Data Structure

Term Project Report

A blue and white logo

Description automatically generated

**Amna Miraj**

**Student ID: F20232661130**

**Section: V1**

**Resource Person: Sir Tahir Sohail**

School of Systems and Technology

UMT Lahore Pakistan

Hotel Management System: Project Documentation

# Introduction

This document describes a Hotel Management System (HMS) developed in C++ to simulate the operations of a hotel, including room bookings, booking requests handling, checkout process, and viewing booking history. The system implements essential features like priority booking, room management, and the ability to process and store completed bookings using data structures such as stacks, queues, and binary search trees (BST).

# System Overview

The Hotel Management System (HMS) consists of multiple components and classes that collectively simulate hotel operations. The major classes in the system include:

- Start: Handles the startup screen and displays a loading animation.  
- Room: Represents a hotel room, including its ID, type, price, and status (whether it’s "Ready" or "Booked").  
- Booking: Stores customer booking information such as the customer’s name, room number, room type, number of nights, and total price.  
- BookingRequest: Handles the booking request made by customers, including priority and preferred floor.  
- Queue: Manages booking requests using a queue structure.  
- Stack: Keeps a history of completed bookings using a stack data structure.  
- Floor: Represents a floor in the hotel, managing rooms on that floor.  
- Hotel: The core class that manages overall operations of the hotel, including processing requests, room allocation, and booking history.

# Class Descriptions

## Start Class

The start class is responsible for the system initialization process. It handles displaying the welcome screen and a loading animation to simulate the hotel system startup.

- Methods:  
 - equals\_to(): Prints borders to create a visual frame around the startup screen.  
 - start(): Displays the loading animation and then clears the screen before showing the welcome message.

## Room Class

The Room class represents a hotel room and contains information about the room’s ID, type (Single, Double, Suite), status (Ready/Booked), and price.

- Methods:  
 - getId(), getType(), getStatus(), getPrice(): Getter functions to access room details.  
 - setStatus(): Updates the status of the room.  
 - display(): Displays room information including ID, type, price, and status.

## Booking Class

The Booking class stores the information for a specific booking made by a customer. It includes customer details, the room number, the number of nights, and the total price.

- Methods:  
 - display(): Displays the booking details including customer name, room number, type, nights, and total price.

## BookingRequest Class

This class represents a customer’s booking request, which includes their name, preferred room type, number of nights, priority (High or Regular), and preferred floor.

- Methods:  
 - display(): Displays the booking request details.

## Queue Class

The Queue class manages the requests for booking rooms. It contains two queues: one for high-priority requests and one for regular requests.

- Methods:  
 - enqueue(): Adds a new booking request to the queue.  
 - dequeue(): Removes the next booking request from the queue.  
 - display(): Displays all requests in the queue.

## Stack Class

The Stack class is used to maintain the booking history. It stores completed bookings and allows users to cancel the last booking.

- Methods:  
 - push(): Adds a booking to the history.  
 - pop(): Removes the most recent booking from the history.  
 - display(): Displays all completed bookings in the history.

## Floor Class

The Floor class manages a floor in the hotel, keeping track of the rooms on that floor and their availability.

- Methods:  
 - displayAvailableRooms(): Displays all available rooms on the floor.  
 - findRoomByNumber(): Searches for a room by its number.  
 - findAvailableRoomByType(): Searches for an available room based on the requested type (Single, Double, Suite).

## Hotel Class

The Hotel class is the main class managing the hotel operations. It handles processing booking requests, room assignments, checkouts, and displays the booking history.

- Methods:  
 - processBookingRequests(): Processes all booking requests in the queue and assigns available rooms.  
 - checkoutRoom(): Handles the checkout process and generates an invoice.  
 - cancelLastBooking(): Cancels the last booking in the history.  
 - displayBookingHistory(): Displays the entire booking history.

# Program Flow

The system operates by displaying a menu that allows the user to interact with the hotel system. The flow of operations is as follows:

1. The program initializes and displays a loading screen followed by the welcome message.  
2. The user is presented with a menu with options like viewing available rooms, processing bookings, and checking out.  
3. The user can view rooms, add a booking request, process requests, and view or cancel bookings.  
4. The system assigns rooms based on availability and priority, and users can check out and receive an invoice.  
5. The booking history is stored in a stack, and users can cancel the last booking made.

# Key Features

The following key features are implemented in the Hotel Management System:

- Room Management: Users can view available rooms and book rooms based on type and availability.  
- Priority Booking: High-priority booking requests are handled before regular requests.  
- Booking History: The system tracks all completed bookings and provides a history of past bookings.  
- Checkout Process: Customers can check out, and the system generates an invoice based on the number of nights stayed.  
- Queue and Stack: The system processes booking requests using a queue, and completed bookings are stored in a stack.

# Suggestions for Improvements

While the system is functional, the following enhancements could be considered for future versions:

- Memory Management: Properly clean up dynamically allocated memory by implementing destructors in relevant classes.  
- Cross-Platform Compatibility: Replace Windows-specific commands like Sleep and system("CLS") with cross-platform alternatives such as std::this\_thread::sleep\_for.  
- Persistence Layer: Introduce file storage or a database system to persist room bookings, requests, and history between program runs.  
- Graphical User Interface (GUI): Implement a GUI for better interaction and user experience.

# Conclusion

The Hotel Management System successfully simulates key operations in hotel management, including room bookings, processing requests, and managing booking history. The system is built using object-oriented principles and implements fundamental data structures like queues, stacks, and binary search trees. Future work can focus on adding more advanced features such as payment handling, a graphical user interface, and persistent storage for bookings and history.