

**Advanced Programming Practice Project Report on**

**Hotel Management System**

**(Code 21CSC203P)**

**B. Tech (CSE) – 2year/3Semester**

**nd rd**

**Group Members : Kaushal Mathur 116**

**Arshdeep Singh 112**

**Samarth Chaturvedi 91**

**Kumar Aditya 71**



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**FACULTY OF ENGINEERING & TECHNOLOGY**

SRM INSTITUTE OF SCIENCE & TECHNOLOGY

DELHI NCR CAMPUS, MODINAGAR

SRM INSTITUTE OF SCIENCE & TECHNOLOGY, NCR CAMPUS

**MAY 2024**

(Under Section 3 of UGC Act, 1956)

**BONAFIDE CERTIFICATE**

*Certified to be the bonafide record of work done by* **Kaushal Mathur RA2311026030116, Arshdeep Singh RA2311026030112,** **Samarth Chaturvedi RA2311026030091, Kumar Aditya RA2311026030071** *of 3rd semester 2nd year* ***B.TECH*** *degree course in* ***SRM INSTITUTE OF SCIENCE AND***  ***TECHNOLOGY, NCR Campus*** *of Department of Computer Science & Engineering in Advance programming practices, during the academic year 2023-2024.*

|  |  |
| --- | --- |
| **SIGNATURE** | **SIGNATURE** |
| Ms.Swati Sheoran | Dr. Avneesh Vashistha |
| Assistant Professor | Head of the Department |
| Computer Science & Engineering | Computer Science & Engineering |

# Odd Semester (2024-2025)

## Project Report: Page 1



**Title: Hotel Management System**



**Team Members: Kaushal Mathur RA2311026030116, Arshdeep Singh RA2311026030112,** **Samarth Chaturvedi RA2311026030091, Kumar Aditya RA2311026030071**



**Abstract:** The Hotel Management System is a comprehensive software solution built using Java and MySQL, aimed at automating the operations of a hotel. The system offers two primary interfaces: one for customers to book rooms and order food online, and another for administrators to manage room availability and restaurant menus. This system enhances customer experience by providing online services, while helping hotel administrators efficiently manage daily tasks.



**Objective:** The goal of this project is to create a user-friendly platform that automates the core functionalities of hotel management, including room bookings, food orders, and administrative updates, thereby improving operational efficiency and customer satisfaction.



**Technologies Used:**

* **Java:** Core programming language for backend logic and system operations.

•**MySQL:** Database management system to store and manage all hotel data, including bookings, menus, and customer information.



**System Overview:** The Hotel Management System facilitates seamless interactions between the hotel administration and its customers. Customers can book rooms online, check availability in realtime, and place food orders directly from the hotel’s restaurant. Administrators can update room availability and modify the restaurant menu through an intuitive admin panel, keeping the system data up to date. The system’s backend is powered by Java, with MySQL used for managing the relational database.

## Project Report: Page 2



**Project Overview**



**Introduction to the System:** The Hotel Management System is a software application designed to manage a hotel's operations efficiently through digital solutions. The system is built using **Java** for the backend logic and **MySQL** for the database, providing a robust and scalable platform for managing hotel services. The system offers two interfaces: one for customers and one for administrators.

•**Customer Interface:** Allows guests to book rooms and order food from the hotel restaurant online, making the experience seamless and hassle-free.

•**Admin Interface:** Enables hotel administrators to manage room availability and restaurant menus through an intuitive dashboard, ensuring that all information is updated in real-time.

By automating these essential tasks, the system helps reduce manual workload, improves accuracy, and enhances customer satisfaction.



**Objective:** The primary objective of the Hotel Management System is to **automate key hotel management tasks** such as room bookings, food orders, and updates to room availability and restaurant menus. This automation streamlines hotel operations, reducing the need for manual processes and enabling efficient management of services. Additionally, the system aims to provide a better user experience for customers by allowing them to handle their bookings and orders online, with real-time updates.



**Key Goals:**

•**Improve Efficiency:** Automate routine administrative tasks such as updating room availability and managing the restaurant menu.

•**Enhance Customer Experience:** Allow customers to book rooms and order food from their mobile devices or computers, with live data on room availability.

•**Real-Time Data Management:** Ensure that all bookings, orders, and menu changes are reflected instantly in the system to avoid conflicts and errors.

## Project Report: Page 3



**Customer Features**



The Hotel Management System offers a range of features aimed at enhancing the customer experience by providing online solutions for room bookings and restaurant services. These features are designed to be user-friendly, allowing customers to manage their hotel stay easily and efficiently.



1. **Room Booking:**

Customers can book rooms online through a simple and intuitive interface. The system provides real-time updates on room availability, allowing customers to:

•**View Available Rooms:** Customers can check the availability of different room types (e.g., single, double, suite) based on their desired dates.

* + **Filter by Preferences:** The system allows customers to filter rooms by price, amenities, and room type to find the option that best suits their needs.

•**Booking Confirmation:** Once the customer selects a room, they can confirm their booking with a few clicks, receiving an instant confirmation with details about their stay.

•**Real-Time Updates:** The system automatically updates room availability to prevent doublebooking, ensuring that the information presented is always accurate.



1. **Food Ordering from the Restaurant:**

Customers can order food from the hotel’s restaurant through the same online platform used for booking rooms. This feature allows guests to enjoy restaurant services conveniently without leaving their rooms. Key functionalities include:

•**View Menu:** Customers can browse the hotel restaurant’s menu, which includes detailed descriptions, prices, and availability of each dish.

* + **Place Orders:** Customers can select items from the menu and place an order directly from their room. The system provides an estimated delivery time for the food.
  + **Order History:** Customers can view their past orders, allowing them to reorder their favorite meals easily.

•**Real-Time Menu Updates:** The menu is updated in real-time by the admin to reflect changes in dish availability, ensuring that customers only see what is currently available.



**User Convenience:**

Both the room booking and food ordering features are designed with user convenience in mind, reducing the need for in-person interactions and allowing customers to manage their stay at their own pace. These features ensure a smooth and enjoyable experience, contributing to customer satisfaction and operational efficiency for the hotel. **Project Report: Page 4**



**Admin Features**



The Hotel Management System includes a dedicated admin interface, which allows hotel staff to manage key operational aspects efficiently. The system is designed to give administrators complete control over room availability and restaurant menu management, ensuring smooth hotel operations.



1. **Menu Management:**

Administrators have full control over the hotel restaurant's menu. This feature allows admins to manage menu items effectively through the following functionalities:

•**Add Items:**

* + Admins can add new dishes to the restaurant menu, specifying details such as the dish name, description, price, and availability.

•**Update Items:**

* + The system allows admins to modify existing menu items, including changing the price, updating descriptions, or adjusting availability based on kitchen inventory.

•**Delete Items:**

* + Admins can remove dishes that are no longer offered by the restaurant, ensuring that customers see an up-to-date and accurate menu at all times.

•**Real-Time Updates:**

* + Any changes made to the menu are instantly reflected in the customer interface, ensuring that only available dishes are shown to customers ordering food.



1. **Room Availability Updates:**

The system simplifies the process of managing room availability, allowing admins to update room statuses in real-time. This ensures accurate information is available for both the hotel staff and customers. Key functionalities include:

•**Set Room Status (Available/Unavailable):**

* + Admins can mark rooms as available or unavailable depending on reservations, maintenance, or other factors. This helps prevent overbooking and ensures customers always see the correct availability.

•**Manage Room Details:** Admins can modify room details such as room type, price, and features (e.g., number of beds, included amenities) to reflect current offerings.

**Operational Efficiency: By providing these powerful management tools, the admin interface ensures that hotel operations run smoothly with minimal manual effort. The system reduces errors related to room booking and menu updates, improving overall efficiency and accuracy.**

## Project Report: Page 5



**System Architecture**



The Hotel Management System is designed with a robust and scalable architecture that integrates a Java-based backend with a MySQL database for data management. This architecture ensures smooth communication between the customer and admin interfaces, with efficient handling of data and system operations.



1. **Java-Based Application:**
   * + **Backend Logic:**
     + The core logic of the system is developed using **Java**, a powerful and versatile programming language known for its reliability and performance. Java handles all the business logic, including room booking, food ordering, and admin functionalities such as updating room availability and managing the menu.
     + **Object-Oriented Design:**
     + The system follows an object-oriented design, breaking down each feature into manageable components such as Customer, Room, MenuItem, and Order. This structure improves code readability, maintainability, and scalability, making it easier to add new features in the future.

•**Real-Time Processing:**

* + - Java's multi-threading capabilities allow the system to handle multiple user requests simultaneously. For example, several customers can book rooms or order food at the same time without conflicts or delays.
    - **User Interfaces:**
    - The system provides two interfaces: one for customers and one for administrators. Both interfaces are integrated with the backend using Java, ensuring smooth and secure data flow between the frontend and backend.



1. **MySQL Database:**

•**Database Management:**

•The system uses **MySQL**, a relational database management system, to store and manage all hotel-related data, including room bookings, customer information, restaurant menus, and orders. MySQL ensures that data is securely stored and can be retrieved quickly when needed.

* + - * **Tables and Relationships:**
      * The database consists of multiple tables, each storing specific information:
      * Customers: Stores customer details (name, contact information, booking

•

* + - * history).
      * Rooms: Contains data about room availability, types, prices, and booking
      * status.

Menu: Stores information about available food items, prices, and availability.

Orders: Tracks food orders placed by customers.

Admins: Stores admin login credentials for secure access to the system.

* + - * **Data Integrity and Relationships:**
      * The system uses **foreign keys** to establish relationships between tables (e.g., linking customer bookings to rooms or orders to menu items), ensuring that the data is consistent and properly linked.



1. **Client-Server Architecture:**

•The Hotel Management System follows a **client-server architecture**, where the client side

(customer and admin interfaces) sends requests to the server (Java backend), which processes the request and communicates with the database (MySQL) to retrieve or update data.

* + **Flow of Operations:**
  + A typical operation, such as room booking, follows this flow:

1.The customer initiates a room booking request via the customer interface.

2.The Java backend processes the request, checks the availability of the room by querying the MySQL database.

3.If the room is available, the booking is confirmed, and the room status is updated in the database.

4.The customer receives a confirmation message through the interface.



**Scalability and Security:**

* + **Scalability:** The architecture is designed to scale easily, allowing the system to handle increasing numbers of customers, rooms, and menu items without compromising performance.
  + **Security:** Java's built-in security features, combined with MySQL's authentication and encryption mechanisms, ensure that customer data, bookings, and admin operations are secure.

## Project Report: Page 6



**Database Design**



The Hotel Management System's database is built using **MySQL**, which manages all the essential data related to hotel operations. The database consists of several tables that store information about customers, rooms, menu items, orders, and admin credentials. These tables are interlinked to ensure efficient data retrieval and to maintain data integrity across the system.



**Key Tables:**

1.**Customers Table:**

* **Purpose:** Stores customer information, such as personal details and booking history.
* **Key Fields:**
* customer\_id: Unique identifier for each customer (Primary Key).
* name: Full name of the customer.
* contact\_info: Contact details (e.g., phone number, email).
* booking\_history: Links to the orders or room bookings made by the customer.

2.**Rooms Table:**

•**Purpose:** Manages the room details and availability status for the hotel.

* **Key Fields:**
* room\_id: Unique identifier for each room (Primary Key).
* room\_type: Type of room (e.g., single, double, suite).
* price: Cost per night for the room.
* availability: Boolean value indicating whether the room is available or booked.

3.**Menu Table:**

* **Purpose:** Stores information about the dishes available in the hotel restaurant.
* **Key Fields:**
* menu\_item\_id: Unique identifier for each menu item (Primary Key).
* item\_name: Name of the food item.
* price: Price of the item.
* availability: Boolean value indicating if the item is currently available.

4.**Orders Table:**

* **Purpose:** Tracks customer food orders placed through the system.
* **Key Fields:**
* order\_id: Unique identifier for each order (Primary Key).
* customer\_id: Links to the customer who placed the order (Foreign Key).
* menu\_item\_id: Links to the ordered food item (Foreign Key).
* order\_status: Status of the order (e.g., pending, delivered).

5.**Admins Table:**

* **Purpose:** Stores credentials and access rights for system administrators.
* **Key Fields:**
* admin\_id: Unique identifier for each admin (Primary Key).
* username: Admin’s username for login.
* password: Hashed password for security purposes.



**Relationships Between Tables:**

•**Customer & Orders:**

* Each customer can place multiple orders, creating a one-to-many relationship between the Customers and Orders tables.

•**Room Availability:**

* The Rooms table is linked with the booking process, ensuring that room availability is automatically updated after each confirmed booking.

•**Menu Management:**

* Admins can update the Menu table, which is directly linked to the Orders table, ensuring that only available dishes are shown to customers.



**Database Design Considerations:**

•**Normalization:** The database is normalized to avoid redundancy and ensure efficient data storage.

* **Data Integrity:** Foreign key constraints ensure that records in related tables are consistent • and valid.

**Scalability:** The design supports scalability, allowing for the easy addition of more rooms, customers, and menu items as the system grows.





## Project Report: Page 7

**Customer Interface**



The **Customer Interface** is the front-facing aspect of the Hotel Management System, where guests interact with the system to book rooms and order food. This interface is designed to be userfriendly, enabling customers to complete their desired actions with minimal effort.



**Key Features:**

1.**Room Booking:**

•**Room Availability Search:** Customers can search for available rooms based on their preferred dates. The system presents a list of available rooms, categorized by type (e.g., single, double, suite).

* **Booking Process:** Customers can select a room, view its details (price, amenities), and proceed with the booking. After booking, the system confirms the reservation and updates the room's availability in the system.
* **Booking History:** Customers can view their previous bookings and upcoming reservations through the interface.

*Example Screenshot:*

* A screenshot displaying the room availability search, showing room types, prices, and availability for selected dates.

2.**Food Ordering:**

•**Menu Browsing:** Customers can browse the restaurant’s menu through the interface. The menu is displayed in categories (e.g., appetizers, main courses, desserts), with each item showing its name, price, and availability.

•**Order Placement:** After selecting the items, customers can place their order with just a few clicks. The system confirms the order and provides an estimated delivery time.

* **Order Tracking:** Customers can track the status of their orders (e.g., pending, in progress, delivered) through the interface.

*Example Screenshot:*

* A screenshot showing the menu browsing interface, with available dishes, their prices, and an "Add to Cart" option for placing orders.



**User Experience:**

* The design of the customer interface focuses on **ease of navigation**. Both the room booking and food ordering processes are simplified, requiring only a few steps to complete.

•The interface includes **real-time updates**, ensuring that customers are always presented with the latest information about room availability and menu options.

* Customers can manage all aspects of their stay directly from the interface, reducing the need for in-person interactions and streamlining the hotel experience.

## Project Report: Page 8



**Admin Interface**



The **Admin Interface** is designed to give hotel administrators full control over the core operations of the hotel, such as managing room availability and updating the restaurant menu. This interface ensures that admins can efficiently handle hotel management tasks with real-time updates to the system.



**Key Features:**

1.**Menu Management:**

•**Add, Update, Delete Items:** Administrators can add new dishes to the restaurant menu, update existing items (e.g., change prices, descriptions), and delete items that are no longer available.

•**Real-Time Updates:** Any changes made by the admin are reflected instantly in the customer-facing interface, ensuring that only currently available dishes are shown to customers.

*Example Screenshot:*

* A screenshot showcasing the admin’s view of the menu management section, where they can see a list of menu items with options to add, edit, or delete items.

2.**Room Availability Management:**

•**Update Room Status:** Admins can mark rooms as available or unavailable in real time based on reservations or maintenance requirements. This helps ensure that customers only see rooms that are available for booking.

•**Manage Room Details:** Administrators can modify room details such as price, room type, and amenities directly from the interface.

*Example Screenshot:*

* A screenshot displaying the room availability management interface, showing a list of rooms with their current status and an option to update the availability.



**Efficiency:** The Admin Interface simplifies the management of both the hotel rooms and the restaurant menu, reducing the time and effort required for manual updates. This automation helps avoid errors such as overbooking rooms or displaying unavailable menu items to customers, ensuring smooth and efficient hotel operations.



## Project Report: Page 9



**Advantages**



The Hotel Management System offers several advantages that benefit both the hotel administration and its customers. By automating key hotel management tasks and providing online solutions for customers, the system significantly improves operational efficiency and customer convenience.



1. **Automation:**

•**Reduced Manual Workload:** Tasks such as room bookings, food orders, and menu updates are automated, reducing the need for manual data entry and minimizing the chances of human error.

•**Real-Time Updates:** The system ensures that all updates related to room availability and menu items are reflected in real-time. This reduces the risk of overbooking or ordering unavailable dishes, improving the overall efficiency of the hotel.

1. **Improved Efficiency:**

•**Streamlined Operations:** By integrating room management, food ordering, and menu updates into one system, the hotel staff can manage day-to-day operations more effectively. The admin interface allows for quick and easy updates to key hotel services, ensuring that the hotel runs smoothly with minimal downtime.

•**Centralized Data Management:** All hotel-related data, including customer information, bookings, and orders, is stored in a centralized MySQL database, making it easier for admins to access and manage information.

1. **Enhanced Customer Convenience:**

•**Online Room Booking:** Customers can check room availability and book rooms online without the need to call the hotel or visit in person. This convenience encourages more bookings and provides a better user experience.

• **Easy Food Ordering:** Customers can browse the restaurant menu, place food orders, and have meals delivered to their rooms with just a few clicks. The process is simplified and can be completed entirely online, enhancing the guest experience.

•**Real-Time Information:** Both room availability and menu items are updated in real time, ensuring that customers always have the latest information when making their bookings or food orders.



**Conclusion:** The Hotel Management System provides significant advantages to both the hotel staff and its guests. By automating core tasks, improving efficiency, and offering convenient online services, the system enhances the overall hotel experience, leading to higher customer satisfaction and smoother operations.

## Project Report: Page 10



**Challenges**



During the development and implementation of the Hotel Management System, several challenges were encountered. These challenges required careful consideration and problem-solving to ensure the system operated smoothly and efficiently.



1. **Database Synchronization:**
   * **Challenge:** Ensuring that the data between the front-end interfaces (customer and admin) and the back-end MySQL database remained synchronized in real-time was a significant challenge. With multiple users accessing and updating the system simultaneously (e.g., room bookings, food orders), maintaining accurate and up-to-date information across the platform was critical.

•**Solution:** We implemented database locking mechanisms and real-time data synchronization techniques to ensure that any changes made by the admin or customers were instantly reflected in the database and across all interfaces.

1. **UI Design:**
   * **Challenge:** Designing a user-friendly interface for both customers and administrators posed challenges in terms of balancing functionality with simplicity. The customer interface needed to be intuitive, while the admin interface required advanced features for managing rooms and menus.

•**Solution:** We adopted a minimalist design approach for the customer interface, focusing on ease of navigation, while the admin interface was designed to provide detailed controls without overwhelming the user. Regular feedback from users helped refine the design.

1. **Error Handling:**
   * **Challenge:** Handling errors such as invalid inputs, failed database connections, and unexpected user behaviors was essential for ensuring the system’s robustness.

•**Solution:** We implemented comprehensive error-handling mechanisms, including input validation on both the front-end and back-end, database transaction rollbacks, and clear error messages for users. This helped maintain system stability and improve the user experience.



**Overcoming Challenges:** By addressing these challenges through thoughtful design and effective coding practices, the system is able to provide reliable, accurate, and efficient hotel management services.

**CODE**

**Main Class (Entry Point)**

java

Copy code

import java.util.Scanner;

public class HotelManagementSystem {

public static void main(String[] args) {

Scanner scanner = new Scanner(System.in);

AdminRoomManagement adminRoomManagement = new AdminRoomManagement();

AdminMenuManagement adminMenuManagement = new AdminMenuManagement();

CustomerBooking customerBooking = new CustomerBooking();

CustomerFoodOrder customerFoodOrder = new CustomerFoodOrder();

while (true) {

System.out.println("Welcome to the Hotel Management System");

System.out.println("1. Customer");

System.out.println("2. Admin");

System.out.println("3. Exit");

System.out.print("Choose your role: ");

int roleChoice = scanner.nextInt();

switch (roleChoice) {

case 1:

customerMenu(customerBooking, customerFoodOrder, scanner);

break;

case 2:

adminMenu(adminRoomManagement, adminMenuManagement, scanner);

break;

case 3:

System.out.println("Thank you for using the system. Goodbye!");

System.exit(0);

default:

System.out.println("Invalid choice. Try again.");

}

}

}

private static void customerMenu(CustomerBooking customerBooking, CustomerFoodOrder customerFoodOrder, Scanner scanner) {

while (true) {

System.out.println("\nCustomer Menu:");

System.out.println("1. View available rooms");

System.out.println("2. Book a room");

System.out.println("3. View menu");

System.out.println("4. Order food");

System.out.println("5. Back to main menu");

System.out.print("Choose an option: ");

int customerChoice = scanner.nextInt();

switch (customerChoice) {

case 1:

customerBooking.viewAvailableRooms();

break;

case 2:

System.out.print("Enter your name: ");

scanner.nextLine(); // Consume newline

String name = scanner.nextLine();

System.out.print("Enter Room ID to book: ");

int roomId = scanner.nextInt();

System.out.print("Enter check-in date (YYYY-MM-DD): ");

String checkIn = scanner.next();

System.out.print("Enter check-out date (YYYY-MM-DD): ");

String checkOut = scanner.next();

customerBooking.bookRoom(name, roomId, checkIn, checkOut);

break;

case 3:

customerFoodOrder.viewMenu();

break;

case 4:

System.out.print("Enter your name: ");

scanner.nextLine(); // Consume newline

String customerName = scanner.nextLine();

System.out.print("Enter Food ID to order: ");

int foodId = scanner.nextInt();

customerFoodOrder.orderFood(customerName, foodId);

break;

case 5:

return; // Go back to the main menu

default:

System.out.println("Invalid choice. Try again.");

}

}

}

private static void adminMenu(AdminRoomManagement adminRoomManagement, AdminMenuManagement adminMenuManagement, Scanner scanner) {

while (true) {

System.out.println("\nAdmin Menu:");

System.out.println("1. Add new room");

System.out.println("2. Add new menu item");

System.out.println("3. Back to main menu");

System.out.print("Choose an option: ");

int adminChoice = scanner.nextInt();

switch (adminChoice) {

case 1:

System.out.print("Enter Room Type: ");

scanner.nextLine(); // Consume newline

String roomType = scanner.nextLine();

System.out.print("Enter Room Price: ");

double roomPrice = scanner.nextDouble();

adminRoomManagement.addRoom(roomType, roomPrice);

break;

case 2:

System.out.print("Enter Food Name: ");

scanner.nextLine(); // Consume newline

String foodName = scanner.nextLine();

System.out.print("Enter Food Price: ");

double foodPrice = scanner.nextDouble();

adminMenuManagement.addMenuItem(foodName, foodPrice);

break;

case 3:

return; // Go back to the main menu

default:

System.out.println("Invalid choice. Try again.");

}

}

}

}

**2. Database Connection Class**

java

Copy code

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class DBConnection {

private static final String URL = "jdbc:mysql://localhost:3306/hotel\_management";

private static final String USER = "root";

private static final String PASSWORD = "password";

public static Connection getConnection() throws SQLException {

return DriverManager.getConnection(URL, USER, PASSWORD);

}

}

**3. CustomerBooking Class**

java

Copy code

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

public class CustomerBooking {

public void bookRoom(String customerName, int roomId, String checkIn, String checkOut) {

try (Connection conn = DBConnection.getConnection()) {

String query = "INSERT INTO bookings (customer\_name, room\_id, check\_in\_date, check\_out\_date) VALUES (?, ?, ?, ?)";

PreparedStatement stmt = conn.prepareStatement(query);

stmt.setString(1, customerName);

stmt.setInt(2, roomId);

stmt.setString(3, checkIn);

stmt.setString(4, checkOut);

stmt.executeUpdate();

System.out.println("Room booked successfully!");

} catch (SQLException e) {

e.printStackTrace();

}

}

public void viewAvailableRooms() {

try (Connection conn = DBConnection.getConnection()) {

String query = "SELECT \* FROM rooms WHERE is\_available = TRUE";

PreparedStatement stmt = conn.prepareStatement(query);

ResultSet rs = stmt.executeQuery();

while (rs.next()) {

System.out.println("Room ID: " + rs.getInt("room\_id") + ", Type: " + rs.getString("room\_type") + ", Price: " + rs.getDouble("price"));

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**4. CustomerFoodOrder Class**

java

Copy code

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

public class CustomerFoodOrder {

public void orderFood(String customerName, int foodId) {

try (Connection conn = DBConnection.getConnection()) {

String query = "INSERT INTO food\_orders (customer\_name, food\_id, order\_date) VALUES (?, ?, NOW())";

PreparedStatement stmt = conn.prepareStatement(query);

stmt.setString(1, customerName);

stmt.setInt(2, foodId);

stmt.executeUpdate();

System.out.println("Food ordered successfully!");

} catch (SQLException e) {

e.printStackTrace();

}

}

public void viewMenu() {

try (Connection conn = DBConnection.getConnection()) {

String query = "SELECT \* FROM menu";

PreparedStatement stmt = conn.prepareStatement(query);

ResultSet rs = stmt.executeQuery();

while (rs.next()) {

System.out.println("Food ID: " + rs.getInt("food\_id") + ", Name: " + rs.getString("food\_name") + ", Price: " + rs.getDouble("price"));

}

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**5. AdminRoomManagement Class**

java

Copy code

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.SQLException;

public class AdminRoomManagement {

public void addRoom(String roomType, double price) {

try (Connection conn = DBConnection.getConnection()) {

String query = "INSERT INTO rooms (room\_type, price) VALUES (?, ?)";

PreparedStatement stmt = conn.prepareStatement(query);

stmt.setString(1, roomType);

stmt.setDouble(2, price);

stmt.executeUpdate();

System.out.println("Room added successfully!");

} catch (SQLException e) {

e.printStackTrace();

}

}

}

**6. AdminMenuManagement Class**

java

Copy code

import java.sql.Connection;

import java.sql.PreparedStatement;

import java.sql.SQLException;

public class AdminMenuManagement {

public void addMenuItem(String foodName, double price) {

try (Connection conn = DBConnection.getConnection()) {

String query = "INSERT INTO menu (food\_name, price) VALUES (?, ?)";

PreparedStatement stmt = conn.prepareStatement(query);

stmt.setString(1, foodName);

stmt.setDouble(2, price);

stmt.executeUpdate();

System.out.println("Menu item added successfully!");

} catch (SQLException e) {

e.printStackTrace();

}

}

}



## Project Report: Page 11



**Future Scope**



The Hotel Management System has the potential for further enhancements and features that could significantly improve its usability and functionality. Some areas for future development include:



1. **Mobile Application:**
   * **Description:** Developing a mobile app version of the system would allow users to access hotel services from their smartphones or tablets. A mobile app would enhance user convenience, enabling them to book rooms, order food, and manage their bookings on the go.
   * **Benefits:** A mobile app would expand the system’s reach, providing an easier and more accessible way for customers to interact with the hotel. It would also offer push notifications for booking confirmations, order status updates, and special promotions.
2. **Payment Gateway Integration:**
   * **Description:** Integrating a secure payment gateway would allow customers to make payments online for their room bookings and food orders. This would streamline the checkout process and eliminate the need for in-person payments.
   * **Benefits:** Online payment options would improve customer convenience and speed up the booking process. Integrating popular payment methods like credit/debit cards, digital wallets, and UPI would also cater to a wider range of customers.
3. **AI Integration:**
   * **Description:** Incorporating artificial intelligence (AI) into the system could introduce features such as personalized recommendations, chatbot-based customer support, and predictive analytics for hotel management.
   * **Benefits:** AI-powered features would enhance the customer experience by offering tailored room and menu suggestions based on past behavior. For the admin, AI could analyze booking patterns, optimize resource allocation, and predict peak periods, improving overall efficiency.



**Long-Term Vision:** The future scope of the system aims to enhance its functionality and improve both customer and administrative experiences by adopting modern technologies like mobile platforms, secure payments, and AI.





## Project Report: Page 12



**Conclusion**



The Hotel Management System successfully automates the core functions of hotel operations, including room bookings, food orders, and administrative tasks. By integrating a **Java-based backend** with a **MySQL database**, the system provides a reliable and scalable solution for managing hotel services.



**Key Benefits:**

•**Improved Efficiency:** The system reduces manual workload for hotel staff by automating tasks such as room availability updates and menu management.

•**Enhanced Customer Experience:** Guests can conveniently book rooms and order food online through a user-friendly interface, improving the overall guest experience.

•**Real-Time Data:** The system ensures real-time updates for room availability and menu options, ensuring that both customers and administrators always have access to accurate information.



**Final Thoughts:** The system demonstrates how modern technology can be applied to **optimize hotel management processes**. Through automation, the hotel can improve operational efficiency and enhance the experience for both customers and staff. With potential future developments such as a mobile app, online payments, and AI integration, the Hotel Management System can evolve into an even more comprehensive solution for the hospitality industry.