Software Requirements Specification (SRS) for Project

---

1. Introduction

1.1 Purpose

This document provides a detailed description of the requirements for **Project Sentinel**, a digital health platform designed to combat water-borne diseases in the rural and tribal belts of the Northeastern Region (NER) of India. The platform aims to enable early detection, monitoring, and prediction of disease outbreaks to facilitate timely intervention by health authorities.

1.2 Document Conventions

This document follows a structure inspired by the IEEE Std 830-1998 standard for Software Requirements Specifications. All requirements are identified with a unique code (e.g., FR-01) for traceability.

1.3 Intended Audience

This SRS is intended for all project stakeholders, including:

- **Development Team:** Backend, Frontend, Mobile, and AI/ML engineers.
- **Project Managers:** For project planning, tracking, and management.
- **SIH Team Members:** To have a centralized, formal record of the project plan.
- **Testers:** To create test cases and plans.
- **Health Authorities & ASHA Workers:** To understand the system's capabilities and their roles within it.

1.4 Project Scope

The project focuses on developing a holistic system that integrates data collection, analysis, and communication to prevent and control water-borne disease outbreaks.

In-Scope:

- A mobile application for field data collection by ASHA workers.
- An SMS/IVR system for community health reporting.
- Integration with water quality sensors and manual test kits.
- An AI/ML engine to predict potential outbreaks based on health, environmental, and water quality data.
- A real-time alert system for health officials and community leaders.
- An administrative dashboard for data visualization and monitoring.
- Educational modules for hygiene awareness.

Out-of-Scope:

- Manufacturing of hardware (e.g., water sensors).
- Providing medical treatment or diagnosis directly to patients.
- Management of medical supply chains beyond providing resource allocation recommendations.

---

2. Overall Description

2.1 Product Perspective

Project Sentinel is a self-contained system that will serve as a critical information bridge between communities, field health workers (ASHA), and regional health authorities. It will replace slow, error-prone manual record-keeping with a streamlined, digital-first workflow. It will interface with external weather and mapping APIs to enrich its dataset for more accurate predictions.

2.2 Product Functions

- **Data Collection:** Gathers data on symptoms, water quality, and environmental factors via a mobile app and SMS/IVR.
- **Data Storage & Management:** Securely stores and organizes collected data in a centralized database.
- **Predictive Analysis:** Utilizes AI/ML models to identify high-risk areas and predict potential outbreaks.
- **Alerting:** Automatically notifies stakeholders (health officials, ASHA workers) about predicted outbreaks or critical water quality levels.
- **Visualization & Reporting:** Presents data through an intuitive web dashboard with maps, graphs, and reports.

- **Community Engagement:** Provides educational content and enables community-driven reporting.

2.3 User Characteristics

1. **ASHA Workers/Field Health Staff:** Have basic smartphone literacy. Will use the mobile app for their daily data entry and to receive alerts. Require an intuitive interface and offline capabilities.
2. **Community Members:** May have low smartphone penetration and varying literacy levels. Will interact primarily through a simple SMS or IVR-based system to report symptoms.
3. **Health Officials/Administrators (Ministry):** Are tech-savvy and require a comprehensive overview of the health situation. Will use the web dashboard to monitor trends, view hotspots, and make data-driven decisions.

2.4 Constraints

- **Connectivity:** The system must operate effectively in areas with low or intermittent internet connectivity.
- **Hardware:** The solution must rely on low-cost, easily maintainable hardware (smartphones, basic sensors).
- **Security:** All sensitive health data must be encrypted both in transit and at rest.
- **Language:** The system must support multiple languages, including English, Hindi, and relevant tribal languages.
- **Usability:** Interfaces must be extremely simple and intuitive, especially for community members and field workers.

2.5 Assumptions and Dependencies

- ASHA workers will have access to basic Android smartphones.
- Health authorities are willing to adopt and integrate this system into their workflow.
- Accurate weather and geolocation data will be available via external APIs.
- A clear definition of an "outbreak" threshold can be established and refined over time.

---

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

- **Mobile Application (React Native):**
  - Simple, form-based data entry for symptoms and water quality.
  - Offline data storage and auto-sync capability.
  - Display of alerts and educational content.
  - Secure login for authorized ASHA workers.
- **Web Dashboard (React):**
  - Interactive map view showing villages, hotspots, and health data.
  - Graphs and charts for trend analysis (cases over time, symptoms distribution).
  - A filterable list of all generated alerts with severity levels.
  - User management and system configuration panels.

3.1.2 Hardware Interfaces

- The system will be designed to accept data from low-cost water quality sensors (for parameters like pH, Turbidity, TDS). The mobile app will serve as the primary interface for inputting this data, whether connected via Bluetooth (future scope) or entered manually from a device's display.

3.1.3 Software Interfaces

- **Twilio API:** For sending SMS and WhatsApp alerts to users and officials. Also used for the SMS/IVR based reporting system.
- **Weather API:** To fetch real-time and historical weather data (e.g., rainfall, humidity) for correlation with disease outbreaks.
- **Geolocation/Maps API:** To tag data with precise locations, calculate distances between villages, and render data on the dashboard map.

3.1.4 Communications Interfaces

- **SMS:** For community reporting and sending critical alerts to all user types.
- **IVR (Interactive Voice Response):** An alternative to SMS for community reporting, catering to users with low literacy.

- **Push Notifications:** For real-time alerts to mobile app users (ASHA workers).
- **WhatsApp:** As a secondary channel for sending richer alerts (e.g., with links or images) to health officials.

3.2 Functional Requirements

FR-01: Data Collection

- The mobile app shall allow ASHA workers to submit reports containing:
  - Patient symptoms (e.g., diarrhea, vomiting, fever).
  - Village/Location details.
  - Water source information.
- The system shall provide an SMS/IVR number for community members to report symptoms by sending a pre-defined code or answering voice prompts.
- The mobile app shall allow ASHA workers to input results from manual water quality test kits (e.g., pH, TDS, bacterial presence).

FR-02: AI-Powered Outbreak Prediction

- The system shall process collected data daily to clean, preprocess, and featurize it.
- Features will include: 7/14-day case counts, rainfall data, water quality metrics, seasonal flags, and proximity to nearby cases.
- An ML model (e.g., XGBoost, Random Forest) will be trained to predict the probability of an outbreak in a given area. An outbreak is defined as a case count exceeding a dynamic threshold within a specific timeframe.
- The model will be evaluated with a strong focus on high Recall to minimize missed outbreaks.

FR-03: Real-time Alerting System

- If the ML model predicts a high outbreak probability or if a simple rule-based threshold (e.g., cases double in a week) is met, the system shall automatically generate an alert.
- Alerts shall be sent via SMS, Push Notification, and/or WhatsApp to pre-defined health officials and ASHA workers for the affected region.
- The alert message will contain the location, the nature of the risk (e.g., "High risk of Cholera outbreak"), and a severity level.

FR-04: Interactive Dashboard

- The dashboard shall display a map view with color-coded indicators for villages based on their current risk level (hotspots).
- It shall provide time-series graphs to visualize case trends against environmental factors like rainfall.
- It must allow administrators to filter data by district, symptom, date range, and alert status.

FR-05: Offline Functionality

- The mobile app must function without an active internet connection.
- All data entered offline must be stored locally on the device.
- The app must automatically synchronize the stored data with the central server when connectivity is restored.

FR-06: Multilingual Support

- All user-facing interfaces (Mobile App, SMS/IVR, Dashboard) must support at least English, Hindi, and one or more specified tribal languages.

3.3 Non-Functional Requirements

- **NFR-01 (Performance):** The system must be optimized for low-bandwidth environments. API responses should be lightweight.
- **NFR-02 (Security):** All user data must be encrypted in transit (using TLS) and at rest. Access control must be role-based.
- **NFR-03 (Scalability):** The architecture must be scalable to support a growing number of villages, users, and data points.
- **NFR-04 (Maintainability):** The system should be built with clean, modular code and follow best practices for the chosen tech stack to ensure easy maintenance.

---

4. System Architecture & Workflow

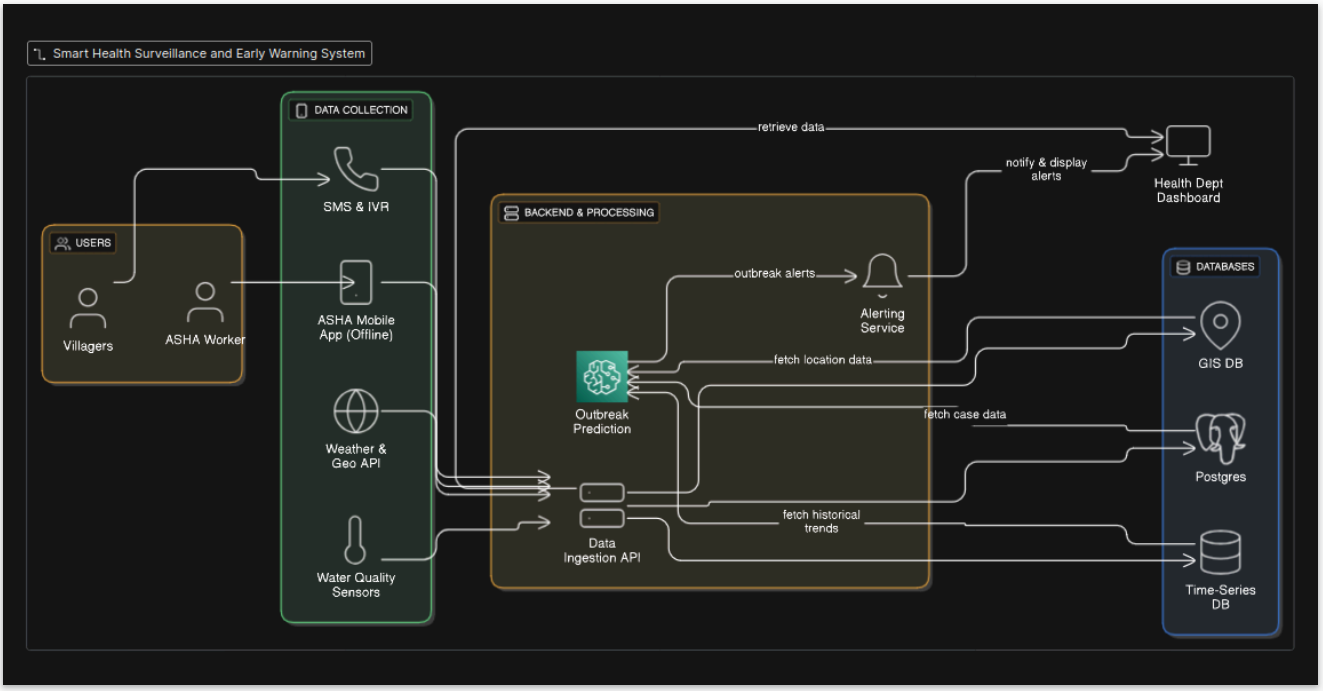
4.1 High-Level Architecture

The system is composed of five main parts:

1. **Data Collection Layer:** Mobile App (React Native), SMS/IVR Gateway (Twilio), and manual inputs.
2. **Data Storage Layer:** A PostgreSQL database with PostGIS for geospatial data and a time-series database/extension for sensor and health data.
3. **Processing & ML Layer (Backend):** A Django-based application that handles data cleaning, feature engineering, and runs the Python-based ML models for prediction.
4. **Alerting Layer:** A notification service integrated with Twilio (SMS, WhatsApp) and Push Notification services.
5. **Presentation Layer:** A React-based web dashboard for administrators.

4.2 Visual Workflow

The following diagram illustrates the high-level data and action flow of the system, as sketched out during the initial planning phase.



4.3 Data Flow

1. **Input:** ASHA workers and community members submit health and water quality data. Weather APIs provide environmental data.
2. **Storage:** Data is sent to the Django backend and stored in the PostgreSQL database.
3. **Processing:** A scheduled daily job fetches the latest data, preprocesses it, and feeds it to the ML model.
4. **Prediction & Alert:** The model outputs a risk score for each location. If the score exceeds the threshold, the alerting layer is triggered.

5. **Action & Visualization:** Alerts are dispatched to relevant personnel. The ASHA worker on the ground performs initial actions (e.g., checking/cleaning water tank). All data and risk levels are updated on the dashboard for official review.

---

6. Technology Stack

Component	Technology
Backend	Django (Python)
Frontend (Web)	React
Mobile App	React Native, Expo
Database	PostgreSQL with PostGIS extension
AI/ML	Python (Pandas, Scikit-learn, XGBoost, etc.)
Alerting & SMS	Twilio API
External APIs	Geolocation API, Maps API, Weather API