 NATIONAL UNIVERSITY OF COMPUTER AND EMERGING SCIENCES

(KARACHI CAMPUS)

FAST School of Computing Fall 2024

**Programming Fundamentals Lab Project Report**

**Room Reservation System**

**-Syed Muhammad Sarim (24k-0718)**

**-Haziq Hussain (24k-0944)**

**-Sheheryar Hasan Nasir (24k-2017)**

**Introduction**

This project aims to build an efficient classroom booking management system to streamline the booking process of classrooms and avoid booking conflicts by storing the details of booked rooms.

**Background**

Booking classrooms manually can be an inefficient and time-consuming activity, especially when managing the bookings, which can lead to potential conflicts as well.   
The designed and developed system leverages file handling and modular programming techniques to automate the entire process accurately and efficiently.

**Project Specification**

The project is divided into three modules, each handled by a specific team member:

1. **Room Management & File Handling**: Handles room initialization, availability management, and file I/O operations.
2. **Booking Management**: Manages room bookings, resolves conflicts, and ensures smooth file handling for booking data.
3. **Administrator & Reports**: Provides tools for administrators to view and edit bookings and generate detailed reports.

**Problem Analysis**

The challenges identified include:

1. Understanding how to create and include header files in C was initially a challenge. Research was conducted online to learn the proper method for defining function prototypes, global variables, and including the header file in the .c files for integration.
2. Using a .txt file to store classroom data and reading it into the program was complex. Implementing file handling to properly read data into the room structures required careful planning and testing.
3. Efficiently tracking and displaying room availability based on booking data.
4. Lack of checks for user inputs (e.g., invalid dates or room numbers).

**Solution Design**

1. **Room Management & File Handling**initializeRooms(): Initializes default room data if none exists in the storage file.  
   saveRoomsToFile(): Saves room data to rooms.dat.  
   loadRoomsFromFile(): Reads room data from a .txt file and loads it into memory. This was implemented by opening the file in read mode and parsing the content into the program's data structures.  
   displayRooms(): Displays a list of available rooms.
2. **Booking Management**makeBooking(): Processes booking requests, checks room availability, and updates room statuses.  
   saveBookingsToFile(): Persists booking data in bookings.dat.  
   loadBookingsFromFile(): Loads booking records into memory.
3. **Administrator and Reports**adminMenu(): A menu-driven interface for administrative tasks like viewing, editing, canceling bookings, and generating reports.  
   viewBookings(): Lists all bookings, showing user details, room numbers, and time slots.  
   generateReport(): Categorizes bookings into Faculty and Student types, calculates room usage, and optionally saves the report to a file.  
   editOrCancelBooking(): Allows administrators to search bookings by user name and date to edit or cancel them.

**Implementation and Testing**

The implementation revolves around creating a reservation management system for admins, faculty, and students. Key features include:

* Secure login for admin, faculty, and students with predefined credentials.
* Admin can view available rooms, their details, and booking statuses.
* Faculty and students can book rooms based on availability. The system enforces priority rules, with faculty overriding student bookings when conflicts arise.
* The system checks for booking conflicts based on time slots and user priority.
* Admin can view and cancel any booking.
* Data for rooms and bookings is saved and loaded from text files (room.txt, bookings.txt).
* Ensures proper date format and checks room capacity during bookings.

The program uses modular functions to ensure code reusability and maintainability. For output formatting, colored text is used with the Windows console to enhance readability.

Testing involved the following steps:

* Tested all login types (Admin, Faculty, Student) with correct and incorrect credentials.
* Verified that only authorized roles can access specific features (e.g., booking, viewing bookings).
* Created a room.txt file with room data and validated successful loading during program startup.
* Verified the accuracy of room details displayed on the console.
* Tested bookings for different users with valid and conflicting time slots.
* Verified that the system prioritizes faculty bookings over student bookings in case of conflicts.
* Simulated bookings for the same room, date, and time slot to ensure conflicts are detected and resolved appropriately.
* Admin cancelled bookings successfully, and the system updated both memory and the bookings.txt file.
* Checked the correctness of data saving and loading in room.txt and bookings.txt.

**Project Workflow Distribution**

1. Room Management Functions and Design **– Syed Muhammad Sarim**(**24K-0718**) (**Week 1-2**)
2. Booking Management **– Haziq Hussain** (**24K-0944**) (**Week 3-4**)
3. Administrative Tools and Reports – **Sheheryar Hasan Nasir** (**24K-2017**) (**Week 5-6**)
4. Integration – **All members** (**Week 7**)

**Results**

1. Automated initialization and availability management for rooms.
2. Conflict-free booking management with accurate updates.
3. Administrative tools for modifying and cancelling bookings.

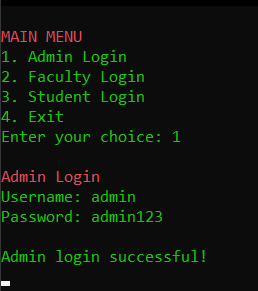
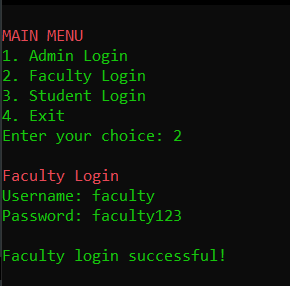
**Conclusion**

The room booking management system effectively facilitates the process of room reservations for administrators, faculty members, and students. The system ensures proper authentication and prioritization of booking conflicts, such as faculty overriding student reservations. Features like displaying available rooms, viewing all bookings (admin-only), making reservations, and canceling bookings provide a comprehensive solution for managing room availability.

**Outputs:**

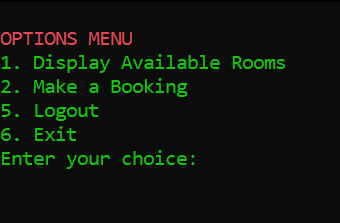
**Admin log in: Faculty Log in: Admin Log in:**

A screen shot of a computer

Description automatically generated

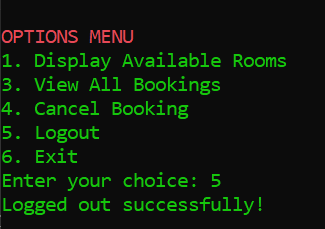
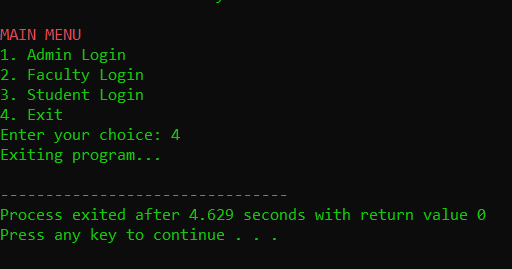
**Admin Menu Faculty Menu: Student Menu:**

A menu screen with green text

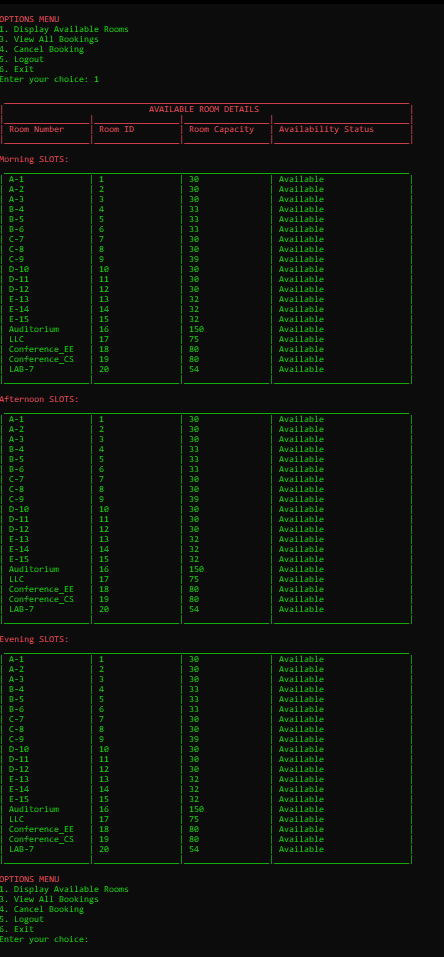
Description automatically generatedA screen shot of a menu

Description automatically generated

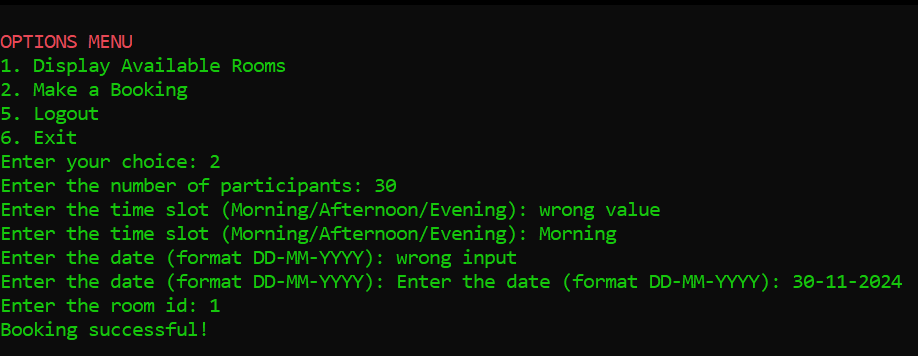
**Log out: Exit:**



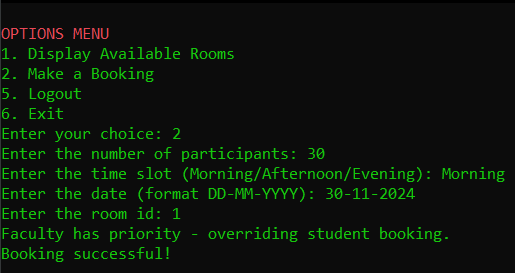
**Display Available Rooms:**



**Make A Booking:**



**Faculty Override Student Bookings:**

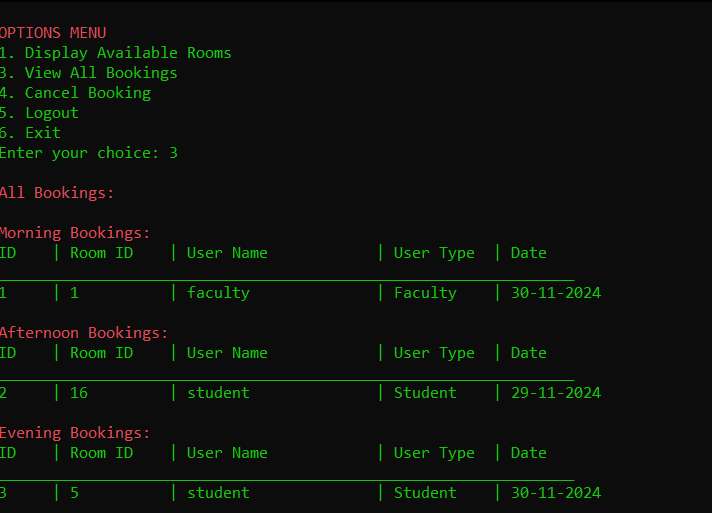


A black screen with green text

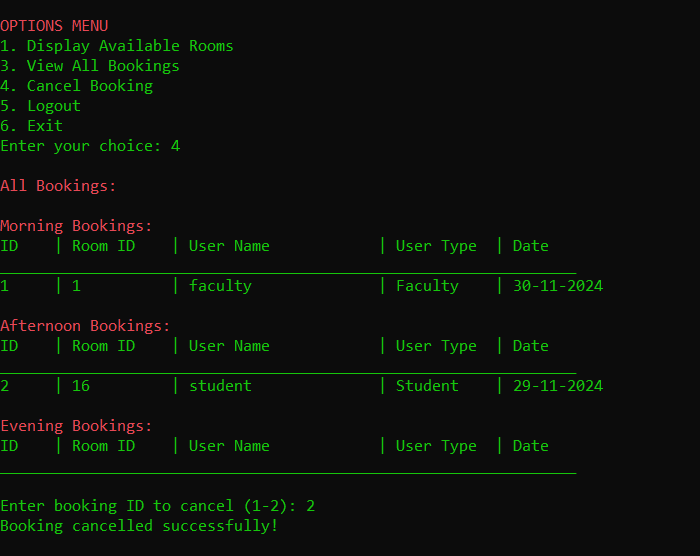
Description automatically generated

**Admin Privileges:**

**View All Bookings:**



**Cancel Bookings:**



**Video Demonstration:**

****