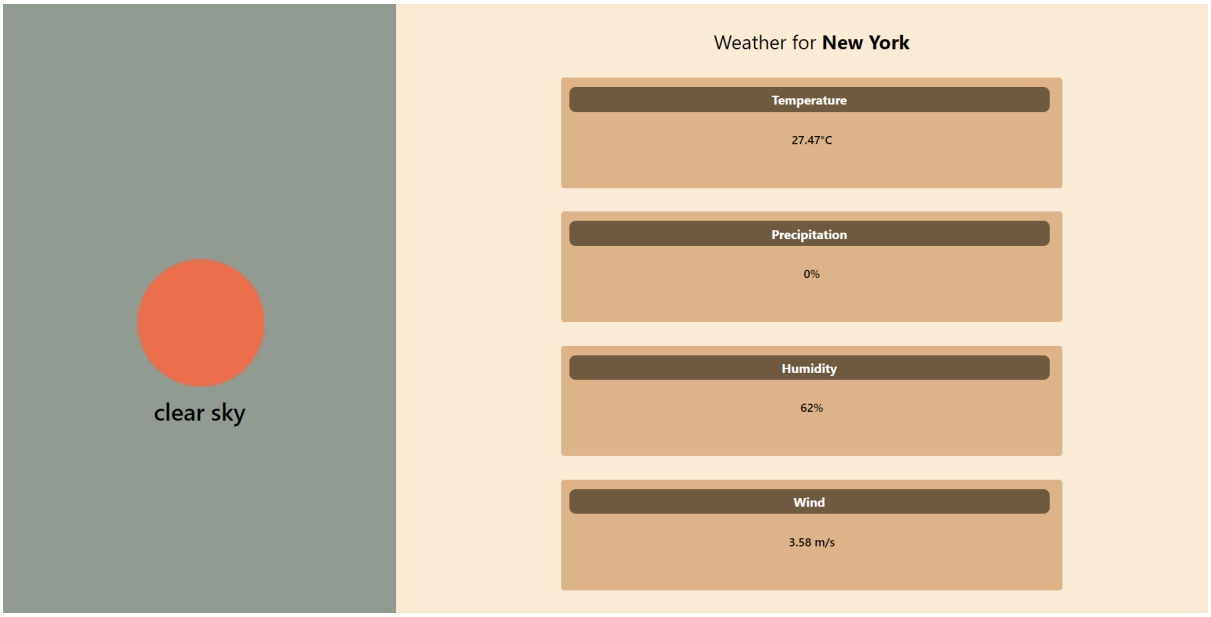
- **Location Input**: Users can enter the name of a city or location in a text input field for which they want to check the weather.



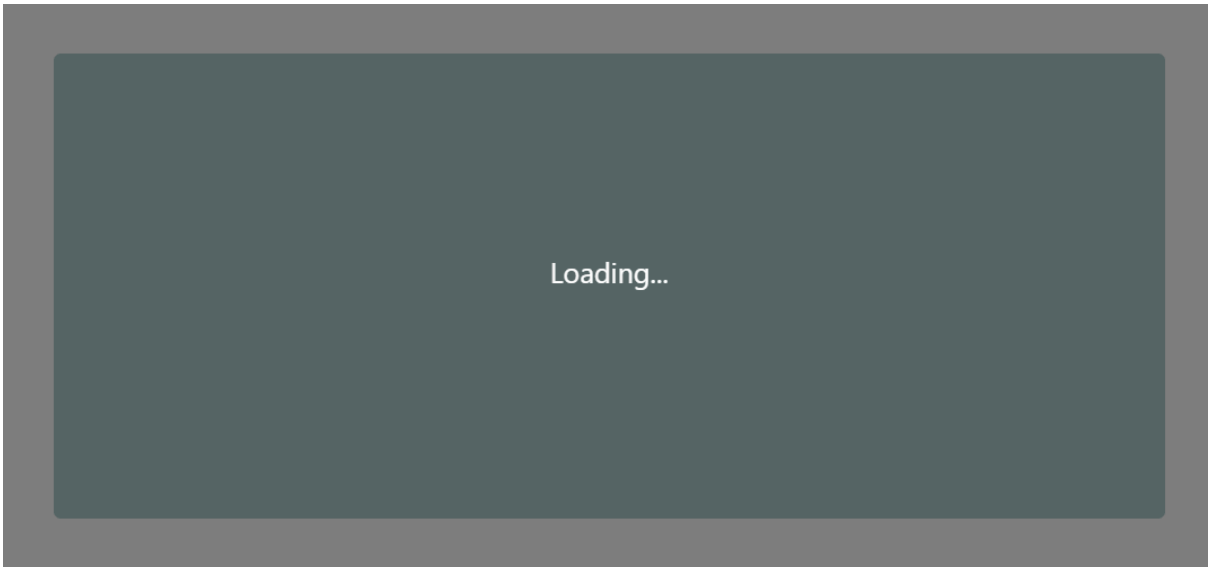


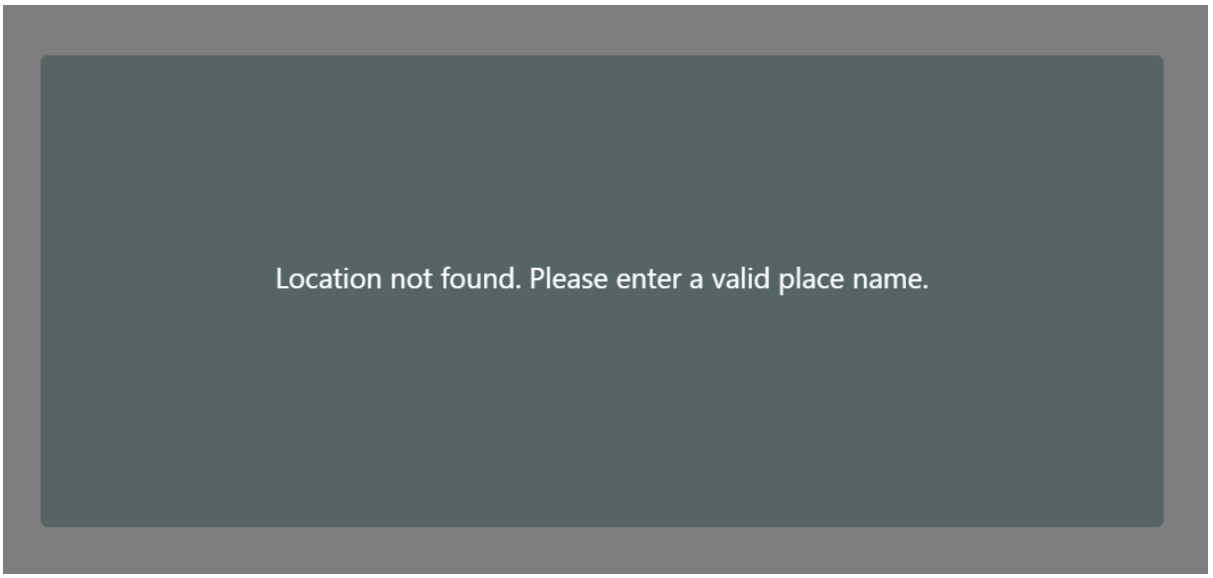
- **Display Weather Data**: After the user submits a location, the current weather conditions for that location is displayed. CSS is used to make the app visually appealing.

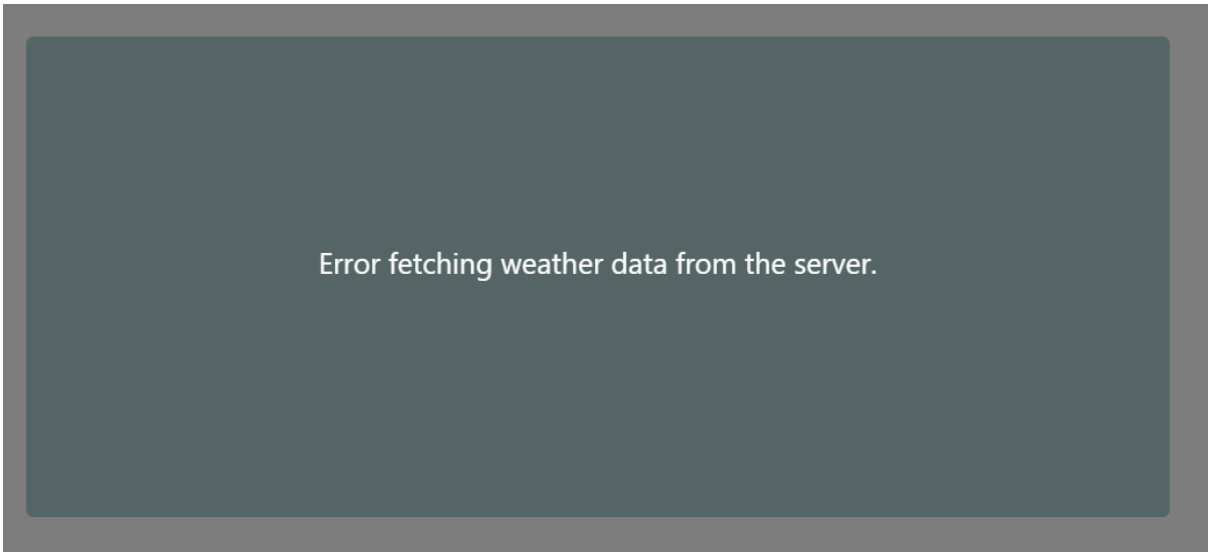




- **Error Handling**: Error handling has been implemented (in case the user has entered an invalid location or if there are issues with the API). The user is alerted of any specific errors that may arise.

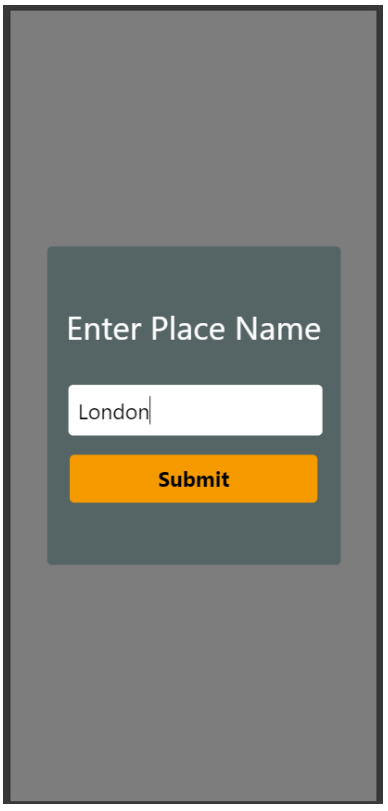


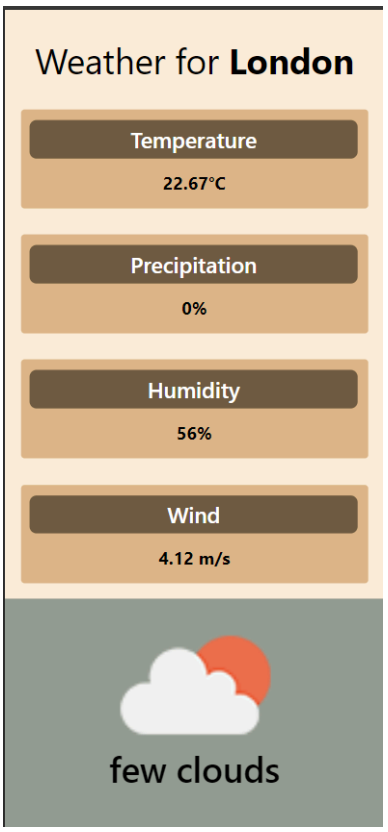




- **Responsive Design**: The user interface is optimized for both desktop and mobile devices.


Enter Place Name
Submit


Enter Place Name
London
Submit



Technologies Used

- **HTML5/JSX**: For structuring of the web application.

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <link rel="icon" type="image/png" href="/favicon.png" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <title>Weather App</title>
  </head>
  <body>
    <div id="root"></div>
    <script type="module" src="/src/main.tsx"></script>
  </body>
</html>
```

HTML

```jsx
// Form for place name submission.
<form className="input-container" onSubmit={handlePlaceNameSubmit}>
  <div className="input-fields-container">
    <div className="input-container-head">Enter Place Name</div>
    <input
      type="text"
      maxLength={100}
      className="input-field text"
      value={tempPlaceName}
      onChange={(e) => setTempPlaceName(e.target.value)}
    />
    <input
      type="submit"
      className={`input-field ${
        submitDisabled ? "disabled" : "submit"
      }`}
      value="Submit"
      disabled={submitDisabled} // User cannot submit until they have entered the place name.
    />
  </div>
</form>
```

JSX

- **CSS3**: For styling and responsiveness.

```css
.input-container {
    z-index: 100;
    position: fixed;
    top: 0;
    left: 0;
    height: 100vh;
    width: 100vw;
    align-content: center;
    background-color: rgba(0, 0, 0, 0.5);
}

.input-fields-container {
    height: 40%;
    width: 50vw;
    margin-left: 24vw;
    padding: 10px;
    align-content: center;
    background-color: rgba(47, 79, 79, 0.5);
    border-radius: 5px;
}
```

- **JavaScript (React.js)**: For dynamic functionality and interactive features.

```jsx
{placeName && weatherData && ( // When place name and weather data are defined.
```

- **Axios**: For handling server requests from the frontend and to fetch data from the API.

```jsx
const response = await axios.post("http://localhost:5000/weather", {
```

- **Express.js**: For working with APIs within the NodeJS framework.

```
import express from "express";
import axios from "axios";
import dotenv from "dotenv";
import cors from "cors";

dotenv.config(); // Initalize the dotenv package.

// Initalize Express application server.
const app = express();
app.use(express.json());
```

Project Structure

1. App.css: The CSS file for styling the application.
2. App.tsx: The main JavaScript file where the application logic and interactions are implemented.
3. main.tsx: Used in conjunction with index.html to render the App.tsx into the DOM.

Setup Instructions

1. Clone the repository from GitHub.
2. Run *npm install* in the terminal while in the main directory.
3. Run *npm run dev* in the terminal while in the main directory.
4. Navigate to *localhost:5173* in a browser.
5. Ensure you have an active internet connection for accessing the OpenWeatherMap API.

Usage Instructions

1. Open the application in a web browser.
2. Input the name of a location in the location input field.
3. View the current weather data for that location.

Project Outcomes

This project provided practical experience in front-end web development, particularly in working with APIs. The skills gained include:

- Implementing API functionality using OpenWeatherMap's geocoding and weather data APIs.

```
const weatherResponse = await axios.get( // Fetching weather data for the coordinates.
  `https://api.openweathermap.org/data/2.5/weather?lat=${lat}&lon=${lon}&appid=${apiKey}&units=metric`
);
```

- Enhancing user experience through interactive elements such as input fields, error handling, and responsive design.
- Developing a full-stack application using HTML, CSS, and JavaScript.