PES UNIVERSITY, BANGALORE
Department of Computer Science and Engineering

B.TECH. (CSE)

VI SEMESTER

UE20CS352 – Object Oriented Analysis and Design with Java

Mini-Project Report

on

E-HealthCare Management System

SUBMITTED BY

NAME SRN

1) Tanish P – PES1UG20CS458
2) Tanvi Rajesh – PES1UG20CS461
3) Utkarsh Bagaria – PES1UG20CS477
4) Varun Satheesh – PES1UG20CS489

Section H

January – May 2023

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

BENGALURU – 560100, KARNATAKA, INDIA

PES UNIVERSITY, BANGALORE
Department of Computer Science and Engineering

Problem Statement and Synopsis

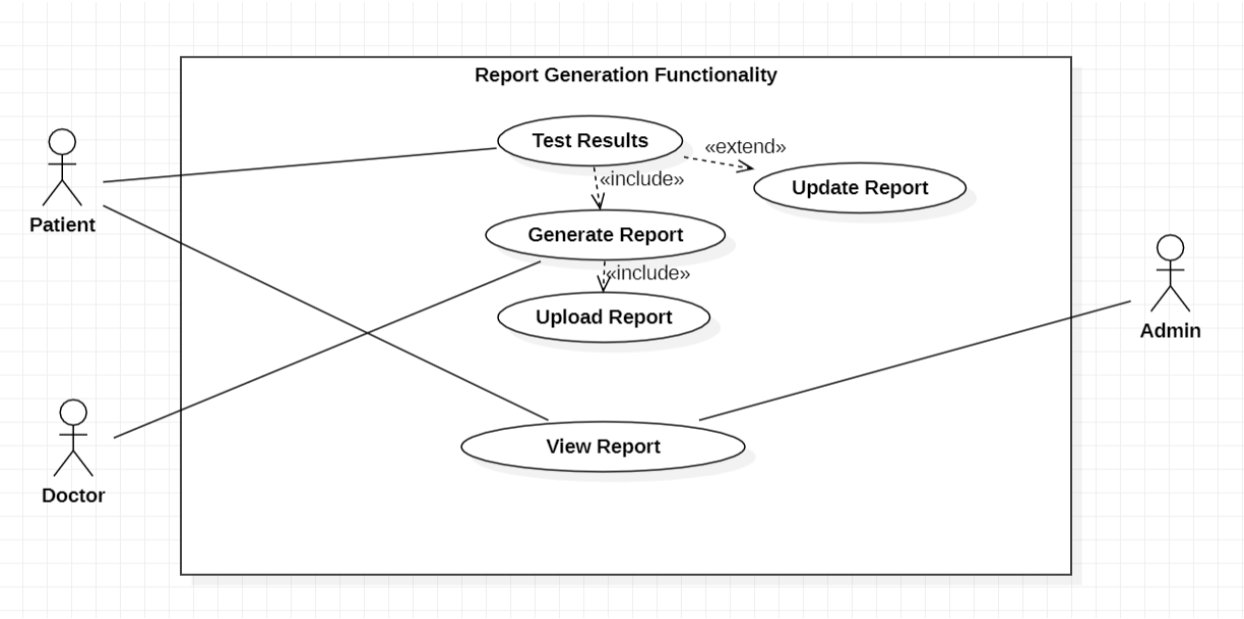
- Health is a very important aspect of life. With the onset of digital systems, the efficiency of hospital systems, staff and doctors can be improved considerably. This reduces wait times for customers and increases workflow for the hospital stakeholders.
- This project is an E-Health-Care Management System. It is an application which is developed using Java. The frontend for this project is hosted using Java Servlets on Apache Tomcat. The backend consists of a MySQL database.
- It aims at aiding both medical staff as well the public. This application can be used by patients as well as doctors in an user-friendly way. It will be maintained by an admin user who will administer over all the features of the software.
- Using this Application, patients can quickly login, book a room if required, make an appointment with the respective doctor at their preferred time, the appointment is successful if the doctor doesn't have a previous appointment at the same time, access their reports, give feedback, make the payment online and logout.
- The admin can add new doctors when they join, view the doctor list, view the appointments, verify the payments made, see feedback given by patients, view reports and logout.
- Doctors can login, view their appointments, and create patient reports and logout.

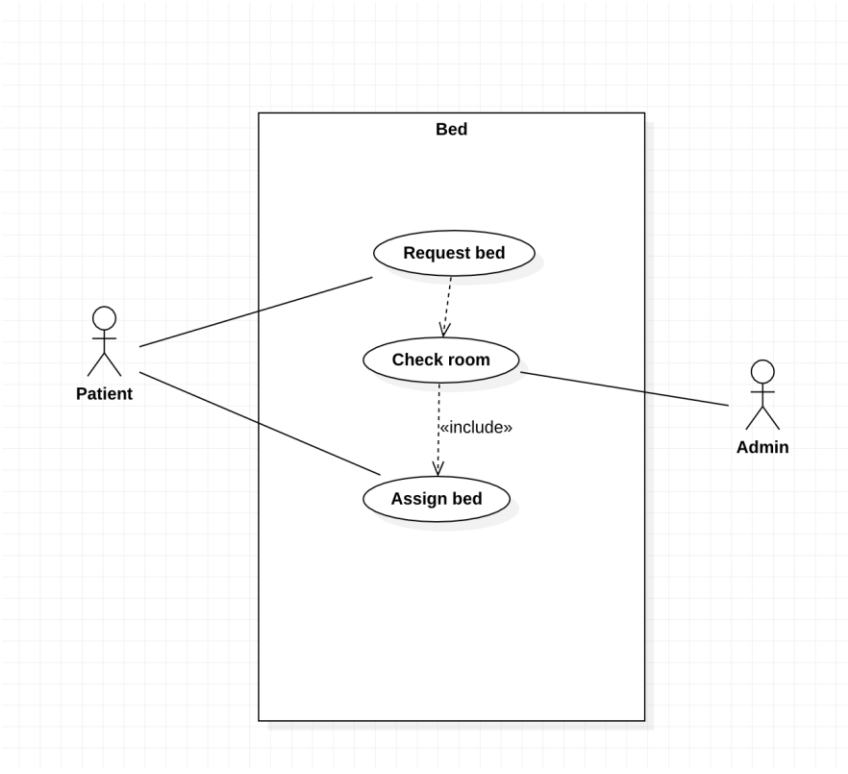
PES UNIVERSITY, BANGALORE
Department of Computer Science and Engineering

Models

Use Case Model

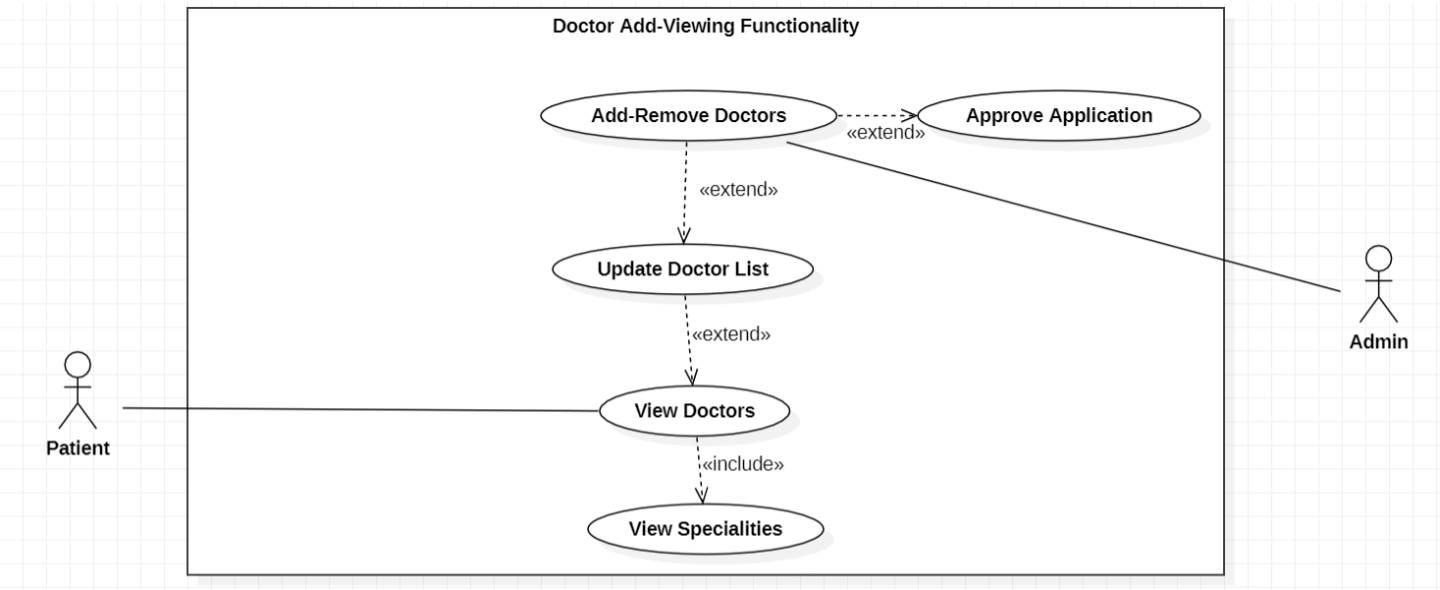
Tanish P - PES1UG20CS458

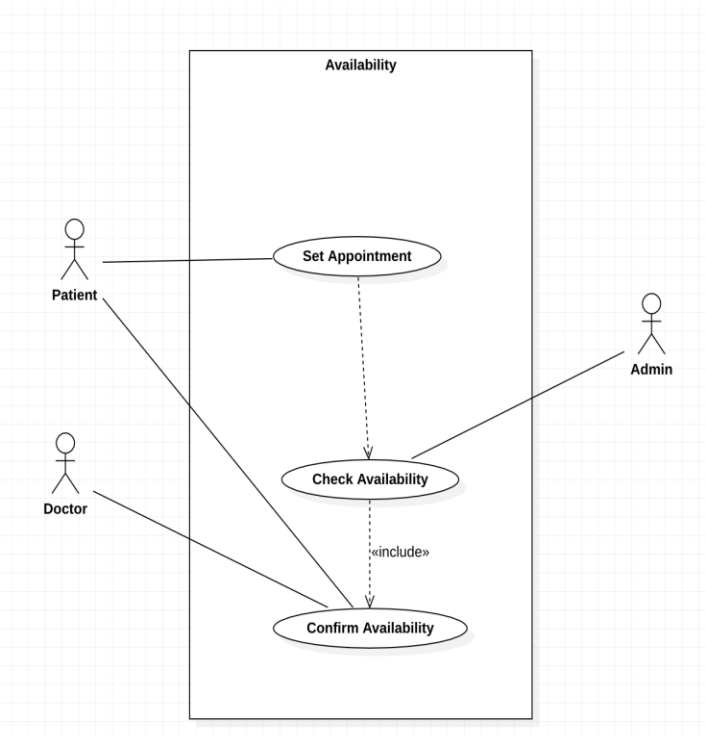




PES UNIVERSITY, BANGALORE
Department of Computer Science and Engineering

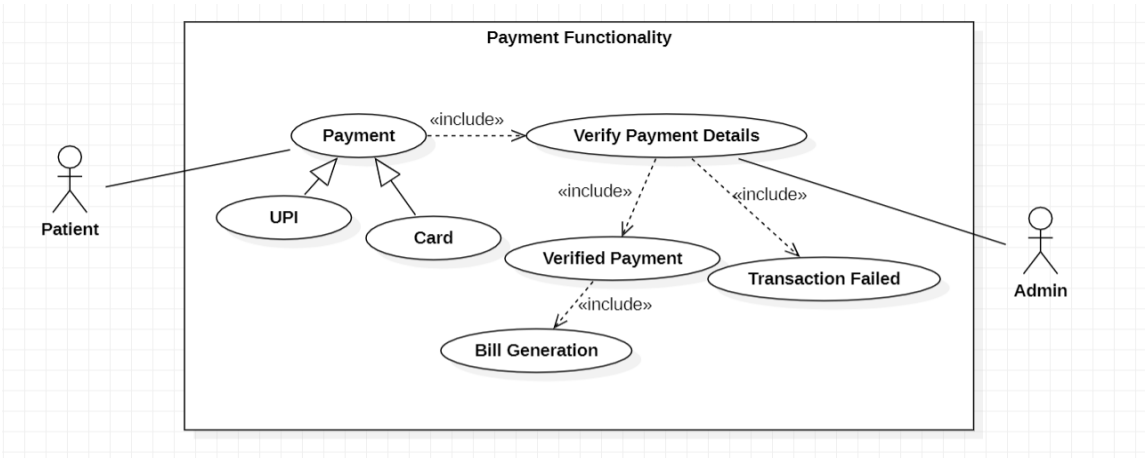
Tanvi Rajesh - PES1UG20CS461

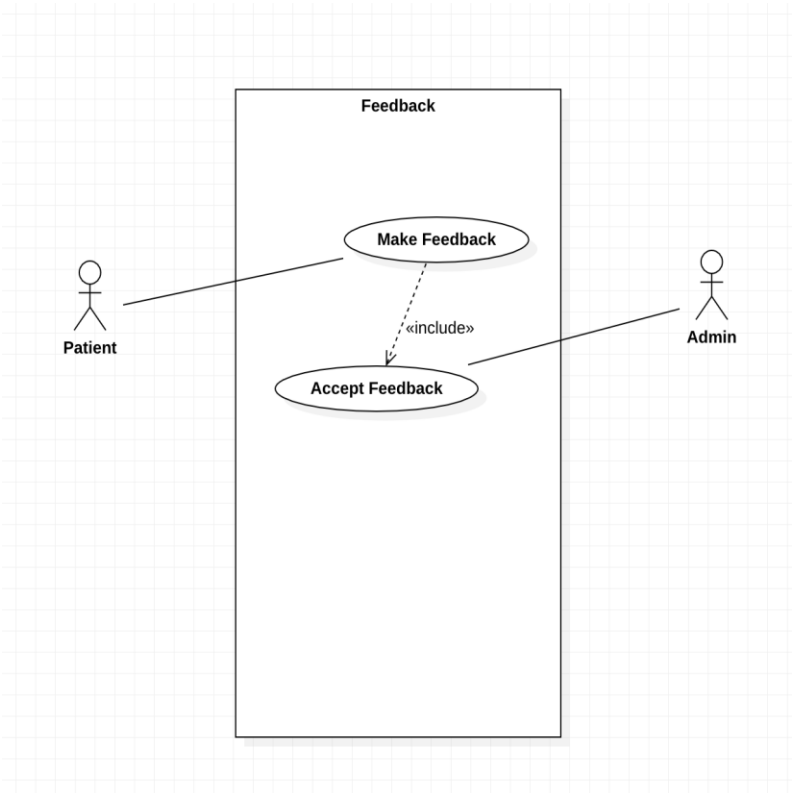




PES UNIVERSITY, BANGALORE
Department of Computer Science and Engineering

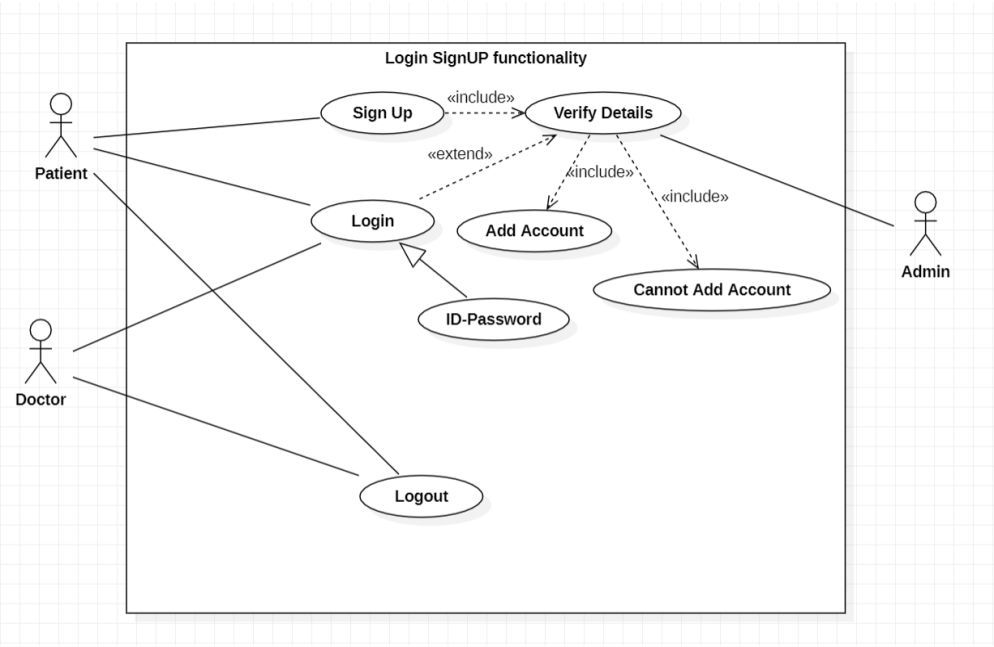
Utkarsh Bagaria - PES1UG20CS477

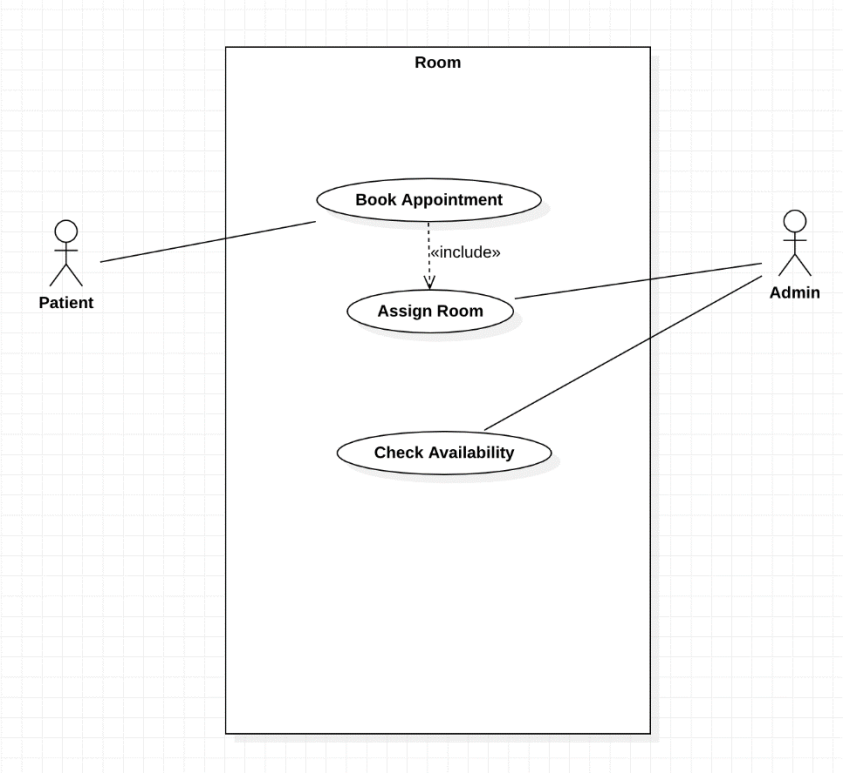




PES UNIVERSITY, BANGALORE
Department of Computer Science and Engineering

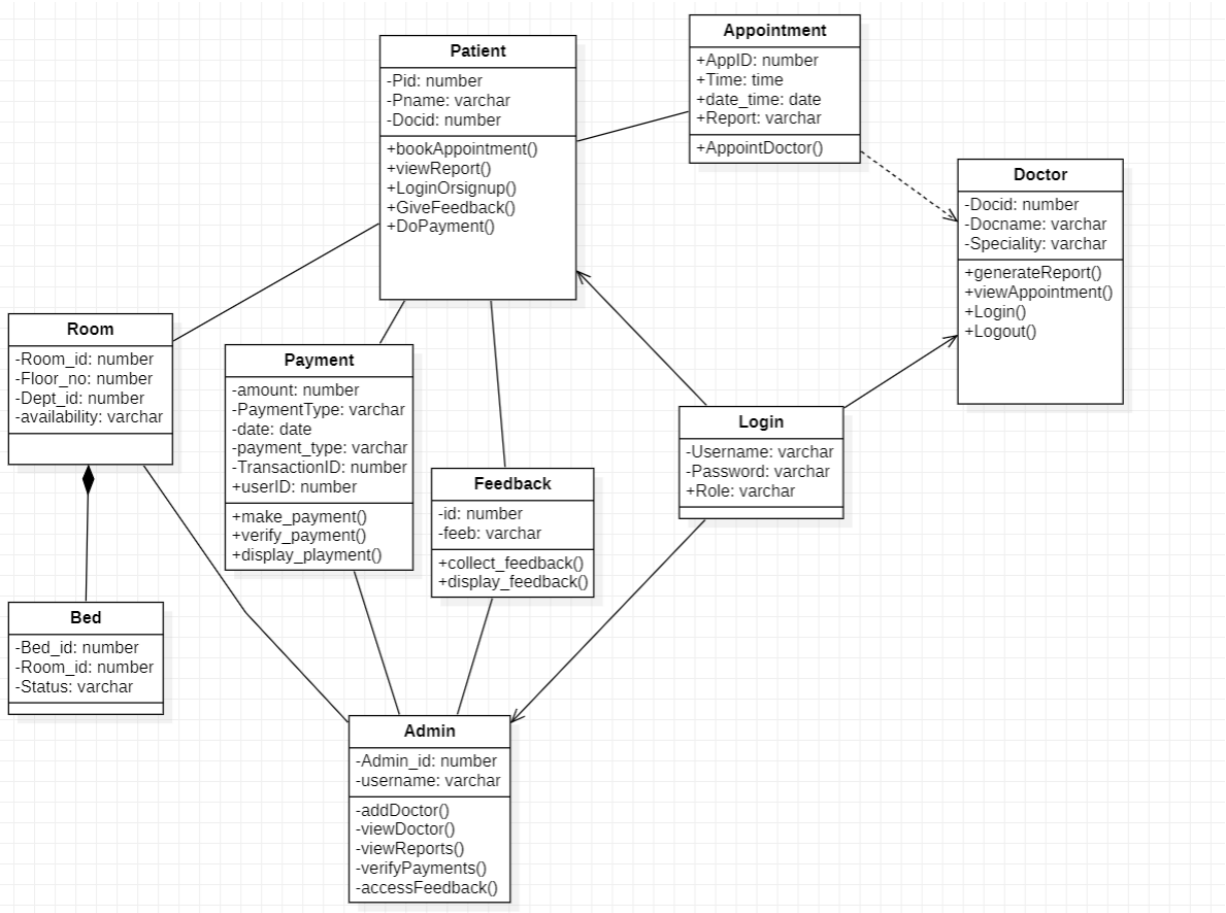
Varun Satheesh - PES1UG20CS489





PES UNIVERSITY, BANGALORE
Department of Computer Science and Engineering

Class Model



Architectural Pattern

Model-View-Controller

MVC (Model-View-Controller) is a software architectural pattern commonly used in developing user interfaces for web applications. The pattern separates an application into three interconnected components: the model, the view, and the controller.

The Model component represents the data and the business logic of the application. It is responsible for managing the data and performing operations on it.

The View component represents the user interface of the application. It is responsible for presenting the data to the user in a way that is understandable and meaningful.

The Controller component acts as an intermediary between the Model and the View. It receives input from the user and manipulates the Model to perform the requested operation. It also updates the View with the changes made to the Model.

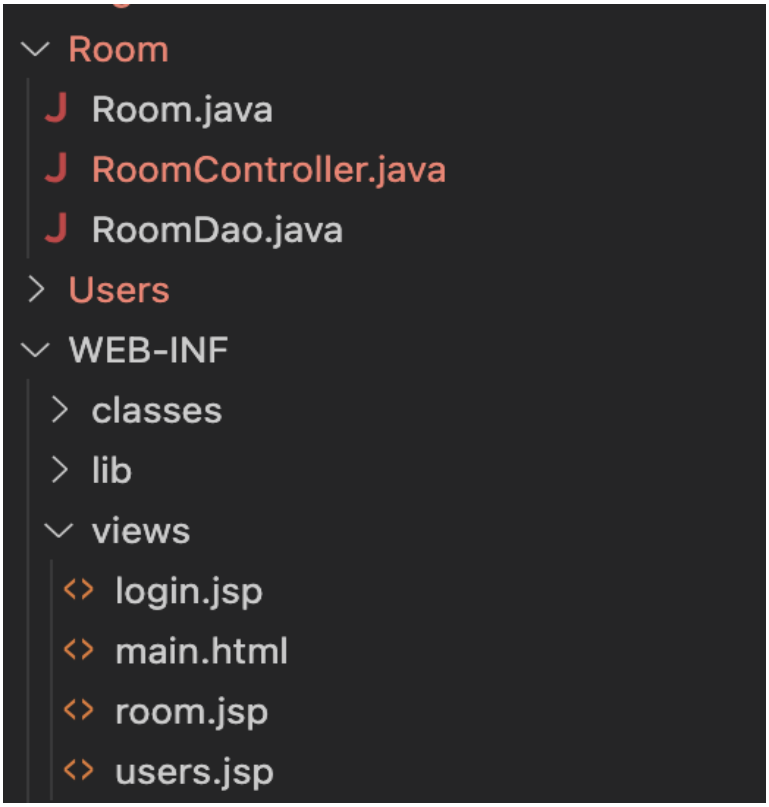
Using the MVC pattern, developers can create software that is modular, reusable, and easy to maintain. It also allows for easier testing and debugging, as each component can be tested separately.

MVC is widely used in developing web applications, but it can also be used in other software development projects where there is a need to separate the presentation layer from the business logic and data.

Model View Controller in our project was done using the DAO (Data Access Object) file, the Controller file, the main Model file and the JSP frontend.

The DAO file basically provides the code for querying the database and to pull, update, add information to the database.

The below snapshot gives a glimpse into the directory structure of our project which gives a clear demarcation of MVC models.


Room
Room.java
RoomController.java
RoomDao.java
Users
WEB-INF
classes
lib
views
login.jsp
main.html
room.jsp
users.jsp

PES UNIVERSITY, BANGALORE
Department of Computer Science and Engineering

Design Principle and Patterns

I. SOLID

1. **ISP – Interface Segregation Principle:** The interfaces for user login and sign-up are segregated so that the classes only implement the required functionalities.
2. **SRP – Single Responsibility Principle:** Every single functionality for different types of users has its own controller and views segregated from other users, i.e. The view and controller of a specific function of a user only handles that functionality's jobs.

II. GRASP

1. **Information expert – Admin:** The admin has all information regarding doctors through which their profiles can be modified. All the information related to various aspects of the project is contained in the admin.
2. **Low coupling – Admin Appointment Payment:** Although the admin verifies the payment, this does not affect or make changes to the actual payment/ payment class which is created from the Appointment class. Hence, this helps us in achieving low coupling between the admin class and the Payment class.
3. **Controller – Login:** The login page controls the flow of data. The login takes in the input from the user and based on the role (patient, admin, doctor) entered it redirects the flow of data to the respective views.
4. **Polymorphism – View:** Polymorphism means "multiple forms", and it occurs when we have many classes that are related to each other by inheritance. Here, we notice that view is used multiple times in all the different classes to essentially display the output.

III. Design Patterns

1. **Factory Method:** Whenever any user tries to add details of any kind to the database, a completely new object is made of that type in the DAO class files for that specific type of data. Thus, the DAO files follow the factory method by making objects of another class.
2. **Adapter:** According to the role provided by the user they are redirected to the view required by them. Thus, the view adapts to the role provided by the user. The feedback page according to the role, that is user or admin changes the view accordingly. Admin can view feedback only while the patient can give feedback only.
3. **Façade:** The main html page acts as the high-level UI. This leads to the login page which according to the role entered leads to the respective dashboards like doctor, patient, and admin. Which are higher level interfaces with buttons to lead the user to the respective functionality pages required by them.
4. **Iterator:** All the display functions display the database content when called by iterating through the resultant set that is collected from the database by the Data Access Object.
5. **Command:** The user based on their role on their dashboard can click on whichever button they want, choosing the command for the functionality they want. This will automatically make the webpage redirect itself to the functionality command requested by the user.

GitHub link to Code Repository - tanvi0904/E-HealthCare (github.com)

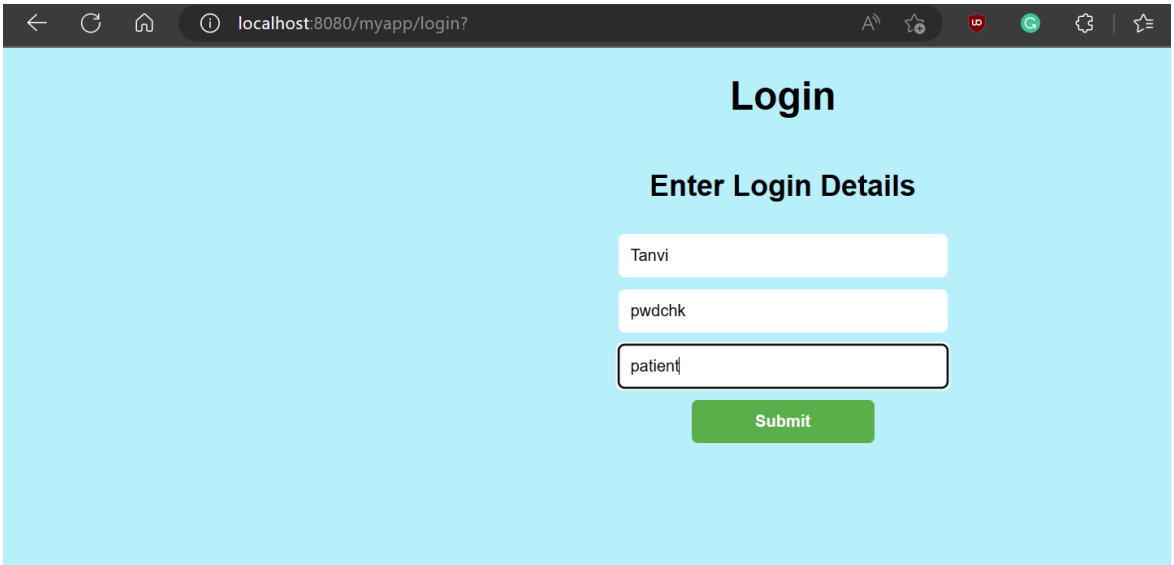
Individual Contribution

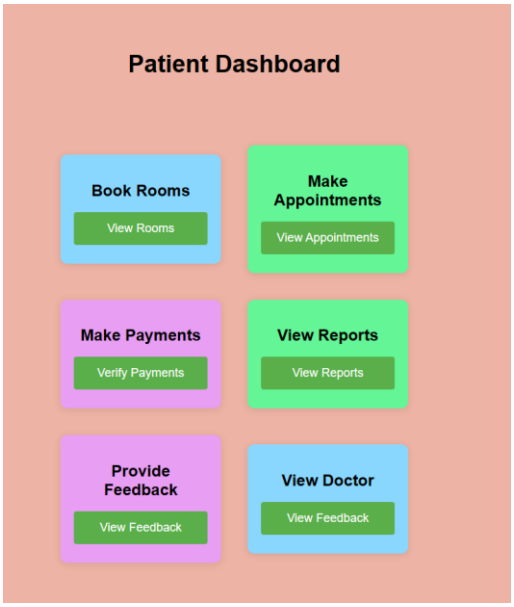
- Tanish P – Report and Bed usecase
- Tanvi Rajesh – Doctor and Appointment usecase
- Utkarsh Bagaria – Payment and Feedback usecase
- Varun Satheesh – Login and Room usecase

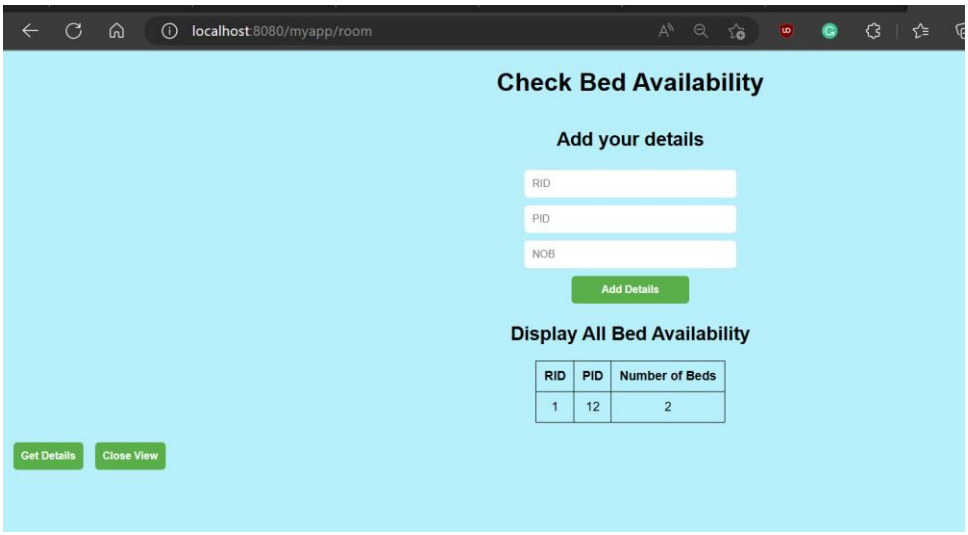
Screenshots

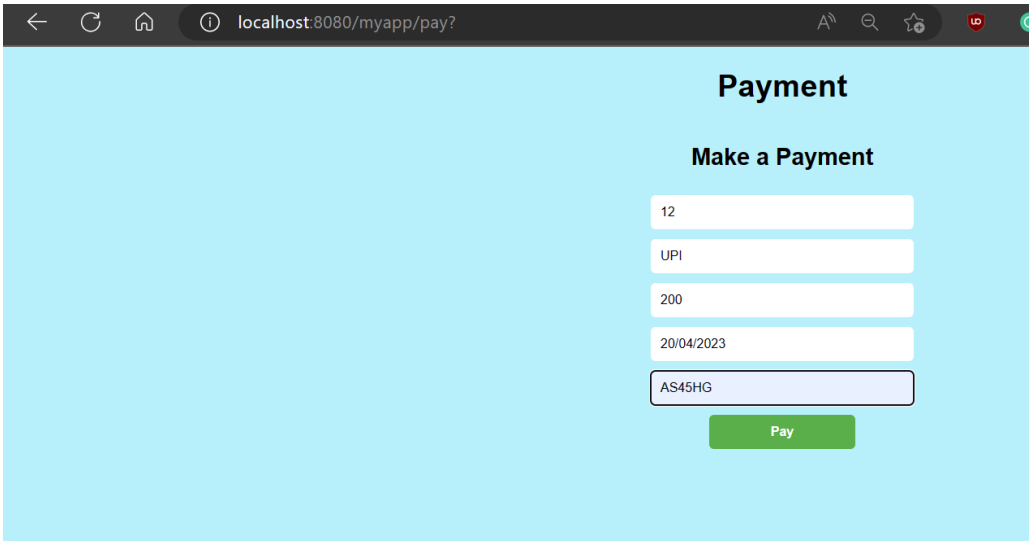
Views

Patient

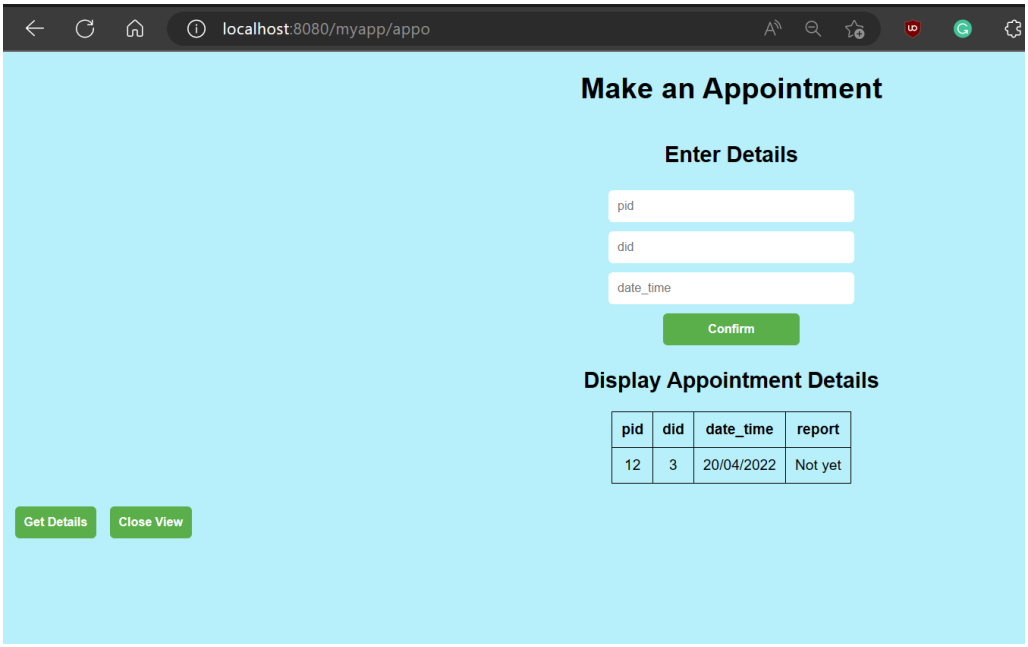


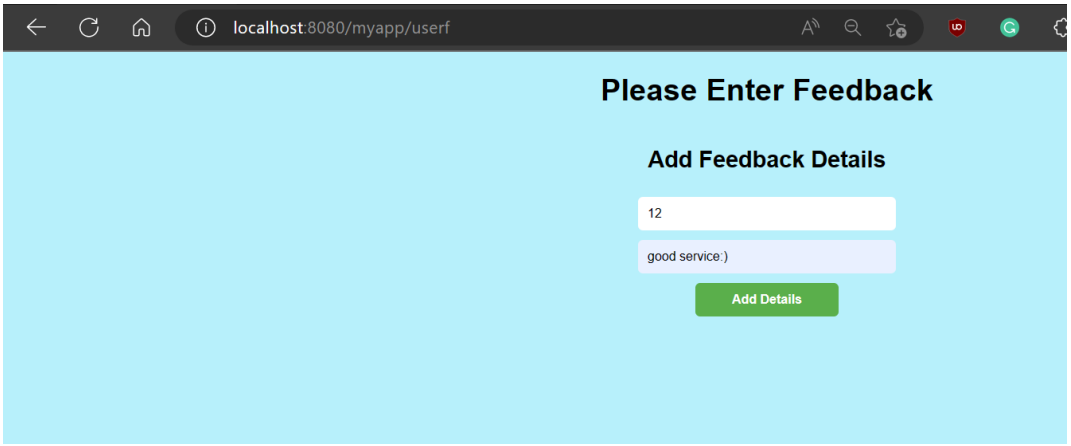

Patient Dashboard
Book Rooms
View Rooms
Make Appointments
View Appointments
Make Payments
Verify Payments
View Reports
View Reports
Provide Feedback
View Feedback
View Doctor
View Feedback

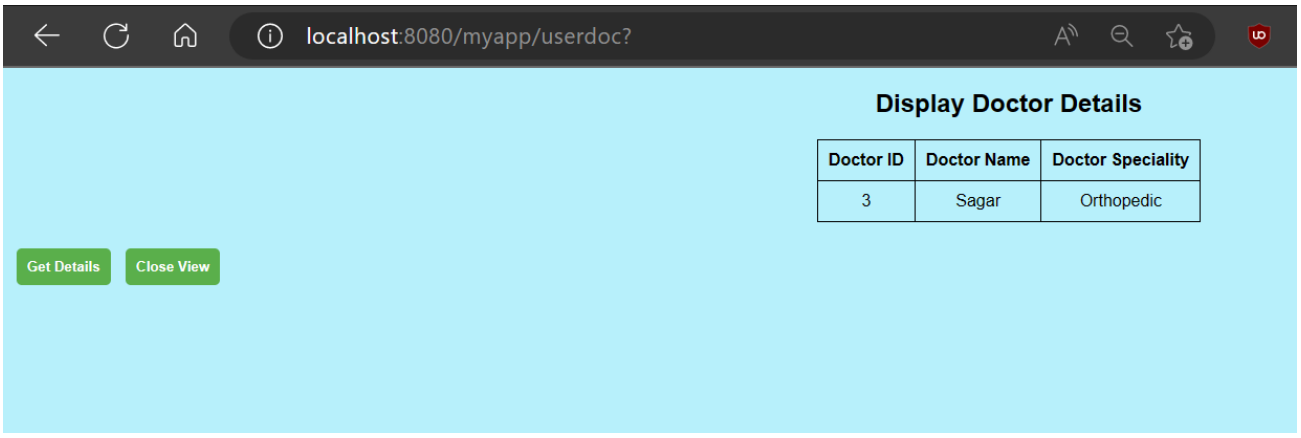

localhost:8080/myapp/room
Check Bed Availability
Add your details
RID
PID
NOB
Add Details
Display All Bed Availability
RID
PID
Number of Beds
1
12
2
Get Details
Close View


localhost:8080/myapp/pay?
Payment
Make a Payment
12
UPI
200
20/04/2023
AS45HG
Pay

PES UNIVERSITY, BANGALORE
Department of Computer Science and Engineering

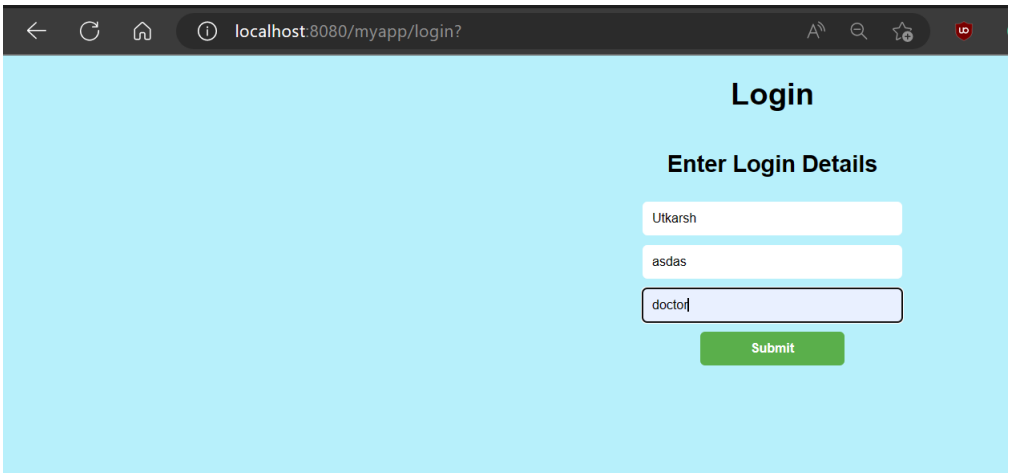


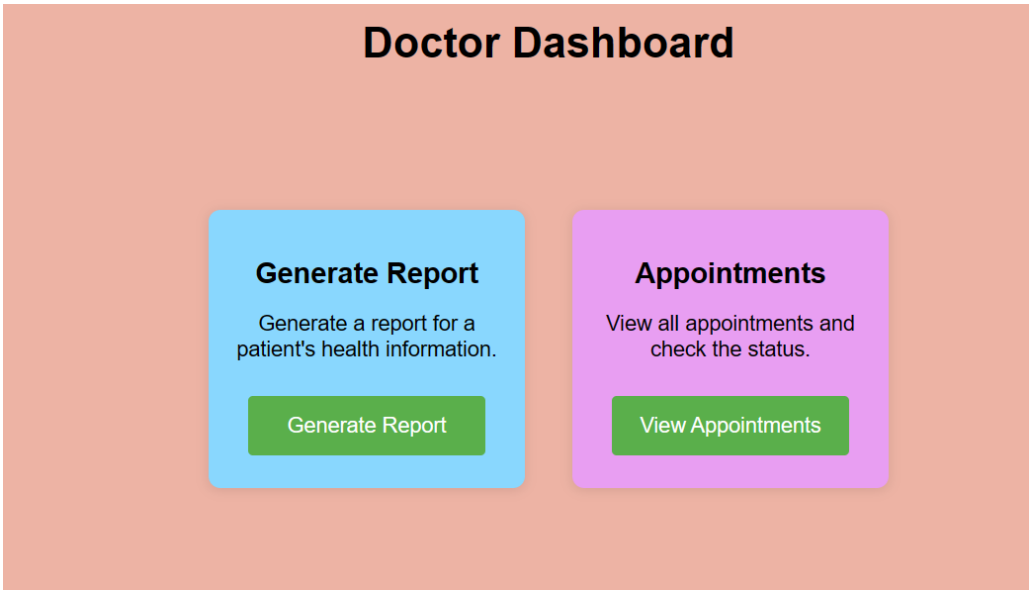


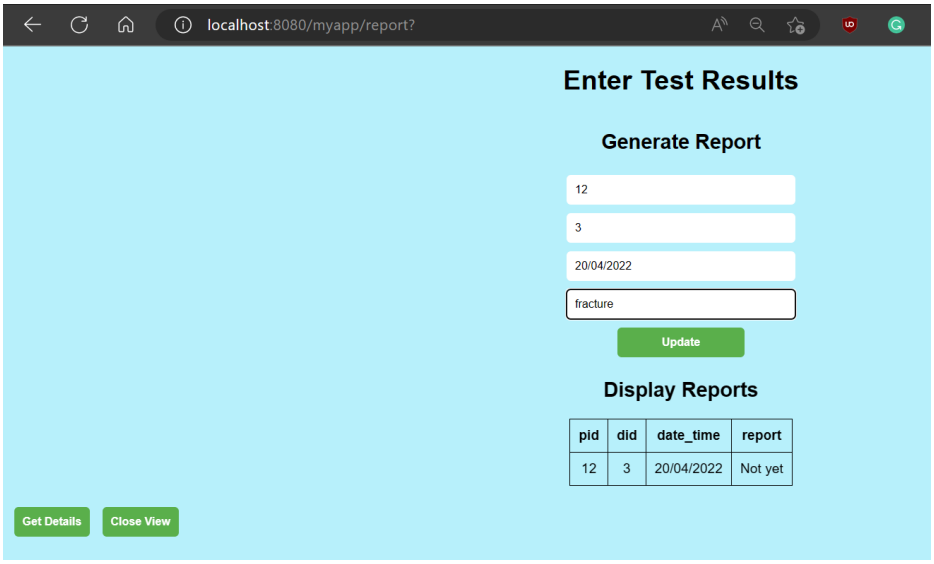


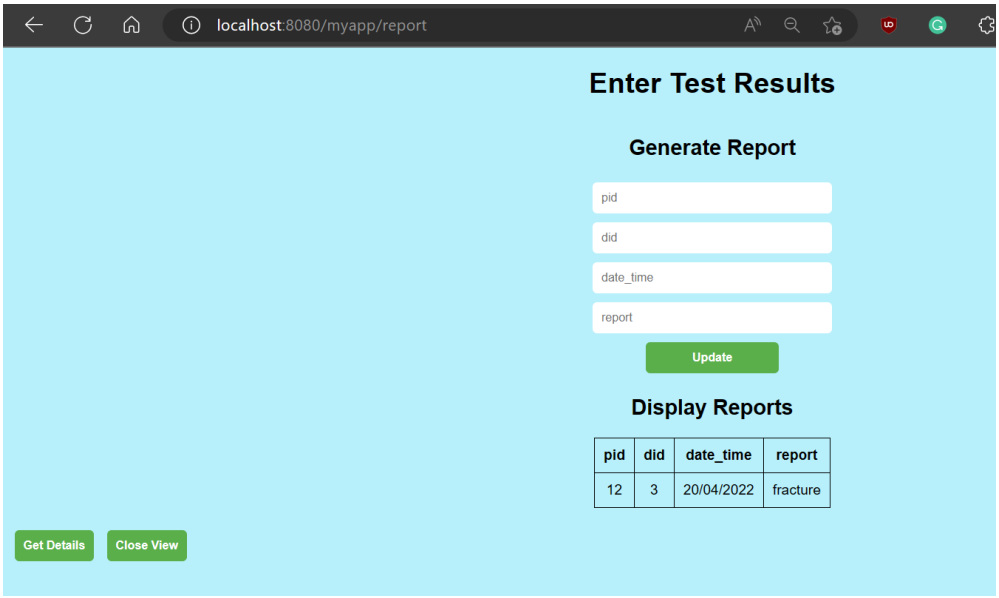
PES UNIVERSITY, BANGALORE
Department of Computer Science and Engineering

Doctor

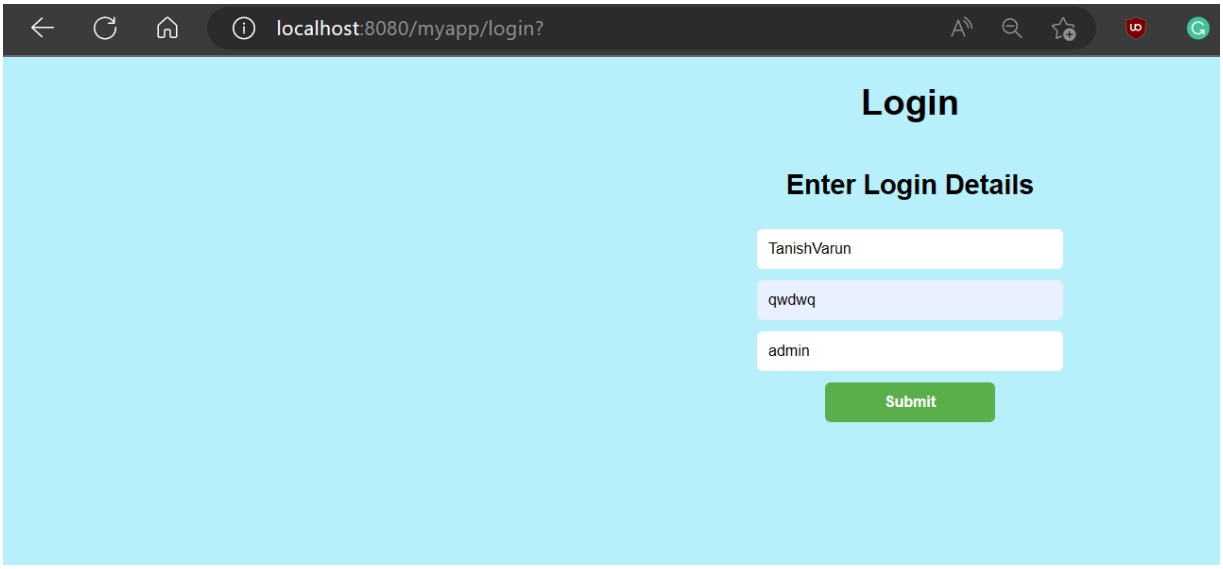


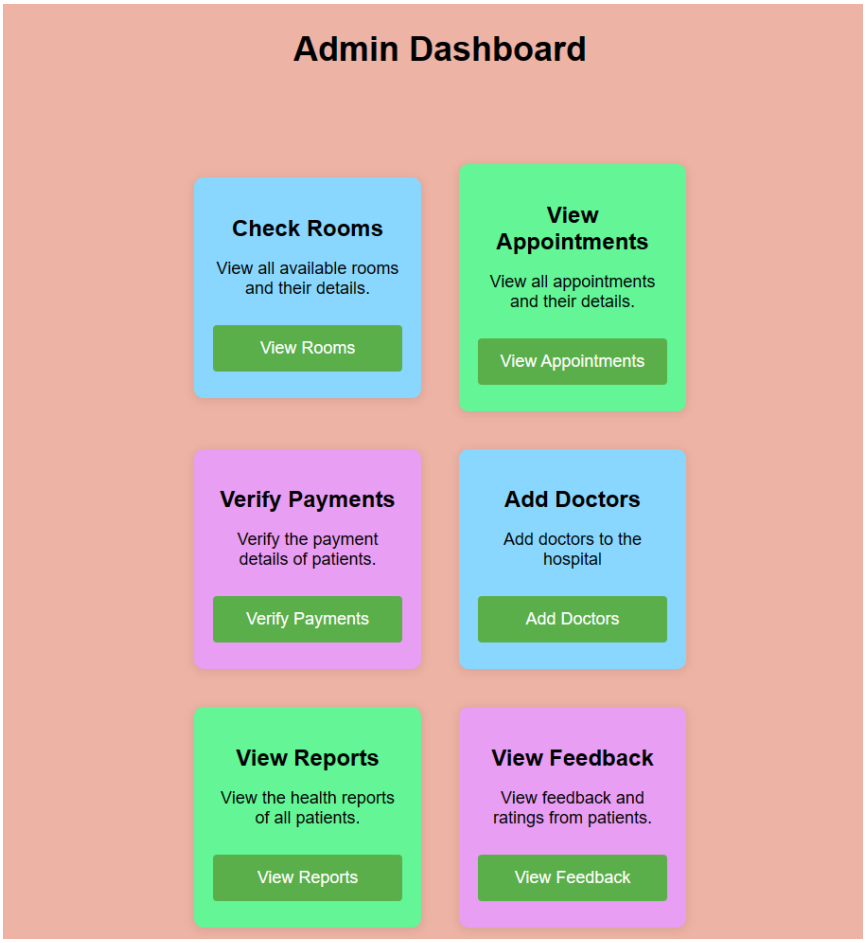


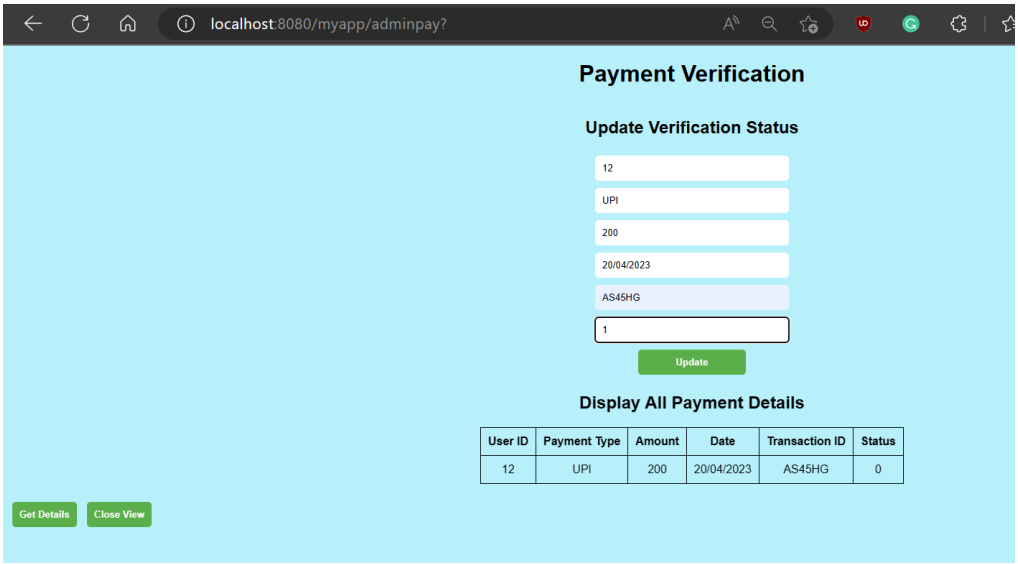



localhost:8080/myapp/report
Enter Test Results
Generate Report
pid
did
date_time
report
Update
Display Reports
pid
did
date_time
report
12
3
20/04/2022
fracture
Get Details
Close View

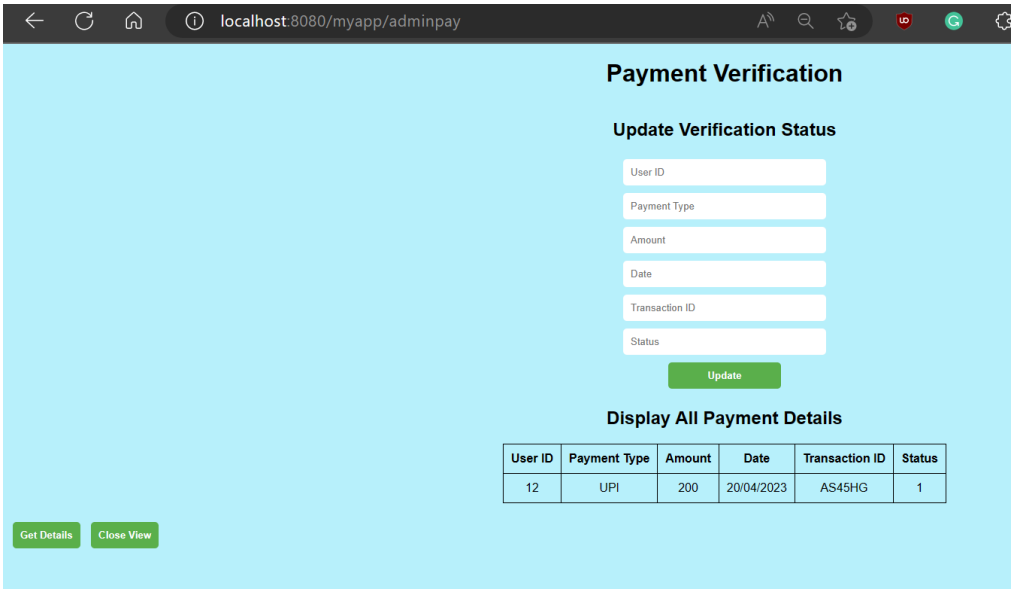
Admin

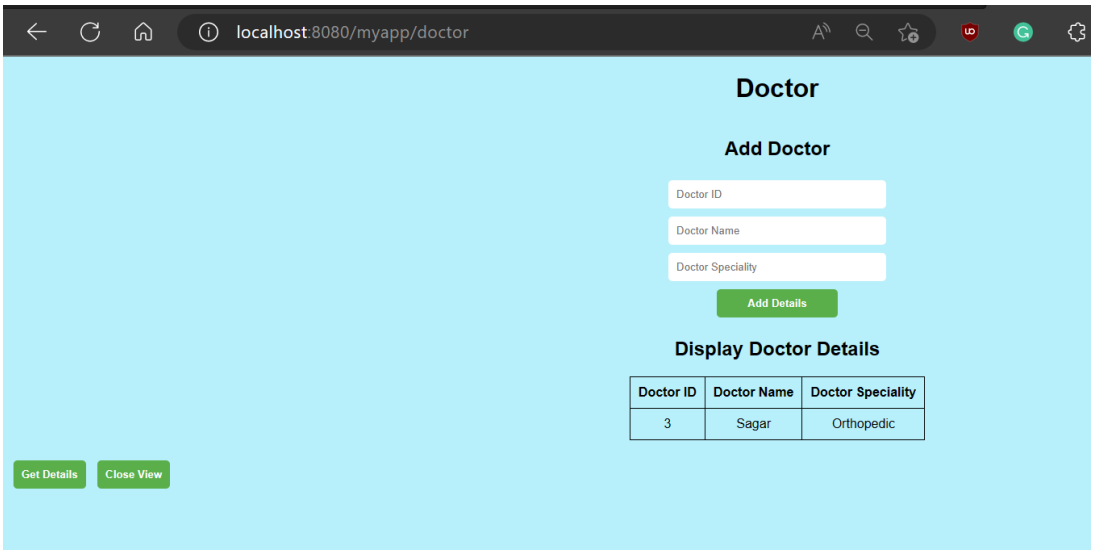

localhost:8080/myapp/login?
Login
Enter Login Details
TanishVarun
qwdwq
admin
Submit

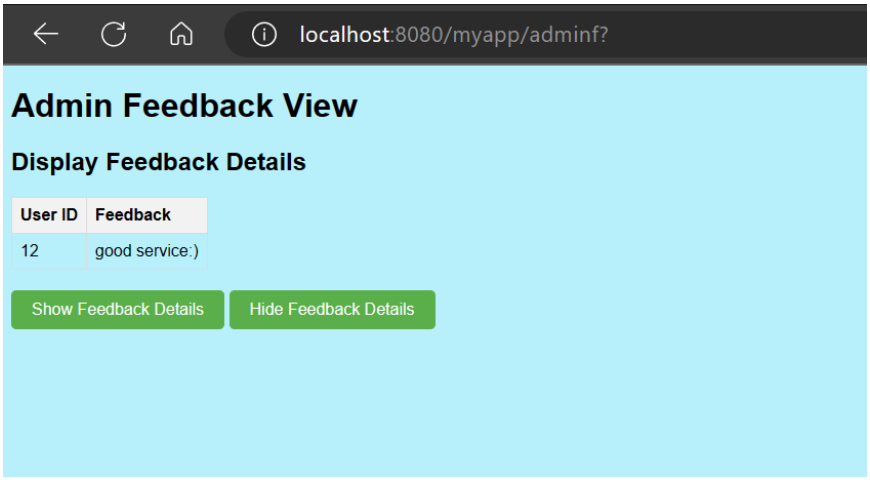

Admin Dashboard
Check Rooms
View all available rooms and their details.
View Rooms
View Appointments
View all appointments and their details.
View Appointments
Verify Payments
Verify the payment details of patients.
Verify Payments
Add Doctors
Add doctors to the hospital
Add Doctors
View Reports
View the health reports of all patients.
View Reports
View Feedback
View feedback and ratings from patients.
View Feedback


localhost:8080/myapp/adminpay?
Payment Verification
Update Verification Status
12
UPI
200
20/04/2023
AS45HG
1
Update
Display All Payment Details
User ID
Payment Type
Amount
Date
Transaction ID
Status
12
UPI
200
20/04/2023
AS45HG
0
Get Details
Close View

PES UNIVERSITY, BANGALORE
Department of Computer Science and Engineering







Database

```
MariaDB [mvc]> show tables;
+---------------+
| Tables_in_mvc |
+---------------+
| appointment   |
| doctorlist    |
| feedback      |
| login         |
| pay           |
| room          |
| users         |
+---------------+
7 rows in set (0.002 sec)
```