ReadMe

Group 9 - Weather Forecast Website

Project Overview

The **Weather Forecast Website** is a complete web application developed to display weather forecasts, leveraging machine learning models for precise predictions. The project is structured with a React-based front-end for user interaction and a Python-based back-end that handles data processing and serves the ML predictions.

Project Structure

```
group 9-Nein-Weather
Forecast Website/
├── backend/

│   ├── __pycache__/

│   ├── main.py              # Main server file

│   ├── model.py              # Contains ML model logic

│   ├── utils.py             # Utility functions for data processing

│   ├── merged_data.csv       # Processed data for training or analysis

│   ├── new_merged_data.csv   # Additional dataset for the model

│   ├── rain_model.pkl        # Pre-trained model for rain prediction

│   ├── temp_model.pkl        # Pre-trained model for temperature prediction

│   ├── randomforest_model.pkl # Random Forest model for feature analysis

│   └── ... (other model and configuration files)

├── frontend/

│   ├── public/

│   │   ├── index.html        # Main HTML file

│   │   └── ... (assets)

│   ├── src/

│   │   ├── App.js           # Main React component

│   │   ├── index.js          # Entry point for React

│   │   ├── TemperaturePage.js # Page component for temperature display
│   │   └── WeatherTypePage.js # Page component for weather type display

│   ├── package.json         # Node.js dependencies
│   └── package-lock.json

└── .venv/                  # Python virtual environment (local use)
```

Technologies Used

- **Front-End**: React.js, Chart.js, Axios, Material-UI
- **Back-End**: FastAPI, Uvicorn, Python
- **Machine Learning**: scikit-learn, numpy, joblib

Setup Instructions

Backend Setup

1. **Navigate to the backend Directory**:

   ```
   bash
   cd backend
   ```

2. **Install Required Packages and Dependencies**: Run the following command to install essential Python packages:

   ```
   bash
   pip3 install -r requirements.txt
   ```

3. **Run the Model Script**: Execute the model script to ensure the model is prepared and integrated properly.

   ```
   bash
   python3 model.py
   ```

4. **Start the FastAPI Server**: Run the following command to start the FastAPI server with live reload:

```bash
uvicorn main:app --reload
```

The server should start on http://127.0.0.1:8000.

Frontend Setup

1. **Navigate to the frontend Directory**:

```bash
cd ../frontend
```

2. **Install Frontend Dependencies**: Run these commands to install necessary npm packages:

```bash
npm install react react-dom react-router-dom @mui/material
@mui/icons-material @mui/lab @fontsource/roboto chart.js
chartjs-plugin-datalabels chartjs-plugin-zoom date-fns axios
html2canvas jspdf
```

3. **Start the Frontend Development Server**: After installation, you can start the React development server with:

```bash
npm start
```

The front-end application should be available on http://localhost:3000.

Configuration for AI Model Integration

- **Model Loading**: Verify that model.py loads models correctly and main.py has routes set for API calls.
- **Preprocessing**: The utils.py file should handle any data preprocessing before model input.
- **Data Files**: Ensure data files like merged_data.csv are formatted properly.

Running the Application

1. **Start the back-end server**:

   ```
   bash
   cd backend
   uvicorn main:app --reload
   ```

2. **Start the front-end development server**:

   ```
   bash
   cd ../frontend
   npm start
   ```

3. **Access the application**: Open your browser and navigate to http://localhost:3000 to use the web application.

Troubleshooting

- Ensure that your Python and Node.js versions meet the requirements.
- If the server fails to start, verify that all required dependencies are installed and that your environment variables are set up correctly. Refer to package.json and requirements.txt for any additional dependencies.
- For any issues with data loading or processing, check the utils.py logs and ensure that the CSV files are formatted as expected.