HOTEL MANAGEMENT SYSTEM

HotelManagementsystem.java

```
package hms;


import javax.swing.*;

import javax.swing.table.*;

import java.awt.*;

import java.awt.event.*;

import java.util.*;


public class HotelManagementSystem extends JFrame implements ActionListener {


    static class Room {

        int number;

        String type;

        boolean isAvailable;

        double pricePerNight;

        String guestName;

        String guestPhone;

        int daysStayed;


        public Room(int number, String type, double pricePerNight) {

            this.number = number;

            this.type = type;

            this.pricePerNight = pricePerNight;

            this.isAvailable = true;

            this.guestName = "";

            this.guestPhone = "";
        this.daysStayed = 0;
    }
}

private java.util.List<Room> rooms = new ArrayList<>();
private JTable table;
private DefaultTableModel model;
private JButton checkInBtn = new JButton("Check-In");
private JButton checkOutBtn = new JButton("Check-Out");
private JButton staffBtn = new JButton("Staff Panel");

public HotelManagementSystem() {
    setTitle("Hotel Management System");
    setExtendedState(JFrame.MAXIMIZED_BOTH);
    setDefaultCloseOperation(EXIT_ON_CLOSE);
    setLayout(new BorderLayout(10, 10));

    // Load data from database
    rooms = DBManager.getAllRooms();

    // Table setup
    String[] columns = {"Room Number", "Type", "Price/Night", "Available"};
    model = new DefaultTableModel(columns, 0) {
        public boolean isCellEditable(int row, int col) {
            return false;
        }
    };
    table = new JTable(model) {
        public Component prepareRenderer(TableCellRenderer renderer, int row, int column) {
            Component c = super.prepareRenderer(renderer, row, column);
        Room r = rooms.get(row);
        c.setBackground(r.isAvailable ? new Color(102, 255, 102) : new Color(255, 102, 102));
        if (isCellSelected(row, column)) {
            c.setBackground(Color.YELLOW);
        }
        c.setForeground(Color.BLACK);
        return c;
    }
};

table.setRowHeight(35);
table.setFont(new Font("Arial", Font.BOLD, 16));
table.getTableHeader().setFont(new Font("Arial", Font.BOLD, 18));
table.getTableHeader().setBackground(Color.ORANGE);
table.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
refreshTable();

add(new JScrollPane(table), BorderLayout.CENTER);

// Button panel
JPanel bottomPanel = new JPanel(new FlowLayout());
bottomPanel.setBackground(Color.CYAN);

checkInBtn.setBackground(Color.GREEN);
checkOutBtn.setBackground(Color.RED);
staffBtn.setBackground(Color.ORANGE);

checkInBtn.setFont(new Font("Arial", Font.BOLD, 16));
checkOutBtn.setFont(new Font("Arial", Font.BOLD, 16));
staffBtn.setFont(new Font("Arial", Font.BOLD, 16));
```

```
        checkInBtn.addActionListener(this);
        checkOutBtn.addActionListener(this);
        staffBtn.addActionListener(this);

        bottomPanel.add(checkInBtn);
        bottomPanel.add(checkOutBtn);
        bottomPanel.add(staffBtn);

        add(bottomPanel, BorderLayout.SOUTH);
    }

    private void refreshTable() {
        rooms = DBManager.getAllRooms();
        model.setRowCount(0);
        for (Room r : rooms) {
            model.addRow(new Object[]{r.number, r.type, r.pricePerNight, r.isAvailable ? "Yes" : "No"});
        }
    }

    public void actionPerformed(ActionEvent e) {
        int selectedRow = table.getSelectedRow();
        if (selectedRow == -1 && e.getSource() != staffBtn) {
            JOptionPane.showMessageDialog(this, "Please select a room first!");
            return;
        }

        Room selectedRoom = selectedRow >= 0 ? rooms.get(selectedRow) : null;

        if (e.getSource() == checkInBtn) {
```

```
        if (!selectedRoom.isAvailable) {
            JOptionPane.showMessageDialog(this, "Room is already occupied!");
            return;
        }

        String name = JOptionPane.showInputDialog(this, "Enter Guest Name:");
        String phone = JOptionPane.showInputDialog(this, "Enter Guest Phone Number:");
        String daysStr = JOptionPane.showInputDialog(this, "Enter number of days to stay:");

        try {
            int days = Integer.parseInt(daysStr);
            selectedRoom.guestName = name;
            selectedRoom.guestPhone = phone;
            selectedRoom.daysStayed = days;
            selectedRoom.isAvailable = false;
            DBManager.checkInRoom(selectedRoom);

            double bill = selectedRoom.pricePerNight * days;
            JOptionPane.showMessageDialog(this,
                    "Booking Confirmed for " + name + "\nTotal Bill: ₹" + bill,
                    "Success", JOptionPane.INFORMATION_MESSAGE);
            refreshTable();
        } catch (NumberFormatException ex) {
            JOptionPane.showMessageDialog(this, "Invalid number of days!");
        }

    } else if (e.getSource() == checkOutBtn) {
        if (selectedRoom.isAvailable) {
            JOptionPane.showMessageDialog(this, "Room is already empty!");
            return;
            }


            DBManager.checkOutRoom(selectedRoom.number);
            JOptionPane.showMessageDialog(this,
                    "Checked out room " + selectedRoom.number + "\nThank you " +
selectedRoom.guestName + "!",
                    "Check-Out", JOptionPane.INFORMATION_MESSAGE);
            refreshTable();


        } else if (e.getSource() == staffBtn) {
            String password = JOptionPane.showInputDialog(this, "Enter Staff Password:");
            if ("admin123".equals(password)) {
                showStaffPanel();
            } else {
                JOptionPane.showMessageDialog(this, "Incorrect password!");
            }
        }
    }


    private void showStaffPanel() {
        StringBuilder sb = new StringBuilder("Current Bookings:\n\n");
        for (Room r : rooms) {
            if (!r.isAvailable) {
                sb.append("Room ").append(r.number).append(" - ").append(r.type)
                  .append(" - Guest: ").append(r.guestName)
                  .append(", Phone: ").append(r.guestPhone)
                  .append(", Days: ").append(r.daysStayed).append("\n");
            }
        }
        if (sb.toString().equals("Current Bookings:\n\n")) {
            sb.append("No rooms are currently occupied.");
        }

        JTextArea textArea = new JTextArea(sb.toString());
        textArea.setFont(new Font("Arial", Font.PLAIN, 16));
        textArea.setEditable(false);
        JOptionPane.showMessageDialog(this, new JScrollPane(textArea), "Staff Panel", JOptionPane.INFORMATION_MESSAGE);
    }

    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> new HotelManagementSystem().setVisible(true));
    }
}
```

DBManager.java

```
package hms;

import java.sql.*;
import java.util.*;

public class DBManager {
    private static final String URL = "jdbc:mysql://localhost:3306/hotel_db";
    private static final String USER = "root";        // replace with your MySQL username
    private static final String PASSWORD = "root"; // replace with your MySQL password

    public static Connection getConnection() throws SQLException {
        return DriverManager.getConnection(URL, USER, PASSWORD);
    }

    public static List<HotelManagementSystem.Room> getAllRooms() {
        List<HotelManagementSystem.Room> rooms = new ArrayList<>();
        String roomQuery = "SELECT * FROM rooms";
        String bookingQuery = "SELECT * FROM bookings WHERE room_number = ?";

        try (Connection conn = getConnection();
             Statement stmt = conn.createStatement();
             ResultSet rs = stmt.executeQuery(roomQuery)) {

          while (rs.next()) {
            int number = rs.getInt("room_number");
            String type = rs.getString("type");
            double price = rs.getDouble("price_per_night");

            HotelManagementSystem.Room r = new HotelManagementSystem.Room(number, type, price);

            try (PreparedStatement ps = conn.prepareStatement(bookingQuery)) {
              ps.setInt(1, number);
              ResultSet brs = ps.executeQuery();
              if (brs.next()) {
                r.isAvailable = false;
                r.guestName = brs.getString("guest_name");
                r.guestPhone = brs.getString("guest_phone");
                r.daysStayed = brs.getInt("days_stayed");
              }
            }

            rooms.add(r);
          }
        } catch (SQLException e) {
            e.printStackTrace();
        }

        return rooms;
    }

    public static void checkInRoom(HotelManagementSystem.Room r) {
        String sql = "INSERT INTO bookings (room_number, guest_name, guest_phone, days_stayed) VALUES (?, ?, ?, ?)";

        try (Connection conn = getConnection();
             PreparedStatement ps = conn.prepareStatement(sql)) {

            ps.setInt(1, r.number);
            ps.setString(2, r.guestName);
            ps.setString(3, r.guestPhone);
            ps.setInt(4, r.daysStayed);
            ps.executeUpdate();

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public static void checkOutRoom(int roomNumber) {
        String sql = "DELETE FROM bookings WHERE room_number = ?";

        try (Connection conn = getConnection();
             PreparedStatement ps = conn.prepareStatement(sql)) {
```

```
            ps.setInt(1, roomNumber);
            ps.executeUpdate();

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```