OVERVIEW

The Hospital Management System (HMS) automates administrative and operational processes for managing patient, doctor, and hospital resource data efficiently. This system supports the registration and management of Inpatient and Outpatient details, doctor scheduling, appointment handling, room allocation, and payment processing. Built with a modular architecture, it includes six key modules: Outpatient, Inpatient, Doctor, Appointment, Allocation, and Payment. Features include ID generation, data retrieval, record updates, and bill calculation, ensuring streamlined workflows and error reduction. HMS leverages Eclipse IDE, MySQL, and batch processing for scalability. It simplifies hospital operations, enhances record accuracy, and improves the overall patient experience.

Here's a brief description of the main menu for the Hospital Management System (HMS):

1. Doctor Section:

Access functionalities related to doctors, such as adding new doctors, updating their availability or fees, and retrieving doctor details. This section ensures efficient management of doctor-related data, including their specialization and schedules.

2. Patient Section:

Manage both Inpatients and Outpatients. Features include patient registration, appointment scheduling, room allocation, bill processing, and data retrieval. It supports updates to patient information like contact details and preferences.

3. Exit Application:

Terminates the application safely, ensuring all operations are saved and resources are released.

This menu provides an intuitive interface for streamlined navigation across key system functionalities.

DOCTOR SECTION USE CASES

The Doctor Module provides efficient tools to manage doctor information within the Hospital Management System. Below are the detailed functionalities:

1. Add Doctor

- The user specifies the number of doctors to add.
- Inputs include doctor ID, name, fee, specialization, available date, and time.
- The `addDoctor()` method:
  - *Validates inputs for correctness (e.g., fee > 0, date format).*
  - *Inserts valid records into the database.*

2. Update Doctor's Fee

- Allows updating consultation fees for a specific doctor.
- Input: Doctor ID and new fee.
- Validation:
  - *Ensure Doctor ID exists.*
  - *Verify new fee is valid.*
  - *Updates the fee in the database upon successful validation.*

3. Update Doctor's Availability Date

- Enables updating a doctor’s available date.
- Input: Doctor ID and the new date.
- Validation:
  - *Ensure Doctor ID exists.*
  - *Confirm the date is valid and not in the past.*
  - *Updates the availability date in the database if valid.*

4. Retrieve Doctor Details

- Fetches the details of a specific doctor based on Doctor ID.
- Process:
  - *Input Doctor ID.*
  - *Search the database for a match.*
  - *Returns details if the ID exists.*

5. Retrieve All Doctors' Details

- Fetches and displays all doctor records using `retrieveAllDoctors()` method.
- Useful for comprehensive data review or analysis.

6. Return to Previous Menu

- Navigates back to the main menu or the preceding screen, ensuring seamless user experience.

These functionalities ensure comprehensive and error-free doctor management in the HMS.

INPATIENT SECTION USE CASES

The Patient Section provides functionalities for managing Inpatient information and related services. Below are the detailed functionalities:

Inpatient Services

1. Add Inpatient Records

- Users specify the number of Inpatient records to add.
- Inputs: Patient ID, name, phone number, room type, admission date, etc.
- The `addInpatient()` method:
  - *Validates details (e.g., unique ID, valid phone number).*
  - *Inserts valid records into the database.*

2. Update Inpatient Contact Details

- Updates the contact details for a specific Inpatient.
- Inputs: Name and phone number.
- Validates if the record exists.
- Updates contact details in the database if valid.

3. Update Room Type for Inpatients

- Allows updating the room type (e.g., General, Private, ICU) for a specific patient.
- Inputs: Patient ID and new room type.
- Validates Patient ID.
- Updates the database if the input is valid.

4. Display Available Room Types

- Lists all available room types with corresponding charges.
- Useful for planning patient admission.

5. Update Meal Preferences for Inpatients

- Updates meal preferences (e.g., YES/NO) for a specific patient.
- Inputs: Patient ID and updated preference.
- Validates Patient ID.
- Updates the database if valid.

6. Retrieve Inpatient Details

- Retrieves specific patient details.
- Search by Patient ID or a combination of name and phone number.
- Returns patient details if found.

7. Bed Allocation Queries

1. Add Bed Allocation

- Assigns a bed to an Inpatient.
- Inputs: Patient ID, room number, admission date, and discharge date.
- Validates Patient ID exists in the Inpatient table.
- Updates the bed allocation in the database if valid.
- If the patient does not exist, prompt registration as an Inpatient first.

2. Retrieve Bed Allocation Details

- Fetches bed allocation details for a specific patient.
- Search by Patient ID or name and phone number.
- Returns room details, admission/discharge dates, etc., if found.

3. Retrieve All Bed Allocation Details

- Retrieves all bed allocation records.
- Uses the `retrieveAllallocation()` method to return a complete list.

4. Remove Allocation Details

- Deletes bed allocation records from the database.
- Input: Patient ID or name and phone number.
- Validates input and deletes the record if valid.

5. Return to Previous Menu

- Navigates back to the main or preceding menu.

8. Retrieve All Inpatient Records

- Fetches all Inpatient records.
- Uses the `retrieveAllPatient()` method for a complete listing.

9. Remove Inpatient Record

- Deletes a patient's record from the database.
- Inputs: Patient ID.
- Validates Patient ID and deletes the record if valid.
- Ensures dependent records (e.g., payments, history) are handled correctly.

10. Return to Previous Menu

- Navigates back to the main menu or the preceding screen.

These use cases ensure comprehensive management of Inpatient services, from registration and updates to room and bed allocation, enhancing the hospital's operational efficiency.

OUTPATIENT SERVICES USE CASES

The Outpatient Services module provides a structured approach to managing Outpatient records, appointments, and related activities in the database. Below are the detailed functionalities:

1. Add Outpatient Records

- Users specify the number of Outpatient records to add.
- Inputs: Patient ID, name, phone number, address, consultation details, etc.
- The `addOutpatient()` method:
  - *Validates details (e.g., unique ID, valid phone number).*
  - *Inserts valid records into the database.*

2. Update Outpatient Contact Details

- Updates the contact details for a specific Outpatient.
- Inputs: Name and phone number.
- Validates if the record exists in the database.
- Updates the contact details if valid.

3. Retrieve Outpatient Details

- Retrieves specific patient details.
- Search by Patient ID or a combination of name and phone number.
- Returns Outpatient details if a match is found.

4. Schedule Appointments for Outpatients

1. Add Appointment Records

- Adds a new appointment record to the database.
- Inputs: Patient name, phone number, doctor ID, specialization, appointment date, and time.
- Validates patient and doctor details (ensure IDs exist in the database).

- If valid, saves the appointment record and generates a unique Appointment ID.

2. Retrieve Doctor Details by Specialization

- Fetches a list of doctors based on their specialization (e.g., Cardiologist, Orthopedic).
- Displays doctor details, including name, availability, and fee.

3. Retrieve Appointment Record

- Retrieves details of a specific appointment.
- Inputs: Patient ID or name.
- Returns appointment details, such as assigned doctor, date, time, and status, if valid.

4. Update Appointment Date and Time

- Updates the date and time for an existing appointment.
- Inputs: Patient ID or name, new date, and time.
- Validates the inputs and ensures the new date and time are valid.
- Updates the database if all inputs are correct.

5. Retrieve All Appointment Records

- Fetches a list of all scheduled appointments.
- Displays details such as patient and doctor information, date, time, and appointment status.

6. Cancel Appointment

- Cancels an existing appointment.
- Inputs: Patient ID or name.
- Validates if the record exists in the database.
- If valid, marks the appointment as canceled or removes it from the database.

7. Return to Previous Menu

- Navigates back to the main menu or preceding screen.

5. Retrieve All Outpatient Records

- Fetches all Outpatient records stored in the database.

- Uses the `retrieveAllOutpatient()` method to return a complete list.

6. Remove Outpatient Records

- Deletes an Outpatient's record from the database.

- Inputs: Patient ID.

- Validates the ID to ensure it exists.

- Deletes the record permanently if valid.

- Ensures dependent records (e.g., payments or history) are handled appropriately.

7. Return to Previous Menu

- Navigates back to the main menu or preceding screen.

These functionalities provide a comprehensive solution for managing Outpatient services and appointments efficiently, supporting the overall operation of the healthcare facility.

BILLING AND PAYMENTS USE CASES

The Billing and Payments module manages payment transactions, ensuring proper tracking and retrieval of payment records in the database. Below are the detailed functionalities:

1. Add Payment Details

- Adds a new payment record to the database.

- Steps Involved:

1. Input payment details:
   - *Patient ID*
   - *Mode of Payment (e.g., cash, card, online)*
   - *Payment amount, date, etc.*
2. Validate the Patient ID:
   - Check if the Patient ID exists in the database.
3. Save the validated payment details into the database.

2. Retrieve Payment Details

- Retrieves the payment record for a specific patient.

- Steps Involved:

1. Input the Patient ID to search for the payment record.
2. Fetch payment details:
   - *Patient Name*
   - *Payment Date*
   - *Mode of Payment*
   - *Bill Amount*
3. If no matching record is found, display:
   - "No record found in database."

3. Retrieve All Payment Details

- Fetches all payment records stored in the database.

- Features:

  - Displays a list of all payment transactions:

    - *Patient details*

    - *Payment dates*

    - *Modes of payment*

    - *Bill amounts*

  - Useful for generating financial reports or reviewing bulk payment data.

4. Return to Previous Menu

- Navigates back to the main menu or preceding screen.

This Billing and Payments module ensures streamlined payment management, supports auditing and reporting, and maintains accurate financial records for the healthcare facility.

****