Hospital Management System in C++

Introduction

This Hospital Management System is a C++ project that simulates real-world hospital queue management across multiple departments. It manages patients' details, prioritizes critical cases, and provides a structured menu-driven interface using Object-Oriented Programming (OOP), STL containers like `vector`, `list`, and `unordered_set`, along with strong data validation and search functionalities.

Objectives

Multi-Department Support

Manages patient queues for four core hospital departments:

- General Clinic
- Heart Clinic
- Lung Clinic
- Plastic Surgery

Patient Queue Management

- **Patient Categorization**:
    - **Critical Patients** are given priority by being added to the front of the queue.
    - **Normal Patients** are added to the end, following FIFO (First-In-First-Out).
- Each department maintains its own queue of patients.

Patient Operations

- **Add Patients**: Register a new patient with full details including a unique ID, name, age, gender, and blood group.
- **Search Patients**: Locate a patient by their unique ID and display their complete details.
- **Serve Patients**: Call the next patient (critical or first in line) and remove them from the queue.
- **Display Queue**: List all current patients in a department with detailed information.
- **Remove Patients**: Delete a specific patient using their unique ID.

Technical Overview

Language

C++ (Modern Standard, using STL)

Data Structures

- `std::list`: Used for dynamic patient queues in each department.
- `std::unordered_set`: Ensures uniqueness of patient IDs for fast lookup and validation.
- `std::vector`: Maintains multiple department queues.

Core Concepts Used

- Classes and Objects
- STL Containers (`vector`, `list`, `unordered_set`)
- Pointers and Dynamic Memory
- Input Validation and Menu Loops
- Structured Programming for Interface Desig
- Basic string handling and character I/O

Key Classes and Structures

`class Patient`

Represents an individual patient with the following attributes:

- ID (unique, e.g., mobile number)
- First and Last Name
- Age
- Gender
- Blood Group
- `display()` method for formatted output

`class HospitalQueue`

Manages a department's queue using a linked list:

- Add Patient (normal or critical)
- Remove Patient by ID
- Search Patient by ID
- Call Next Patient
- Display all patients in the department
- Ensures uniqueness using `unordered_set`

Main Functionalities

- **Menu-Driven Interface**:
    - Department-level menus with 8 operations including "Change Department" and "Exit the System".
- **Search by ID**: Easily locate patient details.
- **User-Friendly Feedback**: Clear success and error messages.
- **Input Validations**:
    - Duplicate ID checks
    - Valid blood group format check (case-insensitive)

Future Enhancements

- **Doctor Record Integration**: Assign doctors based on department/specialty.
- **Appointment Scheduling System**: Time slots for patients and appointment tracking.
- **Data Persistence**: Store and load patient records using file I/O.
- **GUI Interface**: Implement with frameworks like Qt or SFML for better interaction.

Conclusion

This project is a comprehensive implementation of hospital queue management using C++. It demonstrates real-world use of:

- Linked list-based queues
- Class design and encapsulation
- Menu control logic
- Input validation and patient prioritization

It is an ideal project for demonstrating C++ and OOP proficiency and serves well in interviews or academic submissions. The modularity, structure, and expandability make it a great foundation for more advanced healthcare management systems.
