PEDIATRIC MEDICAL CENTRE

By Team 13th Man

Amisha Battha
Jia He
Neelesh Jayaraman
Pranjal Gururani
Sumeer Angra

Contents

1. Business Case

This business case is designed for the Elk County Pediatric Medical Center. ELK County Maine is awarded a federal grant to build a medical center as well as to purchase a medical information system to assist the doctor and the county in managing the medical center. The grant also paid for the medical education of a pediatrician that will practice at the medical center for at least five years. The software and the hardware requirements for the new medical system is determined by the Business Analyst. This is accomplished by conducting multiple interviews with the doctor, personnel from the county, and the other stakeholders. After the business analyst completed her report, she conducted an extensive investigation to see if an off the-shelf software package would meet the functional requirements of the medical center's stakeholders. However, no standard software package was able to satisfy all of the critical requirements and the analysts suggested to design and implement the system as per the requirements.

As per the requirements mentioned, we design a system to store all the doctor information, patient details, appointment information, and all the other essential details for the medical center. This system allows the parent of the patient to book and cancel an appointment. The doctor looks up the system to view all the patient related information and accordingly performs the diagnosis based on the patient report. System also generates bill depending on the services availed by the patient.

2. Requirements

2.1. Functional Requirements

2.1.1 The system registers the patient into the PMC system, generates an MRN for the patient and the details are stored in the database

2.1.2 The system allows the patient to book appointment, notifies the patient and updates the records in the database

2.1.3 The system permits the patient to cancel an existing appointment, notifies him regarding the appointment and the database is updated

2.1.4 The system allocates the room to the patient depending on the availability and the room database is updated accordingly.

2.1.5 The system adds doctor and stores all his details into the database

2.1.6 The system permits the doctor to update his schedule, and the changes are reflected in the database.

2.1.7 The system allows provisions for updating the doctor information (ex: first name, last name) in the system and sustain the changes in the database

2.1.8 The system allows the administrator to grant permission to the staff to access the system resources and persist the permission levels in the database

2.1.9 The system provides provisions to add staff and updates details in the database.

2.1.10 The system generates bills for the facilities used in the PMC and the records are saved in the database

2.1.11 The system allows the patient to pay bill for the services utilized and save the payment transactions in the database

2.1.12 The system records the discharge and check-in for the patient and updates the records in the database.

2.1.13 The system updates the patient medical history in the system and the records are saved in the database

2.1.14 The system generates a report from the pre-defined templates, displays the report and provides option to physically print it

2.1.15 The system allows the staff to perform an inpatient check-in for a patient who needs to be admitted to the hospital and stores all the data in the database

2.2. Nonfunctional Requirements

Operational

2.2.1 The system should be operational on any Operating System (Windows, Linux, Mac OS etc.)

2.2.2 The system should run on any Web Browser

2.2.3 The system should run on handheld devices

Performance

2.2.4 The updates to the database should all be real-time

2.2.5 Each of the system modules must have response time less than 3 secs

2.2.6 The system should be able to support 100 users concurrently

Security

2.2.7 The staff and people should have a login ID and password to use the system

2.2.8 The system should only allow permitted staff members to access patient and family information

2.2.9 The system should be protected from external malware attacks

2.2.10 The system must comply with the HIPAA law which has provisions for the protection of patient data

Cultural and political

2.2.11 The system should be available in English and Spanish

Maintainability

2.2.12 The system should be able to accommodate new changes without breaking the existing system

Availability

2.2.13 The system should be available for use 24 hours per day, 7 days per week

3. Use Cases

3.1. Register Patient

Use Case Name	Register Patient	**ID:** UC - 1	**Priority**	High
Actor	Staff			
Description	A staff member requests to register a patient. The system registers the patient and notifies the staff member that the patient has been registered			
Triggers	Staff member raises request to register a patient			
Type	External			

Preconditions

1. Staff member is authorized to register a patient
2. Staff member is logged into the system
3. There is a patient to be registered

Normal Course	**Information for Steps**
1. Staff raises request to register a patient	← Registration Request
2. Staff enters user first name, last name and date of birth.	← Patient Details
3. System checks if the patient is already registered in the system (Alternate Course 1.1)	← List of Patients
4. If patient is unique, System requests further patient details	→ Registration Status
5. Staff enter required details	← Patient Details
5. System stores patient record in database	→ Patient Record
6. The system notifies Staff patient has been registered	→ Registration Success Notification
Alternate Courses	
1.1 The patient already exists in the system	
1). The system notifies staff member that the patient exists in the system and exits the program	→ Exists Notification

Post Conditions

1. The patient is added to the systems database
2. Unique MRN Generated for patient

Exceptions

E1) The system is unable to connect to the Patient database
1). System displays message that data is currently unavailable
2). System asks staff member to try registering patient on a different time and use manual form right now

Summary

Inputs	**Source**	**Outputs**	**Destination**
Registration Request	Staff	Registration Status	Staff
Patient Details	Staff	Exists Notification	Staff
List of Patients	Patient Database	Registration Success Notification	Staff
		Patient Record	Patient Database

3.2. Book Appointment

Use Case Name	Book appointment	**ID:** UC - 2	**Priority**	High
Actor	Patient’s Parent			
Description	An appointment needs to be booked by Patient’s parent to avail the service for the child			
Triggers	A parent needs an appointment with doctor for his child			
Type	External			

Preconditions	
1.The system is up and working fine	
Normal Course	**Information for Steps**
1. 0 Patient’s Parent click on Book Appointment	
1. Patient’s Parent can view available slots with the doctor (Alternate course 1.1)	List of available slots
2. Parent enters his and patient’s name and selects the desired slot along with reason for visit	Appointment Details
3. System updates appointment database	Updated Slots
3. Success notification is generated and is displayed to the user	Confirmation
Alternate Courses	
1.1. No slot available for the day, System displays message “No slot available for the day, do you want to book slot for other day?”	
1. If user selects “Yes”, system displays the days with available slots and user selects available slot for appointment (normal course 1.0 (2))	List of available slots
2. If user selects “No”, the system exits the current page	
Post Conditions	
1. Appointment database is updated	
Exceptions	
E1. If the website is down, the system displays the below message “The website is facing some issues, please try after sometime”	
Summary	

Inputs	**Source**	**Outputs**	**Destination**
List of Available slots	Appointment Database	Confirmation	Patient’s Parent
Appointment Details	Patient’s Parent	Updated Slots	Appointment Database

3.3. Cancel Appointment

Use Case Name	Cancel an Appointment	**ID:** UC – 3	**Priority**	High
Actor	Patient's Parent			
Description	A parent cancels an appointment which was previously made in Medical Center.			
Triggers	A parent needs to cancel appointment with doctor for his child			
Type	External			

Preconditions

1. The parent has already made an appointment. And the parent has logged in the system.

Normal Course	**Information for Steps**
1. Patient's Parent clicks on Cancel an Appointment.	
2. The system lists all the appointments the parent has made	List of Appointments
3. Parent selects the appointment to be cancelled by clicking it.	Cancel Request
4. The system displays a confirmation window which shows the previous appointment information and two choices 'Confirm Cancellation' and 'Quit Cancellation'.	Cancellation Confirmation
5. The system updates the appointment database	Updated Slots
6. The system sends cancelled appointment notification.	Cancellation Notification

Post Conditions

Once an appointment is cancelled, the time slot should be available for others to make an appointment

Exceptions

E1. If the webpage is closed without following normal steps.
1. Parent has to restart again.

Summary

Inputs	**Source**	**Outputs**	**Destination**
List of Appointments	Appointment Database	Updated Slots	Appointment Database
Cancel Request	Patient's Parent	Cancellation Notification	Patient's Parent
Cancellation Confirmation	Patient's Parent		

3.4. Allocate Room

Use Case Name	Allocate room	**ID:** UC - 4	**Priority**	High
Actor	Staff			
Description	The staff allocates a room for the patient			
Triggers	A patient requires a room			
Type	External			

Preconditions	
1. The patient is registered 2. Staff is authorized to book a room	
Normal Course	**Information for Steps**
1. Staff requests to book a room	← Room Request
2. System requests information such as room type and period	← Room Info
3. Staff enter required information	← Patient Info
4. System checks room availability (Alternative Course 1.1)	← List of Rooms
5.System allocates room to patient and notifies staff	→ Room Status
6.System updates room status in the database	→ Update Room Availability
Alternate Courses	
1.1. The room is not available	
1. System notifies staff and exits	→ Room Status
Post Conditions	
The room is assigned to the patient	
Exceptions	
E1. The system cannot respond to request.	
Summary	

Inputs	**Source**	**Outputs**	**Destination**
Room Request	Staff	Room Status	Staff
Patient info	Staff	Update Room Availability	Room Database
Room info	Room Database		
List of Rooms	Room Database		

3.5. Add Doctor

Use Case Name	Add Doctor	**ID:** UC - 5	**Priority**	High
Actor	Staff(Admin)			
Description	A new doctor joins the Pediatric Center, requests his ID and administrator will add his information in the Doctor Database and grants him required access			
Triggers	A new doctor joins the hospital			
Type	External			
Preconditions				
1. Administrator is authorized to add the doctor details 2. Administrator is logged in the system 3. System is up and running				

Normal Course	**Information for Steps**
1. New Doctor requests his ID and access	
2. Administrator takes all the details of Doctor like Name, experience, practice, qualification, gender, date of joining, contact number, address etc.	Doctor's details
3. Administrator adds all the Doctor's details into Doctor database and sets his status as active	Doctor Database
4. Administrator generates the Doctor's unique ID and credentials for his access credentials	Updated database
5. Doctor is granted access to his Dashboard or portal	Access granted

Post Conditions
The doctor's details are added in the database and access is granted to log in the system
Exceptions
E1) The system is unable to add the doctor in database 1). System displays message that database is currently unavailable 2). System asks staff member to try again on a different time
Summary

Inputs	**Source**	**Outputs**	**Destination**
Doctor's details	Staff(Admin)	Access granted	Staff(Admin)
		Updated database	Doctor Database

3.6. Update Doctor Schedule

Use Case Name	Update Doctor Schedule	**ID:** UC - 6	**Priority**	High
Actor	Doctor			
Description	A doctor wants to update his schedule so that patients can book the appointment as per his availability			
Triggers	Doctor wishes to update his schedule			
Type	External			

Preconditions
1. Doctor is logged in into the system using valid credentials 2. Doctor has authorization to update his schedule

Normal Course	**Information for Steps**
1. Doctor logs in to his portal using his valid credentials	Valid credentials
2. Doctor click on Update schedule and provides time on various days and update as per his availability	Schedule changes
3. Doctor clicks on save schedule and same is updated in database	Updated Appointment Schedule

Post Conditions
The doctor schedule is updated in the database and is visible to patients for booking an appointment

Exceptions
E1) The system is unable to log in the doctor 1). System displays message "Invalid credentials, please check with your administrator" 2). The doctor can check with administrator for credentials or click on forgot password and answer security questions to reset his password

Summary

Inputs	**Source**	**Outputs**	**Destination**
Valid credentials	Doctor	Updated Doctor Schedule	Doctor Database
Schedule changes	Doctor		

3.7. Update Doctor

Use Case Name	Update Doctor	**ID:** UC - 7	**Priority**	High
Actor	Doctor			
Description	A doctor would like to update his details. The administrator searches the doctor and make the requested changes			
Triggers	Doctor wants to update his details like contact, address etc.			
Type	External			
Preconditions				
1. Doctor is logged in into the system 2. Doctor details are present in the database				
Normal Course			**Information for Steps**	
1. Doctor chooses to update his details in in the portal by selecting edit details				
2. Doctor enters his new details like address, contact number etc. he wants to update			Doctor details	
3. Doctor clicks on update to save his new entered details			Updated Doctor Info	
4. A confirmation is displayed on screen with message “your profile has been updated”			Confirmation message	
Post Conditions				
The doctor details are updated in the database and reflected correctly in his dashboard or portal				
Exceptions				
E1) The system is unable to update the doctor details in database 1). System displays message that database is currently unavailable 2). System asks doctor to try again on a different time				
Summary				

Inputs	**Source**	**Outputs**	**Destination**
Doctor details	Doctor	Confirmation message	Doctor
		Updated Doctor Info	Doctor Database

3.8. Grant Permission

Use Case Name	Grant permission	**ID:** UC - 8	**Priority**	High
Actor	Staff(Administrator)			
Description	Authorized staff grants the staff members administrator rights to use the system resource			
Triggers	New staff require administrator rights to use the system			
Type	External			
Preconditions				
1. Staff(Administrator) is authorized to add other staff member details 2. Staff(Administrator) is logged in the system 3. System is up and running				
Normal Course			**Information for Steps**	
1.0 Grant permission				
1. Staff(Administrator) enters the ID of the staff member		$\leftarrow$	Staff ID	
2. The system fetches the staff details		$\leftarrow$	Staff Info	
3. Staff(Administrator) checks the staff information and updates the permission level		$\rightarrow$	Updated Permission Level	
4. The staff get a confirmation message about the modified permission level		$\rightarrow$	Confirmation	
Post Conditions				
The permission is recorded both for the staff and in the system				
Summary				

Inputs	**Source**	**Outputs**	**Destination**
Staff ID	Staff	Updated Permission Level	Staff Database
Staff Info	Staff Database	Confirmation	Staff

3.9. Add Staff

Use Case Name	Add Staff	**ID:** UC - 9	**Priority**	High
Actor	Staff(Admin)			
Description	A new staff member joins the Pediatric Center, requests his credentials and administrator will add his information in the Staff database and grants him required access			
Triggers	A new staff member joins the hospital			
Type	External			
Preconditions				
1. Administrator is authorized to add staff member details 2. Administrator is logged in the system 3. System is up and running				

Normal Course	**Information for Steps**
1.New staff member joins the medical center and requests access	
2. Administrator takes all the details of staff member like Name, experience, practice, qualification, gender, date of joining, contact number, address etc.	← Member details
3. Administrator checks if the new member is an admin or not and sets the role accordingly	
4. Administrator adds the staff member's details into Staff database and sets his status as active along with setting his permission level	→ Updated Staff Info
5. Administrator generates the staff member's required access and email is sent for his credentials granting him access to portal	→ Access granted
Post Conditions	
The staff member's details are added in the database and access is granted to log in the system	
Exceptions	
E1) The system is unable to add the member in database 1). System displays message that database is currently unavailable 2). System asks administrator to try again on a different time	

Summary

Inputs	**Source**	**Outputs**	**Destination**
Member details	Staff(Admin)	Access granted	Staff(Admin)
		Updated Staff Info	Staff Database

3.10. Generate Bill

Use Case Name	Generate Bill	**ID:** UC - 10	**Priority**	High
Actor	Staff			
Description	The system will generate bill for a patient			
Triggers	The patient requests the bill			
Type	External			

Preconditions

1. Staff is logged on to a system
2. Patient has used services at the Pediatric Centre

Normal Course		**Information for Steps**
1. Patient requests to generate bill	$\leftarrow$	Bill Request
2. Staff enters Patient’s MRN	$\leftarrow$	MRN
3. System provide list of services	$\leftarrow$	List of Services
4. Staff selects service which are unbilled		
5. System generates bill with selected services	$\rightarrow$	Bill
6. System marks services which have been billed	$\rightarrow$	Modified List of Services

Post Conditions

1. The bill is generated
2. Services for which bill is created are marked billed

Exceptions

E1) The system is unable to connect to Services database
1). System displays message that data is currently unavailable
2). System asks staff member to try generating bill in some time

Summary

Inputs	**Source**	**Outputs**	**Destination**
Bill Request	Patient	Bill	Bill Database
MRN	Patient Database	Modified List of Services	Service Database
List of Services	Service Database		

3.11. Pay Bill

Use Case Name	Pay Bill	**ID:** UC - 11	**Priority**	High
Actor	Patient(Family)			
Description	Patient pays for service used by him			
Triggers	The patient requests the bill			
Type	External			

Preconditions
1. Bill has been created 2. Patient(Family) is logged in to the website 3. Patient(Family) is able to access payment portal 3. Bill is available in Patient Database

Normal Course	**Information for Steps**
1. Patient(Family) selects pending bill	← Bill Selection
2. System provided option to pay with Credit Card or Insurance (Alternate Course 1.1)	← Payment Options
3. Patient(Family) selects credit card	← Option Selection
4. Patient(Family) enters credit card details	← Credit Card Details
5. System accepts the payment and provides transaction Id	→ Transaction ID
6. System marks bill as paid	→ Modified List of Bills
Alternate Courses	
1.1 Patient(Family) selects insurance	
1. System fetches list of insurance carriers	← List of Insurance Carrier
2. Patient(Family) selects insurance carriers	
3. System sends bill invoice to Insurance carrier	→ Insurance Invoice
4. System generates transaction ID	→ Transaction ID
5. System removes bill from pending bills	→ Modified List of Bills

Post Conditions
1. Transaction Id updated in Patient Database 2. Bill is marked as paid
Exceptions
E1) The system is unable to connect to Bill database 1). System displays message that data is currently unavailable 2). System asks staff member to try registering patient on a different time and use manual form right now
Summary

Inputs	**Source**	**Outputs**	**Destination**
Bill Selection	Patient(Family)	Transaction ID	Bill Database
Payment Options	Bill Database	Modified List of Bills	Bill Database
Option Selection	Patient(Family)	Insurance Invoice	Insurance Database
Credit Card Details	Patient(Family)		

3.12. Discharge Patient

Use Case Name	Discharge Patient	**ID:** UC - 12	**Priority**	High
Actor	Staff			
Description	The doctor grants the permission to discharge the patient after the successful service of the patient and staff fulfills the request			
Triggers	The doctor discharges the patient once the service is done			
Type	External			
Preconditions				
1. The patient has already visited the doctor and allocated a room for service				

Normal Course	**Information for Steps**
1. Staff requests to discharge the patient providing the patient details ←	Discharge Confirmation
2. The patient details are updated in the patient data store for the discharge →	Patient Information
3. The room availability is updated for the patient to be discharged →	Room Information
4. Once the patient information is updated the patient discharge use case is completed. →	Discharge Completion

Post Conditions
Once the patient is discharged, the patient records should be updated in the database and the room allocated should also be free for use.
Exceptions
E1. The database cannot be updated, and the patient information is out of date 1. The admin has to try again.
Summary

Inputs	**Source**	**Outputs**	**Destination**
Discharge Confirmation	Room Database	Discharge Completion	Patient Database
		Patient Information	Patient Database
		Room Information	Room Database

3.13. Update Patient History

Use Case Name	Update Patient History	**ID:** UC - 13	**Priority**	Medium
Actor	Staff			
Description	A doctor requests to update a patient's history. The system searches the patient and allows staff to update patient history			
Triggers	Doctor raises request to update a patient's history			
Type	External			
Preconditions				
1. Staff is logged into the system				

Normal Course	**Information for Steps**
1.0 Request to update patient history	
1. Staff enters MRN of patient in the system	Patient MRN
2. The system displays current Patient History	Patient History
3. Staff adds new medical records of the patient	
4. The system saves the updates	Updated Patient History

Post Conditions
The patient history is updated
Exceptions
E1) The system is unable to connect to the Patient database 1). System displays message that data is currently unavailable 2). System asks staff member to try searching patient on a different time
Summary

Inputs	**Source**	**Outputs**	**Destination**
Patient MRN	Staff	Updated Patient History	Patient Database
Patient History	Patient Database		

3.14. Generate Report

Use Case Name	Generate Report	**ID:** UC - 14	**Priority:**	Medium
Actor	Staff			
Description	A staff requests to generate a report from a set of predefined report templates. The system checks the template and prints the report			
Triggers	Staff raises request to generate report			
Type	External			

Preconditions
1. Staff is authorized to generate a report 2. Staff is logged into the system

Normal Course	**Information for Steps**
1.0 Request to print a report	
1. Staff selects option to generate report on the Homepage ←	Report Request
2. System displays a list of 10 available reports types ←	List of Reports
3. Staff selects a report type ←	Report Type
4. System generates reports as per existing template →	Report

Post Conditions
The system prints the report

Exceptions
E1) The system is unable to connect to the report database 1). System displays message that the data is currently unavailable 2). System asks staff to select the report again 3). Staff re-selects a template or exits the request

Summary

Inputs	**Source**	**Outputs**	**Destination**
Report Request	Staff	Report	Staff
List of Reports	Hospital Database		
Report Type	Staff		

3.15. Check-In InPatient

Use Case Name	Check-In InPatient	**ID:** UC - 15	**Priority**	Medium
Actor	Staff			
Description	A staff checks in an inpatient and records his details			
Triggers	Patient needs to be admitted			
Type	External			
Preconditions				
1. Staff member is authorized to check in a patient 2. Staff member is logged into the system				
Normal Course			**Information for Steps**	
1.0 Request to check in a patient				
1. Staff enters the details of the patient			← Patient Info	
2.Staff selects room requirement details (refer to use case allocate room)			← Room details	
3. System fetches for room status (Alternate Course 1.1)			→ Room Status	
4. Patient is checked in and patient database is updated			→ Updated Patient Database	
Alternate Courses				
1.1 System returns the room status as not available.			→ Room Status	
1.System notifies staff that the request for room allocation is unsuccessful			→ Rejection Notification	
Post Conditions				
1. Patient reflects in room database 2. Patient status changed in patient database				
Exceptions				
E1) The system is unable to connect to the room database 2). System displays message that the data is currently unavailable 3). System asks staff to select the room again				
Summary				

Inputs	**Source**	**Outputs**	**Destination**
Patient Info	Patient Database	Room Status	Room Database
Room Details	Patient Database	Updated patient Database	Patient Database
		Rejection Notification	Staff

4. Sequence Diagram

4.1. Register Patient

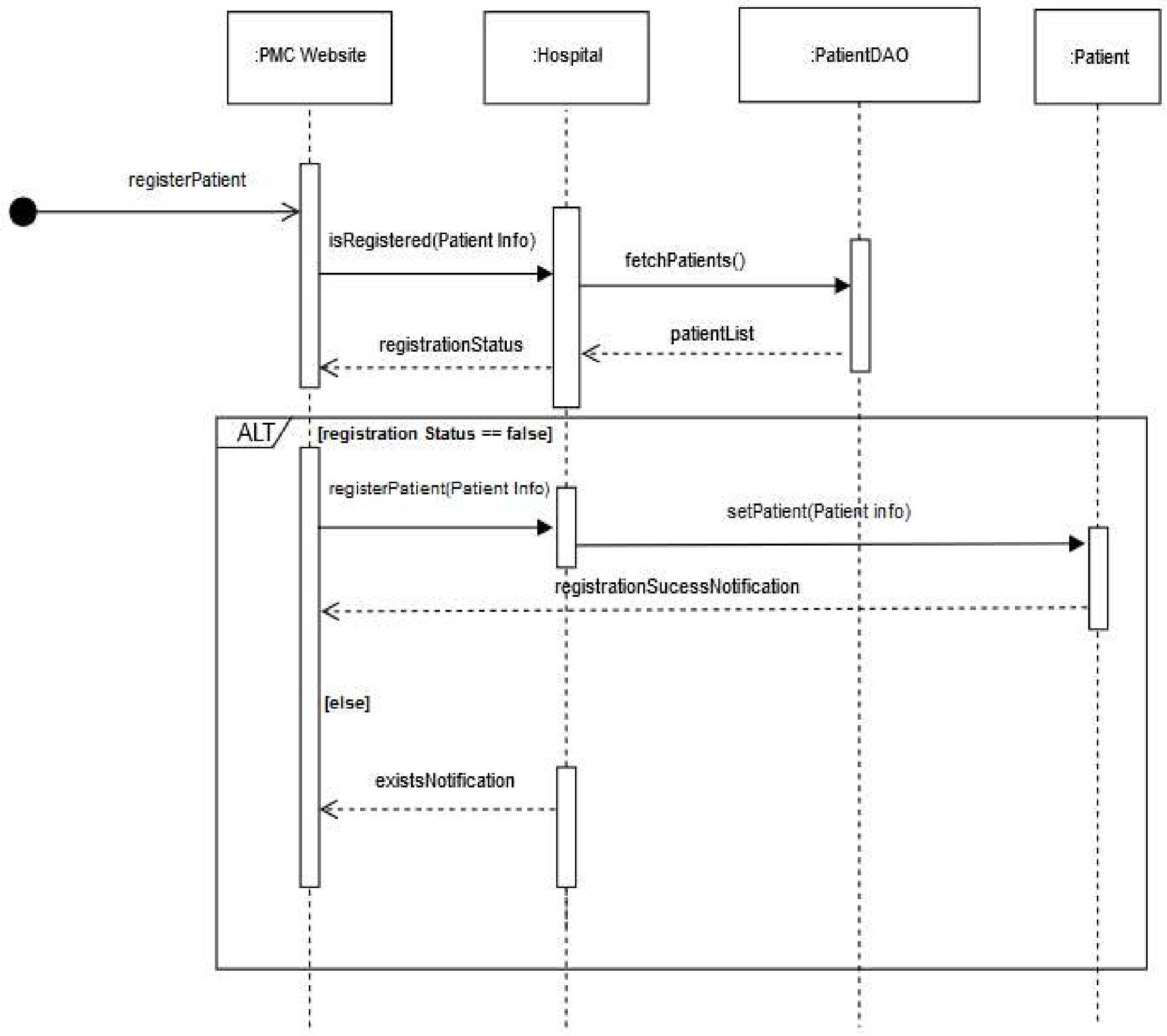


Figure 4. 1 Sequence Diagram for Register Patient

4.2. Book Appointment

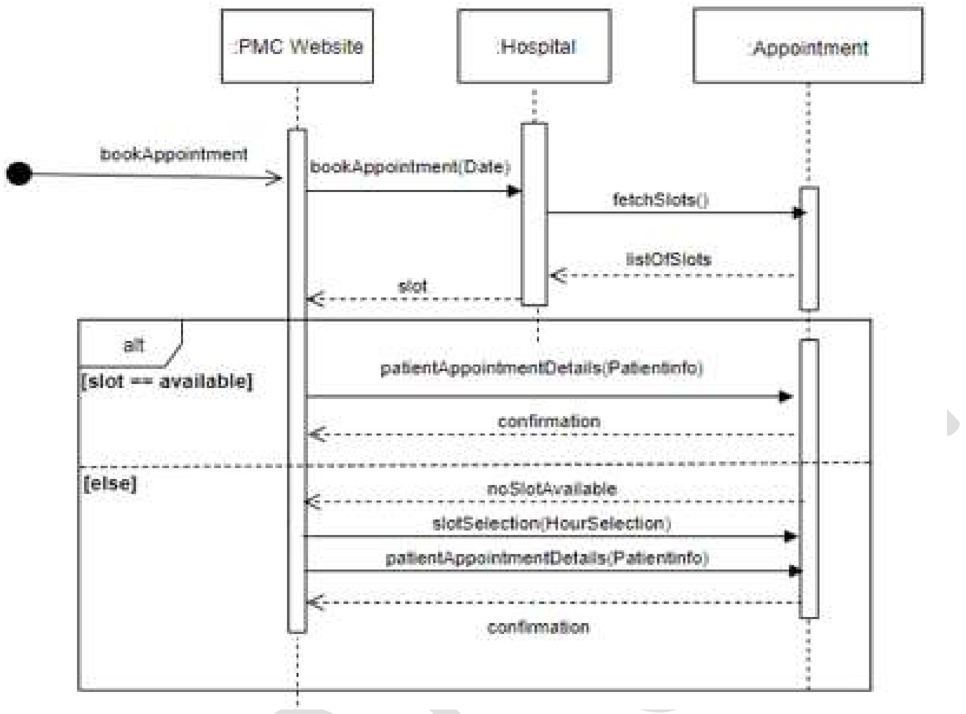


Figure 4. 2 Sequence Diagram for Book Appointment

4.3. Cancel Appointment

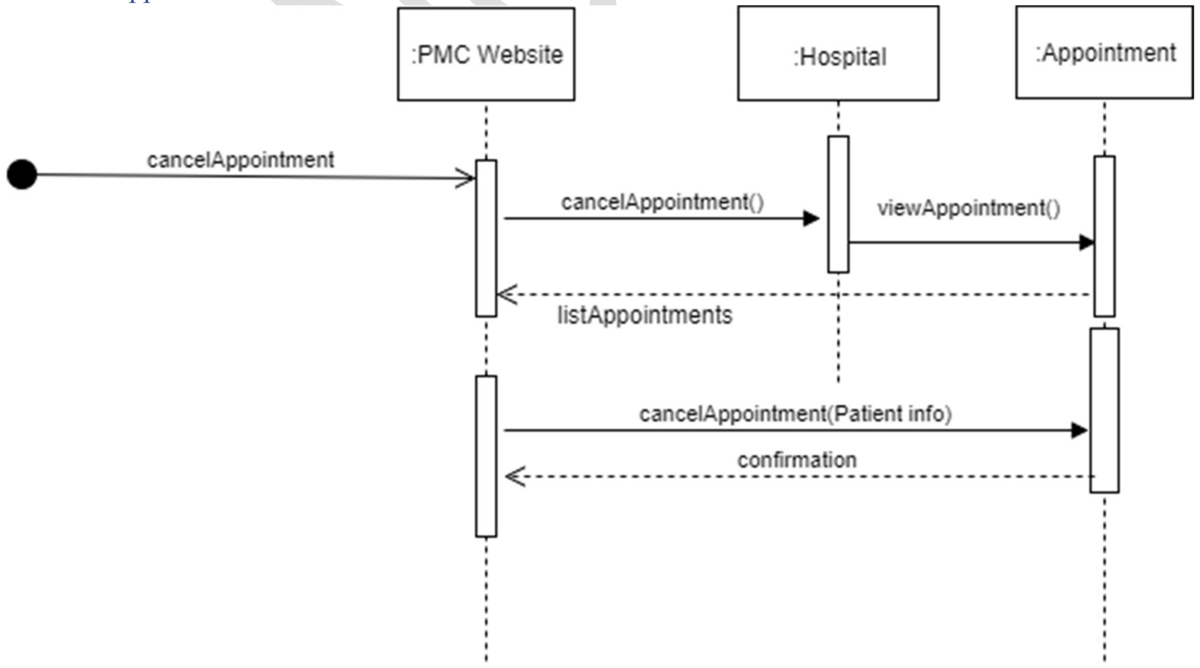


Figure 4. 3 Sequence Diagram for Cancel Appointment

4.4. Allocate Room

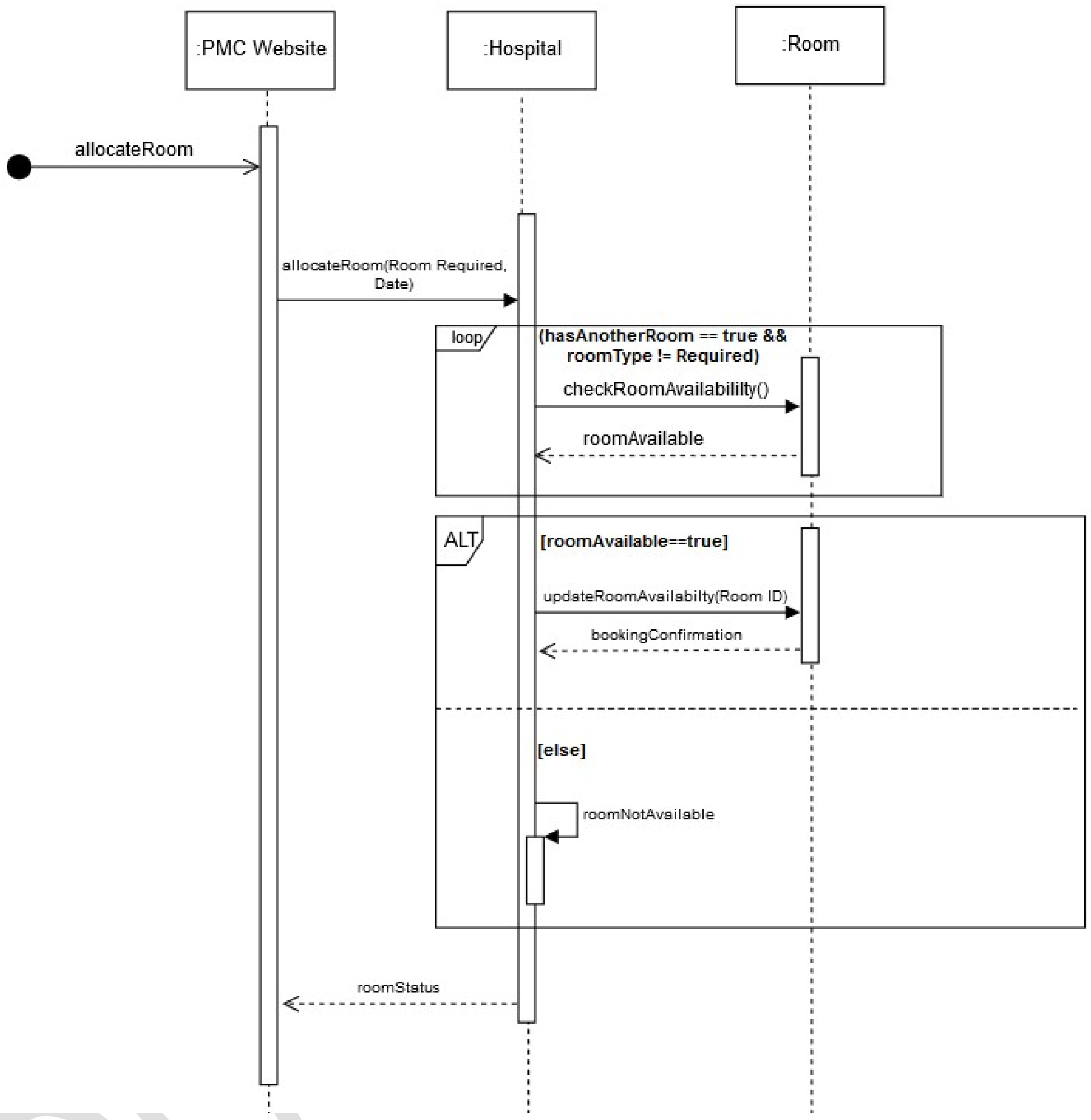


Figure 4. 4 Sequence Diagram for Allocate Room

4.5. Add Doctor

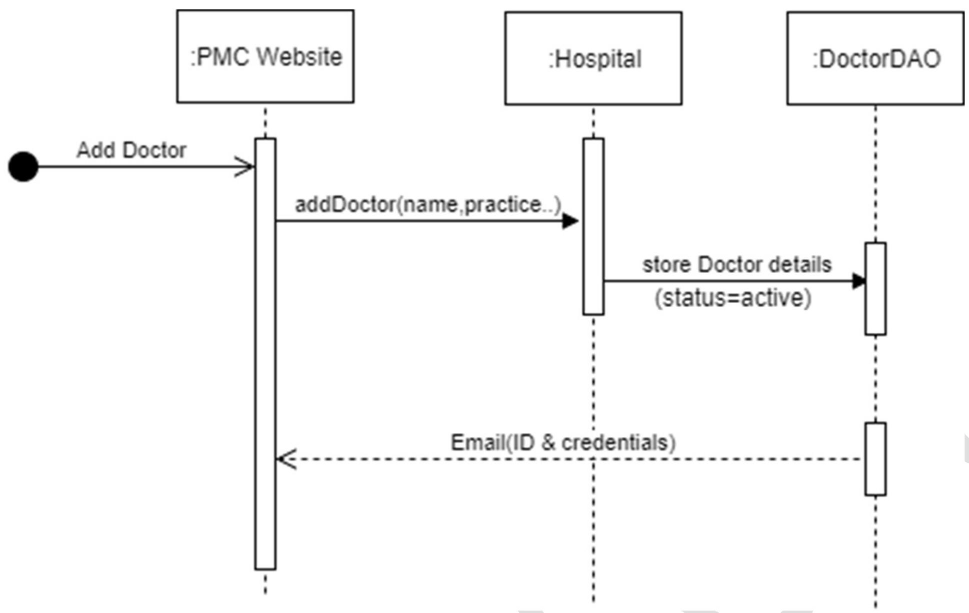


Figure 4. 5 Sequence Diagram for Add Doctor

4.6. Update Doctor Schedule

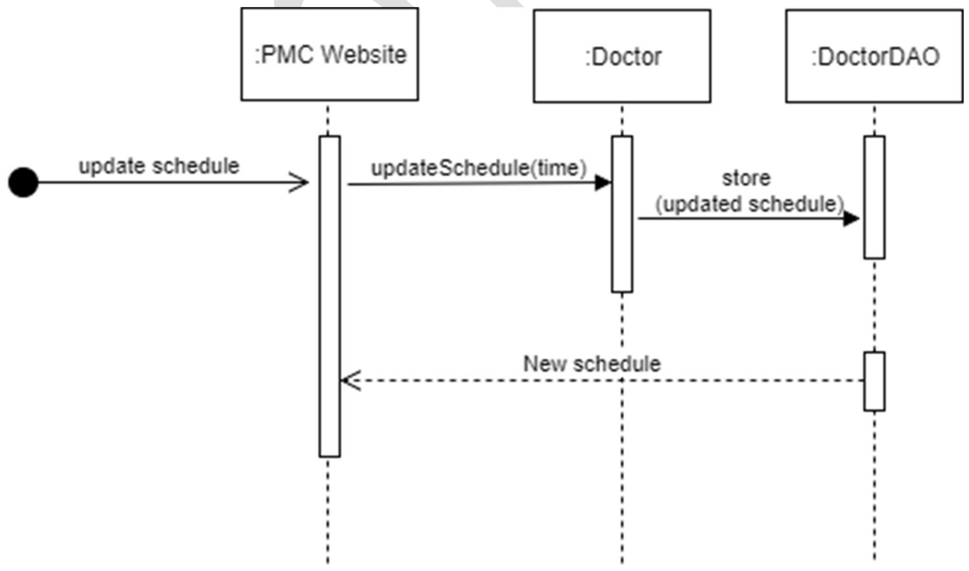


Figure 4. 6 Sequence Diagram for Update Doctor Schedule

4.7. Update Doctor

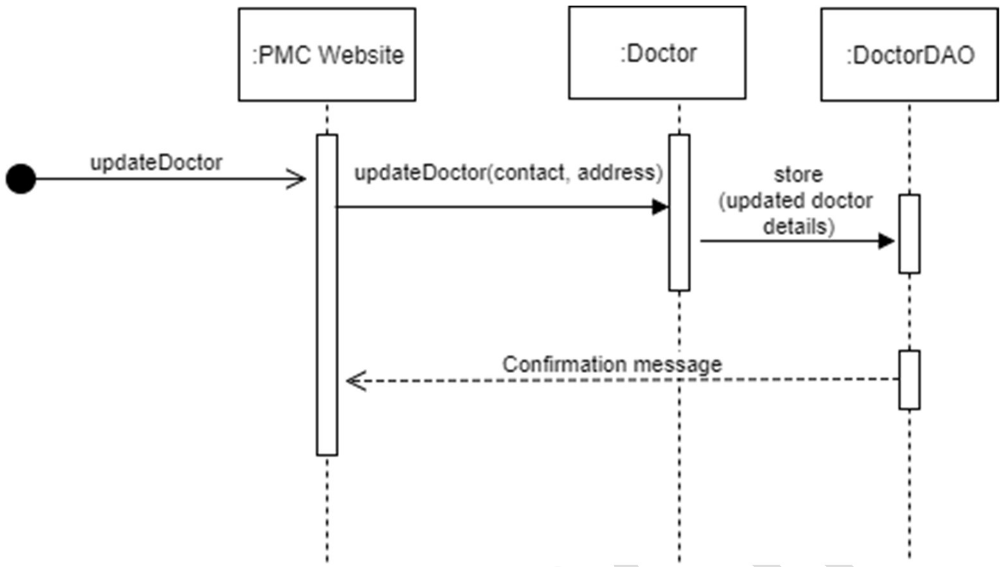


Figure 4. 7 Sequence Diagram for Update Doctor

4.8. Grant Permission

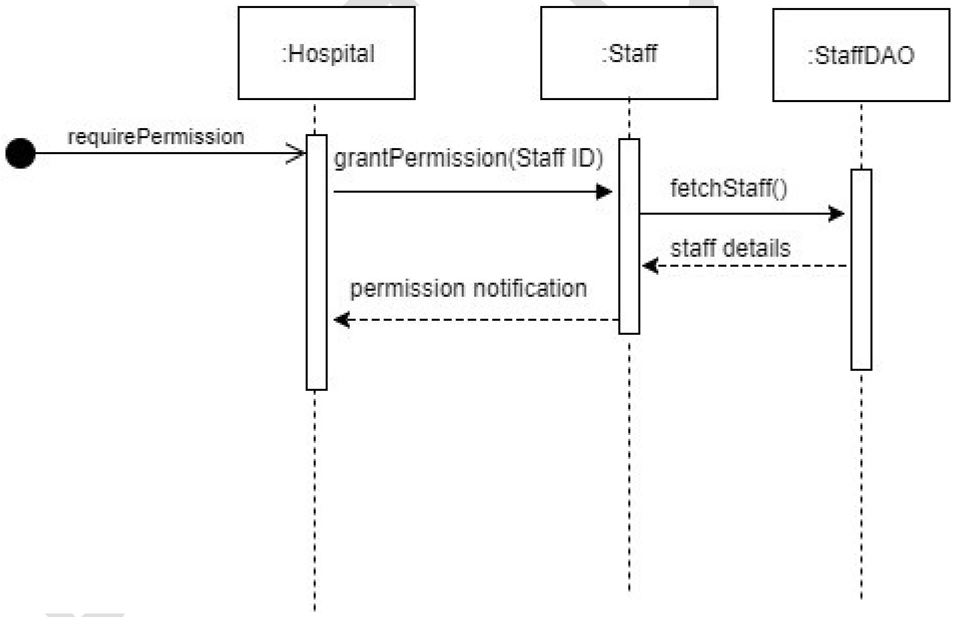


Figure 4. 8 Sequence diagram for Grant Permission

4.9. Add Staff

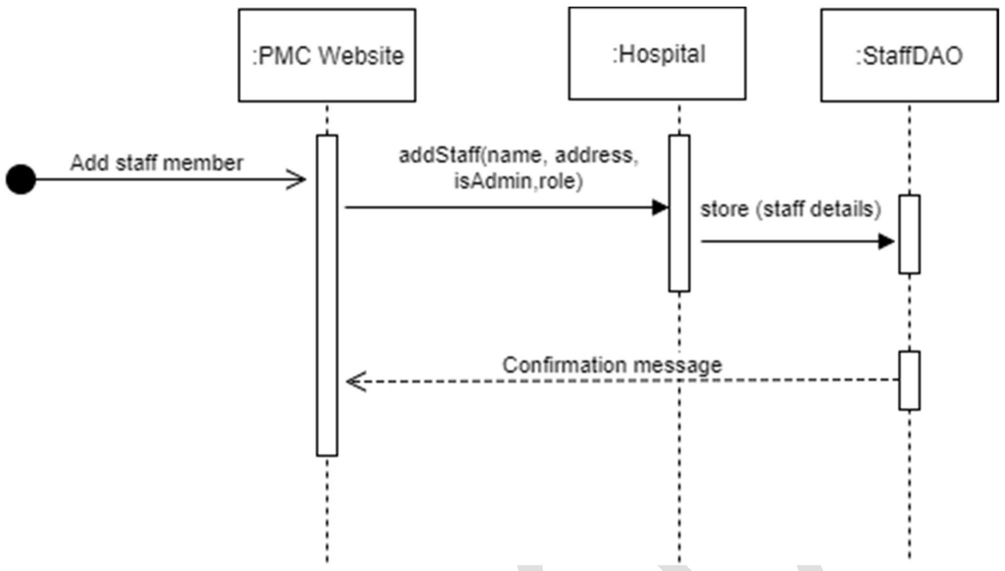


Figure 4. 9 Sequence diagram for Add Staff

4.10. Generate Bill

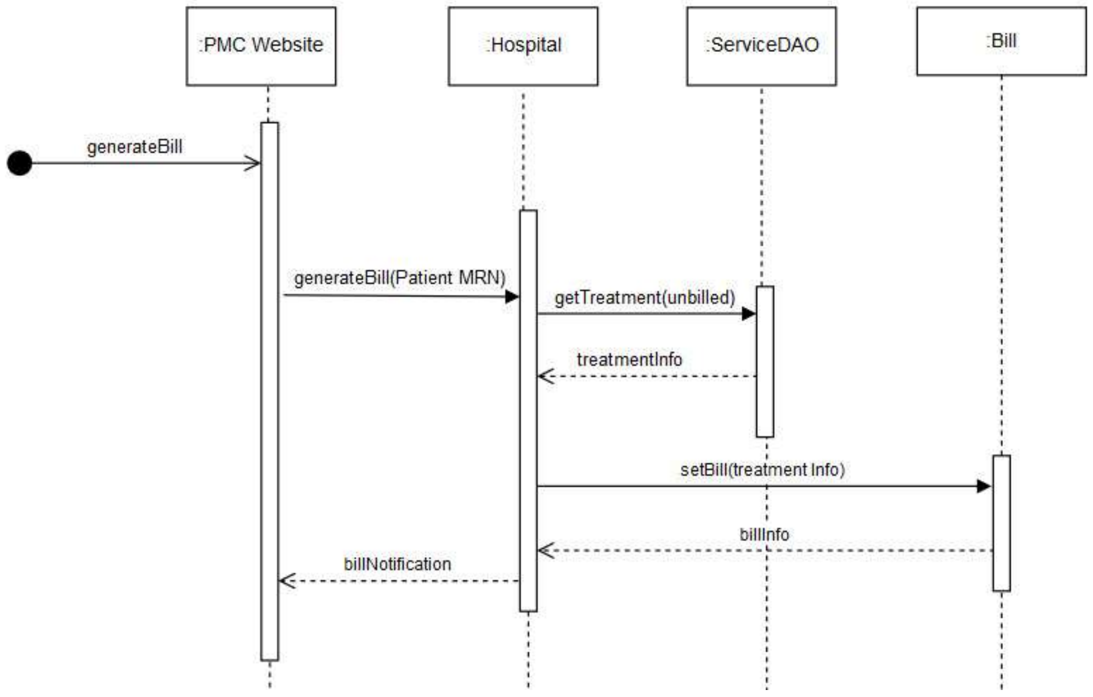


Figure 4. 10 Sequence diagram for Generate Bill

4.11. Pay Bill

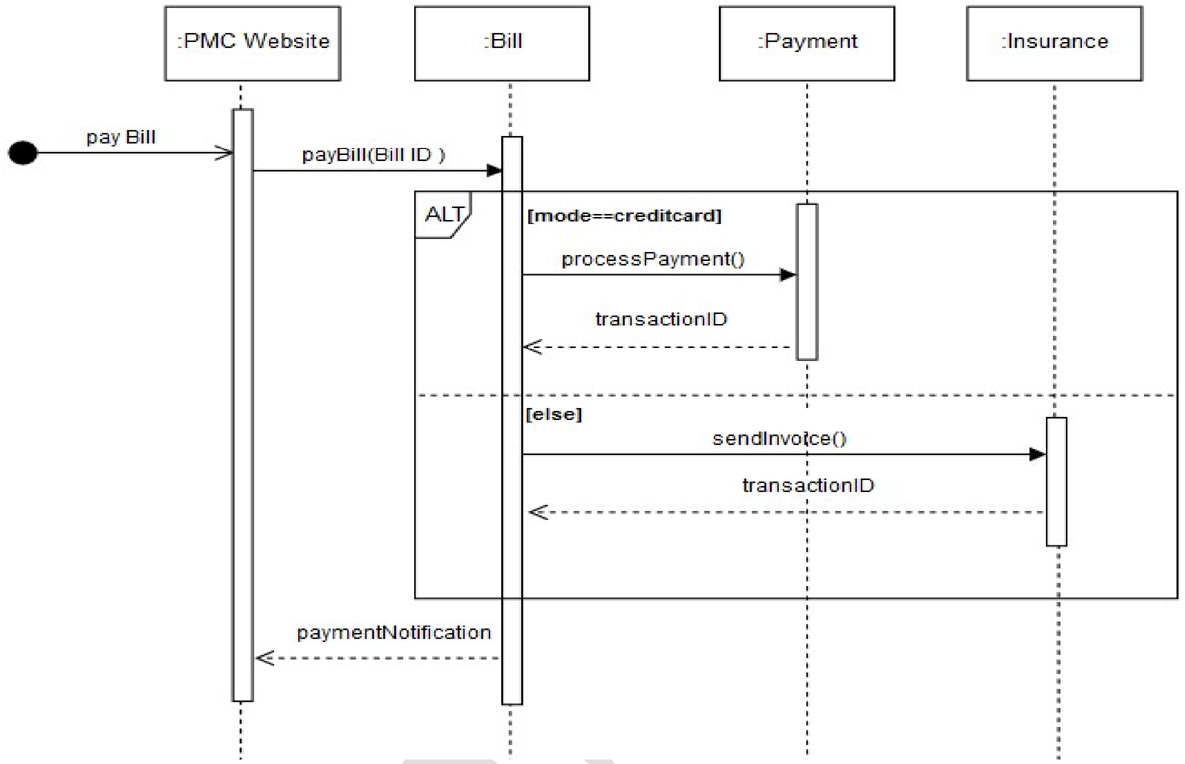


Figure 4. 11 Sequence diagram for Pay Bill

4.12. Discharge Patient

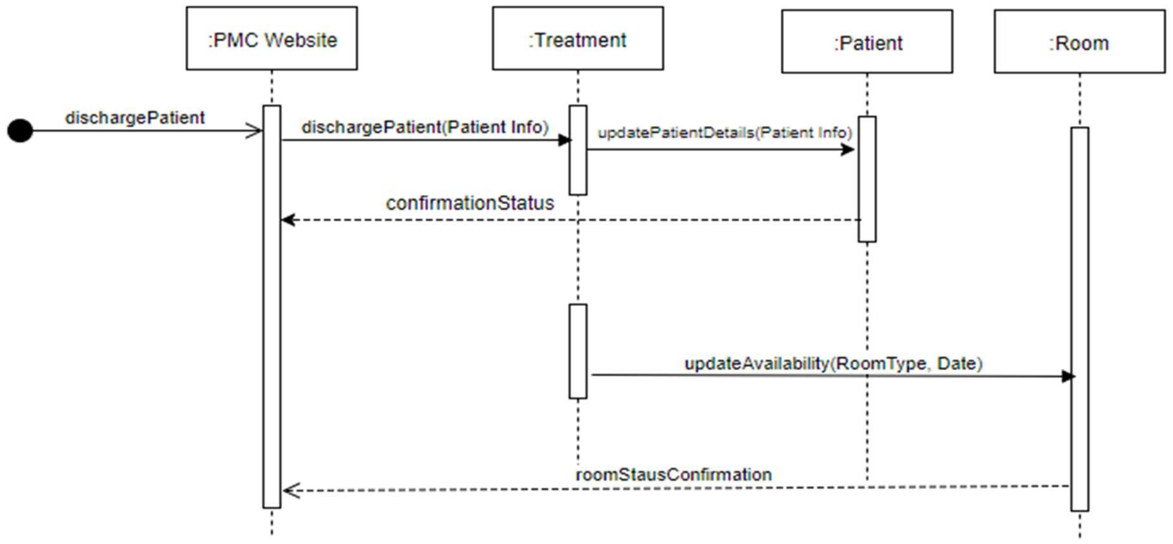

:PMC Website
:Treatment
:Patient
:Room
dischargePatient
dischargePatient(Patient Info)
updatePatientDetails(Patient Info)
confirmationStatus
updateAvailability(RoomType, Date)
roomStausConfirmation

Figure 4. 12 Sequence diagram for Discharge Patient

4.13. Update Patient History

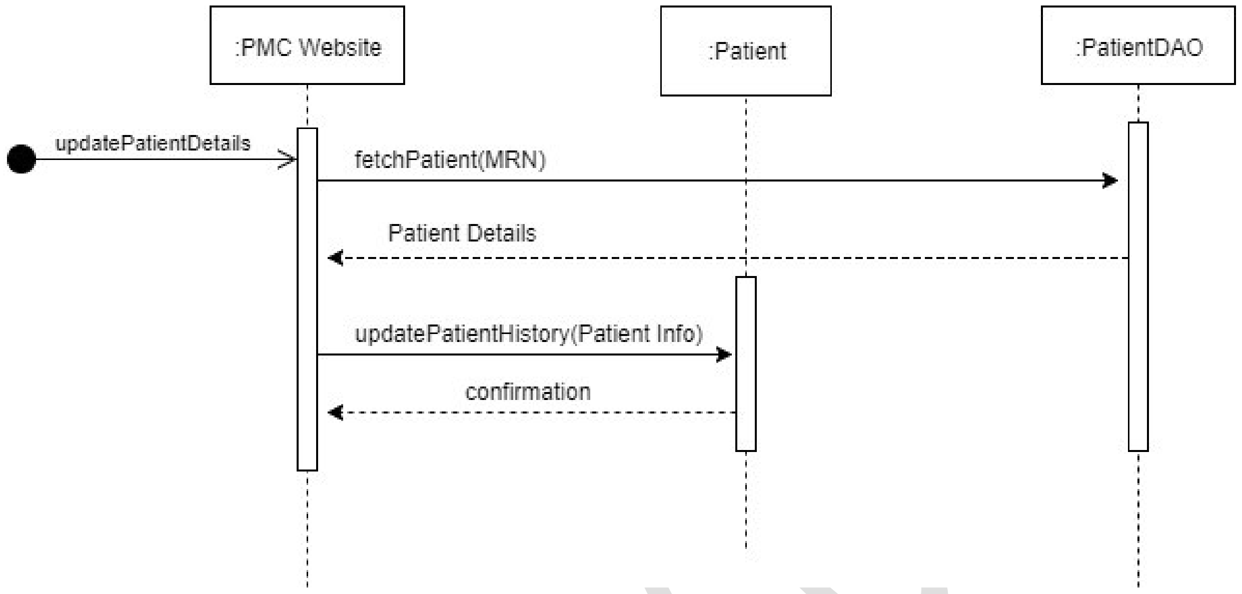


Figure 4. 13 Sequence diagram for Update Patient History

4.14. Generate Report

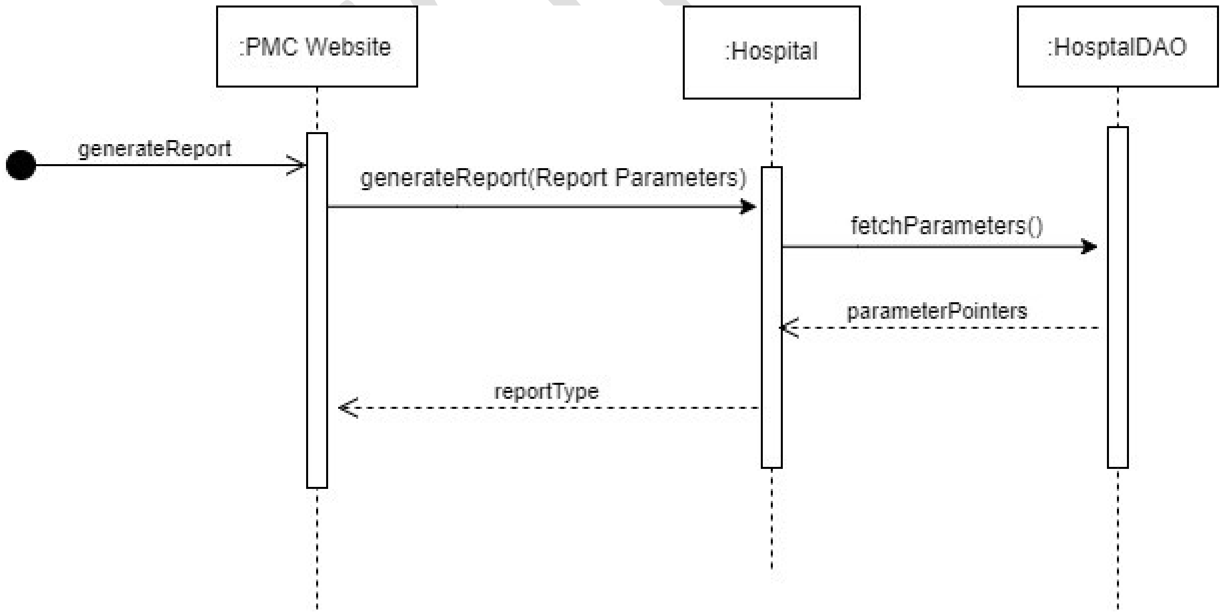


Figure 4. 14 Sequence diagram for Generate Report

4.15. Check-In InPatient

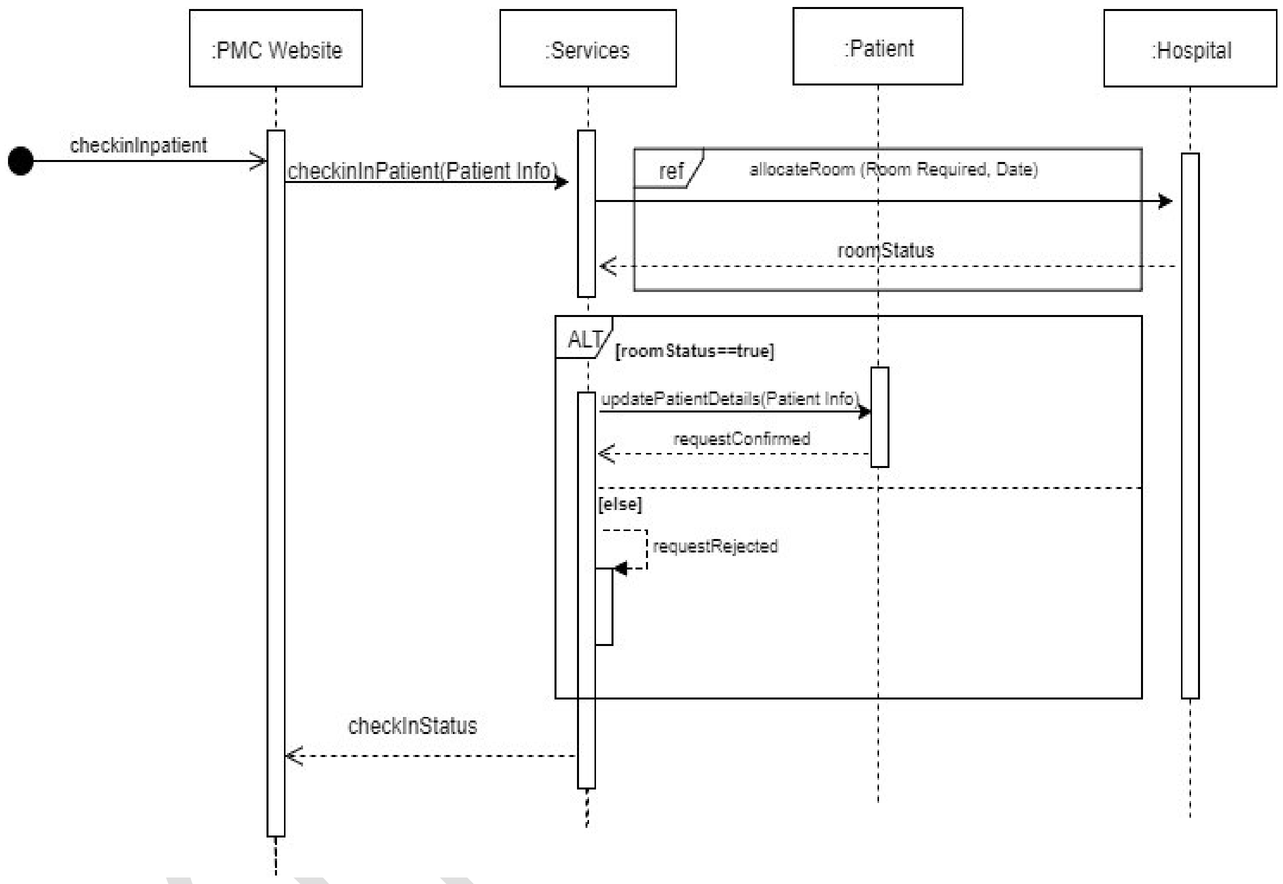


Figure 4. 15 Sequence diagram for Check-In InPatient

5. Schema Diagram

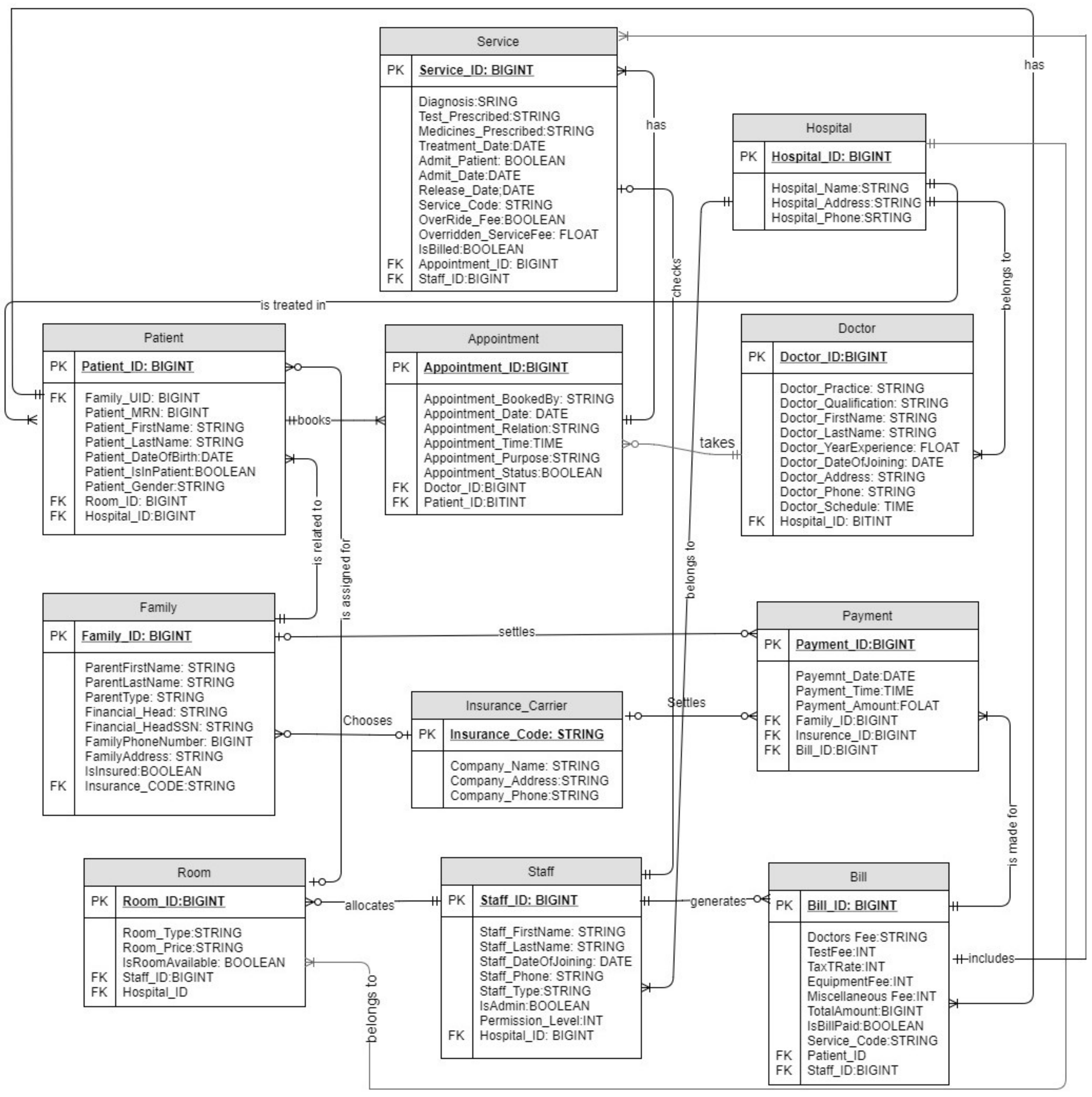


*Figure 5. 1 **Physical Schema Diagram** for the complete Business Case*

6. Class Diagram

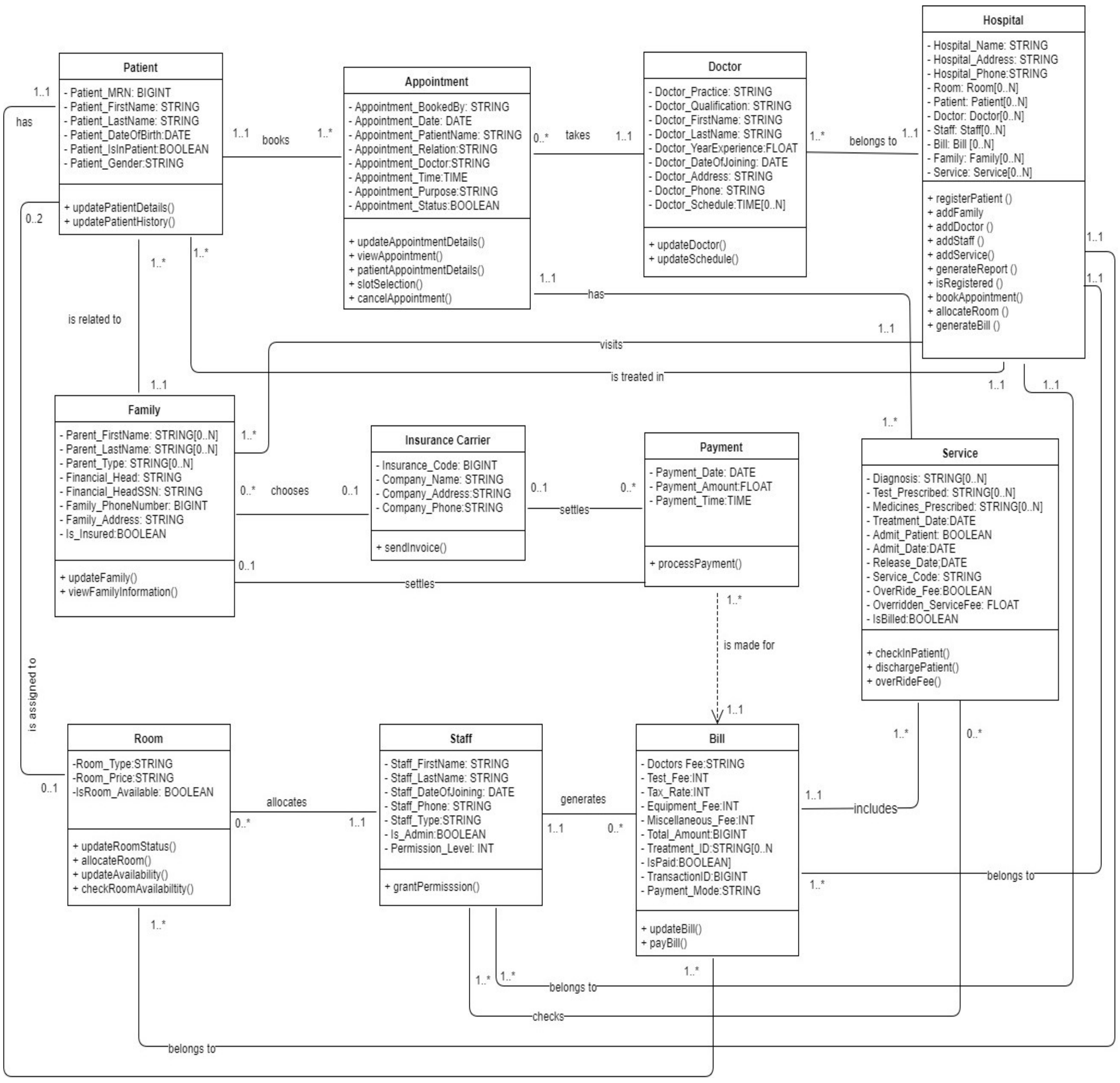


*Figure 6. 1 Class Diagram- **A UML SKETCH***

Note: Due to space constraints, all the parameters passed in methods are shown in sequence diagrams

7. Package Diagram

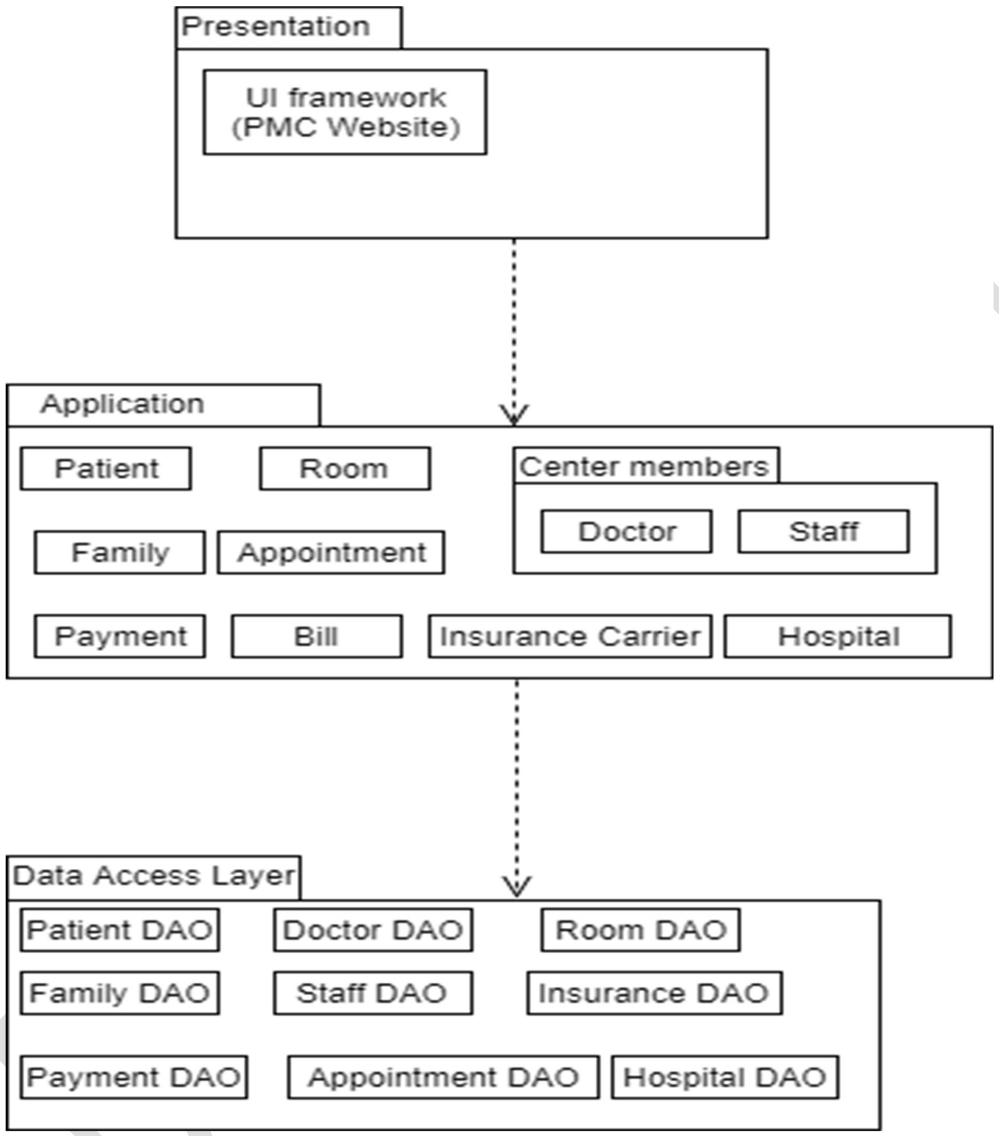


Figure 7. 1 Package Diagram showing presence of classes in different layers of system

8. Deployment Diagram

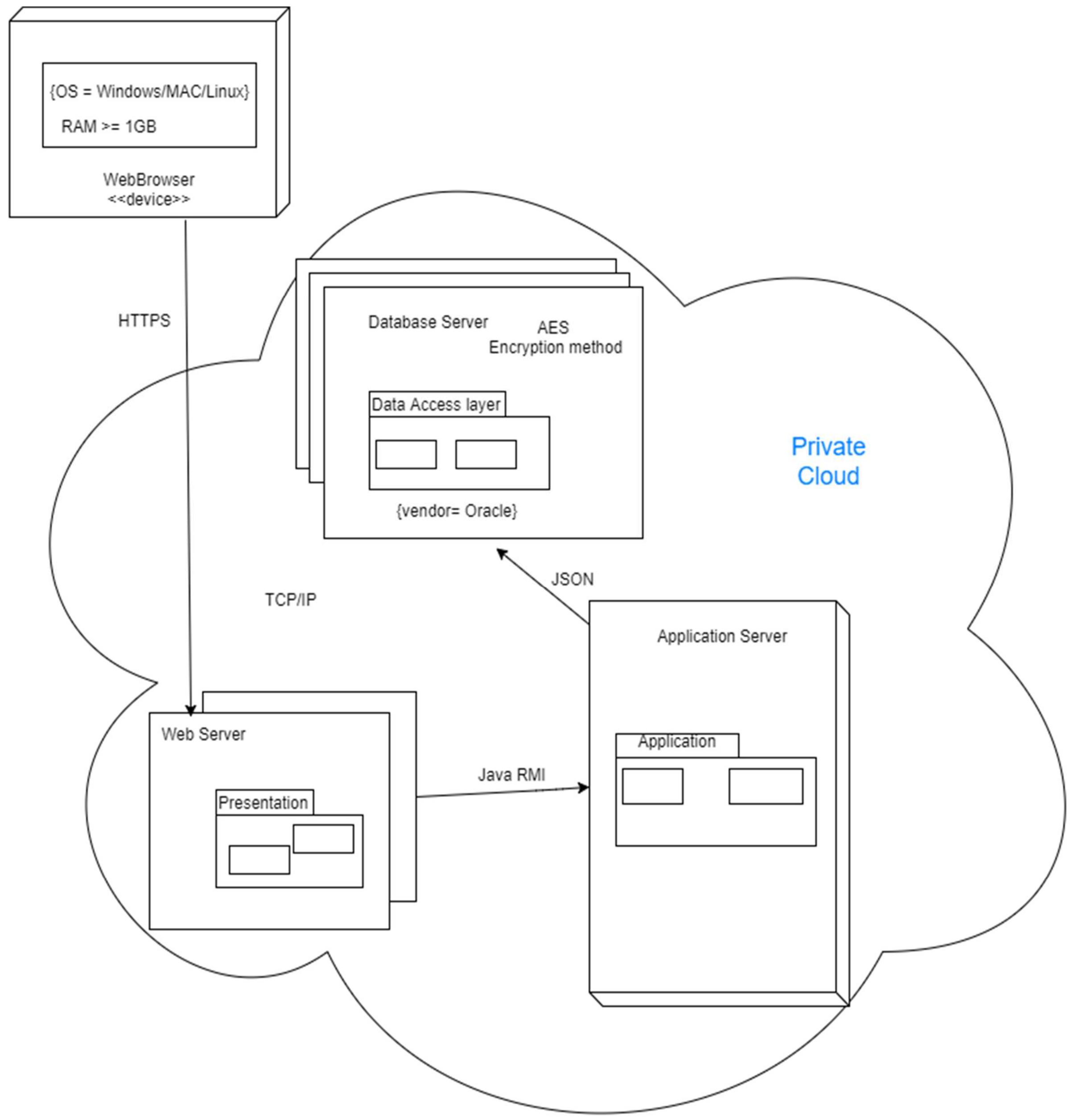


Figure 8. 1 Deployment Diagram showing the architecture of the system and presence of packages at different nodes

We are using Private cloud in order to be complaint with HIPAA so as to protect the confidential information involved in our system.

9. Mockups

9.1. Register Patient

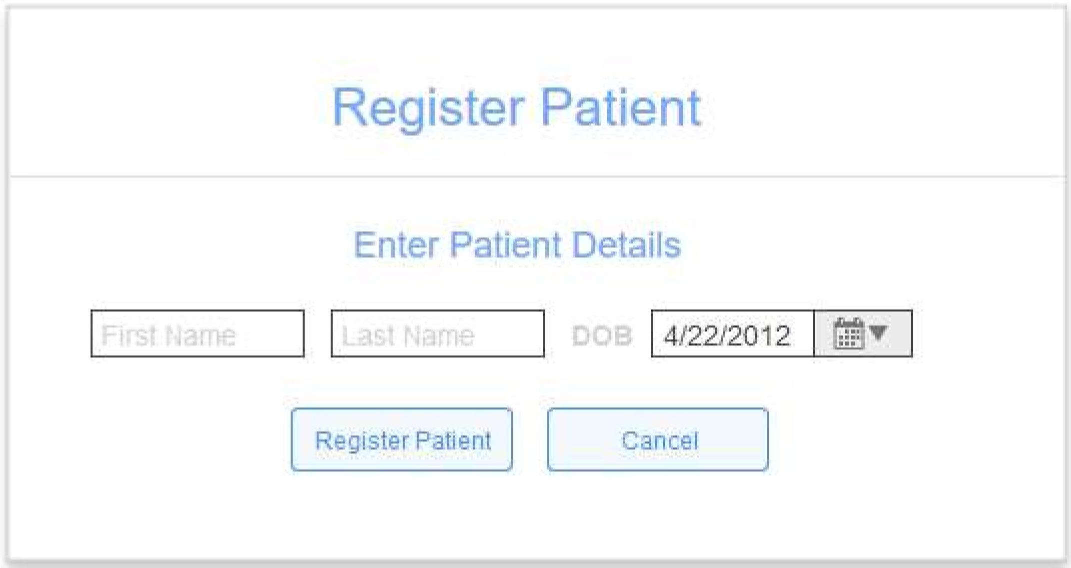
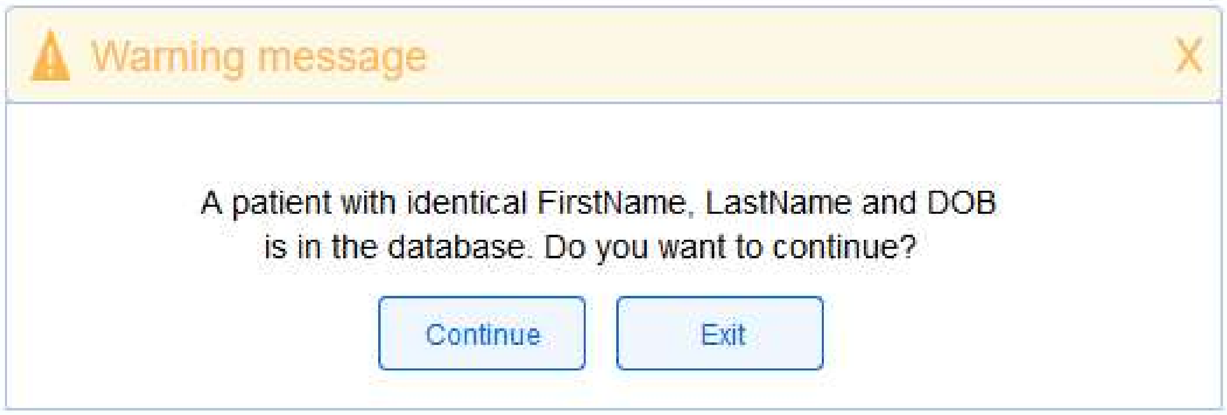
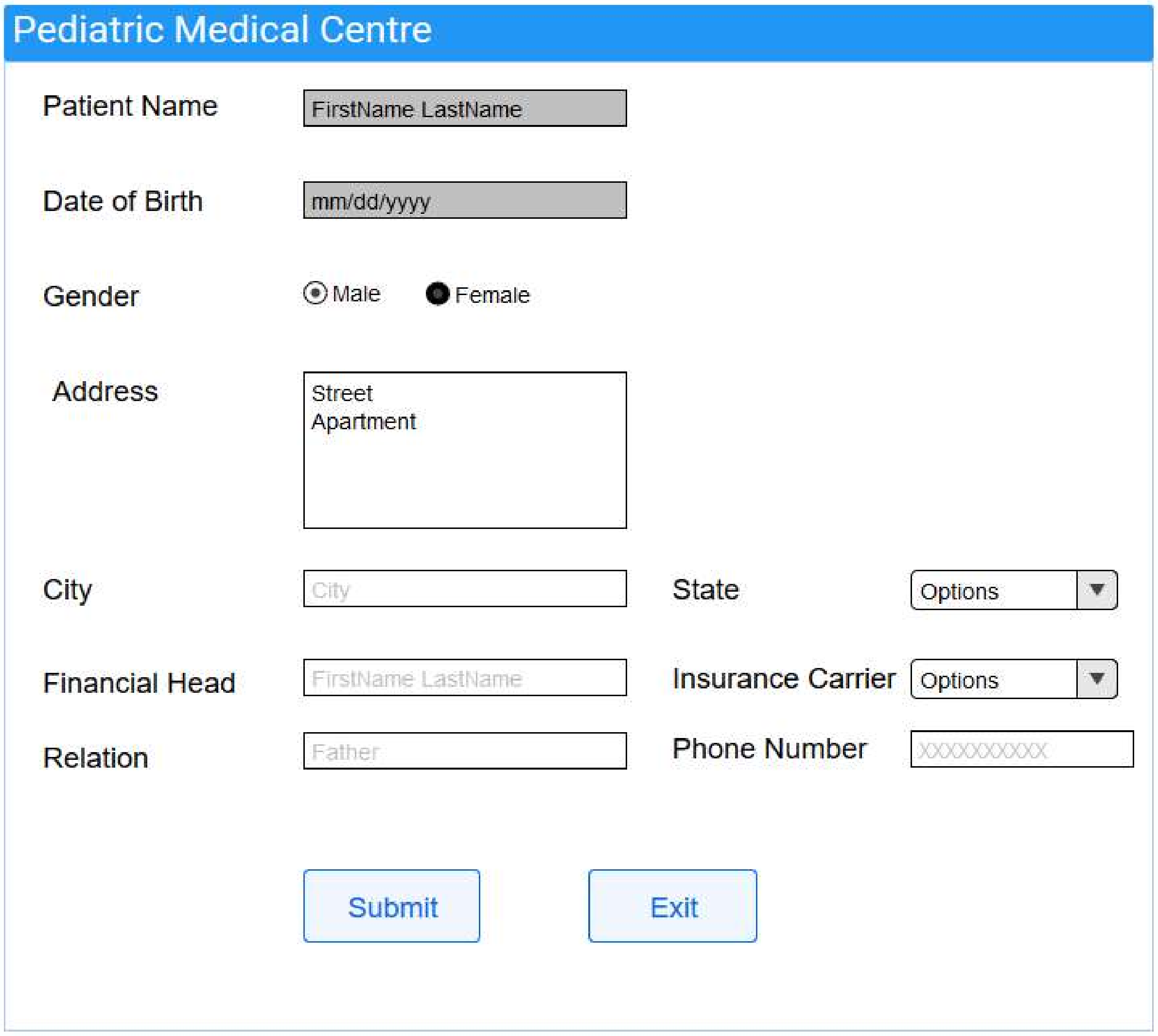


Figure 9.1. 1 Main Screen - Register Patient



A patient with identical FirstName, LastName and DOB is in the database. Do you want to continue?

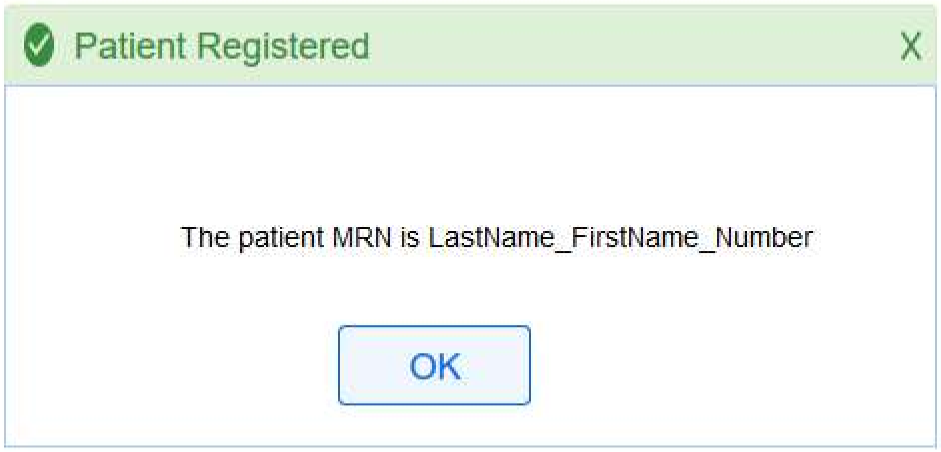
Figure 9.1. 2 Warning Screen - Register Patient



Pediatric Medical Centre

Patient Name	FirstName LastName		
Date of Birth	mm/dd/yyyy		
Gender	◉ Male ● Female		
Address	Street Apartment		
City	City	State	Options ▼
Financial Head	FirstName LastName	Insurance Carrier	Options ▼
Relation	Father	Phone Number	XXXXXXXXXX

Figure 9.1. 3 Form Details - Register Patient



Patient Registered

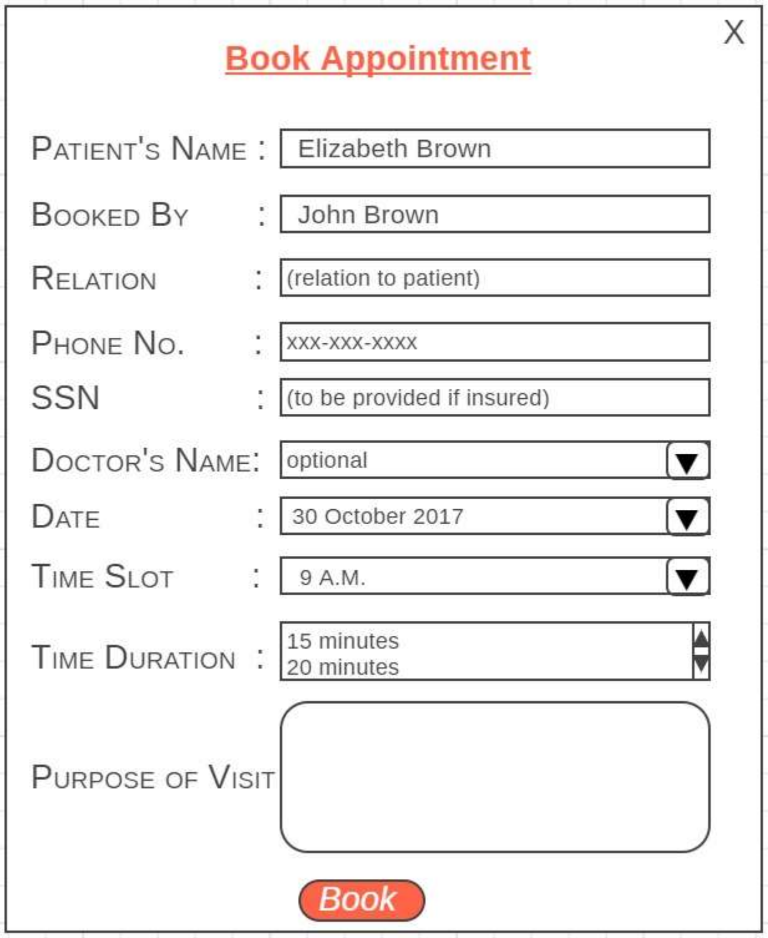
The patient MRN is LastName_FirstName_Number

Figure 9.1. 4 Success Screen - Register Patient

Classes Used

- Hospital
- Patient

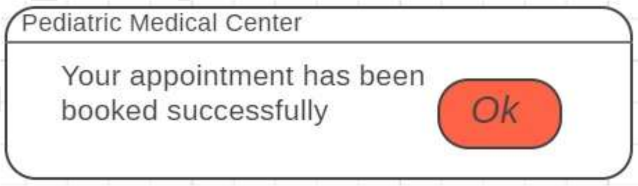
9.2. Book Appointment



Book Appointment

Patient's Name	:	Elizabeth Brown
Booked By	:	John Brown
Relation	:	(relation to patient)
Phone No.	:	xxx-xxx-xxxx
SSN	:	(to be provided if insured)
Doctor's Name	:	optional
Date	:	30 October 2017
Time Slot	:	9 A.M.
Time Duration	:	15 minutes 20 minutes
Purpose of Visit		

Figure 9.2. 1 Form Details - Book Appointment



Pediatric Medical Center

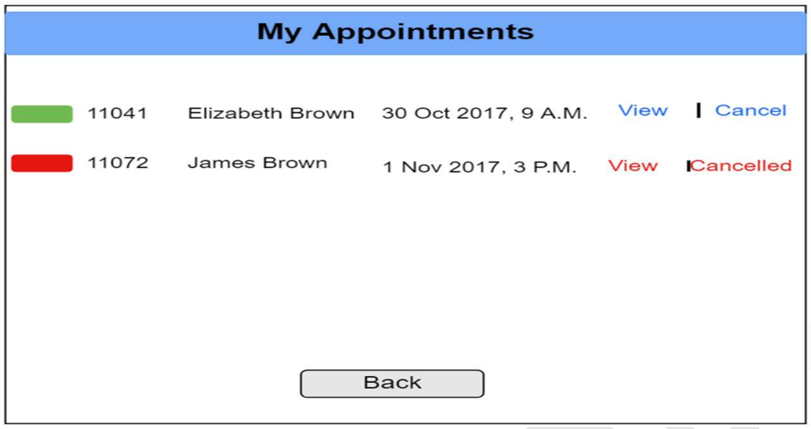
Your appointment has been booked successfully

Figure 9.2. 2 Success Screen Book Appointment

Classes Used

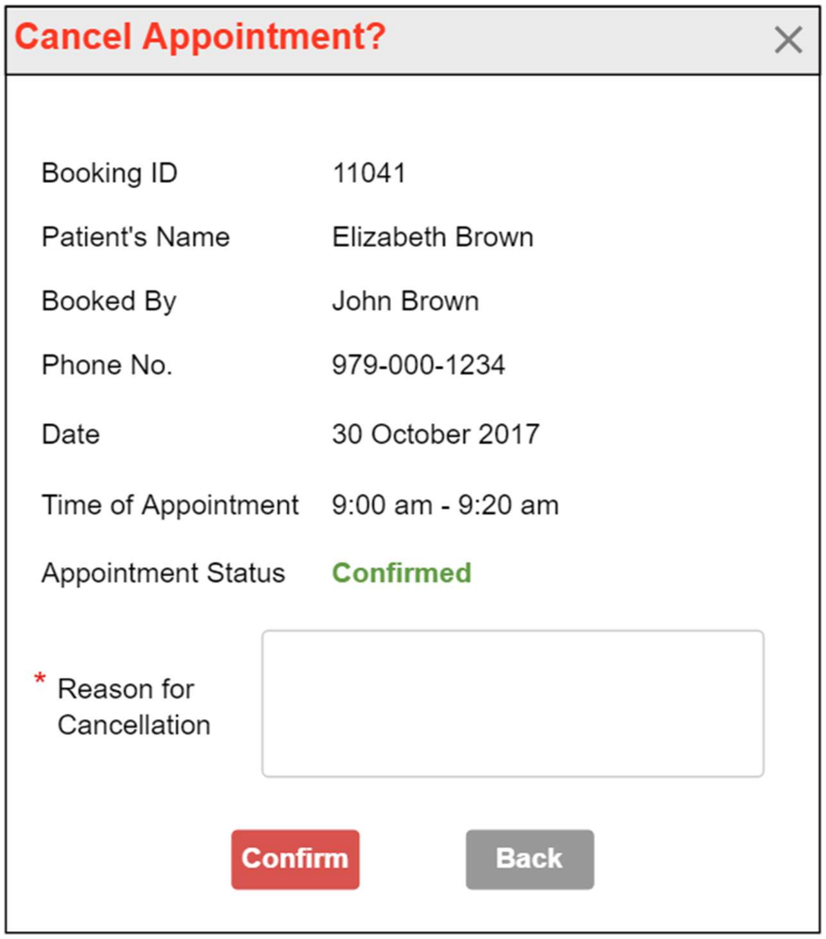
- Hospital
- Appointment

9.3. Cancel Appointment



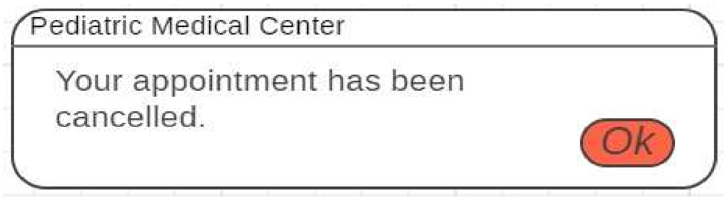
🟩	11041	Elizabeth Brown	30 Oct 2017, 9 A.M.	View	Cancel
🟥	11072	James Brown	1 Nov 2017, 3 P.M.	View	Cancelled

Figure 9.3 1 View Appointments : Cancel Appointments



Booking ID	11041
Patient's Name	Elizabeth Brown
Booked By	John Brown
Phone No.	979-000-1234
Date	30 October 2017
Time of Appointment	9:00 am - 9:20 am
Appointment Status	Confirmed
* Reason for Cancellation	

Figure 9.3 2 Appointment Details - Cancel Appointment


Pediatric Medical Center

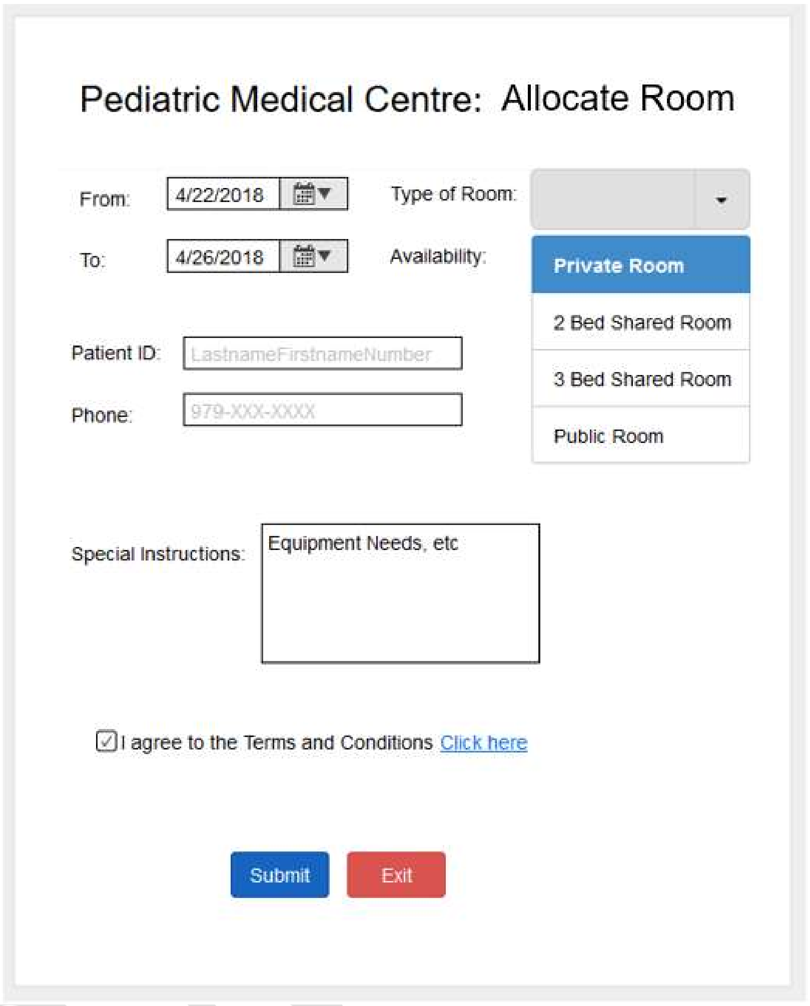
Your appointment has been cancelled.

Figure 9.3 3 Cancel Successful - Cancel Appointment

Classes Used

- Appointment
- Hospital

9.4. Allocate Room



Pediatric Medical Centre: Allocate Room

From:	4/22/2018	Type of Room:	
To:	4/26/2018	Availability:	
Patient ID:	LastnameFirstnameNumber		
Phone:	979-XXX-XXXX		
Special Instructions:	Equipment Needs, etc		

☑ I agree to the Terms and Conditions Click here

Figure 9.4. 1 Allocate Room

Classes Used

- Room
- Hospital

9.5. Login Screen

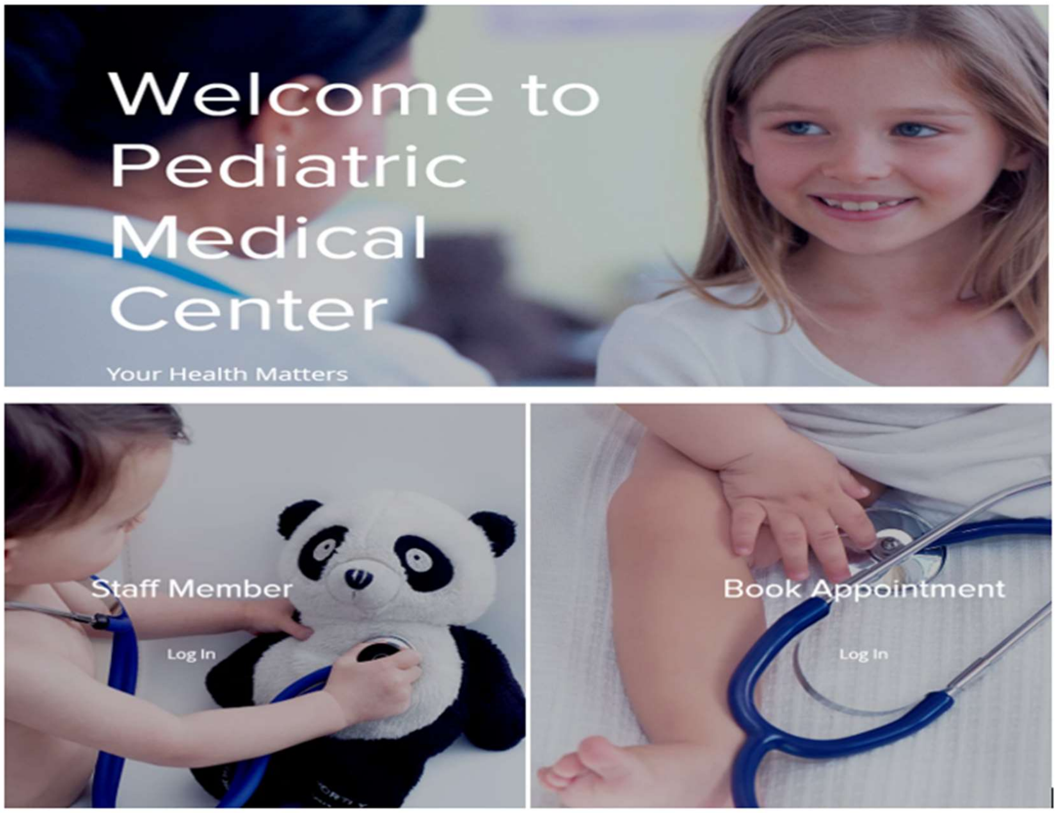


Figure 9.5. 1 Homepage

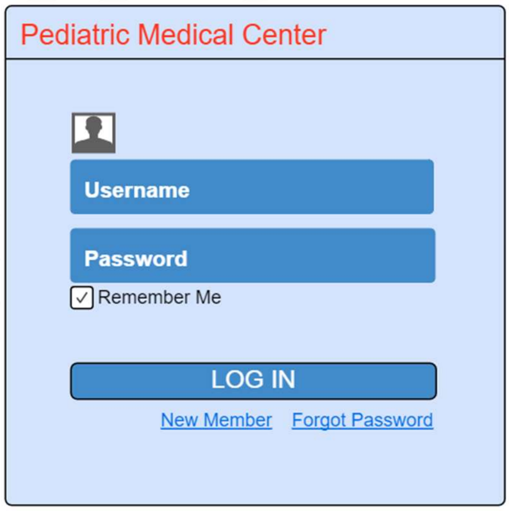


Figure 9.5. 2 Login Page

Classes Used

- PMC Website

10. Testing Plan

10.1 Testing Strategy

Requirement Number	Corresponding Use Case	Testing Strategy	Type of Testing	How to test	When to test
2.1.1	Register Patient	Unit Testing	White Box	Test the code for register patient module. Test the code for edge cases, valid, invalid and null values. The UI is unambiguous and conveinient to use.	At the end of each iteration
		Integration Testing	Black Box	Check if registration is successfull, notification is generated and database is updated. Patients details are stored accurately and are visible only to authorized staff. **Classes Checked:** Hospital, Patient	At the end of each iteration
2.1.2	Book Appointment	Unit Testing	White Box	Test the code for book appointment module. Test the code for edge cases, valid, invalid and null values. The UI is unambiguous and conveinient to use.	At the end of each iteration
		Integration Testing	Black Box	Check if appointment is booked, notification is generated and database is updated. **Classes Checked**: Appointment, Hospital.	At the end of each iteration
2.1.3	Cancel Appointment	Unit Testing	White Box	Test the code for cancel appointmentt module. Test the code for edge cases, valid, invalid and null values. The UI is unambiguous and conveinient to use.	At the end of each iteration
		Integration Test	Black Box	Check if appointment is cancelled notification is generated and database is updated. Check the application on various browsers for navigability from Homepage. **Classes Checked**: Appointment, Hospital.	At the end of each iteration
2.1.4	Allocate Room	Unit Testing	White Box	Check if room is allocated and its status is updated in database. Check accurate information is sent to database.	At the end of each iteration
		Integration Test	Black Box	Check if room is allocated and its status is updated in database. Check accurate information is sent to database. **Classes Checked**: Hospital, Room	At the end of each iteration
2.1.5	Add Doctor	Unit Testing	White Box	Test the code for adding doctor module. Test the code for valid (doctor's information), invalid (null) values. The UI is unambiguous and conveinient to use.	At the end of each iteration
		Integration Test	Black Box	Check if the hospital class points properly to doctor classs. So whenever, a new doctor information is added by adminmistrator it is being saved in doctor database, credentials sent via email and doctor's details are visible in his portal. **Classes Checked**: Hospital and Doctor.	At the end of each iteration
2.1.6	Update Doctor Schedule	Unit Testing	White Box	Test the code for update doctor schedule module. Test the code for valid (future dates and times), invalid(like trying to update schedule for already passed days) and null values. The UI is unambiguous and conveinient to use.	At the end of each iteration
		Integration Test	Black Box	Check if doctor is able to access and update his schedule from his portal. Check accurate information is sent to database and updated details are visible in doctor's portal. **Classes Checked**: Doctor.	At the end of each iteration
2.1.7	Update Doctor	Unit Testing	White Box	Test the code for update doctor module. Test the code for valid (Doctor's information), invalid(like trying enter numbers in doctor's name or letters in contact number) and null values. The UI is unambiguous and conveinient to use.	At the end of each iteration
		Integration Test	Black Box	Check if doctor is able to access and update his details from his portal. Check accurate information is sent to database and updated details are visible in doctor's portal. **Classes Checked**: Doctor	At the end of each iteration

Requirement Number	Corresponding Use Case	Testing Strategy	Type of Testing	How to test	When to test
2.1.8	Grant Permission	Unit Testing	White Box	Test the code for register patient module. Test the code for edge cases, valid, invalid and null values. The UI is unambiguous and conveinient to use.	At the end of each iteration
		Integration Test	Black Box	Check accurate information is sent to database **Classes Checked**: Hospital, Staff	At the end of each iteration
2.1.9	Add Staff	Unit Testing	White Box	Test the code for adding staff module. Test the code for valid(staff member details like name, role, address, contact etc), invalid(letters in contact etc) and null values. The UI is unambiguous and conveinient to use.	At the end of each iteration
		Integration Test	Black Box	Check whether the entered information is sent to staff database and visible for the staff member's portal. Also, Hospital class points to staff class properly. **Classes Checked**: Hospital and Staff.	At the end of each iteration
2.1.10	Generate Bill	Unit Testing	White Box	Test the code for register patient module. Test the code for edge cases, valid, invalid and null values. The UI is unambiguous and conveinient to use.	At the end of each iteration
		Integration Test	Black Box	Check accurate information is sent to database **Classes Checked:** Hospital, Treatment, Bill	At the end of each iteration
2.1.11	Pay Bill	Unit Testing	White Box	Test the code for register patient module. Test the code for edge cases, valid, invalid and null values. The UI is unambiguous and conveinient to use.	At the end of each iteration
		Integration Test	Black Box	Check if bill is getting paid separately or altogether i.e. insurance plus self paid by cash, card or everything self paid either cash or card. This is getting updated in the database properly. **Classes Checked:** Bill, Payment, Insurance	At the end of each iteration
2.1.12	Discharge Patient	Unit Testing	White Box	Test the code for register patient module. Test the code for edge cases, valid, invalid and null values. The UI is unambiguous and conveinient to use.	At the end of each iteration
		Integration Test	Black Box	Check if the patient is discharged and the records are updated in the database. **Classes Checked**: Treatment, Patient, Room	At the end of each iteration
2.1.13	Update Patient History	Unit Testing	White Box	Test the code for register patient module. Test the code for edge cases, valid, invalid and null values. The UI is unambiguous and conveinient to use.	At the end of each iteration
		Integration Test	Black Box	Check if the medical record history is updated and stored in the database properly. **Classes Checked**: Patient	At the end of each iteration
2.1.14	Generate Report	Unit Testing	White Box	Test the code for register patient module. Test the code for edge cases, valid, invalid and null values. The UI is unambiguous and conveinient to use.	At the end of each iteration
		Integration Test	Black Box	Check whether the report includes all the fields with data saved in the database **Classes Checked**: Hospital	At the end of each iteration
2.1.15	Check-In InPatient	Unit Testing	White Box	Test the code for register patient module. Test the code for edge cases, valid, invalid and null values. The UI is unambiguous and conveinient to use.	At the end of each iteration
		Integration Test	Black Box	Check whether the patient is checked in with all formalities in placec including room allocation if required and same is getting saved in the database **Classes Checked**: Services, Patient, Hospital	At the end of each iteration

Requirement Number	Testing Strategy	Type of Test	How	When to test
2.2.1	System Testing	Black Box	Run the PMC Website on the various systems(different OS) to check how well it works . Test each and every module of the system.	At the last iteration
2.2.2	System Testing	Black Box	The system is tested by operating it in various browsers(Edge, Chrome etc.) to see how it performs.	At the last iteration
2.2.3	System Testing	Black Box	The system is tested by running on various devices like mobile phones, tablet computers etc. to check the functioning.	At the last iteration
2.2.4	Performance Testing	Black Box	Modules are tested to see if updates in database reflect within 3 secs from data entry	Done in each iteration when individual modules are created. Then in the iterations where the modules are combined. Once a complete system is created the performance testing is done again
2.2.5	Performance Testing	Black Box	Modules are tested to see if the response time is 3 secs from the time valid inputs are submitted in the UI	Done in each iteration when individual modules are created. Then in the iterations where the modules are combined. Once a complete system is created the performance testing is done again
2.2.6	Stress Testing	Black Box	Modules are tested to see if they can support 100 concurrent users	Done in each iteration when individual modules are created. Then in the iterations where the modules are combined. Once a complete system is created the performance testing is done again
2.2.7 - 2.2.8	Security Testing	Black Box	Dummy access (credentails) of staff members (including administrator) and doctor will be created and tested by testers if access rights provided to members are as per their permission level and only concerned information is displayed in their portal which can be accessed using credentials.	At last iteration and every month post deployment of the system
2.2.9	Security Testing	Black Box	Client web browsers should be connected to server via https and proper anti-virus should be installed in the system. These can be checked by visibility of https with link whenever personal system is connected to PMC website and if using only http the website should not be displayed on screen. Some virus can be introduced to see proper working of anti-virus softwares	At last iteration and every month post deployment of the system
2.2.10	Security Testing	Black Box	Patient's data must be protected everytime- AES encryption has been used for the data at rest in databases. Also the https encrypted data is transferred whenever accessed via server. Audit control is also in place to record all the activities carried out on PMC website and the audit can be checked by one authority person aware of HIPAA complaince concerns.	At last iteration and every month post deployment of the system
2.2.11	Cultural Testing	Black Box	When clicked on Language option as Spanish, the website shows all information in spanish. A translator proficient in Spanish will check and verify whether the information is presented correctly in Spanish	At the last iteration
2.2.12	Maintainability Testing	Black Box	Whenever the system is integrated with a new developed module, it should work without breaking the existing modules of the system and the same is checked by the testers.	At the end of each iteration
2.2.13	AvailabiltyTesting	Black Box	System's data is present in 3 databases. In case one goes down we have the backup to ensure availability. Tester will switch off one database server and check if the site and all functionalities still work with the correct and latest data, which means the system is automatically connecting to the next database server (backup).Also, one notification is received by administrator whenever system goes down.	This will be tested every week post system development

10.2 Testing Tools

Automation Tools: Selenium
Bug Reporting Tool: JIRA

10.3 Test Cases

This section provides demonstration of how modules will be tested. We have used test case examples of two modules from two classes each as below -

Class: Hospital

Module 1: Book Appointment

Module 2: Allocate Room

Class: Doctor

Module 1: Update Doctor

Module 2: Update Doctor Schedule

10.3.1 Book Appointment

Unit Testing

The code will be tested for any programming errors.

We will then test the code for edge cases, valid, invalid and null values. These cases are mentioned below. **The green areas in the table denote valid inputs.**

Module: bookAppointment() **Version:** 1

Tester: **Test Design Date:** 11/20/17 **Tested Date:**

Test ID: 002

Task: Validate bookAppointment()method of Hospital Class

Objective: bookAppointment()method is functioning as expected

Test cases:

#	Patient's name	Booked By	Relation	Phone No.	SSN	Doctor's Name	Date	Time Slot	Pass/ Fail
1	Myra Grant								Pass
2	!quest M90								Fail
3	78777								Fail
4	NULL								Fail
5		Elijah Grant							Pass
6		!quest M90							Fail
7		78777							Fail
8		NULL							Fail
9			Father						Pass
10			Mother						Pass
11			NULL						Fail
12			1234						Fail
13			Tunes						Fail
14				121212abc					Fail
15				979-224-7011					Pass
16				000-000-0000					Pass
17				-90					Fail
18				NULL					Fail
19					111-11-9879				Pass
20					NULL				Fail
21					Acd-dd-ier2				Fail
22						Hugh Walker			Pass
23						!quest M90			Fail
24						78777			Fail
25						NULL			Fail
26						****			Fail

Integration Testing

Type: Bottom-up Testing – Individual modules have been unit tested and then the modules are integrated and tested together

The below classes are tested together:

Class 1: Hospital

Modules Involved: bookAppointment()

Class 2: Appointment

Modules Involved: patientDetails(), slotSelection()

Stress Testing

Appointment is booked by 100 concurrent users

Performance Testing

Once valid inputs have been provided, the appointment information should be saved in the database and the user must be notified within 3 seconds of clicking submit button.

System Testing

Once the information system is completed the bookAppointment is tested to check if it meets its functional requirement number - 2.1.2 and the nonfunctional requirements from Requirement Number to Requirement Number 2.2.1 to 2.2.13.

User Acceptance Testing

Once the information system is completed, bookAppointment module is thoroughly tested in the development environment in order to check if a user is able to book an appointment.

10.3.2 Allocate Room

Unit Testing

The code will be tested for any programming errors.

We will then test the code for edge cases, valid, invalid and null values. These cases are mentioned below. **The green areas in the table denote valid inputs.**

Module: allocateRoom() **Version:** 1

Tester: **Test Design Date:** 11/20/17 **Tested Date:**

Test ID: 001

Task: Validate allocateRoom() method of Hospital Class

Objective: allocateRoom() method is functioning as expected

Test cases:

#	From date	To Date	Type of Room	Patient ID	Phone	Expected Result
1	a/b/cccc					Fail
2	-					Fail
3	NULL					Fail
4	00/00/0000					Fail
5	11/20/2017					Pass
6	02/30/2017					Fail
7		a/b/cccc				Fail
8		-				Fail
9		NULL				Fail
10		00/00/0000				Fail
11		11/20/2017				Pass
12		02/30/2017				Fail
13			Private Room			Pass
14			Public Room			Pass
15			2 Bed Shared Room			Pass
16			3 Bed Shared Room			Pass
17			123			Fail
18			NULL			Fail
19				1098		Pass
20				#@!@		Fail
21				12!!!		Fail
22				NULL		Fail
23					121212abc	Fail
24					979-224-7011	Pass
25					000-000-0000	Fail
26					-90	Fail

Integration Testing

Type: Bottom-up Testing – Individual modules have been unit tested and then the modules are integrated and tested together

The below classes are tested together:

Class 1: Hospital

Modules Involved: allocateRoom()

Class 2: Room

Modules Involved: checkRoomAvailability(), updateRoomAvailability()

Stress Testing

10 separate rooms are booked concurrently

Performance Testing

Once valid inputs have been provided, the room allocation information should be saved in the database and the user must be notified within 3 seconds of clicking submit button.

System Testing

Once the information system is completed the allocateRoom() is tested to check if it meets its functional requirement number 2.1.4 - and the nonfunctional requirements from Requirement Number to Requirement Number 2.2.1 to 2.2.13

User Acceptance Testing

Once the information system is completed, allocateRoom module is thoroughly tested in the development environment in order to check if the staff is able to allocate a room to patient.

10.3.3 Update Doctor

Unit Testing

The code will be tested for any programming errors.

We will then test the code for edge cases, valid, invalid and null values. These cases are mentioned below. **The green areas in the table denote valid inputs.**

Module: updateDoctor() **Version:** 1

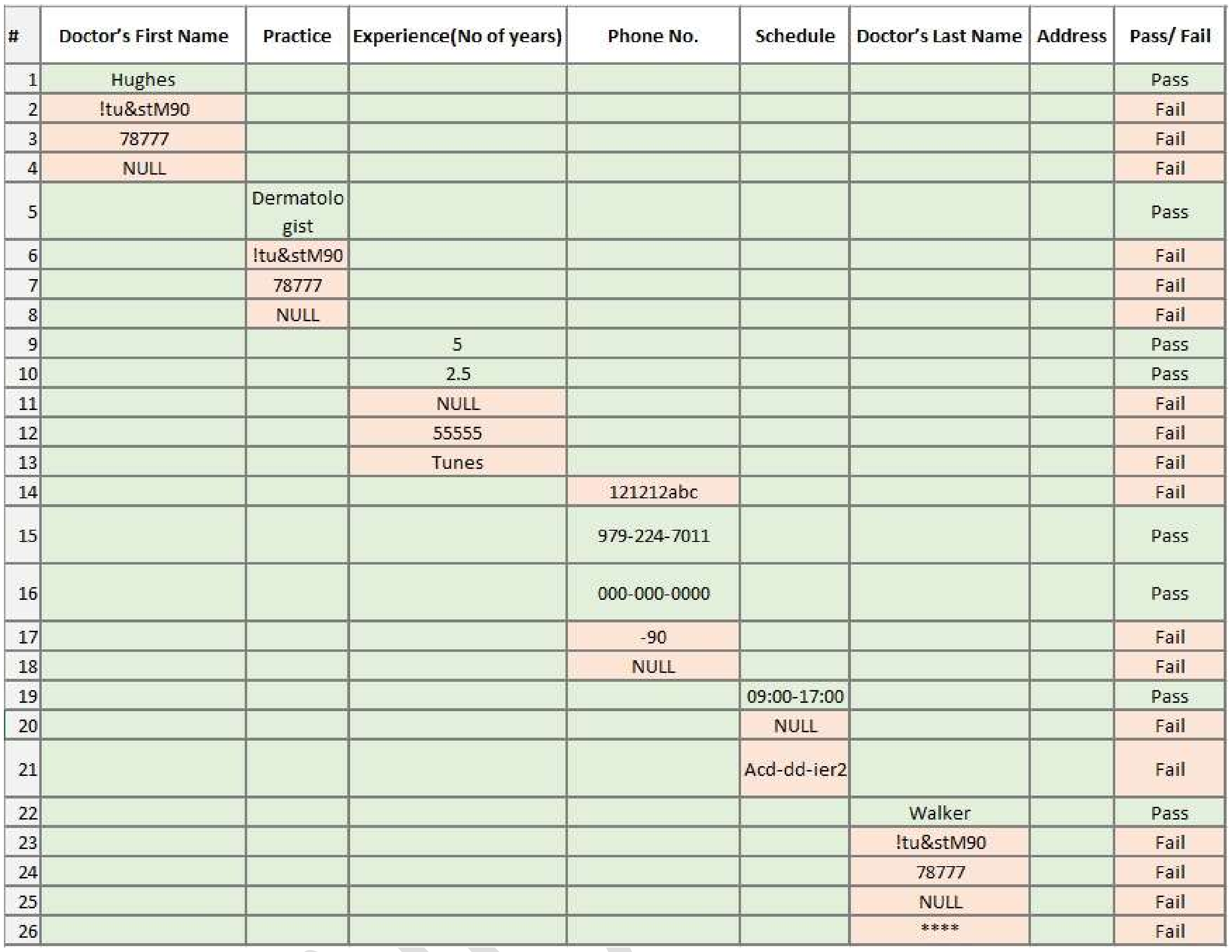
Tester: **Test Design Date:** 12/2/17 **Tested Date:**

Test ID: 003

Task: Validate updateDoctor() method of Doctor Class

Objective: updateDoctor() method is functioning as expected

Test cases:



#	Doctor's First Name	Practice	Experience(No of years)	Phone No.	Schedule	Doctor's Last Name	Address	Pass/ Fail
1	Hughes							Pass
2	!tu&stM90							Fail
3	78777							Fail
4	NULL							Fail
5		Dermatologist						Pass
6		!tu&stM90						Fail
7		78777						Fail
8		NULL						Fail
9			5					Pass
10			2.5					Pass
11			NULL					Fail
12			55555					Fail
13			Tunes					Fail
14				121212abc				Fail
15				979-224-7011				Pass
16				000-000-0000				Pass
17				-90				Fail
18				NULL				Fail
19					09:00-17:00			Pass
20					NULL			Fail
21					Acd-dd-ier2			Fail
22						Walker		Pass
23						!tu&stM90		Fail
24						78777		Fail
25						NULL		Fail
26						****		Fail

Integration Testing

Type: Bottom-up Testing – Individual modules have been unit tested and then the modules are integrated and tested together

The below classes are tested:

Class 1: Doctor

Modules Involved: updateDoctor()

The PMC website interacts with the Doctor class to test if the correct data is being updated or not.

Stress Testing

The doctor details are updated by multiple staff members concurrently.

Performance Testing

Once valid inputs have been provided, the doctor details should be saved in the database and the change should be reflected within 3 seconds.

System Testing

Once the information system is completed, the updateDoctor module is tested to check if it meets its functional requirement number - 2.1.7 and the Nonfunctional Requirements from Number 2.2.1 to 2.2.13

User Acceptance Testing

Once the information system is completed, the updateDoctor module is thoroughly tested in the development environment in order to check if the doctor details are updated properly.

10.3.4 Update Doctor Schedule

Unit Testing

The code will be tested for any programming errors.

We will then test the code for edge cases, valid, invalid and null values. These cases are mentioned below. **The green areas in the table denote valid inputs.**

Module: updateSchedule() **Version:** 1

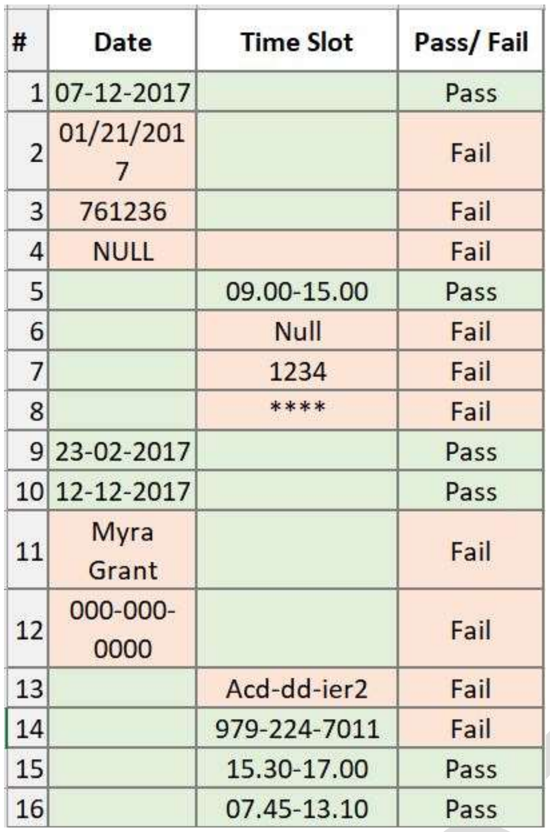
Tester: **Test Design Date:** 11/20/17 **Tested Date:**

Test ID: 004

Task: Validate updateSchedule() method of Hospital Class

Objective: updateSchedule()method is functioning as expected

Test cases:



#	Date	Time Slot	Pass/ Fail
1	07-12-2017		Pass
2	01/21/2017		Fail
3	761236		Fail
4	NULL		Fail
5		09.00-15.00	Pass
6		Null	Fail
7		1234	Fail
8		****	Fail
9	23-02-2017		Pass
10	12-12-2017		Pass
11	Myra Grant		Fail
12	000-000-0000		Fail
13		Acd-dd-ier2	Fail
14		979-224-7011	Fail
15		15.30-17.00	Pass
16		07.45-13.10	Pass

Integration Testing

Type: Bottom-up Testing – Individual modules have been unit tested and then the modules are integrated and tested together

The below classes are tested together:

Class: Doctor

Modules Involved: updateSchedule()

Stress Testing

Appointment is booked by 100 concurrent users

Performance Testing

Once valid inputs have been provided, the appointment information should be saved in the database and the user must be notified within 3 seconds of clicking submit button.

System Testing

Once the information system is completed the updateSchedule module is tested to check if it meets its functional requirement number - 2.1.6 and the nonfunctional requirements from Requirement Number to Requirement Number 2.2.1 to 2.2.13

User Acceptance Testing

Once the information system is completed, the updateSchedule module is thoroughly tested in the development environment in order to check if the doctor is able to update schedule properly.