**HOTEL BOOKING SYSTEM**

A Comprehensive SQL-Based Project

Group 4 – B.Tech 2nd Year

GROUP MEMBERS

**1. CHAYAN MALLICK-23CS8016**

**2. PRIYANK SURYAVANSHI-23CS8017**

**3. B DURGA SAI ROHAN -23CS8018**

**4. AYUSH KUMAR-23CS8019**

**5. SOMNATH REDDY-23CS8020**

**GITHUB LINK:** [**https://github.com/ChayanMallick13/DBMS\_GROUP4\_MINIPROJECT/tree/main**](https://github.com/ChayanMallick13/DBMS_GROUP4_MINIPROJECT/tree/main)

TABLE OF CONTENTS

| Content | Page Number |
| --- | --- |
| Abstract | 4 |
| Introduction | 5 |
| System Requirements & Objectives | 6 |
| Database Design & Implementation Details | 7 |
| ER Diagram | 9 |
| Advanced SQL Features | 10 |
| System Functionality | 11 |
| Testing & Future Enhancements | 12 |
| Acknowledgement | 14 |
| Bibliography | 15 |
| Conclusion | 16 |

Abstract

This project introduces a comprehensive SQL-based Hotel Booking System designed to automate and simplify the management of hotel operations. The system manages guest details, room bookings, and payments using an organized relational database structure. It employs advanced SQL features—such as stored procedures and triggers—to ensure data consistency, enhance security, and streamline the booking process.

By automating routine tasks, the system minimizes manual errors and accelerates operations, making it easier for both hotel staff and guests to manage bookings. Guests can view real-time room availability and make reservations online, while hotel staff benefit from automated status updates and comprehensive reporting capabilities. The system is designed to be scalable, ensuring that it can handle increasing data loads as the hotel expands.

Overall, the project not only demonstrates the efficient use of SQL to solve practical problems in hotel management but also lays the groundwork for future enhancements like user-friendly interfaces and mobile support. This report details the design, implementation, and functionalities of the system, providing insights into how the integration of advanced SQL programming techniques can revolutionize hotel booking management.

Introduction

The Hotel Booking System is a digital tool that replaces traditional paper records to manage hotel reservations more efficiently. It allows guests to check room availability, view room details, and make bookings online from anywhere at any time. This system benefits both guests and hotel staff by reducing paperwork, minimizing errors, and speeding up the booking process.

The system is built using SQL and relies on a well-organized database that keeps track of guests, room information, bookings, and payments. Every time a booking is made, the system automatically updates the room status, ensuring that no room is double-booked. This real-time update is crucial for maintaining an accurate record of available and occupied rooms.

Moreover, the Hotel Booking System simplifies the management of data by using advanced SQL features such as stored procedures and triggers. These features automate routine tasks like booking creation, payment processing, and room status updates, which help maintain data integrity and reduce the need for manual intervention.

Overall, the system is designed to make hotel management faster, more reliable, and user-friendly. It not only streamlines operations for hotel staff but also enhances the experience for guests by providing them with a seamless booking process. The following sections of the report will detail the system requirements, database design, implementation details, and further functionality that contribute to a robust and efficient hotel booking solution.

System Requirements & Objectives

*Requirements:*

* **Database Management System:** The system requires MySQL or a similar relational database management system (RDBMS) that supports advanced SQL features.
* **Hardware & Software:** A standard computer or server capable of running the database software efficiently. Minimal hardware requirements include a modern processor, sufficient memory, and adequate disk space to store data.
* **Development Environment:** Tools such as SQL editors and integrated development environments (IDEs) for testing and refining SQL queries, stored procedures, and triggers.

*Objectives:*

* **Efficient Data Organization:** Design a robust database schema to store details about guests, rooms, bookings, and payments. This structure should allow quick access and modifications while ensuring data integrity.
* **Automation of Processes:** Automate repetitive tasks like room bookings, payment processing, and room status updates through stored procedures and triggers. This reduces manual work and minimizes errors.
* **Data Integrity & Security:** Implement constraints, foreign keys, and triggers to maintain accurate and consistent data, preventing issues like double-booking and unauthorized modifications.
* **Scalability and Flexibility:** Build the system to handle growth, so that as the hotel expands or the volume of bookings increases, the system can easily adapt to the higher data load and more complex queries.

Database Design & Implementation Details

*Core Tables:*

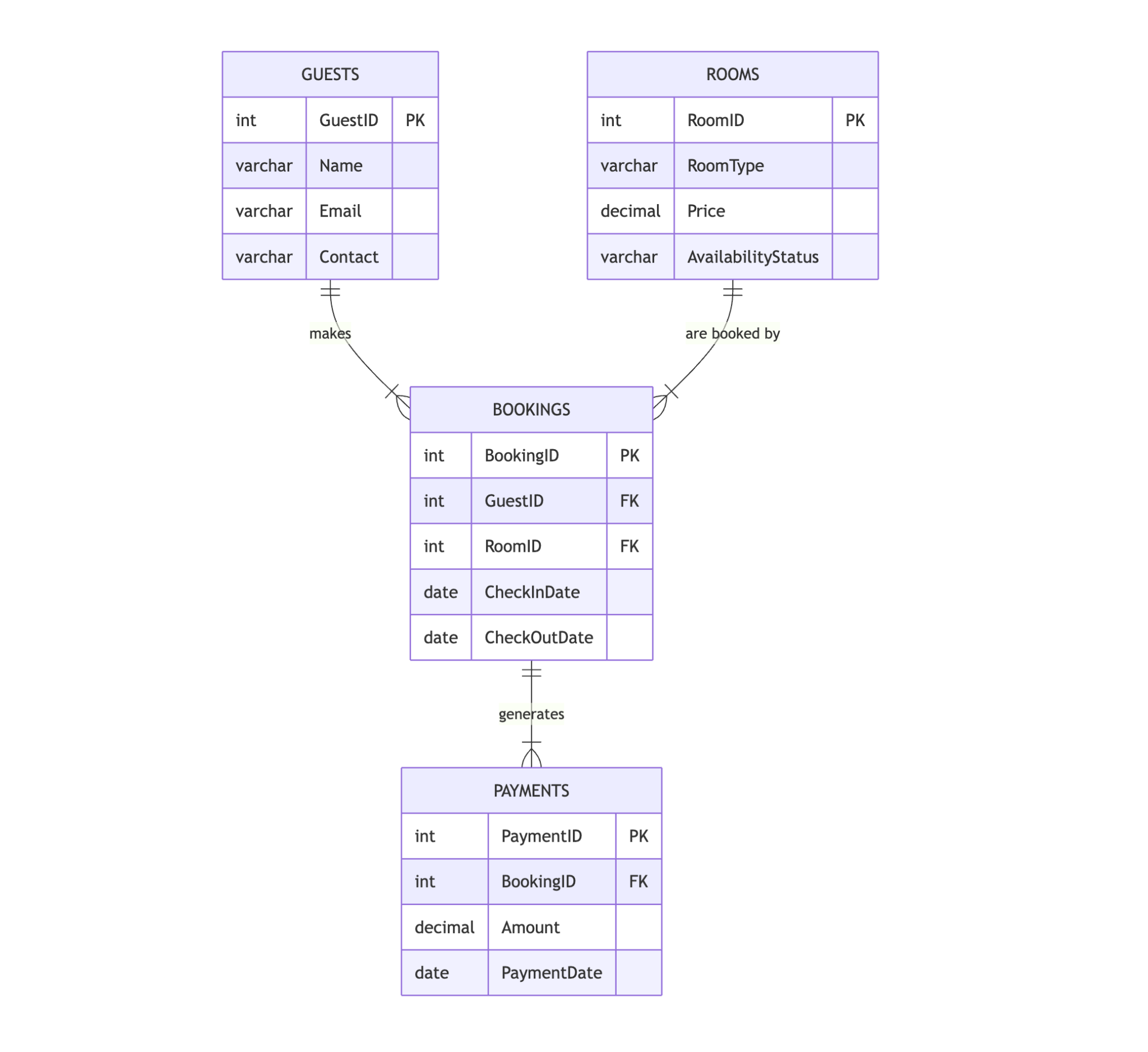
* **Guests:** This table saves guest details, including a unique GuestID, name, contact, and email address. The email field is constrained to ensure uniqueness, which helps in avoiding duplicate records and maintains accurate guest identification.
* **Rooms:** The Rooms table holds important information about each room. It records details such as RoomID, the type of room (e.g., Single, Double, Suite), price, and current availability status. This structure enables quick searches to determine which rooms are available at any given time and their respective rates.
* **Bookings:** Serving as the connection between guests and rooms, the Bookings table captures critical booking details like check-in and check-out dates along with foreign keys that reference the Guests and Rooms tables. This relational design ensures that every booking is linked to a valid guest and room, thereby maintaining data consistency across the database.
* **Payments:** The Payments table is designed to keep track of all transactions related to bookings. It includes a PaymentID, the associated BookingID, the payment amount, and the date of payment. This helps in tracking the financial aspects of each booking, making it easy to generate revenue reports and audit payment history

.

*Implementation Details :-*

* **Table Creation & Data Integrity:  
   Each table is constructed with strict rules (constraints) such as primary keys, unique keys, and foreign keys. These constraints are crucial to prevent errors like duplicate entries and to ensure that relationships between tables (such as between Bookings and Guests or Rooms) are maintained without issues.**
* **Sample Data Insertion:  
   Sample data is preloaded into the tables to simulate real-world usage. This initial dataset allows developers and testers to verify that the database behaves as expected and that all SQL queries return accurate results. It also demonstrates how data flows through the system—from guest registration to room booking and payment processing.**
* **SQL Query Demonstrations:  
   Several SQL queries have been designed to highlight core functionalities of the system:**
  + **Checking Available Rooms: Queries can filter the Rooms table based on the availability status to quickly list all unoccupied rooms.**
  + **Updating Room Status After Booking: Once a booking is made, the system uses SQL scripts (and associated triggers) to automatically update the room's status from available to booked.**
  + **Computing Total Revenue: Aggregation queries on the Payments table provide insights into the total earnings over specific periods, helping in financial analysis and planning.**
* **Automation of Bookings and Payments:  
   The system is further enhanced by automating routine tasks. SQL scripts, along with stored procedures and triggers, manage the insertion of booking and payment records. This automation not only speeds up the process but also reduces the risk of human error, ensuring that every transaction is processed reliably.**

ER DIAGRAM

****

Advanced SQL Features

*Stored Procedures:* Stored procedures are precompiled SQL routines that automate repetitive tasks. For example, the *make\_booking* procedure handles the entire booking process by determining the next booking ID, retrieving the correct room price, inserting the booking record, and recording the corresponding payment. Similarly, the *delete\_Booking* procedure manages the removal of bookings, ensuring that all related records are correctly updated. By encapsulating these processes, stored procedures help reduce manual errors and improve performance by eliminating the need for repetitive SQL parsing.

*Triggers:* Triggers are special routines that automatically execute in response to specific database events. Two main triggers have been implemented:

* **After Insert Trigger:** Once a new booking is added, this trigger automatically updates the corresponding room’s status to "Booked." This ensures that the room is not available for another booking, thereby preventing double-bookings.
* **After Delete Trigger:** When a booking is removed, this trigger resets the room’s status back to "Available." This immediate change keeps the room availability data current without any manual intervention.

*Benefits:* These advanced SQL features provide several key advantages:

* **Automation:** Reduces manual workload and speeds up operations.
* **Consistency:** Ensures data accuracy across related tables by automating status updates.
* **Performance:** Improves efficiency with precompiled procedures, reducing SQL overhead.
* **Real-Time Updates:** Maintains up-to-date information on room availability, which is critical for operational management.
* **Scalability:** Supports the growth of the system by allowing easy integration of new business rules without disrupting existing processes.

System Functionality

The Hotel Management System has been designed to manage essential hotel operations with ease and efficiency. It provides a structured way to handle room availability, guest information, bookings, and payments. Through a combination of SQL queries, procedures, and triggers, the system reduces the need for manual data entry and ensures that all information remains consistent and up to date.

Users can quickly identify which rooms are available, retrieve full details of guest bookings, and track payment histories. The system includes functionality to automatically update room availability after a booking is made or deleted. This is done using SQL triggers, which help maintain real-time accuracy of the data without requiring manual updates.

Stored procedures simplify tasks like making a booking or removing one, as they handle the insertion and deletion of records while also updating payment and room status accordingly. These processes help save time and reduce the possibility of human errors.

In addition to handling operations, the system offers analytical support through SQL queries that summarize useful data, such as total earnings, guest booking frequency, and the most popular room types. These insights can help hotel staff make informed decisions about pricing, room usage, and guest services.

Overall, the system not only handles the day-to-day requirements of hotel management but also improves reliability, ensures data consistency, and supports future scalability for larger hotel operations.

Testing and Future Enhancements

**Testing Process:**

1. **Functional Testing:**
   * Verified core functionalities such as guest registration, room booking, payment entry, and cancellation.
   * Ensured that all operations work correctly through direct SQL queries and stored procedures.
2. **Stored Procedures and Triggers:**
   * Tested the make\_booking and delete\_Booking procedures using various inputs to confirm accurate record insertion and deletion.
   * Triggers were validated to check that room availability updates correctly after each booking or cancellation.
3. **Data Integrity Testing:**
   * Assured that foreign key constraints between tables are working properly to prevent invalid data insertion.
   * Confirmed cascading effects on bookings and payments when guest or room data is removed.
4. **Query Validation:**
   * Verified the accuracy of analytical queries such as total earnings, popular room types, and frequent guests.
   * Cross-checked outputs with expected results to ensure correctness.
5. **Boundary and Edge Case Testing:**
   * Checked how the system handles extreme scenarios like back-to-back bookings, fully occupied hotels, or long stay durations.

**Future Enhancements:**

1. **User Interface Development:**
   * Create a web or desktop application for staff to interact with the system via a simple, user-friendly interface.
2. **Authentication & Security:**
   * Add login systems with role-based access control to prevent unauthorized access and secure sensitive data.
3. **Feedback & Review Module:**
   * Allow guests to provide feedback, enabling the hotel to track satisfaction and improve services.
4. **Inventory & Housekeeping Management:**
   * Introduce modules to track and manage housekeeping schedules and hotel inventory (e.g., toiletries, linens).
5. **Online Booking Integration:**
   * Connect the system with third-party booking platforms to allow real-time room availability and reservations from external websites.

ACKNOWLEDGEMENT

We extend our heartfelt gratitude to our teacher, whose invaluable guidance and unwavering support have been the cornerstone of this project. Their expertise in SQL concepts and database design has not only deepened our understanding of technical subjects but also inspired us to pursue excellence. Their detailed feedback, patience in addressing our queries, and the practical insights they shared have been instrumental in shaping our learning journey and transforming challenges into stepping stones for success.

We are also immensely grateful to our team members, whose collaborative spirit and persistent efforts have been crucial in bringing this project to fruition. Each member contributed unique skills and innovative ideas that enriched our collective work. The camaraderie, mutual respect, and shared dedication to overcoming obstacles have taught us the true meaning of teamwork and reinforced our problem-solving abilities. The synergy within our group has been a testament to the power of collective effort and open communication.

In addition, we sincerely appreciate the wealth of online resources and communities that provided us with a robust foundation of knowledge. These digital platforms offered insightful tutorials, real-world examples, and forums for discussion that significantly broadened our technical perspectives. The accessibility of these resources enabled us to explore advanced concepts and integrate modern practices into our project, thus elevating the overall quality of our work.

Finally, we wish to acknowledge everyone—both directly and indirectly—involved in this journey. Whether through academic advice, technical assistance, or words of encouragement, each contribution has played a vital role in our development. This project not only enhanced our technical skills but also enriched our personal growth, leaving us better prepared for future challenges. We are deeply thankful for the support and inspiration we received throughout this endeavor.

Bibliography

1. **Simran Anand.** (n.d.). *Hotel Room Booking System - MySQL Code.* Retrieved from<https://github.com/SimranAnand1/Hotel-Room-Booking-System/blob/main/MySQL%20Code.docx>
2. **Swathi.** (n.d.). *Hotel Management System.* Retrieved from<https://github.com/Swathi-0/Hotel-Managment-System>
3. **Additional References:**
   * *GeeksforGeeks.* (n.d.). Retrieved from<https://www.geeksforgeeks.org/>
   * DeBarros, A. (2018). *Practical SQL: A Beginner's Guide to Storytelling with Data.*
   * *MySQL Tutorial.* (n.d.). Retrieved from<https://www.mysqltutorial.org/>

Conclusion

The project demonstrates a scalable and efficient Hotel Booking System that not only meets current operational needs but also lays a robust foundation for future expansion. By leveraging SQL automation, the system ensures seamless data management, high operational integrity, and real-time updates, which are crucial for handling dynamic hotel operations. The automation processes significantly reduce manual intervention, thereby minimizing errors and enhancing the overall reliability of the system.

Moreover, the modular architecture of the project allows for easy integration with user-friendly interfaces, making the system accessible to both technical and non-technical users. This integration paves the way for the development of intuitive mobile and web applications that can further streamline booking processes and improve customer experience. The system is also designed with scalability in mind, meaning that additional features and modules can be integrated without disrupting the existing functionality.

Future enhancements could include advanced reporting tools, machine learning algorithms for predicting booking trends, and more sophisticated customer management features. These improvements would not only enrich the user experience but also provide hotel management with deeper insights into operational performance and customer preferences. In summary, this project serves as a comprehensive solution that addresses immediate hotel management needs while also providing a strategic roadmap for continued innovation and system growth.