Software Requirements Specification

for

MediHelp

Version 1.02 approved

Prepared by Saher Dev, Bhargav Shekokar, Namya Dhingra, Nitish Gupta, Devesh Labana, Ishan Bhambare

IITJodhpur

4th March, 2025

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Medihelp SRS v1.01	**10-02-25**	**This is the first draft of our MediHelp SRS Document.**	**V 1.01**
Medihelp SRS v1.02	**04-03-25**	**Changes made on research basis, clarity on using features to implement functional requirements.**	**v1.02**

1. Introduction

1.1 Purpose

The software requirements in this document pertain to MediHelp, an innovative healthcare platform designed to improve patient-provider interactions using advanced technology. This system aims to streamline communication, enhance accessibility, and ensure the secure sharing of critical health information. The primary scope covered by this Software Requirements Specification (SRS) includes core features such as digital prescription management, appointment scheduling, emergency assistance, and a patient-doctor community. These features are designed to aid users—especially students at IIT Jodhpur, located far from healthcare facilities, and rural populations—by providing a secure way to connect with healthcare providers, pharmacies, and emergency services.

1.2 Document Conventions

This document follows standard IEEE guidelines and uses the following conventions:

- **Sectioning and Numbering:** Each major section is numbered (e.g., 1, 2, 3) with sub-sections (e.g., 1.1, 1.2) for clarity and traceability.
- **Requirement Identification:** Every functional requirement is tagged with a unique identifier (e.g., REQ-PG-1) to facilitate traceability during design, implementation, and testing.
- **Priority Ratings:** Requirements are assigned priorities (High, Medium, Low) to indicate their relative importance. When available, benefit, penalty, cost, and risk are quantified on a scale of 1 to 9.
- **Placeholder “TBD”:** “TBD” is used for items or details pending final decisions.
- **Terminology:** Definitions and acronyms are listed in Appendix A (Glossary).

1.3 Intended Audience and Reading Suggestions

This document is for various readers, each with specific interests and needs:

- **The developer's role** centers on technical specifications, system design, and the underlying infrastructure.
- **Project Manager**: Interested in overall project goals, scope, timelines, and milestones.
- **Marketing Staff**: To understand the product's features and target audience to create relevant campaigns.
- **Users**: Students at IIT Jodhpur and rural residents who will benefit from using MediHelp.
- **Testers**: Focus on ensuring the product meets functional requirements, such as appointment scheduling and prescription management.
- **Documentation Writers**: To understand system features for creating user manuals and help documentation.

The document is structured to begin with an overview, including the purpose and scope of the product. Afterward, it delves into the specific functional and non-functional requirements, system design, and implementation details, with sections most pertinent to each reader type. It is suggested to read the overview first and proceed to the most relevant sections based on your role.

1.4 Product Scope

MediHelp is a comprehensive healthcare platform designed to bridge gaps in healthcare access, particularly for students at IIT Jodhpur and rural populations. Its purpose is to offer a streamlined, secure, and user-friendly interface for managing healthcare-related tasks, such as appointment scheduling, medication tracking, and emergency assistance. MediHelp provides value by connecting users to doctors, pharmacies, and healthcare services in their vicinity, ensuring timely access to care and supporting seamless communication. The platform aligns with IIT Jodhpur's vision of enhancing the well-being of its students by making healthcare more accessible, even from remote locations.

1.5 References

- **IEEE Std 830-1998:** IEEE Recommended Practice for Software Requirements Specifications.
- **IEEE Std 29148-2011:** Systems and Software Engineering – Life Cycle Processes – Requirements Engineering.
- **MediHelp Vision Document:** (Version 1.02, Date: TBD)
- **User Interface Style Guidelines:** (Version 1.02, Date: TBD)
- **Regulatory Guidelines:** HIPAA, GDPR, and local healthcare standards.

2. Overall Description

2.1 Product Perspective

MediHelp is a comprehensive healthcare platform that brings together patients, doctors, and pharmacies to make medical services more accessible and efficient. It streamlines important processes like scheduling appointments, ordering medications, managing prescriptions, and providing emergency SOS support. Patients can chat with doctors they know and participate in a supportive community that consists of both pharmacies and medical professionals. MediHelp uses several databases to ensure seamless operation, including *Firestore* medical records, appointments, and real-time updates. It also integrates with pharmacies for simple prescription fulfillment and medicine availability checks, and in an emergency, the platform connects with ambulance services for speedy SOS response. Finally, it uses the *Google Maps API* to make it easy for users to locate nearby doctors and pharmacies.

2.2 Product Functions

MediHelp ensures a smooth healthcare experience with key functions as given below:

- **Secure profiles:** Profiles are created and managed by patients and doctors using Firebase Authentication with role-based access control.
- **Medical Records Management:** Patients can upload and update medical records, while doctors review them before consultations.
- **Emergency SOS Alerts:** Patients can send real-time emergency alerts which will be sent to the nearest available doctors.
- **Medical & Pharmacy Queries:** Patients can check medicine availability and also order them. Patients can also resolve some basic doubts from the FAQs section.
- **Appointment Scheduling:** Patients can book appointments based on doctor availability.
- **Patients submit prescription requests for doctor review.** Pharmacies receive approved prescriptions.
- **Community & Support:** A community of doctors, patients and pharmacies, where people can resolve their doubts by directly chatting with doctors and pharmacists.
- **Effortless Management:** Doctors update schedules, patients manage medical history, and pharmacy staff process prescriptions.

2.3 User Classes and Characteristics

- **Users:** Patients who need medical consultations, appointments, medicine, prescriptions, or emergency help.
- **Doctors**: Medical professionals provide consultations, approve prescriptions, and manage schedules to be made available to the public.
- **Pharmacies** provide medicine-related services and availability updates.

2.4 Operating Environment

MediHelp operates as a cloud-based system accessible via web applications. It leverages:

Backend technologies:

- **Firebase Authentication:** For user authentication and role management
- **Firebase Firestore:** For storage of structured and unstructured data
- **Firebase Realtime Database:** For handling real-time features

Frontend:

- **React.js:** For building the user interface

Server:

- **Node.js with Express.js:** For backend server implementation

APIs & Services:

- **Google Maps API:** For location-based services
- **Twilio API:** For sending SMS alerts
- **Google Calendar API:** For calendar integration

Additional technologies:

- **Socket.io:** For real-time, bidirectional communication

2.5 Design and Implementation Constraints

- **Security & Compliance**: The system must ensure patient data security, follow HIPAA regulations, and implement role-based access control.
- **Real-time Processing**: Emergency SOS alerts require immediate transmission with minimal latency.
- **Scalability**: The system must handle a high volume of concurrent users, requiring optimized database management and caching solutions.

The functionality depends on external services such as Google Maps, WebRTC, and payment gateways. This requires us to consider the uptime of their APIs.

2.6 User Documentation

2.6.1 User Guide for Patients

- **Account Management**
  Creating, updating, and securing an account.
- **Booking Appointments**
  Searching for doctors, selecting time slots, and confirming appointments.
- **Accessing Medical Records**
  Viewing and updating personal health information, and allowing doctors access to previous medical records.
- **Prescription Requests**
  How to request medicines and view doctor-approved prescriptions.
- **Emergency SOS Feature**
  How to trigger an emergency alert and what to expect.

- **Community Forum Participation**
  How to post questions (anonymously or publicly).
- **FAQs & Troubleshooting**
  Answers to common patient-related queries.

2.6.2 User Guide for Doctors

- **Profile Setup & Management**
  Updating qualifications, availability, and consultation preferences.
- **Appointment Management**
  Accepting, rescheduling, or rejecting patient appointments.
- **Reviewing Patient Medical History**
  How to securely access and update records.
- **Prescription Approval Process**
  How to review and approve patient prescription requests.
- **Emergency Response**
  How to respond to SOS alerts and coordinate emergency care.
- **Community Forum Participation**
  How to engage with users and contribute to FAQs.

2.6.3 User Guide for Pharmacies

- **Inventory Management**
  Adding, updating, and removing medicines.
- **Processing Prescription Requests**
  Steps for handling doctor-approved prescriptions.
- **Medicine Availability Updates**
  How to keep the system updated with current stock.

2.6.4 Administrative Documentation

- **User Registration & Management**
  Approving/restricting user accounts.
- **System Security & Compliance**
  Handling authentication, data privacy, and security breaches.
- **Emergency Alert Monitoring**
  Tracking SOS alerts and ensuring timely responses.
- **Audit & Log Maintenance**
  Keeping records for security and compliance purposes.

2.7 Assumptions and Dependencies

- Patients and doctors must have *stable internet connectivity* to access real-time services.
- The system *depends on third-party services* (Google Maps API,) for key functionalities.
- User authentication relies on *Firebase Authentication*, and its availability affects login operations.

- *Medical data privacy laws and regulations* must be followed to ensure compliance with national and international healthcare standards.

3. External Interface Requirements

3.1 User Interfaces

The MediHelp system features distinct user interfaces tailored for each user role:

3.1.1. Patient Interface

- **Dashboard:**
  - Quick access to **appointments, medical records, pharmacy inventory, and emergency help**.
  - Health tips section (has to be updated regularly).
  - Noticeboard: for important announcements regarding common concerns/outbreaks.
- **Appointment Booking Page:**
  - Search for doctors by name, specialty, or location.
  - View doctor availability and select a suitable time slot.
  - Upload medical documents, or allow access to previously uploaded documents if needed.
- **Medical Records Page:**
  - View and update past consultations, prescriptions, and lab results.
- **Emergency Help Page:**
  - One-tap **SOS alert** with real-time location sharing.
  - Displays nearby hospitals and doctors who respond to the alert.
- **Community Forum:**
  - Securely post health-related questions (with an anonymous option).
  - Upvote/downvote responses from doctors, pharmacies, and other users.

3.1.2. Doctor Interface

- **Dashboard:**
  - Displays **pending appointments, patient details, and SOS alerts**.
- **Patient Management:**
  - View the **medical history of patients with active appointments**.
  - Secure chat interface for discussing patient concerns.
- **Appointment Calendar:**
  - Accept, reschedule, or cancel patient appointments.
- **Emergency Alerts:**
  - View **real-time SOS alerts** and **accept emergency cases** within a 5km radius.

3.1.3. Pharmacy Interface

- **Prescription Requests:**

    - View and process **doctor-approved** prescriptions.
- **Inventory Management:**
    - Update medicine stock availability.
    - Display available medicines to patients when prompted for a prescription.

3.1.4. Admin Interface

- **User Management:**
    - Approve or deactivate patient, doctor, and pharmacy accounts.
- **System Monitoring:**
    - Track system usage, security, and emergency response logs.
- **Audit & Compliance:**
    - Ensure privacy policies and legal regulations are maintained.

3.2 Hardware Interfaces

- Any browser for enabling web-based interactions between patients and doctors
- Servers- cloud-based interfacing with application logic, databases, and APIs

3.3 Software Interfaces

- **Firebase Authentication:** For user authentication and role management.
- **Firebase Firestore:** For storage of structured and unstructured data, including appointment scheduling, patient-doctor interactions, and medical records.
- **Firebase Realtime Database:** For handling real-time features such as instant emergency alerts and chat functionality.
- **Google Maps API:** For location-based services, including doctor and pharmacy searches.
- **Socket.io:** For real-time, bidirectional communication in the chat hub.
- **Twilio API:** For sending SMS alerts in the SOS feature.
- **Google Calendar API:** For calendar integration and appointment management.
- **React.js:** For building the frontend user interface.
- **Node.js with Express.js:** For backend server implementation.

3.4 Communications Interfaces

- **Notifications**: Firebase Cloud Messaging is implemented for appointment alerts and emergency notifications.
- **Real-Time Data Transfer**: Will use Firebase Realtime Database for instant alert handling in SOS situations.
- **External Integrations**: APIs to update pharmacy stocks, and coordinate with emergency services.

4. System Features

This section breaks down the main components of MediHelp, explaining what each feature does, its importance, and how it works step-by-step.

4.1 Prescription Generator

4.1.1 Description and Priority

What It Does:
This feature allows doctors and patients to enter prescription information digitally. Patients can upload a picture of their handwritten prescription, which is directly sent to the pharmacy inventory system without using OCR. Doctors can input prescriptions manually, and the data is forwarded to the pharmacy for processing.

Why It Matters:

- **Priority:** High
- **Benefits:** Streamlines the prescription process by ensuring direct and accurate submission to the pharmacy.
- **Potential Downsides:**Manual entry for doctors and reliance on image uploads may require additional verification.
- **Cost & Risk:** Ranges from medium to high, depending on network reliability and manual verification efforts.

4.1.2 How It Works (Stimulus/Response Sequences)

- **User Selection :**
  - **User Action:** The user selects their role (Doctor or Patient) from the main interface
  - **System Response:** The camera interface activates with guidelines for a clear capture.
- **Doctor Prescription Submission:**
  - **User Action:** The doctor inputs the prescription details manually, including the patient ID and prescription information.
  - **System Response:** The system processes the image using OCR and shows the extracted text for verificationThe system forwards the details directly to the pharmacy inventory system.
- **Patient Prescription Upload:**
  - **User Action:** The patient reviews the text and makes any necessary edits.The patient uploads a picture of their prescription and enters their patient ID and address.
  - **System Response:** The system stores the image and submits the request to the pharmacy inventory system for processing.
- **Verification:**
  - **User Action:** The pharmacy reviews the received prescription details and verifies the information before processing.
  - **System Response:** The system logs the transaction and updates

inventory accordingly.

4.1.3 Functional Requirements

- **REQ-PG-1:** Allow patients to initiate prescription capture from the main dashboard.
- **REQ-PG-2:** Activate the device camera and display real-time capture guidelines.
- **REQ-PG-3:** Apply OCR to convert the captured image into editable text.
- **REQ-PG-4:** Display the recognized text and allow inline editing.
- **REQ-PG-5:** Forward the validated digital prescription to update the pharmacy inventory system.
- **REQ-PG-6:** Implement error detection and prompt recapture if the OCR returns ambiguous or invalid data.

4.2 Pharmacy Inventory Display and Access

4.2.1 Description and Priority

What It Does:

This feature now involves creating a web panel for pharmacies to manage inventory and receive order requests. It connects the website's backend APIs to the pharmacy's existing inventory management system using their provided APIs and WebSockets.

Why It Matters:

- **Priority:** High
- **Benefits:** Increases order accuracy and enhances user satisfaction.
- **Potential Downsides:** Inaccurate inventory data could lead to service issues.
- **Cost & Risk:** Moderate.

4.2.2 How It Works (Stimulus/Response Sequences)

- **Inventory Display:**
  - **User Action:** Pharmacy staff updates inventory through their system.
  - **System Response:** WebSocket connection pushes updates to the MediHelp system.
- **Filtering and Search:**
  - **User Action:** The patient applies filters or enters a keyword.
  - **System Response:** The list updates dynamically based on the criteria.
- **Order Placement:**
  - **User Action:** A patient places an order on the MediHelp platform.
  - **System Response:** The system sends an API request to the pharmacy's web panel.
- **Order Confirmation and Update:**
  - **User Action:** Pharmacy staff receives the order notification on their web panel.
  - **System Response:** The system updates the order status and notifies

the patient.

4.2.3 Functional Requirements

- **REQ-PI-1:** Allow users to select their role as Doctor or Patient upon login.
- **REQ-PI-2:** Provide a structured form for doctors to input prescription details manually.
- **REQ-PI-3:** Allow patients to upload prescription images and enter necessary details like patient ID and address.
- **REQ-PI-4:** Forward all submitted data to the pharmacy inventory system without using OCR.
- **REQ-PI-5:**Ensure secure storage and retrieval of prescription data for verification purposes.

4.3 SOS Emergency

4.3.1 Description and Priority

What It Does:

This feature offers an emergency alert mechanism using Twilio's SMS API. When activated, it sends SMS notifications with a link to all registered medical centers within a 5 km radius and the designated Emergency contact number. The system captures the patient's GPS location and ensures that once one center accepts the alert via the link, other notifications are suppressed.

Why It Matters:

- **Priority:** Critical (High)
- **Benefits:** Enables immediate emergency response.
- **Potential Downsides:** Any delay in alerting could compromise patient safety.
- **Cost & Risk:** High, given the urgency and critical nature of emergency response.

4.3.2 How It Works (Stimulus/Response Sequences)

- **SOS Activation:**
  - **User Action:** The patient presses the visible “SOS” button.
  - **System Response:** The system instantly captures the patient’s current GPS coordinates.
- **Alert Dispatch:**
  - **User Action:** If required, the patient confirms the emergency alert.
  - **System Response:** It identifies all nearby medical centers within a 5 km radius and sends SMS notifications with a unique approval link using Twilio's API.
- **Response Logging:**
  - **User Action:** Medical centers receive the SMS and click the approval link, while the emergency contact simply receives the notification alongside the location to the person
  - **System Response:** The system logs the first center's acceptance and

stops further notifications, side by side also updating the same to the emergency contact.

- **Notification Update:**
  - **User Action:** The accepting medical center receives more detailed information.
  - **System Response:** It informs all centers that the emergency is being handled and logs responses for auditing.

4.3.3 Functional Requirements

- **REQ-SOS-1:** Present an easily accessible and clearly labeled SOS button.
- **REQ-SOS-2:** Capture the patient's GPS coordinates immediately upon SOS activation.
- **REQ-SOS-3:** Identify and list all medical centers within a 5 km radius.
- **REQ-SOS-4:** Use Twilio API to dispatch SMS notifications with unique approval links to all identified centers simultaneously.
- **REQ-SOS-5:** Implement a first-come-first-serve system for medical center responses.
- **REQ-SOS-6:** Provide more detailed emergency information to the accepting medical center and to the emergency contact information,
- **REQ-SOS-7:** Maintain an audit log of all SOS events and responses.

4.4 Appointment Scheduling

4.4.1 Description and Priority

What It Does:
This feature makes it easy for patients to request appointments with doctors through a user-friendly form. Once submitted, the appointment request is sent to the designated doctor, who can then accept or reject it. When accepted, the system automatically updates both the patient's and the doctor's calendars.

Why It Matters:

- **Priority:** High
- **Benefits:** It reduces administrative work and helps avoid scheduling conflicts.
- **Potential Downsides:** Missing or erroneous appointments can negatively affect patient care.
- **Cost & Risk:** Moderate.

4.4.2 How It Works (Stimulus/Response Sequences)

- **Form Submission:**
  - **User Action:** The patient fills out an appointment request form with personal details, the reason for the visit, and preferred time slots.
  - **System Response:** The system validates the information and displays a confirmation page.
- **Routing:**
  - **User Action:** The patient submits the request.

  - **System Response:** The request is routed to the selected doctor's interface.
- **Doctor Response:**
  - **User Action:** The doctor reviews the appointment request and chooses to either accept or decline it.
  - **System Response:** Upon acceptance, both calendars are automatically updated.
- **Conflict Resolution:**
  - **User Action:** If there's a scheduling conflict, the system prompts the user to reschedule.
  - **System Response:** It offers alternative time slots and clear error messages.

4.4.3 Functional Requirements

- **REQ-AS-1:** Present a detailed appointment request form that captures all necessary patient information and preferred time slots.
- **REQ-AS-2:** Validate form fields to ensure all required data is correctly provided.
- **REQ-AS-3:** Immediately route the appointment request to the designated doctor.
- **REQ-AS-4:** Allow the doctor to accept or reject the request with a single action.
- **REQ-AS-5:** Automatically update the calendars of both the patient and the doctor upon acceptance.
- **REQ-AS-6:** Notify patients via push notification and mail about appointment requests.

4.5 Secure Doc-Patient Interaction

4.5.1 Description and Priority

What It Does:

This feature offers a secure, real-time messaging interface for communication between patients and doctors. It supports text messages, images, and file attachments. Additionally, an appointment request button is embedded directly in the chat for quick scheduling.

Why It Matters:

- **Priority:** High
- **Benefits:** Enhances secure communication and improves patient care.
- **Potential Downsides:** Any data breach could lead to regulatory violations.
- **Cost & Risk:** High.

4.5.2 How It Works (Stimulus/Response Sequences)

- **Chat Initiation:**
  - **User Action:** The patient starts a chat session by selecting a doctor's name from their contact list.
  - **System Response:** The system establishes a secure, encrypted

connection and opens the chat interface.

- **Message Exchange:**
  - **User Action:** The patient sends a message or file attachment.
  - **System Response:** The message is encrypted and sent to the doctor, who then receives a notification.
- **Embedded Functions:**
  - **User Action:** The patient can use an embedded appointment request button within the chat.
  - **System Response:** This triggers the Appointment Scheduling module while keeping the chat session active.
- **Response Logging:**
  - **User Action:** The doctor replies and the conversation continues.
  - **System Response:** The system securely logs the conversation with timestamps and audit trails.

4.5.3 Functional Requirements

- **REQ-DP-1:** Provide an encrypted, real-time chat interface for secure communication between patients and doctors.
- **REQ-DP-2:** Support file attachments with restrictions on size and format as needed.
- **REQ-DP-3:** Embed appointment requests directly in the chat window.
- **REQ-DP-4:** Log all communications securely and maintain conversation histories for auditing purposes.
- **REQ-DP-5:** Generate real-time notifications for new messages for both the sender and the receiver.

4.6 Community Forum

4.6.1 Description and Priority

What It Does:

This feature now functions more like a review or comment section rather than a full-fledged community forum. It allows users to post comments, reviews, and questions about healthcare services, which are stored and updated in real-time using Firebase.

Why It Matters:

- **Priority**: Medium
- **Benefits**: Provides immediate feedback and user experiences for healthcare services.
- **Potential Downsides**: Requires moderation to prevent misuse or misinformation.
- **Cost & Risk**: Low to Moderate, given Firebase's scalability and ease of integration.

4.6.2 How It Works (Stimulus/Response Sequences)

- **Query Submission:**
  - A user selects "Add Comment" or "Write Review" on a healthcare service page.
  - System Response: The system displays a comment form.
- **Forum Posting:**
  - **User Action:** The user submits their comment or review.
  - **System Response:** Firebase immediately stores and displays the new content.
- **Interaction:**
  - **User Action:** Other users view the page.
  - **System Response:** Firebase pushes any new comments or reviews in real-time to all viewers.
- **Moderation:**
  - **User Action:** A user can flag a post for review.
  - **System Response:** Forum moderators are alerted and take appropriate action.

4.6.3 Functional Requirements

- **REQ-CF-1:** Integrate Firebase into the web application for real-time data storage and retrieval.
- **REQ-CF-2:** Allow users to submit comments and reviews on healthcare services.
- **REQ-CF-3:** Display comments and reviews in real-time without page refresh.
- **REQ-CF-4:** Implement basic moderation features to flag inappropriate content.

4.7 Calendar Management and Reminders

4.7.1 Description and Priority

What It Does:
This feature integrates a calendar to help users schedule and manage appointments, medication reminders, and other healthcare-related events. It also provides tools to add, edit, or delete calendar entries and synchronizes these events with appointment scheduling and prescription reminders.

Why It Matters:

- **Priority:** High
- **Benefits:** Improves organization and helps users keep track of appointments and medication schedules.
- **Potential Downsides:** Missing an appointment or medication dose can negatively affect health.
- **Cost & Risk:** Moderate.

4.7.2 How It Works (Stimulus/Response Sequences)

- **Calendar Access:**
  - **User Action:** The user selects the “Calendar” icon from the main interface.
  - **System Response:** The system displays the current month’s calendar with upcoming events highlighted.
- **Entry Management:**
  - **User Action:** The user adds or edits a calendar entry, such as an appointment or medication reminder.
  - **System Response:** An editable form appears with options to set the date, time, and notification preferences.
- **Synchronization:**
  - **User Action:** When an appointment is confirmed or a prescription is updated, the user sees the changes.
  - **System Response:** The calendar updates automatically with the new event and sends reminders as per the settings.

4.7.3 Functional Requirements

- **REQ-CM-1:** Provide a comprehensive calendar interface accessible from the main dashboard.
- **REQ-CM-2:** Allow users to add, edit, and remove calendar entries.
- **REQ-CM-3:** Synchronize calendar events with the appointment and prescription modules in real-time.
- **REQ-CM-4:** Support customizable reminders and notifications for upcoming events.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

The system shall be designed to meet stringent performance metrics under various operating conditions. For instance:

- **Concurrent Users:** The system must support multiple concurrent users without any degradation in performance.
- **Response Time:** At least 95% of user requests should receive a response within 2 seconds, ensuring a smooth and efficient user experience.
- **Emergency Response:** For critical situations, such as SOS emergency requests, the system must deliver the request to all medical centers within a 5km radius in under 1 second.

Rationale: These requirements ensure application responsiveness in routine and emergency scenarios. Specified timings guide load balancing, resource management, and real-time processing for high performance under peak load.

5.2 Safety Requirements:

While the provided specifications do not list extensive safety measures, the design inherently considers user and system safety through the following:

- **Emergency Handling:** The rapid delivery of SOS emergency requests is a key safety feature, designed to minimize risk and potential harm in urgent situations.
- **Robust Design:** The system's architecture will incorporate safeguards (e.g., redundancy and failover mechanisms) to mitigate risks of data loss or service interruptions that could lead to harmful outcomes.

Note: Additional safety measures may be defined as the system design evolves, particularly in alignment with healthcare safety regulations.

5.3 Security Requirements:

The system must address security and privacy comprehensively, including:

- **Data Encryption:** All communications between patients and doctors must be end-to-end encrypted to prevent unauthorized interception.
- **Secure Storage:** Patient records are to be stored securely with access strictly limited to authorized personnel.
- **Access Control:** Role-based access control (RBAC) must be implemented to ensure that only doctors, pharmacies, or other authorized users can access sensitive information.
- **Regulatory Compliance:** The system will adhere to GDPR and applicable Data Protection Laws, thereby ensuring that privacy and security regulations are met.

Rationale: These measures protect sensitive healthcare information, uphold patient privacy, and ensure that the system complies with industry standards and legal requirements.

5.4 Software Quality Attributes:

Several key quality attributes have been defined to support both customer satisfaction and ease of system maintenance:

- **Scalability:**
  - The system should scale horizontally to manage increasing traffic.
  - A cloud-based architecture will facilitate the dynamic allocation of resources in response to fluctuating user demands.
- **Availability:**
  - The platform is expected to achieve a 99.9% uptime.
  - The SOS emergency system must remain available 24/7, ensuring continuous operational readiness.
- **Usability:**
  - The interface will be simple and intuitive, catering to patients, doctors, and pharmacies.
  - The system will be mobile-friendly and accessible via web browsers, as well as Android and iOS applications.
  - The community forum will allow anonymous posting while enforcing

community guidelines to maintain a respectful environment.

- **Maintainability:**
    - The codebase should follow a modular architecture, simplifying future updates and bug fixes.
    - Automated backups will occur every 24 hours to safeguard against data loss.
    - Reported issues will be addressed within 72 hours under normal conditions and within 24 hours for critical problems.
- **Compliance & Portability:**
    - The system must adhere to industry standards such as HL7 for healthcare data exchange.
    - It should comply with GDPR and other Data Protection Laws to ensure patient privacy.
    - Compatibility is required across Windows, macOS, Android, and iOS, with support for major browsers like Chrome, Firefox, Edge, and Safari.

Rationale: These quality attributes are defined to be specific, measurable, and verifiable, thereby ensuring the system meets both technical and regulatory standards while providing a robust, user-friendly experience.

5.5 Business Rules:

The operational principles and policies that govern system behavior include:

- **Access Control:**

  Patient records and sensitive information are accessible only to users with the appropriate role, as defined by the RBAC policies.

- **Emergency Prioritization:**

  SOS emergency requests are given the highest priority and must adhere to the strict performance requirements outlined (i.e., delivery within 1 second to nearby medical centers).

- **Regulatory Compliance:**

  All system updates and modifications must be by established healthcare standards (e.g., HL7) and privacy regulations (e.g., GDPR).

- **Community Interaction:**

  In the community forum, users must follow the guidelines that promote respectful and anonymous posting, ensuring that interactions do not compromise user safety or privacy.

Rationale: These business rules establish the framework within which the system operates, ensuring that both technical functions and user interactions align with the overarching principles of security, safety, and regulatory compliance.

6. Other Requirements

- **Database Requirements:**

The system's database must support secure and scalable storage of all user data, including prescriptions, appointment records, and audit logs.

- **Internationalization and Localization:**

  The application shall support multiple languages and regional settings to accommodate diverse user populations.

- **Legal and Regulatory Requirements:**

  MediHelp must adhere to healthcare data protection regulations such as HIPAA and GDPR, as well as any local legal requirements.

- **Reusability:**

  Components of the system, such as the secure messaging module, should be designed for potential reuse in future projects.

Appendices

Appendix A: Glossary

- **MediHelp:** The healthcare platform described in this SRS.
- **SOS:** An emergency alert function designed to notify nearby medical centers.
- **COD (Cash on Delivery):** Payment option where the payment is collected upon delivery of the product.
- **API:** Application Programming Interface used to connect software components.
- **JWT:** JSON Web Token used for secure session management.

Appendix B: Analysis Models

- **UML Class Diagram**
  Defines key classes (e.g., User, Prescription, Appointment, Message) and their relationships.
- **UML Component Diagram**
  Shows system modules (Prescription Generator, Pharmacy Inventory, SOS, Appointment Scheduling, Secure Messaging, Community Forum, Calendar Management) with external interfaces (GPS, OCR libraries, healthcare databases).
- **UML Use-case Diagram**
  Identifies main actors (Patients, Doctors, Pharmacists, Administrators) and their interactions.
- **UML Activity Diagram**
  Outlines core workflows (e.g., SOS alert processing, appointment scheduling).
- **UML Sequential Diagram**
  Illustrates the sequence of interactions between components during specific operations

Appendix C: To Be Determined (TBD) List

- **TBD-1:** Confirm integration methods with local healthcare databases and external APIs.

- **TBD-2:** Determine specific performance benchmarks after prototype testing.
- **TBD-3:** Validate final user interface designs and accessibility compliance through user testing.
- **TBD-4**: How exactly are we thinking of triggering the SOS Notifications apart from the App via device trigger?
- **TBD-5**: Deployment of the Website, how is it to be done?

Summary

This detailed SRS for MediHelp provides an extensive roadmap for developing a robust healthcare platform tailored for underserved communities such as IIT Jodhpur students and rural residents. Every section—from the introduction and overall description to the granular functional and nonfunctional requirements—has been expanded to ensure clarity, traceability, and completeness. The document is designed to serve as the single source of truth for all project stakeholders, ensuring that development, testing, and deployment adhere strictly to defined requirements while remaining adaptable to future enhancements.