**CLIMAFY WEATHER API**

**BY: ADENIYI PELUMI**

A robust Weather Forecasting and Alerts REST API, built with Node.js, Express.js, TypeScript, and MongoDB, designed to deliver real-time and forecasted weather information based on user-managed locations, with options for custom weather alerts and personalized preferences.

## TECH STACK

|  |  |
| --- | --- |
| **Tech** | **Usage** |
| Node.js + Express.js | Backend Framework |
| TypeScript | Type Safety |
| MongoDB + Mongoose | NoSQL Database + ODM |
| JWT | Authentication/Authorization |
| OpenWeatherMap API | External Weather Data Provider |
| Swagger (OpenAPI) | API Documentation |
| Weather Provider | OpenWeatherMap API |
| Real time Notification | Websocket (for severe weather alerts) |
| Environment variables | dotenv |

**FEATURES**

**Authentication & Authorization**

* **JWT-based Authentication**
* User Roles:
  + **User:** Can register, manage their locations, set preferences, and receive weather forecasts and alerts.

**Location Management**

* Users can **add, view, update, and delete locations**.
* Each user manages their own list of locations.
* Weather data and alerts are personalized for the user's saved locations.

**Weather Forecasts**

* Fetch weather information from an external provider (e.g., **OpenWeatherMap API**).
* Types of forecasts:
  + **Current Weather**: Live temperature, humidity, wind speed, condition descriptions, etc.
  + **Daily Forecasts**: Up to 7-day forecasts.
  + **Hourly Forecasts**: Hour-by-hour forecasts for the next 48 hours.

**Weather Alerts**

* **Severe Weather Alerts**: Automatically fetched based on user locations.
* **Custom Alerts**: Users can define their own thresholds (e.g., notify if temp > 35°C).
* Subscriber to real time weather warnings via WebSocket

**User Preferences**

* Users can customize:
  + **Units**: Celsius and Fahrenheit
  + **Notification Settings**: Enable/Disable severe or custom alerts

### ****Real-Time Alerts via WebSocket****

**WebSocket Server Integration** (using ws library or socket.io):

* When weather data changes (especially severe warnings), all connected clients get **instant notifications**.
* No need for polling.
* Example:  
  "Severe Thunderstorm Alert in Lagos" sent automatically to subscribed users in real time.

**API DOCUMENTATION:**

Use **Swagger** for full API documentation, including endpoints, request/response bodies, and authentication details.

## FOLDER STRUCTURE (EXPLAINED)

src/

│

├── controllers/ # Handle incoming requests & send responses

│ ├── auth.controller.ts # Authentication logic (register, login)

│ ├── user.controller.ts # User preferences management

│ ├── location.controller.ts # Location CRUD operations

│ ├── weather.controller.ts # Fetch weather forecasts

(current/daily/hourly)

│ ├── alert.controller.ts # Severe & Custom alert handling

│

├── models/ # Mongoose Schemas & Models

│ ├── user.model.ts # User schema (with preferences)

│ ├── location.model.ts # Locations schema (linked to user)

│ ├── alert.model.ts # Custom weather alerts schema

│ ├── preferences.model.ts # Preferences (units, notifications) schema

│

├── routes/ # Route Definitions (Express Router)

│ ├── auth.route.ts # Routes: /auth/login, /auth/register

│ ├── user.route.ts # Routes: User preference management

│ ├── location.route.ts # Routes: Location CRUD

│ ├── weather.route.ts # Routes: Weather fetching

│ ├── alert.route.ts # Routes: Alerts handling

│

├── middlewares/ # Express Middlewares

│ ├── auth.middleware.ts # JWT Authentication middleware

├── error.middleware.ts # central error handling

│

├── services/ # Core Business Logic

│ ├── user.service.ts # User and preferences handling

│ ├── location.service.ts # Location handling logic

│ ├── weather.service.ts # External Weather API data fetching

│ ├── alert.service.ts # Alert creation & evaluation logic

│

├── utils/ # Utility Functions

│ ├── token.utils.ts # JWT generation and verification

│ ├── unit.utils.ts # Converts temperature, wind, etc. based on

user preferences

├── websocket.utils.ts # Websocket Server & broadcasting

├── openwerather.utils.ts # OpenWeatherMap API request helper

│

├── docs/ # Swagger API Documentation

│ └── swagger.ts # OpenAPI 3.0 documentation setup

│

├── config/ # Configuration Files

│ ├── db.config.ts # MongoDB connection setup

│ ├── weather.config.ts # OpenWeatherMap API configuration

│

├── app.ts # Express App Configuration

└── server.ts # Server Entry Point