SOFTWARE REQUIREMENTS SPECIFICATION (SRS) DOCUMENT

FOR DISASTER MANAGEMENT SYSTEM APP

Table of Contents

1. INTRODUCTION

1.1 Purpose

This document outlines the requirements for a mobile/web-based Disaster Management System that enables early warning, real-time alerts, victim reporting, emergency response coordination, and data management for disasters such as floods, fire outbreaks, earthquakes, landslides, and others so as to help prevent disaster from affecting Users.

1.2 Scope

Allow Users to Report Disasters

Explanation:
Users can report incidents they witness or experience, such as floods, fires, earthquakes, or landslides. This includes:

- Selecting the type of disaster
- Pinpointing the location (via GPS or map)
- Uploading images, videos, or descriptions
- Submitting the report in real time
- Being able to submit I local or Native language.

Purpose:
To collect ground-level data quickly, helping authorities verify and act promptly.

Provide Alerts and Early Warnings

Explanation:
The app will push notifications to users about impending or ongoing disasters using in that particular area or location or town:

- Weather and geological data (from integrated APIs or sensors)
- Admin-triggered alerts based on verified reports

Purpose**:**
To reduce casualties by informing people early and guiding them on what to do (e.g., evacuate, stay indoors).

Enable Government and Emergency Services to Coordinate Responses

Explanation:
Admins and first responders will:

- Access a dashboard showing all reports and locations
- Assign response units (fire, ambulance, police)
- Track the status of ongoing disasters

Purpose:
To streamline response times, allocate resources efficiently, and communicate internally.

Display Affected Areas on Maps Stating the Type of Disaster Occurring

Explanation:
Interactive maps will show for the particular area suffering from that disaster:

- Real-time disaster zones (heatmaps or icons)
- Type of disaster (color-coded)
- Verified vs. unverified reports

Purpose:
To give users and responders visual insights into what's happening, where, and how severe it is.

Provide Emergency Contacts and Safety Tips

Explanation:
Static pages and pop-ups will offer:

Hotline numbers (fire service, police, hospitals, shelters)

Safety measures for each disaster type (before, during, after)

Purpose:
To empower users with immediate help and self-protection steps during critical moments.

Maintain a Disaster History Database for Analysis

Explanation:
All reports, responses, and outcomes will be stored with:

- Date/time stamps
- Location
- Type of disaster
- Response taken

Purpose:
For future planning, government policy, academic research, and to identify high-risk areas and response gaps.

1.3 Definitions

User: Citizen or resident using the app.

Admin: Authorized personnel managing data and alerts.

First Responders: Firefighters, police, medics, NGO's Community service (Community helep) etc.

Disaster: Natural or human-made emergency (e.g., earthquake, flood).

2. OVERALL DESCRIPTION

2.1 Product Perspective

The app will function as a centralized platform accessible via mobile and web, with backend support for data management and integration with notification systems and the availability of sensors for most disasters.

2.2 User Classes

General Users: Receive alerts, report incidents.

Admin/Disaster Management Authority: Manage events, verify reports via available google maps on the systems.

First Responders: View locations and take action.

2.3 Assumptions and Dependencies

Internet/GPS access is available.

Users have Android or iOS devices.

Government and emergency agencies cooperate.

3. Functional Requirements

FR1: User Registration/Login

Users must be able to register and log in using email or phone number.

FR2: Real-Time Alerts

Users receive alerts based on location and disaster type in particular area.

FR3: Report Disaster

Users can submit reports including photos, video, location, and type of disaster.

FR4: Admin Dashboard

Admins can view reports, approve or verify incidents, and push alerts.

FR5: First Responder Interface

Displays verified incidents with directions.

FR6: Emergency Contacts and Safety Tips

Static pages with emergency numbers and disaster safety info.

FR7: Map Integration

Displays disaster zones and reported incidents.

FR8: Disaster History Logs

Store and retrieve previous incidents with date, time, and location.

FR9: Community Helep

Request community help

4. Non-Functional Requirements

NFR1: Performance

Must support at least 10,000 concurrent users.

NFR2: Security

Data must be encrypted and user authentication is required.

NFR3: Availability

99.5% uptime with support for cloud hosting and backups.

NFR4: Usability

Intuitive UI/UX with accessibility support.

NFR5: Scalability

Ability to scale up to multiple regions and disaster types.

5. System Architecture

Frontend: React (Web), React Native (Mobile)

Backend: Node.js/Express or Django

Database: MySQL/PostgreSQL

Notification Service: Firebase Cloud Messaging (FCM)

Map API: Google Maps or OpenStreetMap

6. Data Flow Diagram (Level 1) Here's a textual description of the Level 1 DFD

Processes:

User Management

Register/Login

Profile info

Disaster Reporting

Input: Disaster type, location, media

Output: Report entry in database

Alert Distribution

Input: Disaster data

Output: Push notifications, map updates

Admin Actions

Verify report

Send mass alerts

Response Coordination

Notify first responders

Update status logs

Data Stores:

User Database

Disaster Reports

Verified Disasters

Response Logs

Safety Info

External Entities:

Users

Admin

First Responders

Notification Services (e.g., FCM)