**Software Requirements Document (SRD)**

**Project Title: Health Equalizer**

**Hackathon Dates: June 28, 2024 - June 30, 2024**

**1. Introduction**

1.1 Purpose

The purpose of the Health Equalizer is to enhance healthcare accessibility by leveraging AI to connect users with medical resources tailored to their needs and location, improving healthcare outcomes in underserved communities.

1.2 Scope

The Health Equalizer platform will feature:

Personalized health query responses via NLP.

A locator for healthcare providers and emergency services using Google Maps API.

A community forum for health discussions and support.

**2. Overall Description**

2.1 User Needs

Users require reliable, easy-to-access health information and services tailored to their economic circumstances and geographic location.

2.2 Assumptions and Dependencies

Access to OpenAI API for NLP functionalities.

Google Maps API for location services.

Reliable internet access for users.

**3. System Features and Requirements**

3.1 Functional Requirements

User Interface

Reactive and dynamic UI built with React.js.

Mobile-responsive design.

Backend Services

Manage API, database, and AI operations using Python with Flask.

Integration of OpenAI API for processing and responding to user queries.

Database

Use MongoDB to store user data, health queries, and healthcare provider information.

Personalized Health Queries

NLP to interpret and respond to user inquiries.

Input can be via text or voice.

Healthcare Provider Locator

Integration with Google Maps API to display nearby healthcare services.

Community Forum

Features for posting, commenting, and moderating discussions.

Emergency Services Locator

Quick access feature within the app to locate nearest emergency services.

3.2 Non-Functional Requirements

Performance: Responses must be provided within 2 seconds of query submission.

Usability: Easy navigation for all ages.

Reliability: System should be up 99% of the time.

Security: Ensure data privacy and secure handling of user information.

**4. External Interface Requirements**

User Interfaces: Web-based client accessible via standard web browsers.

Hardware Interfaces: No hardware interface required.

Software Interfaces:

Python 3.8 or higher.

React.js for frontend.

Flask for backend.

MongoDB database.

**5. Other Requirements**

Legal: Compliance with data protection regulations (e.g., GDPR, HIPAA).

Project Submission Requirements:

Description document.

Video presentation of the prototype.

Code repository (if applicable).

**6. Milestones and Timeline**

Day 1: Setup environment, initialize all APIs, and start backend and frontend basic structures.

Day 2: Complete core functionalities, integrate all systems, test, and prepare presentation materials.