MINI PROJECT
HOTEL BOOKING MANAGEMENT

AIM:

To develop a hotel booking management system using Python Streamlit and MySQL that provides an efficient, user-friendly, and dynamic platform for customers to book hotel rooms, and for hotel administrators to manage bookings, availability, and customer details effectively. The system will streamline the booking process, ensure data integrity, and offer real-time updates to enhance user experience and operational efficiency.

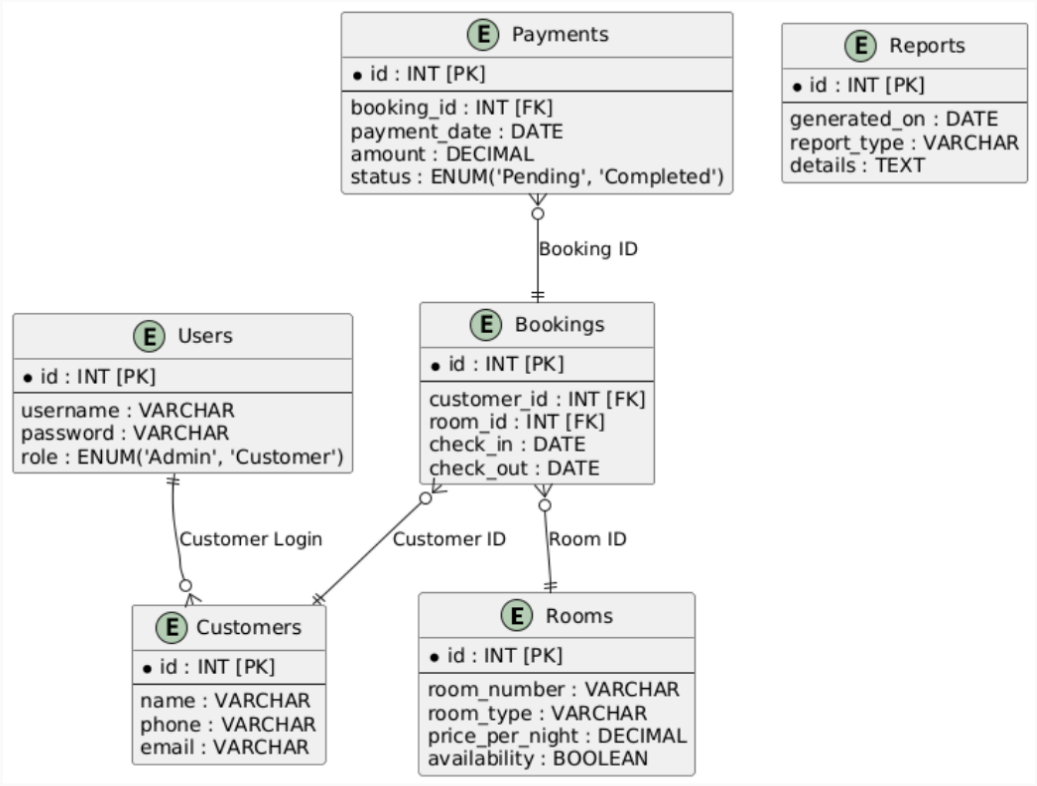
ABSTRACT:

This project focuses on creating a robust and efficient hotel booking management system using Python Streamlit as the front-end framework and MySQL as the back-end database. The system aims to simplify and streamline the process of hotel room bookings by providing an intuitive user interface for customers and a comprehensive management dashboard for administrators.

The application allows users to browse available rooms, check prices, and make bookings seamlessly. Administrators can manage room availability, view customer details, and track reservations efficiently. The integration of Streamlit ensures a responsive and interactive user interface, while MySQL guarantees reliable data storage and retrieval.

Key features include real-time room availability updates, secure user authentication, dynamic pricing options, and a reporting module for analytics. The project prioritizes ease of use, data integrity, and scalability, making it a valuable tool for hotel businesses looking to enhance their booking process

ER-DIAGRAM:


Payments
• id : INT [PK]
booking_id : INT [FK]
payment_date : DATE
amount : DECIMAL
status : ENUM('Pending', 'Completed')
Reports
• id : INT [PK]
generated_on : DATE
report_type : VARCHAR
details : TEXT
Booking ID
Users
• id : INT [PK]
username : VARCHAR
password : VARCHAR
role : ENUM('Admin', 'Customer')
Bookings
• id : INT [PK]
customer_id : INT [FK]
room_id : INT [FK]
check_in : DATE
check_out : DATE
Customer Login
Customer ID
Room ID
Customers
• id : INT [PK]
name : VARCHAR
phone : VARCHAR
email : VARCHAR
Rooms
• id : INT [PK]
room_number : VARCHAR
room_type : VARCHAR
price_per_night : DECIMAL
availability : BOOLEAN

PROGRAM :

For MySql Initialization:

```
CREATE DATABASE hotel_booking;
USE hotel_booking;

CREATE TABLE rooms (
    id INT AUTO_INCREMENT PRIMARY KEY,
    room_number VARCHAR(10),
    room_type VARCHAR(50),
    price_per_night DECIMAL(10, 2),
    availability BOOLEAN DEFAULT TRUE
);

CREATE TABLE bookings (
    id INT AUTO_INCREMENT PRIMARY KEY,
    customer_name VARCHAR(100),
    room_id INT,
    check_in DATE,
    check_out DATE,
    FOREIGN KEY (room_id) REFERENCES rooms(id)
);

INSERT INTO rooms (room_number, room_type, price_per_night, availability)
VALUES
('101', 'Single', 50.00, TRUE),
('102', 'Double', 80.00, TRUE),
('201', 'Suite', 150.00, TRUE);
```

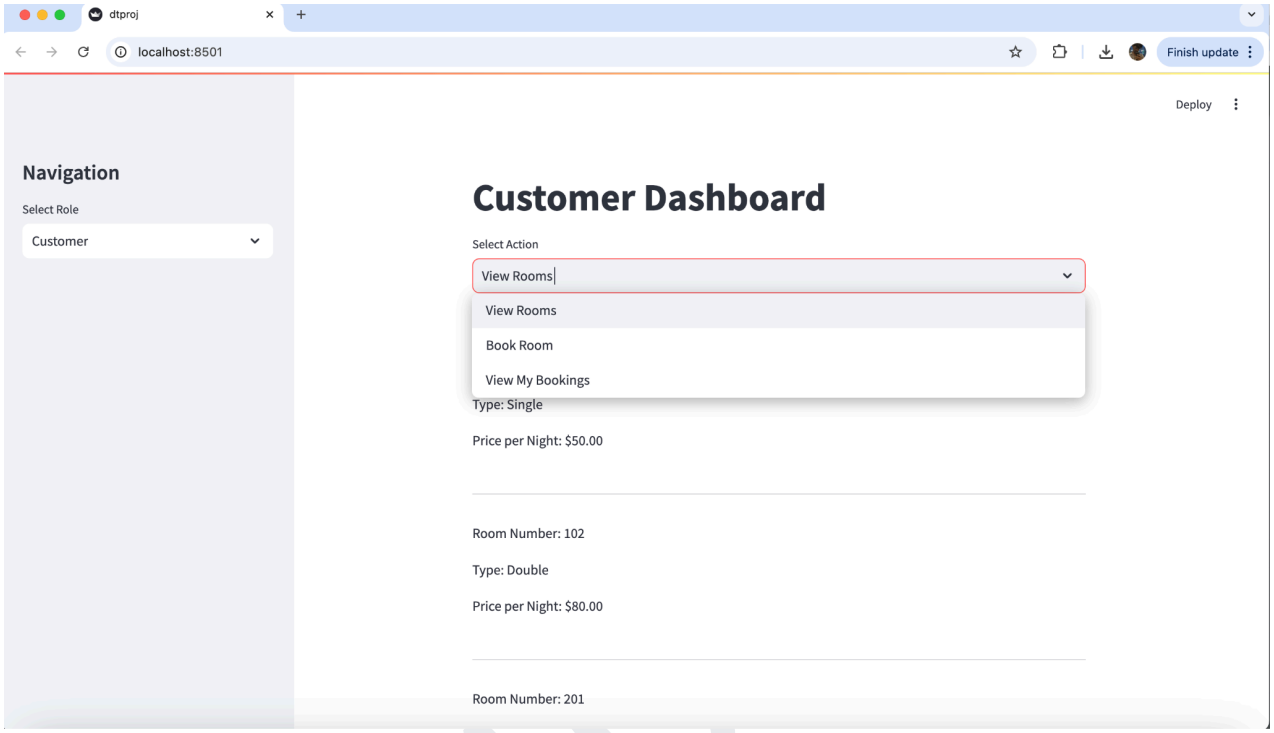
For Python with Streamlit :

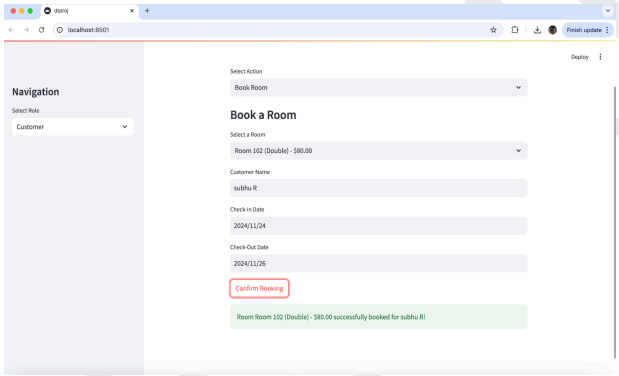
```
import streamlit as st
import mysql.connector
from datetime import date

# Database Connection
def get_connection():
    return mysql.connector.connect(
        host="localhost",
        user="your_username",
        password="your_password",
        database="hotel_booking"
    )

# Authenticate User
def authenticate_user(username, password):
    conn = get_connection()
    cursor = conn.cursor(dictionary=True)
    query = "SELECT * FROM users WHERE username = %s AND password = %s"
    cursor.execute(query, (username, password))
    user = cursor.fetchone()
    cursor.close()
    conn.close()
    return user

# Add Room (Admin)
def add_room():
    st.subheader("Add New Room")
    room_number = st.text_input("Room Number")
    room_type = st.selectbox("Room Type", ["Single", "Double", "Suite"])
    price = st.number_input("Price per Night", min_value=0.0, step=0.01)
    if st.button("Add Room"):
        try:
            conn = get_connection()
            cursor = conn.cursor()
            query = "INSERT INTO rooms (room_number, room_type, price_per_night, availability) VALUES (%s, %s, %s, TRUE)"
            cursor.execute(query, (room_number, room_type, price))
            conn.commit()
            st.success("Room added successfully!")
            cursor.close()
            conn.close()
        except Exception as e:
            st.error(f"Error: {e}")

# View Rooms (Customer)
def view_rooms():
    st.subheader("Available Rooms")
    try:
        conn = get_connection()
        cursor = conn.cursor(dictionary=True)
        query = "SELECT * FROM rooms WHERE availability = TRUE"
        cursor.execute(query)
```

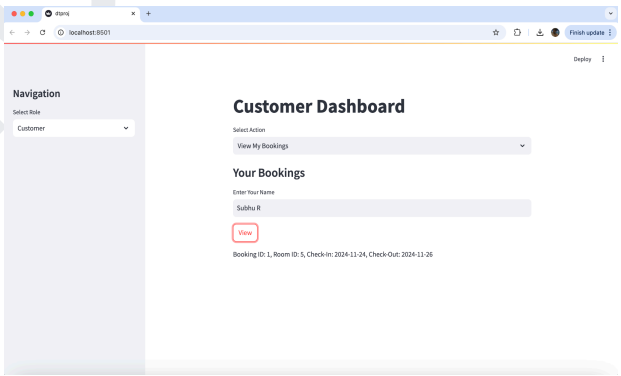
```
        rooms = cursor.fetchall()
        if rooms:
            for room in rooms:
                st.write(f"Room Number: {room['room_number']}")
                st.write(f"Type: {room['room_type']}")
                st.write(f"Price per Night: ${room['price_per_night']}")
                st.write("---")
        else:
            st.write("No rooms are currently available.")
        cursor.close()
        conn.close()
    except Exception as e:
        st.error(f"Error: {e}")

# Add Booking (Customer)
def add_booking():
    st.subheader("Book a Room")
    try:
        conn = get_connection()
        cursor = conn.cursor(dictionary=True)
        query = "SELECT * FROM rooms WHERE availability = TRUE"
        cursor.execute(query)
        rooms = cursor.fetchall()

        if rooms:
            room_options = {f"Room {room['room_number']} ({room['room_type']}) - ${room['price_per_night']}": room['id'] for
room in rooms}
            selected_room = st.selectbox("Select a Room", list(room_options.keys()))
            customer_name = st.text_input("Customer Name")
            check_in = st.date_input("Check-In Date", min_value=date.today())
            check_out = st.date_input("Check-Out Date", min_value=check_in)

            if st.button("Confirm Booking"):
                room_id = room_options[selected_room]
                query = "INSERT INTO bookings (customer_name, room_id, check_in, check_out) VALUES (%s, %s, %s, %s)"
                cursor.execute(query, (customer_name, room_id, check_in, check_out))
                conn.commit()

                cursor.execute("UPDATE rooms SET availability = FALSE WHERE id = %s", (room_id,))
                conn.commit()

                st.success(f"Room {selected_room} successfully booked for {customer_name}!")
        else:
            st.write("No rooms available.")

        cursor.close()
        conn.close()
    except Exception as e:
        st.error(f"Error: {e}")

# View Customer Bookings
def view_customer_bookings():
    st.subheader("Your Bookings")
    customer_name = st.text_input("Enter Your Name")
```

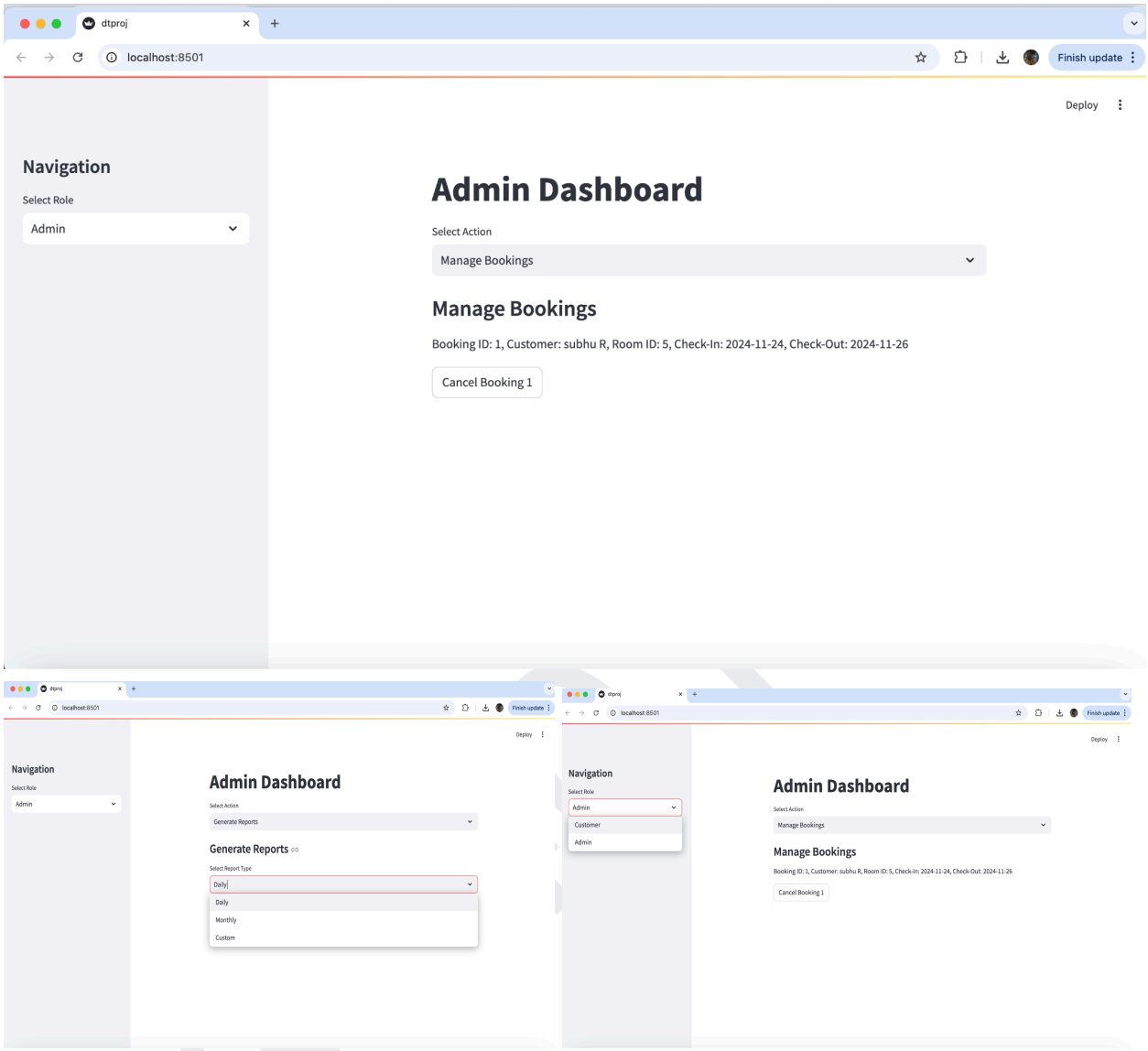
```
    if st.button("View"):
        try:
            conn = get_connection()
            cursor = conn.cursor(dictionary=True)
            query = "SELECT * FROM bookings WHERE customer_name = %s"
            cursor.execute(query, (customer_name,))
            bookings = cursor.fetchall()
            for booking in bookings:
                st.write(f"Booking ID: {booking['id']}, Room ID: {booking['room_id']}, Check-In: {booking['check_in']}, Check-Out:
{booking['check_out']}")
            cursor.close()
            conn.close()
        except Exception as e:
            st.error(f"Error: {e}")

# Admin Dashboard
def admin_dashboard():
    st.title("Admin Dashboard")
    menu = ["Add Room", "Manage Rooms"]
    choice = st.selectbox("Select Action", menu)
    if choice == "Add Room":
        add_room()

# Customer Dashboard
def customer_dashboard():
    st.title("Customer Dashboard")
    menu = ["View Rooms", "Book Room", "View My Bookings"]
    choice = st.selectbox("Select Action", menu)
    if choice == "View Rooms":
        view_rooms()
    elif choice == "Book Room":
        add_booking()
    elif choice == "View My Bookings":
        view_customer_bookings()

# Main Function
def main():
    st.sidebar.title("Hotel Booking System")
    role = st.sidebar.selectbox("Login As", ["Customer", "Admin"])
    if role == "Customer":
        customer_dashboard()
    elif role == "Admin":
        admin_dashboard()

if __name__ == "__main__":
    main()
```

OUTPUT:


dtproj
localhost:8501
Finish update
Deploy
Navigation
Select Role
Customer
Customer Dashboard
Select Action
View Rooms
View Rooms
Book Room
View My Bookings
Type: Single
Price per Night: $50.00
Room Number: 102
Type: Double
Price per Night: $80.00
Room Number: 201


Navigation
Select Role
Customer
Select Action
Book Room
Book a Room
Select a Room
Room 102 (Double) - $80.00
Customer Name
subhu R
Check-In Date
2024/11/24
Check-Out Date
2024/11/26
Confirm Booking
Room Room 102 (Double) - $80.00 successfully booked for subhu R!


Navigation
Select Role
Customer
Customer Dashboard
Select Action
View My Bookings
Your Bookings
Enter Your Name
Subhu R
View
Booking ID: 1, Room ID: 5, Check-In: 2024-11-24, Check-Out: 2024-11-26



CONCLUSION:

The Hotel Booking System project demonstrates the integration of Python, Streamlit, and MySQL to create a functional, interactive, and user-friendly application for managing hotel bookings. This project encapsulates core features such as room management, customer bookings, and admin controls, providing an efficient system for real-world use.