PRODUCT REQUIREMENTS DOCUMENT (PRD)

CLIMAFY WEATHER API

1. PRODUCT OVERVIEW

The **Climafy Weather API** is a robust, RESTful and WebSocket-enabled weather service that provides personalized, real-time, and forecasted weather data. It allows authenticated users to manage their own locations, set custom weather alerts, and configure notification preferences. The API integrates with OpenWeatherMap for weather data and leverages WebSocket for real-time severe weather notifications.

2. CUSTOMER PROBLEM STATEMENT

- Users lack real-time severe weather notifications for their specific locations.
- Existing weather services provide generic and delayed data, relying on polling.
- Users cannot define customized weather alert thresholds (e.g., temperature, wind, humidity).
- Most weather apps tie forecasts to only the current location, with no user-managed location lists.
- Users have limited control over unit preferences (Celsius/Fahrenheit) and alert notification settings.

3. PRODUCT OBJECTIVES

- Provide **real-time and forecasted weather data** personalized to user-selected locations.
- Allow users to create **custom weather alerts** with specific condition thresholds.
- Enable **real-time severe weather alerts** via WebSocket without polling.
- Support **user-managed location lists** (CRUD operations).
- Allow customization of **units (Celsius/Fahrenheit)** and **alert notification preferences**.

4. KEY FEATURES

Feature	Description	Priority
User Authentication & Authorization	JWT-based login, registration, role-based access control.	Must-Have
Location Management	Users can add, update, delete, and view multiple locations.	Must-Have
Weather Forecast Retrieval	Fetch current, hourly (48 hours), and daily (7 days) forecasts using OpenWeatherMap API.	Must-Have
Severe Weather Alerts	Automatically fetch severe alerts based on user locations; real-time delivery via WebSocket.	Must-Have
Custom Alerts	Users define thresholds for specific weather conditions (e.g., temp > 35°C) and receive notifications.	Must-Have
User Preferences	Users set preferred units (°C/°F), enable/disable severe and custom alerts.	Must-Have
Real-Time Notifications via WebSocket	Severe weather warnings broadcasted instantly to connected clients.	Must-Have
API Documentation	Complete Swagger (OpenAPI 3.0) documentation for all endpoints.	Must-Have

5. TECHNICAL REQUIREMENTS

- **Backend Framework:** Node.js + Express.js
- **Language:** TypeScript
- **Database:** MongoDB (via Mongoose)
- **Authentication:** JWT-based
- **External Data:** OpenWeatherMap API
- **Real-Time Communication:** WebSocket (e.g., ws or socket.io)
- **Environment Config:** dotenv
- **API Documentation:** Swagger (OpenAPI 3.0)

6. NON-FUNCTIONAL REQUIREMENTS

- **Scalability:** Must support hundreds of concurrent WebSocket connections.
- **Reliability:** Severe weather alerts should be delivered in real-time with <2s latency.
- **Security:** Secure user authentication, authorization, and data handling.
- **Maintainability:** Modular codebase with clearly separated controllers, services, and models.

7. SUCCESS METRICS

- **API Response Time:** < 500ms for REST requests.
- **WebSocket Latency:** < 2 seconds for alert delivery.
- **User Engagement:** 90% of active users have at least 1 custom alert set.
- **Error Rate:** < 1% failed API or WebSocket requests.

8. OUT OF SCOPE (FOR MVP)

- Admin Dashboard for user management.
- Mobile app client integration.
- Multi-language support.

9. RISKS & MITIGATIONS

Risk	**Mitigation**
Third-party API (OpenWeatherMap) downtime	Implement retries and fallback error handling.
WebSocket overload	Use clustering and connection limit policies.
Data privacy concerns	Ensure encrypted JWT tokens and secured environment variables.