Project Report

On

Hospital Management System

Submitted in partial fulfillment for the award of

Post Graduate Diploma in Advanced Computing (PG-DAC) from C-DAC, ACTS (Pune)

Guided by:
Mr. Mukesh Negi
Presented by:

Mr. Divyesh Jolapara	**Prn: 200240120063**
Ms. Khushboo Joshi	**Prn: 200240120081**
Ms. Nikita Nirhali	**Prn: 200240120100**
Ms. Payal Komte	**Prn: 200240120112**
Mr. Piyush Kumar	**Prn: 200240120114**

Centre for Development of Advanced Computing (C-DAC), Pune.

ACKNOWLEDGEMENT

This project "**Hospital Management System**" was a great learning experience for us and we are submitting this work to Advanced Computing Training School (CDAC ACTS).
We are very glad to mention the name of *Mr. Mukesh Negi* for his valuable guidance to work on this project. His guidance and support helped me to overcome various obstacles and intricacies during the course of project work.

I am highly grateful to Ms. Risha P.R. (Manager (ACTS training Centre), C-DAC, for her guidance and support whenever necessary while doing this course Post Graduate Diploma in *Advanced Computing (PG-DAC)* through C-DAC ACTS, Pune.

My heartfelt thanks goes to *Ms. Shilpi Shalini* (Course Coordinator, PG-*DAC*) who gave all the required support and kind coordination to provide all the necessities to complete the project and throughout the course up to the last day here in C-DAC ACTS, Pune.

From:

Mr. Divyesh Jolapara (200240120063)
Ms. Khushboo Joshi (200240120081)
Ms. Nikita Nirhali (200240120100)
Ms. Payal Komte (200240120112)
Mr. Piyush Kumar (200240120114)

TABLE OF CONTENTS

ABSTRACT

The primary purpose of this project is to automate the process of managing the appointments between doctors and patients and providing means for patients to view his/her medical history.

It deals with the collection of patient's information, appointment booking, upcoming appointment, appointment history, prescriptions, hospital related information, diagnosis details, etc. Traditionally, it was done manually. The main function of the system is to register and store patient details and retrieve these details as and when required, and also to manipulate these details meaningfully. System input contains patient details, appointment booking details, while system output is to get these details on to the screen. The Hospital Management System can be entered using a username and password. It is accessible by patients. Only they can add data into the database. The data can be retrieved easily. The data are well protected for personal use and makes the data processing very fast.

1. Introduction of Project:

Our project hospital management system includes registration of patients, storing their details into the system. Our software has the facility to give a unique id for every patient and stores the details of every patient. The system can be entered using an email and password.

Users can book their appointments from the patient portal after registering and login to our proposed system. They can view their previous appointments from the portal itself as well as view their prescription and reports.

The patients can book their appointments according to the available time slot of particular date and available doctors for their specific appointment reason.

The project 'Hospital Management System' is based on the database, object oriented and networking techniques. As there are many areas where we keep the records in database for which we are using MY SQL software which is one of the best and the easiest software to keep our information.

This project uses angular as frontend and Java as the backend software and has connectivity with MY SQL. The frontend is connected with backend using REST Api. Backend is connected with the database using Spring data JPA.

2. Product Overview and Summary

2.1 Purpose:

This project is aimed to automate the hospital management system. This project is developed mainly to administrate doctor's appointment with the patients. The purpose of the project entitled as HOSPITAL MANAGEMENT SYSTEM is to computerize the Management of patients and their appointments, to develop software which is user friendly, simple, fast, and cost – effective.

It deals with the collection of patient's information, diagnosis details, etc. Traditionally, it was done manually. The main function of the system is to register and store patient details and doctor details and retrieve these details as and when required, and also to manipulate these details meaningfully.

2.2 Scope:

The proposed software product is the Hospital Management system (HMS). The system will be used in any hospital, clinic, dispensary or pathology labs. Clinic, dispensary or pathology to get the information from the patients and then storing that data for future usages. The current system in use is a paper based system. It is too slow and cannot provide updated lists of patients within reasonable time frame. The intention of the system is to reduce over-time pay and increase the number of patients that can be treated accurately. Requirement statements in these documents are both functional and non-functional.

2.3. Requirement of Proposed System

The first step in system development life cycle is the identification of need of change to improve or enhance an existing system. An initial study and observation on an existing system was carried out. The present system of hospital is mostly manual. In the proposed system the primary objective is to automate the patient booking process. Along with that, the system aims at storing patient's historical data such as reports, prescriptions, etc.

2.4. Login/Authentication

For the first time, either patients or doctors needs to give following details for their registration.

Patients: name, email-Id, password, date of birth, contact number, address, blood group, gender.

Patients can login the system using email-id and password.

2.5. Assumptions

1) Our System is available to provide service 24x7.

2) One service one Doctor.

3) Services within one Hospital.

4) Payment is done at the time of booking.

.

3. Overall Description:

3.1 Product Features

The main feature of this system is easy to use and flexible to provide patients with the patient portal to book their appointments and know their all previous appointments, reports, prescriptions.

3.2 Technology Used

3.2.1 SOFTWARE REQUIREMENTS:

Software Requirements deal with defining software resource requirements and pre-requisites that need to be installed on a computer to provide optimal functioning of an application. These requirements or pre-requisites are generally not included in the software installation package and need to be installed separately before the software is installed.

SOFTWARE REQUIREMENTS FOR PRESENT PROJECT:

OPERATING SYSTEM : Windows 10
FRONT END : Html5, CSS, Angular.
SERVER SIDE SCRIPT : J2EE
DATABASE : MySQL

4. Analysis:

4.1EXISTING SYSTEM:

Hospitals currently use a manual system for the management and maintainance of critical information. The current system requires numerous paper forms, with data stores spread throughout the hospital management infrastructure. Often information is incomplete or does not follow management standards. Forms are often lost in transit between departments requiring a comprehensive auditing process to ensure that no vital information is lost. Multiple copies of the same information exist in the hospital and may lead to inconsistencies in data in various data stores.

4.2 PROPOSED SYSTEM:

The Hospital Management System is designed for any hospital to replace their existing manual paper based system. The new system is to control the information of patients. Room availability, staff and operating room schedules and patient invoices. These services are to be provided in an efficient, cost effective manner, with the goal of reducing the time and resources currently required for such tasks .

4.3 FEASIBILITY STUDY:

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are:

4.3.1 Economic Feasibility

This study is carried out to check the economic impact will have on the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customised products have to be purchased.

4.3.2 Technical Feasibility:

This study is carried out to check the technical feasibility, that is,the technical requirements of the system. Any system developed must not have a high demand on the available available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes for the implementing this system.

4.3.3 Operational Feasibility:

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

4.4 SOFTWARE SPECIFICATION

Angular:

Angular is a platform and framework for building single-page client applications using HTML and TypeScript. Angular is written in TypeScript. It implements core and optional functionality as a set of TypeScript libraries that you import into your apps.

The architecture of an Angular application relies on certain fundamental concepts. The basic building blocks of the Angular framework are Angular components that are organized into *NgModules*. NgModules collect related code into functional sets; an Angular app is defined by a set of NgModules. An app always has at least a *root module* that enables bootstrapping, and typically has many more *feature modules*.

- Components define *views*, which are sets of screen elements that Angular can choose among and modify according to your program logic and data.
- Components use *services*, which provide specific functionality not directly related to views. Service providers can be *injected* into components as *dependencies*, making your code modular, reusable, and efficient.

Modules, components and services are classes that use *decorators*. These decorators mark their type and provide metadata that tells Angular how to use them.

- The metadata for a component class associates it with a *template* that defines a view. A template combines ordinary HTML with Angular *directives* and *binding markup* that allow Angular to modify the HTML before rendering it for display.
- The metadata for a service class provides the information Angular needs to make it available to components through *dependency injection (DI)*.

An app's components typically define many views, arranged hierarchically. Angular provides the Router service to help you define navigation paths among views. The router provides sophisticated in-browser navigational capabilities.

Material UI:

Material Design (codenamed **Quantum Paper**) is a visual language that can be used to create digital experiences. It's a set of principles and guidelines across platforms and devices for interactivity, motion and components that simplify the design workflow for teams designing their product.

The Material components allow you to create professional UIs with powerful modularity, theming and customization features.

Bootstrap 4:

- Bootstrap is a free front-end framework for faster and easier web development
- Bootstrap includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many other, as well as optional JavaScript plugins
- Bootstrap also gives you the ability to easily create responsive designs

MySQL:

MySQL is developed, distributed, and supported by Oracle Corporation. MySQL is a database system used on the web it runs on a server. MySQL is ideal for both small and large applications. It is very fast, reliable, and easy to use. It supports standard SQL. MySQL can be compiled on a number of platforms.

The data in MySQL is stored in tables. A table is a collection of related data, and it consists of columns and rows. Databases are useful when storing information categorically.

CONNECTIVITY:

Clients can connect to MySQL Server using several protocols:

- Clients can connect using TCP/IP sockets on any platform.
- On Windows systems in the NT family (NT, 2000, XP, 2003, or Vista), clients can connect using named pipes if the server is started with the --enable-named-pipe option. In MySQL 4.1 and higher, Windows servers also support shared-memory connections if started with the --shared-memory option. Clients can connect through shared memory by using the --protocol=memory option.

J2EE Spring Boot Framework:

Spring Boot is an open source Java-based framework used to create a micro Service. It is developed by Pivotal Team and is used to build stand-alone and production ready spring applications. This chapter will give you an introduction to Spring Boot and familiarizes you with its basic concepts.

5.Demo Screenshots:

Figure 1:Home Page (a)

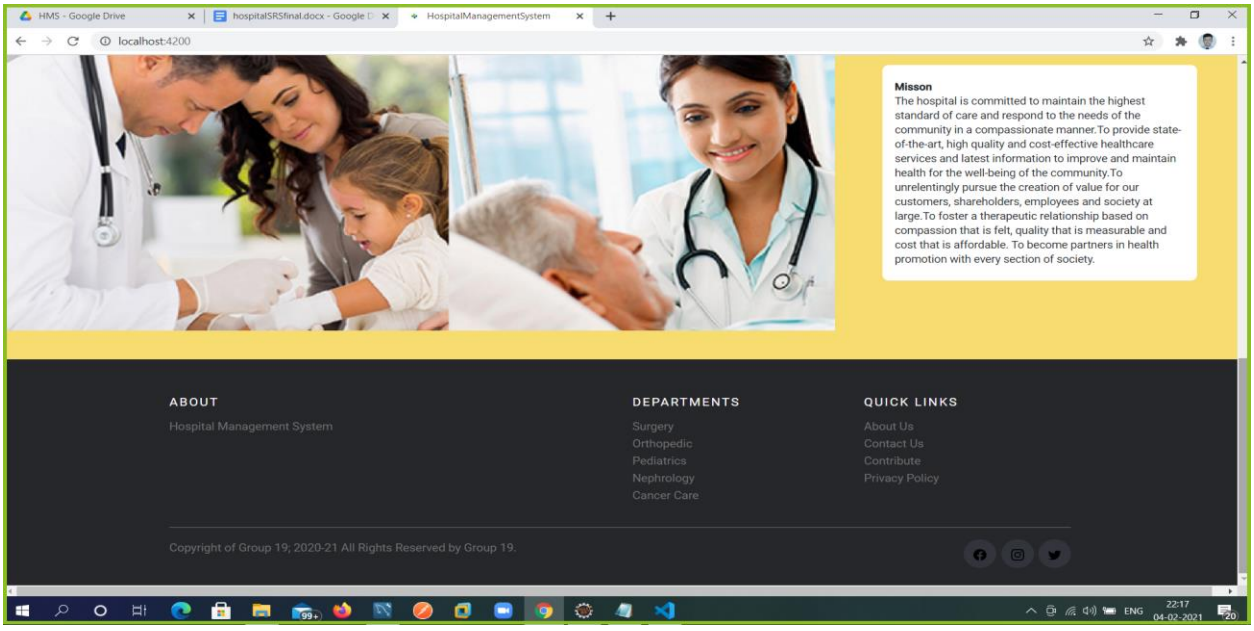


Figure 2: Home Page (b)

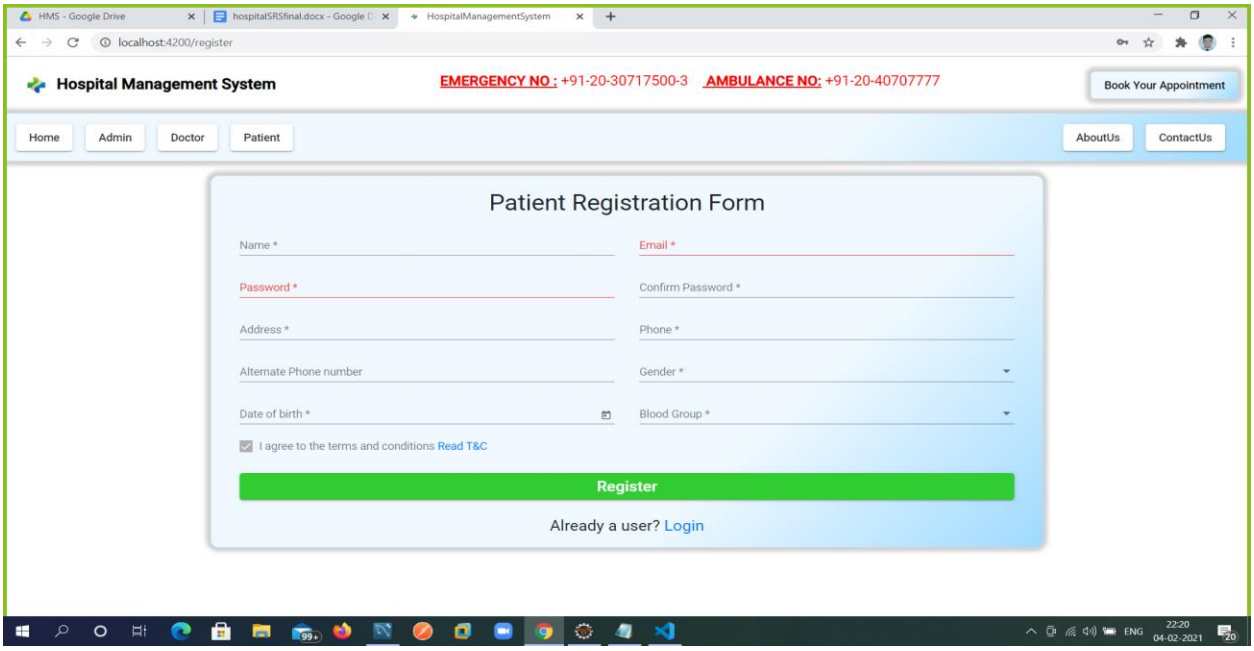


Figure 3:Patient Registration Form

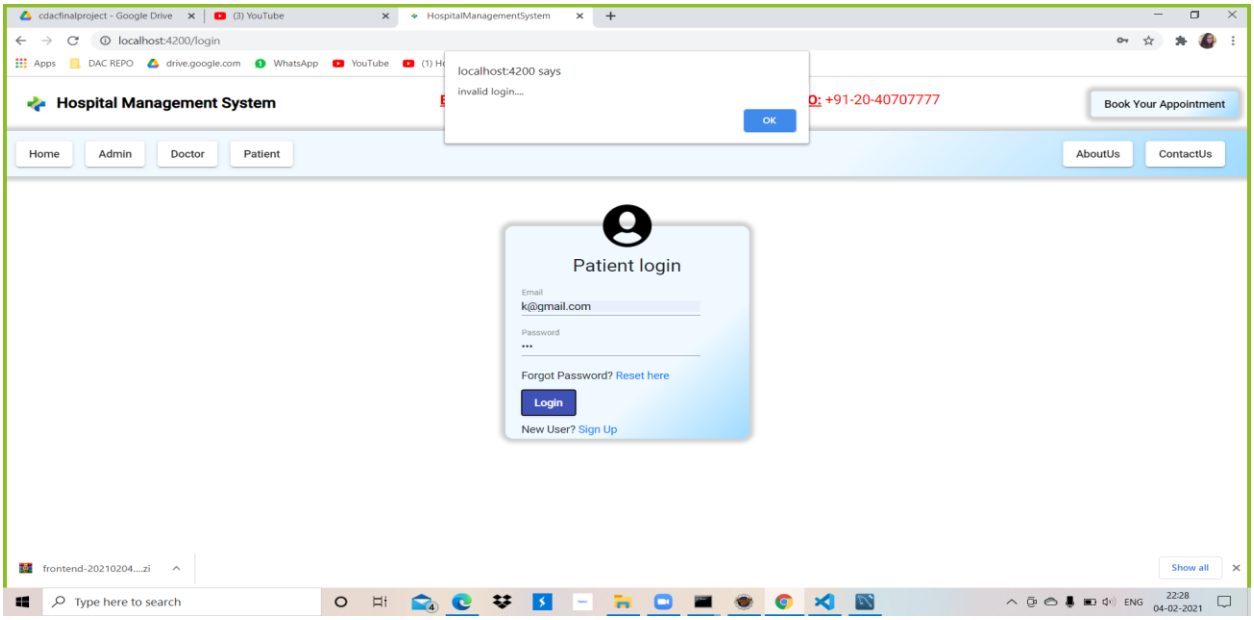


Figure 4:Patient Login Form(for invalid credential)

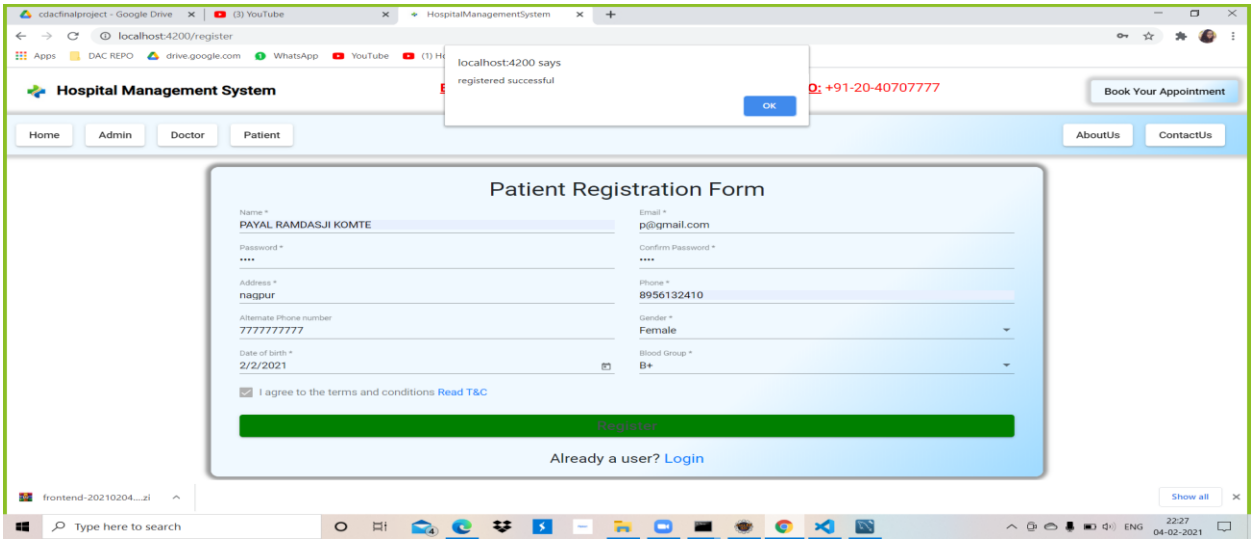


Figure 5:Patient Registration Form(for Successful login)

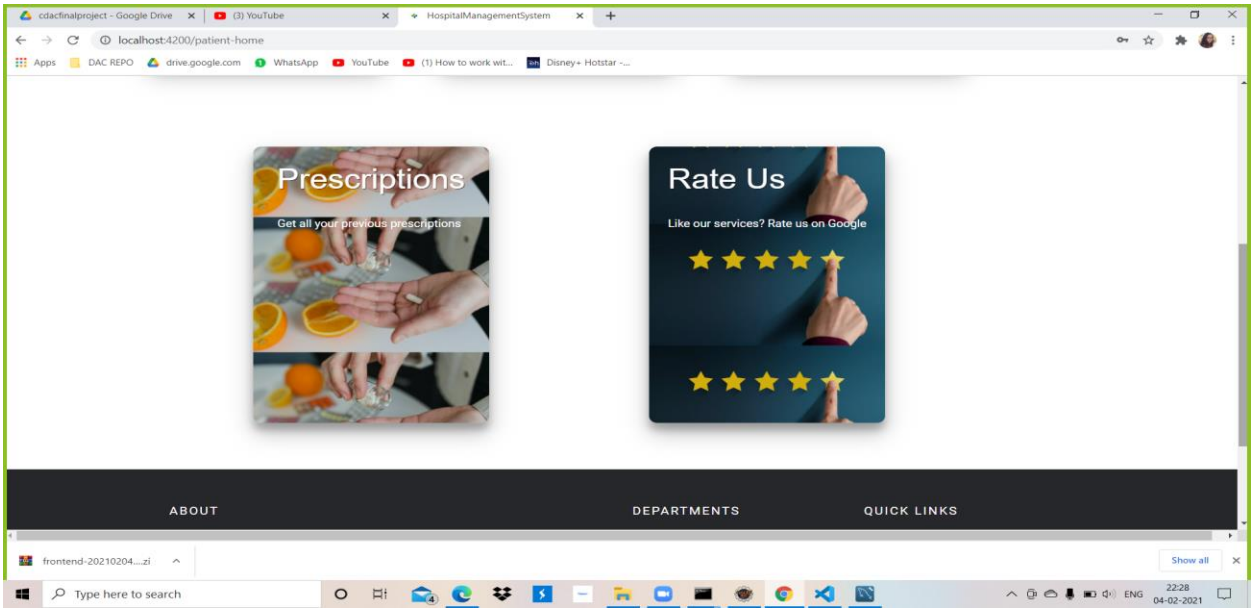


Figure 6:Patient Home Page

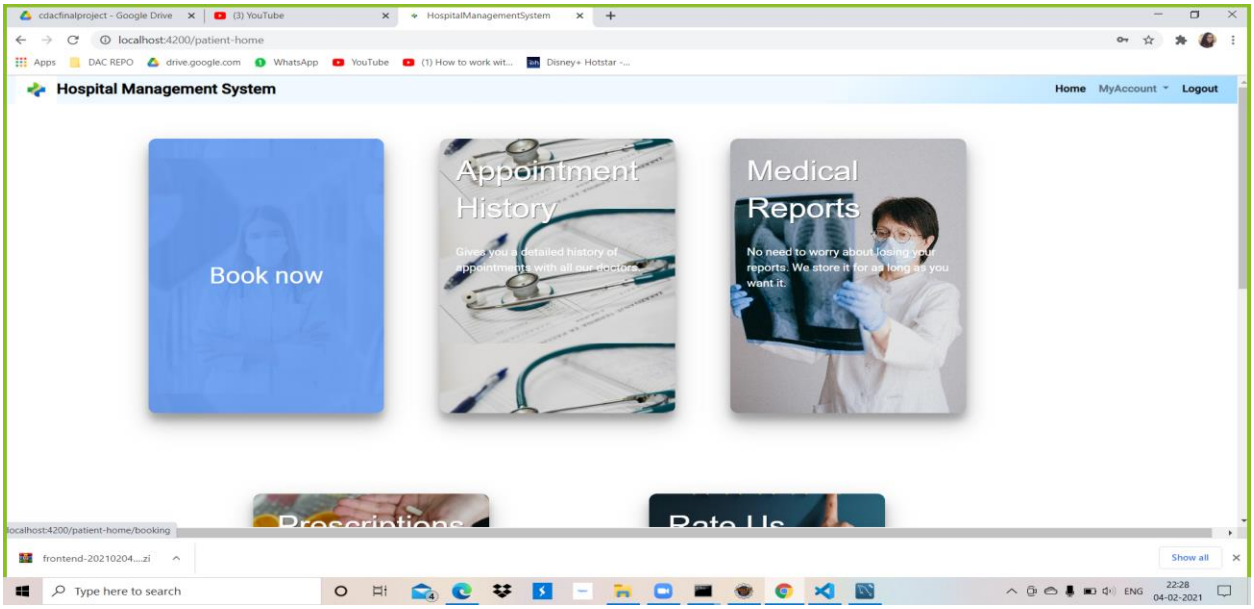


Figure 7:Patient Home Page(Cursor on booking)

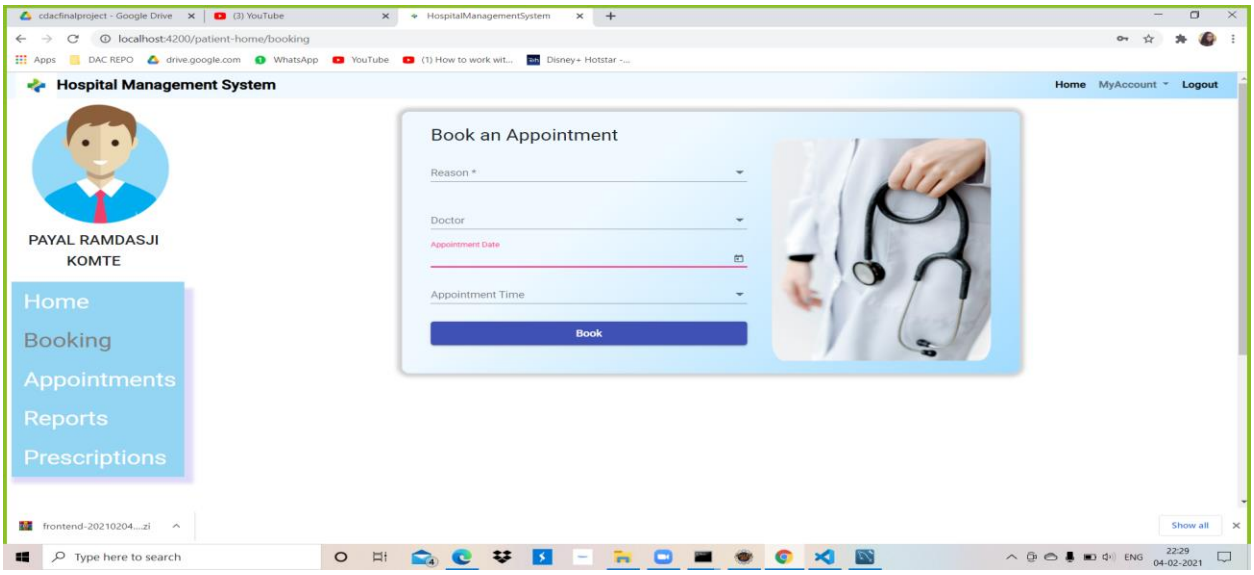


Figure 8:Appointment Booking form

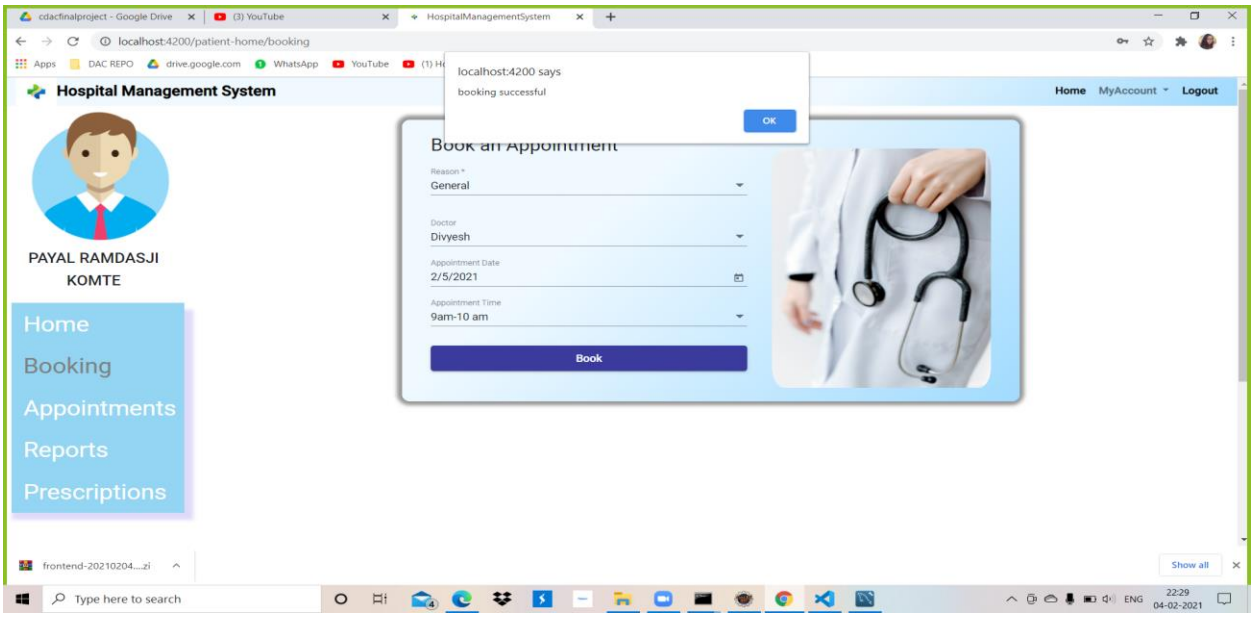


Figure 9:Booking Successful

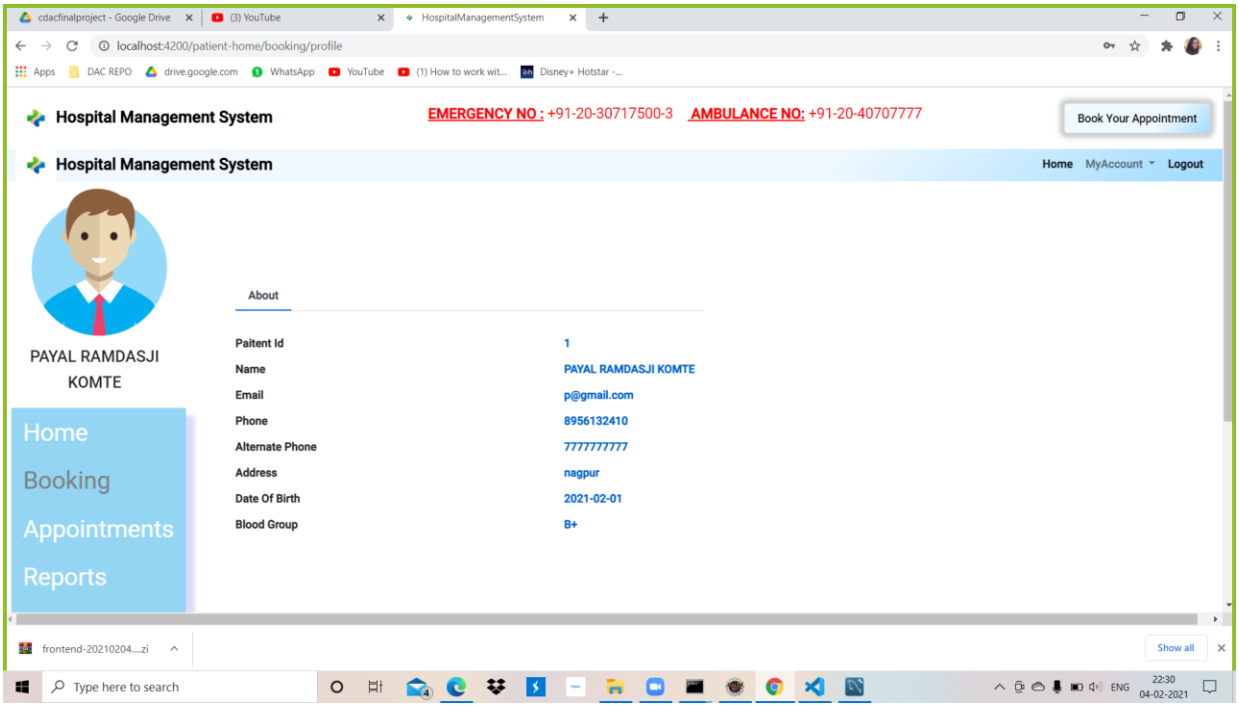


Figure 10:Patient Details

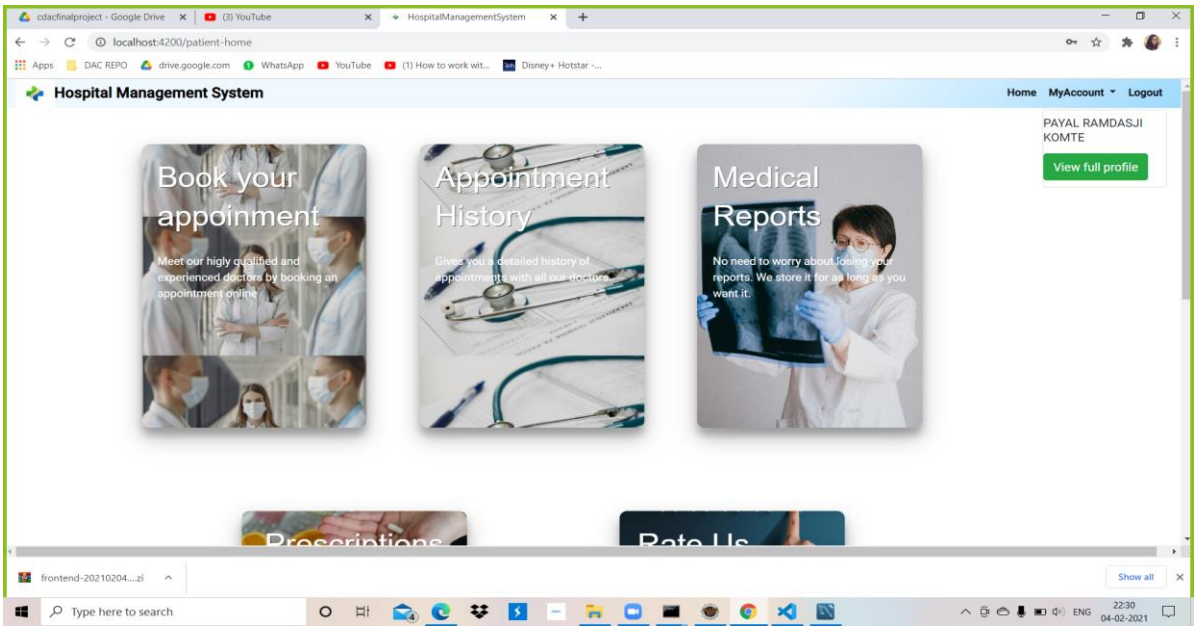


Figure 11:Button for Viewing profile

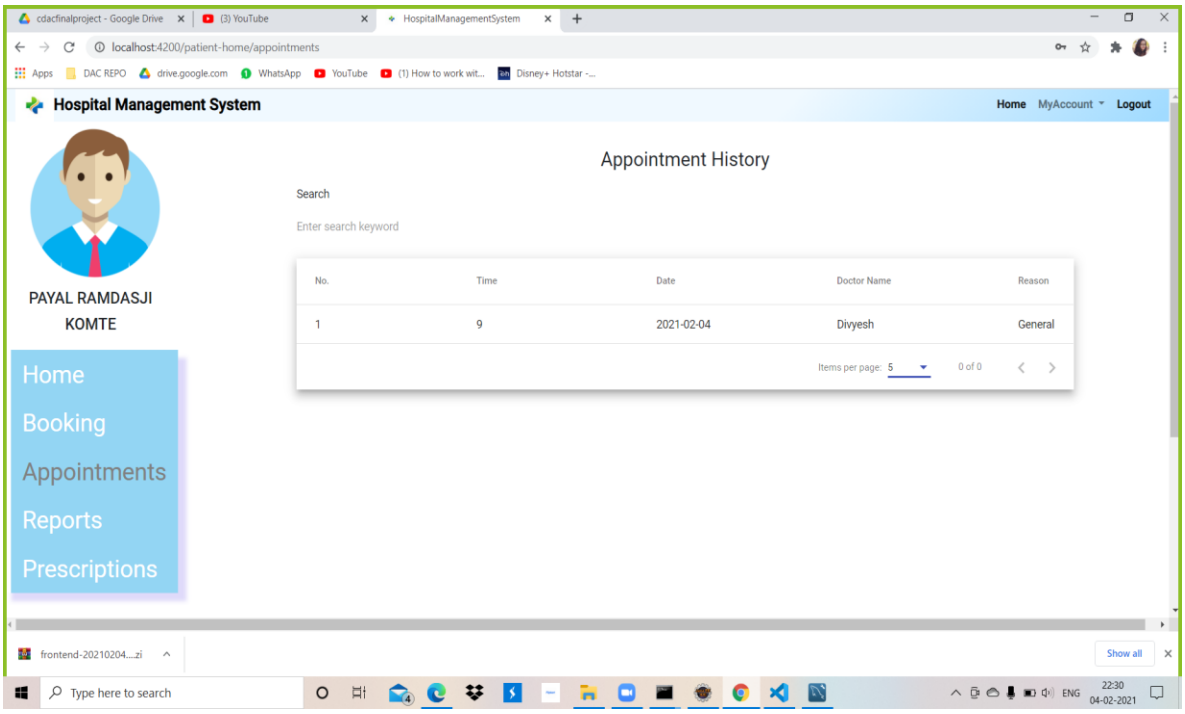


Figure 12:Appointments History

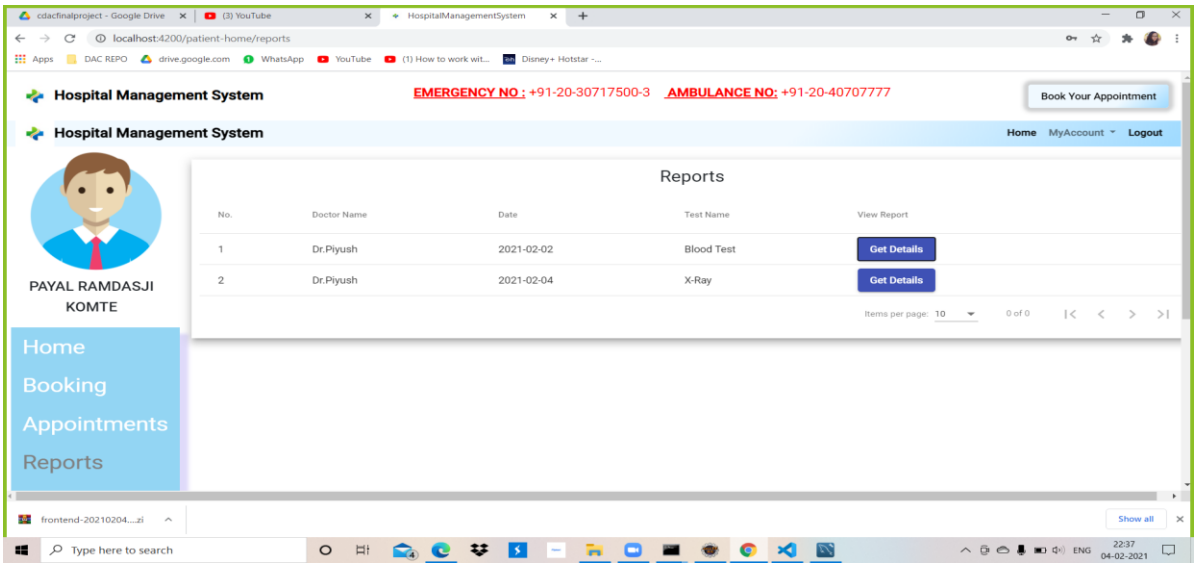


Figure 13:Patient Reports

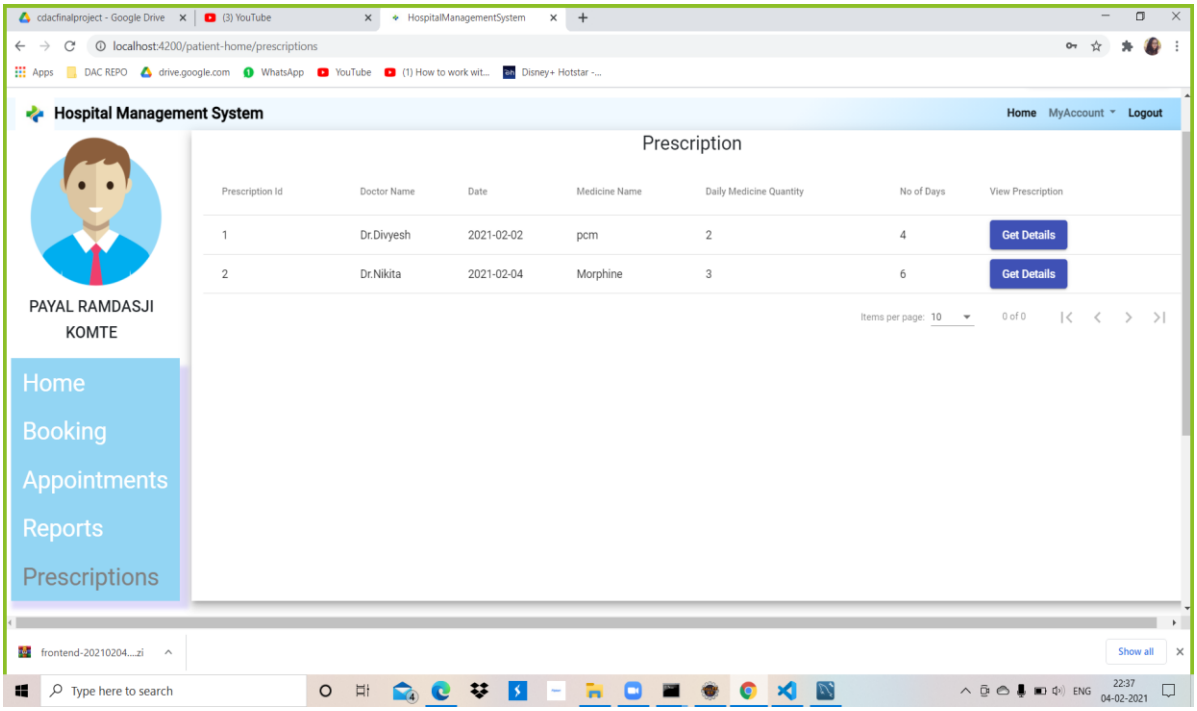


Figure 14:Patient Prescriptions

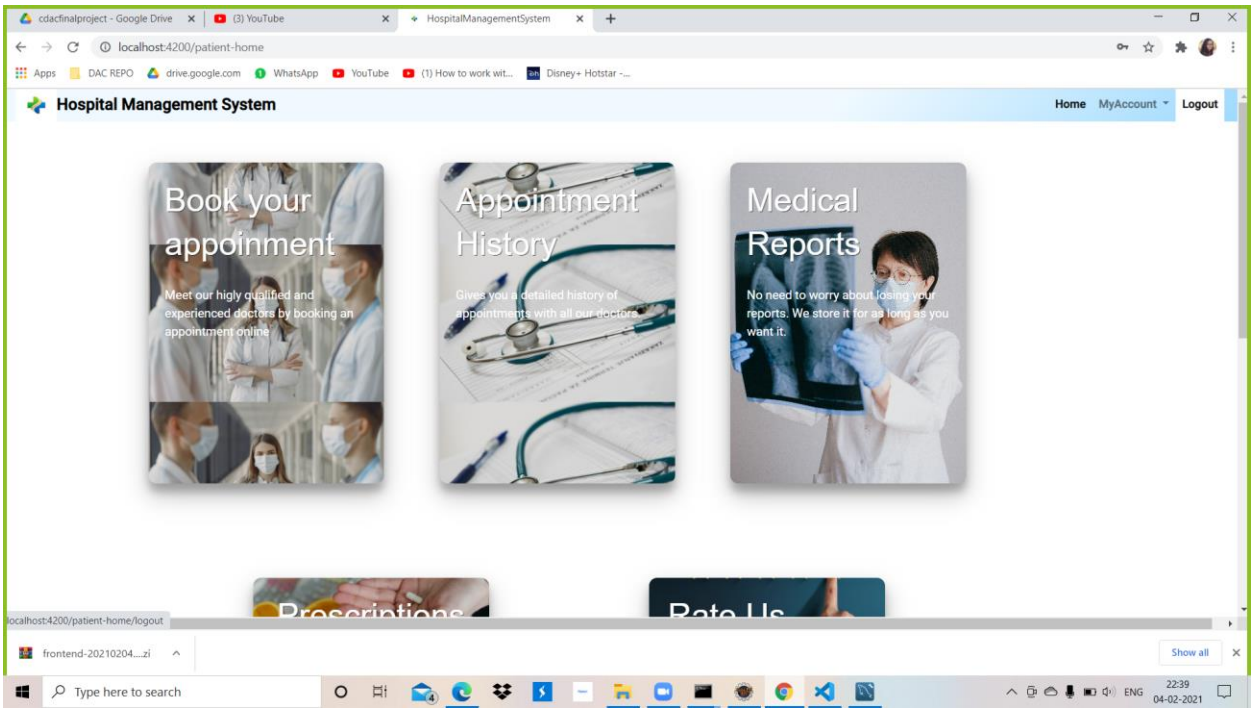


Figure 15:Logout Button

6. Code Snippet:

Registration:

```
@PostMapping("/p_register")
public ResponseEntity<?> savePatientDetails(@RequestBody Patient p) {
System.out.println("in save patient " + p);// transient un marshalled from json ---> student details
                        try {
                                Patient details = service.savePatientDetails(p);
                                return new ResponseEntity<>(details, HttpStatus.CREATED);
                        } catch (RuntimeException e) {
                                System.out.println("err in save " + e);
                                return new
ResponseEntity<>(HttpStatus.INTERNAL_SERVER_ERROR);// empty body content , sending
only err code
                        }
                }
```

Login:

```
        @PostMapping("/p_login")
        public ResponseEntity<?> patientLogin(@RequestBody Patient p) {
                System.out.println("get by email and password " + p.getEmail() + " " +
        p.getPassword());
                List<Patient> patient =
        service.getPatientsByEmailAndPassword(p.getEmail(),p.getPassword());
                if (patient.isEmpty())
                        return new ResponseEntity<>(HttpStatus.NO_CONTENT);
                return new ResponseEntity<>(patient, HttpStatus.OK);
    }
```

Booking Appointment:

```
@PostMapping("/p_booking")
	public ResponseEntity<?> bookAppointment(@RequestBody Booking b){
		System.out.println("in patient booking " + b);

		try {
			Booking book = service.bookAppointment(b);
			return new ResponseEntity<>(book, HttpStatus.CREATED);


		} catch (RuntimeException e) {
			System.out.println("err in booking " + e);
			return new
ResponseEntity<>(HttpStatus.INTERNAL_SERVER_ERROR);// empty body content ,
sending only err code
		}

	}
```

Appointment History:

```
@PostMapping("/appointments")
	public ResponseEntity<?> getAppointments(@RequestBody Patient patient){
		System.out.println("in appointments"+patient);
		List<Booking> patients = service.getAllBooking(patient);
		//System.out.println(patients);//
		if (patients.isEmpty())
			return new ResponseEntity<>(HttpStatus.NO_CONTENT);
		return new ResponseEntity<>(patients, HttpStatus.OK);
	}
```

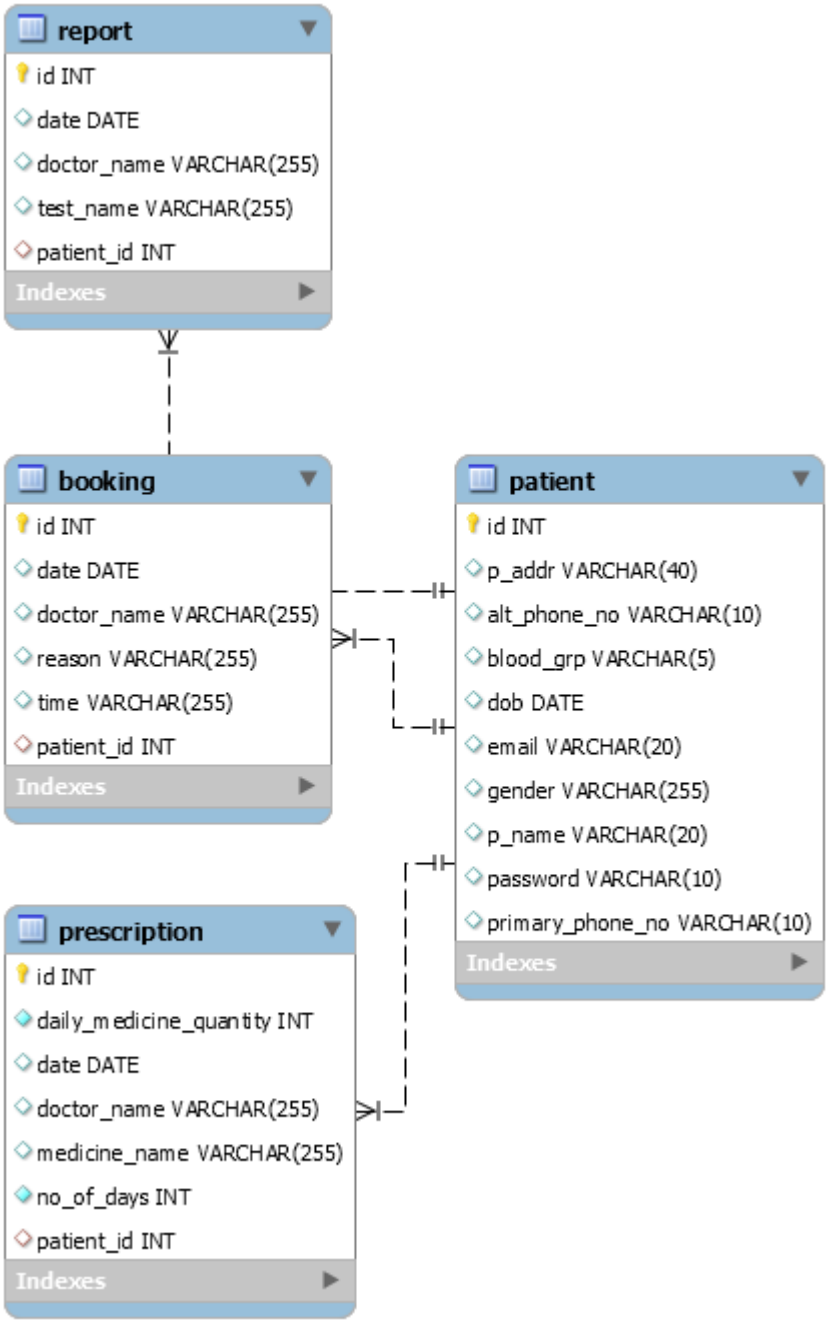
Reports:

```
@PostMapping("/reports")
	public ResponseEntity<?> getReports(@RequestBody Patient patient){
		System.out.println("in Reports"+patient);
		List<Report> patients = service.getAllReports(patient);
		//System.out.println(patients);//
		if (patients.isEmpty())
			return new ResponseEntity<>(HttpStatus.NO_CONTENT);
		return new ResponseEntity<>(patients, HttpStatus.OK);
	}
```

Prescriptions:

```
@PostMapping("/prescriptions")
	public ResponseEntity<?> getPrescriptions(@RequestBody Patient patient){
		System.out.println("in Prescriptions"+patient);
		List<Prescription> patients = service.getAllPrescriptions(patient);
		//System.out.println(patients);//
		if (patients.isEmpty())
			return new ResponseEntity<>(HttpStatus.NO_CONTENT);
		return new ResponseEntity<>(patients, HttpStatus.OK);
	}
```

7. EER Diagram:



8. TEST REPORT

Sr. No	Test Case Title	Description	Expected Outcome	Outcome	Result
1	Verify whether Patient registered successfully.	Check whether the entered Patient details are persisted in the database.	Patient details should be stored in the database.	Patient details stored in the database.	Passed
2	Verify whether Patient login is successful (valid Credentials).	Check whether the entered Patient login credentials match with the existing data in the database.	Patient should be logged in successfully.	Patient logged in successfully.	Passed
3	Verify whether Patient login is unsuccessful (invalid Credentials).	Check whether the entered Patient login credentials does not match with the existing data in the database.	Invalid Credential dialogue box should appears.	Invalid Credential dialogue box appears.	Passed
4	Verify whether Patient appointment	Check whether the entered booking details are stored in the database and is visible	Booking details should be stored in the	Booking details stored in the database and	Passed

	booking is successful	**in appointment history tab.**	**database and should be visible.**	**is visible.**	
5	**Verify whether Patient logout is successful.**	**Check whether the patient is redirected to login page.**	**Patient should be logged out of the portal.**	**Patient is logged out of the portal.**	**Passed .**

9. FUTURE SCOPE:

1) Admin Portal.

2) Doctor Portal.

3) Staff Portal.

10. REFERENCES:

[1] Herbert Scheldt, Java Complete Reference, Fifth Edition, Tata McGraw Hill Edition.
[2] Phil Hanna, JSP 2.0: The Complete Reference, Tata McGraw Hill Edition, 2003.
[3] Elmarsi and Navathe, Fundamentals of Database System (Third Edition), Addision Wesley.
[4] Ian Somerville, Software Engineering, Third Edition, Pearson Education.
[5] Ali Bahrami, Object-Oriented System Development, Third Edition, Tata McGraw Hill Edition.
[6] Ivan Bayross, SQL, PL/SQL programming language of Oracle, Second Edition, BPB Publication