

# **Patient Management System**

A Project Report

Presented to

CMPE-272

Fall, 2021

By

Team – The Wolf Pack

December 12<sup>th</sup> 2021

Copyright © 2021

Avinash Ramesh, Nisha Mohan Devadiga, Poojashree NS

ALL RIGHTS RESERVED



# ABSTRACT

## Patient Management System

By Avinash Ramesh, Nisha Mohan Devadiga, Poojashree NS

For a healthcare provider, patient care is the primary priority. Patient Management System is a powerful, flexible, and easy software for both healthcare providers and patients, which provides improved patient care, better administration, streamlined operations, and control. Existing applications for patient management systems focus on basic patient care while others address specific purposes such as lab tests etc. Our Patient Management System is a holistic enterprise-grade software that can scale to support any number of hospitals. Below are some use case-specific functionalities:

**For Patients:** It allows them to get access to their medical records, make appointments, and ask for prescription refills, communicate with doctors through email, virtual assistance in case of queries, and a document repository that keeps track of patient's lab-reports and visitation summary which can be accessed using single-sign-on.

**For Doctors:** It allows them to access patients' health information, patient's visitation information, upload prescriptions, provide notes on the lab reports, and communicate with patients through email and appointment scheduling.

We have used Python Flask web framework for backend development, HTML, CSS, JavaScript for front-end development, and SQLite as the database to achieve these features.

We have also done the following integrations to our system:

- OAuth/Okta for single-sign-on.
- Chatbot as a pocket doctor
- Google Maps API for finding hospital locations.

## Acknowledgments

We are pleased to present to you this report of a project entitled “**Patient Management System**”

First of all, we would like to thank Professor **Andrew Bond** for sharing with us the most important topics, valuable comments and suggestions needed. He encouraged us to work on this project.

We are also grateful to the quorum for giving us the opportunity to work with them and for providing us with the necessary resources for the project.

We also thank everyone who helped us make this project possible. We are extremely grateful to everyone involved in this project because without their inspiration and valuable suggestions, the project would not have been developed on time.

With our sincere thanks,

"The Wolf Pack"

# **Table of Contents**

## **Chapter 1 Introduction**

- 1.1 Project goals and objectives
- 1.2 Problem and motivation
- 1.3 Project application and impact
- 1.4 Project results and deliverables
- 1.5 Project report structure

## **Chapter 2 Project Background and Related Work**

- 2.1 Background and used technologies
- 2.2 State-of-the-art technologies
- 2.3 Literature survey

## **Chapter 3 System Requirements and Analysis**

- 3.1 Domain and business requirements
- 3.2 Customer-oriented requirements
- 3.3 System function requirements
- 3.4 System behavior requirements
- 3.5 System context and interface requirements
- 3.6 Technology and resource requirements

## **Chapter 4 System Design**

- 4.1. System architecture design
- 4.2. System data and database design (*for software project only*)
- 4.3. System interface and connectivity design
- 4.4. System user interface design (*for software project only*)
- 4.5. System component API and logic design (*for software project only*)
- 4.6. System design problems, solutions, and patterns

## **Chapter 5 System Implementation**

- 5.1. System implementation summary
- 5.2. System implementation issues and resolutions
- 5.3. Used technologies and tools

## **Chapter 6 System Testing and Experiment**

- 6.1. Testing and experiment scope
- 6.2. Testing and experiment approaches
- 6.3. Testing report (or case study results)

## **Chapter 7 Conclusion and Future Work**

- 7.1 Project summary
- 7.2 Future work

## List of Figures

Figure 1. System Action diagram	Page No. 6
Figure 2. User Requirement diagram	Page No. 6
Figure 3. System behavior diagram	Page No. 7
Figure 4. Application Architecture Diagram	Page No. 9
Figure 5. Entity Relationship Diagram	Page No. 10
Figure 6. System Interface Diagram	Page No. 11
Figure 7. Application Use Case Diagram	Page No. 11
Figure 8. Authorization Flow Diagram	Page No. 12
Figure 9. Doctor Flow Diagram	Page No. 12
Figure 10. Patient Flow Diagram	Page No. 13



## **Chapter 1. Introduction**

### **1.1 Project goals and objectives**

The fundamental goal of this project is to broaden an internet site for a hospital to offer a green and in your price range manner of creating appointments and assist in all of the associated tasks: hospital and physician time desk management, queue management, and affected person appointments exportation.

It is needed that sufferers have a quick and smooth entry to hospital's offerings and administer all of the appointments made.

Meanwhile, it's also required that doctors can without difficulty see all the imminent appointments and provide the effects to those appointments to the sufferers.

### **1.2 Problem and motivation**

The high level of competition in the provision of medical services requires not only the delivery of high-quality

healthcare services but also constant work with patients who are the main customers of medical centers. economics.

In the case of manual appointment booking, the limitations are numerous:

- Constant phone calls reduce productivity: the patient, to make an appointment and not have to visit the clinic in person, will phone call from the clinic. This can cause several problems:

- There may be a missed call because the line is busy at that time. It also means losing a customer.
- Front desk staff can often be busy with phone calls and may not be able to reach patients in the lobby. This means that, for adequate service, at least two receptionists are needed, which can be quite expensive for small clinics.
- Limited office hours: Potential customers who can only visit or call the clinic at one time will be lost customers if the clinic is not in operation at that time.
- More staff needed: As we said before, more staff (especially the front desk) may be needed, which translates into increased monthly costs.

### **1.3 Project application and impact**

Patient Portal aims to improve communication between patients and caregivers and educate patients. This allows patients to be better informed about their health, makes office visits more efficient and benefits patients and providers and improves care.

In addition, the Patient Portal allows patients to have 24-hour access to connect with their providers, ask and answer questions, and also a pocket doctor for problems, a small health, and virtual support.

### **1.4 Project results and expected deliverables**

Outcome of our project is a Patient Management Portal, where patients can do a self-registration, store their details, ability to schedule/reschedule or cancel their appointments

with the Doctors provides the ability to upload the lab reports and visit summaries to the Doctors. Project distributions are uploaded to the code repository, i.e., GitHub

## **1.5 Project report Structure**

Since hospitals are such a part of our lives, they must keep track of the daily activities and records of patients' doctor's nurses and other information for the hospital to function properly and successfully.

The current system requires many paper forms with data stores distributed across the hospital's management infrastructure. Often the information is incomplete or does not meet regulatory standards. Forms are often lost in transit between departments requiring a comprehensive audit process to ensure that no important information is lost. Multiple copies of the same information exist in the hospital and can cause data inconsistencies in different data repositories. To make an appointment with a doctor the patient has to call the hospital's reception or go to the hospital which takes a long time.

To overcome this, we have developed a patient management system software that allows patients to self-register, store examination summaries, test reports and essentially book appointments with doctors. directly without inference from management. In addition, we have a chatbot that provides virtual assistance to the user. It also acts as a pocket doctor.

## **Chapter 2 Background and Related Work**

### **2.1 Background and used technologies**

Hospitals are now using a manual system to manage and maintain critical information. The current system requires many paper forms with data stores spread across the hospital's management infrastructure.

Information is often incomplete or does not meet regulatory standards. Forms are often lost in transit between departments requiring a comprehensive audit process to ensure that no important information is lost. Multiple copies of the same information exist in the hospital and can cause data inconsistencies in different data repositories.

To make an appointment with a doctor the patient has to call the hospital's reception or go to the hospital which takes a long time.

### **2.2 Literature survey**

In our present daily life, people no longer tolerate waiting in the doctor's clinic. Lifestyles have changed and people lead busy lives. Many people have to take time off work to go to the doctor's office and they find their time as valuable as the doctors.

Proper and efficient appointment scheduling is critical to the normal operation of a doctor's office. There are many factors to consider when making an appointment. Patients who have made an appointment a week or even a few months in advance want to be seen within 15 to 20 minutes of arriving at the doctor's office. The doctor wants the patient to have smooth circulation during the appointment time. Sick or injured patients want to be

able to see their doctor on the day of their illness or injury. They would rather be given a specific time even if it is later in the day than walk into the office and wait for an empty period.

### **Methods of Scheduling**

Two methods are used to schedule appointments. Appointments can be scheduled manually using an appointment book. They can also be programmed electronically by a computer. These methods are described in more detail below.

#### **Appointment booking**

Each doctor in the office can have his or her own notebook or a notebook can meet the needs of two or more doctors. Appointment books are available in the following formats: pages for a day page that show a week when opened (on two pages) or pages with two or three doctors' schedules for a day. The pages are then divided into time periods. Pages are usually divided into 10- or 15-minute intervals. The medical assistant should choose the format of the appointment book that meets the needs of the practice.

The appointment book is usually kept in pencil so that information can be changed if necessary (e.g., when rescheduling a patient's appointment). When preparing for the day's visits it is a good idea to create a typed or hand-printed patient list called a daily appointment schedule. This list or appointment book (if kept in ink) should be kept as a permanent record.

### **Scheduling Daily Appointments**

The list of patients to see that day known as daily appointment scheduling serves several purposes. It is used as a guide to extract the patients' medical records for that day. It is also used as an office reference for patients with scheduled appointments that day. Usually, the daily appointment schedule contains the patient's name and phone number and the reason for their visit (e.g., new patient physical exam re-check).

If the doctor's office uses an appointment book the daily appointment schedule must be typed or printed by hand by the physician assistant. Typically, a computer is used to schedule appointments, and lists are printed or viewed directly by the employee on the computer. Under Health Insurance Portability and Accountability Act (HIPAA) requirements, daily appointments should never be posted in an office accessible to patients. This list may be updated and reprinted if there are any changes or additions or changes that can be made in dark blue or black ink. The office must maintain an up-to-date electronic or paper file to verify tax returns and insurance claims.

### **Wave planning**

With wave planning three or four patients are scheduled every half hour and are seen in the order they arrive at the office. The purpose is for patients to come in "batches" so that there are always patients waiting for examination. Sometimes sick patients are seen before routine appointments.

This scheduling system assumes that a number of patients will have to be scheduled. Sometimes patients become upset when they realize that another patient has been given the

same appointment time but a simple explanation can often reassure the patient. The physician assistant might say “We schedule all of our patients on time and then they are seen in the order they arrived. There’s always a patient coming in and we’ve found that waiting times are often shorter.

### **Open for Reservations**

Sometimes a patient does not receive a specific appointment time ut is invited to come in for a period of time for example 9:00 a.m. to 11:00 a.m. The patients are then viewed in order of arrival. In an open reservation system, patients with acute injuries or illnesses are often seen before patients with fewer complaints.

Sometimes physician offices and clinics have designated consultation hours for acute conditions before normal business hours. In this situation, patients are seen in order of arrival. Open Reservations work best when there is a constant flow of patients or when the clinic is not busy. Due to the unpredictable flow of patients, patients often have to wait a long time.

## **2.3 State-of-the-art**

Many physician offices allow patients to request routine appointments through the office’s website. Office staff often schedule appointments doing their best to accommodate patient requests. Once the appointment has been made staff will notify the patient electronically or by mail. Some practices even provide patients with limited access to their appointment

scheduling software so that patients can book their own appointments if they have the desired time. This is called patient self-programming.



## Chapter 3 System Requirements and Analysis

### 3.1. Domain and business Requirements

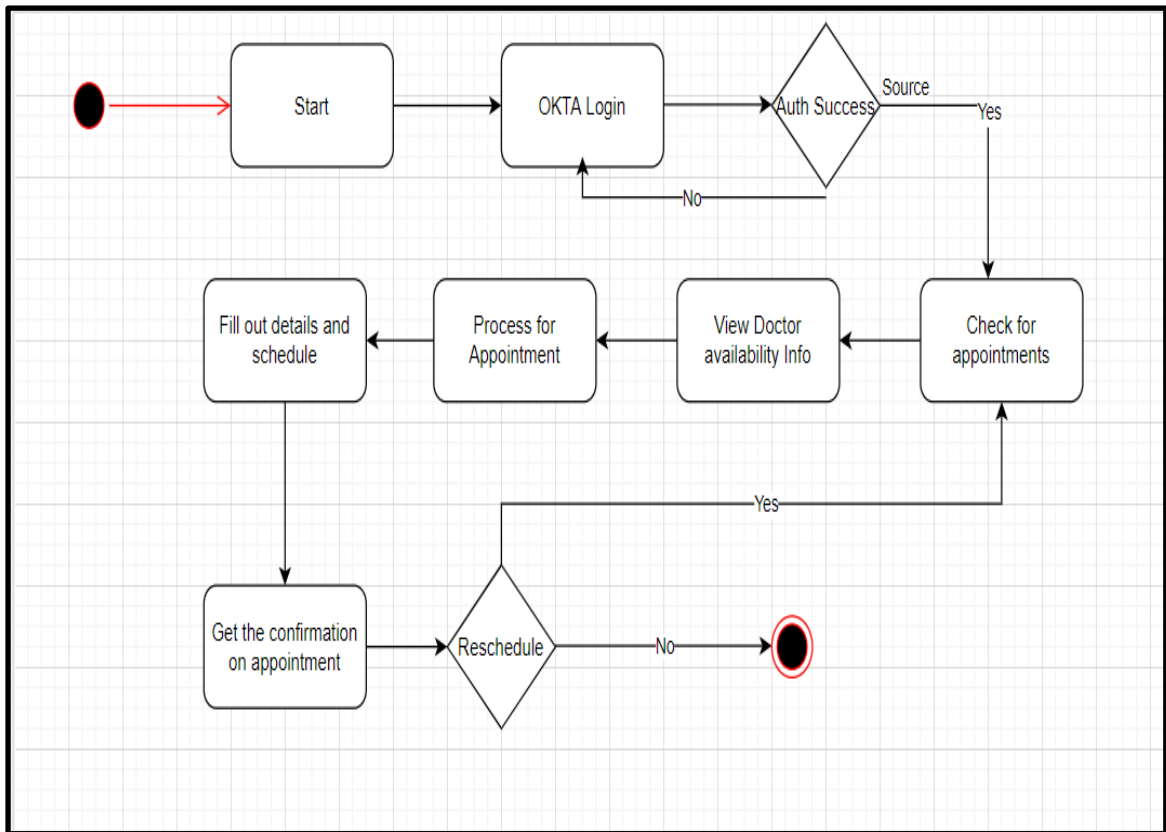


Fig: 1

### 3.2. Customer-oriented requirements

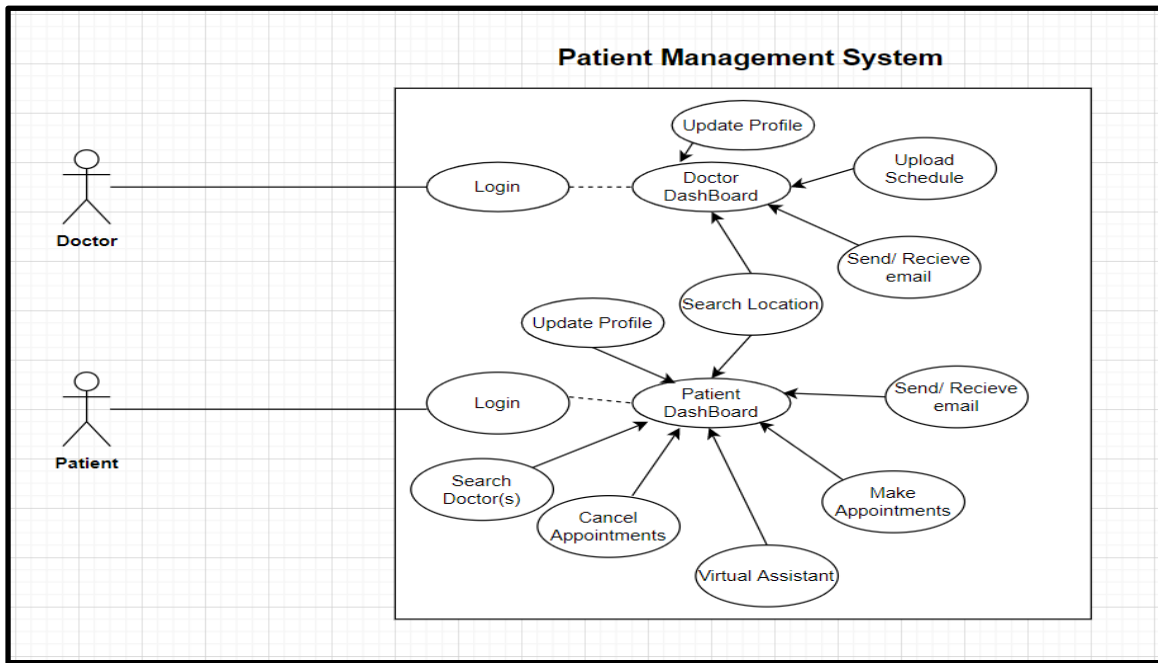


Fig: 2

### 3.3. System function requirements

- Authentication of the user whenever he/she logs into the system.
- A verification email is sent to the user whenever he/she registers for the first time on some software system.
- Appointment Scheduling capability
- Appointment Rescheduling capability
- Confirmation email on booking/canceling appointments.
- View doctor-related information.
- Update doctor availability.

- View patient appointments and messages.
- Users profile update.

### 3.4. System behavior requirements

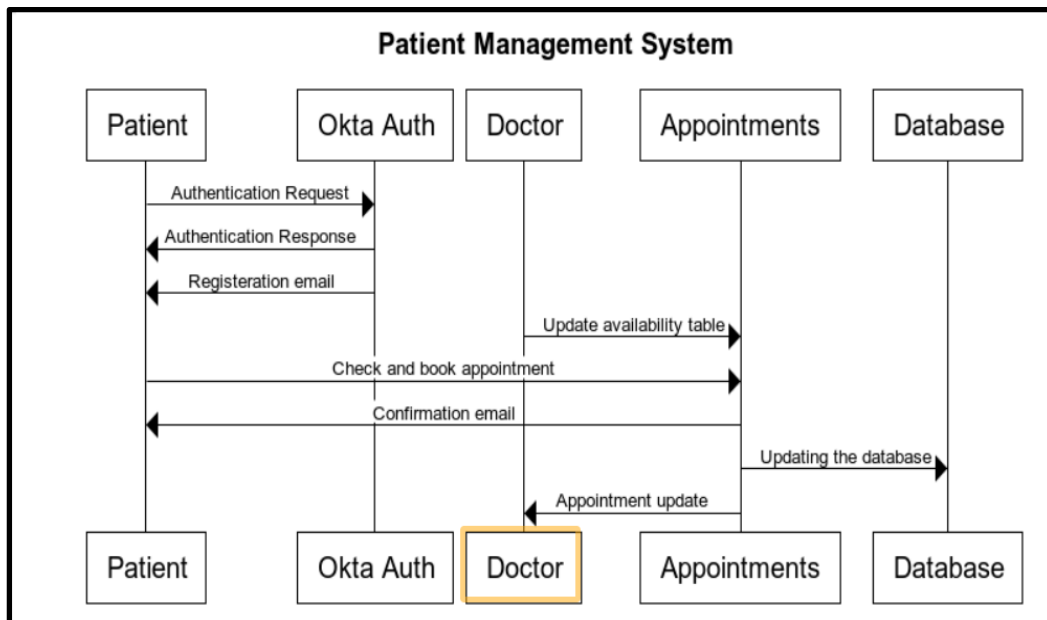


Fig: 3

### 3.5. Context and interface requirements

- Secure Login for users
- Appointment's availability check and booking.
- Appointment rescheduling / Cancellation
- View all available doctors and specialization
- View patient profile

### **3.6. Technology and resource requirements**

We have used Python Flask web framework for backend development, HTML, CSS, JavaScript for front-end development, and SQLite as the database to achieve these features.

## Chapter 4 System Design

### 4.1 System architecture design

The below system architecture diagram shows the Patient Management system interacting with Okta for Single sign-on, user creation, and authentication, SQLite to connect with a database where the data related to doctors, patients, schedule and appointments, doctor's rating are stored. The diagram also shows integration with Google API like google maps and google mail which is used to search hospital locations and send/receive emails from Doctor and Patients respectively. The system also integrates with Chatbot which is a Virtual Assistant to aid patients virtually.

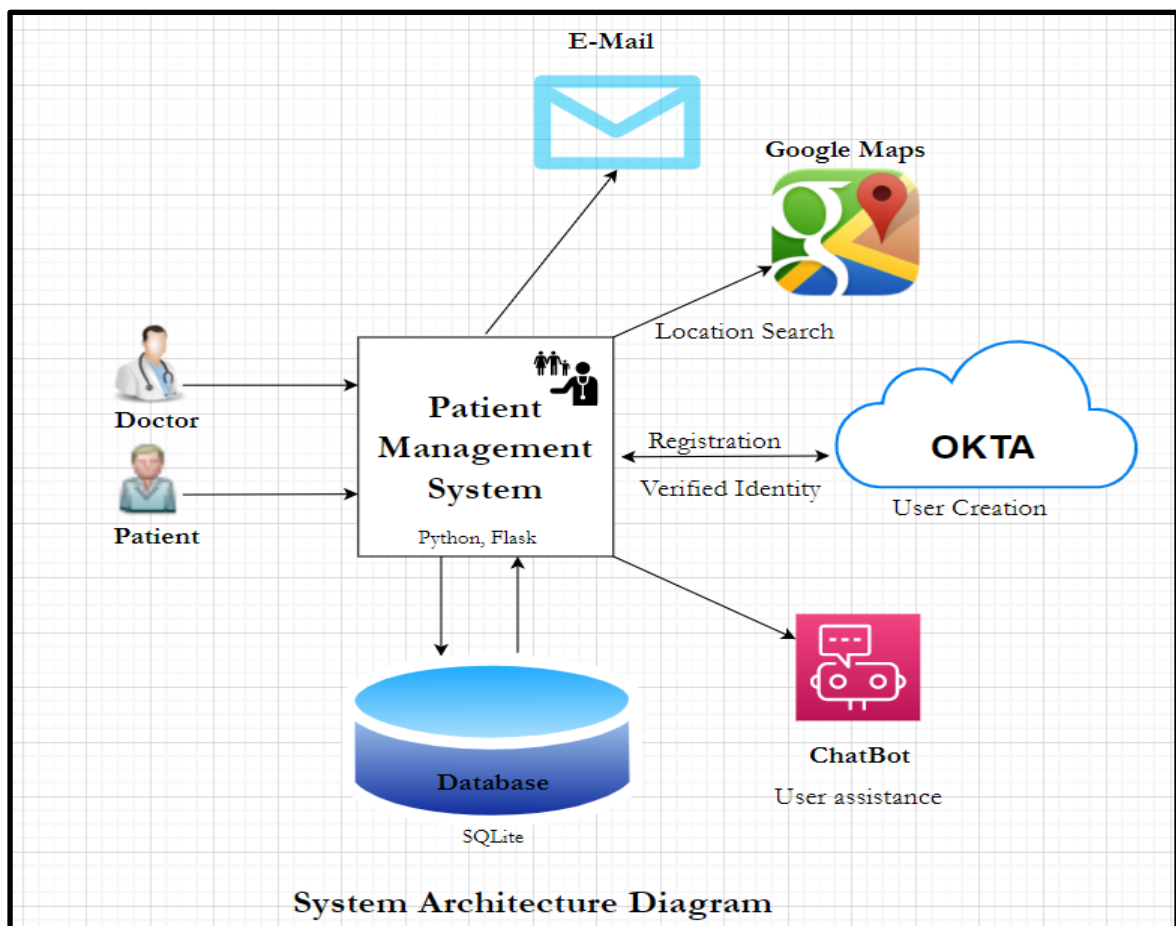


Fig: 4

## 4.2 System data and database design

Below is the ER diagram of the Patient Management Database.

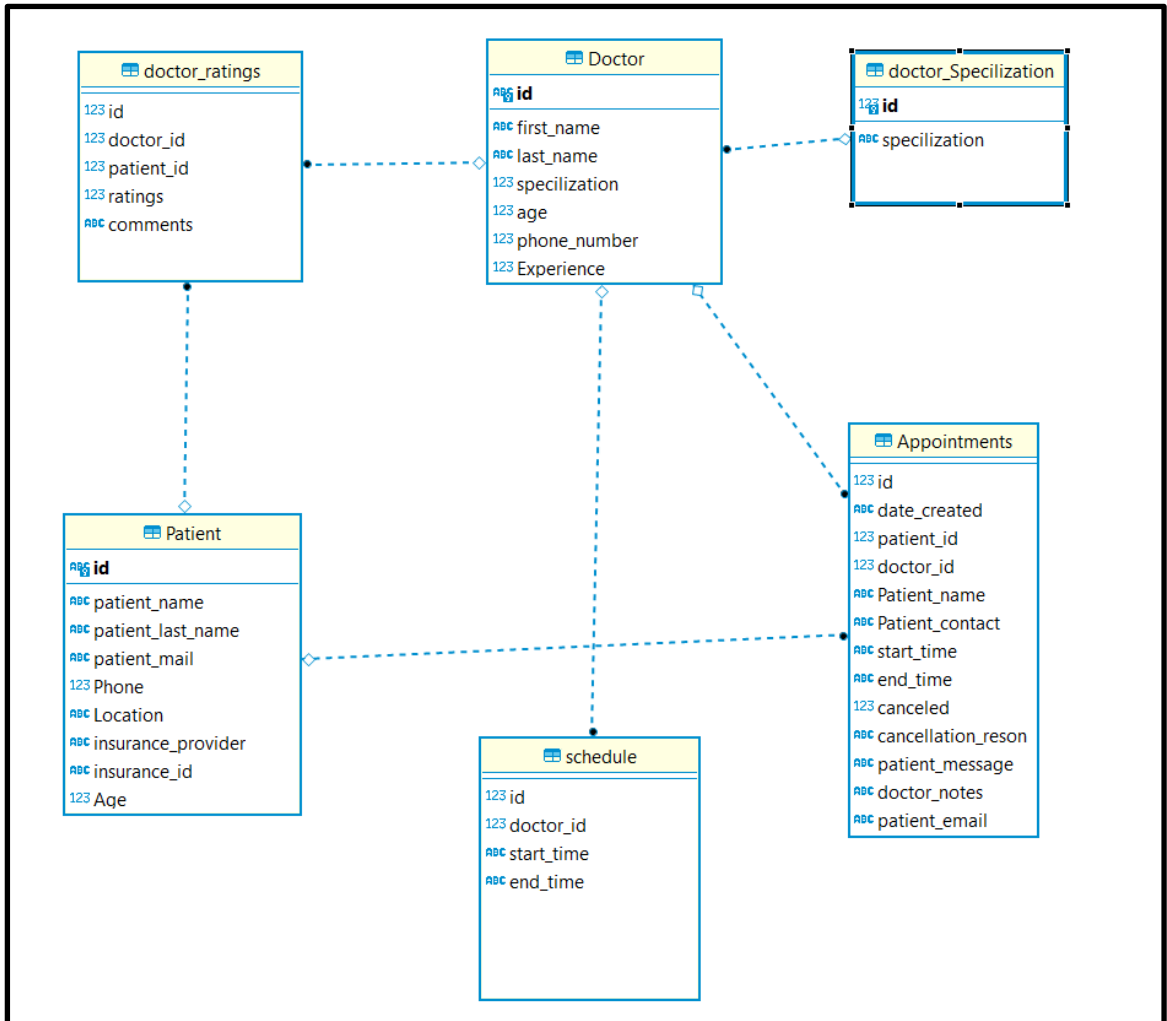


Fig: 5

## 4.3 System interface and connectivity design

The below diagram shows the System Interface diagram of the Patient Management System.

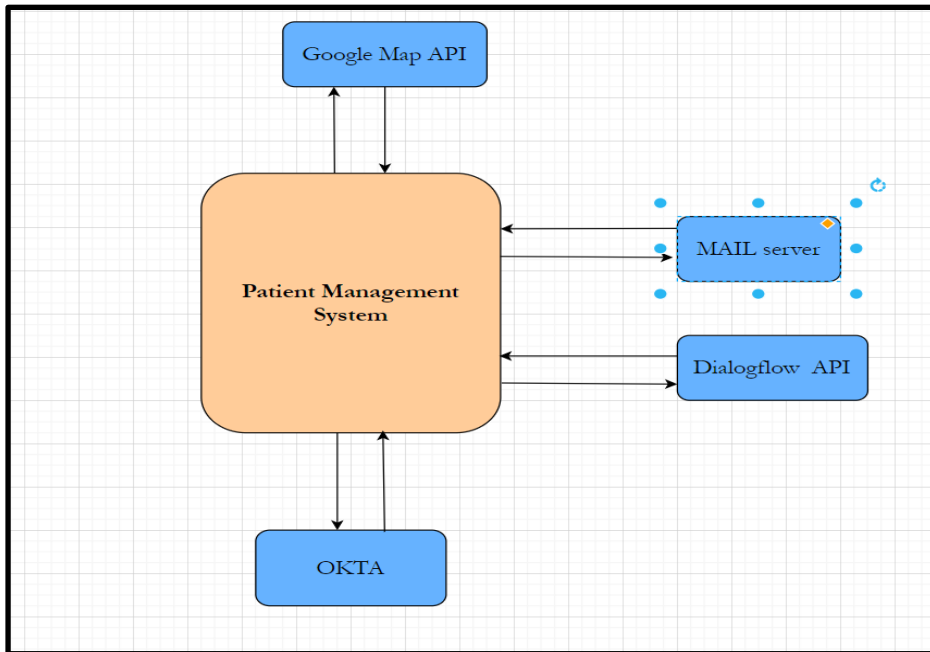


Fig: 6

### System user interface design

The Use Case diagram shown below presents the preliminary system-and-user interface design for Patient Management System.

### Use Case Diagram: -

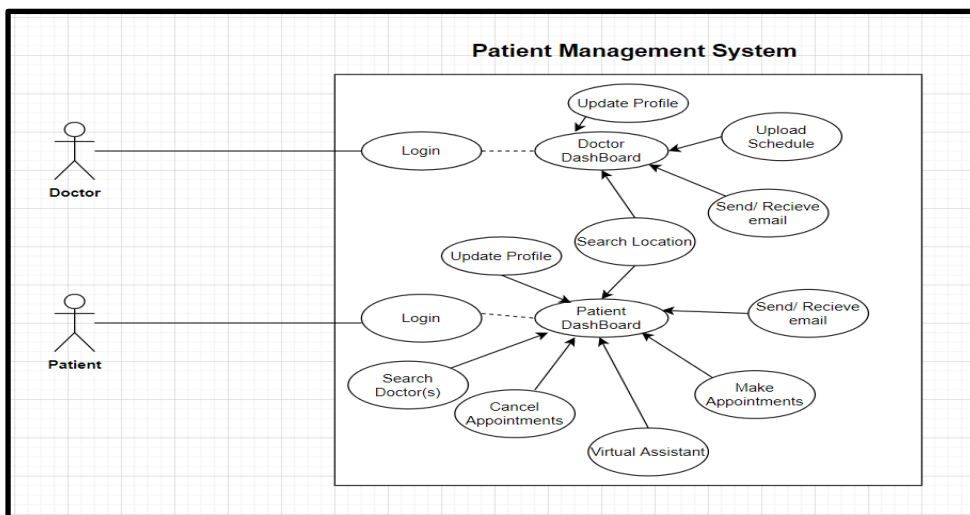


Fig: 7

### Authorization Flow: -

With OKTA as a cloud application security, it supports the patient management system by registering new users, provides a secure connection to users, integrates different profiles, and provides automatic authentication. The authorization flow is as shown below:

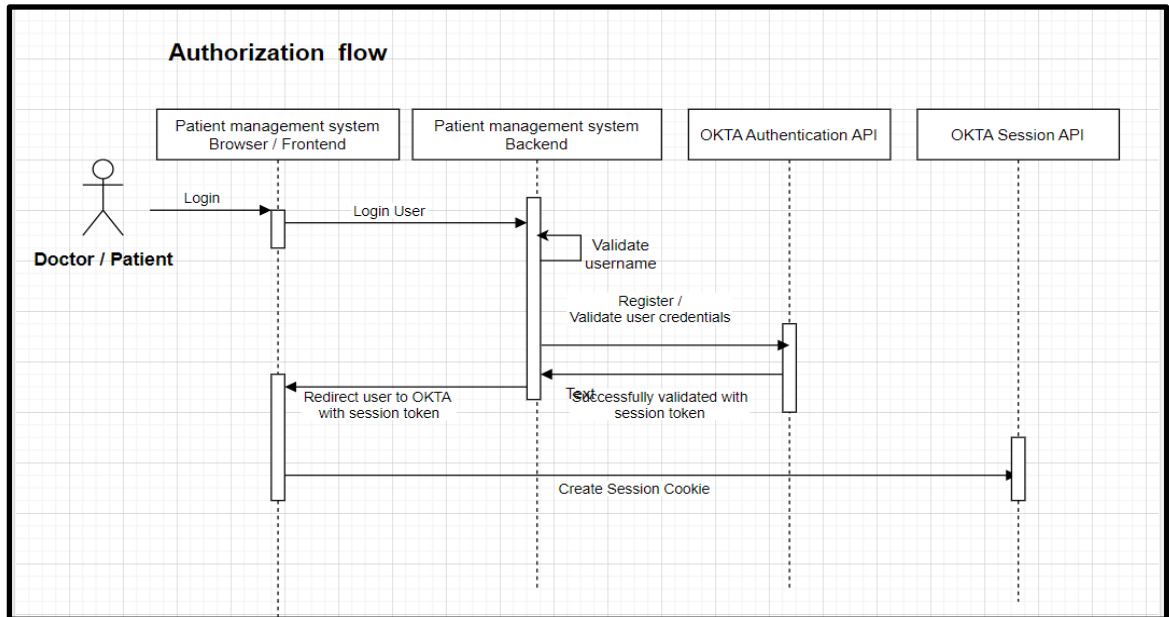


Fig: 8



## Doctor Flow:

The Doctor application is solely built for the Doctor profile. The interaction between different components of the system and Doctor is shown as below: -

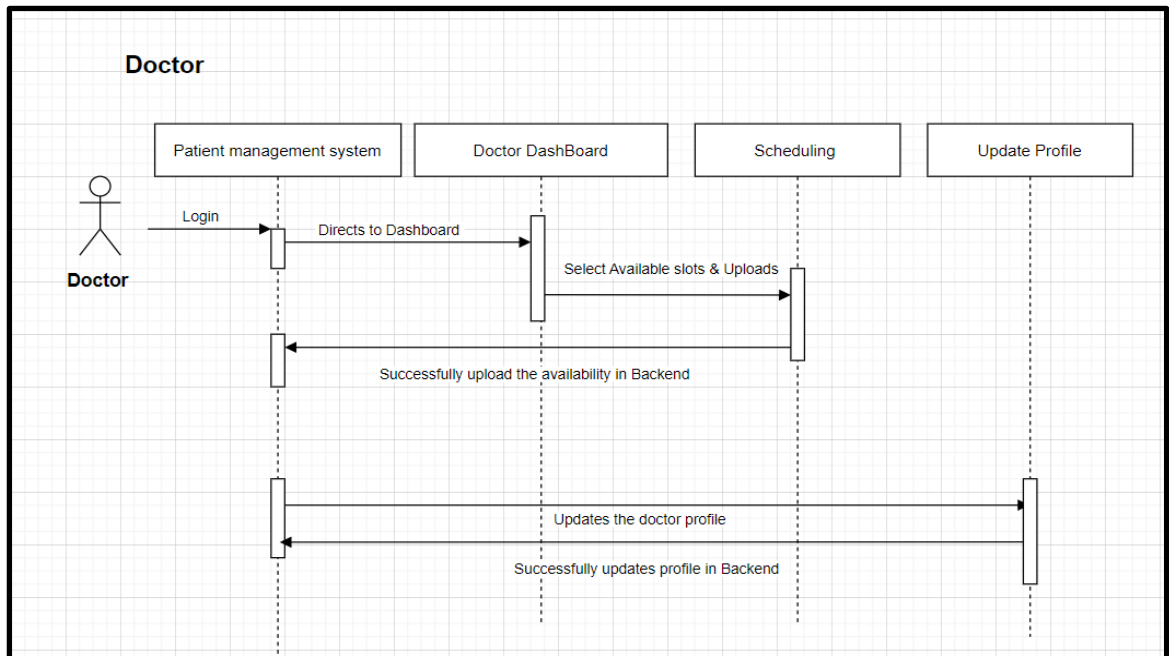


Fig: 9

## Patient Flow:

The Patient application is solely built for the Patient profile. The interaction between different components of the system and the Patient is shown as below: -

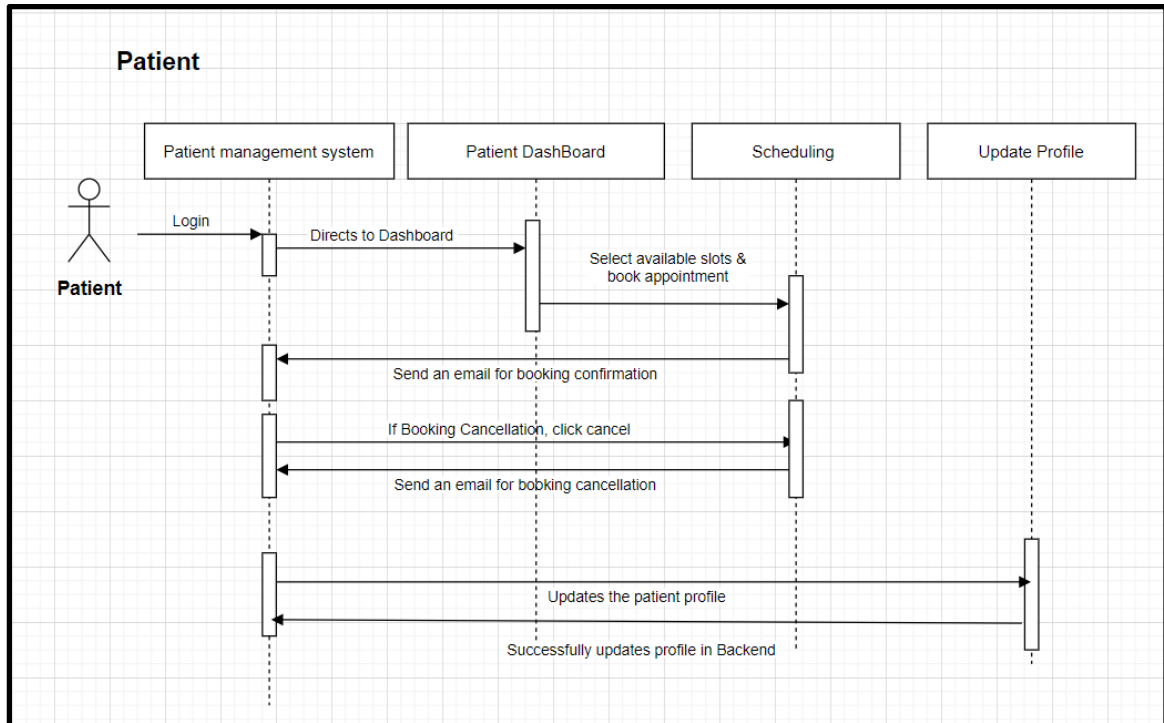


Fig: 10

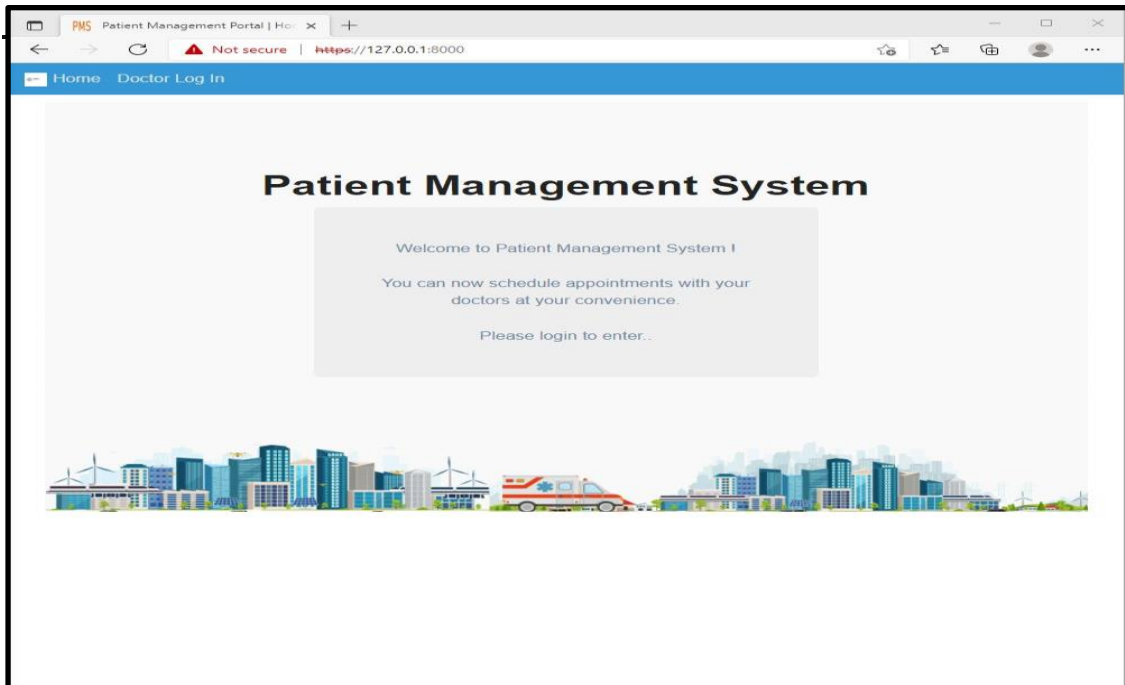
## 4.4 System component API and logic design *(for software project only)*

We have two applications designed as mentioned below:

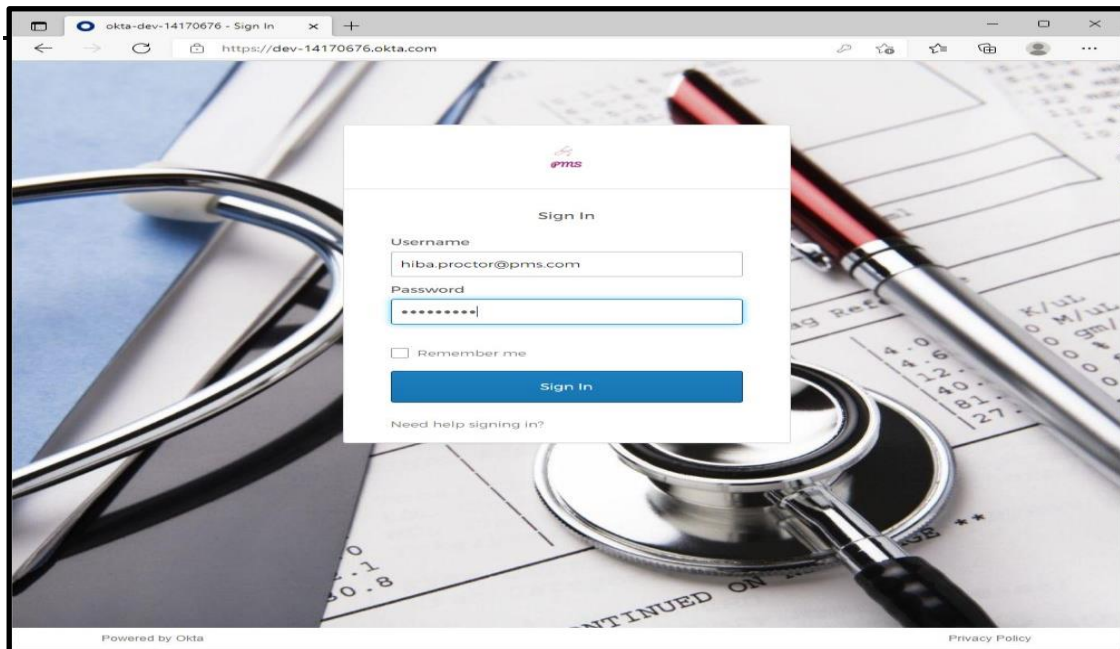
### Doctor Application

The Doctor application is solely built for the Doctor profile. It consists of a Doctor Dashboard where the Doctor can view all the upcoming appointments. It also has a feature for uploading all available time slots for appointments at once. In case of updating the profile, the application gives a platform to users to update the profile.

**Doctor Main Page:** This is the main page after launching the application.



**Doctor Sign-in Page via OKTA:** - The registration process for each new Doctor is carried out by OKTA.



**Doctor Dashboard Page:** - The Landing page after the Doctor logs into his account.

[Home](#)
[Dashboard](#)
[Profile](#)
[Schedule Appointments](#)
[Contact Us](#)
[Logout](#)

## Dashboard

Welcome to the PMS dashboard, Hiba!

**Your Current Appointments**

#	Patient	Patient message	Appointment Date/Time	Booking Date/Time	Appointment Status	Cancel Appointment
1	User1	General checkup	2021-12-10 14:00:00	2021-12-09 13:46:11	upcoming	<a href="#">CANCEL APPOINTMENT</a>
2	Poojashree	General checkup	2021-12-10 11:00:00	2021-12-09 13:47:02	upcoming	<a href="#">CANCEL APPOINTMENT</a>

**Cancelled Appointments**

#	Patient	Patient message	Appointment Date/Time	Booking Date/Time	Appointment Status
1	Avinash	Checkup	2021-12-05 10:00:00	2021-12-02 10:28:38	Cancelled
2	Pooja	test	2021-12-07 08:00:00	2021-12-02 16:28:05	Cancelled
3	Poojashree	General followup	2021-12-10 09:00:00	2021-12-09 11:48:39	Cancelled
4	Poojashree	Checkup	2021-12-10 10:00:00	2021-12-09 11:50:53	Cancelled

**Completed Appointments**

#	Patient	Patient message	Appointment Date/Time	Booking Date/Time	Appointment Status
1	Poojashree NS	General follow up	2021-12-04 09:00:00	2021-12-02 19:05:11	Completed

**Doctor Details Page:** This page lists out all the doctor details in the hospital.

[Home](#)
[Dashboard](#)
[Profile](#)
[View Doctor](#)
[Book Appointment!](#)
[Contact Us](#)
[Logout](#)

## Doctor Details

#	Doctor Name	Specialization	Ratings
1	Tarun Felix	Dermatology	5.0
2	Ali Barton	Oncology	4.75
3	Luci Emery	General Physician	4.0
4	Ember French	General Physician	4.0
5	Ottillie Decker	General Physician	4.0
6	Mark Steven	Cardiology	3.67
7	Kylo Hudson	Cardiology	3.5
8	Madina Barnard	Cardiology	3.5

***Doctor Schedule Upload Page:*** - This page provides an option for doctors to upload their weekly schedule at once.

The screenshot shows the 'Schedule Slots' page with a navigation bar at the top containing links: Home, Dashboard, Profile, Schedule Appointments, Contact Us, and Logout. The main heading 'Schedule Slots' is centered in a dark blue box. Below the heading, there are two tabs: 'Add Appointment slots' (active) and 'Modify/View Existing Slots'. Under the 'Add Appointment slots' tab, there is a section 'CHOOSE A DATE' with a dropdown menu showing '2021-12-11'. Below this is a section 'PICK YOUR PREFERRED TIME SLOT' with a grid of time slots: 8:00 - 9:00, 09:00 - 10:00, 10:00 - 11:00, 11:00 - 12:00, 12:00 - 13:00, 13:00 - 14:00, 14:00 - 14:00, 15:00 - 16:00, 16:00 - 17:00, and 17:00 - 18:00. At the bottom, there is a blue button labeled 'ADD SLOTS'.

The screenshot shows the 'Schedule Slots' page after a successful submission. A yellow banner at the top displays the message 'Slots added successfully'. The rest of the page, including the 'Schedule Slots' heading, tabs, date selector, time slot grid, and 'ADD SLOTS' button, remains the same as in the previous screenshot.

**Doctor Profile Page:** - This is the Doctor Profile page.

Home Dashboard Profile Schedule Appointments Contact Us Logout

## Hiba

To Upload a different photo/avatar, please visit [gravatar.com](https://www.gravatar.com) and add your photo

Profile

First name: Hiba Last name: Proctor

Phone: None Speciality: Oncology

Age: 50 Experience: 21

Activity

Appointments Completed: 3

Cancelled Appointments: 4

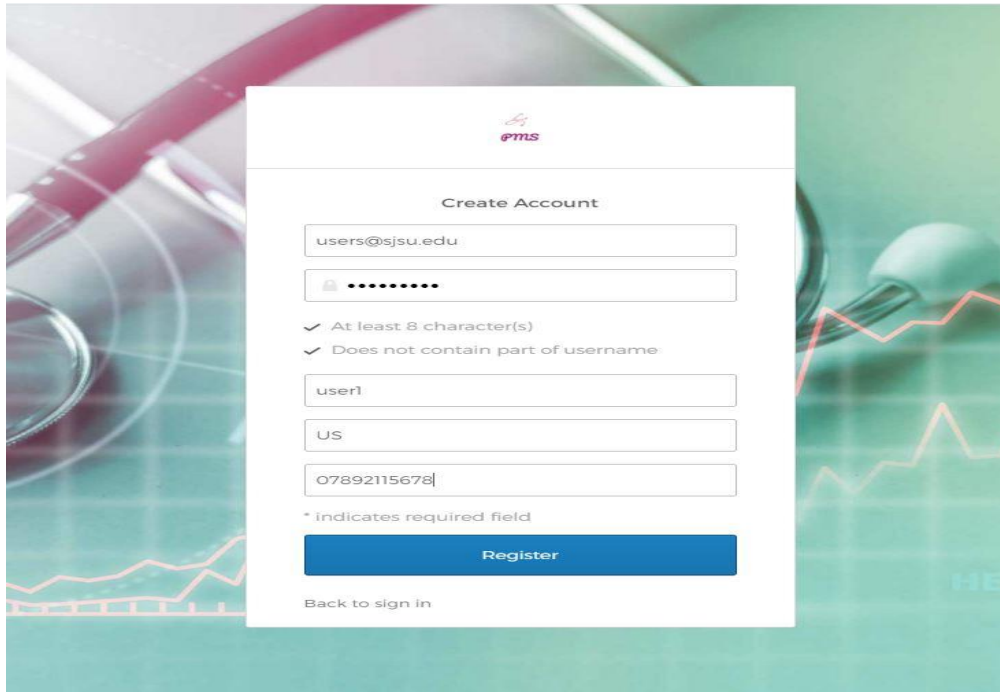
SAVE CHANGES

## Patient Application

The Patient application is solely built for the patient profile.

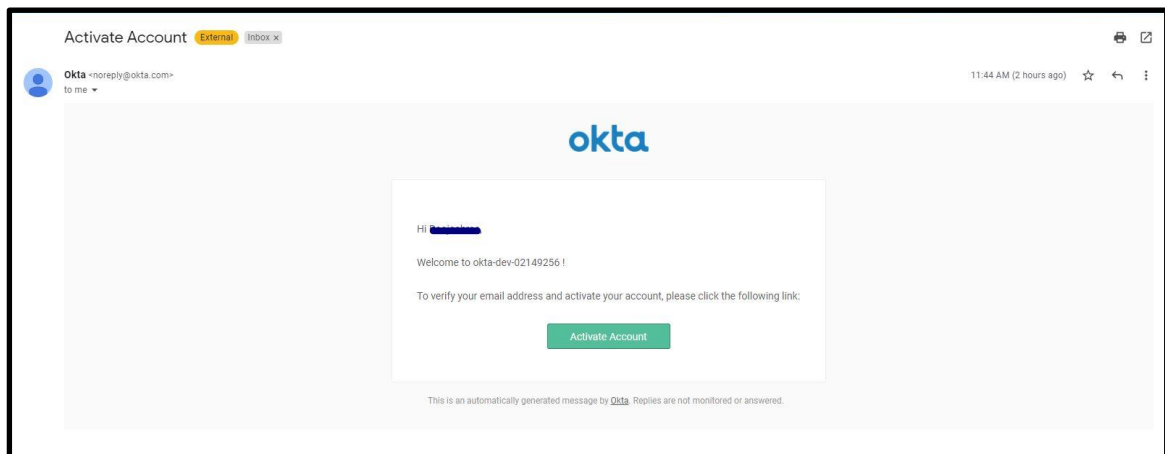
It consists of a Patient Dashboard where the patient can view all the upcoming, past, and canceled appointments. It also has a feature for reserving the appointment by selecting the time slots uploaded by the doctor. In case of cancellation, the patient can also cancel the appointment by providing a valid reason for cancellation. As for updating the profile, the application gives a platform to the patient to update the patient profile. In case the patient does not make an appointment or needs immediate consultation, a chatbot, a virtual assistant, is available to help the patient with his queries. In addition, the patient can also search for a doctor as needed.

**Patient Registration Page:** - The registration process for each new Patient is carried out by OKTA.



The image shows a web form titled "Create Account" for PTMS. The form is set against a background of a stethoscope and a green grid with a red line graph. The form fields include: a username field with "users@sjsu.edu", a password field with masked characters, two checkboxes for password requirements ("At least 8 character(s)" and "Does not contain part of username"), a first name field with "user1", a last name field with "US", and a phone number field with "07892115678". A blue "Register" button is at the bottom, with a "Back to sign in" link below it. A note states "\* indicates required field".

**OKTA registration mail:** The mail sent by OKTA to verify the patient.



**Patient Dashboard Page:** -The Landing page after the Patient logs into his account.

[Home](#) [Dashboard](#) [Profile](#) [View Doctor](#) [Book Appointment!](#) [Contact Us](#) [Logout](#)

## Dashboard

Welcome to the PMS dashboard, Poojashree!

### Your Current Appointments

#	Doctor	Specialization	Appointment Date/Time	Booking Date/Time	Appointment Status	Cancel Appointment
1	Hiba Proctor	Oncology	2021-12-10 10:00:00	2021-12-09 11:50:53	upcoming	<a href="#">CANCEL APPOINTMENT</a>

### Cancelled Appointments

#	Doctor	Specialization	Appointment Date/Time	Booking Date/Time	Appointment Status	Cancellation reason if Any
1	Hiba Proctor	Oncology	2021-12-10 09:00:00	2021-12-09 11:48:39	Cancelled	Cannot make it up

### Completed Appointments


#	Doctor	Specialization	Appointment Date/Time	Booking Date/Time	Appointment Status
---	--------	----------------	-----------------------	-------------------	--------------------

**Patient Booking Appointment Page:** - The page where the patient books the appointment with the desired doctor.

[Home](#) [Dashboard](#) [Profile](#) [View Doctor](#) [Book Appointment!](#) [Contact Us](#) [Logout](#)

Book your Seat

## Appointment

  
Call for an Emergency Service!  
**+408** ☆☆☆☆☆

### Book an appointment

Oncology

2021-12-10

User1

user1@pms.com

Hiba Proctor

14:00:00

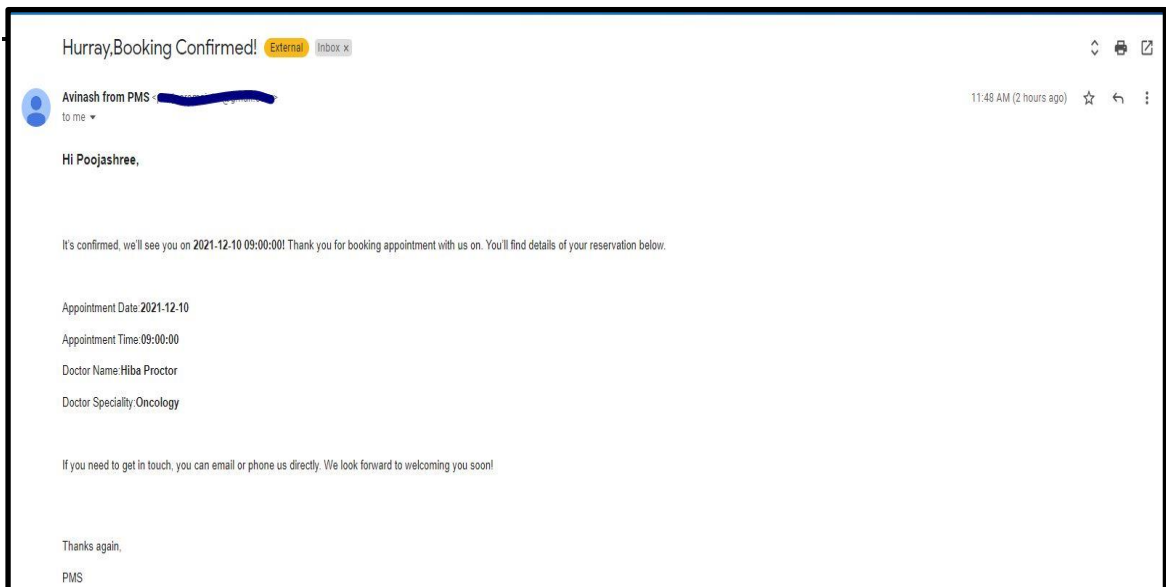
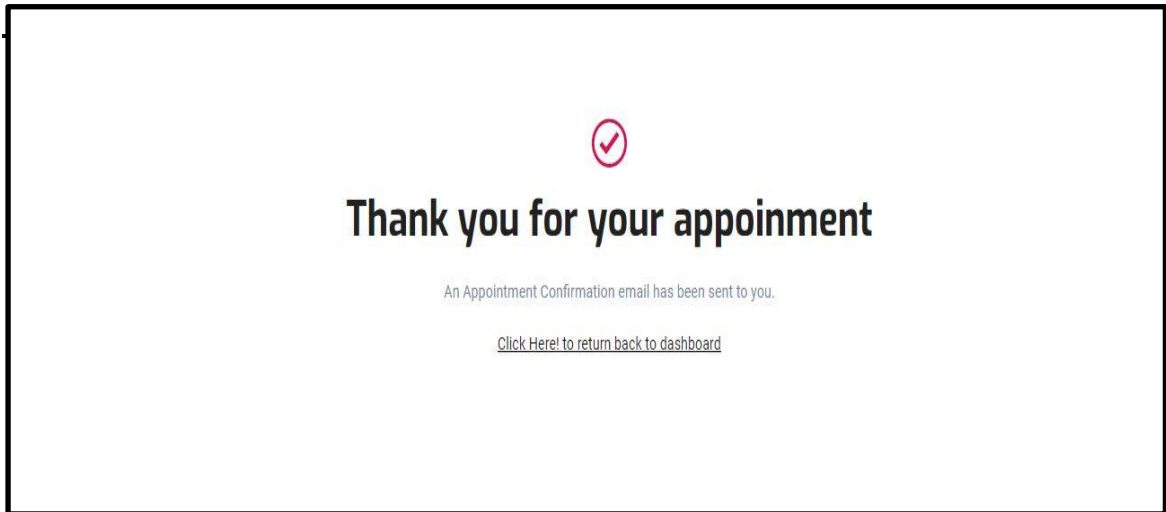
12345678

General checkup

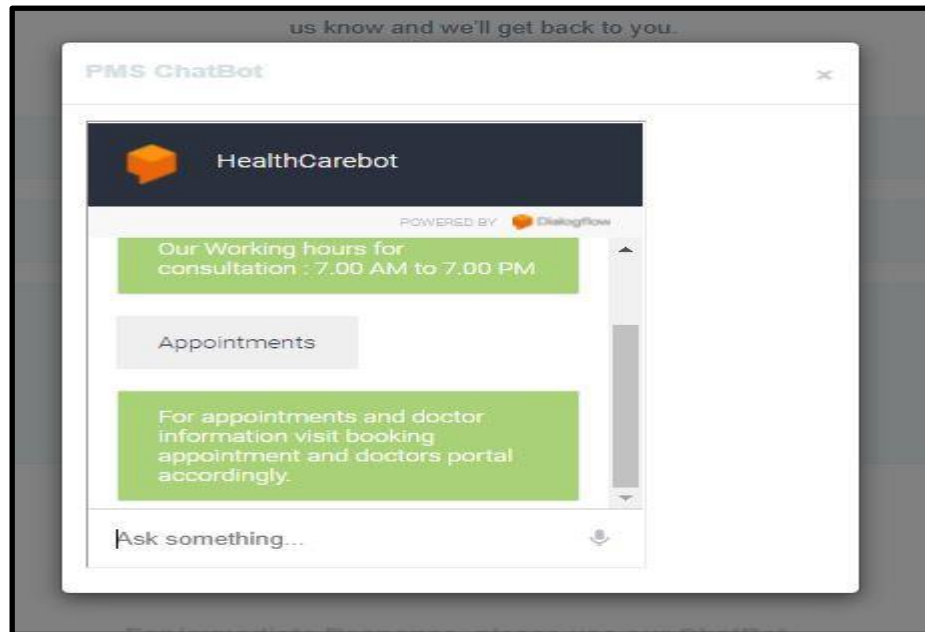
[MAKE APPOINTMENT](#)



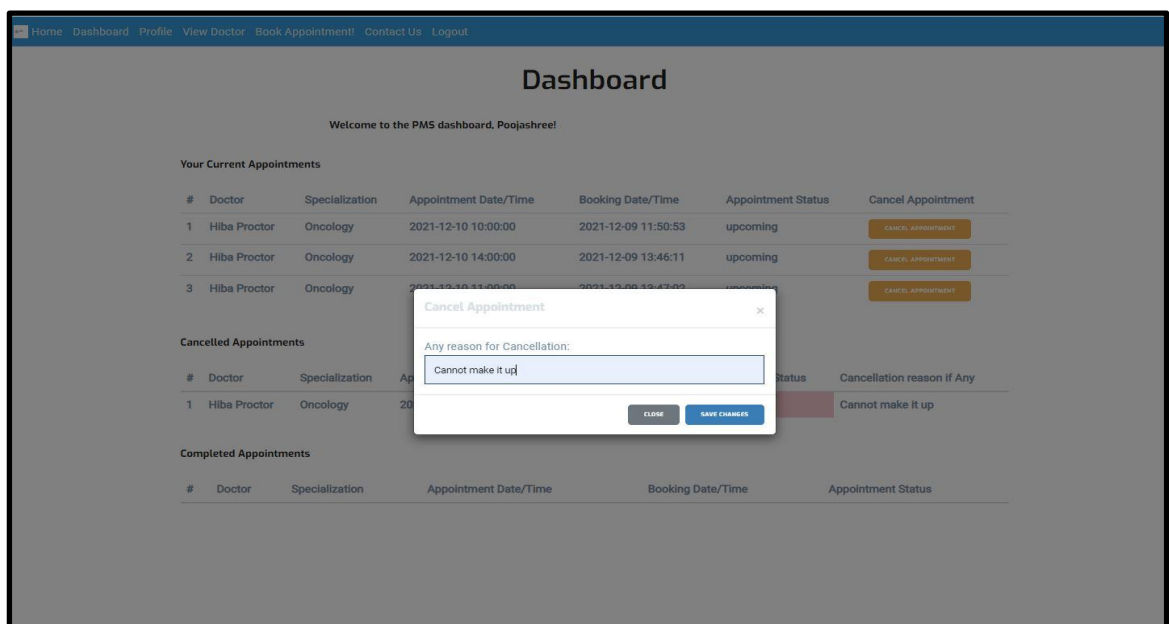
**Patient Booking Confirmation Page:** -\_The page and the mail the patient sees once the appointment is successfully booked.



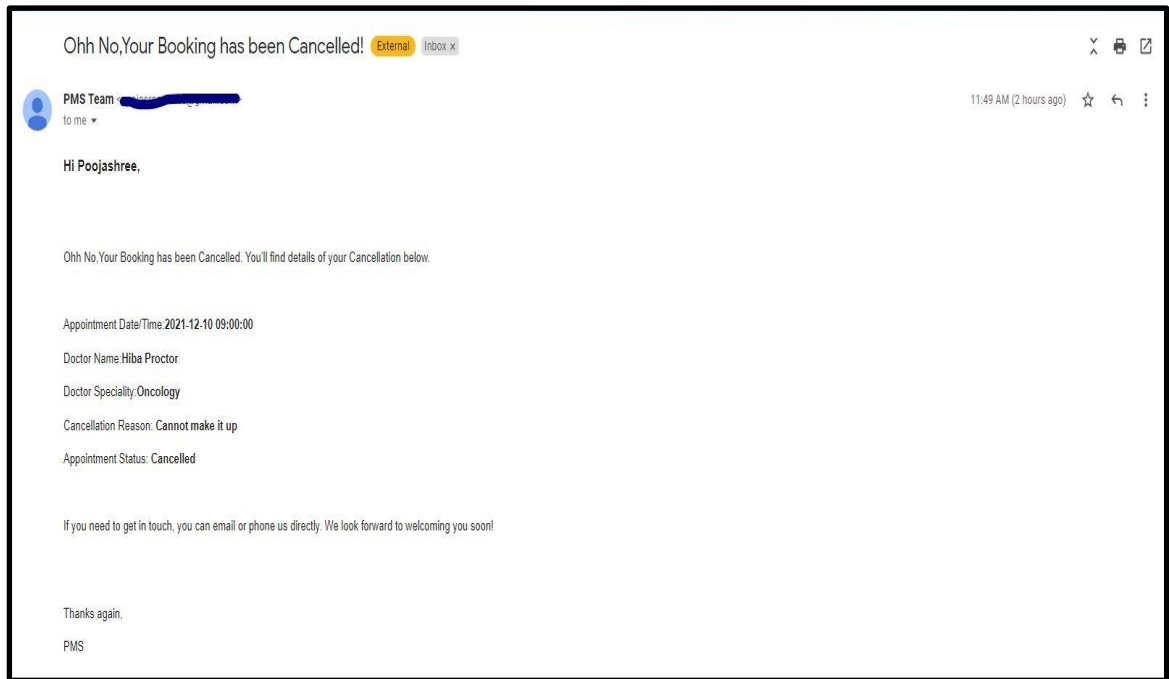
**Patient Interaction with Chatbot:** - This page shows the patient interaction with Chatbot.



**Patient Booking Cancellation Page:** - This page is the Booking Cancellation page for patients to cancel the appointment with some valid reason.



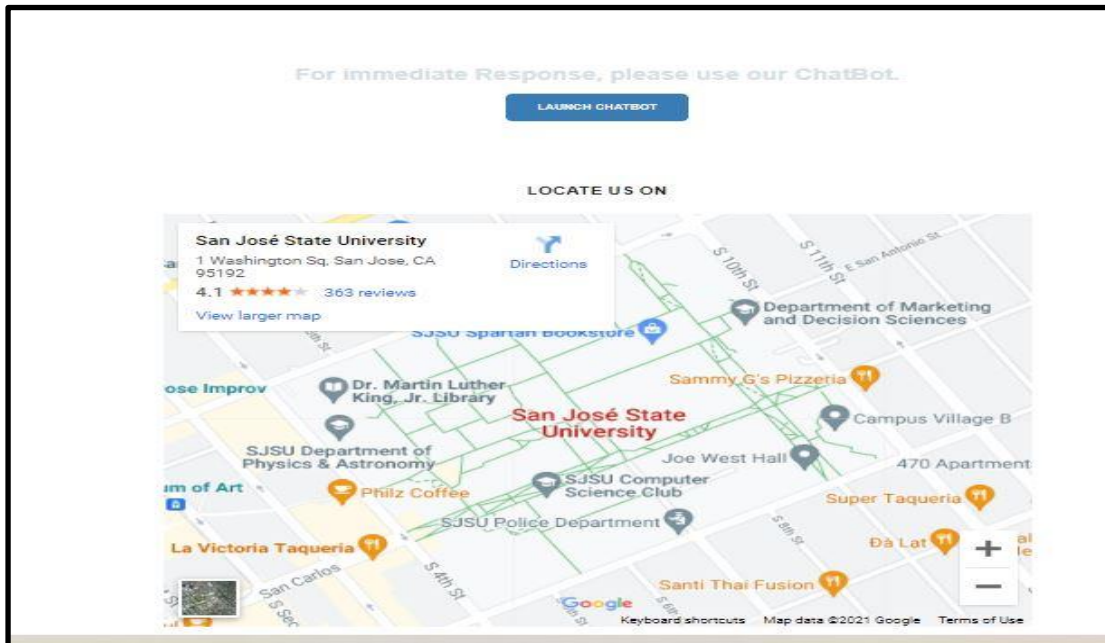
The cancellation mail is shown as below:




**Patient Profile Page:** - This is the Patient Profile page.

The screenshot shows a web application interface for a patient profile. The top navigation bar includes links: Home, Dashboard, Profile, View Doctor, Book Appointment!, Contact Us, and Logout. The main heading is "Poojashree" with a purple geometric profile picture. Below the name is a section titled "Profile" with a sub-section "Activity" showing "Appointments Completed: 1" and "Cancelled Appointments: 1". The main profile form has fields for: First name (User1), Last name (user), Phone (123456789), Age (None), Email (poojashree.ns@sjsu.edu), Location (None), Insurance ID (None), and Insurance Provider (None). A green "SAVE CHANGES" button is at the bottom.


**Maps:** This is the place the hospital is located. Just for our project, we have provided the address of SJSU.




**Contact us page:** This is the contact page for the hospital.



Call Us  
(408)-214-6720



Email Us  
pms\_support@pms.com



Location  
1 Washington Sq, San José, CA 95192

### Contact us

Have questions about our products, support services, or anything else? Let us know and we'll get back to you.

**Source Code link - [https://github.com/poojashreeNS/CMPE272\\_PMS/tree/master](https://github.com/poojashreeNS/CMPE272_PMS/tree/master)**

#### **4.5 Design problems, solutions, and patterns**

With the last-minute exit of a team member, the team experienced a few hiccups and hence had to let go of a few project requirements and adhere to the basic functionality of the application – Scheduling Appointments.

The free developer OKTA account had minimum daily usage per application. This hindered our application to include both doctor and patient under one application. Hence two applications were created for Doctor and Patient profiles respectively. This reduced usage warnings for okta and hence could bring up the application without any issues.

Since we chose developer OKTA as SSO for this application, the deployment of the Patient Management System on Heroku was quite tricky, and saw some unforeseen issues during deployment. Hence the deployment was done locally.

## **Chapter 5 System Implementation**

### **5.1. System implementation summary**

We have used Python Flask web framework for backend development, HTML, CSS, JavaScript for front-end development, and SQLite as the database to achieve these features. We also have the following application integrated into our system:

- OAuth/Okta for single sign-on.
- Chatbot for virtual assistance.
- Google Mail to send and receive emails from different roles.
- Google Maps to search the location.
- Selenium for test case automation.

Due to OKTA integration issues, while deploying on Heroku, we had to deploy the application locally.

### **5.2. System implementation issues and resolutions**

#### **Deployment on Heroku:**

We attempted to deploy it on Heroku and however the deployment of our application using OKTA caused unknown issues for the developer. Hence local deployment was proposed.

### **5.3. Used technologies and tools**

Since Python is highly flexible and extensible, has a vast collection of libraries to cater to our every need and also makes web development easy to work with and also all the developers are new to python. Hence, we chose to use the Python Flask web framework for backend development.

HTML, CSS, JavaScript was used for frontend development, and SQLite as the database to achieve these features.

We also have the following application integrated into our system:

- OAuth/Okta for single sign-on.
- Chatbot for virtual assistance.
- Google Mail to send and receive emails from different roles.
- Google Maps to search the location.

## Chapter 6 System Testing and Experiment

### 6.1 Testing and experiment scope

Since our application endpoint is the web, we tested various functionalities as mentioned below: -

**Functional Testing:** Testing the core functionality of both doctor and patient portals. Performing unit testing on separate modules and then performing integration testing.

**Usability Testing:** It is based on how the user interacts with our application. Performing multiple tests on our application with user interaction to test for bugs.

**Interface Testing:** Interface testing ensures that all interactions between the web server and application server interfaces go as planned. This includes inspecting communication processes and ensuring that error messages are displayed correctly.

**DB Testing:** Perform database testing to ensure that the schema and data types are correct in order to avoid DML failures during real-time operations. Maintain data integrity (foreign key constraints, check constraints, and PK constraints) throughout the application flow.

**Compatibility Testing:** Checking application Browser compatibility, Device compatibility, etc

**Performance Testing:** Testing load times of application.

**API testing** - Postman for testing OKTA API.

**Security testing** - Ensure wildcards are disabled in login/registration to avoid sql injection.

### 6.2 Testing and experiment approaches

In our application, we utilize Postman to test the Okta APIs. We used Postman to create collections to extract data from Okta and test it for use in our application.



We also disabled wildcard characters in the login/registration form.

OKTA OIDC is only used for authentication and authorization. All of the other critical aspects of our application are linked to our application database.

So, when a new user registers or logs in, we use api calls with Okta and our existing database to see if the user already exists in our database.

If he already exists, we leave it alone; otherwise, we add user information to our application database.

For Database testing, we tested the integrity of the database to ensure that it does not violate any existing constraints. For example: once the user plans an appointment with the doctor, his scheduling information must be recorded into the Appointments fact database table, together with the relevant doctor id and patient id. The FK constraints Doctor Id and Patient come from the Doctor and Patient tables, respectively.

We have multiple modules with functionality identical to that described above, which we thoroughly evaluated in order to maintain the integrity of our application.

We guarantee that users can interact with the program successfully without any bugs during usability testing, such as Jinja templates working well with Flaks modules without any issues. Our application's user input/output redirection works as planned.

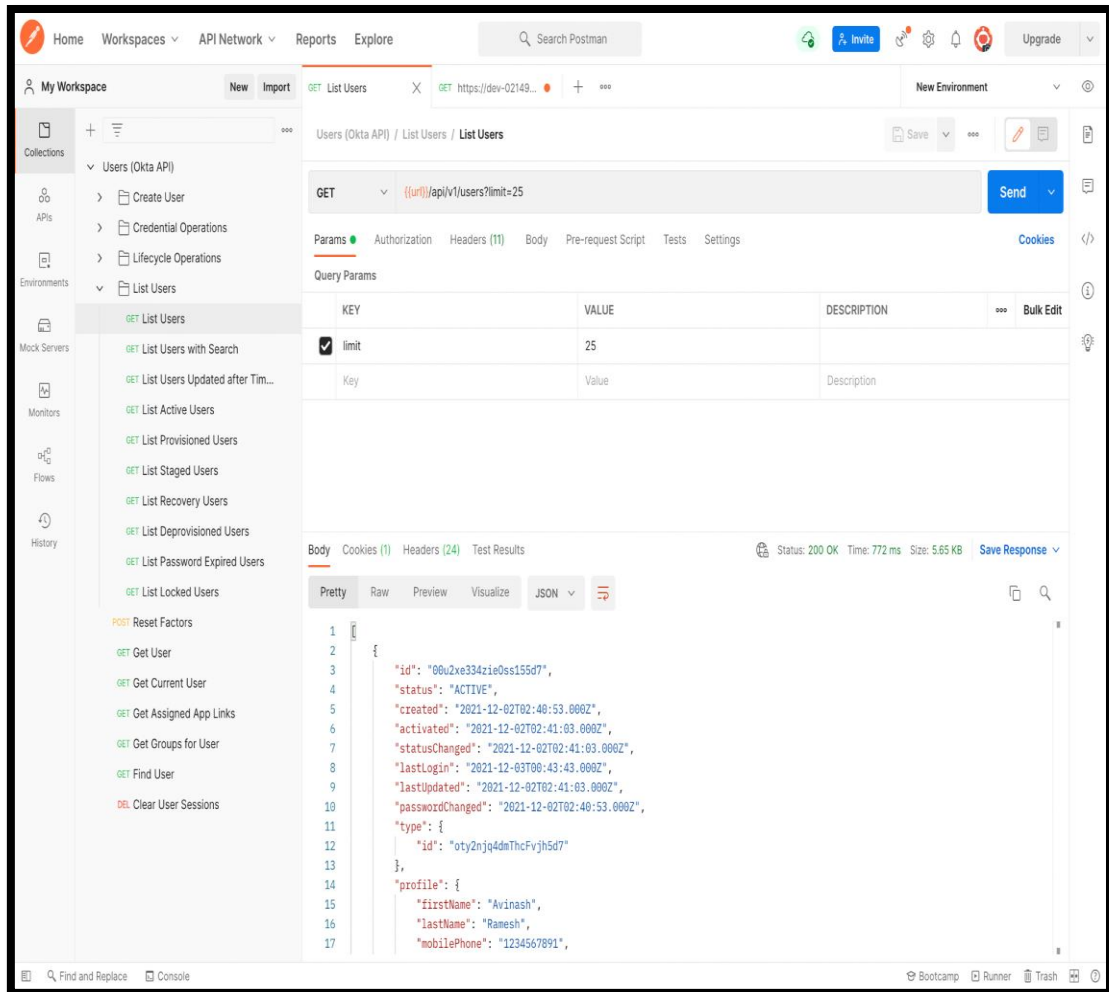
We have tested our app on a variety of browsers (Chrome, Safari, Firefox) and devices (laptop, desktop, mobile phones), and because we used bootstrap css library for most parts, everything works well.

We also put our application's email features to the test. Once a user schedules, cancels, or sends a query, he should be able to receive an email from our server with the details of his appointment/question.

We tested the login/logout functionality of our application with various criterias. If a user tries to register when he is already an existing user, prompt the user stating username/email id already exists.

## 6.3 Testing and experiment

- **Sample Postman API Testing Snippet:**



## Testing Table:

APPLICATION	MODULE	FLASK MODULE	DB READ	API	DB WRITE	UI Page	NOTES	OVERALL
DOCTOR	SCHEDULE SLOTS- ADD SLOTS	save_schedule	Yes	Yes	Yes	schedule_timings.html	Working as expected	PASS
DOCTOR	SCHEDULE SLOTS-VIEW SLOTS	display_slots	Yes	Yes	No	schedule_timings.html	Ajax working as expected	PASS
DOCTOR	SCHEDULE SLOTS-DELETE SLOTS	delete_schedule	Yes	Yes	Yes	schedule_timings.html	Working as expected	PASS
DOCTOR	CANCEL APPOINTMENTS	cancel_appointments	Yes	Yes	Yes	doctor_dashboard	Redirect to bootstrap modal and get cancel reason and perform the cancellation on submit. Working as expected	PASS
DOCTOR	DASHBOARD	doctor_dashboard	Yes	Yes 	Yes	doctor_dashboard.html	List all Current/canceled and completed appointments	PASS
DOCTOR	UPDATE PROFILE	update_profile	Yes	Yes	Yes	doctor_profile.html	Update functionality working as expected	PASS
DOCTOR	LOGIN/LOGOUT	login	OKta DB/Yes	Yes	OKta DB/Yes	Customized login/logout page created and redirect to dashboard.	Redirect after authentication and authorization from okta using open-id connect. Logout removes user sessions and tokens and directs them to the home page.	PASS
PATIENT	SCHEDULE APPOINTMENT GET Dates	appointments	Yes	Yes	No	booking.html	Only the next 7 days are shown	PASS
PATIENT	SCHEDULE APPOINTMENT Doctor list	test	Yes	Yes	No	booking.html	Dynamic dropdown using ajax and flask, working as expected	PASS
PATIENT	SCHEDULE APPOINTMENT timeslot	timeslot	Yes	Yes	No	booking.html	Dynamic dropdown using ajax and flask, working as expected and shows time slots only post current time	PASS
PATIENT	SCHEDULE APPOINTMENT Save appointment	saving_appointment	Yes	Yes	Yes	confirmation.html	Post, user submission, the details are gathered and stored in the database accordingly.	PASS
PATIENT	CANCEL APPOINTMENT	cancel_appointments	Yes	Yes	No	Redirect to Modal for cancellation reason	Redirect to cancellation modal and get cancellation reason and then perform from post-operation.	PASS
PATIENT	CANCEL APPOINTMENT	cancel_appointments	Yes	Yes	Yes	dashboard	Write to DB and direct to dashboard once cancellation is done	PASS
PATIENT	UPDATE PROFILE	update_profile	Yes	Yes	Yes	profile.html	Update functionality working as expected	PASS
PATIENT	VIEW DOCTORS	view_doctor	Yes	No	No	view_doctor.html	List all doctors from DB in tabular form	PASS
PATIENT	DASHBOARD	dashboard	Yes	Yes	Yes	dashboard.html	List all Current/canceled and completed appointments	PASS
PATIENT	LOGIN/LOGOUT	dashboard	OKta DB/Yes	Yes	OKta DB/Yes	Customized login/logout page created and redirect to dashboard.	Redirect after authentication and authorization from okta using open-id connect. Logout removes user sessions and tokens and directs them to the home page.	PASS

## **Chapter 7 Conclusion and Future Work**

### **7.1 Project summary**

The Project patient management system is to aid patients to schedule appointments with their doctor directly without running into hassles of reaching out to hospital or clinic or any scheduling assistant for that matter. The instant appointment scheduling has sped up the scheduling process. The Patient management system is thoroughly tested with the test data and therefore found to be very reliable, robust, and trustworthy. The current software takes care of the basic requirements of an average clinic and can facilitate a working environment for both doctor and patient.

Developing this current system, as a developer, got a sense of satisfaction for working on this project from scratch with minimal knowledge of python and other web frameworks. With these learnings, more features could have been incorporated into the application given there were no time constraints.

### **7.2 Future work**

The proposed system is the Patient Management system. We can enhance this system by including more facilities like adding document repositories for storing the medical records and prescriptions, pharmacy system with stocking the medical supplies into the pharmacy. Providing features to enable Doctors to add more comments like notes or summaries for each consultation visit. Adding more roles like Nurse and IT Admin can be considered in the next scope.

## References

- [1] Doccure - <https://www.doccure.io/>
- [2] Laud Amofah, <https://www.slideshare.net/Yawamofah/patient-management-system-project>
- [3] Mark Lutz, Learning Python, 5th Edition
- [4] Miguel Grinberg, Flask Web Development: Developing Web Applications with Python 2nd Edition.
- [5] Krishna Rungta, Learn SQLite in 1 Day: Definitive Guide to Learn SQLite for Beginners Kindle Edition.
- [6] Jon Duckett, HTML and CSS: Design and Build Websites 1st Edition