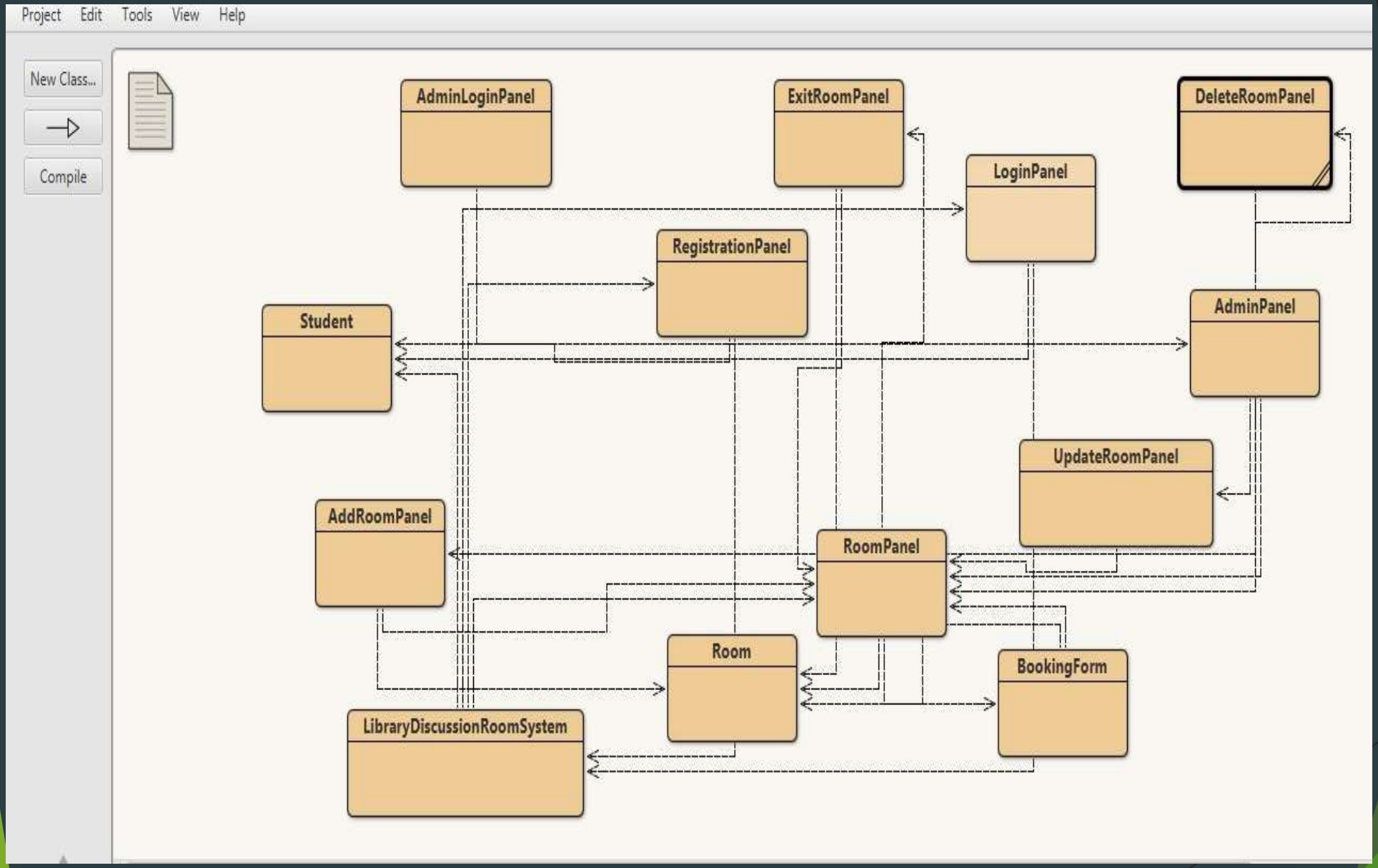
OOP PROJECT PRESENTATION

TOPIC: LIBRARY DISCUSSION ROOM SYSTEM

Members:

Sonia	F23605098
Arjia Arshad	F23605088
Umme Aimen	F23605065
Tahira Tahir	F236050107

Project Overview



List of Classes

- Student
- Room Panel
- Admin-Login Panel
- Admin Panel
- Library Discussion
- Add Room
- Registration
- Booking Form
- Update-Room Panel
- Delete Room
- Login Panel
- Exit Room Panel

Division of Classes Amongst Group Mem

ARJIA	SONIA	AIMEN	T
Student	RoomPanel	Admin-Login Panel	Admin
RegistrationPanel	BookingForm	Update-Room Panel	Delet
LoginPanel	Exit RoomPanel	LibraryDiscussion	AddRo
		Room	

Purpose Of The Project

A Library Discussion Room Management System streamlines the process of reserving and utilizing discussion rooms in a library. It efficiently allocates resources, allows users to book rooms in advance, and manages time slots to ensure fair access. With features like access control and automated notifications, it enhances security and user convenience. Usage tracking and data analysis provide insights for resource optimization and decision-making. Overall, it improves the efficiency of library services and enhances the user experience.

Student Class

```java
import java.util.regex.Pattern;

public class Student {
    private String studentID;
    private String name;
    private String batch;
    private String department;
    private String password;

    public Student(String studentID, String name, String batch, String department, String password) {
        this.studentID = studentID;
        this.name = name;
        this.batch = batch;
        this.department = department;
        this.password = password;
    }

    public boolean validate() {
        // Validation rules
        boolean isValid = true;

        // Student ID validation
        isValid = isValid && Pattern.matches("[fF]\\d{8}", studentID);

        // Other fields not empty validation
        isValid = isValid && !name.isEmpty() && !batch.isEmpty() && !department.isEmpty() && !password.isEmpty

        // Batch validation
        isValid = isValid && Pattern.matches("\\d{4}", batch);

        return isValid;
```

Student Class

```
}

public String getStudentID() {
    return studentID;
}

public String getName() {
    return name;
}

public String getBatch() {
    return batch;
}

public String getDepartment() {
    return department;
}

public String getPassword() {
    return password;
}
```

Registration-Panel Class

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.util.regex.Pattern;

public class RegistrationPanel extends JPanel {
    public RegistrationPanel(JFrame frame) {
        setLayout(new GridBagLayout());
        GridBagConstraints gbc = new GridBagConstraints();

        // Labels and Text Fields
        JLabel studentIDLabel = new JLabel("Student ID:");
        JTextField studentIDField = new JTextField(20);
        JLabel nameLabel = new JLabel("Name:");
        JTextField nameField = new JTextField(20);
        JLabel batchLabel = new JLabel("Batch:");
        JTextField batchField = new JTextField(20);
        JLabel departmentLabel = new JLabel("Department:");
        JTextField departmentField = new JTextField(20);
        JLabel passwordLabel = new JLabel("Password:");
        JPasswordField passwordField = new JPasswordField(20);

        JButton registerButton = new JButton("Register");
        JLabel messageLabel = new JLabel("", JLabel.CENTER);
        JButton switchToLoginButton = new JButton("Go to Login");

        // Add components to the panel
        gbc.insets = new Insets(10, 10, 10, 10); // Padding
```

Registration-Panel Class

```
gbc.gridx = 0;
gbc.gridy = 0;
gbc.anchor = GridBagConstraints.LINE_END;
add(studentIDLabel, gbc);

gbc.gridx = 1;
gbc.anchor = GridBagConstraints.LINE_START;
add(studentIDField, gbc);

gbc.gridx = 0;
gbc.gridy = 1;
gbc.anchor = GridBagConstraints.LINE_END;
add(nameLabel, gbc);

gbc.gridx = 1;
gbc.anchor = GridBagConstraints.LINE_START;
add(nameField, gbc);

gbc.gridx = 0;
gbc.gridy = 2;
gbc.anchor = GridBagConstraints.LINE_END;
add(batchLabel, gbc);

gbc.gridx = 1;
gbc.anchor = GridBagConstraints.LINE_START;
add(batchField, gbc);

gbc.gridx = 0;
gbc.gridy = 3;
gbc.anchor = GridBagConstraints.LINE_END;
add(departmentLabel, gbc);
```

```
gbc.gridx = 1;
gbc.anchor = GridBagConstraints.LINE_ST
add(departmentField, gbc);

gbc.gridx = 0;
gbc.gridy = 4;
gbc.anchor = GridBagConstraints.LINE_EN
add(passwordLabel, gbc);

gbc.gridx = 1;
gbc.anchor = GridBagConstraints.LINE_ST
add(passwordField, gbc);

gbc.gridx = 0;
gbc.gridy = 5;
gbc.gridwidth = 2;
gbc.anchor = GridBagConstraints.CENTER;
add(registerButton, gbc);

gbc.gridy = 6;
add(messageLabel, gbc);

gbc.gridy = 7;
add(switchToLoginButton, gbc);
```

Registration-Panel Class

```
// Action Listener for Register Button
registerButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String studentID = studentIDField.getText();
        String name = nameField.getText();
        String batch = batchField.getText();
        String department = departmentField.getText();
        String password = new String(passwordField.getPassword());

        try {
            Student student = new Student(studentID, name, batch, department, password);

            if (student.validate()) {
                if (LibraryDiscussionRoomSystem.registeredStudents.containsKey(studentID)) {
                    throw new Exception("Student ID already registered.");
                }
                LibraryDiscussionRoomSystem.registeredStudents.put(studentID, student);
                messageLabel.setText("Registration Successful!");
                messageLabel.setForeground(Color.GREEN);
                CardLayout cl = (CardLayout) frame.getContentPane().getLayout();
                cl.show(frame.getContentPane(), "Login");
            } else {
                throw new Exception("Invalid input! Please check your data.");
            }
        } catch (Exception ex) {
            messageLabel.setText(ex.getMessage());
            messageLabel.setForeground(Color.RED);
        }
    }
});
```

Registration-Panel Class

```
        switchToLoginButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                CardLayout cl = (CardLayout) frame.getContentPane().getLayout();
                cl.show(frame.getContentPane(), "Login");
            }
        });
    }
}
```

Login-Panel Class

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
public class LoginPanel extends JPanel {
    public LoginPanel(JFrame frame) {
        setLayout(new GridBagLayout());
        GridBagConstraints gbc = new GridBagConstraints();

        JLabel studentIDLabel = new JLabel("Student ID:");
        JTextField studentIDField = new JTextField(20);
        JLabel passwordLabel = new JLabel("Password:");
        JPasswordField passwordField = new JPasswordField(20);

        JButton loginButton = new JButton("Login");
        JLabel messageLabel = new JLabel("", JLabel.CENTER);
        JButton switchToRegisterButton = new JButton("Go to Registration");
        // Add components to the panel
        gbc.insets = new Insets(10, 10, 10, 10); // Padding

        gbc.gridx = 0;
        gbc.gridy = 0;
        gbc.anchor = GridBagConstraints.LINE_END;
        add(studentIDLabel, gbc);

        gbc.gridx = 1;
        gbc.anchor = GridBagConstraints.LINE_START;
        add(studentIDField, gbc);
```

Login-Panel Class

```
gbc.gridx = 0;
gbc.gridy = 1;
gbc.anchor = GridBagConstraints.LINE_END;
add(passwordLabel, gbc);

gbc.gridx = 1;
gbc.anchor = GridBagConstraints.LINE_START;
add(passwordField, gbc);

gbc.gridx = 0;
gbc.gridy = 2;
gbc.gridwidth = 2;
gbc.anchor = GridBagConstraints.CENTER;
add(loginButton, gbc);

gbc.gridy = 3;
add(messageLabel, gbc);

gbc.gridy = 4;
add(switchToRegisterButton, gbc);
```

Login-Panel Class

```
// Action Listener for Login Button
loginButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String studentID = studentIDField.getText();
        String password = new String(passwordField.getPassword());

        try {
            if (!LibraryDiscussionRoomSystem.registeredStudents.containsKey(studentID)) {
                throw new Exception("Student ID not registered.");
            }
            Student registeredStudent = LibraryDiscussionRoomSystem.registeredStudents.get(studentID);
            if (registeredStudent.getPassword().equals(password)) {
                messageLabel.setText("Login Successful!");
                messageLabel.setForeground(Color.GREEN);

                // Show RoomPanel on successful login
                CardLayout cl = (CardLayout) frame.getContentPane().getLayout();
                cl.show(frame.getContentPane(), "RoomPanel");

            } else {
                throw new Exception("Invalid password.");
            }
        } catch (Exception ex) {
            messageLabel.setText(ex.getMessage());
            messageLabel.setForeground(Color.RED);
        }
    }
});
```

Login-Panel Class

```
        switchToRegisterButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                CardLayout cl = (CardLayout) frame.getContentPane().getLayout();
                cl.show(frame.getContentPane(), "Registration");
            }
        });
    }
}
```

Room-Panel Class

```
import javax.swing.*;
import java.awt.*;
import java.util.ArrayList;
import java.util.List;
public class RoomPanel extends JPanel {
    private List<Room> rooms;
    private List<JPanel> roomPanels;
    private List<JLabel> statusLabels;
    private List<JButton> bookButtons;
    private List<JButton> exitButtons;
    public RoomPanel() {
        rooms = new ArrayList<>();
        roomPanels = new ArrayList<>();
        statusLabels = new ArrayList<>();
        bookButtons = new ArrayList<>();
        exitButtons = new ArrayList<>();

        setLayout(new GridLayout(0, 1)); // Use a dynamic grid layout
        // Initial room setup
        for (int i = 0; i < 6; i++) {
            addRoom(new Room("Room " + (i + 1)));
        }
    }
```

Room-Panel Class

```
public void addRoom(Room room) {
    rooms.add(room);
    JPanel roomPanel = new JPanel(new BorderLayout());
    JLabel statusLabel = new JLabel("Status: " + room.getStatus());
    JButton bookButton = new JButton("Book Room");
    JButton exitButton = new JButton("Exit Room");

    bookButton.addActionListener(e -> {
        if (room.isBooked()) {
            JOptionPane.showMessageDialog(null, "Room already booked!", "Error", JOptionPane.ERROR_MESSAGE);
        } else {
            BookingForm bookingForm = new BookingForm(RoomPanel.this, room);
            bookingForm.setVisible(true);
        }
    });

    exitButton.addActionListener(e -> {
        if (!room.isBooked()) {
            JOptionPane.showMessageDialog(null, "Room is not booked yet!", "Error", JOptionPane.ERROR_MESSAGE);
        } else {
            ExitRoomPanel exitRoomPanel = new ExitRoomPanel(RoomPanel.this, room);
            exitRoomPanel.setVisible(true);
        }
    });
```

Room-Panel Class

```
        roomPanel.add(new JLabel(room.getName()), BorderLayout.NORTH);
        roomPanel.add(statusLabel, BorderLayout.CENTER);
        roomPanel.add(bookButton, BorderLayout.SOUTH);
        roomPanel.add(exitButton, BorderLayout.EAST);

        roomPanels.add(roomPanel);
        statusLabels.add(statusLabel);
        bookButtons.add(bookButton);
        exitButtons.add(exitButton);

        add(roomPanel);
        revalidate();
        repaint();
    }
    public void deleteRoom(int roomId) {
        if (roomId >= 0 && roomId < rooms.size()) {
            remove(roomPanels.get(roomId));
            rooms.remove(roomId);
            roomPanels.remove(roomId);
            statusLabels.remove(roomId);
            bookButtons.remove(roomId);
            exitButtons.remove(roomId);

            revalidate();
            repaint();
        } else {
            JOptionPane.showMessageDialog(this, "No such room exists", "Error", JOptionPane.ERROR_MESSAGE);
        }
    }
```

Room-Panel Class

```
public void updateRoomStatus(Room room) {
    int index = rooms.indexOf(room);
    if (index != -1) {
        statusLabels.get(index).setText("Status: " + room.getStatus());
    }
}
public List<Room> getRooms() {
    return rooms;
}
public void changeRoomStatus(int roomId, String newStatus) {
    if (roomId >= 0 && roomId < rooms.size()) {
        Room room = rooms.get(roomId);
        if (newStatus.equalsIgnoreCase("booked")) {
            room.book("Admin", "End Time", "1234");
        } else if (newStatus.equalsIgnoreCase("available")) {
            room.exit();
        } else {
            JOptionPane.showMessageDialog(this, "Invalid status", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }
        updateRoomStatus(room);
    } else {
        JOptionPane.showMessageDialog(this, "No such room exists", "Error", JOptionPane.ERROR_MESSAGE);
    }
}
```

Booking Form Class

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
public class BookingForm extends JFrame {
    private JTextField studentIdField;
    private JTextField nameField;
    private JTextField departmentField;
    private JTextField batchField;
    private JTextField endTimeHoursField;
    private JTextField endTimeMinutesField;
    private RoomPanel roomPanel;
    private Room room;
```

Booking Form Class

```
public BookingForm(RoomPanel roomPanel, Room room) {
    this.roomPanel = roomPanel;
    this.room = room;

    setLayout(new GridLayout(7, 2));

    add(new JLabel("Student ID:"));
    studentIdField = new JTextField();
    add(studentIdField);

    add(new JLabel("Name:"));
    nameField = new JTextField();
    add(nameField);

    add(new JLabel("Department:"));
    departmentField = new JTextField();
    add(departmentField);

    add(new JLabel("Batch:"));
    batchField = new JTextField();
    add(batchField);

    add(new JLabel("End Time Hours:"));
    endTimeHoursField = new JTextField();
    add(endTimeHoursField);

    add(new JLabel("End Time Minutes:"));
    endTimeMinutesField = new JTextField();
    add(endTimeMinutesField);
```

Booking Form Class

```
    JButton confirmButton = new JButton("Confirm Booking");
    confirmButton.addActionListener(new ActionListener() {
        @Override
        public void actionPerformed(ActionEvent e) {
            if (validateFields()) {
                room.book(nameField.getText(), endTimeHoursField.getText() + ":" + endTimeMinutesField.getText(), studentIdFiel
                roomPanel.updateRoomStatus(room);
                dispose();
            }
        }
    });
    add(confirmButton);

    setSize(400, 300);
}
```

Booking Form Class

```
private boolean validateFields() {
    String studentId = studentIdField.getText().trim();
    if (!studentId.matches("^[fF]\\d{8}$")) {
        JOptionPane.showMessageDialog(this, "Invalid Student ID. Must be 9 characters long starting with 'f' followed by digits.", "Val
        JOptionPane.ERROR_MESSAGE);
        return false;
    }

    if (nameField.getText().trim().isEmpty() || departmentField.getText().trim().isEmpty() ||
        batchField.getText().trim().isEmpty() || endTimeHoursField.getText().trim().isEmpty() ||
        endTimeMinutesField.getText().trim().isEmpty()) {
        JOptionPane.showMessageDialog(this, "All fields must be filled out.", "Validation Error", JOptionPane.ERROR_MESSAGE);
        return false;
    }

    return true;
}
}
```

Admin-Login Class

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class AdminLoginPanel extends JFrame {
    private JTextField adminIdField;
    private JPasswordField passwordField;
    private static final String ADMIN_ID = "admin123";
    private static final String ADMIN_PASSWORD = "oop123";

    public AdminLoginPanel() {
        setTitle("Admin Login");
        setSize(300, 200);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new GridLayout(3, 2));

        JLabel adminIdLabel = new JLabel("Admin ID");
        adminIdField = new JTextField();

        JLabel passwordLabel = new JLabel("Password");
        passwordField = new JPasswordField();

        JButton loginButton = new JButton("Login");
        loginButton.addActionListener(new LoginButtonListener());

        add(adminIdLabel);
        add(adminIdField);
        add(passwordLabel);
        add(passwordField);
        add(loginButton);
    }
```

Admin-Login Class

```
}

private class LoginButtonListener implements ActionListener {
    @Override
    public void actionPerformed(ActionEvent e) {
        String adminId = adminIdField.getText();
        String password = new String(passwordField.getPassword());

        if (ADMIN_ID.equals(adminId) && ADMIN_PASSWORD.equals(password)) {
            AdminPanel adminPanel = new AdminPanel();
            adminPanel.setVisible(true);
            dispose();
        } else {
            JOptionPane.showMessageDialog(AdminLoginPanel.this, "Invalid admin ID or password", "Error", JOptionPane.ERROR_MESSAGE);
        }
    }
}

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

Update-Room-Panel Class

```
import javax.swing.*;
import java.awt.*;
public class UpdateRoomPanel extends JPanel {
    private JTextField roomIdField;
    private JTextField statusField;
    private RoomPanel roomPanel;
    public UpdateRoomPanel(RoomPanel roomPanel) {
        this.roomPanel = roomPanel;
        setLayout(new GridLayout(3, 2));

        add(new JLabel("Room ID:"));
        roomIdField = new JTextField();
        add(roomIdField);

        add(new JLabel("New Status:"));
        statusField = new JTextField();
        add(statusField);

        JButton updateButton = new JButton("Update Status");
        updateButton.addActionListener(e -> updateRoomStatus());
        add(updateButton);
    }
```

Update-Room-Panel Class

```
private void updateRoomStatus() {
    try {
        int roomId = Integer.parseInt(roomIdField.getText().trim()) - 1;
        String newStatus = statusField.getText().trim();
        if (!newStatus.isEmpty()) {
            roomPanel.changeRoomStatus(roomId, newStatus);
            JOptionPane.showMessageDialog(this, "Room status updated successfully!");
        } else {
            JOptionPane.showMessageDialog(this, "Status cannot be empty.", "Error", JOptionPane.ERROR_MESSAGE);
        }
    } catch (NumberFormatException | IndexOutOfBoundsException e) {
        JOptionPane.showMessageDialog(this, "Invalid room ID.", "Error", JOptionPane.ERROR_MESSAGE);
    }
}
}
```

Library Discussion Class

```java
import javax.swing.*;
import java.awt.*;
import java.util.HashMap;
import java.util.Map;

public class LibraryDiscussionRoomSystem {
    static Map<String, Student> registeredStudents = new HashMap<>();

    public static void main(String[] args) {
        JFrame frame = new JFrame("Library Discussion Room System");
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.setSize(660, 500);
        frame.setLayout(new CardLayout());

        RegistrationPanel registrationPanel = new RegistrationPanel(frame);
        LoginPanel loginPanel = new LoginPanel(frame);
        RoomPanel roomPanel = new RoomPanel();

        frame.add(registrationPanel, "Registration");
        frame.add(loginPanel, "Login");
        frame.add(roomPanel, "RoomPanel");

        frame.setVisible(true);
    }
}
```

Admin Panel Class

```
import javax.swing.*;
import java.awt.*;
public class AdminPanel extends JFrame {
    private RoomPanel roomPanel;
    public AdminPanel() {
        setTitle("Admin Panel");
        setSize(600, 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLayout(new BorderLayout());

        roomPanel = new RoomPanel();

        JButton addRoomButton = new JButton("Add Room");
        addRoomButton.addActionListener(e -> {
            AddRoomPanel addRoomPanel = new AddRoomPanel(roomPanel);
            JFrame addRoomFrame = new JFrame("Add Room");
            addRoomFrame.setContentPane(addRoomPanel);
            addRoomFrame.pack();
            addRoomFrame.setVisible(true);
        });

        JButton deleteRoomButton = new JButton("Delete Room");
        deleteRoomButton.addActionListener(e -> {
            DeleteRoomPanel deleteRoomPanel = new DeleteRoomPanel(roomPanel);
            JFrame deleteRoomFrame = new JFrame("Delete Room");
            deleteRoomFrame.setContentPane(deleteRoomPanel);
            deleteRoomFrame.pack();
            deleteRoomFrame.setVisible(true);
        });
```

Admin Panel Class

```
        JButton updateRoomStatusButton = new JButton("Change Room Status");
        updateRoomStatusButton.addActionListener(e -> {
            UpdateRoomPanel updateRoomPanel = new UpdateRoomPanel(roomPanel);
            JFrame updateRoomFrame = new JFrame("Change Room Status");
            updateRoomFrame.setContentPane(updateRoomPanel);
            updateRoomFrame.pack();
            updateRoomFrame.setVisible(true);
        });

        JPanel controlPanel = new JPanel();
        controlPanel.add(addRoomButton);
        controlPanel.add(deleteRoomButton);
        controlPanel.add(updateRoomStatusButton);

        add(controlPanel, BorderLayout.NORTH);
        add(new JScrollPane(roomPanel), BorderLayout.CENTER);
    }

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

Delete Room Class

```
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class DeleteRoomPanel extends JPanel {
    private JTextField roomIdField;
    private RoomPanel roomPanel;

    public DeleteRoomPanel(RoomPanel roomPanel) {
        this.roomPanel = roomPanel;
        setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));

        roomIdField = new JTextField(20);

        add(new JLabel("Room ID:"));
        add(roomIdField);

        JButton deleteButton = new JButton("Delete Room");
        deleteButton.addActionListener(e -> {
            try {
                int roomId = Integer.parseInt(roomIdField.getText()) - 1;
                roomPanel.deleteRoom(roomId);
                JOptionPane.showMessageDialog(DeleteRoomPanel.this, "Room deleted!");
            } catch (NumberFormatException ex) {
                JOptionPane.showMessageDialog(DeleteRoomPanel.this, "Invalid room ID", "Error", JOptionPane.ERROR_MESSAGE);
            }
        });
        add(deleteButton);
    }
}
```

Add Room Class

```java
import javax.swing.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

public class AddRoomPanel extends JPanel {
    private JTextField roomNameField;
    private RoomPanel roomPanel;

    public AddRoomPanel(RoomPanel roomPanel) {
        this.roomPanel = roomPanel;
        setLayout(new BoxLayout(this, BoxLayout.Y_AXIS));

        roomNameField = new JTextField(20);

        add(new JLabel("Room Name:"));
        add(roomNameField);

        JButton addButton = new JButton("Add Room");
        addButton.addActionListener(e -> {
            String roomName = roomNameField.getText();
            if (!roomName.isEmpty()) {
                roomPanel.addRoom(new Room(roomName));
                JOptionPane.showMessageDialog(AddRoomPanel.this, "Room added!");
            } else {
                JOptionPane.showMessageDialog(AddRoomPanel.this, "Room name cannot be empty", "Error", JOptionPane.ERROR_MESSAGE);
            }
        });
        add(addButton);
    }
}
```

Login Panel Class

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
public class LoginPanel extends JPanel {
    public LoginPanel(JFrame frame) {
        setLayout(new GridBagLayout());
        GridBagConstraints gbc = new GridBagConstraints();

        JLabel studentIDLabel = new JLabel("Student ID:");
        JTextField studentIDField = new JTextField(20);
        JLabel passwordLabel = new JLabel("Password:");
        JPasswordField passwordField = new JPasswordField(20);

        JButton loginButton = new JButton("Login");
        JLabel messageLabel = new JLabel("", JLabel.CENTER);
        JButton switchToRegisterButton = new JButton("Go to Registration");
        // Add components to the panel
        gbc.insets = new Insets(10, 10, 10, 10); // Padding

        gbc.gridx = 0;
        gbc.gridy = 0;
        gbc.anchor = GridBagConstraints.LINE_END;
        add(studentIDLabel, gbc);

        gbc.gridx = 1;
        gbc.anchor = GridBagConstraints.LINE_START;
        add(studentIDField, gbc);
```

Login Panel Class

```
gbc.gridx = 0;
gbc.gridy = 1;
gbc.anchor = GridBagConstraints.LINE_END;
add(passwordLabel, gbc);

gbc.gridx = 1;
gbc.anchor = GridBagConstraints.LINE_START;
add(passwordField, gbc);

gbc.gridx = 0;
gbc.gridy = 2;
gbc.gridwidth = 2;
gbc.anchor = GridBagConstraints.CENTER;
add(loginButton, gbc);

gbc.gridy = 3;
add(messageLabel, gbc);

gbc.gridy = 4;
add(switchToRegisterButton, gbc);
```

Login Panel Class

```
// Action Listener for Login Button
loginButton.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        String studentID = studentIDField.getText();
        String password = new String(passwordField.getPassword());

        try {
            if (!LibraryDiscussionRoomSystem.registeredStudents.containsKey(studentID)) {
                throw new Exception("Student ID not registered.");
            }
            Student registeredStudent = LibraryDiscussionRoomSystem.registeredStudents.get(studentID);
            if (registeredStudent.getPassword().equals(password)) {
                messageLabel.setText("Login Successful!");
                messageLabel.setForeground(Color.GREEN);

                // Show RoomPanel on successful login
                CardLayout cl = (CardLayout) frame.getContentPane().getLayout();
                cl.show(frame.getContentPane(), "RoomPanel");

            } else {
                throw new Exception("Invalid password.");
            }
        } catch (Exception ex) {
            messageLabel.setText(ex.getMessage());
            messageLabel.setForeground(Color.RED);
        }
    }
});
```

Login Panel Class

```
        switchToRegisterButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                CardLayout cl = (CardLayout) frame.getContentPane().getLayout();
                cl.show(frame.getContentPane(), "Registration");
            }
        });
    }
}
```

DEMONSTRATION OF CODE

THANK YOU