**Weather Forecast Application**

**Overview**

This project is a weather forecast application that retrieves weather data from the OpenWeatherMap API, stores it in a MySQL database, and displays it to users. The backend is built with Laravel, and the frontend is built with React. The application allows users to search for weather forecasts by city name and displays the current weather conditions, temperature, humidity, and wind speed.

**Functionality**

* **Backend Development:**
  + Used Laravel for backend development.
  + Integrated MySQL database for storing weather data.
  + Created a WeatherService class responsible for fetching weather data from the OpenWeatherMap API and storing it in the database.
* **Frontend Development:**
  + Built a user interface using React where users can enter a city name and view the weather forecast.
  + The frontend interacts with backend API endpoints to fetch and display weather data.
* **API Integration:**
  + Integrated the OpenWeatherMap API to fetch current weather data for specified cities.
  + Handled API requests, responses, and errors gracefully.
* **Additional Features:**
  + Implemented error handling for cases like invalid city names or failed API requests.

**Design Decisions**

1. **Backend with Laravel:**
   * **API Endpoint:** Created an endpoint in Laravel to fetch weather data from OpenWeatherMap and return it to the frontend.
   * **Middleware:** Implemented CORS middleware to handle cross-origin requests between the frontend and backend.
   * **Database:** Used MySQL for managing weather data due to its robustness and ease of integration.
2. **Frontend with React:**
   * **Component Design:** Designed a Weather component to manage user input, fetch weather data, and display it using Bootstrap cards for a clean UI.
   * **State Management:** Used React's useState to manage form input, API response data, and error handling.
3. **Bootstrap Integration:**
   * **Styling:** Used Bootstrap's classes to ensure the interface is responsive and visually appealing without installing additional libraries.

**Challenges Faced**

* **Middleware Registration:** Encountered difficulties registering middleware in Laravel 11 due to changes in the framework's structure.
* **API Integration:** Handling API responses and errors to provide a smooth user experience was a key challenge.
* **Error Handling:** Ensuring robust error handling on both the frontend and backend to manage edge cases like invalid city names or API request failures.

These decisions and solutions helped create a functional, user-friendly weather application with a clear separation between frontend and backend responsibilities.

**How to Run the Project**

**Backend (Laravel)**

1. **Clone the repository:**

git clone - <https://github.com/techfamabhi/weather-app.git>

cd weather-app

2 - Install dependencies

composer install

3 - Import the SQL file

weather-app.sql

#### Frontend (React)

1. **Navigate to the React directory:**

cd frontend

2 - **Install dependencies:**

npm install

3 - **Start the React development server:**

npm start

4 - **Open the application:**

* The application should be running on http://localhost:3000.

### Project Files

* **Laravel Project**: Contains the backend code (routes, controllers, models, services).
* **React Frontend**: Contains the frontend code (components, styling).
* **weather-app.sql**: SQL file to set up the MySQL database.