Software Requirement Specifications

1. Introduction

1.1 Purpose

This SRS document outlines the requirements and design specifications for MoMe – a real-time travel risk monitoring app that provides drivers with early warnings of hazardous weather conditions (hail, flood, storms, etc.) along their routes. The goal is to reduce vehicle damage and insurance claims by enabling safe rerouting and timely awareness.

1.2 Scope

MoMe is a web and mobile solution that:

- Uses GPS and route input to monitor user travel.
- Integrates weather APIs (e.g., OpenWeatherMap) for hazard detection.
- Alerts users of threats with real-time, actionable notifications.
- Offers vehicle-specific customization and risk scoring dashboards.
- Optionally gamifies safe driving behavior and enables insurance reward simulation.

1.3 Intended Audience

- Hackathon judges
- Developers
- Stakeholders in insurance and mobility tech

1.4 Definitions

- **Risk Score**: A numerical indicator of the travel danger level based on weather severity.

- **Hazard**: A potentially dangerous weather condition (hail, heavy rain, floods, high wind).

2. Overall Description

2.1 Product Perspective

MoMe is an independent application that interacts with external weather APIs, geolocation services, and cloud infrastructure (AWS). The frontend is React-based; the backend is powered by Node.js.

2.2 User Classes and Characteristics

- **Commuters**: Need to safely get to work or home.
- **Delivery Drivers**: Sensitive to timing and route disruptions.
- **Rural Drivers**: Higher vulnerability to weather extremes and limited network access.

2.3 Operating Environment

- **Frontend**: React (Web), optional mobile (PWA)
- **Backend**: Node.js + Express
- **Database**: AWS RDS (PostgreSQL)
- **Deployment**: AWS EC2 / Beanstalk
- **APIs**: OpenWeatherMap, Google Maps API

2.4 Constraints

- Must work with public/free weather APIs.
- Needs reliable GPS access.
- Offline mode requires pre-caching of data.

2.5 Assumptions and Dependencies

- Users have internet and GPS access.

- Weather APIs provide accurate forecasts.

3. Functional Requirements

3.1 Use Case: View Weather Risk for Current Location

- **Actor**: User
- **Description**: App fetches geolocation and displays current hazards with severity.

3.2 Use Case: Simulate Travel Route and Receive Risk Alerts

- **Actor**: User
- **Description**: User inputs a route and receives alerts for high-risk areas ahead.

3.3 Use Case: Real-Time Travel Monitoring and Alerts

- **Actor**: User
- **Description**: App runs in the background and notifies user of incoming threats.

3.4 Use Case: Customize Vehicle Risk Preferences

- **Actor**: User
- **Description**: User sets vehicle sensitivity to specific weather conditions.

3.5 Use Case: View Risk Score Dashboard

- **Actor**: User
- **Description**: User views historical data, travel risks, and behavior insights.

3.6 (Stretch) Use Case: Earn Gamified Rewards

- **Actor**: User
- **Description**: User earns badges or simulated discounts for safe driving decisions.

4. Non-Functional Requirements

4.1 Performance Requirements

- Route risk assessment results should be delivered within 3 seconds.
- Alerts must trigger with <5 seconds delay from hazard detection.

4.2 Reliability

- System must be operational 99% of the time.
- Critical alerts should function even under degraded conditions.

4.3 Usability

- Interface must be simple, accessible, and intuitive.
- Mobile responsiveness is required.

4.4 Security

- JWT authentication
- AES encryption of stored travel history
- User opt-in for analytics

4.5 Maintainability

- Modular API design for extensibility
- Clear documentation for future development

5. External Interfaces

5.1 Weather API

- OpenWeatherMap for live hazard data

5.2 Maps API

- Google Maps API or Leaflet for route visualization

5.3 GPS Geolocation

- Browser or mobile GPS for location tracking

5.4 Notification System

- In-app banners or modal popups
- Optional: push notifications or sound alerts

6. Future Enhancements

- AI/ML-based predictive modeling for risk anticipation
- Integration with insurance platforms
- Offline support with pre-cached route risks
- Multilingual support for South African languages