COLLEGE HOSTEL BOOKING SYSTEM

A PROJECT REPORT

Submitted by

NISHAR AHAMED S(2303811710421108)

in partial fulfillment of requirements for the award of the course

CGB1201 - JAVA PROGRAMMING

In

COMPUTER SCIENCE AND ENGINEERING

K. RAMAKRISHNAN COLLEGE OF TECHNOLOGY

(An Autonomous Institution, affiliated to Anna University Chennai and Approved by AICTE, New Delhi)

SAMAYAPURAM–621112

NOVEMBER- 2024

K.RAMAKRISHNANCOLLEGEOFTECHNOLOGY (AUTONOMOUS)

SAMAYAPURAM–621112

BONAFIDECERTIFICATE

Certified that his project report on **" College Hostel Booking System"** is the bonafide work of **NISHAR AHAMED S(2303811710421108)** who carried out the project work during the academic year 2024 - 2025 under my supervision.

SIGNATURE	**SIGNATURE**
Dr.A.Delphin Carolina Rani, M.E.,Ph.D.,	Mrs.K.Valli Priyadharshini, M.E.,(Ph.D.,),
HEAD OF THE DEPARTMENT	**SUPERVISOR**
PROFESSOR	ASSISTANT PROFESSOR
Department of CSE	Department of CSE
K. Ramakrishnan College of Technology (Autonomous)	K. Ramakrishnan College of Technology (Autonomous)
Samayapuram–621112.	Samayapuram–621112.

Submitted for the viva-voce examination held On 03.12.2024

INTERNAL EXAMINER EXTERNAL EXAMINER

DECLARATION

I declare that the project report on **"COLLEGE HOSTEL BOOKING SYSTEM"** is the result of original work done by us and best of our knowledge, similar work as not been submitted to **"ANNA UNIVERSITY CHENNAI"** for the requirement of Degree of **BACHELOR OF ENGINEERING**. This project report is submitted on the partial fulfilment of the requirement of the completion of the course **CGB1201 - JAVA PROGRAMMING.**

.

NISHAR AHAMED S

Place: Samayapuram

Date:03.12.2024

ACKNOWLEDGEMENT

It is with great pride that I express our gratitude and in-debt to our institution "**K.Ramakrishnan College of Technology (Autonomous)**", for providing us with the opportunity to do this project.

I glad to credit honourable chairman **Dr. K. RAMAKRISHNAN**, **B.E.,** for having provided for the facilities during the course of our study in college.

I would like to express our sincere thanks to our beloved Executive Director **Dr. S. KUPPUSAMY, MBA, Ph.D.,** for forwarding to our project and offering adequate duration in completing our project.

I would like to thank **Dr. N. VASUDEVAN, M.Tech., Ph.D.,** Principal, who gave opportunity to frame the project the full satisfaction.

I whole heartily thanks to **Dr. A. DELPHIN CAROLINA RANI, M.E.,Ph.D.,** Head of the department, **COMPUTER SCIENCE AND ENGINEERING** for providing her encourage pursuing this project.

I express our deep expression and sincere gratitude to our project supervisor **Mrs.K.VALLI PRIYADHARSHINI, M.E.,(Ph.D.,),** Department of **COMPUTER SCIENCE AND ENGINEERING,** for his incalculable suggestions, creativity, assistance and patience which motivated us to carry out this project.

I render our sincere thanks to Course Coordinator and other staff members for providing valuable information during the course.

I wish to express our special thanks to the officials and Lab Technicians of our departments who rendered their help during the period of the work progress.

VISION OF THE INSTITUTION

To serve the society by offering top-notch technical education on par with global standards

MISSION OF THE INSTITUTION

- Be a center of excellence for technical education in emerging technologies by exceeding the needs of the industry and society.
- Be an institute with world class research facilities
- Be an institute nurturing talent and enhancing the competency of students to transform them as all-round personality respecting moral and ethical values

VISION OF DEPARTMENT

To be a center of eminence in creating competent software professionals with research and innovative skills.

MISSION OF DEPARTMENT

M1: Industry Specific: To nurture students in working with various hardware and software platforms inclined with the best practices of industry.

M2: Research: To prepare students for research-oriented activities.

M3: Society: To empower students with the required skills to solve complex technological problems of society.

PROGRAM EDUCATIONAL OBJECTIVES

1. PEO1: Domain Knowledge

To produce graduates who have strong foundation of knowledge and skills in the field of Computer Science and Engineering.

2. PEO2: Employability Skills and Research

To produce graduates who are employable in industries/public sector/research organizations or work as an entrepreneur.

3. PEO3: Ethics and Values

To develop leadership skills and ethically collaborate with society to tackle real-world challenges.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Domain Knowledge

To analyze, design and develop computing solutions by applying foundational concepts of Computer Science and Engineering.

PSO 2: Quality Software

To apply software engineering principles and practices for developing quality software for scientific and business applications.

PSO 3: Innovation Ideas

To adapt to emerging Information and Communication Technologies (ICT) to innovate ideas and solutions to existing/novel problems

PROGRAM OUTCOMES (POs)

Engineering students will be able to:

1. **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences
3. **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations
4. **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions

5. **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations

6. **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice

7. **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development

8. **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

9. **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

10. **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

12. **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

ABSTRACT

The College Hostel Booking System is a comprehensive Java application developed to facilitate efficient room allocation and management for students in a hostel environment. This system enables students to view real-time room availability, ensuring transparency and minimizing manual interventions. The application includes a Booking Management Module that dynamically updates room availability upon confirmation, preventing overbooking and ensuring accurate tracking of occupied and vacant rooms. A priority-based allocation system is integrated, which considers criteria such as the application date and special requirements to provide fair and systematic room assignments. With its real-time update capabilities and intuitive interface, the system streamlines the room booking process, enhances user experience, and addresses the operational challenges of managing hostel accommodations.

ABSTRACT WITH POs AND PSOs MAPPING

CO 5 : BUILD JAVA APPLICATIONS FOR SOLVING REAL-TIME PROBLEMS.

ABSTRACT	POs MAPPED	PSOs MAPPED
The College Hostel Booking System is a comprehensive Java application developed to facilitate efficient room allocation and management for students in a hostel environment. This system enables students to view real-time room availability, ensuring transparency and minimizing manual interventions. The application includes a Booking Management Module that dynamically updates room availability upon confirmation, preventing overbooking and ensuring accurate tracking of occupied and vacant rooms.	**PO1 -3** **PO2 -3** **PO3 -3** **PO5 -3** **PO6 -3** **PO10 -3** **PO11-3** **PO12 -3**	**PSO1 -3** **PSO2 -3** **PSO3 -3**

Note: 1- Low, 2-Medium, 3- High

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	viii
1	**INTRODUCTION**	1
	1.1 Objective	1
	1.2 Overview	1
	1.3 Java Programming concepts	2
2	**PROJECT METHODOLOGY**	3
	2.1 Proposed Work	3
	2.2 Block Diagram	3
3	**MODULE DESCRIPTION**	4
	3.1 User Authentication Module	4
	3.2 Room Booking Module	4
	3.3 View Available Rooms Module	4
	3.4 Admin Control Module	5
	3.5 Logout Module	5
4	**CONCLUSION & FUTURE SCOPE**	6
	4.1 Conclusion	6
	4.2 Future Scope	6
	REFERENCES	19
	APPENDIX A (SOURCE CODE)	8
	APPENDIX B (SCREENSHOTS)	16

CHAPTER 1

INTRODUCTION

1.1Objective

The objective of the College Hostel Booking System is to create a Java-based application that allows students to view available rooms in real-time and manage their bookings. The system enables students to log in, book rooms, and check room availability, with immediate updates to prevent overbooking. It prioritizes booking requests based on application date or special requirements. An administrative interface allows room management, including the release of rooms and monitoring of bookings. The application aims to be user-friendly and ensure smooth hostel room allocation. It ensures real-time availability updates and efficient interaction for both students and administrators. The goal is to improve the hostel booking process's transparency and efficiency.

1.2Overview

The College Hostel Booking System is a Java-based application designed to streamline the hostel room allocation process for students and administrators. It provides students with the ability to view available rooms, book them in real-time, and manage their stay details, including room preferences and payment information. The system updates room availability instantly to prevent overbooking and allows for prioritization of requests based on application date or special requirements. Administrators can monitor and manage room bookings, release rooms, and ensure the overall efficiency of the system. The application is designed to be user-friendly, with separate interfaces for students and administrators, enhancing the overall booking experience. By automating and centralizing the booking process, the system 2 reduces administrative workload and improves transparency for both students and staff.

1.3 Java Programming Concepts

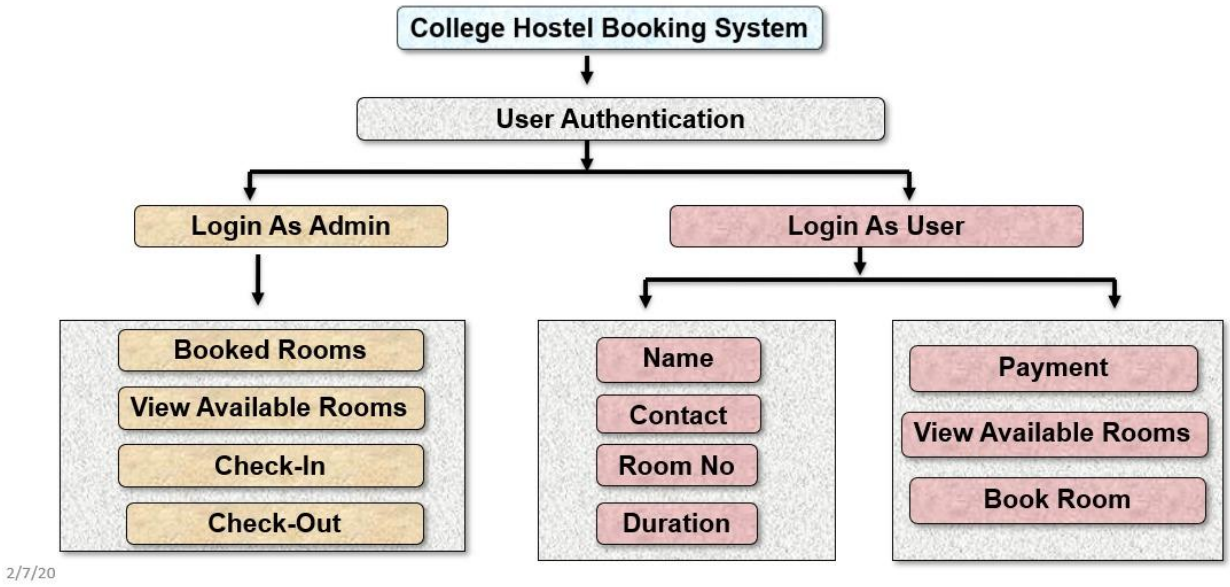
- **Encapsulation**: The concept of bundling the data (attributes) and methods (functions) that operate on the data into a single unit (class). Encapsulation also involves restricting access to some of the object's components to safeguard the internal state.
- **Inheritance:** A mechanism where a new class derives properties and behaviors from an existing class. It allows for code reuse and extension of existing functionality.
- **Polymorphism:** The ability for different classes to be treated as instances of the same class through inheritance, allowing objects of different types to be treated as objects of a common super class.
- **Event Handling:** Used to handle GUI interactions such as button clicks. Implemented using ActionListener or other event listener interfaces.
- **GUI Programming (AWT):** Provides components like Button, Label, TextField, and TextArea to create user interfaces. Example: Using a TextArea to display room availability.
- **Exception Handling:** Handles runtime errors gracefully using try-catch blocks. Essential for input validation and error recovery.

CHAPTER 2
PROJECT METHODOLOGY

2.1Proposed Work

The proposed work for the hostel booking system comprises three key modules: User Interface, Application Logic, and Methods. The User Interface enables users to log in, providing authentication for access. The Application Logic serves as the backbone, managing key processes like user login and room booking. Within this, the login(username, password) function validates credentials. The Methods module contains operations such as Login(), Display Available Rooms(), Display All Room Status(), and Logout(). These methods handle core functionalities, allowing users to view and book rooms. The system ensures efficient interaction between the user interface and application logic. It provides features for administrators to monitor and update room availability. The integration of modules promotes streamlined session management. The entire system is user-friendly, scalable, and practical for managing hostel accommodations effectively.

2.2 Block Diagram



CHAPTER 3

MODULE DESCRIPTION

3.1 User Authentication Module

The User Authentication Module handles user login by requesting a username and password. It verifies the entered credentials against a predefined list or database to ensure the user is authorized. If successful, the user gains access to the system and can proceed with booking or viewing rooms. If the credentials are incorrect, the user is prompted to try again. This module is essential for securing the system and ensuring that only authorized users can perform actions within the application.

3.2 Room Booking Module

The Room Booking Module enables users to reserve a room by entering required details such as their name, contact information, room number, duration of stay, and payment. Upon booking, the system checks if the room is available. If the room is free, the booking is confirmed, and the room status is updated to "booked." The module ensures that only available rooms are booked, and the system manages the room's availability accordingly. This ensures a smooth booking process and prevents double bookings.

3.3 View Available Rooms Module

The View Available Rooms Module allows users to see which rooms are unoccupied and available for booking. It displays a list of all available rooms by checking the current status of each room in the system. If a room is marked as "booked," it will not appear in the list. This module helps users easily identify available options and make an informed decision when choosing a room. It is crucial for streamlining the booking process and enhancing the user experience.

3.4 Admin Control Module

The Admin Control Module is a management interface for administrators, allowing them to log in and access advanced features. Admins can view the status of all rooms (whether booked or available) and can release rooms, making them available for future bookings. This module ensures that admins can efficiently manage room allocations, monitor room usage, and maintain accurate records of the system's operations. It plays a key role in overseeing the entire hostel booking system.

3.5 Logout Module

The Logout Module ensures that users and admins can securely exit the system after completing their tasks. When a user logs out, all session data is cleared to prevent unauthorized access. The module prompts the user to confirm their logout action before signing them out of the system. It is crucial for maintaining security and ensuring that sensitive data is not exposed when the system is accessed by another individual. This helps protect personal information and ensures secure user interaction with the application.

CHAPTER 4
CONCLUSION AND FUTURE SCOPE

4.1 CONCLUSION

The Hostel Booking System provides a user-friendly interface for both users and administrators, enabling efficient room management and booking. It simplifies the booking process by allowing users to log in, check available rooms, and make bookings with minimal effort. The admin control module ensures smooth operations by enabling room management, user authentication, and booking verification. This system streamlines hostel room management, making it more organized and reducing administrative workload. Additionally, it offers the flexibility to release rooms and track room availability in real-time. The system can be easily scaled to handle more rooms and users as required. The future scope of the system includes the integration of advanced features like online payment and mobile access, enhancing user experience. By automating key processes, it minimizes human errors and improves overall operational efficiency. This system is a step toward modernizing hostel management and can be adapted to various institutional requirements. In conclusion, the Hostel Booking System serves as a reliable solution for managing hostel accommodations efficiently and effectively.

4.2 FUTURE SCOPE

The Hostel Booking System can be enhanced by integrating online payment gateways, making the booking process more convenient and secure. A mobile application could be developed to allow users to access the system on-the-go, expanding its reach. Implementing an automated room allocation feature based on availability and user preferences could improve efficiency. Additionally, integrating advanced analytics to track booking trends and room utilization would help administrators optimize operations. A dynamic pricing model based on demand could be introduced for better revenue management. Incorporating AI-based chat

bots for instant customer support would improve user experience. The system could be extended to support multi-location hostel bookings, making it scalable for larger institutions. A loyalty program could be added to incentivize repeat customers. The system could be further optimized for handling group bookings, conferences, or events. Finally, an API for third-party integrations could be developed to connect with external services like travel agencies or local transport providers.

APPENDIX A

```
import java.awt.*;
import java.awt.event.*;
import java.time.LocalDate;
import java.util.*;
class HostelBookingSystem extends Frame {
    private Label titleLabel, usernameLabel, nameLabel, contactLabel,
durationLabel, roomLabel, paymentLabel;
    private TextField usernameField, nameField, contactField, durationField,
roomField, paymentField;
    private Button bookButton, viewButton, loginButton, adminButton,
logoutButton, releaseButton, closeButton;
    private TextArea outputArea;
    private Panel contentPanel;
    private Map<String, Boolean> rooms = new LinkedHashMap<>();
    private String adminUsername = "admin";
    private String adminPassword = "admin123";
    private boolean isAdminLoggedIn = false;
    private boolean isUserLoggedIn = false;
    public HostelBookingSystem() {
        // Initialize frame
        setTitle("College Hostel Booking System");
        setSize(600, 700);
        setLayout(new BorderLayout());
        setResizable(false);
        // Set background color
        contentPanel = new Panel();
        contentPanel.setLayout(new FlowLayout());
```

```
contentPanel.setBackground(new Color(230, 240, 255)); // Mild blue
// Initialize room availability
for (int i = 1; i <= 10; i++) {
   rooms.put("Room " + i, true);
}
// Title Label
titleLabel = new Label("Welcome to Hostel Booking System");
titleLabel.setFont(new Font("Arial", Font.BOLD, 16));
// User Interface Components
usernameLabel = new Label("Username:");
usernameField = new TextField(20);
nameLabel = new Label("Name:");
nameField = new TextField(20);
contactLabel = new Label("Contact:");
contactField = new TextField(20);
durationLabel = new Label("Duration (days):");
durationField = new TextField(20);
roomLabel = new Label("Room No:");
roomField = new TextField(20);
paymentLabel = new Label("Payment Amount:");
paymentField = new TextField(20);
bookButton = new Button("Book Room");
bookButton.addActionListener(e -> bookRoom());
viewButton = new Button("View Available Rooms");
viewButton.addActionListener(e -> displayAvailableRooms());
closeButton = new Button("Close");
closeButton.addActionListener(e -> System.exit(0)); // Closes the application

outputArea = new TextArea(20, 50);
```

```
loginButton = new Button("Login as User");
loginButton.addActionListener(e -> userLogin());
adminButton = new Button("Login as Admin");
adminButton.addActionListener(e -> adminLogin());
logoutButton = new Button("Logout");
logoutButton.addActionListener(e -> logout());
releaseButton = new Button("Release Room");
releaseButton.addActionListener(e -> releaseRoom());
// Adding Components to Panel
contentPanel.add(titleLabel);
contentPanel.add(usernameLabel);
contentPanel.add(usernameField);
contentPanel.add(loginButton);
contentPanel.add(adminButton);
contentPanel.add(logoutButton);
contentPanel.add(releaseButton);
contentPanel.add(nameLabel);
contentPanel.add(nameField);
contentPanel.add(contactLabel);
contentPanel.add(contactField);
contentPanel.add(durationLabel);
contentPanel.add(durationField);
contentPanel.add(roomLabel);
contentPanel.add(roomField);
contentPanel.add(paymentLabel);
contentPanel.add(paymentField);
contentPanel.add(bookButton);
contentPanel.add(viewButton);
contentPanel.add(closeButton);
```

```
    contentPanel.add(outputArea);
    // Add panel to frame
    add(contentPanel, BorderLayout.CENTER);
    // Handle window closing
    addWindowListener(new WindowAdapter() {
        public void windowClosing(WindowEvent e) {
            System.exit(0);
        }
    });
    setVisible(true);
}
private void userLogin() {
    String username = usernameField.getText().trim();
    if (username.isEmpty()) {
        outputArea.setText("Username is required to log in as a user.\n");
        return;
    }
    outputArea.setText("User Login Successful. Welcome, " + username + "!\n");
    isUserLoggedIn = true;
    // Clear all fields for manual entry
    nameField.setText("");
    contactField.setText("");
    durationField.setText("");
    roomField.setText("");
    paymentField.setText("");
    outputArea.append("Please enter your booking details .\n");
}

private void adminLogin() {
```

```
        String username = getInput("Enter Admin Username:");
        String password = getInput("Enter Admin Password:");
        if (username.equals(adminUsername) && password.equals(adminPassword)) {
            isAdminLoggedIn = true;
            outputArea.setText("Admin Login Successful.\nRooms Status:\n");
            displayAllRooms();
        } else {
            outputArea.setText("Invalid Admin Credentials!\n");
        }
    }
    private void displayAllRooms() {
        for (Map.Entry<String, Boolean> room : rooms.entrySet()) {
            outputArea.append(room.getKey() + ": " + (room.getValue() ? "Available" :
"Booked") + "\n");
        }
    }
    private void displayAvailableRooms() {
        outputArea.setText("Available Rooms:\n");
        boolean available = false;
        for (Map.Entry<String, Boolean> room : rooms.entrySet()) {
            if (room.getValue()) { // Check if the room is available
                outputArea.append(room.getKey() + "\n");
                available = true;
            }
        }
        if (!available) {
            outputArea.append("No rooms are available at the moment.\n");
        }
    }
```

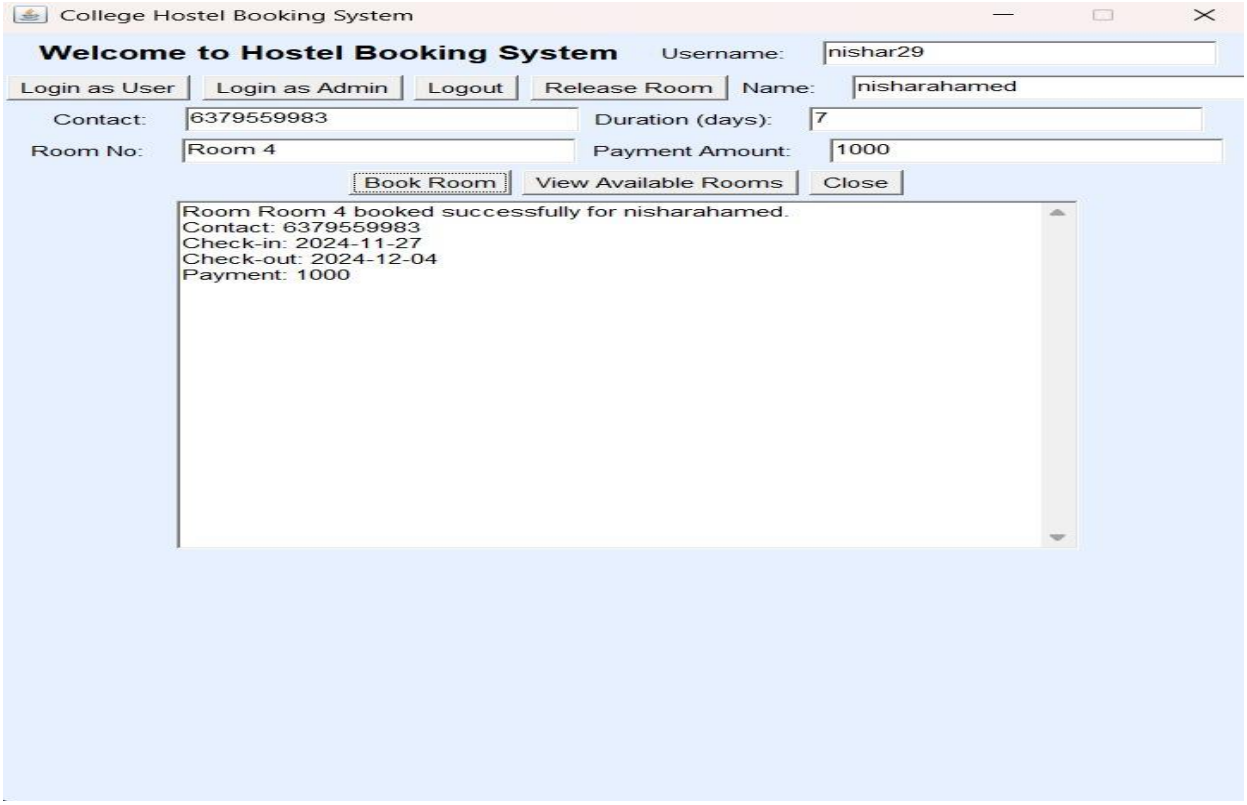
```
    private void bookRoom() {
        if (!isUserLoggedIn) {
            outputArea.setText("Please log in as a user to book a room.\n");
            return;
        }
        String name = nameField.getText();
        String contact = contactField.getText();
        String durationStr = durationField.getText();
        String room = roomField.getText();
        String payment = paymentField.getText();
        if (name.isEmpty() || contact.isEmpty() || durationStr.isEmpty() ||
room.isEmpty() || payment.isEmpty()) {
            outputArea.setText("All fields are required!\n");
            return;
        }
        try {
            int duration = Integer.parseInt(durationStr);
            if (rooms.containsKey(room) && rooms.get(room)) {
                LocalDate checkInDate = LocalDate.now();
               LocalDate checkOutDate = checkInDate.plusDays(duration);
                rooms.put(room, false); // Mark room as booked
                outputArea.setText("Room " + room + " booked successfully for " + name
+ ".\n");
                outputArea.append("Contact: " + contact + "\n");
                outputArea.append("Check-in: " + checkInDate + "\n");
                outputArea.append("Check-out: " + checkOutDate + "\n");
                outputArea.append("Payment: " + payment + "\n");
            } else {
                outputArea.setText("Room " + room + " is not available. Try another
```

```
room.\n");
            }
        } catch (NumberFormatException e) {
            outputArea.setText("Duration must be a valid number!\n");
        }
    }
    private void logout() {
        isAdminLoggedIn = false;
        isUserLoggedIn = false;
        outputArea.setText("You have been logged out.\n");
    }
    private void releaseRoom() {
        if (!isAdminLoggedIn) {
            outputArea.setText("Only admin can release rooms. Please log in as
admin.\n");
            return;
        }
        String room = getInput("Enter room number to release:");
        if (rooms.containsKey(room) && !rooms.get(room)) {
            rooms.put(room, true);
            outputArea.setText("Room " + room + " has been successfully released and
is now available.\n");
        } else {
            outputArea.setText("Room " + room + " is either already available or does
not exist.\n");
        }
    }
    private String getInput(String prompt) {
        return javax.swing.JOptionPane.showInputDialog(this, prompt);
```

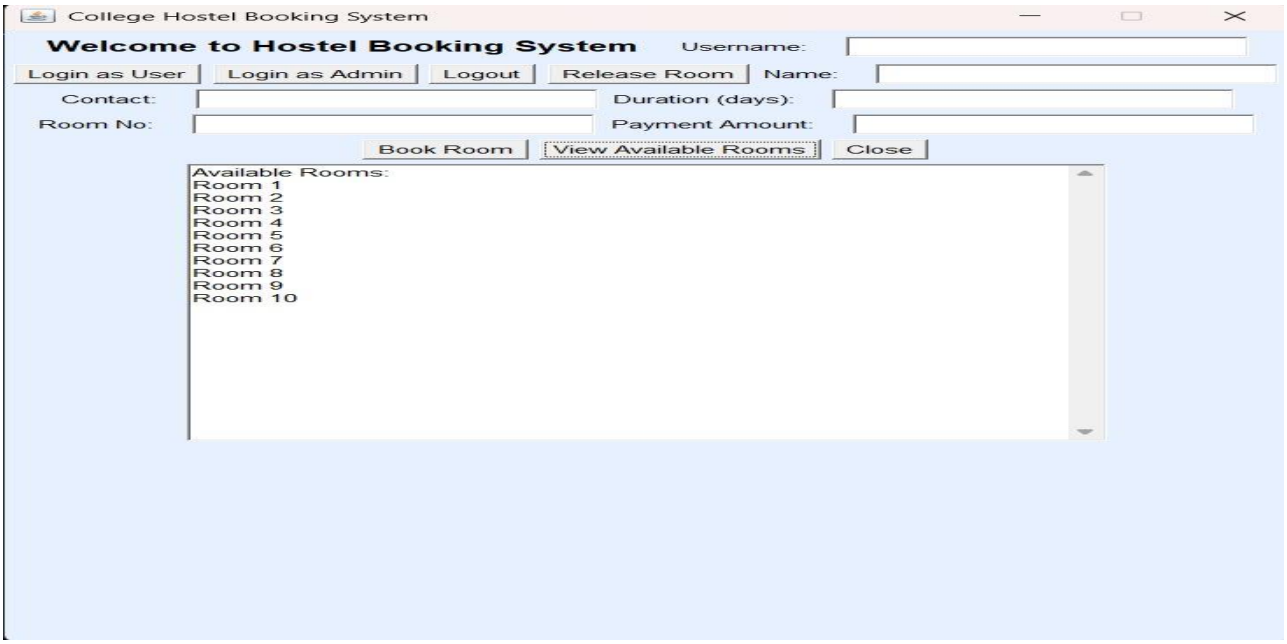
```
    }
    public static void main(String[] args) {
        new HostelBookingSystem();
    }
}
```

APPENDIX B

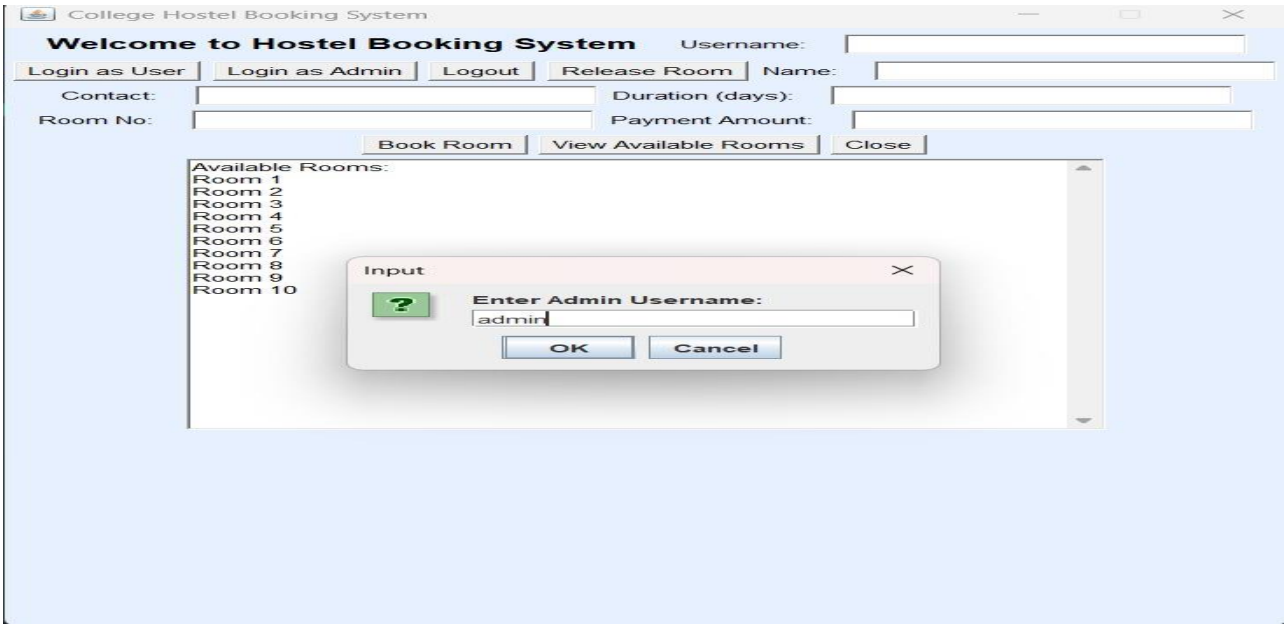
Hostel Room Booking Confirmation Screen

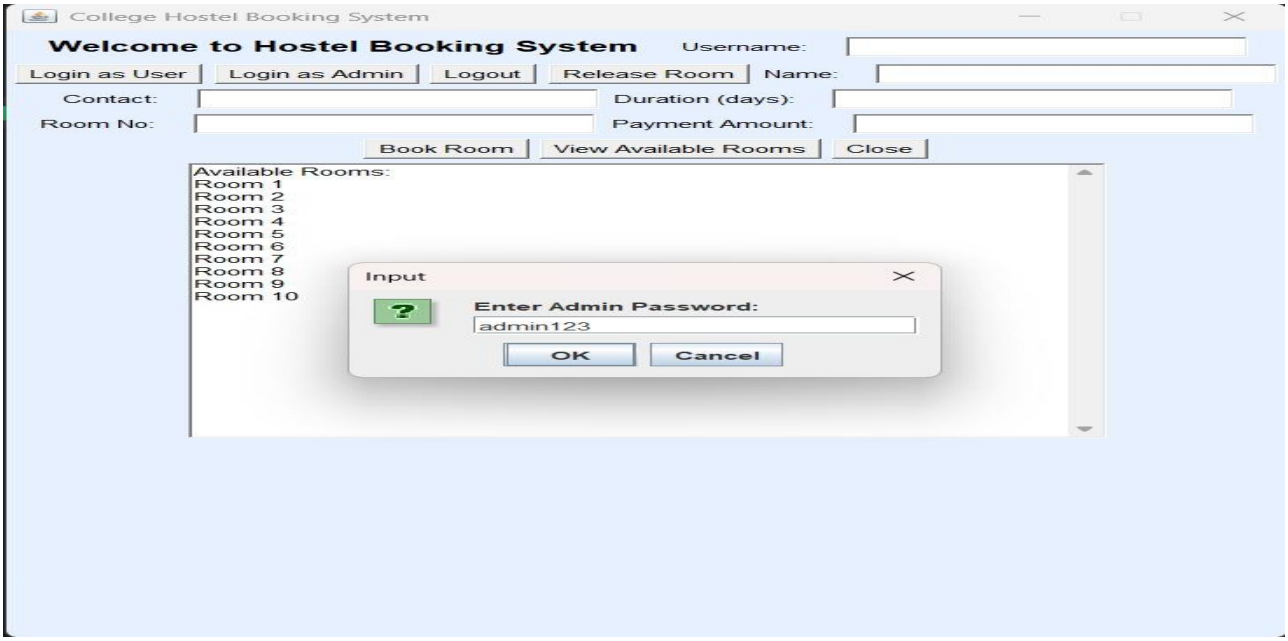


College Hostel Booking System Interface to check the availability of rooms

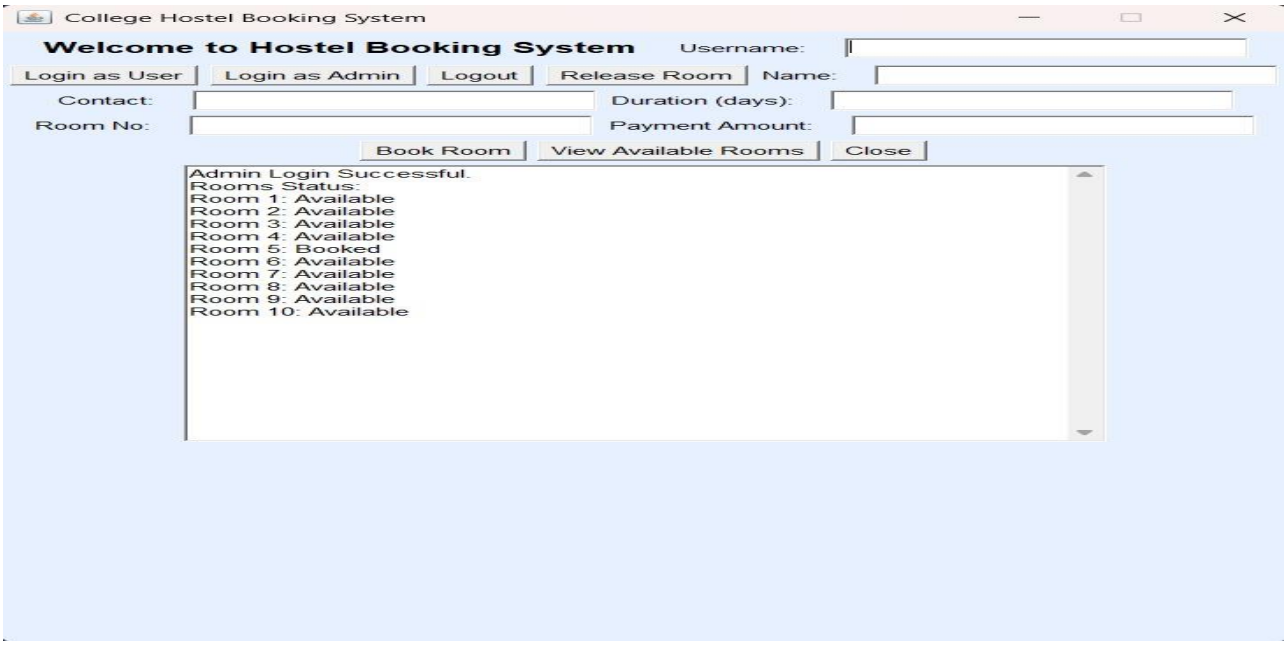


Admin login


College Hostel Booking System
Welcome to Hostel Booking System
Username:
Login as User
Login as Admin
Logout
Release Room
Name:
Contact:
Duration (days):
Room No:
Payment Amount:
Book Room
View Available Rooms
Close
Available Rooms:
Room 1
Room 2
Room 3
Room 4
Room 5
Room 6
Room 7
Room 8
Room 9
Room 10
Input
Enter Admin Username:
admin
OK
Cancel


College Hostel Booking System
Welcome to Hostel Booking System
Username:
Login as User
Login as Admin
Logout
Release Room
Name:
Contact:
Duration (days):
Room No:
Payment Amount:
Book Room
View Available Rooms
Close
Available Rooms:
Room 1
Room 2
Room 3
Room 4
Room 5
Room 6
Room 7
Room 8
Room 9
Room 10
Input
Enter Admin Password:
admin123
OK
Cancel

Admin login availability check



REFERENCES

WEBSITES

- https://www.researchgate.net/publication/380400944_HOSTEL_MANAGEMENT_SYSTEM_PROJECT
- https://github.com/topics/hostel-management-system?l=java&o=asc&s=forks

YOUTUBE LINK

- https://www.youtube.com/watch?v=eVTmf4KNn5g
- https://www.youtube.com/watch?v=_NUzf7IYQ0A

BOOK

- **The Complete Reference** Herbert Schildt
- **The Complete Reference** McGraw-Hill Education, 2018