Software Requirements Specification

for

Hospital Appointment Booking Application

Version 1.0 approved

Prepared by

Roll No.	Gr. No.	Name
232036	21910632	Mayank Sahu
232032	21910416	Siddhesh Kothadi
232045	21910716	Atharva Parikh
232030	21910598	Sidhant Khamankar

<Group 2>

22-04-2021

Table of Contents

1. Introduction

1.1 Purpose

The purpose of this document is to build an online system for a multispeciality hospital to automate their appointment booking process.

1.2 Document Conventions

This document uses the following conventions:

DB	Database
MERN	MongoDB, ExpressJS, ReactJS, NodeJS
JS	Javascript

1.3 Intended Audience and Reading Suggestions

This project is a prototype of a hospital appointment booking system and is restricted within the college campus. It has been implemented under the guidance of college professors. This project is useful for multispeciality hospitals and their patients. This document is intended for developers, users, testers and documentation writers.

1.4 Product Scope

The purpose of the hospital appointment booking system is to ease the appointment booking process and to create a convenient and easy-to-use web application for patients,who wish to see a doctor of that particular hospital. The system is based on a MERN stack. We can have the database server supporting data of hundreds of patients and numerous doctors of that hospital. Above all, we hope to provide a comfortable user experience along with the best pricing available.

1.5 References

Frontend

[React Hooks]
[Bootstrap for React]
[OAuth2 - Sign in with Google]
[Google Calendar API documentation]
[Calendar Sample in JS]

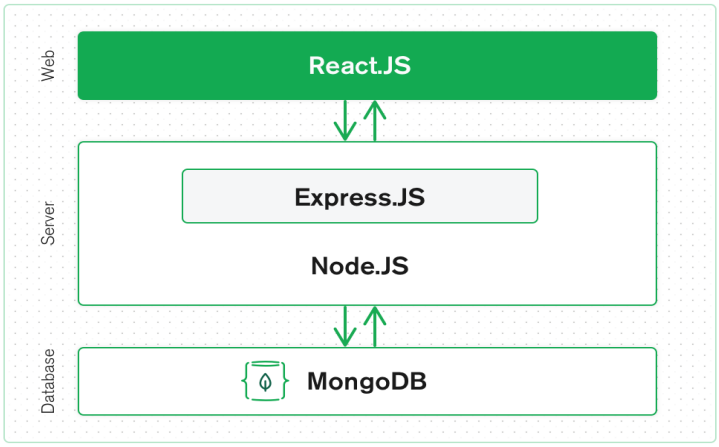
Backend

[Mongoose for DB connection and operations]
[JWT based authentication]
[Expressjs]
[.env file]
[Payment Gateway]

2. Overall Description

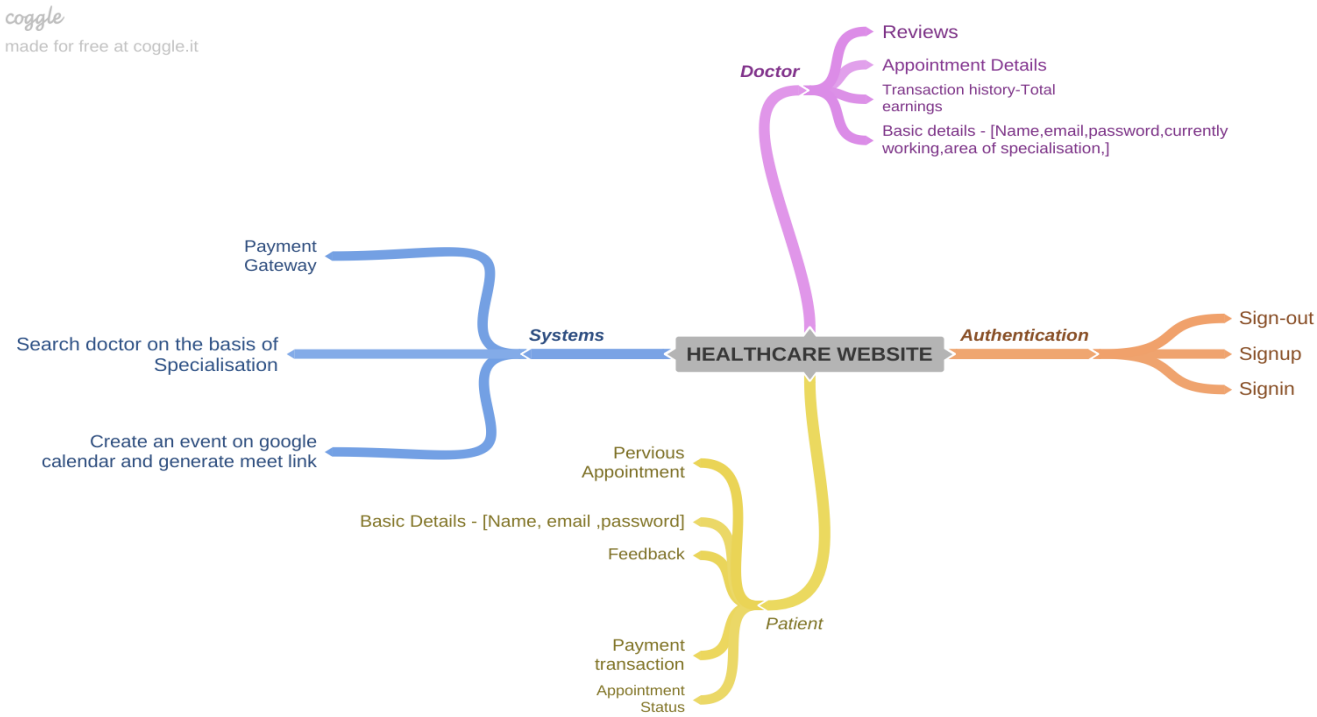
2.1 Product Perspective

This is a new, self-contained product with the following subsystem interconnections.The DB stores the information of doctors, patients and all the related information.



2.2 Product Functions

Highlight of features of the system are shown in the flowchart below.



2.3 User Classes and Characteristics

This system will be used by doctors and patients and managed by a system administrator. Due to a simplistic and minimal User Interface elderly patients and doctors who aren't much tech savvy will also be able to use this system without much effort. This has been the main aim while designing the system and will continually remain a first priority during following versions.

With the help of this system patients should be able to:

- Make their own medical profile.
- Book an appointment with a doctor.
- Receive meet link directly in google calendar.
- Perform direct secure payments.
- See his previous appointments.
- Provide feedback to doctors about the service they recieved.

With the help of this system doctors should be able to:

- see appointments and accept or reject them directly using google calendar.
- see the transaction history and total earnings.
- see reviews given by their patients.
- view the medical history of their patients.

The job of system administrators is to work closely with the hospital and add only the verified doctors into the database.

2.4 Operating Environment

This is a Web based application which requires a modern Browser to handle Client-Server requests efficiently.
Following Browsers are technically compatible with this Web App:
Firefox,
Chrome,
Safari,
Opera and
Internet Explorer 8 and 9

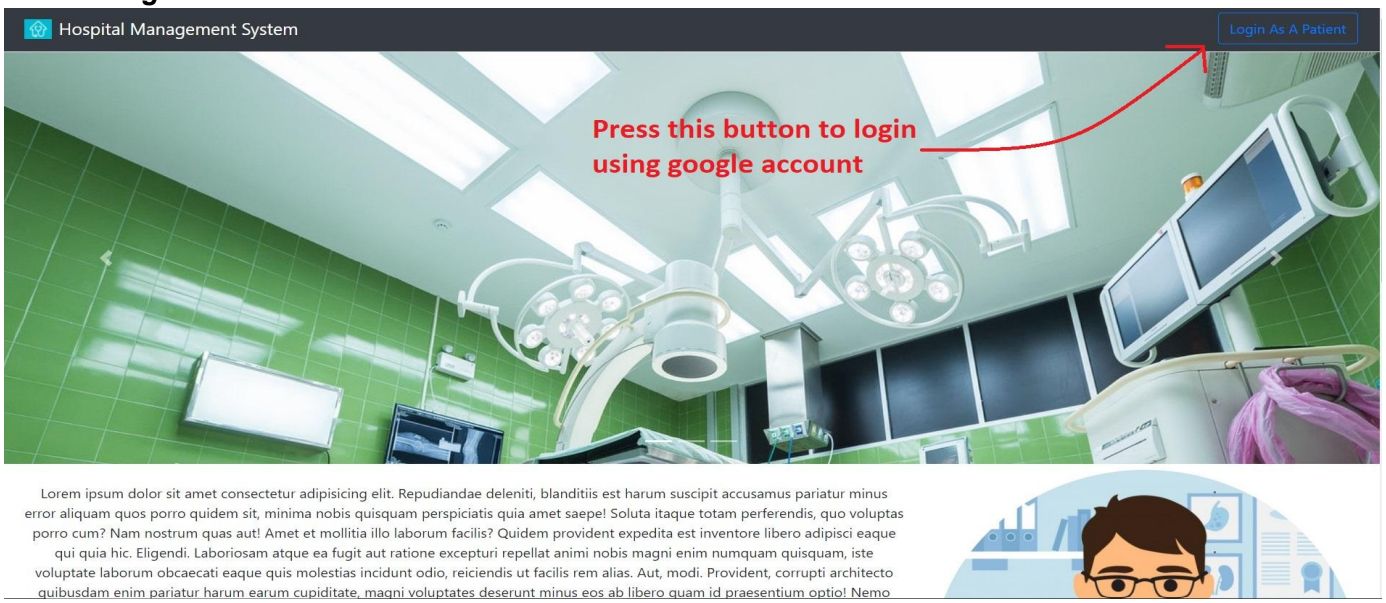
2.5 Design and Implementation Constraints

Regulatory issues: Data Breaches, False Claims and fraudulent behaviour, Anti-Kickback Statute, Antitrust Issues, Hidden Fees.
Hardware : Usage of CPU, RAM, and storage space can vary significantly based on user behavior. These hardware recommendations are based on traditional deployments and may grow or shrink depending on how active the users are.
Specific technology: MERN stack.
Language: Javascript.
Programming Standards: Coding Standards and Guidelines.

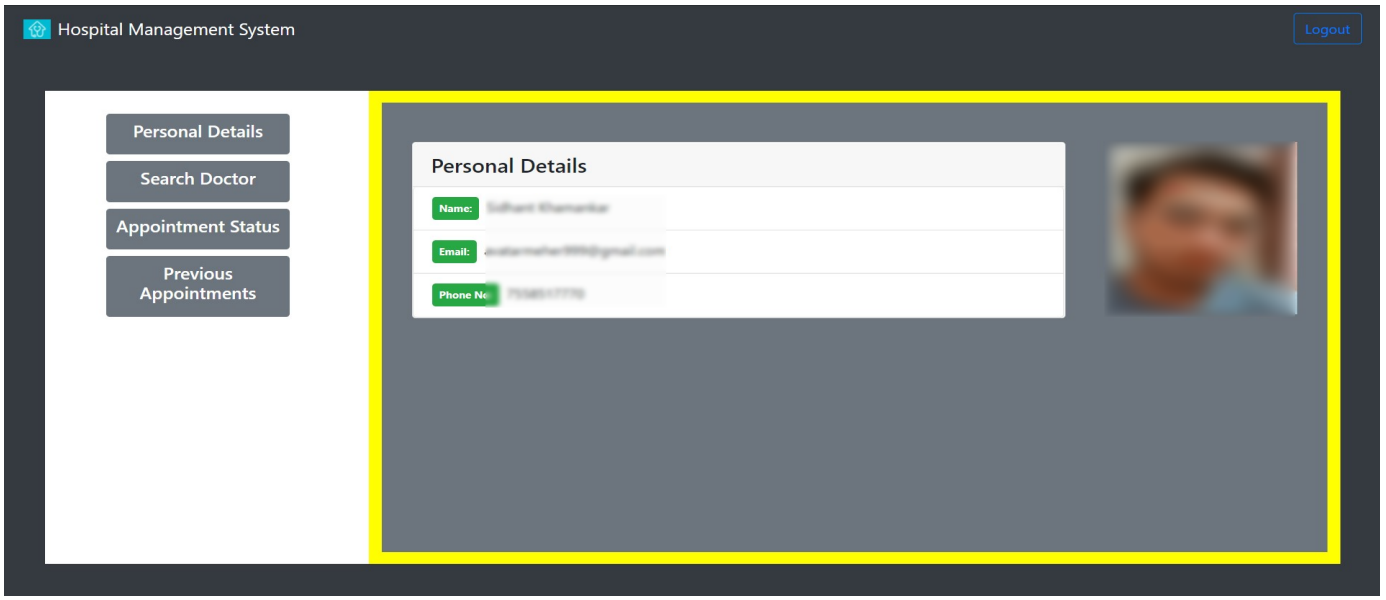
2.6 User Documentation

Patient Guide

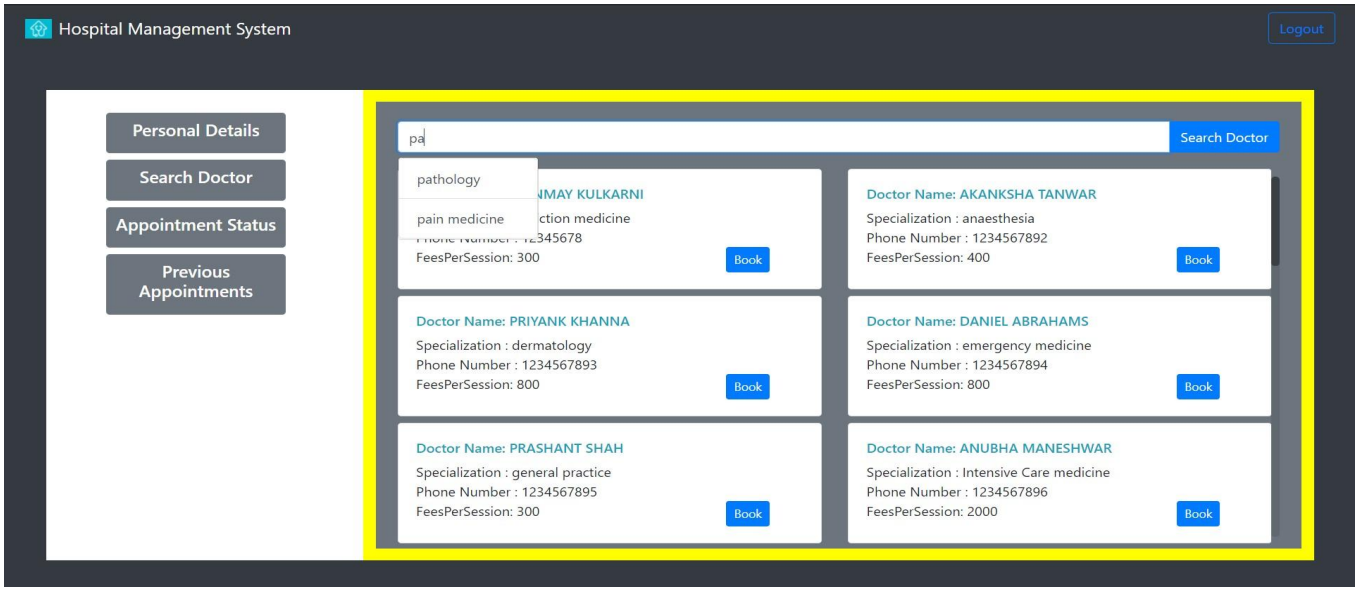
Patient Login



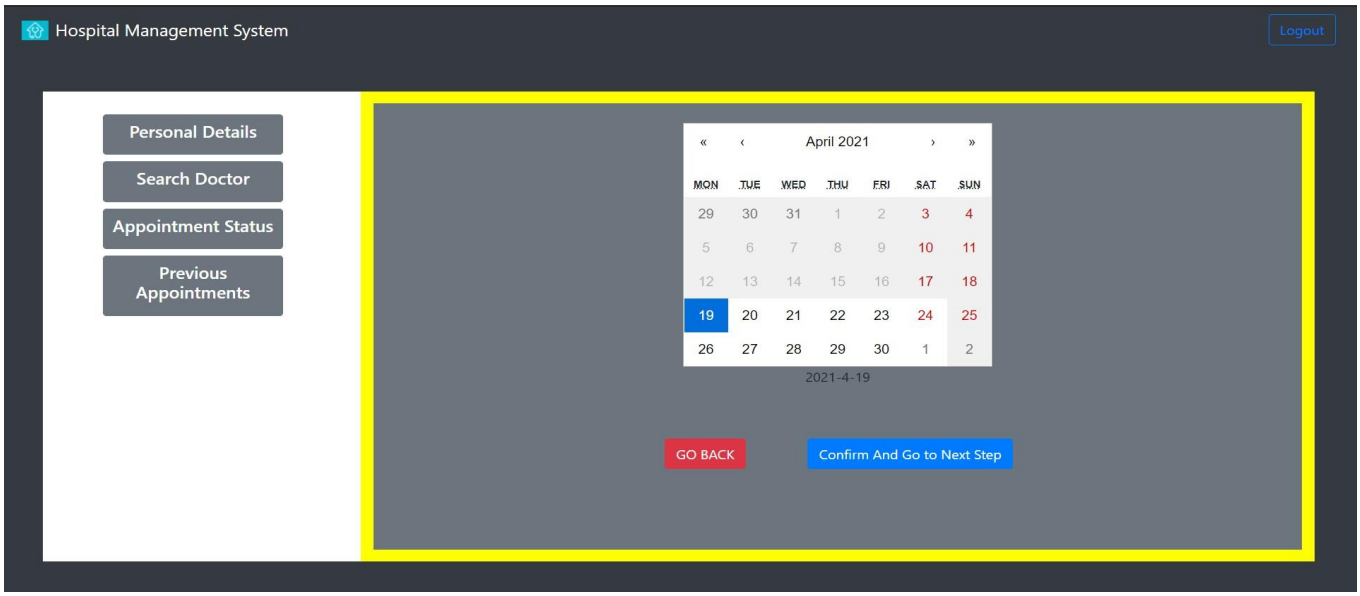
Patient Personal Details



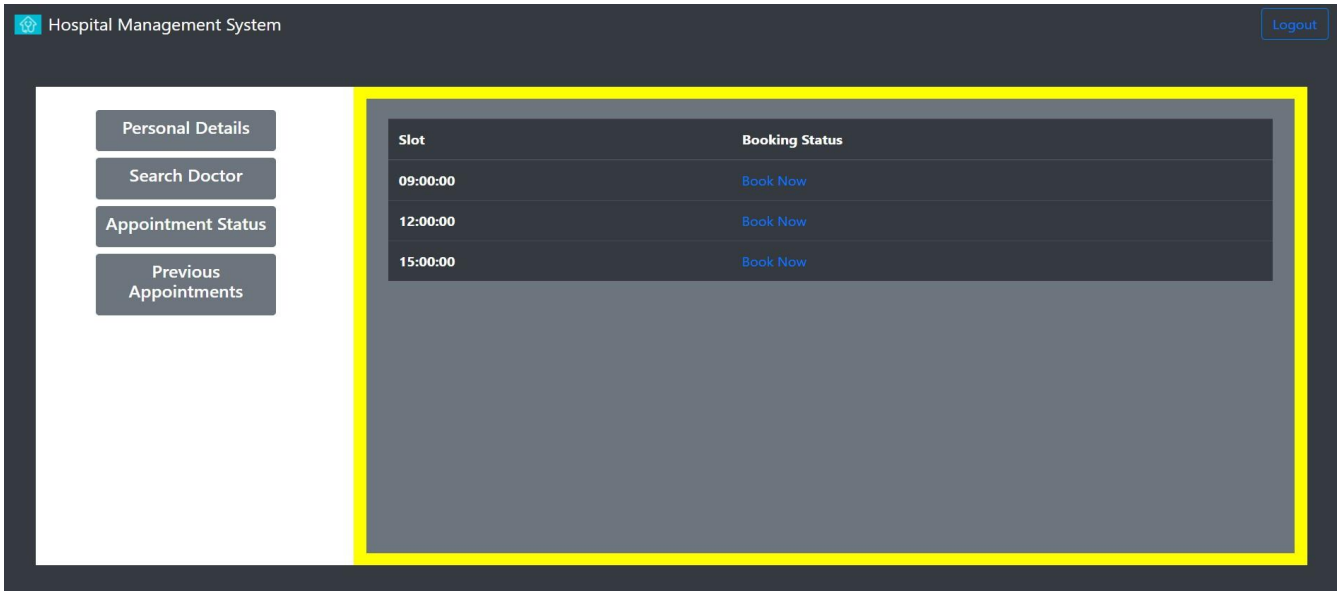
Search



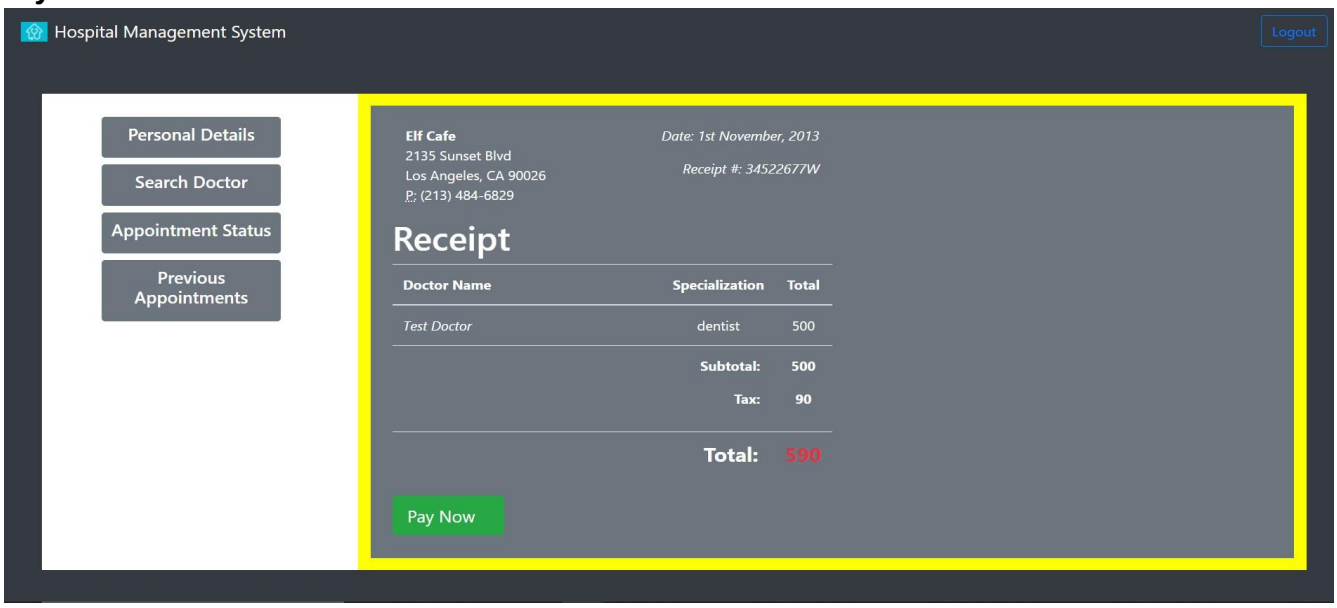
Select Date

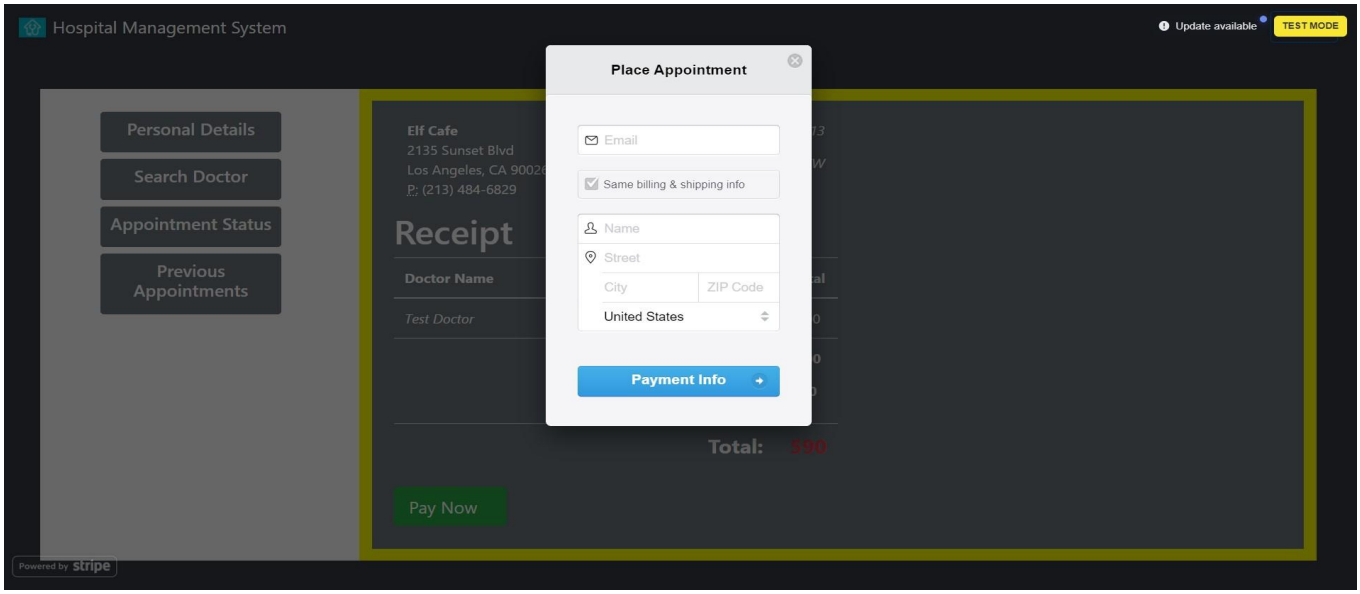


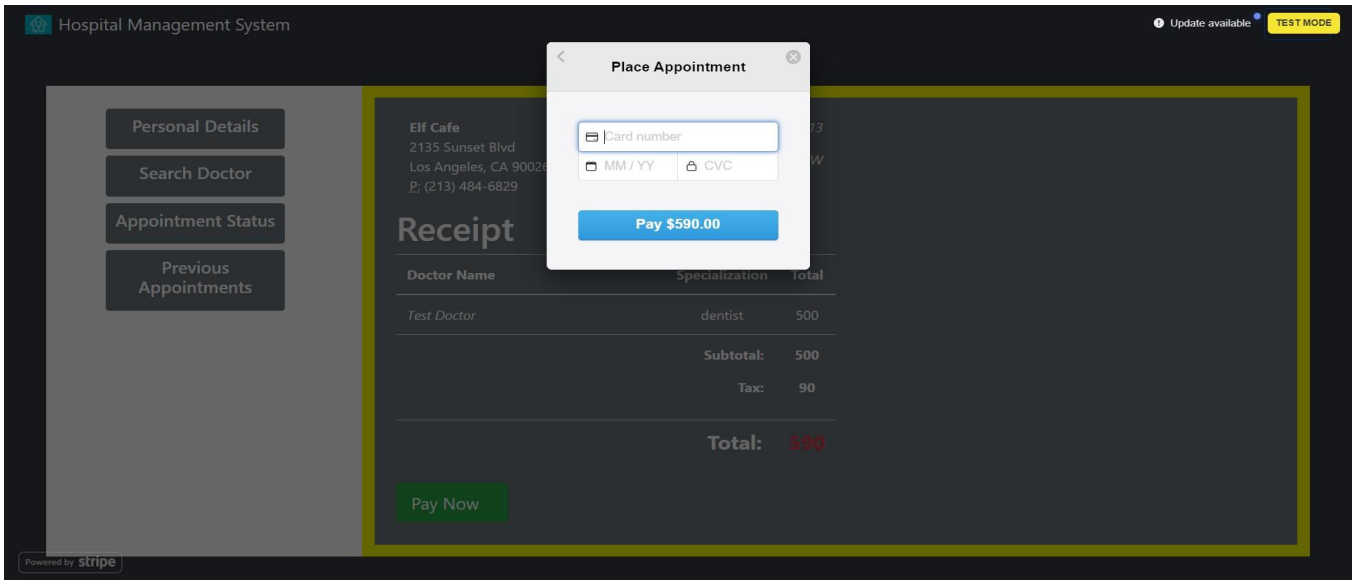
Select Slot



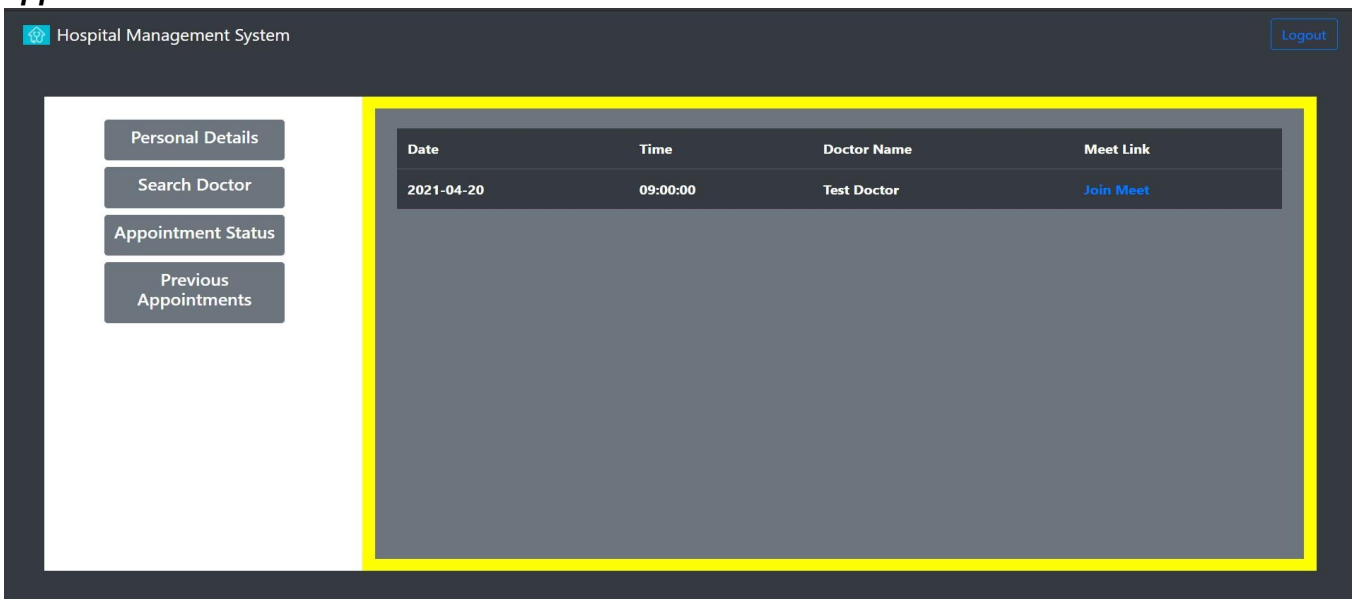
Payment



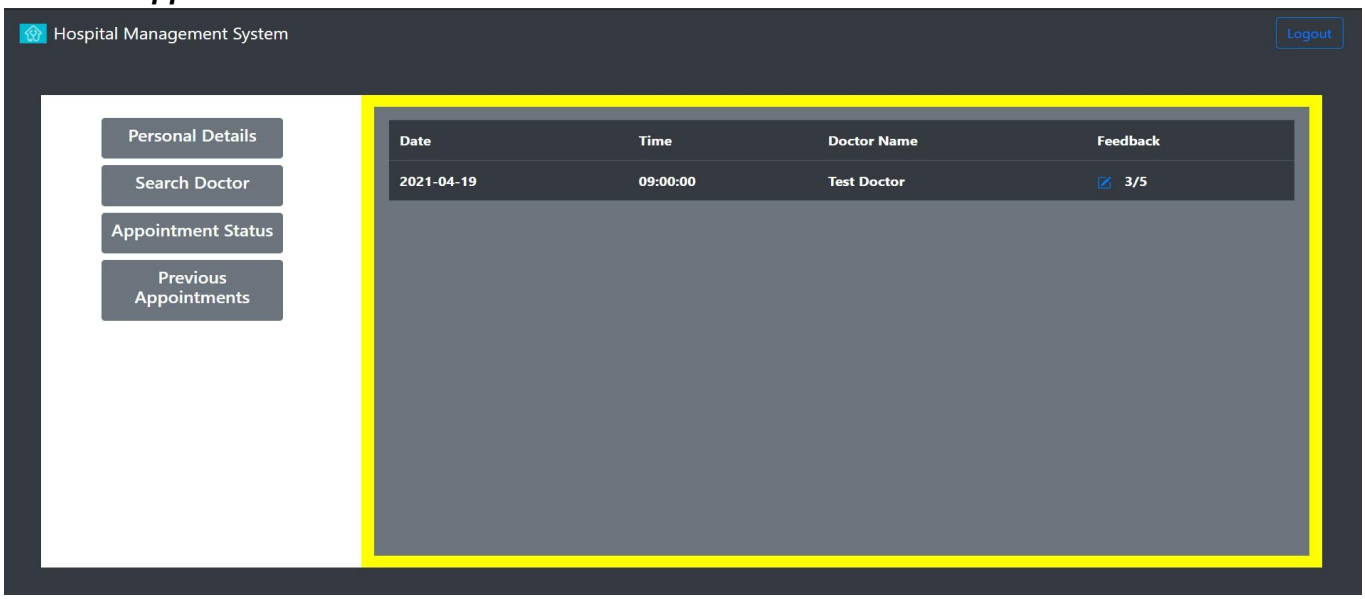

Hospital Management System
Update available
TEST MODE
Personal Details
Search Doctor
Appointment Status
Previous Appointments
Elf Cafe
2135 Sunset Blvd
Los Angeles, CA 9002
P: (213) 484-6829
Receipt
Doctor Name
Test Doctor
Total:
Pay Now
Powered by stripe
Place Appointment
Email
Same billing & shipping info
Name
Street
City
ZIP Code
United States
Payment Info


Hospital Management System
Update available
TEST MODE
Personal Details
Search Doctor
Appointment Status
Previous Appointments
Elf Cafe
2135 Sunset Blvd
Los Angeles, CA 9002
P: (213) 484-6829
Receipt
Doctor Name
Specialization
Total
Test Doctor
dentist
500
Subtotal:
500
Tax:
90
Total:
590
Pay Now
Powered by stripe
Place Appointment
Card number
MM / YY
CVC
Pay $590.00

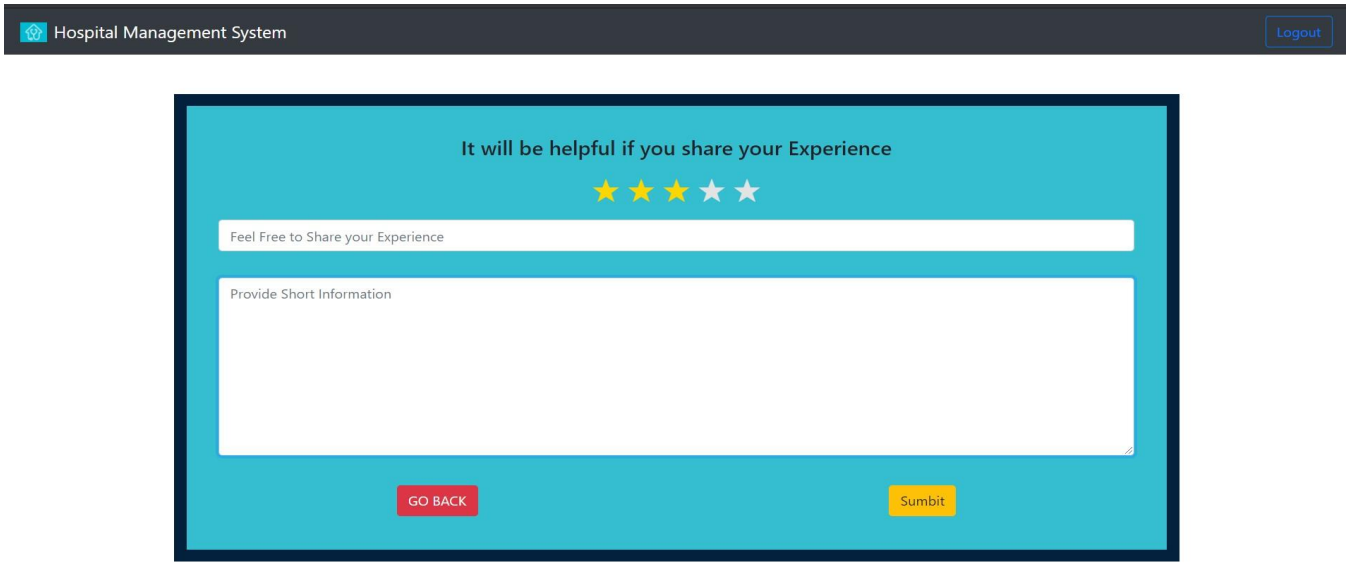
Appointment Status



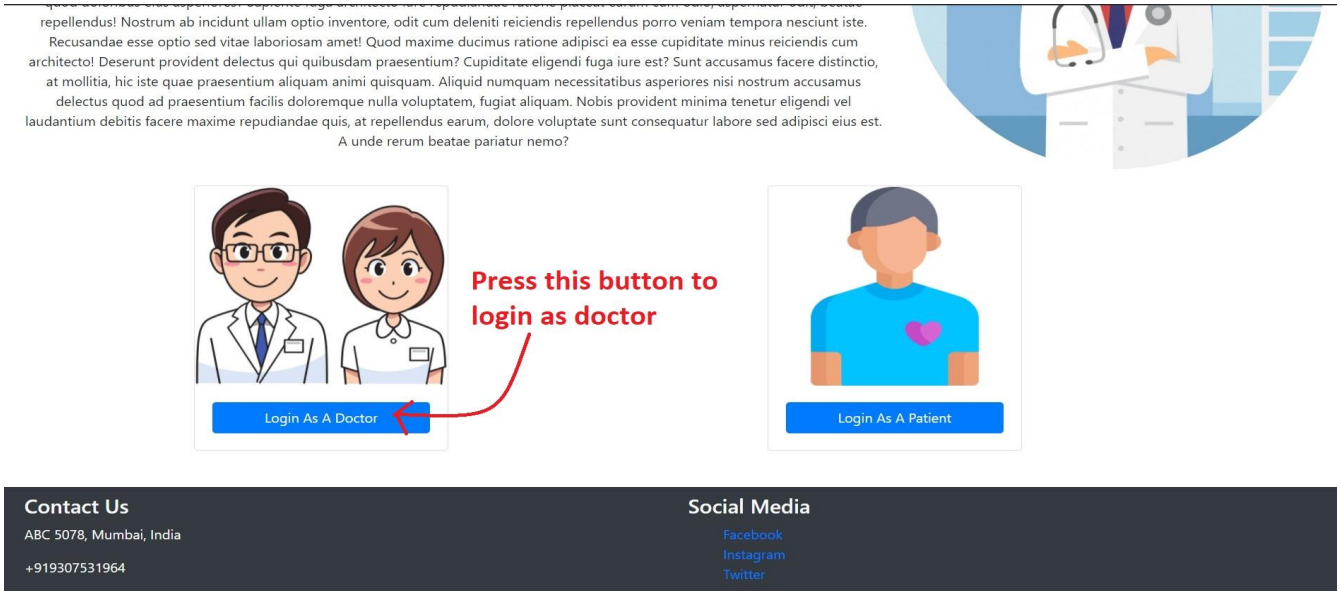
Previous Appointments



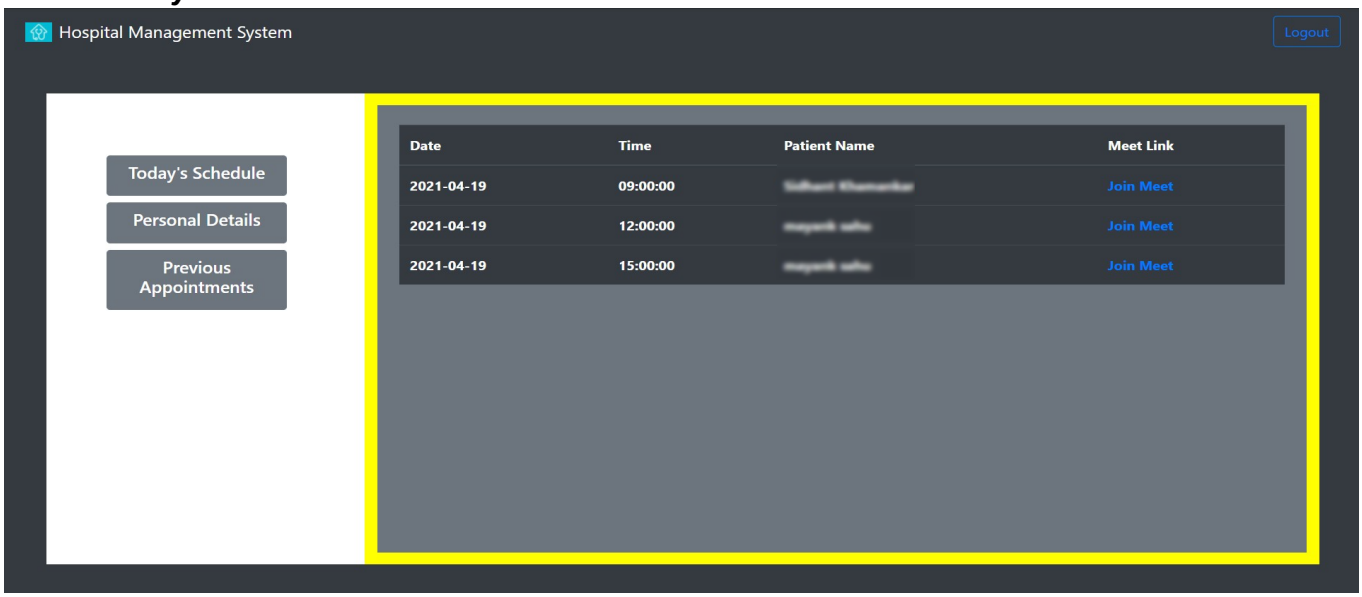
Patient Feedback



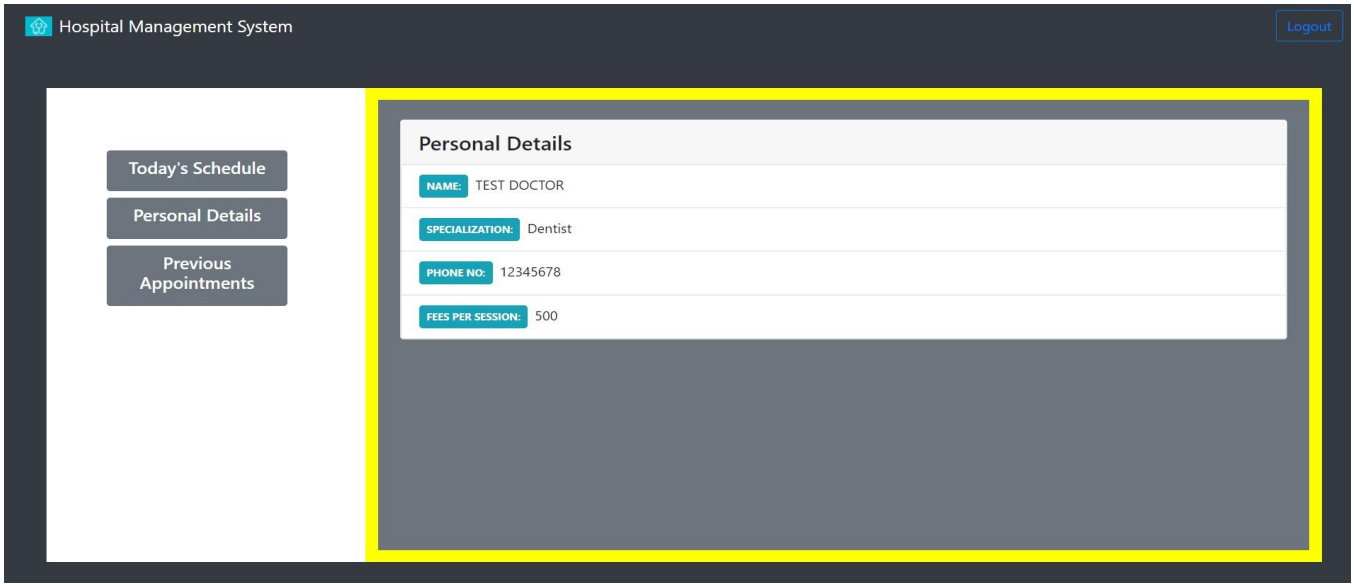
Doctor Guide:



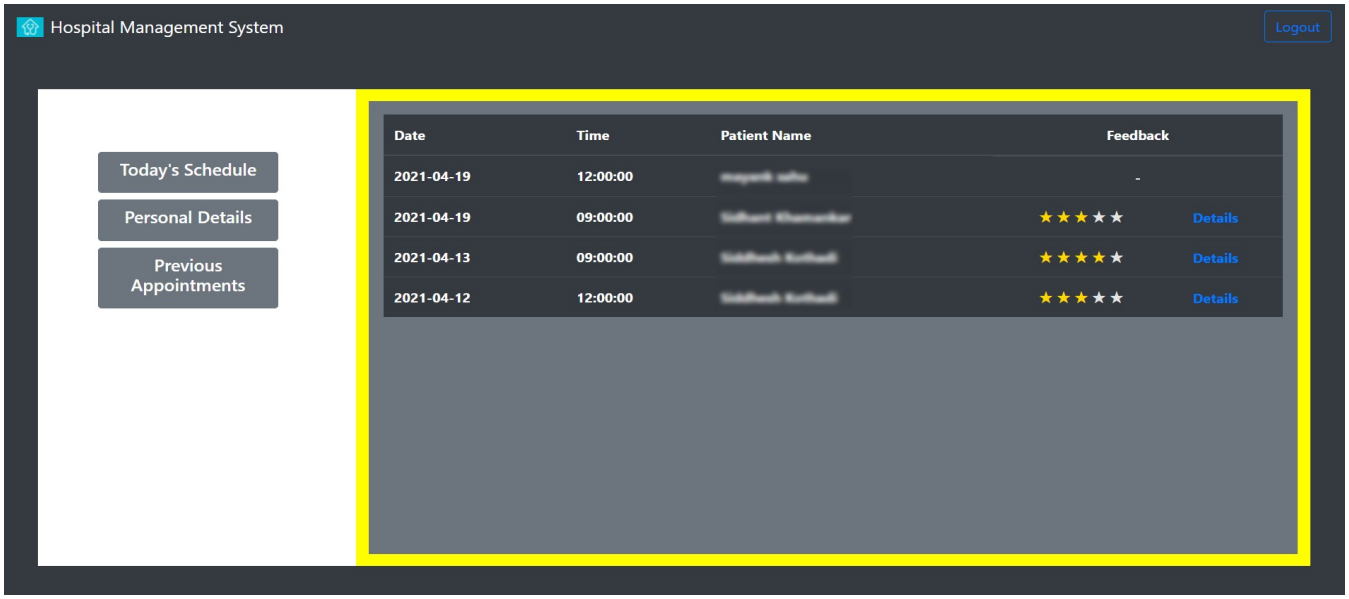
Doctor Today's Schedule



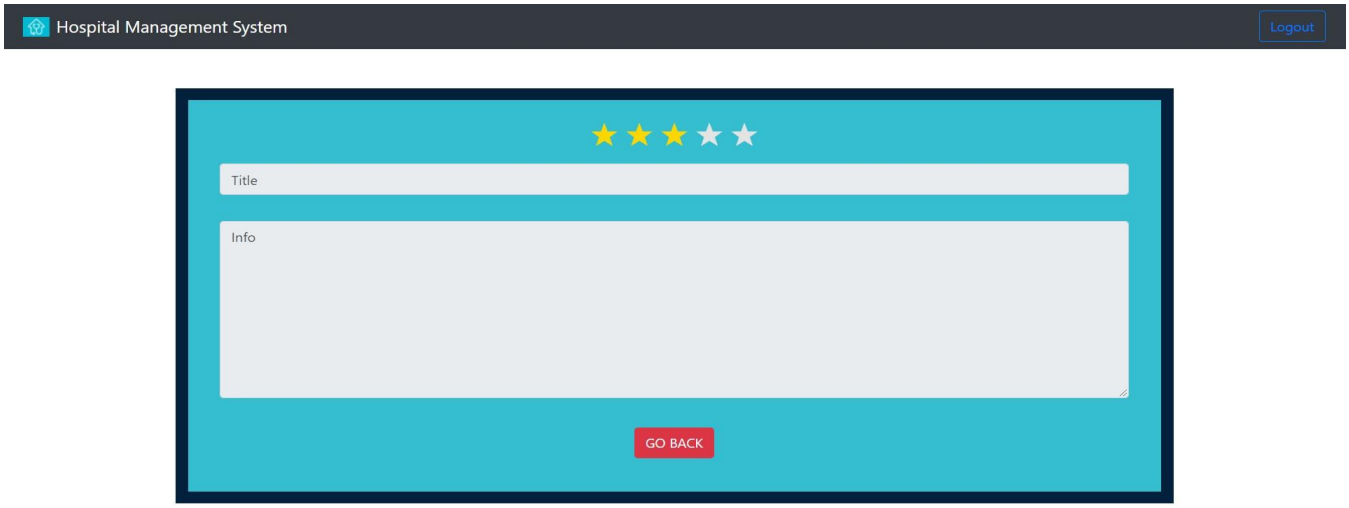
Doctor Personal Details



Doctor Previous Appointments



Doctor Feedback



2.7 Assumptions and Dependencies

Assuming that the Google Calendar along with the meet feature remains free to use.
MongoDB query performance may degrade over time as new records get added.
Software components like bootstrap, react-scrollbar, react-calendar, react-icons and jwt-decode are reused.
Dependency and performance might differ with OS and Browser type and version the system is used on.

3. External Interface Requirements

3.1 User Interfaces

Fullscreen desktop layout is recommended for best user experience
Patient Interface:
Sub Interfaces: Login, Personal Details, Search Doctor, Appointment Status, Previous Appointments
Doctor Interface:
Sub Interfaces: Login, Today's Schedule, Personal Details, Previous Appointments

3.2 Hardware Interfaces

- *Windows*
- *Web Browser that supports JavaScript*
- *Internet connection on both client and server side.*
- *Storage on server for verified data of doctor and patient profiles.*
- *REST APIs use URLs and the HTTP protocol to identify resources and transfer data.*

3.3 Software Interfaces

Server generates JSON objects depending upon the request from the client using MongoDB database and sends it to the client.

API documents:

- **Google Calendar API documentation**
- **Mongoose for DB connection and operations**
- **JWT based authentication**
- **Expressjs**

Software used	Description
Operating system	We have chosen Windows 10 operating system for its best support and user-friendliness.
Database	To store data of doctors and patients we have used MongoDB Database version 4.4.
Web technology	MERN stack is used

3.4 Communications Interfaces

HTTPS communication standard is used.
Google Calendar API: for accepting/rejecting appointments.
OAuth 2.0 protocol: Protocol used by Google for authentication and authorization.
Google meet: adheres to IETF security standards for Datagram Transport Layer Security (DTLS) and Secure Real-time Transport Protocol (SRTP)
Stripe: Stripe uses HTTPS for all services using TLS(SSL)
JWT Encryption: Login credentials are encoded using JWT and sent to the client as a token stored in the local storage. This uses HS256 algorithm.

4. System Features

4.1 Login

4.1.1 Description and Priority

This feature allows patients and doctors to login into the system for accessing their respective dashboards.
This is a high priority feature as it acts as a starting point for other features.

4.1.2 Stimulus/Response Sequences

Patient clicks on Login as Patient button and is redirected to google OAuth interface to login using his google account.
Doctors click on login as Doctor and are redirected to the credentials input page.

4.1.3 Functional Requirements

REQ-1: System denies login when login credentials do not match with the credentials stored in the database.
REQ-2: If New Patient logs in then system creates its record in the database
REQ-3: Google OAuth should be allowed for testing or production for the given website.
REQ-4: Server should be online for validating the credentials.

REQ-5: Database should be able to do the query operation.

4.2 Booking

4.2.1 Description and Priority

Patients will use this feature to book appointments with doctors according to their free slots.
The booking feature is the main objective of the project and hence has high priority.

4.2.2 Stimulus/Response Sequences

- Search for a doctor on the basis of specialization.
- Displays all doctors practicing a particular specialization
- Book a doctor.
- Schedule appointment.
- Perform payment
- Calendar Event and Video conferencing meeting is scheduled

4.2.3 Functional Requirements

REQ-1: Separate Dashboard for Patients and doctor
REQ-2: Patient dashboard has a search bar with an autocomplete feature.
REQ-3: After searching data for a matching doctor appears and the book button is seen.
REQ-4: Payment gateway is integrated for seamless payments
REQ-5: After Payment, event is created on doctor's google calendar using which they can accept/decline the appointment
REQ-6: Booking data will be appended in the database.

4.3 Feedback

4.2.1 Description and Priority

Patients will use this feature to give their feedback to a particular doctor after their consultation/appointment. This feedback can be viewed by the doctor too.
The feature has medium priority.

4.2.2 Stimulus/Response Sequences

- In the Patient dashboard if a patient clicks on previous appointments then a table with previous appointments will be shown which has a feedback icon in the last column where the patient can give rating and write title and details of the feedback and submit.
- In Doctor Dashboard, if a doctor clicks on previous appointments the a table with previous appointments will be shown with the last column showing the ratings (out of 5 star) and an icon, clicking on which doctor can see the particular patient's feedback.

4.2.3 Functional Requirements

REQ-1: Database stores the ratings and feedback.
REQ-2: There should be a previous Appointment present for giving feedback.

5. Other Nonfunctional Requirements

5.1 Performance Requirements

Some performance requirements identified are as follows:

- *The database should be able to accommodate items details according to its size.*
- *The software shall support use of multiple users at a time.*
- *The login action should not take more than 25 seconds.*
- *The application should be able to work 24X7*
- *Payment API should be efficient and should handle multiple requests.*
- *Search Query on the database should be fast to prevent unnecessary lag on the client side.*

5.2 Safety Requirements

As the database is completely on cloud there isn't a risk of catastrophic failure.
Secured Payment gateway is used.
Be extremely cautious when logging into the system using credentials as social engineering can happen.

5.3 Security Requirements

HTTPS (SSL) SSL Certification to secure data between users computer and site.
Security — MongoDB Manual Authentication using credentials and IP address is provided

5.4 Software Quality Attributes

- *AVAILABILITY: Software is available 24/7 for its users.*
- *CORRECTNESS: Information on software is accurate and fully updated.*
- *MAINTAINABILITY: System is easily manageable and maintainable.*
- *USABILITY: Friendly GUI allows seamless user interaction.*
- *SCALABILITY: System is scalable with some adjustments by the system manager.*
- *PORTABILITY: Highly portable and platform independent(provided requirements are met)*
- *TESTABILITY: Efficiently divided code makes it easy to find bugs and fix them.*

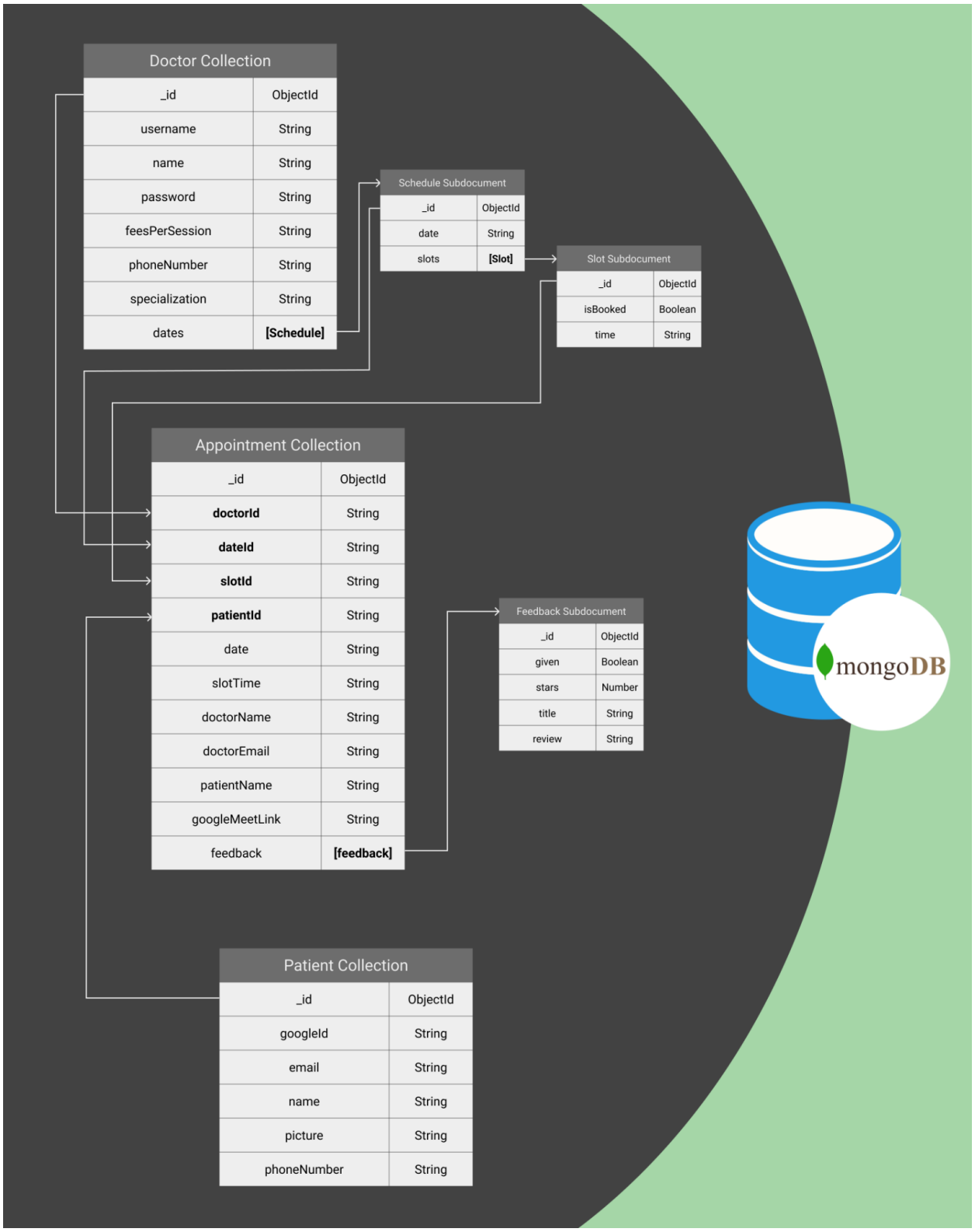
5.5 Business Rules

Developers can remove both patients and doctors if they found necessary under certain conditions.
Management of the Hospital would know about the activities taking place throughout the system to catch any fraudulent behaviour.

6. Other Requirements

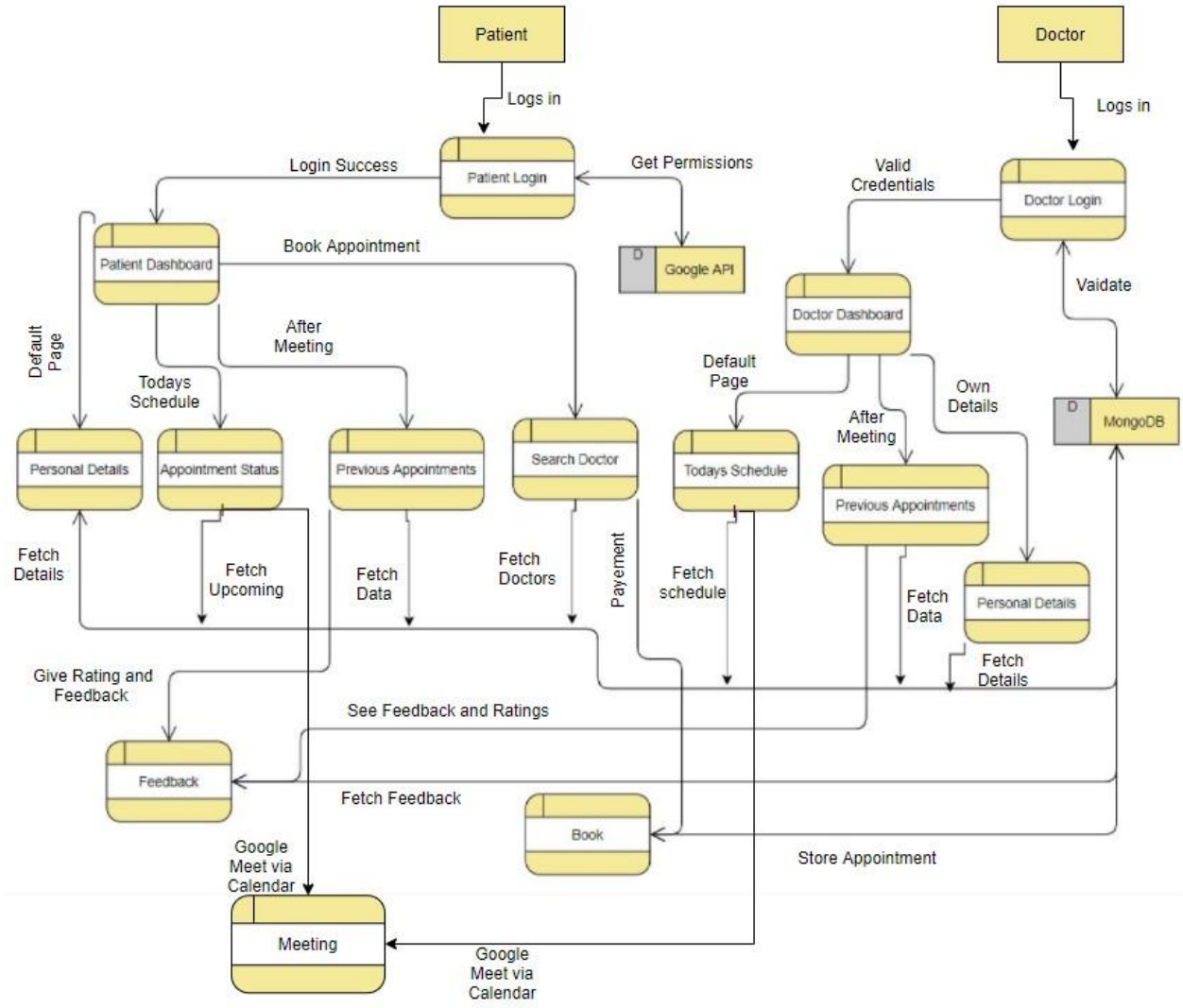
Latest version of Browser and of the required tech stack is recommended for best experience of this application. Components of this project are reusable and can be used only by the developer team having access to it.

Appendix A: Analysis Models

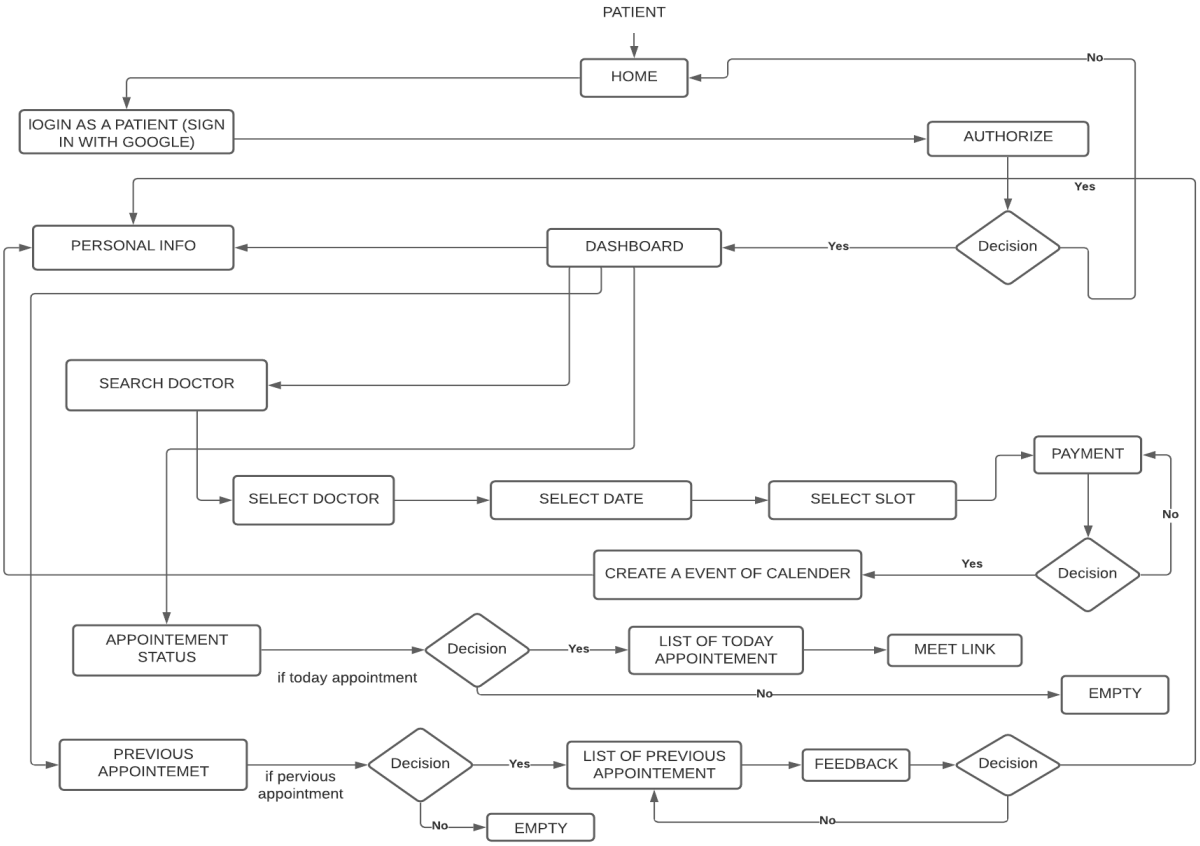


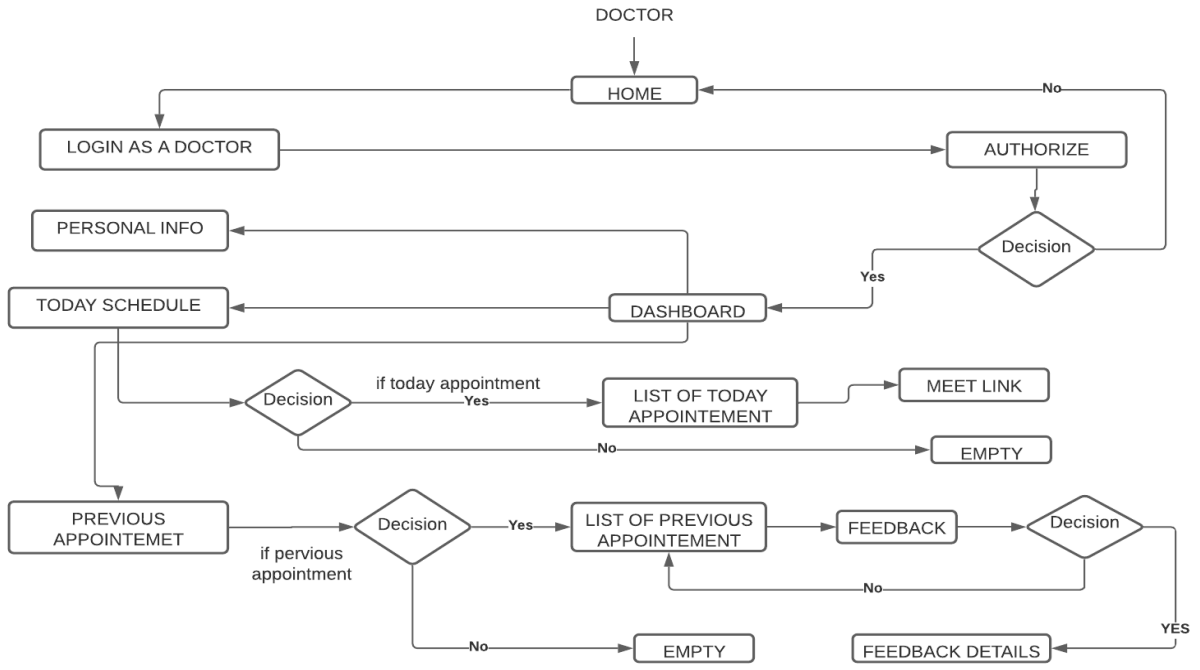
Entity-Relationship diagram

Data Flow Diagram

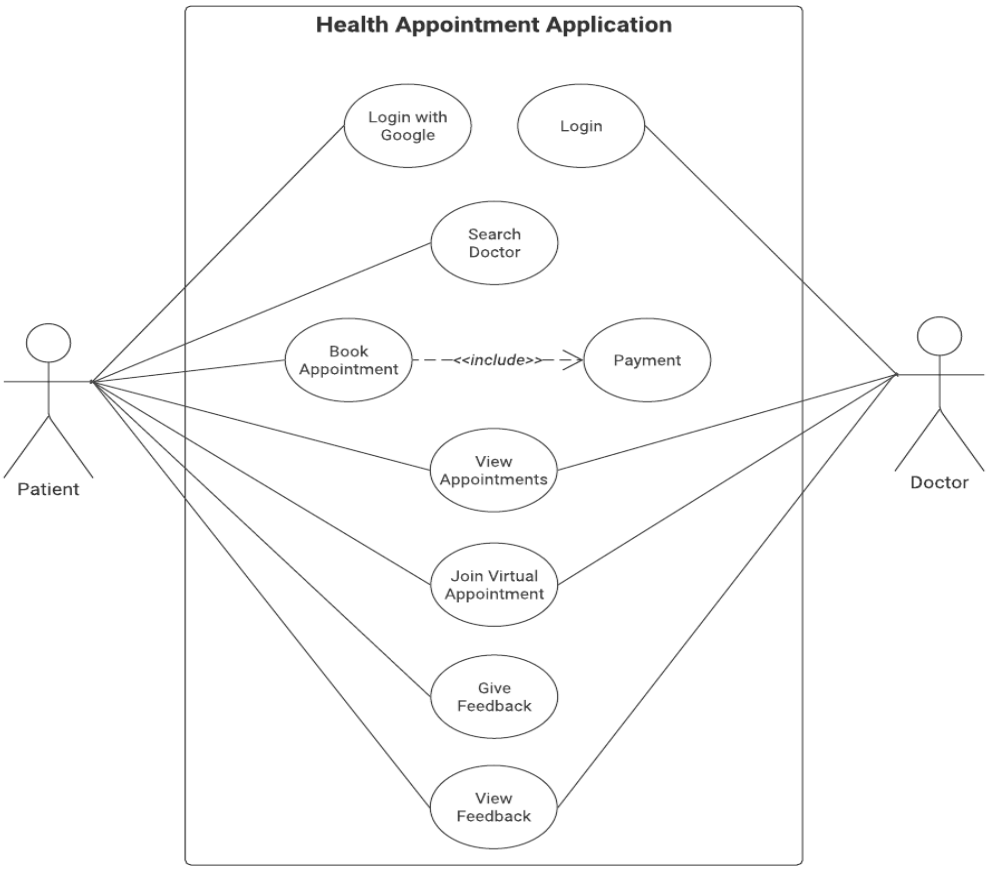


Activity Diagrams

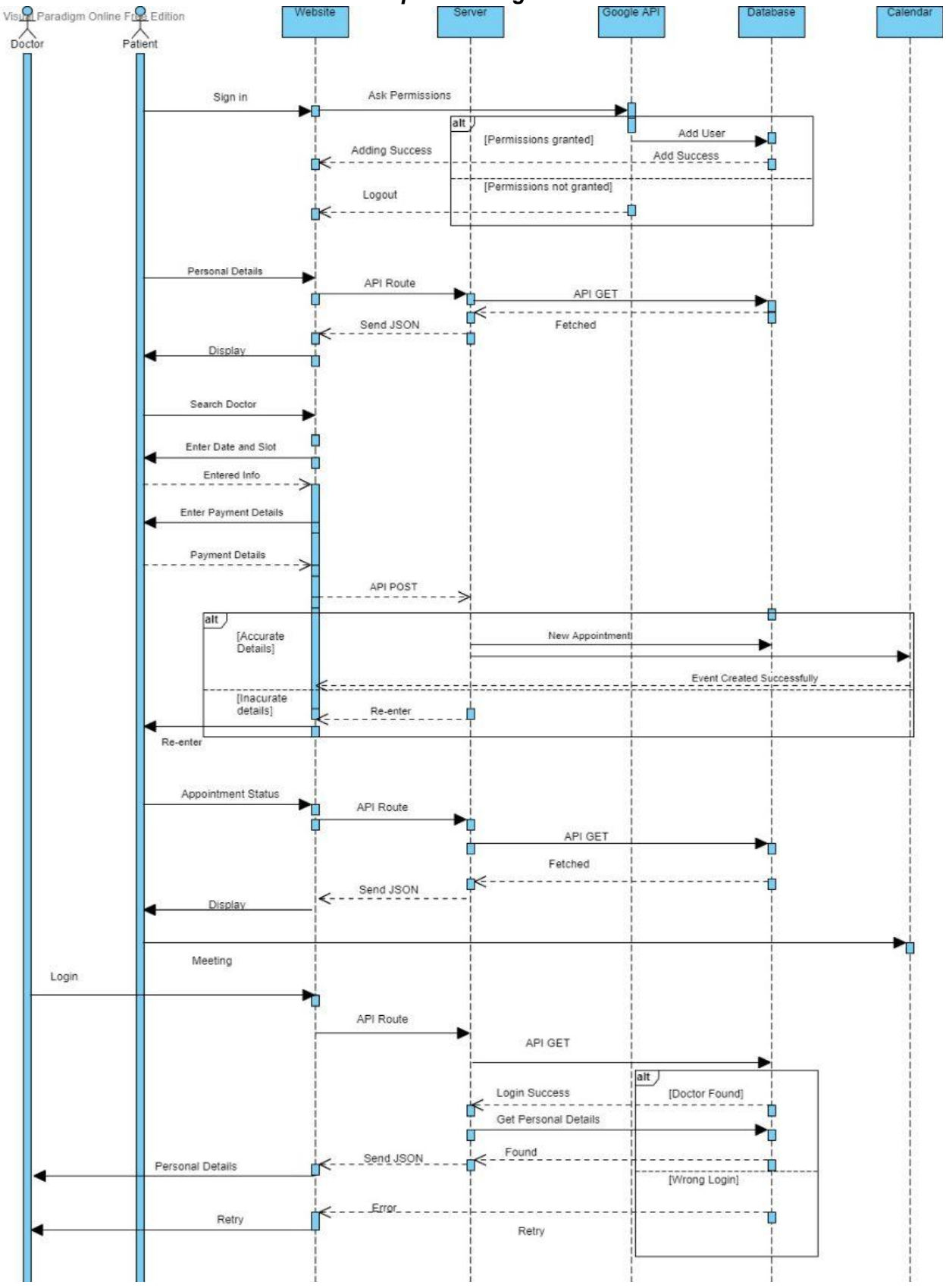


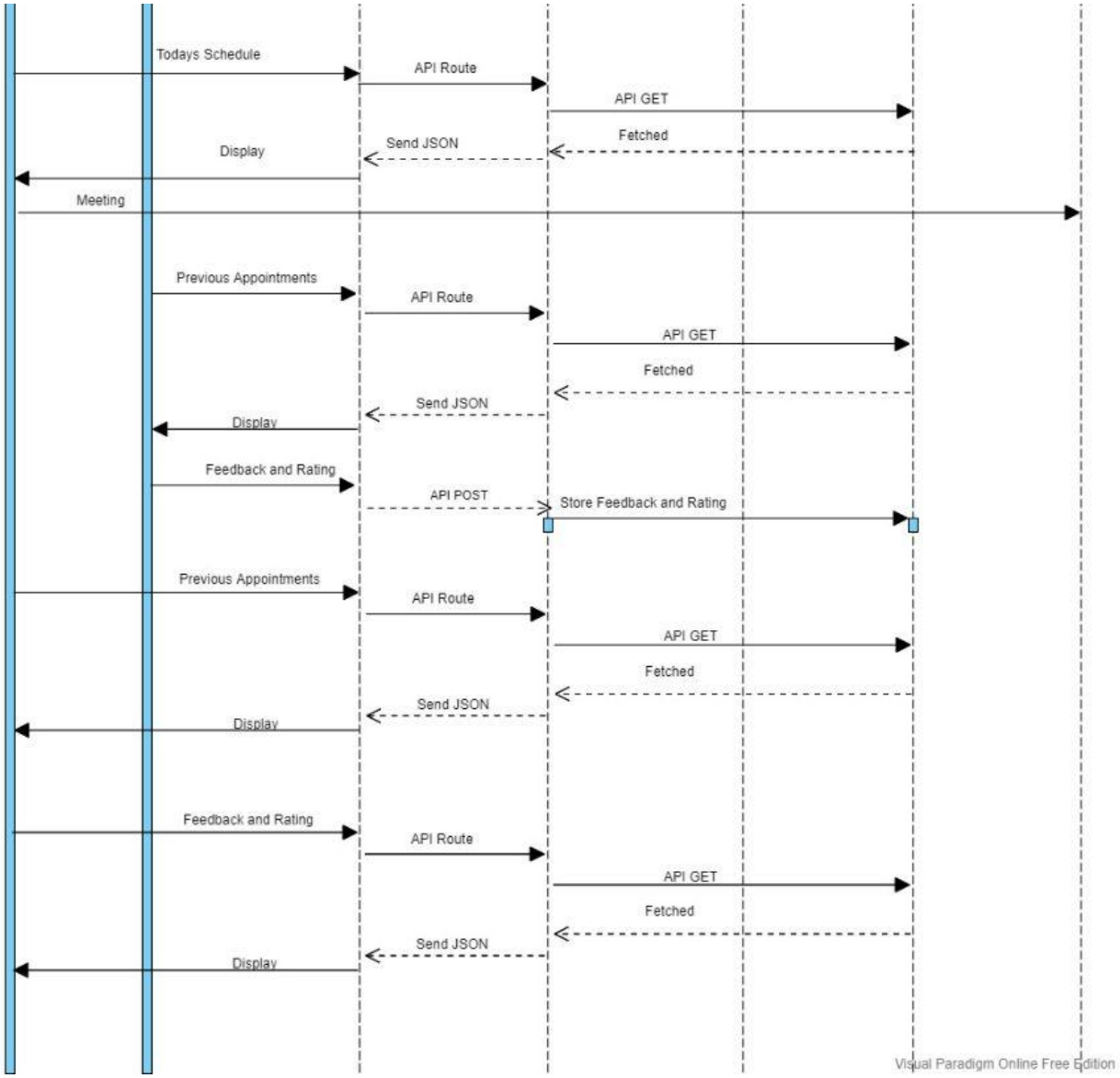


Use Case Diagram



Sequence Diagram




Todays Schedule
API Route
API GET
Fetched
Send JSON
Display
Meeting
Previous Appointments
API Route
API GET
Fetched
Send JSON
Display
Feedback and Rating
API POST
Store Feedback and Rating
Previous Appointments
API Route
API GET
Fetched
Send JSON
Display
Feedback and Rating
API Route
API GET
Fetched
Send JSON
Display
Visual Paradigm Online Free Edition