Software Requirements Specification (SRS)

Project Name: Medify

Prepared By:
Ahmed Sameh 202202151
Salma Rami 202201759
Shahd Tarek 202202263
Mohamed Tarek 202201058
Seif Amr 202201510

Table of Contents

---

1. Introduction

1.1 Purpose

In preparation for the other tasks later on, this document outlines the functional and non-functional requirements for *Medify*. More importantly, it describes the system's functionalities, its parts, and the limitations that can assist developers, testers, and other stakeholders in the development of the Medify hospital management system.

1.2 Scope

Medify collates various aspects of hospital management including the patient's past information, the existance of such a doctor, pharmacy availability and the making of appointments into a unified interface online. This integration makes for easier processes and interactions between the patients and their doctors as well as the staff in charge. The system will offer functionalities which shall include:

- **Self-Patient Registration:** Giving patients the means to register in and update their details on their profile.
- **Booking and Tracking of Appointments**: Enabling users to schedule, reschedule, and keep track of their appointments for doctors' visits.
- **Doctor Allocation and View Profile**: Enabling the appointment of doctor to a patient and allowing for profile views in order to enhance office coordination.
- –**Patient / Staff Interaction Portal**: Giving provisions for secure interaction between patients and staff of the hospital whereby they are able to send messages and other information.

1.3 Definitions, Acronyms, and Abbreviations

- ***API***: Application Programming Interface.
- ***UAT***: User Acceptance Testing.
- ***SDS***: Software Design Specification.
- ***UI***: User Interface.
- ***HTTPS***: Hypertext Transfer Protocol Secured, standard for safe distribution of information.
- ***DBMS***: Data Base Management System, specifically MySQL in this project.
- ***AWS***: Amazon Web Services, who provide the operating platform of the system.

1.4 References

- *SRS: Software Requirements Specification.* It has to do with how the system is meant to function or Medify's expected functionality.
- *API Documentation:* This provides an explanation of the RESTful API endpoints used to communicate the frontend and backend
- *Design Guidelines Document*: Provides a means in which standardized UI/UX shall be developed and maintained across the platform.
- *Testing Guidelines*: This describes unit testing, integration testing, UAT and performance testing protocols.

---

2. System Overview

2.1 Product Perspective

This system is [part of a larger system / a standalone system], and it has [other systems, external services, or users] who it interfaces with. It comprises of:

- **Frontend**: the user's means of interacting with the system
- **Backend**: the server's mechanisms which comprise business logic and processing.
- **Database**: this is a repository for all the persistence data.

2.2 Product Functions

The basic functions of the system include:

- **Function 1**: Patient Registration and Profile Management: Patients can create and manage profiles which include health history that assists the doctors in giving personalized care to patients.
- **Function 2**: Appointment Scheduling: This allows seamless scheduling by enabling patients to see available doctors with whom they can book an appointment. It has features like reminders, conflict management and also rescheduling options.
- **Function 3**: Patient-to-Doctor Assignment and Profile View: Provides for easy patient-to-physician matching, viewing of Professional Profile supporting doctors in the patients records retrieval and reliability for recording what happened from each visit.
- **Function 4**: Communication Portal: Secure messaging system with a seamless integrated communication between the patients, doctors and staff ensuring that care is not interrupted whilst at the same time all patient questions are well attended to.

2.3 User Classes and Characteristics

The following different types of classes of users will interface the system:

- **Admin Users:** Users who have unrestricted access to all the system features such as user management, and system configurations as well as monitoring the operations of all modules. These are users who are hospital administrators or IT personnel in charge of security and day – to – day running of the system.

- **Doctors**: Able to view the patient files, appointments and update the patients file after the consultation. Doctor can enter in the systems information regarding the provision of care and record but settings are restricted.

- **Patients**: Patients can register and manage their profile, schedule and make an appointment, and message the hospital's staff through the communication portal. Patients can have their own data but are not allowed to see other patients data or change the configuration of the systems.

- **Hospital Staff:**The hospital employees consist of personal aides like receptionists, medical assistants, people who possess, integrate, and exchange information, maintain schedules and organize patients, as well as doctors and nurses. Staff has access to scheduling and organizational functions, but not to clinical documents.

2.4 Operating Environment

The above-mentioned Medify system will be deployed in the following environments:

- **Client Side:** Works on most recent web browsers such as chrome, Firefox, Safari, Microsoft Edge, and others, making it up to date with the newest browser version for web and mobile users.
- **Server Side**: It is built on a cloud server working on Linux or any OS compatible for web development using Python with Flask as the backend language and API development.
- **Database**: Employs MySQL as the database to be utilized for the preservation and management of patient records, booking, doctor's profiles ,and even the configuration of the system.

3. Functional Requirements

1. Patient Registration
2. Appointment Scheduling
3. Doctor Profile Access
4. Patient Visit History Access
5. Reports and Analytics Generation
6. Communication Portal
7. Patient Profile Update
8. Admin User Management
9. Appointment Reminders
10. Appointment Cancellation

3.1 Use Case Diagrams / User Stories

- **Use Case 1**: [Patient Registration]
    - Description: It is aimed at the inclusion of new patients who are able to provide us with their details and other medical history.
    - Actors: A set of people who register, i.e. patients.
    - Preconditions: The patient is required to obtain a registration form and to accept the terms and conditions.
    - Postconditions: A new patient account is established within the system. Steps:
        1) Patient navigates to the registration page.
        2) Patient fills in required fields (name, contact info, medical history).
        3) Patient submits the form.
        4) System verifies data and creates an account.
        5) Confirmation message is displayed.

- **use case 2**: [Appointment Scheduling]
    - **Description**: Enables a patient to make an appointment with the doctor of his/her choice according to availability.
    - **Actors**: Patient, Doctor
    - **Preconditions**: It is required that the patient is registered on the site and that the patient logs in.

- **Postconditions**: The appointment has been allocated into the system database.
- Steps:

- **Steps**:

  1) Patient logs in and selects appointment scheduling.
  2) Patient chooses a doctor and available time slot.
  3) Patient confirms the appointment.
  4) System saves the appointment and sends a confirmation notification.

Repeat for each use case or user story.

3.2 Feature Requirements

Feature 1: [Patient Registration]

- **Description**: Enable patients to register and provide relevant information to create a profile
- **Inputs**:Patient basic details including name, contacts and medical records.
- **Outputs**: There should be a message acknowledging that the registration was successful.
- **Error Handling**:The user gets an error message notifying them of the need to fill in the required fields.

- **Feature 2: [Appointment Scheduling]**

- **Description**: Gives the patients options to book for their appointments
- **Inputs**: The patient selects the doctor they require and the time they want.
- **Outputs**: The system sends a notification indicating the patient's appointment.
- **Error Handling**:The system displays an error message notifying the patient that the time however is already booked.

---

4. Non-Functional Requirements

4.1 Performance Requirements

- The system shall accept user requests and respond to them within 2 seconds.
- It has to manage up to 500 users concurrently with persistence of quality.

4.2 Security Requirements

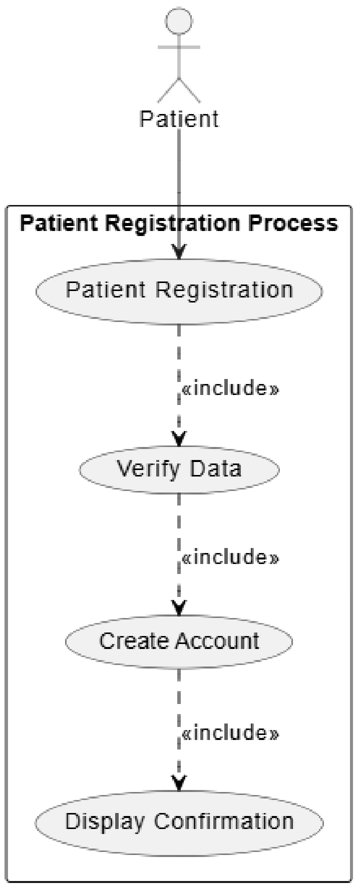
- User data has to be protected in the transmission by the HTTPS protocols
- Specific system functions should only be available to specific users who have been authorized and given role – based access permissions to these functions.

---

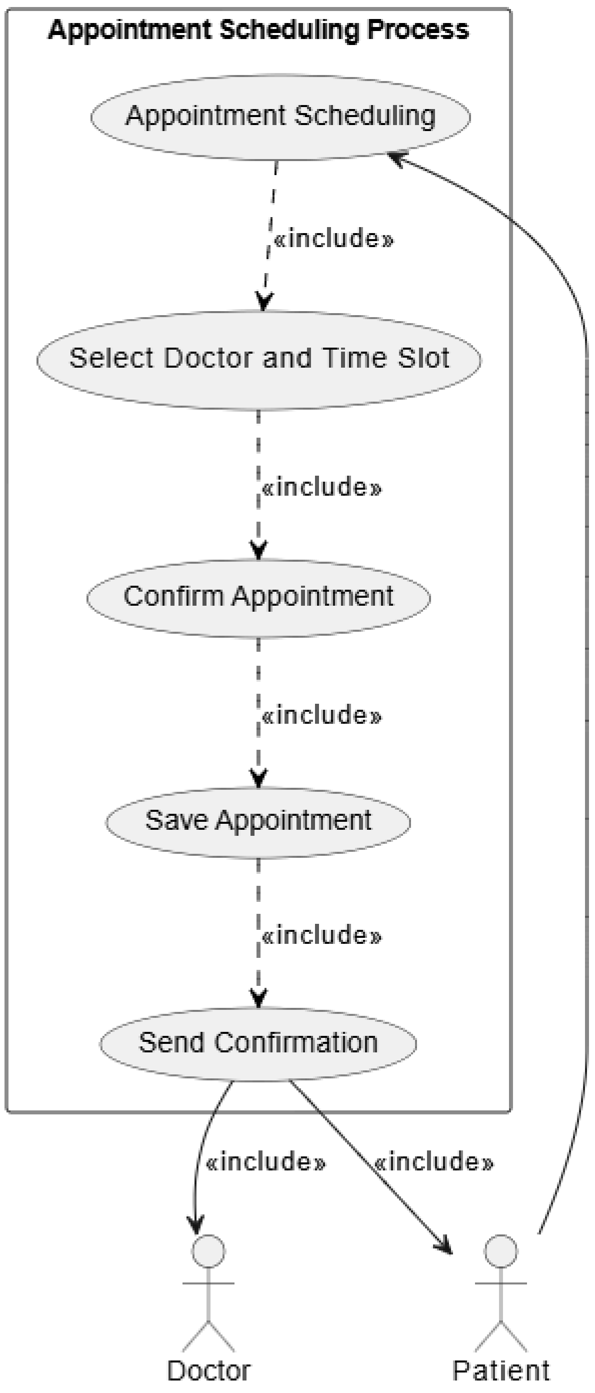
5. System Models

5.1 Use Case Diagrams

use case 1



use case 2


Appointment Scheduling Process
Appointment Scheduling
«include»
Select Doctor and Time Slot
«include»
Confirm Appointment
«include»
Save Appointment
«include»
Send Confirmation
«include»
«include»
Doctor
Patient

6. External Interface Requirements

6.1 User Interfaces

- The system should have an easy to use UI with at least the following major components:

    1. **Home Screen**: This screen shows what user worked on mostly, the last logged appointments and relevant alerts.

    2. **Dashboard**: The user is able to undertake certain activities like making new appointments, managing patient records and reports.

    3. **Navigation Bar**: This provides shortcuts that may include appointments, management of profiles, the portal for communication, etc.

    4. **Notifications**: It shows notifications of what appointments are still due, new updates of the system or some messages from the health professionals.

6.2 API Interfaces

- The system will offer API for the following functionality:

  - **Data Retrieval**: APIs that will draw information about the patient, the patient's appointments and prescriptions when they were made.

  - **User Management:Such as Registration of the user, signing in or out and giving the particular user appropriate view based on their profile.**

  - **Appointment Management**:These APIs are used to set, delete, and retrieve appointments..

- This will be required under the REST architecture style, API graphical education and sample using Postman HTTP tool. Built APIs will follow Uniform resource locators that will conform to the International communication protocol standard with four basic methods GET/POST/PUT/DELETE.

6.3 Hardware Interfaces

- The system will interface with other devices e.g. External.

  - **Barcode Scanners**:For scanning of the ID cards of patients or any prescription.

  - **Printers**: Is for printing of the appointment confirmation documents, the prescription or even the report.

---

7. Other Requirements

7.1 Legal and Regulatory Requirements

The system is required to observe appropriate legal and regulatory measures in a way that does not violate the privacy and confidentiality of users' data.

- The user data shall always be protected as per the GDPR laws

7.2 Documentation Requirements

- Developer guides and user guides shall be made available.

  -User Manuals: Basic instructions on the usage of the system including registration, appointment scheduling and profile management.

  - API Documentation: comprehensive overview of the APIs available within the system including END points, types of request/response and how to authenticate.

7.3 Data Backup Requirements

- This section defines the backup policies that shall be employed for the system.

  - Backup Frequency: The system implements a gradual increase in significant data over a period of one day.

  - Backup Retention: Backups to be kept for a period of 30 days.

- Backup Security: provisions for encryption and safeguarding data against unauthorized accessibility.

---

8. Conclusion

The aim of this Software Requirements Specification (SRS) document is to detail the functional and nonfunctional as well as the external interface requirements for the Healthcare Management System. Given the requirements as details, the development team can build a comprehensive system that is secure, reliable, and easy to use for patients and other related users.