Hospital Management System

Description:
Hospitals face challenges in managing large volumes of patient data, appointments, billing, and communication efficiently. Manual processes can lead to errors, delays, and loss of critical information, negatively impacting patient care and operational efficiency. The Hospital Management System (HMS) aims to solve these issues by providing a centralized platform to manage patient records, schedule appointments, generate bills, and facilitate communication between staff, doctors, and patients. This system will ensure data accuracy, reduce manual workload, and improve the overall hospital experience.

Functional Requirments:

1- Patient Data Mangement
The system should allow to addition of new patient records
The system should enable to update and delete patient information as needed.
The system should store and retrieve the patient's medical history

2- Patient Access
The system should allow patients to know the scheduling and timetable for each doctor
The system should allow patients to check the history of their visits and upcoming appointments

3-Billing and payment

The system should support multiple ways to treat money
The system should allow staff to view and print billing information
The system should generate bills for services and treatments for patients

4-Appointment Management

Doctors will be able to manage their scheduling
The system allows patients to schedule or cancel
The system should provide the doctor or patient a notification about any change that happened

4-Reporting
The system should generate a report that has many details about patient data and their medical state

5-Communication System

The system should provide a communication system between patients and doctors if updates happen
Communication between staff and deparetenmnts

6-Doctor Schedule Management
The system should be able to provide a schedule for doctors that has all daily, and monthly schedules
Doctors should receive notifications about any cancellation
The system allows doctors to update their schedules.

Non Functional Requirments

-Response Time:
-The system must make user processor requests within 3 seconds.
-The system must support at least 50 concurrent users

Security:
-All data of patients must be secure and encrypted and must have a strong password

Availability:

-The system must have 99.9% uptime to ensure access.

Usability:
- The system must be easy to use for staff and patients and the interface should be simple.

Data Backup:
-The system must back up data and all information daily.

Scalability
-The system should handle an increased amount of data for patients.

Uses Cases:

Use Case 1:Add Patient Record

Use Case ID	UC-001

Use Case Name	**Add patient Record**
Actors	Staff(Primary),Doctor(Secondary)
Precondition	1-The staff or doctor must be logged into the system. 2-The patient must provide valid details.
Basic Flow:	1-The actor selects "Add Patient Record" from the system menu. 2-The actor enters patient details, including name, age, contact information, and medical history. 3-The actor confirms the details. The system saves the patient record and displays a success message.
Alternative Flows:	If any mandatory fields are missing, the system displays an error message and prompts the actor to complete them.
Postconditions	The patient record is stored in the system and available for future use

Use Case 2: Schedule Appointment

Use Case Name	**Schedule Appointment**
Use Case ID	UC-002
Actors:	Patient (Primary), Doctor (Secondary)
Preconditions:	The patient must have an account and be logged into the system. Appointment slots must be available
Basic Flow:	1-The patient selects "Schedule Appointment" from the dashboard. 2-The patient chooses a doctor and views available time slots.

	3-The patient selects a preferred slot and confirms. 4-The system notifies the doctor of the new appointment and updates the schedule.
Alternative Flows:	If the selected slot is no longer available, the system prompts the patient to choose another slot.
Postconditions:	The appointment is added to the system and visible to both the patient and the doctor.

Use Case 3: Generate Bill

Use case name	**generate bill**
Use Case ID	UC-003
Actor:	Staff (Primary), Patient (Secondary)
precondition:	1-The patient must have used hospital services. 2-Staff must be logged into the system
Main Flow:	1- actor select to generate bill 2- enter the name of the patient 3-actor review the data of the patient and check his services 4-system calculates the total bill amount 5-actor print the bill to the patient
Alternative Flows:	If the patient ID is invalid, the system displays an error and prompts for reentry.

Postcondition	The bill is generated and available for printing

UseCase 4 Generate Report

Use case name	**generate Report**
Use Case ID:	UC-004
Actor:	Doctor (Primary), Admin (Secondary)
Precondition:	the patient logged into the system and has already accounted for and dealt with this system
Main flow:	1-actor select generate Report 2-actor selects the patient and makes sure of data 3-actor can view, print, or email to the patient
Alternative flow:	If insufficient data exists, the system alerts the actor and cancels the operation.
Postconditions:	The report is generated and accessible to the actor

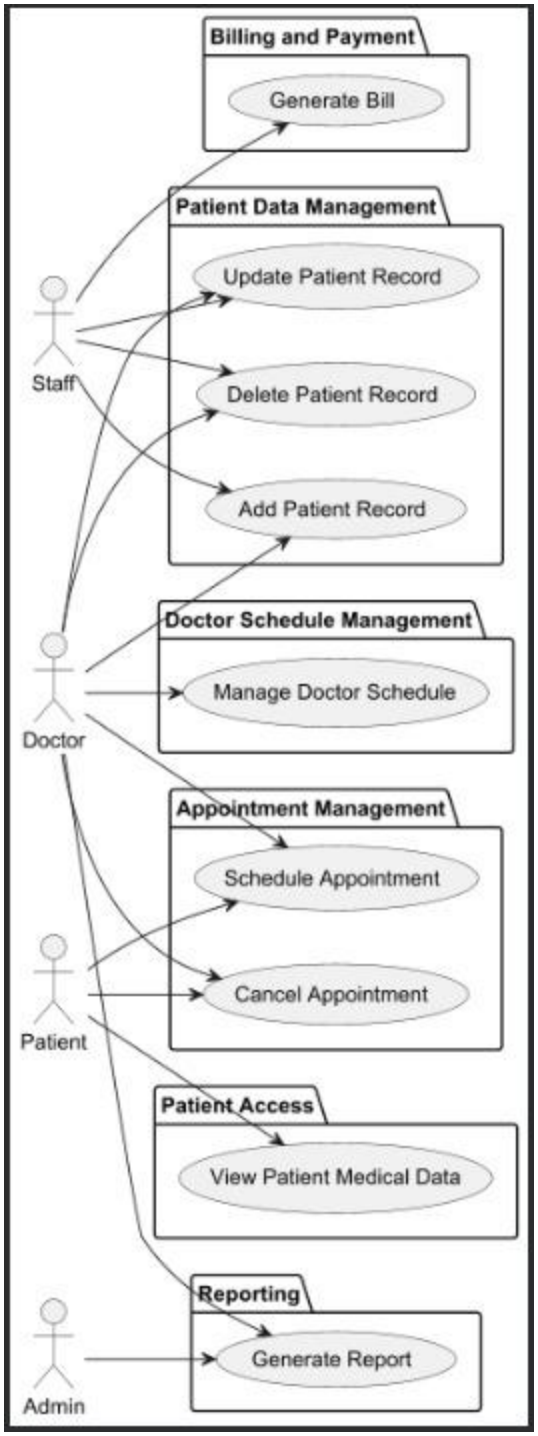
Use Case5: view patient Medical Data

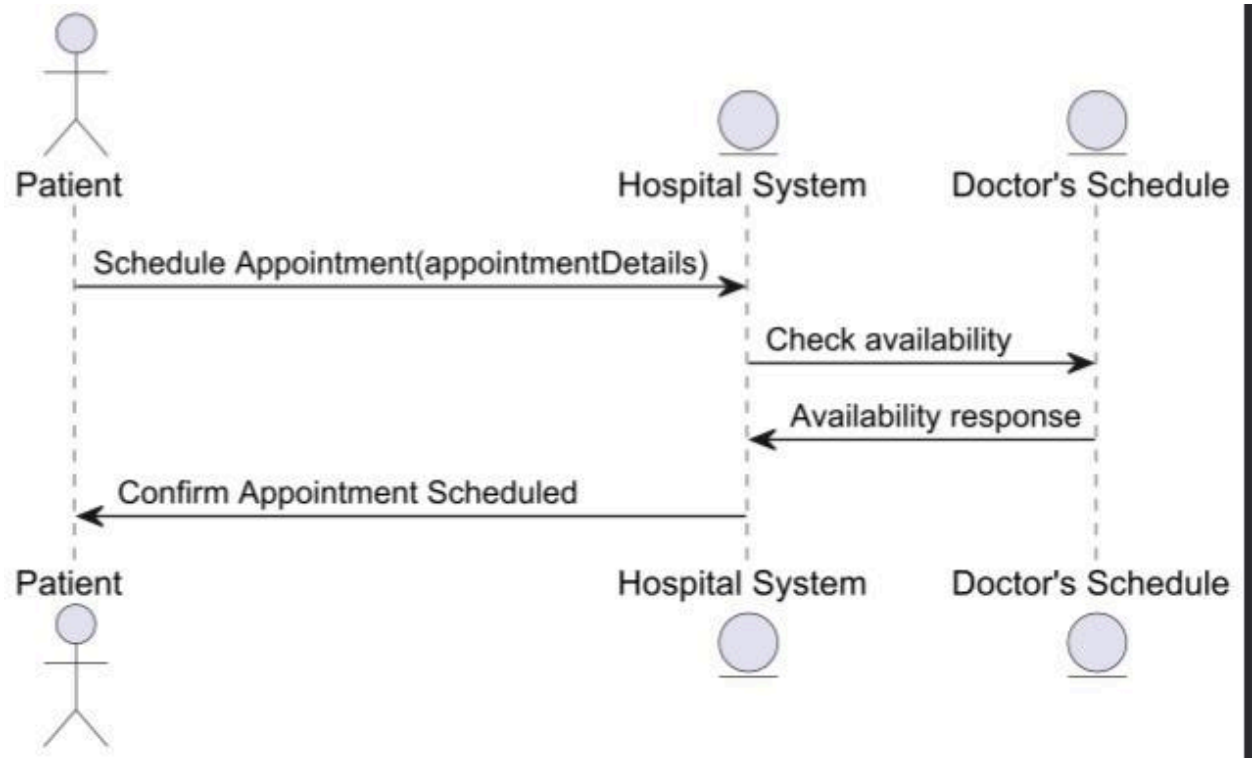
Use Case Name	**View patient Medical Data**
Use Case ID	UC-005
Actors	Patient(Primary)
Precondition:	The patient already has an account and logged into the system
Main flow:	1- The patient logged into the system

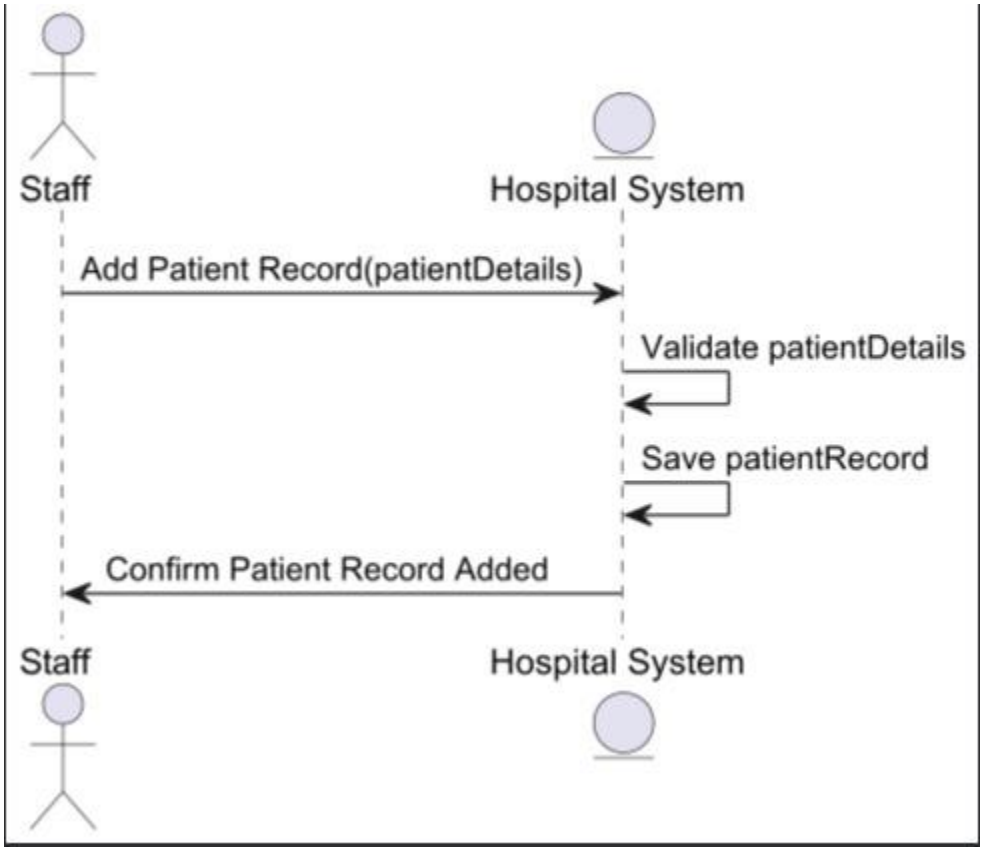
	2- The system verifies the patient's identity 3-system appears for the user all the treatments and next appointments 4-the patient can view or print data
Alternative flow:	if a name or email is already used system prompts for a different email.
Postconditions:	The patient successfully views their medical data.

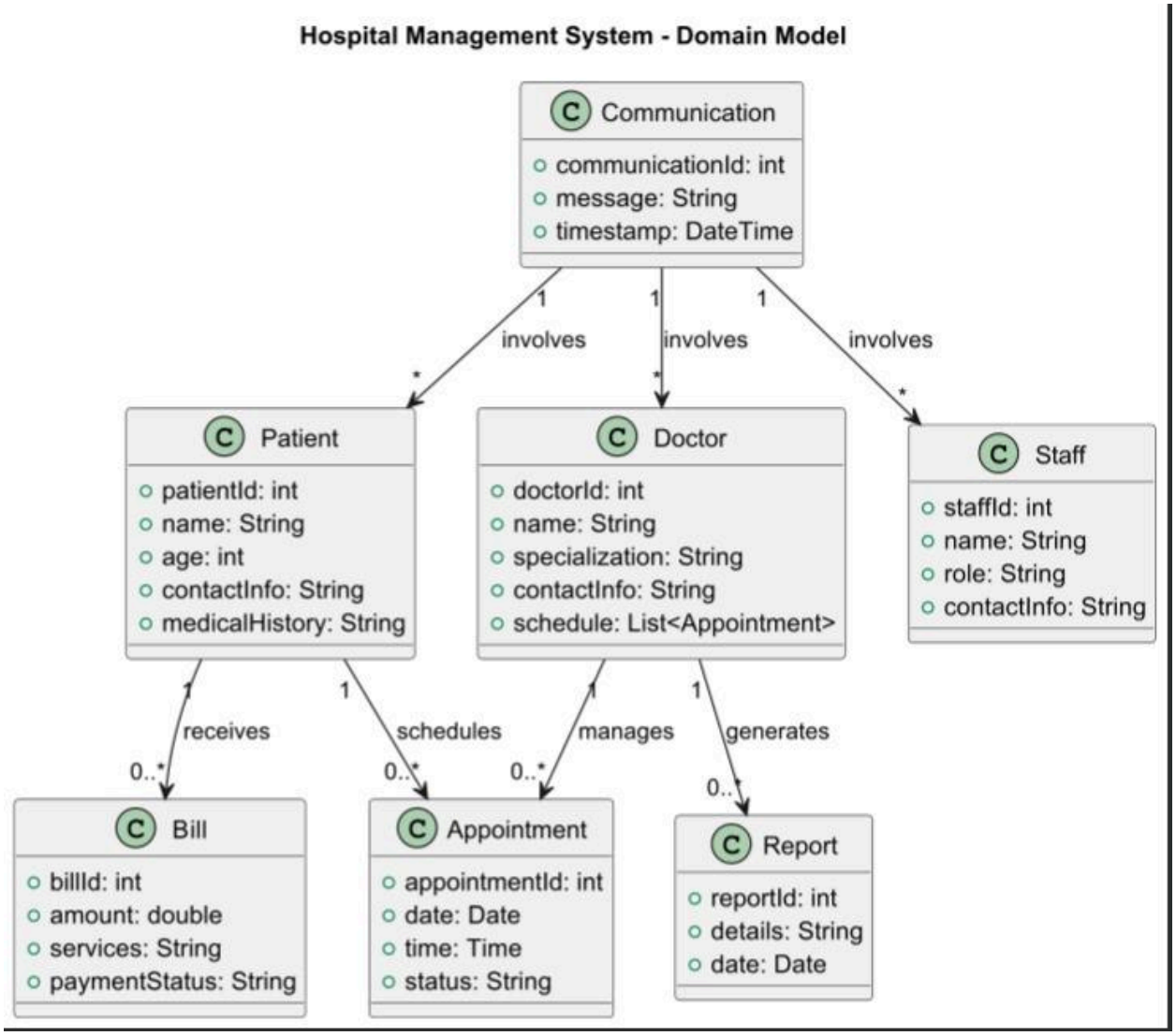
Use Case 6: Manage Doctor Schedule

Use Case Name	Manage Doctor Schedule
Use Case ID:	UC-006
Actors:	Doctor (Primary)
Preconditions:	The doctor must be logged into the system.
Basic Flow:	1-The doctor selects "Manage Schedule" from the menu. 2-The doctor views the current schedule. 3-Doctor updates or cancels appointments as needed. 4-The system notifies affected patients of the changes.
Alternative Flows:	If the doctor tries to cancel a slot that has a booked appointment, the system prompts for confirmation.
Postconditions:	The updated schedule is saved and reflected in the system


Billing and Payment
Generate Bill
Patient Data Management
Update Patient Record
Delete Patient Record
Add Patient Record
Staff
Doctor Schedule Management
Manage Doctor Schedule
Doctor
Appointment Management
Schedule Appointment
Cancel Appointment
Patient
Patient Access
View Patient Medical Data
Reporting
Generate Report
Admin


Patient
Hospital System
Doctor's Schedule
Schedule Appointment(appointmentDetails)
Check availability
Availability response
Confirm Appointment Scheduled
Patient
Hospital System
Doctor's Schedule


Staff
Hospital System
Add Patient Record(patientDetails)
Validate patientDetails
Save patientRecord
Confirm Patient Record Added
Staff
Hospital System


Hospital Management System - Domain Model
Communication
communicationId: int
message: String
timestamp: DateTime
1
involves
*
1
involves
*
1
involves
*
Patient
patientId: int
name: String
age: int
contactInfo: String
medicalHistory: String
Doctor
doctorId: int
name: String
specialization: String
contactInfo: String
schedule: List<Appointment>
Staff
staffId: int
name: String
role: String
contactInfo: String
1
receives
0..*
1
schedules
0..*
1
manages
0..*
1
generates
0..*
Bill
billId: int
amount: double
services: String
paymentStatus: String
Appointment
appointmentId: int
date: Date
time: Time
status: String
Report
reportId: int
details: String
date: Date