SOFTWARE REQUIREMENTS SPECIFICATION

for

SMART WELLNESS TRACKER SYSTEM

Group 3

Mahendraker Gaurav (210001036)

Gourav Ahlawat (210001019)

Princy Soundarva (210001068)

Kirtan Kushwaha (210001030)

Antim Jain (2402101002)

CS 416: Service Oriented Systems

Guided By:

Prof. Abhishek Shrivastava

February 07, 2025

Contents

1 Introduction

1.1 Purpose

Managing medical records and booking doctor's appointments can be challenging for patients. The Smart Wellness Tracker System aims to simplify healthcare management by providing a centralized, secure platform for patients to manage their health records, book appointments, and access other healthcare services. This system aims to improve patient engagement, reduce administrative burden on doctors, and enhance the overall healthcare experience.

1.2 Project Scope

The Smart Wellness Tracker System is a web-based platform designed to help patients manage their medical history and enable doctors to provide better care. The application automates appointment booking and integrates patient health management features to ensure a comprehensive healthcare experience. Key features include:

- Secure login via Google Auth.
- Doctor search by specialty, location, and availability.
- Appointment booking with secure payments.
- Medical record storage including prescriptions, lab results, and imaging scans.
- Teleconsultation via video calls.
- Personalized medication reminders.

1.3 Intended Audience and Reading Suggestions

1.3.1 Intended Audience

This document is intended for:

- Developers
  To implement system features.
- Product Owners & Business Analysts
  To align project goals and requirements.
- QA Engineers & Testers
  To verify system functionality.
- Clients & Stakeholders
  To understand the product scope and usability.
- Operations & Support Teams
  To manage deployment and maintenance.

1.3.2 Reading Suggestions

- Developers should focus on Functional & Non-Functional Requirements, System Features, and API Dependencies.
- Product Owners & Business Analysts should refer to Introduction, System Overview, and Specific Requirements.

- QA Engineers & Testers should check Functional Requirements & Error Handling.
- Clients & Stakeholders should review Purpose, Product Perspective, and Key Features.
- Operations & Support Teams should look into Database, API, and Security Requirements.

1.4 Document Conventions

This document follows these conventions:

- Priority Levels:
  - High (H): Critical features required for system operation
  - Medium (M): Important but not critical features
  - Low (L): Desirable features for future implementation
- Requirement Categories:

  MH : Must Have - Essential for system functionality
  SH : Should Have - Important but not critical
  CH : Could Have - Desired features if time/resources permit
- Requirement IDs follow the format: REQ-[Category]-[Number]
  - SEC: Security requirements
  - FUNC: Functional requirements
  - UI: User Interface requirements
  - PERF: Performance requirements
- Text Formatting:
  - **Bold text** indicates important terms or concepts
  - *Italic text* indicates user interface elements
  - `Monospace text` indicates code or technical terms

1.5 References

- React.js Documentation
- Node.js Documentation
- Express Documentation
- MongoDB Documentation
- Google Calendar API Documentation
- Google Maps API Documentation
- Stripe/PayPal API Documentation
- Twilio API Documentation
- Google Auth API Documentation
- Zoom/WebRTC API Documentation

2 Overall Description

2.1 Product Perspective

The Smart Wellness Tracker System is a web-based application built using React.js for the frontend and Node.js with Express for the backend. It integrates with MongoDB for data storage and various APIs such as Google Calendar, Google Maps, Stripe/PayPal, Twilio, Google Auth, and Zoom/WebRTC to provide a comprehensive healthcare management experience.

2.2 Product Functions

- Secure User Authentication using Google Auth API.
- Doctor Search based on specialty, location, and availability.
- Appointment Booking with schedule management.
- Secure Payment Processing using Stripe or PayPal.
- Health History Management including medical records, prescriptions, and lab results.
- Teleconsultation via video calls.
- Medication Reminders for timely doses.
- Digital Prescriptions generated and shared by doctors.

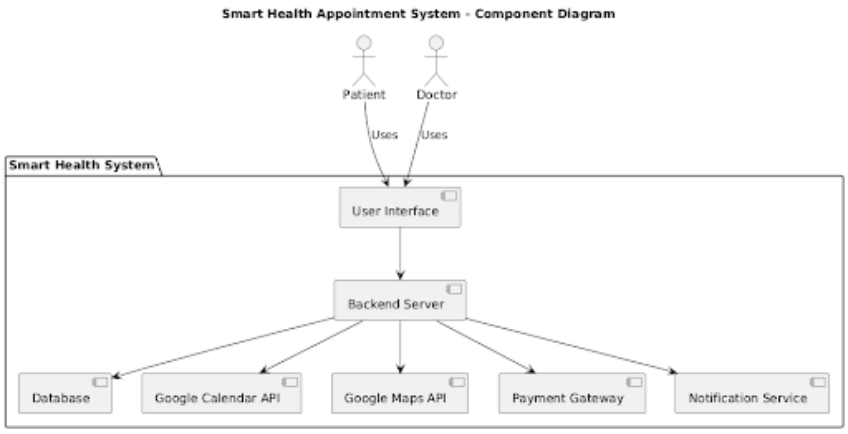

Figure 1: Flow Diagram

2.3 User Characteristics

- Patients: Need a secure way to manage health records, book appointments, and consult doctors.
- Doctors: Need a system to manage schedules, view patient health information, and conduct virtual consultations.

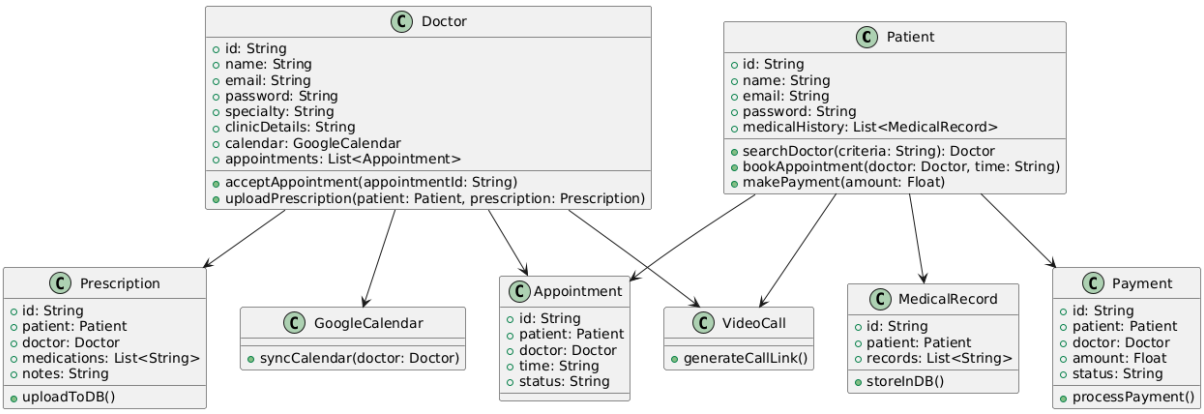

Figure 2: Class Diagram

2.4 Constraints

- Requires a stable internet connection.
- Dependent on third-party API availability.
- Compliance with standard data privacy and security protocols.

3 Specific Requirements

3.1 User Registration and Login

3.1.1 Description and Priority

Priority: High
Both patients and doctors must be able to register and log in securely using Google Auth.

3.1.2 Stimulus/Response Sequences

1. User clicks on the *Login with Google* button.
2. System redirects the user to the Google authentication page.
3. User enters their Google account credentials.
4. System verifies the credentials with Google.
5. Successful login redirects the user to their respective dashboard (patient or doctor).
6. If the user is new, the system prompts them to complete their profile information.

3.1.3 Functional Requirements

- REQ-1: Users must be able to register and log in using Google Auth.
- REQ-2: The system must securely store user profile information.
- REQ-3: The system should redirect users to the appropriate dashboard based on their role (patient or doctor).

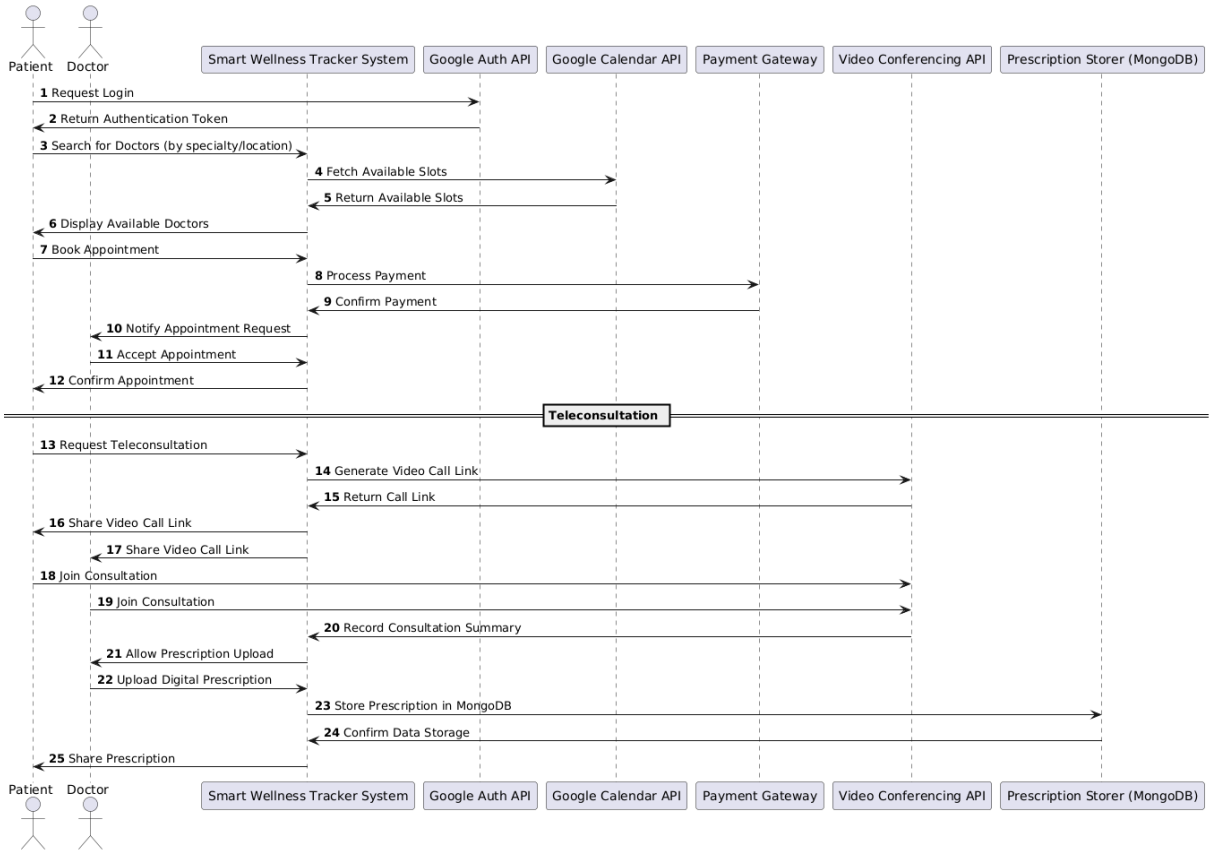

Figure 3: Sequence Diagram

3.2 Doctor Search

3.2.1 Description and Priority

Priority: High
Patients must be able to search for doctors based on specialty, location, and availability.

3.2.2 Stimulus/Response Sequences

1. User enters search criteria (e.g., specialty, location).
2. System queries the database for matching doctors.
3. System displays a list of doctors that match the search criteria, including their profiles and available time slots.

3.2.3 Functional Requirements

- REQ-4: Users must be able to search for doctors based on specialty, location, and availability.
- REQ-5: The system should display doctor profiles with relevant information (e.g., qualifications, experience, contact details).
- REQ-6: The system should integrate with Google Maps API to display doctor locations on a map.

3.3 Appointment Booking

3.3.1 Description and Priority

Priority: High
Patients must be able to book appointments with doctors and manage their schedules.

3.3.2 Stimulus/Response Sequences

1. User selects a doctor and an available time slot.
2. System verifies the availability of the selected time slot.
3. User confirms the appointment booking.
4. System sends a confirmation notification to both the patient and the doctor.
5. The appointment is added to the patient's and doctor's schedules.

3.3.3 Functional Requirements

- REQ-7: Users must be able to book appointments with doctors for specific time slots.
- REQ-8: The system should integrate with Google Calendar API to sync appointments with the doctor's and patient's calendars.
- REQ-9: The system should send confirmation and reminder notifications for appointments.

3.4 Payment System

3.4.1 Description and Priority

Priority: High
Secure payments for appointments via Stripe or PayPal API.

3.4.2 Stimulus/Response Sequences

1. User selects a payment method (Stripe/Paypal).
2. System redirects the user to the selected payment gateway.
3. User enters their payment details.
4. System processes the transaction securely.
5. Confirmation is sent upon successful payment.

3.4.3 Functional Requirements

- REQ-10: Users must be able to pay via Stripe or PayPal API.
- REQ-11: System should support multiple payment methods (credit/debit cards, wallets, etc.).
- REQ-12: The system must ensure secure transaction processing.

3.5 Health History Management

3.5.1 Description and Priority

Priority: High
Patients must be able to upload and securely store their medical records, prescriptions, lab results, and imaging scans.

3.5.2 Stimulus/Response Sequences

1. User navigates to the health history section.
2. User uploads medical documents (e.g., reports, prescriptions).
3. System securely stores the documents in the user's medical record vault.
4. User can view and download their stored documents.

3.5.3 Functional Requirements

- REQ-13: Users should be able to upload various types of medical documents.
- REQ-14: The system must ensure the secure storage and retrieval of medical records.
- REQ-15: Access to medical records should be restricted to the patient and authorized healthcare providers.

3.6 Teleconsultation

3.6.1 Description and Priority

Priority: Medium
Patients can consult doctors via video calls directly through the platform. Ideal for non-urgent consultations or follow-ups, reducing the need for in-person visits.

3.6.2 Stimulus/Response Sequences

- Patient and doctor schedule a teleconsultation appointment.
- At the scheduled time, both patient and doctor initiate the video call.
- System connects the patient and doctor via Zoom or WebRTC.
- The teleconsultation takes place.

3.6.3 Functional Requirements

- REQ-16: Users should be able to initiate and participate in video consultations.
- REQ-17: The system must provide a stable and secure video conferencing platform.
- REQ-18: The system should support features such as screen sharing and chat during the teleconsultation.

3.7 Medication Reminders

3.7.1 Description and Priority

Priority: Medium
Send personalized reminders to patients to take prescribed medications on time and notify patients when it's time to refill a prescription.

3.7.2 Stimulus/Response Sequences

- Doctor prescribes medication and sets up reminders.
- System sends personalized reminders to patients to take their medications on time.
- System notifies patients when it's time to refill a prescription.

3.7.3 Functional Requirements

- REQ-19: The system should allow doctors to set up medication reminders for patients.
- REQ-20: The system must send timely and personalized medication reminders to patients.
- REQ-21: The system should notify patients when it's time to refill their prescriptions.

3.8 Doctor Portal

3.8.1 Description and Priority

Priority: High
Doctors need a dedicated portal to manage their profiles, schedules, and patient information.

3.8.2 Stimulus/Response Sequences

- Doctor logs into the system.
- System redirects the doctor to their portal.
- Doctor can update their profile, manage their schedule, view patient health records (with consent), and generate prescriptions.

3.8.3 Functional Requirements

- REQ-22: Doctors must have a dedicated portal to manage their information.
- REQ-23: Doctors should be able to update their profile and manage their schedule.
- REQ-24: Doctors should be able to view patient health records (with consent) and generate prescriptions.

4 External Interface Requirements

4.1 User Interfaces

Web interface developed using React.js with responsive design for multiple devices, ensuring cross-platform compatibility and user-friendly navigation.

4.2 Hardware Interfaces

- Desktop and mobile computing devices
- Webcam and microphone support for teleconsultation

4.3 Software Interfaces

- Google Calendar API for scheduling
- Google Maps API for location services
- Stripe/PayPal API for payment processing
- Google Authentication API
- WebRTC/Zoom for video consultations
- MongoDB for data management

5 Non-Functional Requirements

- Performance: Responsive system with minimal latency
- Security: End-to-end data encryption and privacy protection
- Reliability: 99.9% system uptime
- Usability: Intuitive interface with minimal learning curve
- Scalability: Support for concurrent user sessions
- Compliance: HIPAA and data protection regulations

6 Additional Requirements

6.1 Database Requirements

MongoDB database configuration for:

- User profile management
- Appointment scheduling
- Medical record storage
- Transaction logging

6.2 API Integration Requirements

- Secure authentication mechanisms
- Real-time data synchronization
- Error handling and logging
- Rate limiting and access controls

6.3 Future Enhancements

- Machine learning-based health insights
- Multi-language support
- Advanced analytics dashboard
- Insurance claim integration

7 Other Non-Functional Requirements

- Performance: System should be responsive and provide timely feedback.
- Security: Ensures data privacy and protection.
- Reliability: High availability with minimal downtime.
- Usability: Easy navigation with an intuitive interface.

8 Other Requirements

8.1 Database Requirements

MongoDB will store user details, appointments, medical records, and payment information.

8.2 API Requirements

The system will integrate with external APIs for calendar syncing, location services, payments, authentication, and teleconsultation.

9 Appendix A: Glossary

Term/Acronym	Definition
SRS	Software Requirements Specification
EHR	Electronic Health Record - Digital version of a patient's medical history
Google Auth	Authentication service provided by Google for secure user login
MongoDB	NoSQL database used for storing structured healthcare data and user profiles
React.js	JavaScript library for building user interfaces
Node.js	JavaScript runtime environment for server-side development
WebRTC	Web Real-Time Communication - enables video consultations
API	Application Programming Interface - enables communication between software components
HIPAA	Health Insurance Portability and Accountability Act - U.S. healthcare data privacy law
Stripe/PayPal	Payment gateway services for secure online transactions
Teleconsultation	Remote medical consultation via video conferencing
2FA	Two-Factor Authentication - additional security layer for user authentication
UI/UX	User Interface/User Experience - aspects of system design and interaction
SSL	Secure Sockets Layer - protocol for secure data transmission
REST API	Representational State Transfer - architectural style for web services