1.0 Introduction

This document outlines the technical blueprint for ResQNet, an integrated, MULTI-LINGUAL software platform engineered to enhance disaster response efforts during landslide events. The system is designed to facilitate resilient communication, strategic coordination, and data-driven decision-making for three key user groups: affected Citizens, coordinating NGOs, and strategic Government agencies.

2.0 Core Technology Architecture

To ensure scalability, reliability, and financial viability (leveraging open-source solutions), the project will be built upon the following technology stack:

- **Mobile Application (User):** React Native. This framework allows us to develop a single, high-performance application that runs natively on both iOS and Android devices.
- **Web Dashboards (NGO/Govt):** React.js. A robust library for building the complex, interactive, and real-time user interfaces required by our command-and-control dashboards.
- **Backend API (The "Engine"):** Node.js or Python. This will be the central server that handles all business logic, data processing, and communication between our app and dashboards.
- **Primary Database:** PostgreSQL with the PostGIS extension. A powerful, open-source database that is crucial for our project, as PostGIS gives it the ability to store, index, and query geographic locations efficiently.
- **Secondary Database:** MongoDB. A flexible database well-suited for handling unstructured data, such as our system logs or the metadata for user-uploaded photos and videos.
- **Geospatial Mapping:** OpenStreetMap (OSM)/Mapbox with Leaflet.js. This combination provides a 100% free and highly capable alternative to paid mapping solutions, allowing us to display rich, interactive maps.
- **Real-time Communication:** Socket.IO. A library that enables our backend to send information directly to the web dashboards. This is what will make the dashboards update live (e.g., a new report appearing) without users needing to press "refresh."
- **Push Notifications:** Firebase Cloud Messaging (FCM). The industry-standard, free solution for sending critical push notification alerts to our mobile app users.
- **External Communication Gateway:** Twilio API. We will use a trial/developer account to integrate SMS and WhatsApp capabilities, providing a critical reporting channel for users without the app or a data connection.

3.0 Feature Specification: The User (Citizen) Interface

The mobile application is the primary data-gathering and citizen-safety tool. Its design must prioritize simplicity and functionality, especially in low-connectivity environments.

1. **Asynchronous SOS Beacon:**
    - **Feature:** A primary, one-press SOS button.
    - **Implementation:** The app will capture the user's GPS coordinates and a timestamp. This report is saved locally on the device first. A background service will then persistently monitor for any network connection and automatically upload the queued reports once a signal is found. This offline-first design is critical for reliability.
2. **Multi-Channel Distress Reporting:**
    - **Feature:** Allows users without the app to report emergencies.
    - **Implementation:** We will configure a dedicated phone number (via Twilio). When a user texts "HELP" with their details to this number, our backend will receive it, parse the information, and place it on the central map alongside all other reports.
3. **Geotagged Multimedia Reporting:**
    - **Feature:** Users can provide vital context by attaching photos, videos, or audio clips.
    - **Implementation:** To keep the database fast, media files will be uploaded to a separate, open-source storage server (e.g., MinIO). The database will only store a link to this media, associated with the geographic coordinates of the report.
4. **Dynamic Offline Maps & Safety Cache:**
    - **Feature:** Provides access to essential information even with zero network connectivity.
    - **Implementation:** The app will be designed to download and store a "safety packet" for the user's district. This includes offline map tiles (from OpenStreetMap) and a file containing the locations of all known shelters and hospitals. The app will automatically detect an offline state and switch to using this cached data.
5. **Localized Weather & Hazard Alerts**
    - **Feature**: Provides real-time, location-specific weather forecasts and official hazard warnings.
    - **Implementation**: The app will use the user's GPS coordinates to query our backend, which in turn fetches data from the IMD (India Meteorological Department). The app will display a simple forecast (e.g., "Heavy rain expected in 2 hours") and prominently show any official alerts (e.g., "Red Alert: Flash Floods") issued by the government.
6. **P2P Mesh Networking (Advanced/Research Goal):**
    - **Feature:** In a total-blackout scenario, nearby devices can relay SOS messages to find a "gateway" device that has a signal.
    - **Implementation:** This is a high-level goal, ideal for collaboration with our MNC partners. It involves using Bluetooth/Wi-Fi Direct for device-to-device

communication to pass message packets until one device finds a network and can upload them.

4.0 Feature Specification: The NGO Interface

This web-based dashboard is the tactical command center for on-the-ground rescue teams. It is built for real-time coordination and case management.

1. **Unified Real-Time Triage Dashboard:**
    - **Feature:** A single, live-updating screen showing all incoming distress reports from all channels.
    - **Implementation:** Using Socket.IO, the dashboard will have a direct, persistent connection to the server. New reports will appear on the list and map instantly, allowing for immediate triage and prioritization.
2. **Live GIS Heatmap & Report Clustering:**
    - **Feature:** A smart map visualization that prevents clutter, reveals patterns, and assists navigation.
    - **Implementation:** The map will use clustering to group dense collections of reports into a single, clickable icon (e.g., "15 reports"). It will also display a heatmap to provide an immediate visual understanding of the disaster's epicenter and scale. Furthermore, geotagged reports from citizens (e.g., photos of blocked roads) will be used to dynamically update a routing layer on the map, providing suggested alternate navigation routes for rescue teams.
3. **Case Management & Dispatch:**
    - **Feature:** Enables operators to track a report from "new" to "resolved."
    - **Implementation:** An operator can "claim" a new report, change its status (e.g., Pending, Assigned, Resolved), and assign it to a specific volunteer team. This ensures accountability and prevents two teams from being dispatched to the same incident.
4. **Integrated Tactical Weather Forecast:**
    - **Feature:** A dashboard widget showing the localized forecast for the operational area.
    - **Implementation:** A dedicated component on the dashboard will call the same IMD-powered API as the user app. This allows the dispatch team to make critical tactical decisions (e.g., "We need to move Team B before the heavy rain hits in 1 hour" or "Air operations are impossible for the next 4 hours").
5. **Resource & Volunteer Inventory:**
    - **Feature:** A simple database for managing available assets.
    - **Implementation:** A management tab where NGOs can log their available resources (e.g., vehicles, medical supplies) and personnel (e.g., doctors, drivers) to aid in dispatch planning.

5.0 Feature Specification: The Government Interface

This web-based dashboard is the high-level, strategic view for officials. It focuses on analytics, resource allocation, and mass communication.

1. **Aggregated Analytics Dashboard:**
    - **Feature:** A "bird's-eye view" of the disaster, showing trends rather than individual reports.
    - **Implementation:** This dashboard will display key metrics using charts (e.g., "Reports per Hour," "Most Affected Districts," "Resource Allocation Status"). This data will be generated by running analytical queries on our PostgreSQL database.
2. **Predictive Vulnerability Model (MNC Collaboration):**
    - **Feature:** A data-driven tool to help predict where the next critical area might be.
    - **Implementation:** This is a key data science component. We will train a model (using Python) by combining our static data (e.g., terrain data from **ISRO/NRSC**, population density, elevation) with real-time data inputs (e.g., rainfall data from **IMD**, live reports from our heatmaps). The model can then flag areas that are highly vulnerable and are just beginning to see a spike in reports, enabling proactive resource deployment.
3. **Multi-Channel Mass Communication Tool:**
    - **Feature:** An authenticated tool for officials to broadcast verified alerts to all users.
    - **Implementation:** A secure form where an official can type a message. On "Send," our backend will simultaneously use Firebase to send a push notification to all app users and use Twilio to send an SMS to all numbers in our database.
4. **Inter-Agency Data Sharing & Audit Trail:**
    - **Feature:** A secure, professional system for accountability and collaboration.
    - **Implementation:** We will provide a secure, read-only API for other government agencies (e.g., police, medical) to access our data. Furthermore, our database will be configured with triggers to log every action (e.g., who assigned what report, when a broadcast was sent), creating a full audit trail for post-disaster analysis and accountability.