

Project Title

Hotel Management System

1. Abstract

This Hotel Management System, built with Java (NetBeans) and MySQL, provides an efficient solution for managing hotel room operations. The system features real-time room tracking with details like room number, type, price, and status. Administrators can easily add new rooms with automatic validation and instant database updates. Designed with a user-friendly Swing interface and secure JDBC connectivity, it offers a reliable foundation for small to medium hotels. The system is easily expandable for additional functionalities like bookings, staff management, and billing.

2. Introduction

This Hotel Management System is a Java-based desktop application designed to streamline room operations for small hotels. Developed with NetBeans IDE and MySQL, it provides an intuitive interface for staff to manage room inventory efficiently. The system displays real-time room status (available/occupied/maintenance) with key details like pricing and room type

3. Technologies Used

- Programming (Java)
- Backend (MySQL)
- IDE (NetBeans)
- Communication (Socket (TCP))
- Architecture (Client-Server)

4. Features

- Add/Edit/Delete Rooms
- Guest Details (name, phone, ID)
- Bill Generation (room price)
- Update Room Status (Available/Occupied/Maintenance)

5. Project Description

Modules:

- **DB.java**: establishes and manages secure MySQL database connections for the Hotel Management System using JDBC.
- **RoomManagement.java**: handles the display, addition, and management of hotel rooms through a Swing GUI while syncing with the MySQL database
- **GuestCheckIn.java**: handles guest registration, check-in/check-out processes, and updates room status in the database.
- **MainMenu.java**: provides the main navigation interface with buttons to access all system modules (Rooms, Guests, Bookings, etc.)

6. Code:

```
1 import java.sql.*;
2
3 public class DB {
4     public static Connection getConnection() {
5         try {
6             Class.forName("com.mysql.jdbc.Driver");
7             Connection con = DriverManager.getConnection(
8                 "jdbc:mysql://localhost:3306/mini_hotel",
9                 "root",
10                "");
11             return con;
12         } catch (Exception e) {
13             System.out.println(e);
14             return null;
15         }
16     }
17 }
```

```

Source History
1 import javax.swing.*;
2 import java.awt.*;
3 import java.sql.*;
4
5 public class GuestCheckIn extends JFrame {
6
7     private JTextField nameField, phoneField, roomField;
8
9     public GuestCheckIn() {
10         setTitle("Guest Check-In");
11         setSize(400, 300);
12         setLayout(new GridLayout(5, 2));
13
14         add(new JLabel("Guest Name:"));
15         nameField = new JTextField();
16         add(nameField);
17
18         add(new JLabel("Phone:"));
19         phoneField = new JTextField();
20         add(phoneField);
21
22         add(new JLabel("Room No:"));
23         roomField = new JTextField();
24         add(roomField);
25
26         JButton saveBtn = new JButton("Check In");
27         saveBtn.addActionListener(e -> saveGuest());
28         add(saveBtn);
29
30         setVisible(true);
31     }
32
33     private void saveGuest() {
34         String name = nameField.getText();
35         String phone = phoneField.getText();
36         String roomNo = roomField.getText();
37
38         try {
39             Connection con = DB.getConnection();
40             PreparedStatement ps = con.prepareStatement(
41 "INSERT INTO guests(name, phone, room_no, check_in) VALUES(?,?,?,CURDATE())");
42             ps.setString(1, name);
43             ps.setString(2, phone);
44             ps.setString(3, roomNo);
45             ps.executeUpdate();
46
47             // Update room status
48             Statement stmt = con.createStatement();
49             stmt.executeUpdate(
50 "UPDATE rooms SET status='Occupied' WHERE room_no='"+roomNo+"'");
51
52 JOptionPane.showMessageDialog(this, "Guest checked in successfully!");
53             con.close();
54         } catch (Exception e) {
55             JOptionPane.showMessageDialog(this, e.getMessage());
56         }
57     }
58
59     public static void main(String[] args) {
60         new GuestCheckIn();
61     }
62 }

```

```

1 import javax.swing.*;
2 import javax.swing.table.DefaultTableModel;
3 import java.awt.*;
4 import java.awt.event.*;
5 import java.sql.*;
6
7 public class RoomManagement extends JFrame {
8     private JTable table;
9     private DefaultTableModel model;
10    private JButton refreshBtn, addBtn;
11
12    public RoomManagement() {
13        setTitle("Room Management");
14        setSize(700, 400);
15        setLayout(new BorderLayout());
16        String[] columns = {"Room ID", "Room Number", "Type", "Price", "Status"};
17        model = new DefaultTableModel(columns, 0) {
18            @Override
19            public boolean isCellEditable(int row, int column) {
20                return false;
21            }
22        };
23        table = new JTable(model);
24        table.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
25        JPanel buttonPanel = new JPanel(new FlowLayout());
26        refreshBtn = new JButton("Refresh Data");
27        addBtn = new JButton("Add New Room");
28        refreshBtn.addActionListener(e -> loadRooms());
29        addBtn.addActionListener(e -> showAddRoomDialog());
30        buttonPanel.add(refreshBtn);
31        buttonPanel.add(addBtn);
32        add(new JScrollPane(table), BorderLayout.CENTER);
33        add(buttonPanel, BorderLayout.SOUTH);
34        loadRooms();
35        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
36        setLocationRelativeTo(null);
37        setVisible(true);
38    }
39
40    private void loadRooms() {
41        model.setRowCount(0);
42        try (Connection con = DB.getConnection();
43             Statement stmt = con.createStatement()) {
44            ResultSet rs = stmt.executeQuery("SELECT * FROM rooms ORDER BY room_no");
45            while(rs.next()) {
46                Object[] row = {
47                    rs.getInt("id"),
48                    rs.getString("room_no"),
49                    rs.getString("type"),
50                    rs.getDouble("price"),
51                    rs.getString("status")
52                };
53                model.addRow(row);
54            }
55        } catch (SQLException e) {
56            JOptionPane.showMessageDialog(this,
57 "Error loading rooms: " + e.getMessage(),
58 "Database Error", JOptionPane.ERROR_MESSAGE);
59        }
60    }
61
62    private void showAddRoomDialog() {
63        JTextField roomNoField = new JTextField(10);
64        JComboBox<String> typeCombo = new JComboBox<>(new String[]{"Standard", "Deluxe", "Suite"});
65        JTextField priceField = new JTextField(10);
66        JComboBox<String> statusCombo = new JComboBox<>(new String[]{"Available", "Maintenance"});
67        JPanel panel = new JPanel(new GridLayout(4, 2, 5, 5));
68        panel.add(new JLabel("Room Number:"));
69        panel.add(roomNoField);
70        panel.add(new JLabel("Room Type:"));
71        panel.add(typeCombo);
72        panel.add(new JLabel("Price per Night:"));
73        panel.add(priceField);
74        panel.add(new JLabel("Initial Status:"));
75        panel.add(statusCombo);
76        int result = JOptionPane.showConfirmDialog(
77 this, panel, "Add New Room",
78 JOptionPane.OK_CANCEL_OPTION, JOptionPane.PLAIN_MESSAGE);
79        if (result == JOptionPane.OK_OPTION) {
80            addNewRoom(
81                roomNoField.getText().trim(),
82                (String) typeCombo.getSelectedItem(),
83                priceField.getText().trim(),
84                (String) statusCombo.getSelectedItem()
85            );
86        }
87    }
88
89    private void addNewRoom(String roomNo, String type, String priceStr, String status) {
90        if (roomNo.isEmpty() || priceStr.isEmpty()) {
91            JOptionPane.showMessageDialog(this,
92 "Room number and price are required",
93 "Validation Error", JOptionPane.WARNING_MESSAGE);
94            return;
95        }
96        try {
97            double price = Double.parseDouble(priceStr);
98            if (price <= 0) {
99                throw new NumberFormatException();
100            }
101            try (Connection con = DB.getConnection();
102                 PreparedStatement ps = con.prepareStatement(
103 "INSERT INTO rooms (room_no, type, price, status) VALUES (?, ?, ?, ?)")) {
104                ps.setString(1, roomNo);
105                ps.setString(2, type);
106                ps.setDouble(3, price);
107                ps.setString(4, status);
108                ps.executeUpdate();
109                int rowsAffected = ps.executeUpdate();
110                if (rowsAffected > 0) {
111                    JOptionPane.showMessageDialog(this,
112 "Room added successfully!",
113 "Success", JOptionPane.INFORMATION_MESSAGE);
114                    loadRooms();
115                }
116            } catch (SQLException e) {
117                if (e.getMessage().contains("Duplicate entry")) {
118                    JOptionPane.showMessageDialog(this,
119 "Room number already exists",
120 "Error", JOptionPane.ERROR_MESSAGE);
121                } else {
122                    throw e;
123                }
124            }
125        } catch (NumberFormatException e) {
126            JOptionPane.showMessageDialog(this,
127 "Please enter a valid positive number for price",
128 "Invalid Price", JOptionPane.WARNING_MESSAGE);
129        } catch (SQLException e) {
130            JOptionPane.showMessageDialog(this,
131 "Error adding room: " + e.getMessage(),
132 "Database Error", JOptionPane.ERROR_MESSAGE);
133        }
134    }
135
136    public static void main(String[] args) {
137        SwingUtilities.invokeLater(() -> new RoomManagement());
138    }
139 }

```

```

1 import javax.swing.*;
2 import java.awt.*;
3 import java.awt.event.*;
4
5 public class MainMenu extends JFrame {
6
7     public MainMenu() {
8         setTitle("Hotel Management System");
9         setSize(300, 200);
10
11         // Create a panel with GridLayout
12         JPanel mainPanel = new JPanel(new GridLayout(3, 1, 5, 5)); // 3 rows, 1 column, 5px gaps
13
14         JButton roomBtn = new JButton("Room Management");
15         roomBtn.addActionListener(e -> {
16             new RoomManagement().setVisible(true);
17             dispose();
18         });
19
20         JButton guestBtn = new JButton("Guest Check-In");
21         guestBtn.addActionListener(e -> {
22             new GuestCheckIn().setVisible(true);
23             dispose();
24         });
25
26         JButton exitBtn = new JButton("Exit");
27         exitBtn.addActionListener(e -> System.exit(0));
28
29         // Add components to panel
30         mainPanel.add(roomBtn);
31         mainPanel.add(guestBtn);
32         mainPanel.add(exitBtn);
33
34         // Add panel to frame with some border spacing
35         add(mainPanel, BorderLayout.CENTER);
36         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
37         setLocationRelativeTo(null); // Center window
38     }
39
40     public static void main(String[] args) {
41         SwingUtilities.invokeLater(() -> new MainMenu().setVisible(true));
42     }
43 }

```

7. Output:

Room Management				
Room ID	Room Number	Type	Price	Status
5	101	Standard	1000.0	Maintenance
1	11	Standard	2000.0	Occupied
2	12	Deluxe	3000.0	Available
3	13	Suite	5000.0	Occupied
6	33	Standard	6000.0	Available
<div> <div>Refresh Data</div> <div>Add New Room</div> </div>				

Guest Check-In

Guest Name:

mostakim

Phone:

016366661899

Room No:

33

Check In

				id	name	phone	room_no	check_in	check_out			
<input type="checkbox"/>		Edit		Copy		Delete	5	mosta	0177	11	2025-06-19	NULL
<input type="checkbox"/>		Edit		Copy		Delete	6	foysal	0987	13	2025-06-19	NULL
<input type="checkbox"/>		Edit		Copy		Delete	8	mostakim	016366661899	33	2025-06-19	NULL

8. Conclusion:

This Hotel Management System project delivers a complete digital solution using Java (Swing), MySQL, and TCP sockets. It automates and streamlines hotel operations through room booking, guest management, and real-time updates. The modular design allows easy addition of new features (e.g., payment gateway, SMS alerts). With built-in security (JDBC Prepared Statements) and a user-friendly interface, it's an ideal solution for small and mid-sized hotels. Future upgrades could include Android apps or cloud integration for enhanced scalability.