**A&R's Hostel Management System**

**Purpose:**

The program is designed to streamline room bookings and management in a girl's hostel. It is an easy-to-use interface for managing room bookings and related services in a hostel.

**Key Features:**

1. **Room Availability Check**  
   Displays all unbooked rooms with their respective prices.
2. **Room Booking**
   * Enables users to book a room by entering their name, contact details, and payment status.
   * Offers additional services like lunch and cleaning.
3. **Display Booking Details**  
   Lists all booked rooms in a formatted table with occupants details and their selected services.
4. **Cancel Booking**
   * Allows users to cancel a booking by entering the room number.
   * Clears user details and marks the room as available.
5. **Summary Display**  
   Provides a quick overview of:
   * Total rooms.
   * Counts available rooms at the moment and booked rooms.
6. **Unpaid Bookings Display**  
   Identifies and lists bookings with payment status as "Not Paid."
7. **Exit Confirmation**  
   At the end, it asks for confirmation before closing the program to prevent accidental exits.

**How the Program Works:**

**Main Menu**

The program begins by displaying a menu with the following options:

1. Check available rooms.
2. Book a room.
3. Show all booking details.
4. Cancel a booking.
5. Display summary.
6. Display unpaid bookings.
7. Exit.
8. **Room Booking Process**
   * Shows available rooms.
   * Validates the entered room number:
     + Rejects invalid or already booked rooms.
   * Collects user details (name and contact).
   * Handles payment status (Paid/Not Paid).
   * Allows users to select additional services:
     + **Lunch**: Membership or pay-per-meal.
     + **Cleaning Services**: Room cleaning, laundry, or both.
9. **Display All Booking Details**  
   Lists all booked rooms in a formatted table, showing:
   * Room details (type and price).
   * User details (name, contact).
   * Payment status.
   * Selected services.
10. **Cancellation Booking Process**
    * User enters the room number for cancellation.
    * Clears user details and marks the room as available.
11. **Display Summary**
    * Shows a summary of total, available, and booked rooms.
12. **Display Unpaid Bookings**
    * Lists all rooms with unpaid bookings.
    * Displays user names for follow-up.
13. **Exit Confirmation**  
    Asks the user to confirm before exiting the program.

**Core Concepts Applied:**

**Header Files**

* <iostream>: For input and output operations.
* <string>: For handling user input and room details.
* <iomanip>: For formatted table output.

**Functions**

|  |  |
| --- | --- |
| Function Name | Purpose |
| showAvailableRooms() | Displays all available rooms. |
| bookRoom() | Handles room booking and user details. |
| makePayment(int roomIndex) | Tracks payment status as “Paid” or “Not Paid”. |
| selectServices(int roomIndex) | Allows users to choose lunch and cleaning services. |
| displayDetails() | Displays all booked room details in a table. |
| cancelBooking() | Cancels a booking and clears user details. |
| displaySummary() | Provides a summary of booked and available rooms. |
| displayUnpaidBookings() | Lists unpaid bookings for follow-up. |
| confirmExit() | Prompts the user for confirmation before exiting the program |

**Structures (struct)**   
The program uses a **Room** structure to store data for each room, including:

* + Room description.
  + Booking status.
  + User information (name, contact).
  + Payment and service details.

**Global Arrays**  
A global array of **Room** structures is used to represent all rooms in the hostel.

**Conditional Statements**  
*if-else* and *switch* statements are used extensively to handle user choices and validation.

**Loops**

* + **for loops**: Iterates over the room array for displaying data.
  + **Infinite while loop**: Ensures the main menu runs continuously until the user exits.

**Input Handling**  
Functions like cin, cin.ignore(), and getline() are used for user input, including handling names with spaces.

**Known Bugs:**

1. **Input Validation for Menu Choices**:
   * If a user enters a non-integer (e.g., a character or string) for the menu choice, the program will enter an infinite loop or unexpected behavior.
   * **Bug Example:** Inputting a letter (e.g., a) at the main menu will cause a failure due to improper handling of invalid input.
2. **Unhandled Edge Cases for Room Services**:
   * If users enter invalid options (e.g., 5) for **lunch services** or **cleaning services**, no proper validation or error message is given, and the default service is set to "None".
3. **No Validation for Contact Number**:
   * Any string (e.g., "abc123") can be entered as a contact number. This could lead to unreliable data storage.
4. **Display Details Output Formatting Issue**:
   * If the user inputs a very long string for name or contact, it can break the table's alignment in *displayDetails().*

**Restrictions:**

1. **Fixed Number of Rooms**:
   * The total number of rooms is hardcoded (totalRooms = 5). Adding more rooms would require manually updating both the *rooms* array and the hardcoded descriptions.
2. **Lack of Multi-User Capability**:
   * The program assumes only one user is interacting with the system at a time. There's no mechanism to log or track multiple administrators.
3. **Limited Payment Options**:
   * Payment is only marked as "Paid" or "Not Paid". There's no feature to update the payment status later if a payment is made after booking.
4. **No Persistent Data Storage**:
   * Data about bookings, cancellations, and room status are stored in memory. Once the program exits, all data is lost.
5. **No Search Functionality**:
   * You can't search for a specific booking by name, contact, or room number, which might be helpful for managing bookings in larger hostels.
6. **Static Room Details**:
   * Room descriptions are predefined. There's no way to dynamically add or modify room details (e.g., price or type).

**Future Improvements:**

1. **Refactor Room Index Calculation**:
   * Use a mapping function or a more robust system (like storing room numbers in a separate array) to avoid relying on assumptions about room numbers.
2. **Enhance Error Messages**:
   * Provide clear feedback for invalid inputs (e.g., "Invalid menu option" or "Invalid service selection").
3. **Introduce Data Persistence**:
   * Save room data (bookings, payments, services) to a file (e.g., using file I/O) and reload it when the program restarts.
4. **Dynamic Room Management**:
   * Allow the admin to add, modify, or delete room details dynamically instead of hardcoding them.
5. **Improve Summary**:
   * Include additional statistics in the summary, such as the **total revenue** from bookings or services.
6. **Add Search Functionality**:
   * Allow searching for bookings by name, contact, or room number to make the system more user-friendly.
7. **Improve User Interface**
8. **Support Larger Room Numbers**:
   * Update room handling logic to accommodate rooms with non-standard numbering (e.g., 301, 401).
9. **Multi-User System** (Introduce roles for users, such as "Admin" and "Guest.")
10. **Automatic Room Numbering** (using unique IDs)
11. **Additional Services** (Adding more services like **internet and parking** etc.)

## Explanation of Steps in the Flowchart

## 1. **Start**

* The program begins execution and displays the **Main Menu** with various options for the user to choose from.

## 2. **Main Menu Options**

The user can select one of the following options:

* **Check Available Rooms:** Displays all unbooked rooms.
* **Book a Room:** Guides the user through the room booking process.
* **Show All Details:** Displays booking information for all rooms.
* **Cancel a Booking:** Allows the user to cancel an existing booking.
* **Search Booking by Name:** Searches and displays bookings based on the user’s name.
* **Display Summary:** Shows the count of available and booked rooms.
* **Edit Booking Details:** Allows the user to modify booking details or additional services.
* **Display Unpaid Bookings:** Lists bookings where payment status is "Not Paid".
* **Exit:** Ends the program.

## 3. **Check Available Rooms**

* Displays the list of rooms that are not yet booked.
* Returns to the **Main Menu** after showing the availability.

## 4. **Book a Room**

* Calls **Check Available Rooms** to show unbooked rooms.
* Prompts the user to **Enter Room Number** for booking.
* **Check Availability:**
  + If the room is available:
    - User enters personal details.
    - Proceeds to **Make Payment** to update payment status.
    - Moves to **Select Services** where the user chooses lunch and cleaning options.
    - Confirms the booking and returns to the **Main Menu**.
  + If the room is already booked:
    - Displays a message and redirects to the **Main Menu**.

## 5. **Show All Details**

* Displays a formatted table of all bookings, showing user information, payment status, and selected services.
* Returns to the **Main Menu**.

## 6. **Cancel a Booking**

* Prompts the user to **Enter Room Number**.
* **Check Booking Status:**
  + If valid:
    - Cancels the booking by clearing all user details and marking the room as available.
    - Returns to the **Main Menu**.
  + If invalid:
    - Displays an error and returns to the **Main Menu**.

## 7. **Search Booking by Name**

* Prompts the user to **Enter Name**.
* Searches through all bookings:
  + If found, displays booking details for that name.
  + If not, informs the user that no booking exists.
* Returns to the **Main Menu**.

## 8. **Display Summary**

* Calculates and displays the number of available and booked rooms.
* Returns to the **Main Menu**.

## 9. **Edit Booking Details**

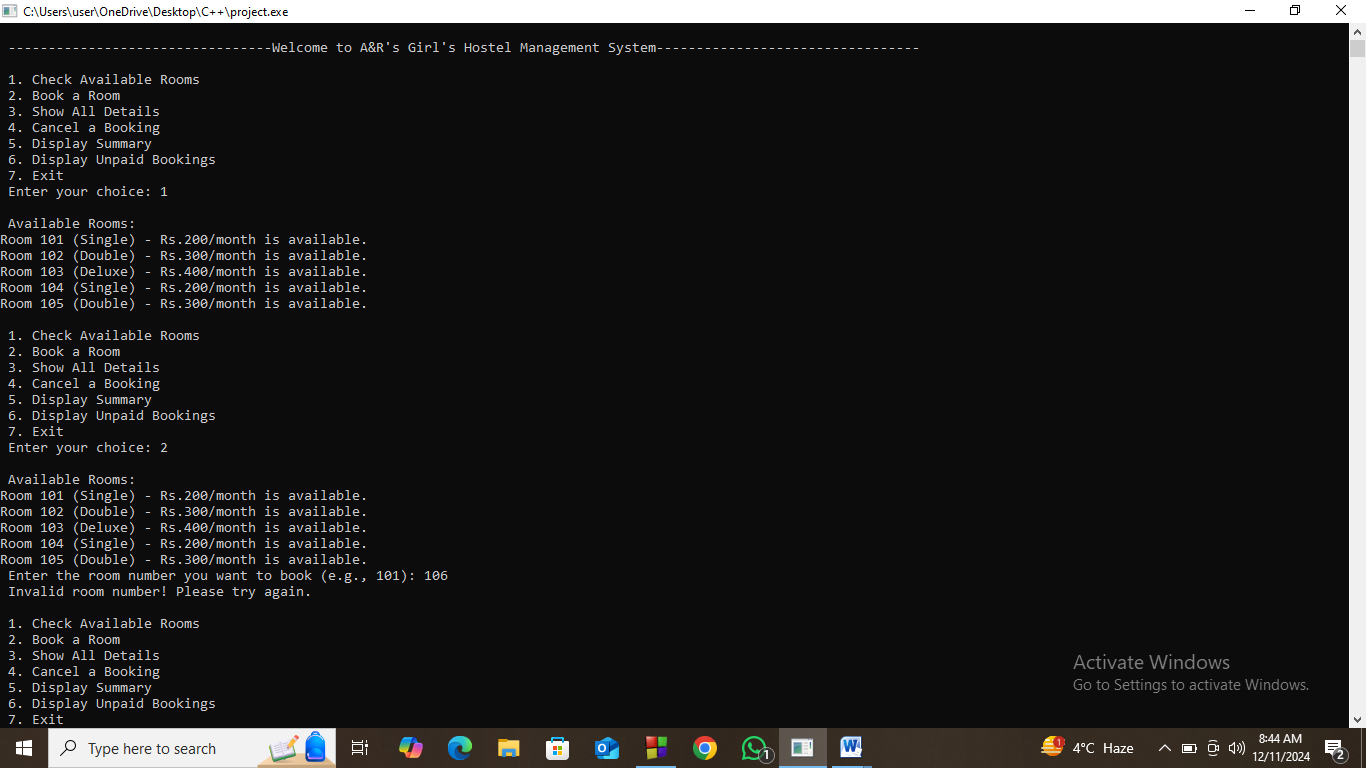
* Prompts the user to **Enter Room Number**.
* **Check Booking Status:**
  + If valid:
    - Allows editing the contact information.
    - Optionally updates selected services.
    - Confirms the update and returns to the **Main Menu**.
  + If invalid:
    - Displays an error and returns to the **Main Menu**.

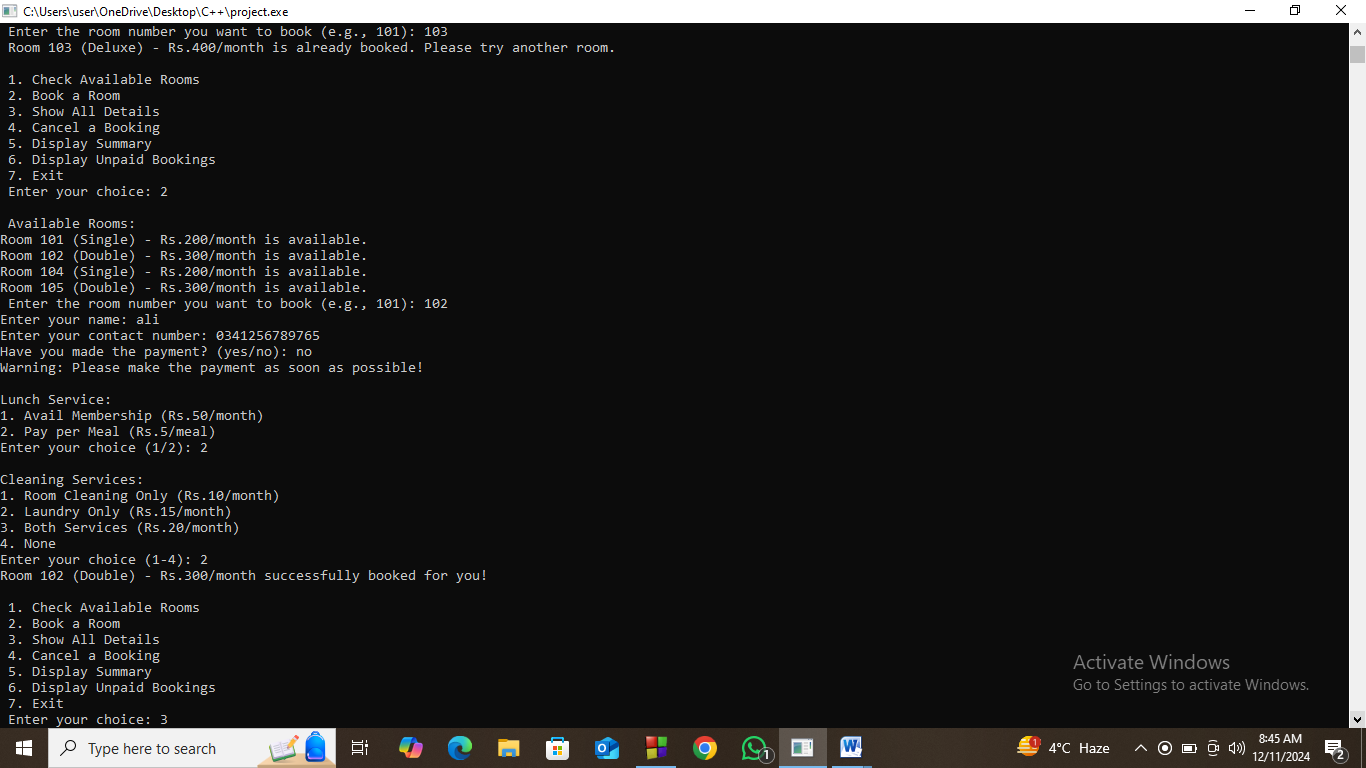
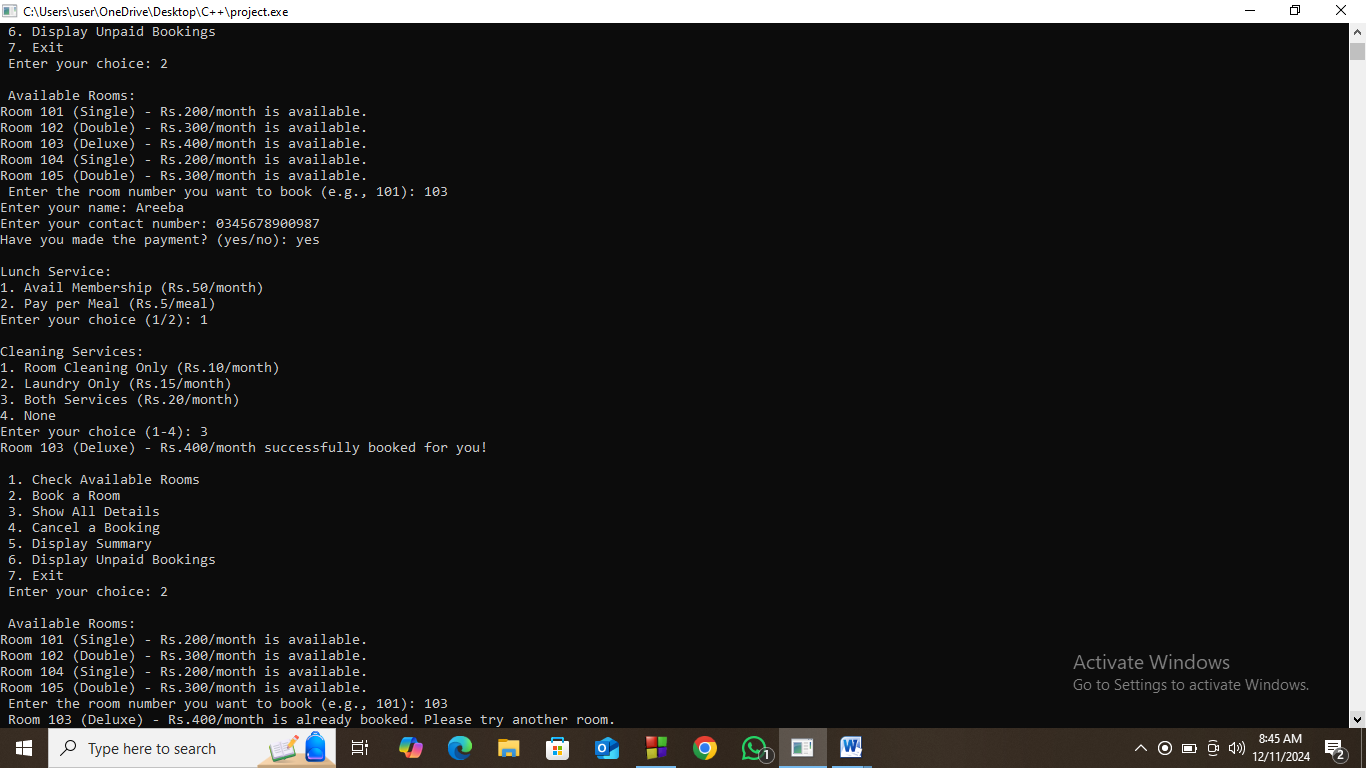
## 10. **Display Unpaid Bookings**

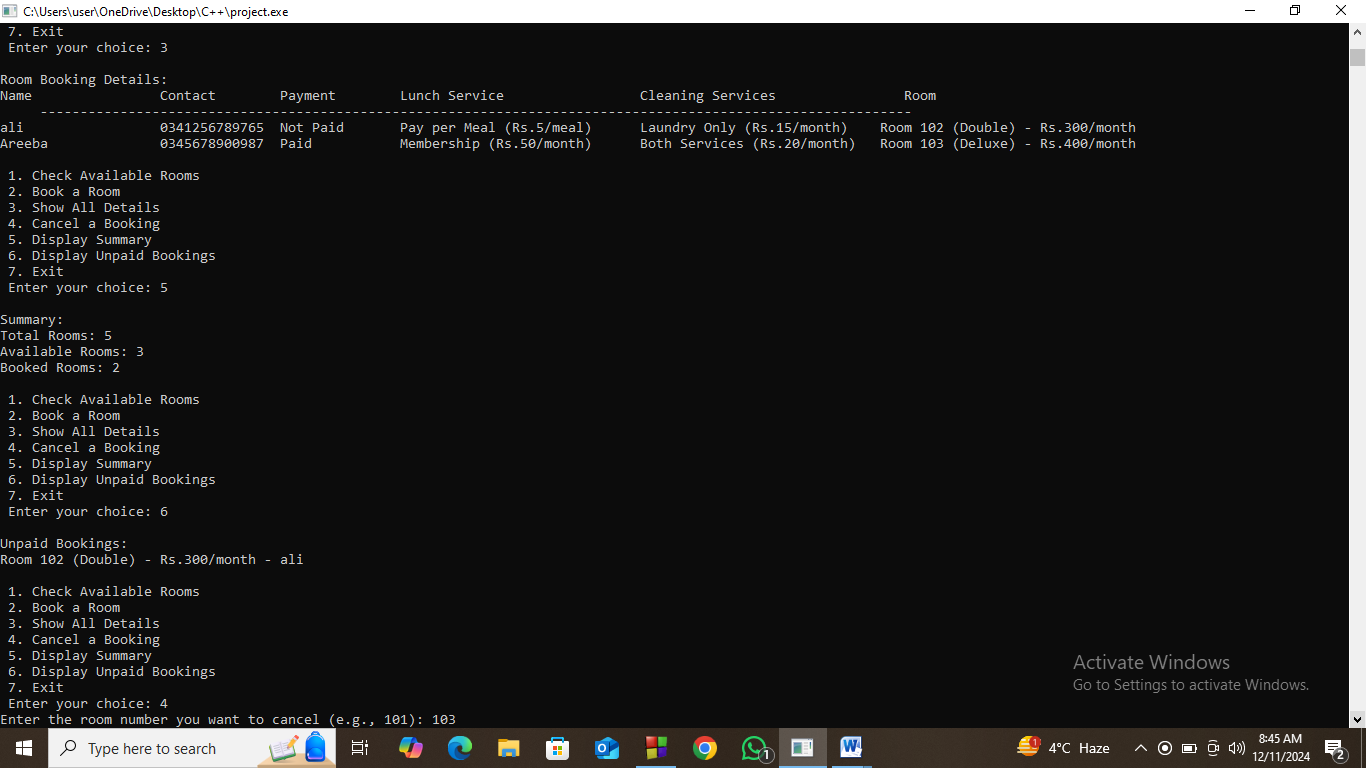
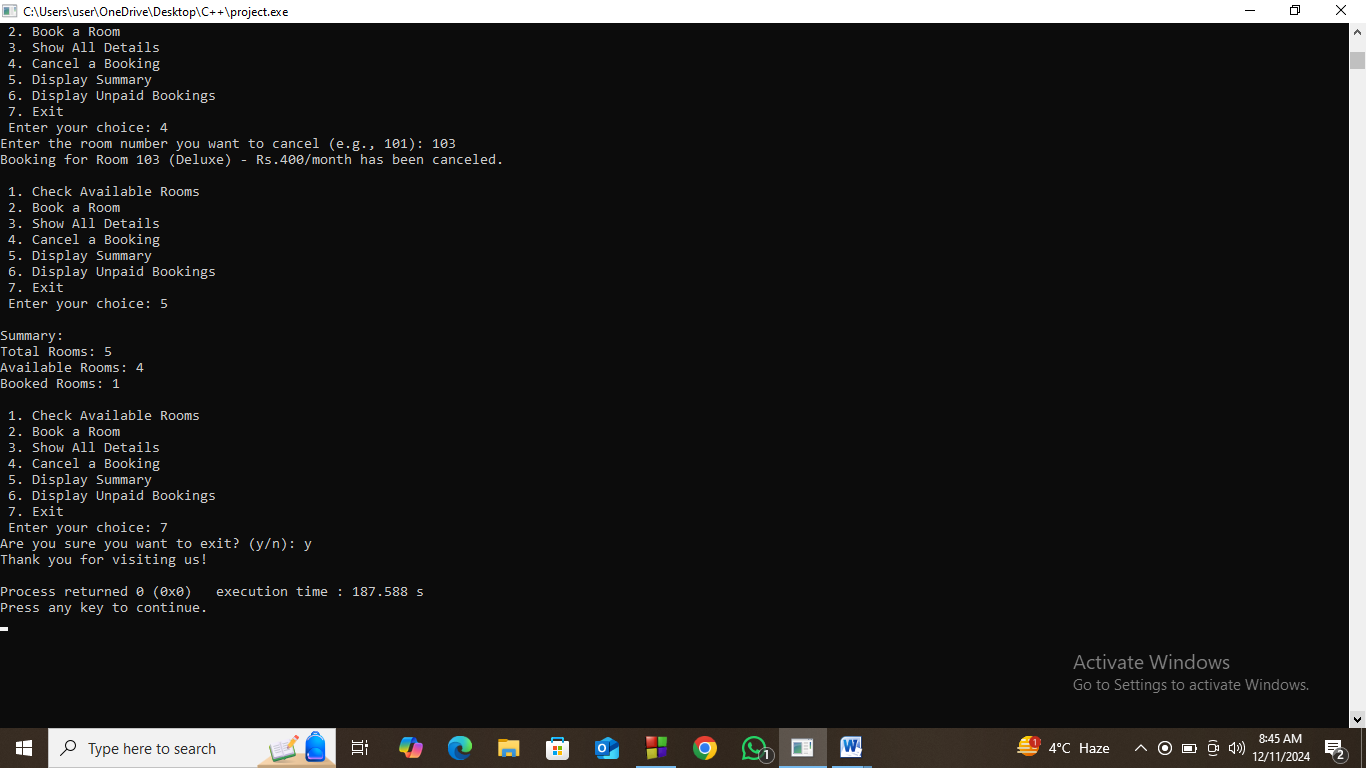
* Searches through all bookings to find those marked as "Not Paid."
* Displays the list of unpaid bookings if found.
* If none, informs the user.
* Returns to the **Main Menu**.

## 11. **Exit**

* The program ends execution after displaying a thank-you message.

**User Interface:**

****

****