Software Requirements Specification

for

Online Medical Consultation System

Version 1.0 approved

Prepared by :-

1. **Ashwani Kumar Kamal**
2. **Archit Mangrulkar**
3. **Shiladitya De**

Team: ASMR

Group: 48

15th March, 2022

Table of Contents

Revision History

Name	Date	Reason For Changes	Version

1 Introduction

Scheduling has always been a famous problem in real life applications. Scheduling of events and managing them is important in many areas like transport management, airline management, train ticket booking etc. One such problem in similar domain is medical consultation system. Management of appointments bookings, doctor availability, patient feedback needs to be done efficiently using an interface.

1.1 Purpose

The purpose of this document is to build an online system to manage medical consultation and patient appointments to ease the management.

1.2 Document Conventions

This document uses the following conventions.

Table 1

Abbreviation	*Meaning*
DB	*Database*
DDB	*Distributed Database*
ER	*Entity Relationship*

1.3 Intended Audience and Reading Suggestions

This project is a prototype for the medical management system and it is restricted within the medical domain. This project is useful for the medical management team and as well as to the patients.

1.4 Product Scope

The purpose of the online medical consultation system is to ease consultation management and to create a convenient and easy-to-use application for patients, trying to book medical appointments. The system is based on a relational database with its management and reservation functions. We will have a database server supporting hundreds of major cities around the world as

well as thousands of doctors of various hospitals. Above all, we hope to provide a comfortable user experience along with the best resources and tools available.
More specifically, this system is designed to allow a doctor to manage and communicate with the interface and create their profile. A patient will have options to select their area pin code which will then fetch the list of all active doctors in that area. Patient will have the interface to select one doctor and book their appointment with the doctor.

1.5 References

The basic outline of a SRS document has been given to us. We also took references from the following websites for better understanding.

1. *https://medium.com/@enisinanaj/writing-a-software-requirements-specification-document-97d622805aef*

2. *https://krazytech.com/projects/sample-software-requirements-specificationsrs-report-airline-database*

3. *https://www.utdallas.edu/ ∼ chung/RE/Presentations07S/Team_1_Doc/Documents/SRS4.0.doc*

2 Overall Description

2.1 Product Perspective

This application is useful for the hospitals for automating its daily tasks and freeing up its employees from the mundane and repetitive task as well as eliminating human error. This in turn increases productivity and and ease in keeping track of scheduled events and availability.

2.2 Product Functions

The following features are proposed for this application-

- *Dashboard for patients to check for available doctors pertaining to some category (cardiac, orthopaedic, optician etc)*

- *Option to choose doctors from a user input pin code*

- *Dashboard for managing profile at doctor's end and updating available slots*

- *Option to accept or reject a patients request at doctor's dashboard*

- *Feedback portal for patients to increase interaction and transparency*

- *Confirmation mail will be sent after doctor's approval*

2.3 User Classes and Characteristics

There will be 3 type of login roles-

- *Admin: Responsible for managing doctor database of hospitals, adding / removing doctors as per the update*
- *Doctor: Responsible for managing their profile dashboard and reviewing patient requests and feedback comments*
- *Patient: Responsible for making appointment / consultation requests and giving their feedback*

2.4 Operating Environment

Application will be deployed as a web app

- *Operating system: Windows / Linux / MacOS*
- *Deployed using heroku / netlify / local server*
- *Database can be SQLite (currently) or hosted PostgreSQL DB*
- *Frontend: ReactJS, SASS, CSS, Webpack (for building)*
- *Frontend package manager: Yarn*
- *Backend: Django, DjangoRestFramework*
- *Backend package manager: Poetry*
- *Emailing using Sendgrid python package*
- *Login feature will be accomplished using JWT (JSON Web Tokens)*

2.5 Design and Implementation Constraints

Design is decoupled frontend (in React) and backend (in Django). Implementation constraint would be the robustness / performance on DB side.

2.6 User Documentation

A README documentation (Markdown) will be included for delivering the instructions for using and deploying the app. A quick reference at patient's side documenting all the features and usability will be very useful.

2.7 Assumptions and Dependencies

- *The user is supposed to know the basics of computer handling.*
- *The user is supposed to know and operate in English language as the website is implemented in English language*
- *The doctor can change his/her working hours from his tab without telling it to the admin.*
- *Once the appointment is finalised and approved by the doctor, a mail is sent to the patient with all the details of the appointment. After the finalisation of the appointment the same cannot be cancelled.*
- *The feedback of the respective patients will be visible to the doctors.*

3 External Interface Requirements

3.1 User Interfaces

- *Admin Interface: Inherited from django-admin portal, will manage the doctor database directly. Admin is responsible for adding a new doctor's portal as well as removing also.*
- *Doctor Interface: Creating and managing their credentials / available hours, reviewing patient requests and feedback.*
- *Patient Interface: Dashboard for showing available doctors in the entered pin code and making appointment requests to the selected doctor. Notification on patient side in case a request is rejected via mail.*

3.2 Hardware Interfaces

Not Applicable

3.3 Software Interfaces

- *Pin code Search: Filter the list of doctors on the basis of pin code of hospital*
- *Doctor Login: Role based login which will redirect to doctor interface*
  - *Profile details*
  - *Available hours in the week*
  - *Pending requests by patients*
  - *Completed requests, feedback and comments*
- *Patient Login: Role based login which will redirect to patient interface*
  - *Profile details*

  - *Pin code search interface for filtering out doctors*
  - *Option to send appointment request for a selected doctor*
  - *Feedback portal*

- *Contact List : Contacts list of doctors will be displayed in a drop down (Names will be displayed which on click will show the doctor's contact details).*
- *Admin Login (Superuser Mode) : The Admin through this portal can access and control the doctor's portal and change according to the need.*

3.4 Communications Interfaces

1. *Login will be managed by password encryption in the database*
2. *Patients by logging into their portal can make requests to specific doctor*
3. *The doctor can then view the made request and accept / reject accordingly*
4. *The patient will get the notification about the status of request (via mail) and they can then give feedback*
5. *The software is implemented using the HTTP protocol*

4 System Features

4.1 Login Page

4.1.1 Description and Priority

The OCMS handles logins for Admin, Doctor and Patient such that they can operate without any interference from the each other. Passwords stored in the database will be encrypted using a hashing algorithm. Inherited from django-admin, the Admin Login allows the user to access the database and modify it directly. Doctor and Patient login are handled through role based login and redirect to respective interfaces.

4.1.2 Stimulus/Response Sequences

- *Admin will have the read and write access to doctor database*
- *Doctor and patient login will redirect to respective interfaces*

4.1.3 Functional Requirements

Whole login will be extensively managed by Django and its dependencies hence Python and Django will be required.

4.2 Pin code search

The list of doctors will be filtered according to the latest database on the basis of entered pin code by patient in their portal.
This will be managed by an SQL query.

4.3 Sending and receiving appointment requests

4.3.1 Desription and Priority

This feature will be used by the patients and doctors for booking / approving / cancelling appointments.

4.3.2 Stimulus/Response Sequences

- *Patient will be able to book an appointment by first sending a request to the concerned doctor.*
- *The sent request will be reflected on the concerned doctor's portal.*
- *Doctor will have the option to accept / reject that request with appropriate message.*

4.3.3 Functional Requirements

This needs a database for managing the data of requests. So a database management system is needed which will be implemented with Django.
Required: PostgreSQL database

4.4 Feedback

Patient will have the option to send their feedback and comments after successful completion of an appointment. This will use the database and a SQL query will be used to fetch the corresponding feedback of the specific doctor.

5 Other Nonfunctional Requirements

5.1 Performance Requirements

- *A High speed internet is required for proper loading of the website.*
- *Any browser with a recent version is fine.*

5.2 Safety Requirements

- *The user must remember their login credentials. Feature of "Forget Password" is not implemented.*

5.3 Security Requirement

- *The information of Patients and Doctors are stored in their portal secured by password(Defined by the user).*
- *There are separate logins for Doctor and Patients. And only the admin can add a new doctor or remove an existing one.*

5.4 Software Quality Attributes

1. *RELIABILITY: The software can be used by multiple users (Patients and Doctors) at the same time.*
2. *MAINTENANCE: The admins can see the doctors portals and change them according to the need.*
3. *PORTABILITY: The software can be accessed on any device be it PC or Tablet or a mobile.*
4. *AVAILABILTY: The OMCS website would be available throughout the day.*
5. *ROBUSTNESS: The website will be throughly tested before the deployment to rule out any possibility of a bug.*

5.5 Business Rules

The OMCS software(website) can be used by the doctors and patients and managed by the admins (like Medical Store etc.) to which it is sold.

6 Other Requirements

Database:

Database is very much required for purposes like storing data of Patients, Doctors. Hence, it stores a lot of data in it, so it is suggested that the platform in which it is hosted has sufficient memory reserve.

Appendix A: Glossary

- *SQL : Structured Query Language*
- *CSS: Cascading Style Sheets*
- *SASS: Syntactically Awesome Style Sheets*
- *JS: JavaScript*
- *DB: Database*

Appendix B: Analysis Models

Use Case Diagram :

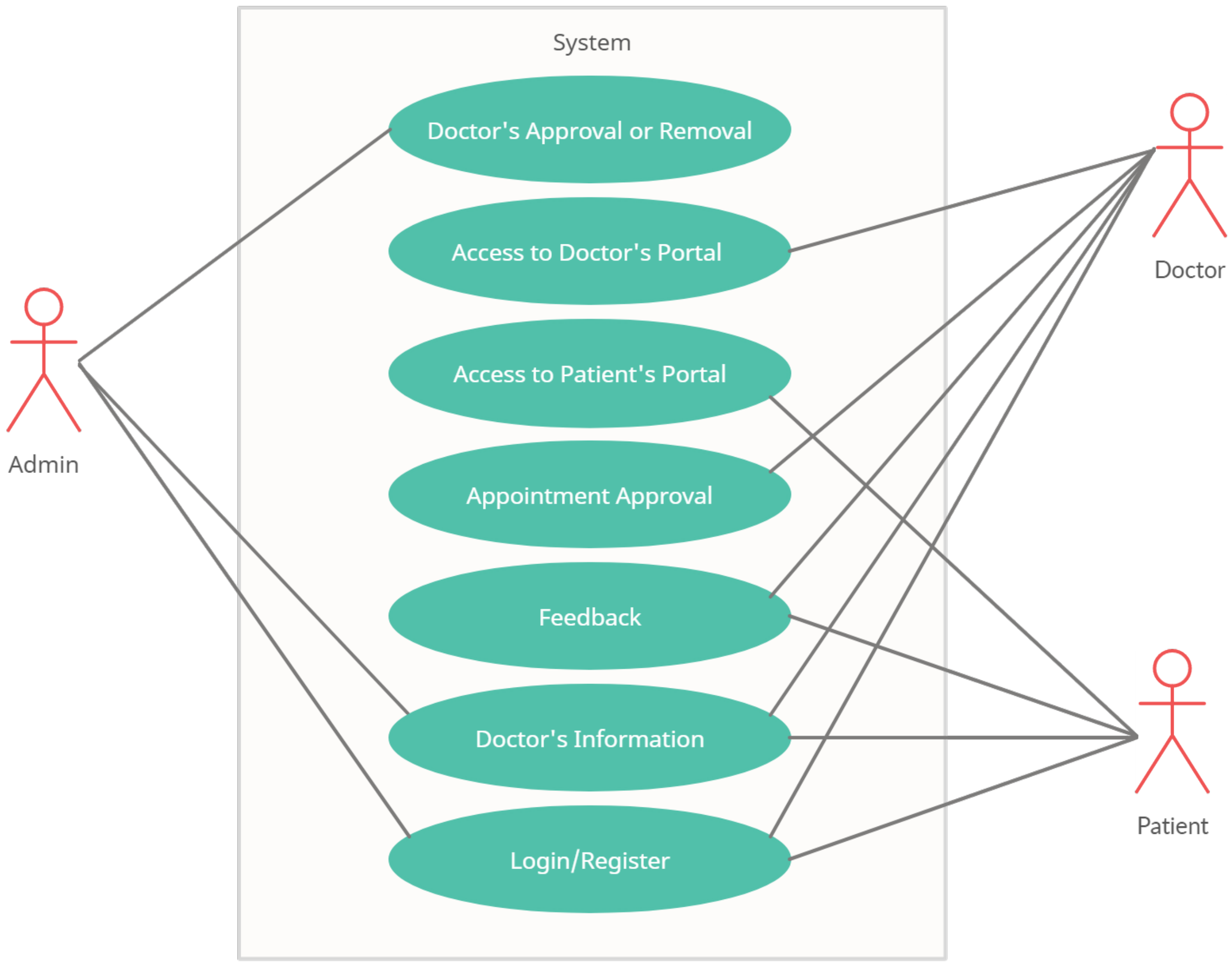

Figure 1: Use Case Diagram for OMCS

Class Diagram :

The class diagram is the one of the most important part of any Software Development project. It shows the basic layout of classes and their interactions with other classes. It also includes the important functions and methods associated with that class. It can be used to translate the models into code.
There are 6 classes in the project-

- *Admin*
- *Doctor*
- *Patient*
- *Appointments*
- *Working hours*
- *Feedback*

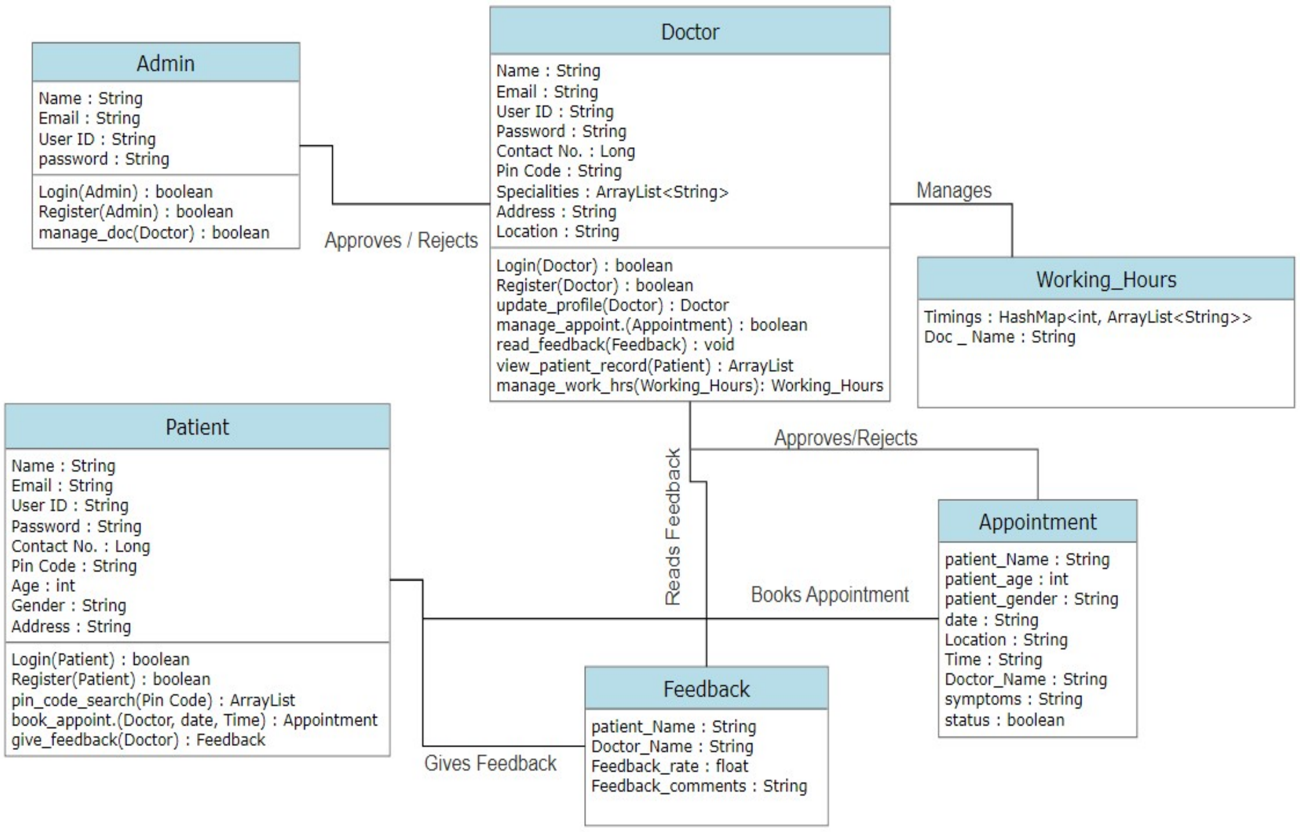

Figure 2: Class Diagram for OMCS

Appendix C:
To Be Determined List