

HOTEL DATABASE MANAGEMENT SYSTEM

Submitted by

**M.RITHICKA [RA2211003020384]
RUHI FATHIMA [RA2211003020394]
D.LALITHA SREE [RA2211003020409]**

Under the guidance of

Mr. A.VADIVELU

**(Assistant Professor, Department of Computer Science and
Engineering)**

**21CSC205P/DATABASE MANAGEMENT SYSTEM
PROJECT REPORT**

IV SEMESTER/II YEAR

COMPUTER SCIENCE AND ENGINEERING

FACULTY OF ENGINEERING AND TECHNOLOGY



**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
RAMAPURAM, CHENNAI**

MAY 2024

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
(Deemed to be University U/S 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this project report titled “**HOTEL DATABASE MANAGEMENT SYSTEM**” is the bonafide work of **M.RITHICKA [RA2211003020384]**, **RUHI FATHIMA [RA2211003020394]**, **D.LALITHA SREE [RA2211003020409]** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an occasion on this or any other candidate. This project work confirms **21CSC205P/DATABASE MANAGEMENT SYSTEM**, IV Semester, II year, 2024.

SIGNATURE

Mr. A.VADIVELU

Assistant Professor

Computer Science and Engineering,
SRM Institute of Science and Technology,
Ramapuram, Chennai.

SIGNATURE

Dr. K. RAJA, M.E., Ph.D.,

Professor and Head

Computer Science and Engineering,
SRM Institute of Science and Technology,
Ramapuram, Chennai.

Submitted for the project viva-voce held on _____ at SRM Institute of Science and Technology, Ramapuram, Chennai.

**SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
RAMAPURAM, CHENNAI
DECLARATION**

We hereby declare that the entire work contained in this project report titled “**HOTEL DATABASE MANAGEMENT SYSTEM**” has been carried out by **M.RITHICKA [RA2211003020384]**, **RUHI FATHIMA [RA2211003020394]**, **D.LALITHA SREE [RA2211003020409]** at SRM Institute of Science and Technology, Ramapuram, Chennai, under the guidance of **Mr. A.VADIVELU S, Assistant Professor**, Department of Computer Science and Engineering.

Place: Chennai

Date:

M.RITHICKA

RUHI FATHIMA

D.LALITHA SREE

ABSTRACT

Efficient management of hotel operations is paramount in ensuring a seamless guest experience and maximizing profitability within the hospitality industry. This abstract presents a comprehensive database management system tailored specifically to address the multifaceted needs of hotel management. By encompassing various aspects such as reservation handling, guest information management, room allocation, billing, and inventory control, the system provides a holistic solution to streamline administrative tasks and enhance service delivery.

At the core of the proposed database management system lies its ability to facilitate real-time updates of room availability, automate billing processes, and implement robust security measures to safeguard sensitive guest information. These features not only ensure smooth operations but also instill confidence in guests regarding the privacy and security of their personal data.

Furthermore, the system integrates analytical tools capable of generating insightful reports on occupancy rates, revenue trends, and guest preferences. This empowers hoteliers with valuable insights to optimize resource allocation, tailor their offerings to meet customer demands effectively, and make data-driven decisions that drive sustainable growth in the competitive hospitality landscape.

Through the implementation of this database management system, hotels can elevate operational efficiency, improve guest satisfaction, and ultimately achieve their business objectives. By leveraging technology to streamline processes and harnessing data-driven insights, hotels can position themselves for success in a rapidly evolving industry, where meeting and exceeding guest expectations is paramount.

TABLE OF CONTENTS

CHAPTER NUMBER	TITLE	PAGE NUMBER
	Abstract	iii
	List of figures	vi
	List of Abbreviations	vii
1	Introduction	1
	1.1 Introduction	1
	1.1.1 Database Management System	1
	1.1.2 Relational Database Management System	3
	1.1.3 Structured Query Language	4
	1.2 Problem Statement	6
	1.3 Objective	6
	1.4 Scope	6
2	Existing System	7
3	Design	8
	3.1 ER Design	
	3.2 Use Case Diagram	
	3.3 Flow Diagram	
	3.4 Block Diagram	
	3.5 Frontend Design	11
4	Proposed Methodology	12
	4.1 Modules Description	12
	4.2 Database Connectivity	13
5	Implementation	15
	5.1 Implementation of Login	15
	5.2 Implementation of Registration	15
	5.3 Implementation of GUI	17
	5.4 Implementation of Guest services	18
	5.5 Implementation of Booking	20

6	Result and Discussion	22
	6.1 Login Page	22
	6.2 Login success message	23
	6.3 Tables	23
	6.4 Table description	24
	6.5 Room Booking page	24
	6.6 Booking details	25
	6.7 Room booking confirmation message box	25
	6.8 Room booking confirmed	26
	6.9 Table after room booking	27
	6.10 Table after room cancellation	27
	6.11 ID for cancellation	28
	6.12 Cancellation	28
	6.13 Table after room cancellation	29
	6.14 Guest services details	29
	6.15 Guest service confirmation	30
7	Conclusion	31
8	Reference	32
9	Online Course Completion Certificate	33

LIST OF FIGURES

FIGURE NUMBER	FIGURE NAME	PAGE NUMBER
3.1	ER Diagram	8
6.1	Login Page	22
6.2	Login Success message	23
6.3	Tables	23
6.4	Table Description	24
6.5	Room booking page	24
6.6	Booking details	25
6.7	Room booking confirmation Message box	26
6.8	Room booking confirmed	26
6.9	Table after room booking	27
6.10	Table after room cancellation	27
6.11	ID for cancellation	28
6.12	Cancellation	28
6.13	Table after room cancellation	29
6.14	Guest services details	29
6.15	Table after room booking	30

LIST OF ABBREVIATION

HDMS	Hotel Database Management System
DBMS	Database Management System
RDBMS	Relational Database Management System
SQL	Structured Query Language
UI	User Interface
GUI	Graphical User Interface
API	Application Programming Interface
ER	Entity Relationship

CHAPTER 1

INTRODUCTION

1.1 INTRODUCTION:

A Hotel Database Management System (DBMS) is a sophisticated software platform meticulously crafted to streamline and optimize the intricate operations within hospitality establishments. It serves as the nerve center for managing a plethora of hotel-related data, spanning from guest information and reservations to room allocations, inventory management, and financial transactions.

In a Hotel Database Management System (DBMS), a combination of Database Management Systems (DBMS), Relational Database Management Systems (RDBMS), and Structured Query Language (SQL) synergistically converge to create a robust framework for storing, organizing, and retrieving hotel data with unparalleled efficiency and accuracy. Here's a breakdown of how each component seamlessly integrates into the system:

1.1.1. Database Management System (DBMS):

A Database Management System (DBMS) is a software system designed to manage, store, retrieve, and manipulate data in a database. It serves as an interface between the database and the end users or application programs, ensuring efficient and secure access to data. Here are some key aspects to understand about DBMS:

1. **Data Organization:** DBMS organizes data in a structured manner, typically using tables, rows, and columns. This organization allows for easy querying and manipulation of data.
2. **Data Integrity and Security:** DBMS ensures data integrity by enforcing constraints such as uniqueness, referential integrity, and data validation rules. It also provides security features to control access to the database and protect sensitive information.

3. **Data Manipulation:** DBMS equips users with powerful tools and languages, such as SQL (Structured Query Language), to manipulate data seamlessly. Through SQL commands, users can execute operations like insertion, modification, deletion, and querying of data stored within the database.
4. **Concurrency Control:** In multi-user environments, DBMS manages concurrent access to the database by multiple users or applications. It employs techniques such as locking and transaction management to ensure data consistency and integrity.
5. **Query Language:** DBMS provides a query language, such as SQL (Structured Query Language), for interacting with the database. SQL allows users to retrieve, update, insert, and delete data from the database, as well as perform various operations like joining tables and aggregating data.
6. **Scalability and Performance:** A good DBMS is designed to scale with growing data and user demands while maintaining performance. It optimizes query execution through techniques like indexing, query optimization, and caching to minimize response times and resource usage.
7. **Backup and Recovery:** DBMS includes mechanisms for backing up data and restoring it in case of system failures, errors, or data corruption. Regular backups are crucial for data preservation and disaster recovery.
8. **Data Analysis and Reporting:** Many modern DBMS platforms integrate sophisticated tools for data analysis, reporting, and business intelligence. Users can unleash the power of complex queries, generate insightful reports, and craft interactive dashboards to glean actionable insights from the wealth of data housed within the database.

Overall, a well-designed DBMS plays a crucial role in modern information systems by providing efficient, reliable, and secure management of data, which is essential for businesses, organizations, and applications to function effectively in today's data-driven world.

1.1.2. Relational Database Management System (RDBMS):

A Relational Database Management System (RDBMS) is a type of database management system that stores and manages data in a structured format based on the relational model. The relational model organizes data into tables, where each table consists of rows and columns. Here's a closer look at the key components and features of RDBMS:

1. **Tables:** In an RDBMS, data is stored in tables, also known as relations. Each table represents an entity or a concept, and each row in the table represents a unique record or instance of that entity. Columns in the table represent attributes or properties of the entity.
2. **Primary Keys:** RDBMS enforces the concept of primary keys, which uniquely identify each row in a table. Primary keys ensure data integrity and provide a way to establish relationships between tables.
3. **Relationships:** RDBMS allows for the establishment of relationships between tables through the use of foreign keys. A foreign key in one table references the primary key of another table, creating a link between them. This enables data normalization and reduces redundancy in the database.
4. **Structured Query Language (SQL):** RDBMS typically uses SQL (Structured Query Language) as its query language. SQL provides a standardized way to interact with the database, allowing users to retrieve, insert, update, and delete data, as well as perform various operations like joining tables, filtering data, and aggregating results.

5. **Data Integrity:** RDBMS ensures data integrity through constraints such as primary key constraints, foreign key constraints, unique constraints, and check constraints. These constraints help maintain the accuracy and consistency of data within the database.
6. **ACID Properties:** RDBMS guarantees ACID properties for transactions. ACID stands for Atomicity, Consistency, Isolation, and Durability, which are essential characteristics of reliable transaction processing. Transactions in an RDBMS are either fully completed or fully aborted, ensuring data consistency and reliability.
7. **Data Independence:** RDBMS provides a level of abstraction between the physical storage of data and the way users or applications interact with it. This allows for data independence, where changes to the database schema or storage structures do not affect the way data is accessed or manipulated.
8. **Concurrency Control:** RDBMS employs concurrency control mechanisms to manage simultaneous access to the database by multiple users or transactions. Techniques such as locking and timestamp-based protocols ensure data consistency and prevent conflicts between concurrent transactions.
9. **Examples of RDBMS:** There are several popular RDBMS implementations, including MySQL, PostgreSQL, Oracle Database, Microsoft SQL Server, and SQLite. Each RDBMS has its own features, performance characteristics, and suitability for different use cases.

1.1.3. Structured Query Language (SQL):

Structured Query Language (SQL) is a powerful and standardized programming language used for managing and manipulating relational databases. It provides a set of commands for performing various operations on databases, including querying data, updating data, creating and modifying database schemas, and managing user permissions. Here's an overview of SQL and its key features:

1. **Querying Data:** SQL allows users to retrieve data from a database using the **SELECT** statement. Users can specify the columns they want to retrieve and apply conditions to filter the results.
2. **Manipulating Data:** SQL provides commands for adding, updating, and deleting data in a database. The **INSERT**, **UPDATE**, and **DELETE** statements are used for these purposes.
3. **Creating and Modifying Schemas:** SQL allows users to create and modify the structure of a database using the **CREATE**, **ALTER**, and **DROP** statements. These statements are used to create tables, define relationships between tables, add or modify columns, and drop tables or other database objects.
4. **Data Integrity Constraints:** SQL supports the definition of various constraints to maintain data integrity within a database. Common constraints include primary keys, foreign keys, unique constraints, and check constraints. These constraints ensure that the data stored in the database follows specific rules and relationships.
5. **Aggregation Functions:** SQL provides built-in functions for performing aggregate calculations on data, such as **SUM**, **AVG**, **MIN**, **MAX**, and **COUNT**. These functions allow users to analyze and summarize data within a database.

6. Joins: SQL supports various types of joins to combine data from multiple tables based on related columns. Common join types include INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL JOIN. Joins allow users to retrieve data from multiple tables in a single query.

SQL is a fundamental tool for database management and data manipulation. Its versatility, ease of use, and widespread adoption make it an essential skill for database administrators, developers, analysts, and anyone working with relational databases.

1.2 PROBLEM STATEMENT:

The hospitality industry faces operational challenges stemming from manual and disjointed processes in hotel management, leading to inefficiencies in reservation handling, room allocation, billing, and guest services. The absence of a unified Hotel Management System (HMS) contributes to errors, delays, and a lack of real-time insights, hindering the industry's ability to meet the increasing demand for seamless and personalized guest experiences. To address these issues, there is an urgent need for the development and implementation of a comprehensive HMS that streamlines operations, enhances efficiency, and ensures a secure and adaptable platform for hotels of varying scales.

1.3 OBJECTIVE:

The objective of a hotel database management system is to effectively organize, store, retrieve, and manage information crucial to hotel operations, including guest details, reservations, room assignments, billing, inventory, and staff schedules. This system aims to streamline administrative tasks, enhance communication between hotel staff and guests, optimize resource allocation, track hotel performance metrics, and generate insightful reports for informed decision-making and continuous improvement in service delivery and overall guest satisfaction.

1.4 SCOPE:

The Hotel Database Management System (HDBMS) provides a comprehensive solution for hospitality establishments to efficiently manage all aspects of hotel operations. It encompasses guest data management, reservation handling, room inventory tracking, billing and invoicing processes, staff scheduling and management, communication tools for guest interactions, automation of administrative tasks such as check-ins and check-outs, robust reporting and analytics features, as well as stringent security measures to safeguard sensitive information.

CHAPTER 2

EXISTING SYSTEM

The current landscape of Hotel Database Management Systems (HDMS) typically involves a combination of manual processes and fragmented digital solutions. Hotels often rely on paper-based methods or standalone software for managing guest reservations, room assignments, inventory tracking, billing, and staff scheduling. These disjointed systems can lead to inefficiencies, data errors, and challenges in coordinating various aspects of hotel operations. Additionally, legacy systems developed in-house or acquired from vendors may lack integration capabilities, making it difficult to streamline processes and provide a seamless guest experience.

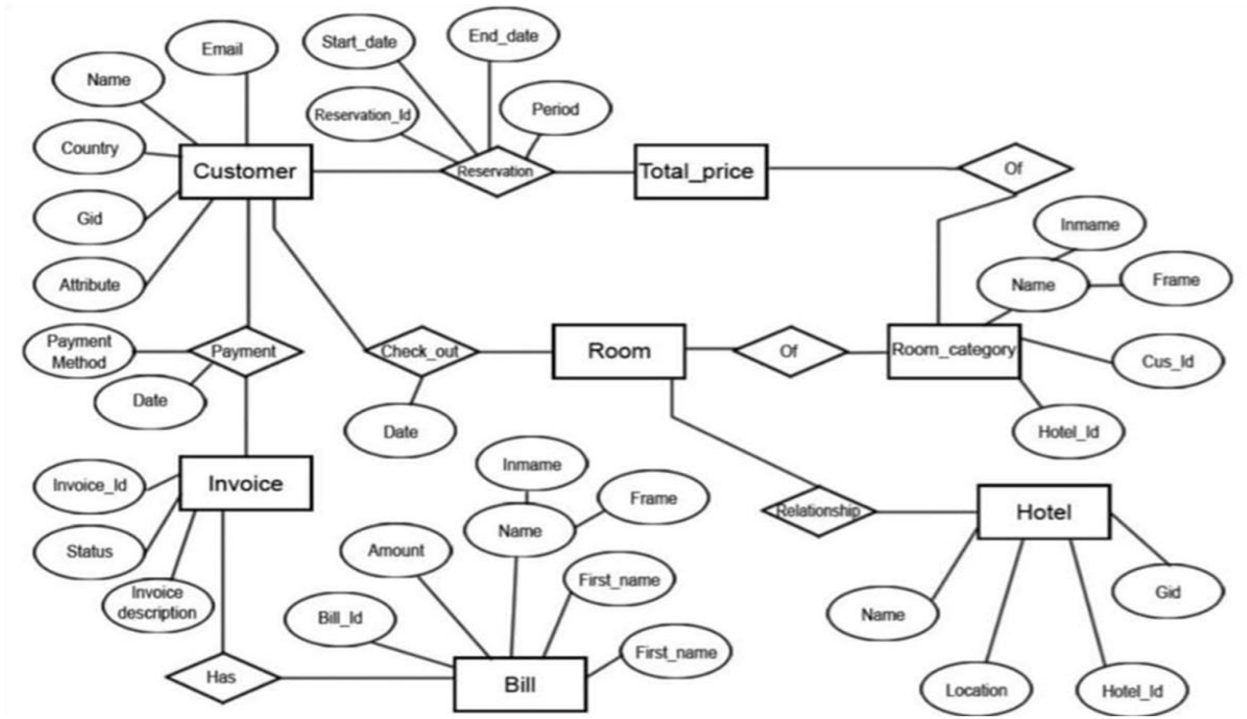
Commercial offerings from established vendors like Oracle Hospitality, Opera PMS, or Sabre Hospitality Solutions dominate the market, offering comprehensive features to address the diverse needs of hotels. These systems often include modules for reservation management, room inventory tracking, billing and invoicing, guest relationship management, and reporting functionalities. However, they can be complex to implement and require substantial upfront investment, making them less accessible to smaller establishments with limited budgets.

Open-source alternatives such as HotelDruid and FrontDesk offer more cost-effective options for hotels seeking customizable solutions. These platforms provide basic functionalities for managing reservations, room assignments, and billing processes while allowing for flexibility in customization and integration with other software tools. Despite their affordability, open-source systems may require additional technical expertise for setup and maintenance, limiting their adoption among hotels without dedicated IT resources.

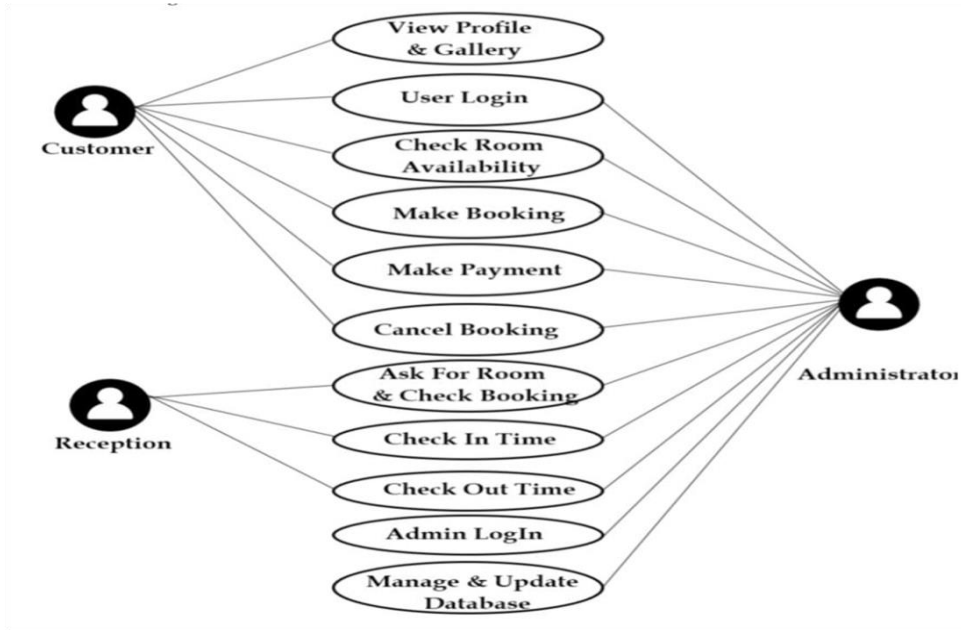
CHAPTER 3

DESIGN:

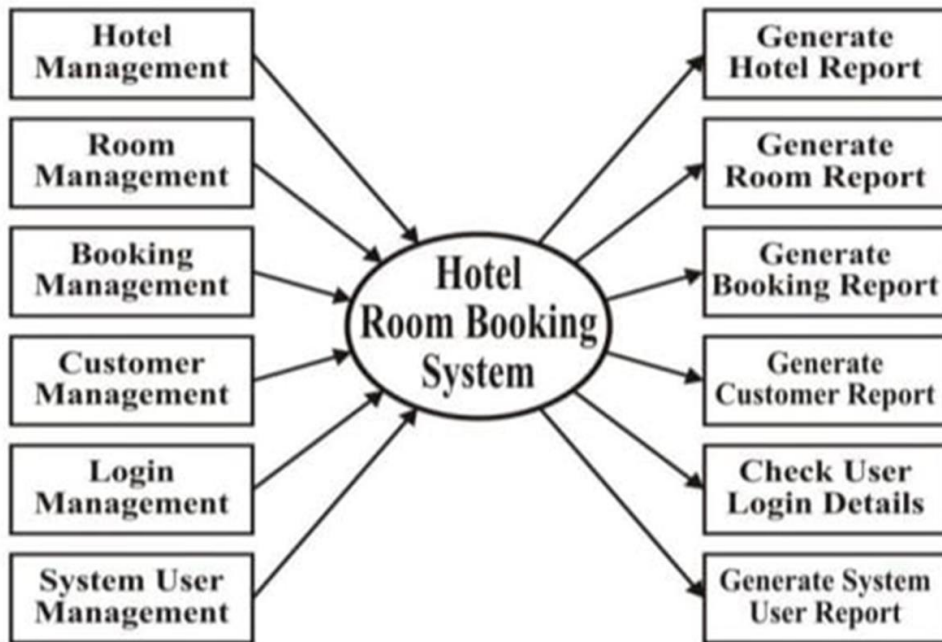
3.1 ER DIAGRAM:



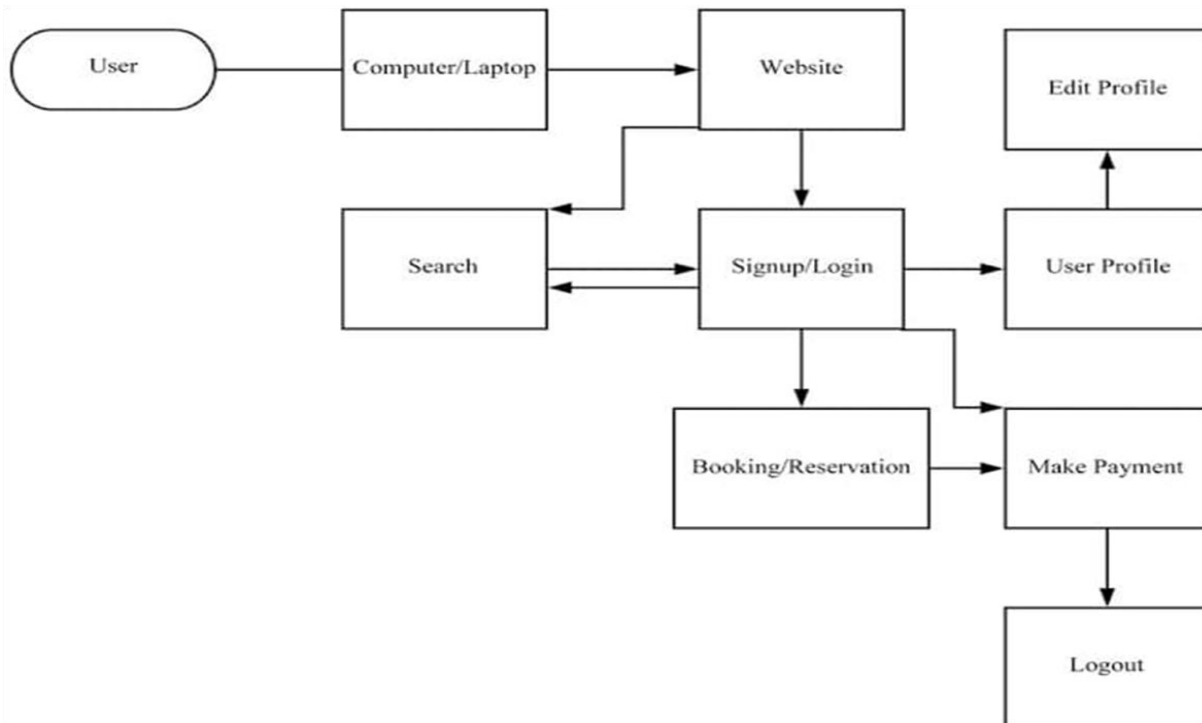
3.2 USE CASE DIAGRAM:



3.3 FLOW DIAGRAM:



3.4 BLOCK DIAGRAM:



3.5 FRONTEND DESIGN:

Designing the frontend for a Hotel Database Management System (HDMS) using Python and MySQL involves creating a user-friendly interface for managing hotel-related data effectively. Here's a breakdown of the key components and features to include:

- **Header:** Include a header section at the top of the interface with the name or logo of the system. You can also include navigation links to different sections such as Home, Rooms, Guests, Reservations, and Transactions.
- **Sidebar/Navigation:** Implement a sidebar or navigation menu for easy access to different sections of the system. This sidebar can contain links to various functionalities like adding new rooms, managing guest information, handling reservations, and viewing transaction history.
- **Room Management:** Design a section where users can view and manage information about hotel rooms. Include fields for room number, type, availability status, price, and any additional details. Provide options for adding new rooms, editing existing ones, and marking rooms as available or occupied.
- **Guest Information:** Create a section for managing guest information. Include fields for guest name, contact details, check-in/out dates, room assigned, and any special requests or preferences. Allow users to add new guest entries, update existing ones, and search for guests by name or booking ID.
- **Reservation Handling:** Design a feature for handling hotel reservations. Include fields for reservation ID, guest name, room type, check-in/out dates, and status (confirmed, pending, cancelled). Provide options for creating new reservations, modifying existing ones, and cancelling reservations if necessary.
- **Responsive Design:** Ensure your design is responsive to different screen sizes, making it accessible on both desktop and mobile devices.

CHAPTER 4

PROPOSED SYSTEM

The proposed hotel management system is designed to revolutionize operations by seamlessly integrating a user-friendly reservation system, efficient front desk services, and a centralized payment module, all aimed at elevating the overall guest experience. Harnessing cutting-edge technologies, this system will offer a robust solution for room management, providing real-time updates on room availability, facilitating swift and accurate bookings, and optimizing housekeeping tasks to enhance operational efficiency significantly.

At the core of the proposed system lies a comprehensive guest profile database, enabling hotels to deliver personalized services and tailored experiences to each guest. By leveraging guest preferences, stay histories, and special requests, the system ensures a seamless and memorable stay for every guest, ultimately fostering customer loyalty and garnering positive reviews. Through intuitive interfaces and seamless integration, this system empowers hotel staff to deliver exceptional service at every touchpoint, from reservation inquiries to check-out procedures, creating lasting impressions that drive guest satisfaction and repeat business.

4.1 MODULES DESCRIPTION:

The module description for a student information management system can include the following components:

Registration, Login:

Seamlessly onboard users and authenticate access with robust registration and login functionalities, ensuring secure and personalized interactions within the system

Administrator:

Empower administrators with comprehensive tools for managing user accounts, permissions, and system configurations, facilitating efficient oversight and control of the hotel management system.

Reservation Module:

Simplify the booking process with an intuitive reservation module, enabling guests to effortlessly secure accommodations while providing hotel staff with real-time availability updates and reservation management capabilities..

Guest Services Module:

Enhance guest satisfaction and streamline service delivery with a dedicated guest services module, offering personalized assistance, concierge services, and seamless communication channels for fulfilling guest requests and preferences.

Room Allocation Module:

Optimize room assignments and occupancy management with a dynamic room allocation module, facilitating efficient distribution of rooms based on guest preferences, availability, and operational considerations.

Billing and Invoicing Module:

Streamline financial transactions and revenue management with a robust billing and invoicing module, automating the generation of accurate invoices, tracking payment status, and ensuring seamless financial operations within the hotel.

4.2 DATABASE CONNECTIVITY:

Database connectivity is fundamental for a Hotel Database Management System (HDMS) as it facilitates interaction between the application and the underlying database where hotel-related data is stored. Here's how database connectivity works in an HDMS:

Database Setup:

Firstly, you need to have a database system installed and configured, such as MySQL, PostgreSQL, or SQLite. You'll also need to create a database schema that defines the structure of your hotel management system, including tables for rooms, guests, reservations, billing, etc.

Database Drivers:

To establish a connection between your HDMS application and the database, you'll need to utilize database drivers specific to the type of database you're using. These drivers provide the necessary functionality to communicate with the database server.

Database Design:

Commence by designing the database schema to store hotel-related information. This may entail tables such as rooms, guests, reservations, billing, etc., depending on the system's requirements. Define the structure of each table, including fields, data types, and relationships between tables using SQL commands.

Integration with Python and MySQL:

Utilize Python for backend logic, including handling user requests, processing data, and interacting with the MySQL database. Establish a connection to a MySQL database employing a Python MySQL connector library such as `pymysql` or `mysql-connector-python`. Utilize SQL queries to retrieve and manipulate data stored in the MySQL database based on user actions and requests.

Establishing Connection:

Upon the startup of the HDMS application or when it necessitates database access, it establishes a connection employing the configured connection settings. This connection enables the application to dispatch SQL queries to the database server and receive results in return.

CHAPTER 5

IMPLEMENTATION

5.1 IMPLEMENTATION OF LOGIN:

```
import tkinter as tk
from tkinter import messagebox,simpledialog
import mysql.connector
import random

class HomePage(tk.Tk):
    def __init__(self):
        super().__init__()

        self.title("Hotel Management System")

        # Login Page
        self.login_label = tk.Label(self, text="Login Page", font=("Arial", 14))
        self.login_label.pack(pady=20)

        self.username_label = tk.Label(self, text="Username:")
        self.username_label.pack()
        self.username_entry = tk.Entry(self)
        self.username_entry.pack()

        self.password_label = tk.Label(self, text="Password:")
        self.password_label.pack()
        self.password_entry = tk.Entry(self, show="*")
        self.password_entry.pack()

        self.login_button = tk.Button(self, text="Login", command=self.login)
        self.login_button.pack(pady=10)

        self.room_booking_button = tk.Button(self, text="Room Booking",
            command=self.reservation_gui)
        self.room_booking_button.pack(pady=20)

        self.hotel_services_button = tk.Button(self, text="Other Hotel Services",
            command=self.book_services)
        self.hotel_services_button.pack()
```


5.2 IMPLEMENTATION OF REGISTRATION:

```
def book_services(self):
    self.guestservices()

def reservation_gui(self):
    connection = mysql.connector.connect(
        host="localhost",
        user="root",
        password="root",
        database="hotelmanagementsystem")
    cursor = connection.cursor()
    root = tk.Tk()
    root.title("Room Reservation")
    def reserve_room():
        guestid = int(guestid_entry.get())
        bookingid = random.randint(10000, 99999)

        room_options = [("DELUXE ROOM", 7000), ("PRESIDENTIAL SUITE", 15000),
("ECONOMY ROOM", 5000)]
        choiceofroom = room_choice_var.get()

        roomname, roomtype = room_options[choiceofroom]

        checkin = checkin_entry.get()
        checkout = checkout_entry.get()

        confirmation = messagebox.askyesno("Confirmation", f"Confirm Room
Reservation?\n\nBooking ID: {bookingid}\nGuest ID: {guestid}\nRoom Name:
{roomname}\nCheck-in Date: {checkin}\nCheck-out Date: {checkout}")

        if confirmation:
            total_cost = roomtype * (int(checkout.split('-')[2]) - int(checkin.split('-')[2])) +
2000 # Adding a fixed service tax of 2000
            paymentid = random.randint(10000, 99999)
            messagebox.showinfo("Payment Information", f"Your room reservation is
confirmed!\n\nPayment ID: {paymentid}\nTotal Cost (incl. service tax): {total_cost}")
            root.destroy() # Close the Tkinter window after confirmation

        insert_query = "INSERT INTO booking
(bookingID,guestID,roomname,checkIndate,checkOutdate) VALUES (%s, %s,%s,%s,%s)"
        cursor.execute(insert_query, (bookingid,guestid,roomname,checkin,checkout))
        connection.commit()
```

5.3 IMPLEMENTATION OF GUI:

```
# GUI elements
guestid_label = tk.Label(root, text="Guest ID:")
guestid_label.grid(row=0, column=0, padx=5, pady=5)
guestid_entry = tk.Entry(root)
guestid_entry.grid(row=0, column=1, padx=5, pady=5)

room_choice_var = tk.IntVar()
def update_room_choice_var(value):
    room_choice_var.set(value)

for i, (room_name, _) in enumerate([("DELUXE ROOM --7000", 7000),
("PRESIDENTIAL SUITE--15000", 15000), ("ECONOMY ROOM--5000", 5000)]):
    room_radio = tk.Radiobutton(root, text=room_name, variable=room_choice_var,
value=i,command=lambda i=i: update_room_choice_var(i))
    room_radio.grid(row=i+1, column=0, padx=5, pady=5)

checkin_label = tk.Label(root, text="Check-in Date (YYYY-MM-DD):")
checkin_label.grid(row=4, column=0, padx=5, pady=5)
checkin_entry = tk.Entry(root)
checkin_entry.grid(row=4, column=1, padx=5, pady=5)

checkout_label = tk.Label(root, text="Check-out Date (YYYY-MM-DD):")
checkout_label.grid(row=5, column=0, padx=5, pady=5)
checkout_entry = tk.Entry(root)
checkout_entry.grid(row=5, column=1, padx=5, pady=5)

reserve_button = tk.Button(root, text="Reserve Room", command=reserve_room)
reserve_button.grid(row=6, column=0, columnspan=2, padx=5, pady=5)

root.mainloop()
```

5.4 IMPLEMENTATION OF GUEST SERVICES:

```
def guestservices(self):
    def confirm_service():
        service_choice = experience_var.get()
        people = int(people_entry.get())
        bookingid=int(bookingid_entry.get())

        service_names = ["Trekking", "Wine Tasting", "Swimming Classes"]
        costs_per_person = [3500, 2000, 1700]

        service_name = service_names[service_choice - 1]
        cost_per_person = costs_per_person[service_choice - 1]

        total_cost = people * cost_per_person + 50 # Assuming 50 is a fixed service charge
        payment_id = random.randint(10000, 99999)

        # Insert service details into payments table
        insert_query = "INSERT INTO payment (PaymentID, Amount,BookingID)
VALUES (%s, %s,%s)"
        cursor.execute(insert_query, (payment_id, total_cost,bookingid))
        connection.commit()

        messagebox.showinfo("Confirmation", f"The total cost of your {service_name} is
{total_cost} for {people} and will be billed to your booking ID {bookingid}")

    # Connect to MySQL database
    connection = mysql.connector.connect(
        host="localhost",
        user="root",
        password="root",
        database="hotelmanagementsystem")
    cursor = connection.cursor()

    # Create Tkinter window
    root = tk.Tk()
    root.title("Guest Services")

    # Service menu labels and entries
    service_label = tk.Label(root, text="Choose a service:")
    service_label.pack()

    experience_var = tk.IntVar()
    for i, name in enumerate(["Trekking", "Wine Tasting", "Swimming Classes"], start=1):
```

```

service_radio = tk.Radiobutton(root, text=name, variable=experience_var, value=i)
service_radio.pack()

people_label = tk.Label(root, text="Number of people:")
people_label.pack()
people_entry = tk.Entry(root)
people_entry.pack()

bookingid_label = tk.Label(root, text="BookingID:")
bookingid_label.pack()
bookingid_entry = tk.Entry(root)
bookingid_entry.pack()

# Confirm button
confirm_button = tk.Button(root, text="Confirm Service", command=confirm_service)
confirm_button.pack()

# Close MySQL connection when the window is closed
def on_closing():
    cursor.close()
    connection.close()
    root.destroy()

root.protocol("WM_DELETE_WINDOW", on_closing)
root.mainloop()

```

5.5 IMPLEMENTATION OF BOOKING:

```
def login(self):
    username = self.username_entry.get()
    password = self.password_entry.get()

    # Add your login logic here
    if username == "admin" and password == "password":
        messagebox.showinfo("Login Successful", "Welcome, Admin!")
        self.adminuser = tk.Label(self, text="ADMIN MENU", font=("Arial", 14))
        self.adminuser.pack(pady=20)
        self.room_cancellation_button = tk.Button(self, text="Room Cancellation",
command=self.room_cancellation)
        self.room_cancellation_button.pack(pady=20)

    else:
        messagebox.showerror("Login Failed", "Invalid username or password")

def open_room_booking(self):
    messagebox.showinfo("Room Booking", "Redirecting to Room Booking Page")
def open_room_cancellation(self):
    messagebox.showinfo("ROOM CANCELLATION", "Redirecting to room cancellation
page")

def room_cancellation(self):
    connection = mysql.connector.connect(
        host="localhost",
        user="root",
        password="root",
        database="hotelmanagementsystem")
    cursor = connection.cursor()
    booking_id = simpledialog.askinteger("Enter Booking ID", "Enter the Booking ID to
cancel:")
    cursor.execute("SELECT * FROM booking WHERE bookingID = %s", (booking_id,))
    booking = cursor.fetchone()
    if booking:
        cursor.execute("DELETE FROM booking WHERE bookingID = %s", (booking_id,))
        connection.commit()
        messagebox.showinfo("Cancellation Successful", f"Booking with ID {booking_id}
has been cancelled.")
    else:
        messagebox.showinfo("Cancellation Failed", "Booking does not exist.")
```

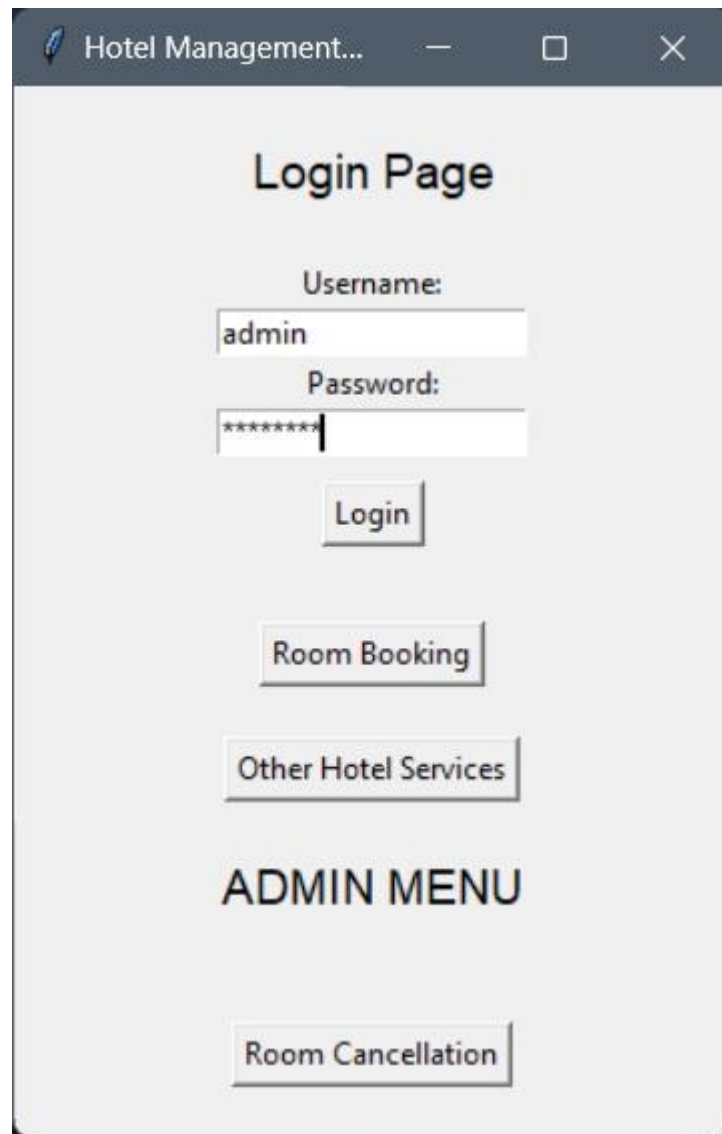
```
def open_hotel_services(self):
    messagebox.showinfo("Other Hotel Services", "Redirecting to Other Hotel Services
Page")

if __name__ == "__main__":
    home_page = HomePage()
    home_page.mainloop()
```

CHAPTER 6

RESULT AND DISCUSSION

6.1 LOGIN PAGE:



The screenshot shows a web application window titled "Hotel Management...". The main heading is "Login Page". Below the heading, there are two input fields: "Username:" with the value "admin" and "Password:" with masked characters "*****". A "Login" button is positioned below the password field. Further down, there are two more buttons: "Room Booking" and "Other Hotel Services". At the bottom of the page, the heading "ADMIN MENU" is displayed, followed by a "Room Cancellation" button.

Hotel Management...

Login Page

Username:
admin

Password:

Login

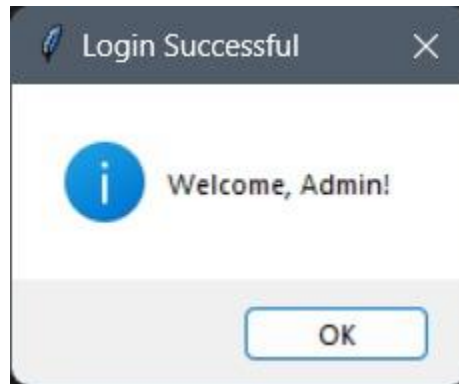
Room Booking

Other Hotel Services

ADMIN MENU

Room Cancellation

6.2 LOGIN SUCCESS:



6.3 TABLES:

```
MySQL 8.0 Command Line Cli  X  +  -  
  
mysql> describe booking;  
+-----+-----+-----+-----+-----+-----+  
| Field      | Type      | Null | Key | Default | Extra      |  
+-----+-----+-----+-----+-----+-----+  
| bookingID  | int       | NO   | PRI | NULL    | auto_increment |  
| guestID    | int       | YES  |     | NULL    |              |  
| roomname   | varchar(255) | YES  |     | NULL    |              |  
| CheckInDate | date      | YES  |     | NULL    |              |  
| CheckOutDate | date      | YES  |     | NULL    |              |  
+-----+-----+-----+-----+-----+-----+  
5 rows in set (0.04 sec)  
  
mysql> describe guests;  
+-----+-----+-----+-----+-----+-----+  
| Field      | Type      | Null | Key | Default | Extra      |  
+-----+-----+-----+-----+-----+-----+  
| GuestID    | int       | NO   | PRI | NULL    | auto_increment |  
| FirstName  | varchar(50) | NO   |     | NULL    |              |  
| LastName   | varchar(50) | NO   |     | NULL    |              |  
| Email      | varchar(100) | YES  |     | NULL    |              |  
| Phone      | varchar(20) | YES  |     | NULL    |              |  
| Address    | varchar(255) | YES  |     | NULL    |              |  
+-----+-----+-----+-----+-----+-----+  
6 rows in set (0.01 sec)  
  
mysql> describe payment;  
+-----+-----+-----+-----+-----+-----+  
| Field      | Type      | Null | Key | Default | Extra      |  
+-----+-----+-----+-----+-----+-----+  
| PaymentID  | varchar(10) | YES  |     | NULL    |              |  
| Amount     | int       | YES  |     | NULL    |              |  
| BookingID  | varchar(10) | YES  |     | NULL    |              |  
+-----+-----+-----+-----+-----+-----+  
3 rows in set (0.00 sec)
```


6.4 TABLE DESCRIPTION:

```
MySQL 8.0 Command Line Cli x + v

mysql> describe booking;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| bookingID | int | NO | PRI | NULL | auto_increment |
| guestID | int | YES | | NULL | |
| roomname | varchar(255) | YES | | NULL | |
| CheckInDate | date | YES | | NULL | |
| CheckOutDate | date | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.04 sec)

mysql> describe guests;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| GuestID | int | NO | PRI | NULL | auto_increment |
| FirstName | varchar(50) | NO | | NULL | |
| LastName | varchar(50) | NO | | NULL | |
| Email | varchar(100) | YES | | NULL | |
| Phone | varchar(20) | YES | | NULL | |
| Address | varchar(255) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
6 rows in set (0.01 sec)

mysql> describe payment;
+-----+-----+-----+-----+-----+-----+
| Field | Type | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| PaymentID | varchar(10) | YES | | NULL | |
| Amount | int | YES | | NULL | |
| BookingID | varchar(10) | YES | | NULL | |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

6.5 ROOM BOOKING PAGE:

Room Reservation

Guest ID:

☐

 DELUXE ROOM --7000

☒ PRESIDENTIAL SUITE--15000

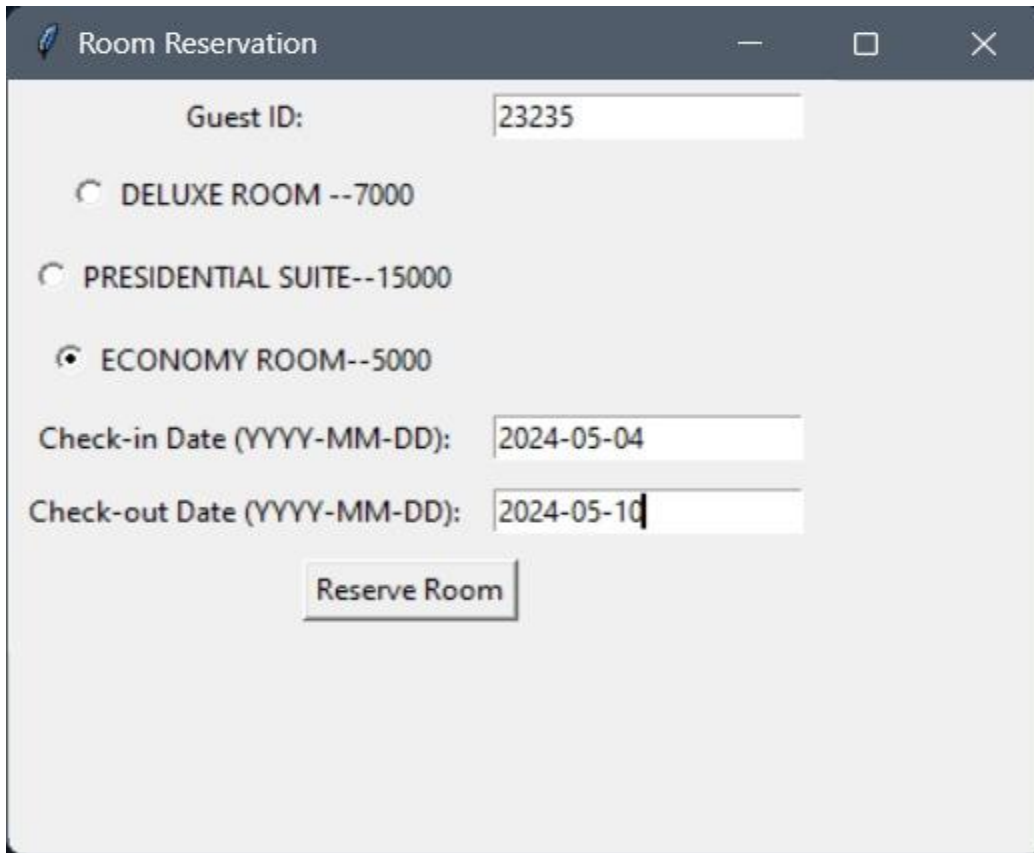
☒ ECONOMY ROOM--5000

Check-in Date (YYYY-MM-DD):

Check-out Date (YYYY-MM-DD):

Reserve Room

6.6 BOOKING DETAILS:



A screenshot of a 'Room Reservation' window. It features a dark header bar with a feather icon and the title 'Room Reservation'. The form contains a 'Guest ID' field with the value '23235'. Below this are three radio button options: 'DELUXE ROOM --7000', 'PRESIDENTIAL SUITE--15000', and 'ECONOMY ROOM--5000', with the last one selected. There are two date fields: 'Check-in Date (YYYY-MM-DD):' with the value '2024-05-04' and 'Check-out Date (YYYY-MM-DD):' with the value '2024-05-10'. At the bottom is a 'Reserve Room' button.

Room Reservation

Guest ID: 23235

☐ DELUXE ROOM --7000

☐ PRESIDENTIAL SUITE--15000

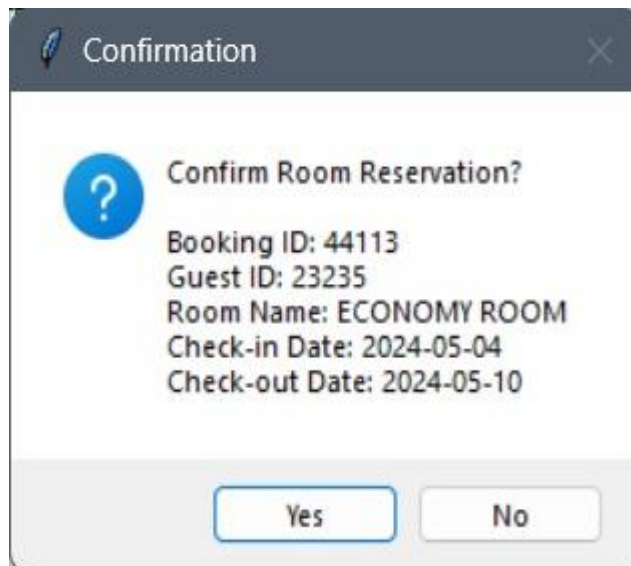
☒ ECONOMY ROOM--5000

Check-in Date (YYYY-MM-DD): 2024-05-04

Check-out Date (YYYY-MM-DD): 2024-05-10

Reserve Room

6.7 ROOM BOOKING CONFIRMATION MESSAGE BOX:



A screenshot of a 'Confirmation' message box. It has a dark header bar with a feather icon and the title 'Confirmation'. The main content area features a blue circular icon with a white question mark. To the right of the icon, the text reads: 'Confirm Room Reservation?', 'Booking ID: 44113', 'Guest ID: 23235', 'Room Name: ECONOMY ROOM', 'Check-in Date: 2024-05-04', and 'Check-out Date: 2024-05-10'. At the bottom are two buttons: 'Yes' and 'No'.

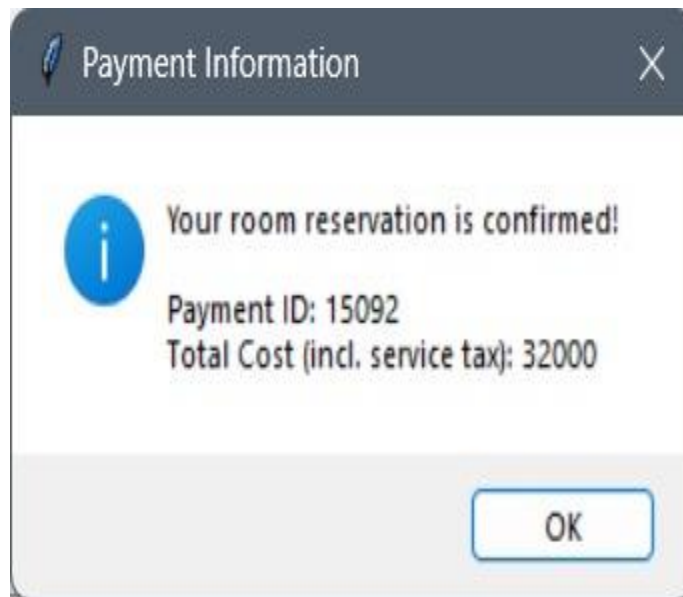
Confirmation

Confirm Room Reservation?

Booking ID: 44113
Guest ID: 23235
Room Name: ECONOMY ROOM
Check-in Date: 2024-05-04
Check-out Date: 2024-05-10

Yes No

6.8 ROOM BOOKING CONFIRMED:



6.9 TABLE AFTER ROOM BOOKING:

```
mysql> select * from booking;
```

bookingID	guestID	roomname	CheckInDate	CheckOutDate
16206	1552	PRESIDENTIAL SUITE	2020-05-16	2020-05-21
24378	1225	DELUXE ROOM	2020-04-15	2020-04-19
27436	15155	PRESIDENTIAL SUITE	2020-08-18	2020-08-20
37697	1596	DELUXE ROOM	2020-08-18	2020-08-22
38855	5525	DELUXE ROOM	2020-05-18	2020-05-22
42869	4521	PRESIDENTIAL SUITE	2024-12-20	2024-12-25
44113	23235	ECONOMY ROOM	2024-05-04	2024-05-10
52874	1234	DELUXE ROOM	2020-05-15	2020-05-20
98971	78964	DELUXE ROOM	2024-05-19	2024-05-21

```
9 rows in set (0.01 sec)
```

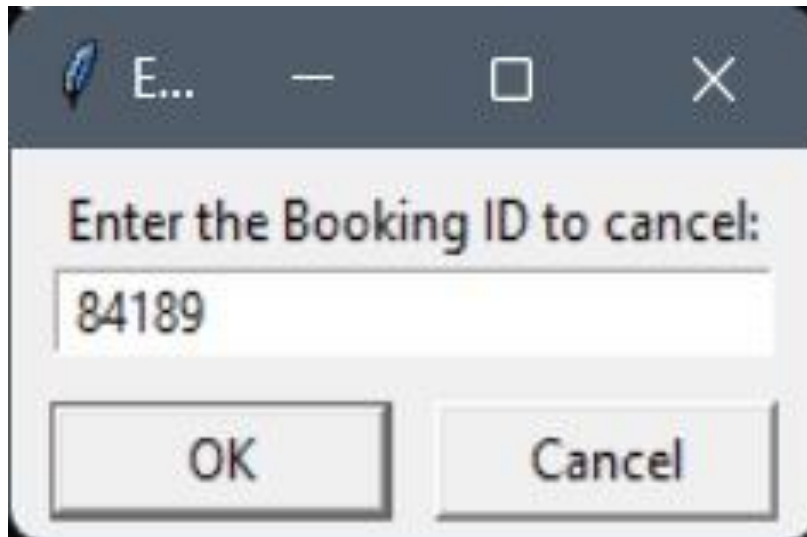
6.10 TABLE BEFORE ROOM CANCELLATION:

```
mysql> select * from booking;
```

bookingID	guestID	roomname	CheckInDate	CheckOutDate
16206	1552	PRESIDENTIAL SUITE	2020-05-16	2020-05-21
24378	1225	DELUXE ROOM	2020-04-15	2020-04-19
27436	15155	PRESIDENTIAL SUITE	2020-08-18	2020-08-20
37697	1596	DELUXE ROOM	2020-08-18	2020-08-22
38855	5525	DELUXE ROOM	2020-05-18	2020-05-22
42869	4521	PRESIDENTIAL SUITE	2024-12-20	2024-12-25
52874	1234	DELUXE ROOM	2020-05-15	2020-05-20
84189	148996	0	2020-08-18	2020-08-19
98971	78964	DELUXE ROOM	2024-05-19	2024-05-21

9 rows in set (0.01 sec)

6.11 ID FOR CANCELLATION:

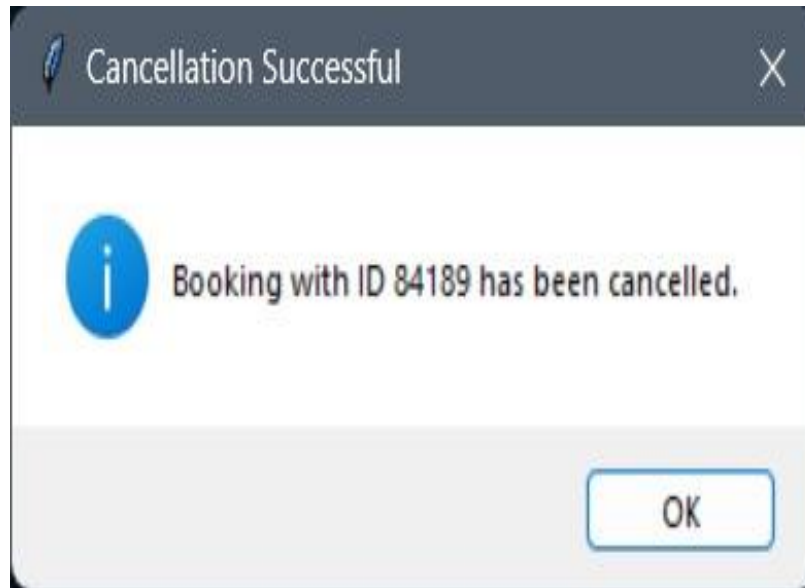


Enter the Booking ID to cancel:

84189

OK Cancel

6.12 ID FOR CANCELLATION:



6.13 TABLE AFTER CANCELLATION:

```
mysql> select * from booking;
```

bookingID	guestID	roomname	CheckInDate	CheckOutDate
16206	1552	PRESIDENTIAL SUITE	2020-05-16	2020-05-21
24378	1225	DELUXE ROOM	2020-04-15	2020-04-19
27436	15155	PRESIDENTIAL SUITE	2020-08-18	2020-08-20
37697	1596	DELUXE ROOM	2020-08-18	2020-08-22
38855	5525	DELUXE ROOM	2020-05-18	2020-05-22
42869	4521	PRESIDENTIAL SUITE	2024-12-20	2024-12-25
52874	1234	DELUXE ROOM	2020-05-15	2020-05-20
98971	78964	DELUXE ROOM	2024-05-19	2024-05-21

8 rows in set (0.00 sec)

6.14 GUEST SERVICES DETAILS:



A dialog box titled "Guest..." with standard window controls. It contains a "Choose a service:" section with three radio button options: "Trekking", "Wine Tasting", and "Swimming Classes" (which is selected). Below this is a "Number of people:" label followed by a text input field containing the number "2". Underneath is a "BookingID:" label followed by a text input field containing "23235". At the bottom is a "Confirm Service" button.

6.1 GUEST SERVICES CONFIRMATION DETAILS:



A confirmation dialog box titled "Confirmation" with a close button. It features an information icon (a blue circle with a white 'i') on the left. To the right of the icon, the text reads: "The total cost of your Swimming Classes is 3450 for 2 and will be billed to your booking ID 23235". At the bottom right, there is an "OK" button.

CHAPTER 7

CONCLUSION

In conclusion, the Hotel Database Management System (HDMS) represents a pivotal evolution in the hospitality industry, offering a comprehensive solution to streamline operations and enhance guest experiences. By seamlessly integrating reservation systems, front desk services, and payment modules, HDMS not only improves operational efficiency but also fosters greater guest satisfaction and loyalty. Leveraging advanced technologies, such as real-time room management and personalized guest profiles, HDMS empowers hoteliers to deliver exceptional service and tailor experiences to individual preferences.

Moreover, HDMS provides hotel administrators with invaluable tools for managing user accounts, permissions, and system configurations, enabling efficient oversight and control of the entire hotel management process. Through intuitive interfaces and seamless integration, administrators can optimize resource allocation, monitor performance metrics, and make informed decisions to drive business growth and profitability. Additionally, the system's robust billing and invoicing module ensures seamless financial transactions, enhancing transparency and accountability in revenue management.

As the hospitality landscape continues to evolve, HDMS remains at the forefront of innovation, continually adapting to meet the changing needs and expectations of both guests and hoteliers. By embracing technological advancements and prioritizing guest-centric approaches, HDMS enables hotels to stay competitive in a dynamic and increasingly digital marketplace, ultimately paving the way for continued success and excellence in hospitality management.

CHAPTER 8

REFERENCE

- “HOTEL MANAGEMENT SYSTEM”-International Research Journal of Modernization in Engineering Technology and Science Volume:02/Issue:03/March-2020 Mounika Nandiraju, Salluri Rachana, Shaik Chandini, Sandhu Srilatha, G.Sabitha, Seema Nazneen
- “Web Based Hotel Management System”- International Journal of Research Publication and Reviews Volume 3, November 2022 Dukare Siddhesh Sudam, Bhalerao Akanksha Santosh.
- “Hotel Booking Management System”- International Journal of Creative Research Thoughts Volume 10, Issue 6 June 2022 Azgar Pattan, Boggula Niranjana, Shaik Jakeer.
- “A STUDY ON HOTEL MANAGEMENT SYSTEM”-International Journal of Emerging Technologies and Innovative Research May 2023, Volume 10, Issue 5 Sahil, Deepak Kumar, Surender, Dr. Praveen Kumar.

ONLINE COURSE COMPLETION CERTIFICATE



||||| COURSE COMPLETION CERTIFICATE |||||

The certificate is awarded to

RITHICKA M (RA2211003020384)

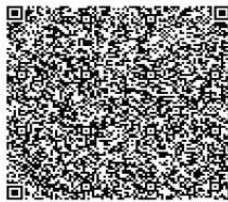
for successfully completing the course

Database and SQL

on March 28, 2024



Congratulations! You make us proud!



Issued on: Saturday, March 30, 2024
To verify, scan the QR code at <https://verify.onwingspan.com>


Thirumala Arohi
Senior Vice President and Head
Education, Training and Assessment (ETA)
Infosys Limited



||||| COURSE COMPLETION CERTIFICATE |||||

The certificate is awarded to

Ruhi Fathima

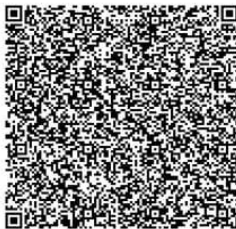
for successfully completing the course

Database and SQL


on March 28, 2024



Congratulations! You make us proud!



Issued on: Thursday, March 28, 2024
To verify, scan the QR code at <https://verify.onwingspan.com>


Thirumala Arohi
Senior Vice President and Head
Education, Training and Assessment (ETA)
Infosys Limited



||| COURSE COMPLETION CERTIFICATE |||

The certificate is awarded to

D Lalitha Sree

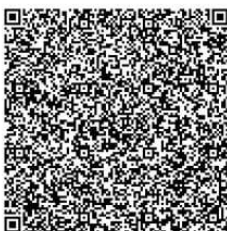
for successfully completing the course

Database and SQL

on March 28, 2024

Infosys | Springboard

Congratulations! You make us proud!



Issued on: Friday, March 29, 2024
To verify, scan the QR code at <https://verify.onwingspan.com>

Thirumala Arohi
Senior Vice President and Head
Education, Training and Assessment (ETA)
Infosys Limited