HOTEL MANAGEMENT SYSTEMS.

A MINI-PROJECT REPORT

Submitted by

SHANGAMITHRA TS 230701305

SANGAMITHRAN A 230701283

In partial fulfillment of the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

RAJALAKSHMI ENGINEERING COLLEGE, CHENNAI

An Autonomous Institute

CHENNAI

NOVEMBER 2023

BONAFIDE CERTIFICATE

Certified that this project “HOTEL MANAGEMENT SYSTEMS” is the

bonafide work of “SHANGAMITHRA TS AND SANGAMITHRAN A” who carried out the

project work under my supervision.

SIGNATURE

MRS V JANANEE

ASSISTANT PROFESSOR

Dept. of Computer Science and Engg,

Rajalakshmi Engineering College

Chennai

SIGNATURE

MR SARAVANA GOKUL

ASSISTANT PROFESSOR

Dept. of Computer Science and Engg,

Rajalakshmi Engineering College

Chennai

This mini project report is submitted for the viva voce examination to be held on _____

TABLE OF CONTENTS

S.NO	TITLE	PAGE
	Abstract	
1	Introduction	05
2	Scope of Project	06
3	UI Diagrams	09
4	Code Implementation	22
5	Conclusion	31
6	Reference	35

ABSTRACT

Abstract:

A hotel management website serves as a comprehensive digital platform designed to streamline and enhance the overall guest experience, operations, and management of hotels. It typically offers features like online booking, reservation management, room availability, guest check-in/check-out, payment processing, and customer feedback systems. The website may also integrate with property management systems (PMS) to automate administrative tasks, track guest preferences, and handle housekeeping and maintenance requests.

Additionally, it often includes sections for marketing purposes such as special promotions, packages, and virtual tours of the hotel. The aim is to provide a seamless interface for both guests and hotel staff, improving efficiency, optimizing guest satisfaction, and driving revenue through enhanced accessibility and real-time service capabilities. With the rise of mobile and responsive design, hotel management websites are increasingly mobile-friendly, ensuring a consistent user experience across devices. By offering a central hub for all operational and service-related needs, these websites play a crucial role in modern hospitality management.

INTRODUCTION

.

In modern hotel management, efficient handling of bookings, guest information, room availability, and service requests is essential for ensuring a seamless guest experience and smooth hotel operations. Java Swing, a part of the Java Foundation Classes (JFC), provides a robust framework for building graphical user interfaces (GUIs) in Java applications. Leveraging Swing for a hotel management system allows developers to create interactive and user-friendly desktop applications with a professional interface.

When creating a hotel management page using Java Swing, the main objective is to design a simple yet effective interface that allows hotel staff to manage guest reservations, track room status, update billing information, and perform other essential tasks. Swing provides various UI components, such as buttons, text fields, labels, tables, and dialog boxes, which can be used to build the core functionalities of the hotel management system.

Key Features of a Hotel Management System Using Java Swing

1. User Authentication and Role-Based Access

- Login Page: A secure login page where hotel staff (administrators, receptionists, managers, etc.) can sign in with their credentials.
- **Role-Based Access Control:** Depending on the user's role (admin, manager, or receptionist), the system grants different levels of access to features like room management, billing, reports, and user settings.
- **Password Protection:** Ensures sensitive data is accessed only by authorized personnel.

---

2. Room Reservation and Booking Management

- Room Availability: A calendar or a room status grid that allows users to check available rooms, and dates, and make bookings in real-time.
- **Room Types & Pricing:** Displays room categories (single, double, suite) and associated prices, including seasonal or promotional rates.
- **Guest Reservation Details:** Allows entering and viewing guest details (name, contact info, check-in/check-out dates, etc.).
- **Booking Confirmation:** Sends booking confirmations via email or generates printable booking receipts.

---

3. Guest Information Management

- Guest Profiles: Stores and retrieves guest information, including personal details, booking history, and preferences.
- **Search and Filter Options:** Allows hotel staff to quickly find guest records based on name, booking ID, or other criteria.
- **Special Requests:** Ability to note and manage guest requests (e.g., late check-out, room preferences).

---

4. Check-In and Check-Out Process

- Quick Check-In/Check-Out: Streamlines the guest check-in and check-out process, including assigning rooms and processing payments.
- **Room Assignment:** Automatically assigns rooms based on guest preferences or availability, or allows manual assignment by the staff.
- **Billing and Payments:** Generates invoices based on the length of stay, services used, and any additional fees. Integration with payment gateways for online or card payments is possible.

SCOPE OF THE PROJECT

Scope of the Project: Hotel Management System Using Java Swing

The **scope of the project** for a **Hotel Management System (HMS)** developed using Java Swing includes the design, development, and implementation of a desktop-based application that automates and simplifies various hotel management tasks. This system aims to support the efficient management of hotel operations, including reservations, guest management, billing, room status tracking, and reporting.

Here is a detailed breakdown of the **scope of the project**:

Scope of the Project: Hotel Management System Using Java Swing

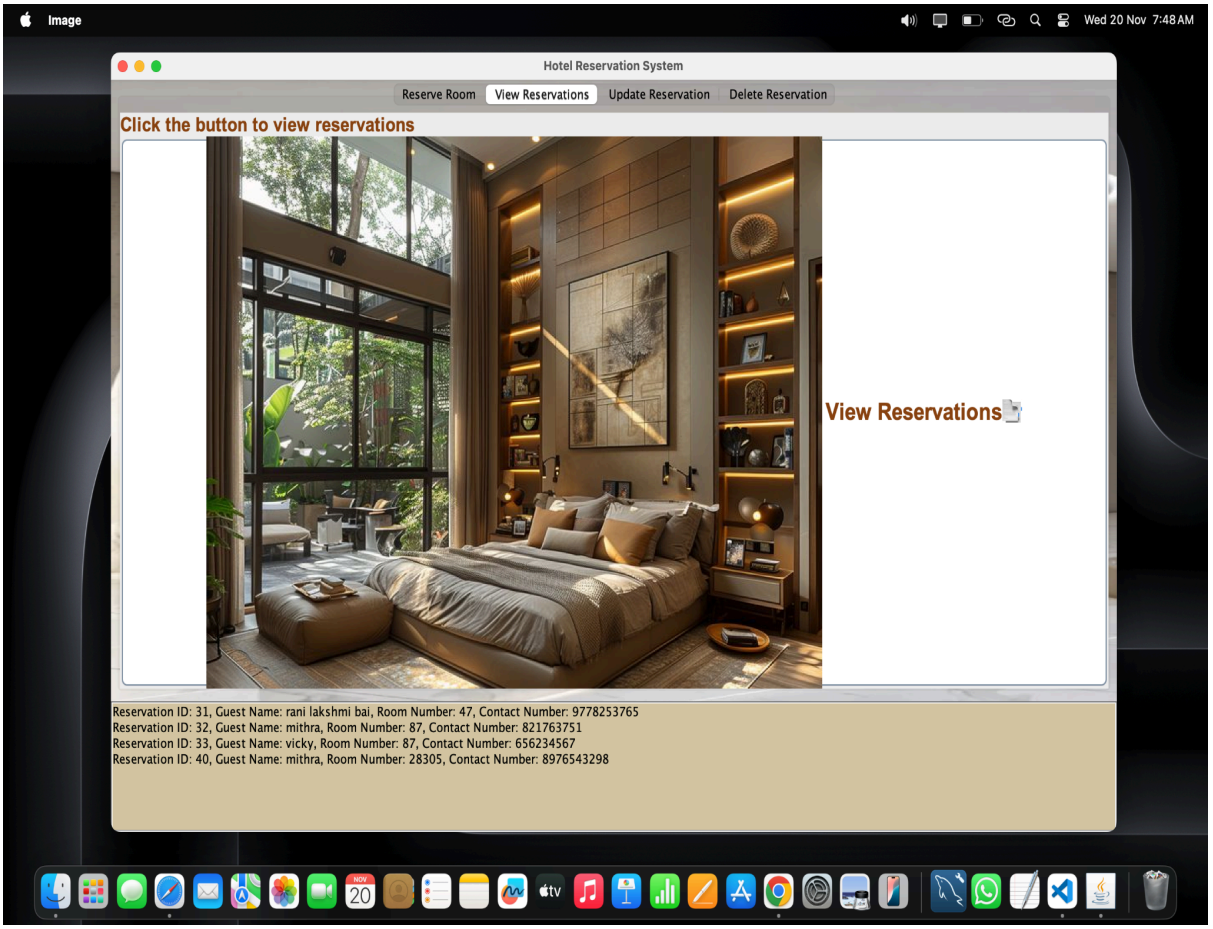
The project involves creating a **desktop-based Hotel Management System (HMS)** using **Java Swing** to automate hotel operations like **reservations**, **billing**, **room management**, and **guest information**. Key features include:

- **User Authentication**: Role-based login for staff.
- **Reservation & Room Management**: Track bookings and room availability.
- **Billing & Payments**: Generate invoices and process payments.
- **Reports**: Generate occupancy and revenue reports.
- **Maintenance & Feedback**: Manage guest feedback and room maintenance.

The system will integrate with a **relational database** (e.g., MySQL) and use **Java Swing** for a user-friendly interface. It

will be a standalone desktop application, designed for small to medium-sized hotels, with potential for future enhancements.

JAVA UI PICTURES



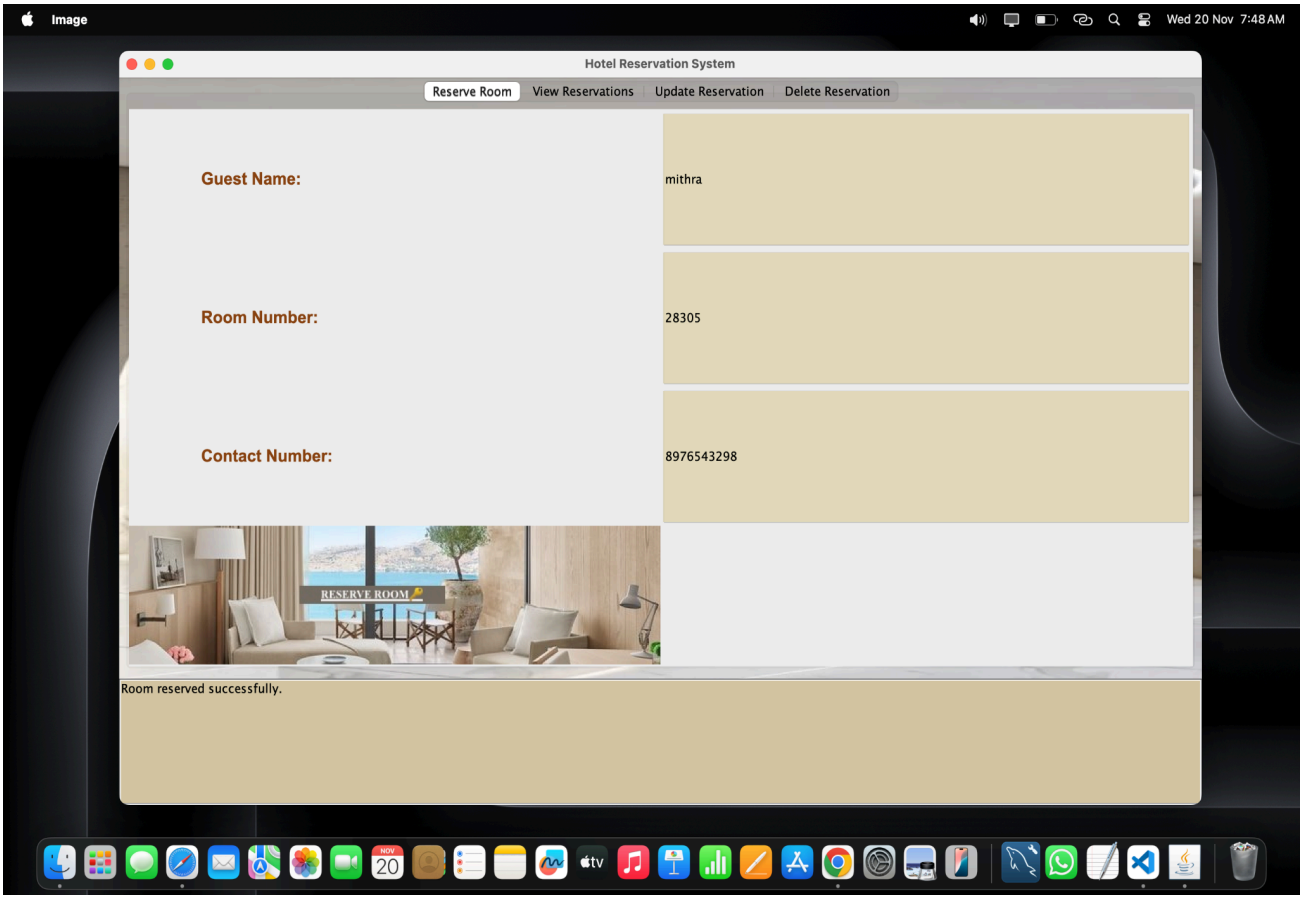


Image
Wed 20 Nov 7:48 AM
Hotel Reservation System
Reserve Room
View Reservations
Update Reservation
Delete Reservation
Guest Name:
mithra
Room Number:
28305
Contact Number:
8976543298
RESERVE ROOM
Room reserved successfully.

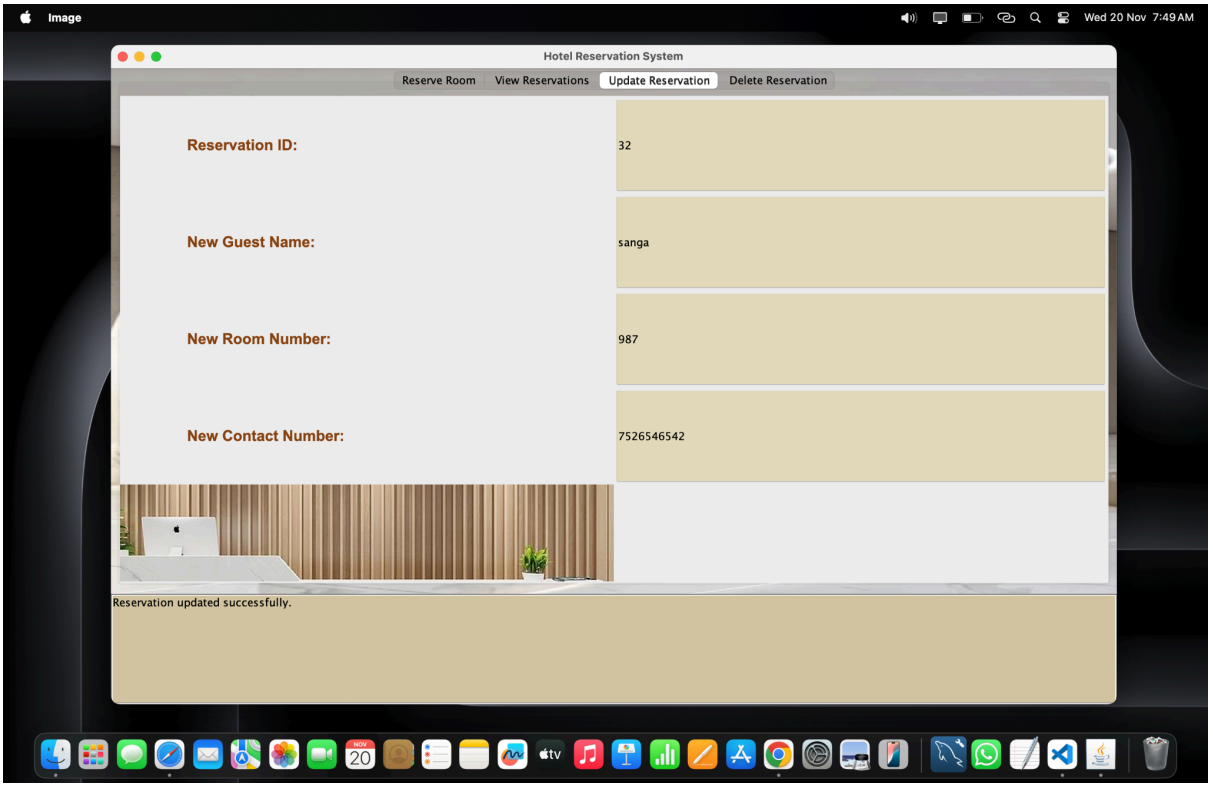


Image
Wed 20 Nov 7:49 AM
Hotel Reservation System
Reserve Room
View Reservations
Update Reservation
Delete Reservation
Reservation ID:
32
New Guest Name:
sanga
New Room Number:
987
New Contact Number:
7526546542
Reservation updated successfully.

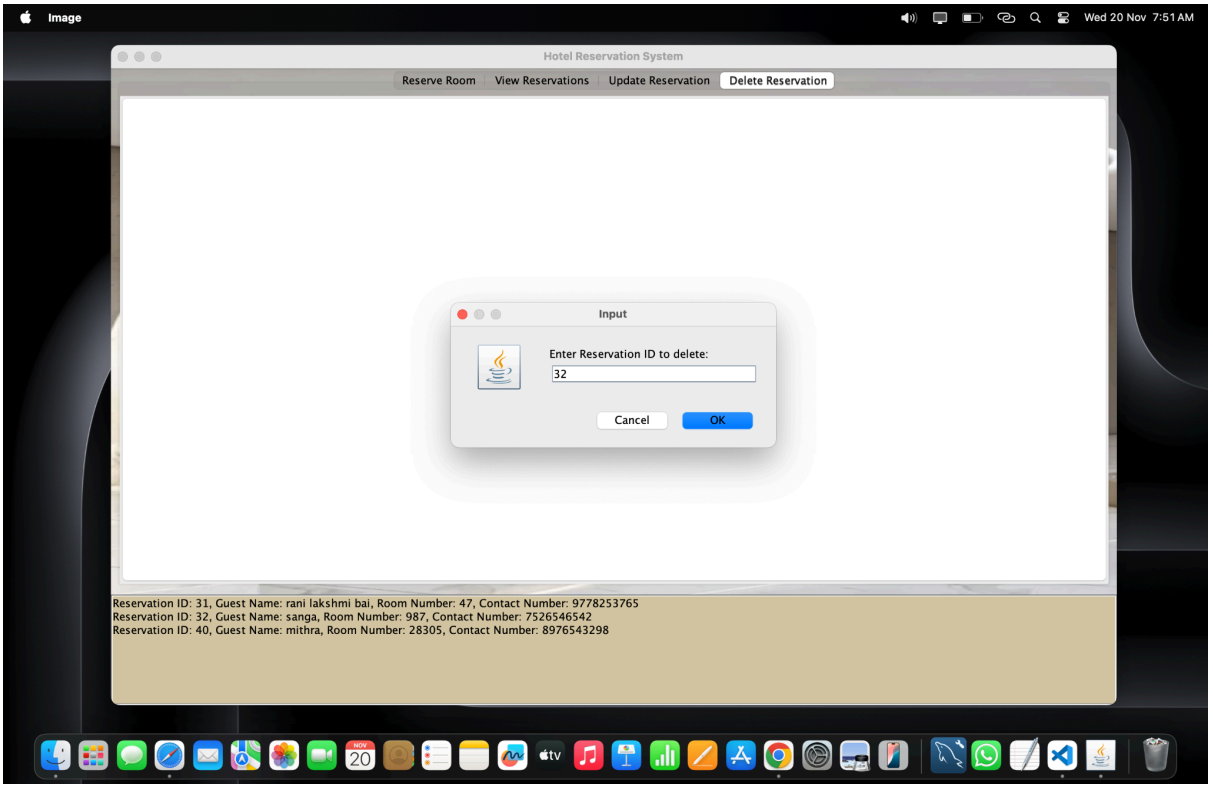


Image
Wed 20 Nov 7:51 AM
Hotel Reservation System
Reserve Room
View Reservations
Update Reservation
Delete Reservation
Input
Enter Reservation ID to delete:
32
Cancel
OK
Reservation ID: 31, Guest Name: rani lakshmi bai, Room Number: 47, Contact Number: 9778253765
Reservation ID: 32, Guest Name: sanga, Room Number: 987, Contact Number: 7526546542
Reservation ID: 40, Guest Name: mithra, Room Number: 28305, Contact Number: 8976543298

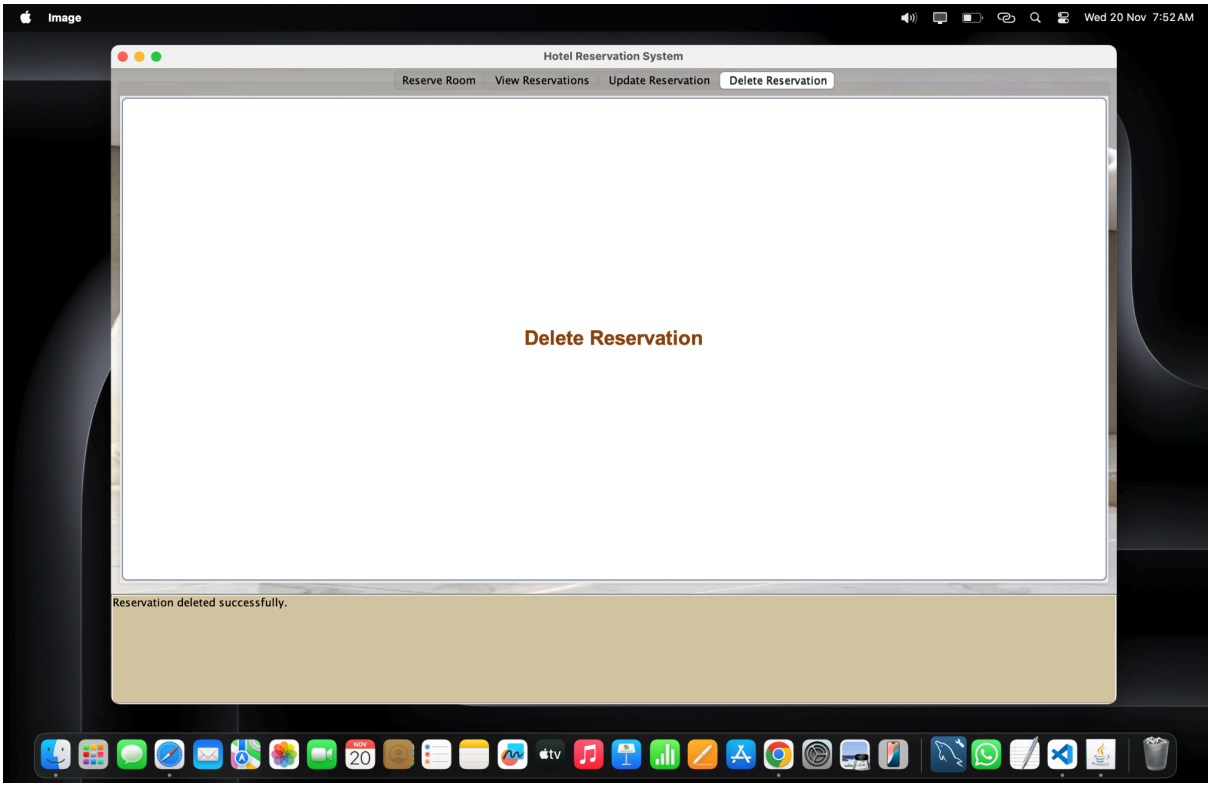
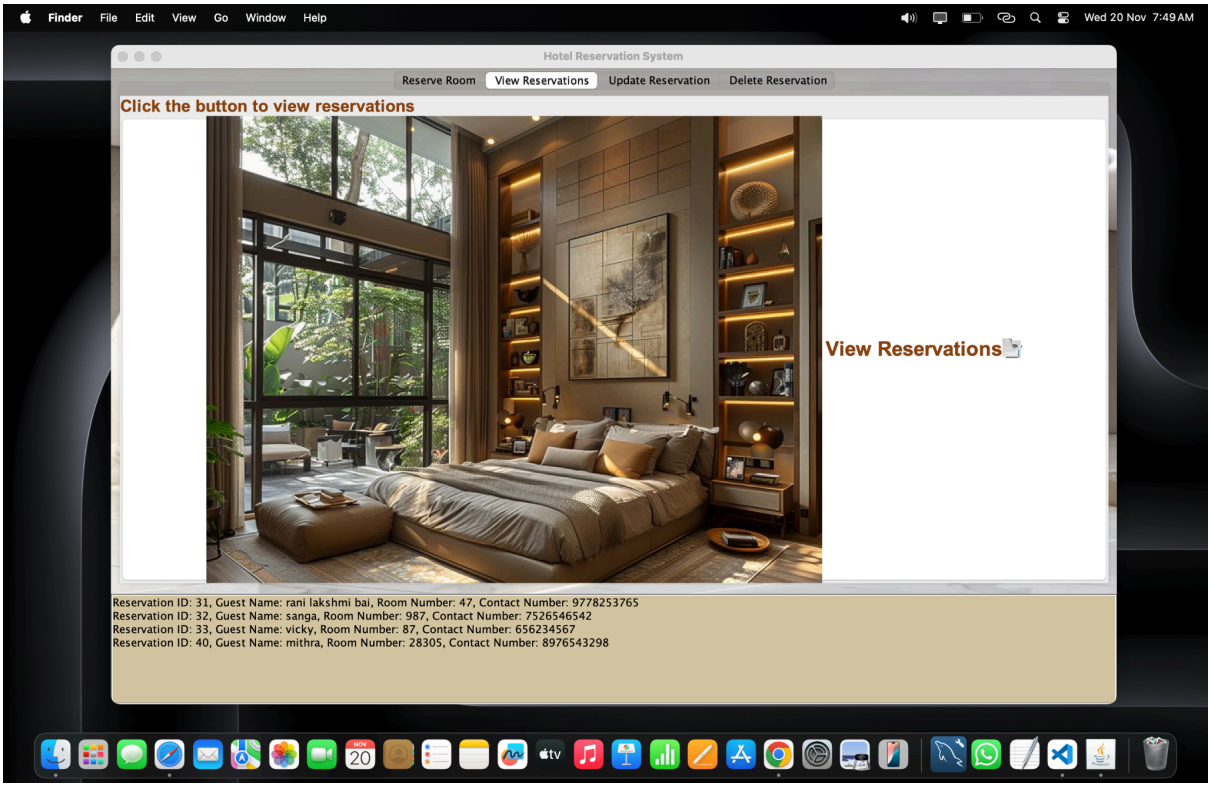
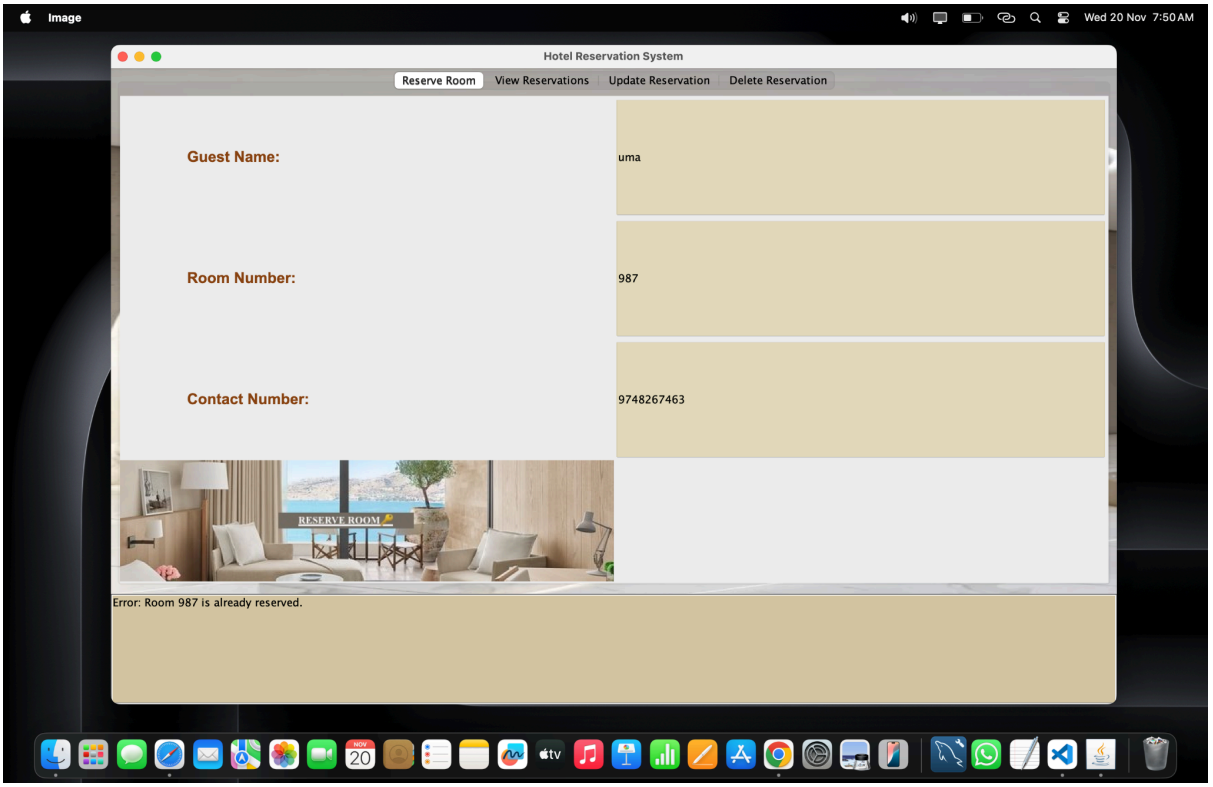


Image
Wed 20 Nov 7:52 AM
Hotel Reservation System
Reserve Room
View Reservations
Update Reservation
Delete Reservation
Delete Reservation
Reservation deleted successfully.


Finder
File
Edit
View
Go
Window
Help
Wed 20 Nov 7:49 AM
Hotel Reservation System
Reserve Room
View Reservations
Update Reservation
Delete Reservation
Click the button to view reservations
View Reservations
Reservation ID: 31, Guest Name: rani lakshmi bai, Room Number: 47, Contact Number: 9778253765
Reservation ID: 32, Guest Name: sanga, Room Number: 987, Contact Number: 7526546542
Reservation ID: 33, Guest Name: vicky, Room Number: 87, Contact Number: 656234567
Reservation ID: 40, Guest Name: mithra, Room Number: 28305, Contact Number: 8976543298



PROGRAM :

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.io.IOException;
import javax.imageio.ImageIO;
import java.io.File;
import java.sql.*;
public class Image extends JFrame {
 private static final String url = "jdbc:mysql://localhost:3306/
```

```
hotel_db";
 private static final String username = "root";
 private static final String password = "Vigshan@2116";
     private JTextField guestNameField, roomNumberField,
contactNumberField,
reservationIdField;
 private JTextField newGuestNameField, newRoomNumberField,
newContactNumberField;
 private JTextArea outputArea;
 private Connection connection;
 // ImagePanel to handle the background image
 class ImagePanel extends JPanel {
 private java.awt.Image backgroundImage;
 public ImagePanel(String imagePath) {
 try {
 backgroundImage = ImageIO.read(new File(imagePath)); // Load
the image
 } catch (IOException e) {
 System.out.println("Error: " + e.getMessage());
 }
 }
 @Override
 protected void paintComponent(Graphics g) {
 super.paintComponent(g);
 // Draw the image to fit the panel
 g.drawImage(backgroundImage, 0, 0, getWidth(), getHeight(),
```

```
this);
 }
 }
 public Image() {
 try {
 connection = DriverManager.getConnection(url, username,
password);
 } catch (SQLException e) {
 e.printStackTrace();
 JOptionPane.showMessageDialog(this, "Database connection
failed", "Error", JOptionPane.ERROR_MESSAGE);
 }
 setTitle("Hotel Reservation System");
 setSize(1080, 670);
 setLocationRelativeTo(null); // Center the frame on the screen
 setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
 initUI();
 }
 private void initUI() {
 // Create a custom panel with background image
                ImagePanel panel = new
ImagePanel("/Users/shangamithra/Downloads/
hotel.jpeg");
 panel.setLayout(new BorderLayout());
 // Create a tabbed pane to organize operations
```

```
JTabbedPane tabbedPane = new JTabbedPane();
// Create tabs for different operations
tabbedPane.addTab("Reserve Room", createReserveRoomTab());
tabbedPane.addTab("View Reservations", createViewReservationsTab());
tabbedPane.addTab("Update Reservation", createUpdateReservationTab());
tabbedPane.addTab("Delete Reservation", createDeleteReservationTab());
// Add the tabbed pane to the main panel
panel.add(tabbedPane, BorderLayout.CENTER);
// Output Area to show results (set background to nude)
outputArea = new JTextArea(8, 7);
outputArea.setEditable(false);
outputArea.setBackground(new Color(211, 198, 161)); // Nude color for output
outputArea.setForeground(Color.BLACK); // Set text color to black for contrast
JScrollPane scrollPane = new JScrollPane(outputArea);
panel.add(scrollPane, BorderLayout.SOUTH);
// Add panel to the frame
add(panel);
}
// Create Reserve Room Tab
private JPanel createReserveRoomTab() {
JPanel reservePanel = new JPanel();
```

```
reservePanel.setLayout(new GridLayout(4, 2));
JLabel guestNameLabel = new JLabel(" Guest Name:");
guestNameLabel.setForeground(new Color(139, 69, 19));
guestNameLabel.setFont(new Font("Arial", Font.BOLD, 18));
guestNameField = new JTextField();
guestNameField.setBackground(new Color(225, 218, 185));
JLabel roomNumberLabel = new JLabel(" Room Number:");
roomNumberLabel.setForeground(new Color(139, 69, 19));
roomNumberLabel.setFont(new Font("Arial", Font.BOLD, 18));
roomNumberField = new JTextField();
roomNumberField.setBackground(new Color(225, 218, 185));
JLabel contactNumberLabel = new JLabel(" Contact
Number:");
contactNumberLabel.setFont(new Font("Arial", Font.BOLD, 18));
contactNumberLabel.setForeground(new Color(139, 69, 19));
contactNumberField = new JTextField();
contactNumberField.setBackground(new Color(225, 218, 185));
JButton reserveButton = new JButton("Reserve Room");
reserveButton.setBackground(new Color(139, 69, 19));
reserveButton.setFont(new Font("Arial", Font.BOLD, 18));
reserveButton.setForeground(Color.WHITE);
                                        reserveButton.setIcon(new
ImageIcon("/Users/shangamithra/Downloads/
revr.jpeg"));
reserveButton.addActionListener(new ActionListener() {
@Override
```

```
public void actionPerformed(ActionEvent e) {
reserveRoom();
}
});
reservePanel.add(guestNameLabel);
reservePanel.add(guestNameField);
reservePanel.add(roomNumberLabel);
reservePanel.add(roomNumberField);
reservePanel.add(contactNumberLabel);
reservePanel.add(contactNumberField);
reservePanel.add(reserveButton);
return reservePanel;
}
// Create View Reservations Tab
private JPanel createViewReservationsTab() {
JPanel viewPanel = new JPanel();
viewPanel.setLayout(new BorderLayout());
JButton viewButton = new JButton("View Reservations
");
viewButton.setForeground(new Color(139, 69, 19));
viewButton.setFont(new Font("Arial", Font.BOLD, 24));
viewButton.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
viewReservations();
```

```
 }
 });
 JLabel viewReservationsLabel = new JLabel("Click the button to view
reservations");
 viewReservationsLabel.setForeground(new Color(139, 69, 19));
 viewReservationsLabel.setFont(new Font("Arial", Font.BOLD, 20));
 viewPanel.add(viewReservationsLabel, BorderLayout.NORTH);
 viewPanel.add(viewButton, BorderLayout.CENTER);
 return viewPanel;
 }
 // Create Update Reservation Tab
 private JPanel createUpdateReservationTab() {
 JPanel updatePanel = new JPanel();
 updatePanel.setLayout(new GridLayout(5, 2));
 JLabel reservationIdLabel = new JLabel(" Reservation
ID:");
 reservationIdLabel.setForeground(new Color(139, 69, 19));
 reservationIdLabel.setFont(new Font("Arial", Font.BOLD, 18));
 reservationIdField = new JTextField();
 reservationIdField.setBackground(new Color(225, 218, 185));
 JLabel newGuestNameLabel = new JLabel(" New Guest
Name:");
 newGuestNameLabel.setForeground(new Color(139, 69, 19));
 newGuestNameLabel.setFont(new Font("Arial", Font.BOLD, 18));
 newGuestNameField = new JTextField();
```

```
newGuestNameField.setBackground(new Color(225, 218, 185));
JLabel newRoomNumberLabel = new JLabel(" New Room
Number:");
newRoomNumberLabel.setForeground(new Color(139, 69, 19));
newRoomNumberLabel.setFont(new Font("Arial", Font.BOLD, 18));
newRoomNumberField = new JTextField();
newRoomNumberField.setBackground(new Color(225, 218, 185));
JLabel newContactNumberLabel = new JLabel(" New
Contact Number:");
newContactNumberLabel.setForeground(new Color(139, 69, 19));
  newContactNumberLabel.setFont(new  Font("Arial",  Font.BOLD,
18));
newContactNumberField = new JTextField();
newContactNumberField.setBackground(new Color(225, 218, 185));
JButton updateButton = new JButton("Update Reservation");
updateButton.setForeground(new Color(139, 69, 19));
updateButton.setFont(new Font("Arial", Font.BOLD, 18));
                                        updateButton.setIcon(new
ImageIcon("/Users/shangamithra/Downloads/
ho.jpeg"));
updateButton.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
String reservationId = reservationIdField.getText().trim();
String newGuestName = newGuestNameField.getText().trim();
String newRoomNumber = newRoomNumberField.getText().trim();
```

```
String newContactNumber =
newContactNumberField.getText().trim();
if (reservationId.isEmpty() || newGuestName.isEmpty() ||
newRoomNumber.isEmpty() || newContactNumber.isEmpty()) {
outputArea.setText("Please fill all fields");
} else {
updateReservation(reservationId, newGuestName,
newRoomNumber, newContactNumber);
}
}
});
updatePanel.add(reservationIdLabel);
updatePanel.add(reservationIdField);
updatePanel.add(newGuestNameLabel);
updatePanel.add(newGuestNameField);
updatePanel.add(newRoomNumberLabel);
updatePanel.add(newRoomNumberField);
updatePanel.add(newContactNumberLabel);
updatePanel.add(newContactNumberField);
updatePanel.add(updateButton);
return updatePanel;
}
// Create Delete Reservation Tab
private JPanel createDeleteReservationTab() {
JPanel deletePanel = new JPanel();
```

```
deletePanel.setLayout(new BorderLayout());
JButton deleteButton = new JButton("Delete Reservation");
deleteButton.setFont(new Font("Arial", Font.BOLD, 24));
deleteButton.setForeground(new Color(139, 69, 19));
deleteButton.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
deleteReservation();
}
});
deletePanel.add(deleteButton, BorderLayout.CENTER);
return deletePanel;
}
// Method to Reserve Room
private void reserveRoom() {
String guestName = guestNameField.getText().trim();
String roomNumber = roomNumberField.getText().trim();
String contactNumber = contactNumberField.getText().trim();
if (guestName.isEmpty() || roomNumber.isEmpty() ||
contactNumber.isEmpty()) {
outputArea.setText("Please fill all fields.");
return;
}
try {
// Check if the room number is already reserved
```

```
String checkRoomSQL = "SELECT * FROM reservations WHERE
room_number = ?";
PreparedStatement checkStmt =
connection.prepareStatement(checkRoomSQL);
checkStmt.setString(1, roomNumber);
ResultSet rs = checkStmt.executeQuery();
if (rs.next()) {
outputArea.setText("Error: Room " + roomNumber + " is
already reserved.");
return;
}
// If the room is not already reserved, proceed to insert the
reservation
String sql = "INSERT INTO reservations (guest_name, room_number,
comtact_number) VALUES (?, ?, ?)";
PreparedStatement stmt = connection.prepareStatement(sql);
stmt.setString(1, guestName);
stmt.setString(2, roomNumber);
stmt.setString(3, contactNumber);
stmt.executeUpdate();
outputArea.setText("Room reserved successfully.");
} catch (SQLException e) {
e.printStackTrace();
outputArea.setText("Error while reserving room.");
}
```

```
}
// Method to View Reservations
private void viewReservations() {
try {
String sql = "SELECT * FROM reservations";
PreparedStatement stmt = connection.prepareStatement(sql);
ResultSet rs = stmt.executeQuery();
StringBuilder sb = new StringBuilder();
while (rs.next()) {
sb.append("Reservation ID:
").append(rs.getInt("reservation_id"))
.append(", Guest Name:
").append(rs.getString("guest_name"))
.append(", Room Number:
").append(rs.getString("room_number"))
.append(", Contact Number:
").append(rs.getString("comtact_number"))
.append("\n");
}
outputArea.setText(sb.toString());
} catch (SQLException e) {
e.printStackTrace();
outputArea.setText("Error while fetching reservations.");
}
}
```

```
// Method to Update Reservation
private void updateReservation(String reservationId, String
newGuestName, String newRoomNumber, String
newContactNumber) {
try {
String sql = "UPDATE reservations SET guest_name = ?,
room_number = ?, comtact_number = ? WHERE reservation_id = ?";
PreparedStatement stmt = connection.prepareStatement(sql);
stmt.setString(1, newGuestName);
stmt.setString(2, newRoomNumber);
stmt.setString(3, newContactNumber);
stmt.setInt(4, Integer.parseInt(reservationId));
int rowsAffected = stmt.executeUpdate();
if (rowsAffected > 0) {
outputArea.setText("Reservation updated successfully.");
} else {
outputArea.setText("No reservation found with the given
ID.");
}
} catch (SQLException e) {
e.printStackTrace();
outputArea.setText("Error while updating reservation.");
}
}
// Method to Delete Reservation
private void deleteReservation() {
```

```
String reservationId = JOptionPane.showInputDialog(this, "Enter
Reservation ID to delete:");
if (reservationId != null && !reservationId.trim().isEmpty()) {
try {
String sql = "DELETE FROM reservations WHERE reservation_id
= ?";
PreparedStatement stmt = connection.prepareStatement(sql);
stmt.setInt(1, Integer.parseInt(reservationId));
int rowsAffected = stmt.executeUpdate();
if (rowsAffected > 0) {
outputArea.setText("Reservation deleted successfully.");
} else {
outputArea.setText("No reservation found with the given
ID.");
}
} catch (SQLException e) {
e.printStackTrace();
outputArea.setText("Error while deleting reservation.");
}
}
}
public static void main(String[] args) {
SwingUtilities.invokeLater(new Runnable() {
@Override
public void run() {
```

```
new Image().setVisible(true);
}
});
}
}
```

CONCLUSION:

The **Hotel Management System** project using **Java Swing** provides a user-friendly interface for managing hotel operations such as booking rooms, checking room availability, handling customer check-ins and check-outs, and generating bills. By utilizing Java Swing for the graphical user interface (GUI), the system ensures an interactive and responsive experience for users. The project allows efficient management of guest details, room assignments, and transactions, while also offering features like data validation and error handling. It demonstrates the application of object-oriented principles and the use of event-driven

programming in Java for real-world business management tasks.

REFERENCE:

https://www.google.com/