REVISED CASE STUDY

```
import javax.swing.*;
import java.awt.*;
import java.awt.event.*;
import java.text.ParseException;
import java.text.SimpleDateFormat;
import java.util.ArrayList;
import java.util.Calendar;
import java.util.Date;
import javax.swing.event.DocumentEvent;
import javax.swing.event.DocumentListener;


public class HotelStarPlatinumCase extends JFrame {

public HotelStarPlatinumCase() {
// Set JFrame properties for the main hotel window
setTitle("Star Platinum Hotel");
setSize(900, 500);
setDefaultCloseOperation(EXIT_ON_CLOSE);
setLocationRelativeTo(null);
setLayout(null);

// Welcome label
JLabel helloLabel = new JLabel("WELCOME TO STAR PLATINUM HOTEL");
helloLabel.setBounds(300, 100, 900, 30);
helloLabel.setFont(new Font("Arial", Font.BOLD, 18));

// Button to open the login page
JButton navigateButton = new JButton("CLICK HERE TO START BOOKING");
navigateButton.setBounds(300, 350, 300, 50);

// Add action listener to button to open login page
navigateButton.addActionListener(new ActionListener() {
@Override
public void actionPerformed(ActionEvent e) {
new LoginPage().setVisible(true);
dispose(); // Close the current window
}
});
```

```
// Add components to JFrame
add(navigateButton);
add(helloLabel);
}

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

class LoginPage extends JFrame {

public LoginPage() {
// Set up the LoginPage window
setTitle("Login Page");
setSize(400, 250);
setDefaultCloseOperation(EXIT_ON_CLOSE);
setLocationRelativeTo(null);
setLayout(new GridBagLayout());

// Create components for email and password
JLabel emailLabel = new JLabel("Enter your Gmail:");
JTextField emailField = new JTextField(15);

JLabel passwordLabel = new JLabel("Password:");
JPasswordField passwordField = new JPasswordField(15);

JButton loginButton = new JButton("Login");
JButton clearButton = new JButton("Clear");

// Set up layout constraints
GridBagConstraints gbc = new GridBagConstraints();
gbc.insets = new Insets(5, 5, 5, 5);

// Add components to the login form
gbc.gridx = 0;
gbc.gridy = 0;
add(emailLabel, gbc);

gbc.gridx = 1;
gbc.gridy = 0;
```

```
add(emailField, gbc);

gbc.gridx = 0;
gbc.gridy = 1;
add(passwordLabel, gbc);

gbc.gridx = 1;
gbc.gridy = 1;
add(passwordField, gbc);

gbc.gridx = 0;
gbc.gridy = 2;
add(loginButton, gbc);

gbc.gridx = 1;
gbc.gridy = 2;
add(clearButton, gbc);

// Action listener for login button
loginButton.addActionListener(e -> handleLogin(emailField, passwordField));

// Action listener for clear button
clearButton.addActionListener(e -> handleClear(emailField, passwordField));
}

private void handleLogin(JTextField emailField, JPasswordField passwordField) {
String email = emailField.getText();
String password = new String(passwordField.getPassword());

// Regular expression to validate Gmail address
String gmailRegex = "^[a-zA-Z0-9._%+-]+@gmail\\.com$";

// Check if email matches the Gmail pattern and password is correct
if (email.matches(gmailRegex) && password.equals("7777777")) {
// Successful login
JOptionPane.showMessageDialog(this, "Login Successful!", "Success",
JOptionPane.INFORMATION_MESSAGE);

// Open the second page and close the current one
new SecondPageCase().setVisible(true);
dispose();
} else if (!email.matches(gmailRegex)) {
```

```
// Invalid email
JOptionPane.showMessageDialog(this, "Invalid Gmail address. Please use a valid Gmail account.",
"Error", JOptionPane.ERROR_MESSAGE);
} else if (!password.equals("7777777")) {
// Invalid password
JOptionPane.showMessageDialog(this, "Incorrect password. Please try again.", "Error",
JOptionPane.ERROR_MESSAGE);
}
}

private void handleClear(JTextField emailField, JPasswordField passwordField) {
emailField.setText("");
passwordField.setText("");
}
}

class SecondPageCase extends JFrame {

private ButtonGroup localDestGroup, internationalDestGroup;

public SecondPageCase() {
// Set JFrame properties for the second page
setTitle("Second Page - Destinations");
setSize(900, 500);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setLocationRelativeTo(null);
setLayout(null);

// Header Label
JLabel destinationsLabel = new JLabel("Destinations available");
destinationsLabel.setFont(new Font("Segoe UI", Font.BOLD, 18));
destinationsLabel.setBounds(280, 15, 300, 30);
add(destinationsLabel);

// Section - Local Destinations
JLabel localDestinationsLabel = new JLabel("LOCAL DESTINATIONS");
localDestinationsLabel.setFont(new Font("Segoe UI", Font.BOLD, 12));
localDestinationsLabel.setBounds(130, 60, 200, 25);
add(localDestinationsLabel);

// Section - International Destinations
JLabel internationalDestinationsLabel = new JLabel("INTERNATIONAL DESTINATIONS");
```

```
internationalDestinationsLabel.setFont(new Font("Segoe UI", Font.BOLD, 12));
internationalDestinationsLabel.setBounds(470, 60, 250, 25);
add(internationalDestinationsLabel);

// Radio buttons for Local Destinations
localDestGroup = new ButtonGroup();
JRadioButton baguioButton = createRadioButton("Baguio", 150, 100);
JRadioButton boracayButton = createRadioButton("Boracay", 150, 150);
JRadioButton elNidoButton = createRadioButton("El Nido", 150, 200);
JRadioButton siargaoButton = createRadioButton("Siargao", 150, 250);

localDestGroup.add(baguioButton);
localDestGroup.add(boracayButton);
localDestGroup.add(elNidoButton);
localDestGroup.add(siargaoButton);

// Radio buttons for International Destinations
internationalDestGroup = new ButtonGroup();
JRadioButton hongKongButton = createRadioButton("Hong Kong", 500, 100);
JRadioButton japanButton = createRadioButton("Japan", 500, 150);
JRadioButton singaporeButton = createRadioButton("Singapore", 500, 200);
JRadioButton southKoreaButton = createRadioButton("South Korea", 500, 250);

internationalDestGroup.add(hongKongButton);
internationalDestGroup.add(japanButton);
internationalDestGroup.add(singaporeButton);
internationalDestGroup.add(southKoreaButton);

// Placeholder for additional information
JLabel additionalInfo = new JLabel("Discover the Philippines in these locations");
additionalInfo.setBounds(50, 400, 400, 30);
add(additionalInfo);

JLabel anotherInfo = new JLabel("Discover the rest of Asia in these locations");
anotherInfo.setBounds(450, 400, 300, 30);
add(anotherInfo);

// Confirm Button
JButton confirmButton = new JButton("CONFIRM");
confirmButton.setBounds(350, 370, 200, 40);
confirmButton.addActionListener(e -> validateSelection()); // Validation logic
add(confirmButton);
```

```
}

private JRadioButton createRadioButton(String name, int x, int y) {
JRadioButton radioButton = new JRadioButton(name);
radioButton.setBounds(x, y, 100, 30);
radioButton.setActionCommand(name); // Set action command to the name
add(radioButton);
return radioButton;
}

private void validateSelection() {
String validationMessage = "";

// Check if a local destination is selected
if (localDestGroup.getSelection() != null) {
String selectedLocal = localDestGroup.getSelection().getActionCommand();
validationMessage = "Local destination chosen: " + selectedLocal;
}
// Check if an international destination is selected
else if (internationalDestGroup.getSelection() != null) {
String selectedInternational = internationalDestGroup.getSelection().getActionCommand();
validationMessage = "International destination chosen: " + selectedInternational;
}
else {
validationMessage = "Please select a destination!";
}

// Show validation message in a JOptionPane
JOptionPane.showMessageDialog(this, validationMessage);

// After confirmation, navigate to the third page
if (!validationMessage.equals("Please select a destination!")) {
navigateToThirdPage();
}
}

private void navigateToThirdPage() {
// Assuming you have a ThirdPage class in your project
ThirdPage thirdPage = new ThirdPage();
thirdPage.setVisible(true); // Open ThirdPage
dispose(); // Close current window
}
```

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

class ThirdPage extends JFrame {
private JTextField checkInTextField, checkOutTextField, numAdultsTextField, numChildrenTextField;
private JButton confirmButton;
private JPanel childAgePanel;
private final SimpleDateFormat dateFormat = new SimpleDateFormat("MM/dd/yyyy");

private ArrayList<JTextField> childAgeFields;

public ThirdPage() {
initializeUI();
}

private void initializeUI() {
setTitle("Third Page - Booking Details");
setSize(900, 700); // Updated JFrame size
setLocationRelativeTo(null);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setLayout(new BorderLayout());

childAgeFields = new ArrayList<>();

// Create Main Panel
JPanel mainPanel = new JPanel();
mainPanel.setLayout(new GridBagLayout());
GridBagConstraints gbc = new GridBagConstraints();
gbc.insets = new Insets(5, 5, 5, 5);

// Create and add Check-in Date field
JLabel checkInLabel = new JLabel("Check-in Date (MM/dd/yyyy):");
gbc.gridx = 0;
gbc.gridy = 0;
mainPanel.add(checkInLabel, gbc);

checkInTextField = new JTextField(10);
gbc.gridx = 1;
mainPanel.add(checkInTextField, gbc);
```

```
// Create and add Check-out Date field
JLabel checkOutLabel = new JLabel("Check-out Date (MM/dd/yyyy):");
gbc.gridx = 0;
gbc.gridy = 1;
mainPanel.add(checkOutLabel, gbc);

checkOutTextField = new JTextField(10);
gbc.gridx = 1;
mainPanel.add(checkOutTextField, gbc);

// Create and add Number of Adults field
JLabel numAdultsLabel = new JLabel("Number of Adults:");
gbc.gridx = 0;
gbc.gridy = 2;
mainPanel.add(numAdultsLabel, gbc);

numAdultsTextField = new JTextField(5);
gbc.gridx = 1;
mainPanel.add(numAdultsTextField, gbc);

// Create and add Number of Children field
JLabel numChildrenLabel = new JLabel("Number of Children:");
gbc.gridx = 0;
gbc.gridy = 3;
mainPanel.add(numChildrenLabel, gbc);

numChildrenTextField = new JTextField(5);
numChildrenTextField.getDocument().addDocumentListener(new DocumentListener() {
@Override
public void insertUpdate(DocumentEvent e) {
handleChildGuestNumber();
}

@Override
public void removeUpdate(DocumentEvent e) {
handleChildGuestNumber();
}

@Override
public void changedUpdate(DocumentEvent e) {
handleChildGuestNumber();
```

```
}
});
gbc.gridx = 1;
mainPanel.add(numChildrenTextField, gbc);

// Create and add Age of Child/ren label
JLabel ageOfChildLabel = new JLabel("Age of Child/ren:");
gbc.gridx = 0;
gbc.gridy = 4;
mainPanel.add(ageOfChildLabel, gbc);

// Dynamic child age input fields
childAgePanel = new JPanel();
childAgePanel.setLayout(new GridLayout(0, 2, 5, 5)); // 2 columns for labels and text fields
JScrollPane scrollPane = new JScrollPane(childAgePanel);
scrollPane.setPreferredSize(new Dimension(400, 200)); // Set scrollable area size
gbc.gridx = 1;
gbc.gridy = 4;
gbc.gridwidth = 2;
mainPanel.add(scrollPane, gbc);

// Confirm button to process booking
confirmButton = new JButton("Confirm Booking");
confirmButton.addActionListener(e -> handleConfirmation());
gbc.gridx = 0;
gbc.gridy = 5;
gbc.gridwidth = 2;
mainPanel.add(confirmButton, gbc);

add(mainPanel, BorderLayout.CENTER);
pack();
}

private void handleChildGuestNumber() {
childAgePanel.removeAll();
childAgeFields.clear();

try {
int numChildren = Integer.parseInt(numChildrenTextField.getText().trim());
if (numChildren < 0) {
JOptionPane.showMessageDialog(this, "Number of children cannot be negative.", "Error",
JOptionPane.ERROR_MESSAGE);
```

```
numChildrenTextField.setText("");
return;
}

for (int i = 0; i < numChildren; i++) {
JLabel childLabel = new JLabel("Age of child #" + (i + 1) + ":");
JTextField ageField = new JTextField(5);
childAgePanel.add(childLabel);
childAgePanel.add(ageField);
childAgeFields.add(ageField);
}
} catch (NumberFormatException ex) {
childAgePanel.revalidate();
childAgePanel.repaint();
}

childAgePanel.revalidate();
childAgePanel.repaint();
}

private String getSeason(Date checkInDate) {
// Normalize the check-in date to ignore the year
Calendar checkInCalendar = Calendar.getInstance();
checkInCalendar.setTime(checkInDate);
int checkInMonth = checkInCalendar.get(Calendar.MONTH);
int checkInDay = checkInCalendar.get(Calendar.DAY_OF_MONTH);

// Lean Season: June 1 to October 31
Calendar leanStart = Calendar.getInstance();
leanStart.set(Calendar.MONTH, Calendar.JUNE);
leanStart.set(Calendar.DAY_OF_MONTH, 1);

Calendar leanEnd = Calendar.getInstance();
leanEnd.set(Calendar.MONTH, Calendar.OCTOBER);
leanEnd.set(Calendar.DAY_OF_MONTH, 31);

// High Season: November 1 to December 19, January 6 to February 28
Calendar highStart1 = Calendar.getInstance();
highStart1.set(Calendar.MONTH, Calendar.NOVEMBER);
highStart1.set(Calendar.DAY_OF_MONTH, 1);

Calendar highEnd1 = Calendar.getInstance();
```

```
highEnd1.set(Calendar.MONTH, Calendar.DECEMBER);
highEnd1.set(Calendar.DAY_OF_MONTH, 19);

Calendar highStart2 = Calendar.getInstance();
highStart2.set(Calendar.MONTH, Calendar.JANUARY);
highStart2.set(Calendar.DAY_OF_MONTH, 6);

Calendar highEnd2 = Calendar.getInstance();
highEnd2.set(Calendar.MONTH, Calendar.FEBRUARY);
highEnd2.set(Calendar.DAY_OF_MONTH, 28);

// Peak Season: March 1 to May 31
Calendar peakStart = Calendar.getInstance();
peakStart.set(Calendar.MONTH, Calendar.MARCH);
peakStart.set(Calendar.DAY_OF_MONTH, 1);

Calendar peakEnd = Calendar.getInstance();
peakEnd.set(Calendar.MONTH, Calendar.MAY);
peakEnd.set(Calendar.DAY_OF_MONTH, 31);

// Super Peak Season: December 20 to January 5
Calendar superPeakStart = Calendar.getInstance();
superPeakStart.set(Calendar.MONTH, Calendar.DECEMBER);
superPeakStart.set(Calendar.DAY_OF_MONTH, 20);

Calendar superPeakEnd = Calendar.getInstance();
superPeakEnd.set(Calendar.MONTH, Calendar.JANUARY);
superPeakEnd.set(Calendar.DAY_OF_MONTH, 5);

// Check for Lean Season
if ((checkInMonth == leanStart.get(Calendar.MONTH) && checkInDay >=
leanStart.get(Calendar.DAY_OF_MONTH)) ||
(checkInMonth == leanEnd.get(Calendar.MONTH) && checkInDay <=
leanEnd.get(Calendar.DAY_OF_MONTH)) ||
(checkInMonth > leanStart.get(Calendar.MONTH) && checkInMonth < leanEnd.get(Calendar.MONTH))) {
return "Lean";
}

// Check for High Season
if ((checkInMonth == highStart1.get(Calendar.MONTH) && checkInDay >=
highStart1.get(Calendar.DAY_OF_MONTH)) ||
(checkInMonth == highEnd1.get(Calendar.MONTH) && checkInDay <=
```

```
highEnd1.get(Calendar.DAY_OF_MONTH)) ||
(checkInMonth == highStart2.get(Calendar.MONTH) && checkInDay >=
highStart2.get(Calendar.DAY_OF_MONTH)) ||
(checkInMonth == highEnd2.get(Calendar.MONTH) && checkInDay <=
highEnd2.get(Calendar.DAY_OF_MONTH)) ||
(checkInMonth > highStart1.get(Calendar.MONTH) && checkInMonth < highEnd1.get(Calendar.MONTH))
||
(checkInMonth > highStart2.get(Calendar.MONTH) && checkInMonth <
highEnd2.get(Calendar.MONTH))) {
return "High";
}

// Check for Peak Season
if ((checkInMonth == peakStart.get(Calendar.MONTH) && checkInDay >=
peakStart.get(Calendar.DAY_OF_MONTH)) ||
(checkInMonth == peakEnd.get(Calendar.MONTH) && checkInDay <=
peakEnd.get(Calendar.DAY_OF_MONTH)) ||
(checkInMonth > peakStart.get(Calendar.MONTH) && checkInMonth < peakEnd.get(Calendar.MONTH)))
{
return "Peak";
}

// Check for Super Peak Season
if ((checkInMonth == superPeakStart.get(Calendar.MONTH) && checkInDay >=
superPeakStart.get(Calendar.DAY_OF_MONTH)) ||
(checkInMonth == superPeakEnd.get(Calendar.MONTH) && checkInDay <=
superPeakEnd.get(Calendar.DAY_OF_MONTH)) ||
(checkInMonth > superPeakStart.get(Calendar.MONTH) && checkInMonth <
superPeakEnd.get(Calendar.MONTH))) {
return "Super Peak";
}

return "Unknown"; // If it doesn't fall within any season, return "Unknown"
}

private void handleConfirmation() {
try {
String checkInStr = checkInTextField.getText().trim();
String checkOutStr = checkOutTextField.getText().trim();
Date checkInDate = dateFormat.parse(checkInStr);
Date checkOutDate = dateFormat.parse(checkOutStr);
```

```
if (checkInDate.before(new Date())) {
JOptionPane.showMessageDialog(this, "Check-in date cannot be in the past.", "Error",
JOptionPane.ERROR_MESSAGE);
return;
}

if (!checkOutDate.after(checkInDate)) {
JOptionPane.showMessageDialog(this, "Check-out date must be after the check-in date.", "Error",
JOptionPane.ERROR_MESSAGE);
return;
}

int numAdults = Integer.parseInt(numAdultsTextField.getText().trim());
if (numAdults < 0) {
JOptionPane.showMessageDialog(this, "Number of adults cannot be negative.", "Error",
JOptionPane.ERROR_MESSAGE);
return;
}

for (JTextField childField : childAgeFields) {
int age = Integer.parseInt(childField.getText().trim());
if (age < 0) {
JOptionPane.showMessageDialog(this, "Child age cannot be negative.", "Error",
JOptionPane.ERROR_MESSAGE);
return;
}
}

// Get season based on check-in date
String season = getSeason(checkInDate);
JOptionPane.showMessageDialog(this, "Booking Confirmed!\nSeason: " + season + "\nAdults: " +
numAdults + "\nChildren: " + childAgeFields.size(), "Success", JOptionPane.INFORMATION_MESSAGE);

// Navigate to FourthPage
new FourthPage().setVisible(true);
dispose(); // Close ThirdPage

} catch (ParseException e) {
JOptionPane.showMessageDialog(this, "Please use the correct date format (MM/dd/yyyy).", "Error",
JOptionPane.ERROR_MESSAGE);
} catch (NumberFormatException e) {
JOptionPane.showMessageDialog(this, "Please enter valid numbers.", "Error",
```

```
JOptionPane.ERROR_MESSAGE);
}
}

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

class FourthPage extends JFrame {

private JButton jButton1;
private JButton jButton2;
private JButton jButton3;
private JButton jButton4;
private JButton jButton5;
private JButton jButton6;
private JLabel jLabel1;
private JLabel jLabel2;
private JLabel jLabel3;
private JLabel jLabel5;
private JLabel jLabel6;
private JLabel jLabel7;
private JLabel jLabel8;
private JLabel jLabel9;
private JLabel jLabe20;
private JLabel jLabe21;
private JLabel jLabe22;
private JLabel jLabe23;
private JLabel jLabe24;
private JLabel jLabe25;
private JLabel jLabe26;
private JLabel jLabe27;
private JLabel jLabe28;
private JLabel jLabe29;
private JLabel jLabe30;
private JLabel jLabe31;
private JLabel jLabe32;
private JLabel jLabe33;
private JLabel jLabe34;
private JLabel jLabe35;
private JLabel jLabe36;
```

```
private JLabel jLabe37;
private JLabel jLabe38;
private JLabel jLabe39;
private JLabel jLabe40;
private JLabel jLabe41;
private JLabel jLabe42;
private JLabel jLabe43;
private JLabel jLabe44;
private JLabel jLabe45;
private JLabel jLabe46;
private JLabel jLabe47;
private JLabel jLabe48;
private JLabel jLabe49;
private JLabel jLabe50;
private JLabel jLabe51;
private JLabel jLabe52;
private JLabel jLabe53;
private JLabel jLabe54;
private JLabel jLabe55;
private JLabel jLabe56;
private JLabel jLabe57;
private JLabel jLabe58;
private JLabel jLabe59;
private JLabel jLabe60;
private JLabel jLabe61;
private JLabel jLabe62;
private JLabel jLabe63;
private JLabel jLabe64;
private JLabel jLabe65;
private JLabel jLabe66;
private JLabel jLabe67;
private JLabel jLabe68;
private JLabel jLabe69;
private JLabel jLabe70;
private JLabel jLabe71;
private JLabel jLabe72;
private JLabel jLabe73;
private JLabel jLabe74;
private JLabel jLabe75;


public FourthPage() {
```

```
initComponents();
}

private void initComponents() {
// Set up JFrame
setSize(1200, 700);
setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
setLayout(null); // Use absolute positioning

// Initialize Buttons
jButton1 = new JButton("Standard");
jButton2 = new JButton("Deluxe");
jButton3 = new JButton("Quadruple");
jButton4 = new JButton("Family");
jButton5 = new JButton("Suite");
jButton6 = new JButton("Suggested Room");

// Set positions and sizes for buttons
jButton1.setBounds(40, 100, 120, 40);
jButton2.setBounds(40, 170, 120, 40);
jButton3.setBounds(40, 240, 120, 40);
jButton4.setBounds(40, 310, 120, 40);
jButton5.setBounds(40, 380, 120, 40);
jButton6.setBounds(40, 450, 150, 40);

// Initialize Labels
jLabel1 = new JLabel("Room Type");
jLabel2 = new JLabel("Capacity");
jLabel3 = new JLabel("Available Rooms");

jLabel5 = new JLabel("1 guestw/extra bed");
jLabel6 = new JLabel("2 guestsw/extra bed");
jLabel7 = new JLabel("4 guestsw/extra bed");
jLabel8 = new JLabel("6 guestsw/extra bed");
jLabel9 = new JLabel("4 guests");
jLabe20 = new JLabel("5");
jLabe21 = new JLabel("4");
jLabe22 = new JLabel("5");
jLabe23 = new JLabel("3");
jLabe24 = new JLabel("2");
jLabe25 = new JLabel("Rate Per Night");
jLabe26 = new JLabel("Lean");
```

```
jLabe27 = new JLabel("P 2,000");
jLabe28 = new JLabel("P 3,000");
jLabe29 = new JLabel("P 4,000");
jLabe30 = new JLabel("P 5,000");
jLabe31 = new JLabel("P 6,000");
jLabe32 = new JLabel("High");
jLabe33 = new JLabel("P 4,000");
jLabe34 = new JLabel("P 5,000");
jLabe35 = new JLabel("P 7,000");
jLabe36 = new JLabel("P 9,000");
jLabe37 = new JLabel("P 11,000");
jLabe38 = new JLabel("Peak");
jLabe39 = new JLabel("P 6,000");
jLabe40 = new JLabel("P 8,000");
jLabe41 = new JLabel("P 10,000");
jLabe42 = new JLabel("P 12,000");
jLabe43 = new JLabel("P 14,000");
jLabe44 = new JLabel("Super Peak");
jLabe45 = new JLabel("P 9,000");
jLabe46 = new JLabel("P 12,000");
jLabe47 = new JLabel("P 15,000");
jLabe48 = new JLabel("P 18,000");
jLabe49 = new JLabel("P 21,000");
jLabe50 = new JLabel("Standard");
jLabe51 = new JLabel("Deluxe");
jLabe52 = new JLabel("Quadruple");
jLabe53 = new JLabel("Family");
jLabe54 = new JLabel("Suite");
jLabe55 = new JLabel("P 2,500");
jLabe56 = new JLabel("P 5,000");
jLabe57 = new JLabel("P 7,500");
jLabe58 = new JLabel("P 10,000");
jLabe59 = new JLabel("P 12,500");
jLabe60 = new JLabel("P 4,500");
jLabe61 = new JLabel("P 7,000");
jLabe62 = new JLabel("P 9,500");
jLabe63 = new JLabel("P 12,000");
jLabe64 = new JLabel("P 14,500");
jLabe65 = new JLabel("P 6,500");
jLabe66 = new JLabel("P 9,000");
jLabe67 = new JLabel("P 11,500");
jLabe68 = new JLabel("P 14,000");
```

```
jLabe69 = new JLabel("P 16,500");
jLabe70 = new JLabel("P 10,000");
jLabe71 = new JLabel("P 13,000");
jLabe72 = new JLabel("P 16,000");
jLabe73 = new JLabel("P 19,000");
jLabe74 = new JLabel("P 22,000");
jLabe75 = new JLabel("International Prices");



// Set bounds for the first header row
jLabel1.setBounds(50, 50, 100, 20);
jLabel2.setBounds(270, 50, 100, 20);
jLabel3.setBounds(450, 50, 120, 20);

// Set bounds for the first capacity labels
jLabel5.setBounds(250, 100, 150, 20);
jLabel6.setBounds(250, 170, 150, 20);
jLabel7.setBounds(250, 240, 150, 20);
jLabel8.setBounds(250, 310, 150, 20);
jLabel9.setBounds(250, 380, 150, 20);
jLabe20.setBounds(500, 100, 150, 20);
jLabe21.setBounds(500, 170, 150, 20);
jLabe22.setBounds(500, 240, 150, 20);
jLabe23.setBounds(500, 310, 150, 20);
jLabe24.setBounds(500, 380, 150, 20);
jLabe25.setBounds(800, 50, 100, 20);
jLabe26.setBounds(655, 90, 100, 20);
jLabe27.setBounds(650, 130, 100, 20);
jLabe28.setBounds(650, 170, 100, 20);
jLabe29.setBounds(650, 210, 100, 20);
jLabe30.setBounds(650, 250, 100, 20);
jLabe31.setBounds(650, 290, 100, 20);
jLabe32.setBounds(775, 90, 100, 20);
jLabe33.setBounds(770, 130, 100, 20);
jLabe34.setBounds(770, 170, 100, 20);
jLabe35.setBounds(770, 210, 100, 20);
jLabe36.setBounds(770, 250, 100, 20);
jLabe37.setBounds(770, 290, 100, 20);
jLabe38.setBounds(875, 90, 100, 20);
jLabe39.setBounds(870, 130, 100, 20);
jLabe40.setBounds(870, 170, 100, 20);
```

```
jLabe41.setBounds(870, 210, 100, 20);
jLabe42.setBounds(870, 250, 100, 20);
jLabe43.setBounds(870, 290, 100, 20);
jLabe44.setBounds(960, 90, 100, 20);
jLabe45.setBounds(970, 130, 100, 20);
jLabe46.setBounds(970, 170, 100, 20);
jLabe47.setBounds(970, 210, 100, 20);
jLabe48.setBounds(970, 250, 100, 20);
jLabe49.setBounds(970, 290, 100, 20);
jLabe50.setBounds(1070, 130, 100, 20);
jLabe51.setBounds(1070, 170, 100, 20);
jLabe52.setBounds(1070, 210, 100, 20);
jLabe53.setBounds(1070, 250, 100, 20);
jLabe54.setBounds(1070, 290, 100, 20);
jLabe55.setBounds(650, 350, 100, 20);
jLabe56.setBounds(650, 390, 100, 20);
jLabe57.setBounds(650, 430, 100, 20);
jLabe58.setBounds(650, 470, 100, 20);
jLabe59.setBounds(650, 510, 100, 20);
jLabe60.setBounds(770, 350, 100, 20);
jLabe61.setBounds(770, 390, 100, 20);
jLabe62.setBounds(770, 430, 100, 20);
jLabe63.setBounds(770, 470, 100, 20);
jLabe64.setBounds(770, 510, 100, 20);
jLabe65.setBounds(870, 350, 100, 20);
jLabe66.setBounds(870, 390, 100, 20);
jLabe67.setBounds(870, 430, 100, 20);
jLabe68.setBounds(870, 470, 100, 20);
jLabe69.setBounds(870, 510, 100, 20);
jLabe70.setBounds(970, 350, 100, 20);
jLabe71.setBounds(970, 390, 100, 20);
jLabe72.setBounds(970, 430, 100, 20);
jLabe73.setBounds(970, 470, 100, 20);
jLabe74.setBounds(970, 510, 100, 20);
jLabe75.setBounds(790, 325, 150, 20);



// Add action listeners to buttons
jButton1.addActionListener(evt -> navigateToFifthPage());
jButton2.addActionListener(evt -> navigateToFifthPage());
```

```
jButton3.addActionListener(evt -> navigateToFifthPage());
jButton4.addActionListener(evt -> navigateToFifthPage());
jButton5.addActionListener(evt -> navigateToFifthPage());
jButton6.addActionListener(evt -> suggestRoom());

// Add components to the JFrame
add(jButton1);
add(jButton2);
add(jButton3);
add(jButton4);
add(jButton5);
add(jButton6);

add(jLabel1);
add(jLabel2);
add(jLabel3);

add(jLabel5);
add(jLabel6);
add(jLabel7);
add(jLabel8);
add(jLabel9);
add(jLabe20);
add(jLabe21);
add(jLabe22);
add(jLabe23);
add(jLabe24);
add(jLabe25);
add(jLabe26);
add(jLabe27);
add(jLabe28);
add(jLabe29);
add(jLabe30);
add(jLabe31);
add(jLabe32);
add(jLabe33);
add(jLabe34);
add(jLabe35);
add(jLabe36);
add(jLabe37);
add(jLabe38);
add(jLabe39);
```

```
add(jLabe40);
add(jLabe41);
add(jLabe42);
add(jLabe43);
add(jLabe44);
add(jLabe45);
add(jLabe46);
add(jLabe47);
add(jLabe48);
add(jLabe49);
add(jLabe50);
add(jLabe51);
add(jLabe52);
add(jLabe53);
add(jLabe54);
add(jLabe55);
add(jLabe56);
add(jLabe57);
add(jLabe58);
add(jLabe59);
add(jLabe60);
add(jLabe61);
add(jLabe62);
add(jLabe63);
add(jLabe64);
add(jLabe65);
add(jLabe66);
add(jLabe67);
add(jLabe68);
add(jLabe69);
add(jLabe70);
add(jLabe71);
add(jLabe72);
add(jLabe73);
add(jLabe74);
add(jLabe75);



// Center the JFrame on screen
setLocationRelativeTo(null);
```

```
// Make the frame visible
setVisible(true);
}

private void navigateToFifthPage() {
FifthPage fifthPage = new FifthPage();
fifthPage.setVisible(true);
dispose();
}

private void suggestRoom() {
JOptionPane.showMessageDialog(this, "If it is 5 people that booking and below RoomTypes
Standard,Deluxe,Quadruple,and Suite if 6 or more Family");
}

public static void main(String[] args) {
java.awt.EventQueue.invokeLater(() -> new FourthPage());
}
}

class FifthPage extends JFrame {

private JTextField bedTextField, pillowsTextField, blanketsTextField, toiletriesTextField;
private JButton confirmButton;

public FifthPage() {
// Set up the frame
setTitle("Add-Ons");
setSize(500, 500);
setLocationRelativeTo(null);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setLayout(null);

// Title Label at the top
JLabel titleLabel = new JLabel("Add-Ons", SwingConstants.CENTER);
titleLabel.setFont(new Font("Segoe UI", Font.BOLD, 18));
titleLabel.setBounds(150, 10, 200, 30);
add(titleLabel);

// Create and position "Bed" options
JLabel bedLabel = new JLabel("Bed:");
bedLabel.setBounds(50, 50, 100, 25);
```

```
add(bedLabel);
bedTextField = new JTextField();
bedTextField.setBounds(150, 50, 200, 25);
add(bedTextField);
JLabel bedCostLabel = new JLabel("Php 650");
bedCostLabel.setBounds(360, 50, 120, 25);
add(bedCostLabel);

// Create and position "Pillows" options
JLabel pillowsLabel = new JLabel("Pillows:");
pillowsLabel.setBounds(50, 100, 100, 25);
add(pillowsLabel);
pillowsTextField = new JTextField();
pillowsTextField.setBounds(150, 100, 200, 25);
add(pillowsTextField);
JLabel pillowsCostLabel = new JLabel("Php 150 each");
pillowsCostLabel.setBounds(360, 100, 120, 25);
add(pillowsCostLabel);

// Create and position "Blankets" options
JLabel blanketsLabel = new JLabel("Blankets:");
blanketsLabel.setBounds(50, 150, 100, 25);
add(blanketsLabel);
blanketsTextField = new JTextField();
blanketsTextField.setBounds(150, 150, 200, 25);
add(blanketsTextField);
JLabel blanketsCostLabel = new JLabel("Php 200 each");
blanketsCostLabel.setBounds(360, 150, 120, 25);
add(blanketsCostLabel);

// Create and position "Toiletries" options
JLabel toiletriesLabel = new JLabel("Toiletries:");
toiletriesLabel.setBounds(50, 200, 100, 25);
add(toiletriesLabel);
toiletriesTextField = new JTextField();
toiletriesTextField.setBounds(150, 200, 200, 25);
add(toiletriesTextField);
JLabel toiletriesCostLabel = new JLabel("Php 100 each");
toiletriesCostLabel.setBounds(360, 200, 120, 25);
add(toiletriesCostLabel);

// Confirm Button with action listener
```

```
confirmButton = new JButton("Confirm");
confirmButton.setBounds(200, 300, 100, 40);
confirmButton.addActionListener(e -> handleConfirmButton());
add(confirmButton);

setVisible(true);
}

private void handleConfirmButton() {
try {
// Trim input to remove any leading or trailing spaces
int bedCount = Integer.parseInt(bedTextField.getText().trim());
int pillowCount = Integer.parseInt(pillowsTextField.getText().trim());
int blanketCount = Integer.parseInt(blanketsTextField.getText().trim());
int toiletriesCount = Integer.parseInt(toiletriesTextField.getText().trim());

// Bed count validation: Bed must have the value '0' or '1'
if (bedCount != 0 && bedCount != 1) {
JOptionPane.showMessageDialog(
this,
"You can only request 1 bed if you do not want to request an extra bed type in 0",
"Invalid Input",
JOptionPane.ERROR_MESSAGE
);
return;
}

// Check if any count is negative
if (pillowCount < 0 || blanketCount < 0 || toiletriesCount < 0) {
JOptionPane.showMessageDialog(
this,
"Please enter non-negative numbers for all fields.",
"Invalid Input",
JOptionPane.ERROR_MESSAGE
);
return;
}

// Calculate costs
double totalBedCost = bedCount * 650.0; // Bed price is Php 650 if bedCount is 1, otherwise 0
double totalPillowCost = pillowCount * 150.0;
double totalBlanketCost = blanketCount * 200.0;
```

```
double totalToiletriesCost = toiletriesCount * 100.0;

double totalCost = totalBedCost + totalPillowCost + totalBlanketCost + totalToiletriesCost;

// Show computed costs
JOptionPane.showMessageDialog(
this,
String.format(
"Costs:\nBed: Php %.2f\nPillows: Php %.2f\nBlankets: Php %.2f\nToiletries: Php %.2f\nTotal: Php %.2f",
totalBedCost, totalPillowCost, totalBlanketCost, totalToiletriesCost, totalCost
),
"Cost Calculation",
JOptionPane.INFORMATION_MESSAGE
);

// If validation is successful, navigate to the SixthPage
navigateToSixthPage();

} catch (NumberFormatException ex) {
JOptionPane.showMessageDialog(
this,
"Please enter valid numeric values for all fields.",
"Input Error",
JOptionPane.ERROR_MESSAGE
);
}
}

// Method to navigate to the SixthPage
private void navigateToSixthPage() {
new SixthPage().setVisible(true); // Open SixthPage
dispose(); // Close current window (FifthPage)
}

public static void main(String[] args) {
SwingUtilities.invokeLater(FifthPage::new);
}
}

class SixthPage extends JFrame {

public SixthPage() {
```

```
initComponents();
}

private void initComponents() {
// Create JFrame
JFrame frame = new JFrame("AMENITIES");
frame.setSize(600, 800);
frame.setLayout(new BoxLayout(frame.getContentPane(), BoxLayout.Y_AXIS));

// Assume total number of guests is predefined
final int totalGuests = 10; // Example value

// Create checkbox panels with price and questions
JPanel poolPanel = createCheckboxWithQuestions("Pool", new String[]{
"How many guests would like to avail for the pool:",
"Is there an elderly or PWD guest/s availing for the pool:",
"How many days would the pool availment last:"
}, 500.00); // Pool price per day

JPanel gymPanel = createCheckboxWithQuestions("Gym", new String[]{
"How many guests would like to avail for the gym:",
"Is there an elderly or PWD guest/s availing for the gym:",
"How many days will the gym availment last:"
}, 500.00); // Gym price per day

JPanel footSpaPanel = createCheckboxWithQuestions("Foot Spa", new String[]{
"How many guests would like to avail for the foot spa:",
"Is there an elderly or PWD guest/s availing for the foot spa:",
"How many session/s will the foot spa availment last:"
}, 850.00); // Foot Spa price per session

JPanel facialPanel = createCheckboxWithQuestions("Facial Massage", new String[]{
"How many guests would like to avail for the facial massage:",
"Is there an elderly or PWD guest/s availing for the facial massage:",
"How many session/s will the facial massage availment last:"
}, 1045.00); // Facial Massage price per session

JPanel thaiPanel = createCheckboxWithQuestions("Thai Massage", new String[]{
"How many guests would like to avail for the Thai massage:",
"Is there an elderly or PWD guest/s availing for the Thai massage:",
"How many session/s will the Thai massage availment last:"
}, 1540.00); // Thai Massage price per session
```

```
// Add panels to the frame
frame.add(poolPanel);
frame.add(gymPanel);
frame.add(footSpaPanel);
frame.add(facialPanel);
frame.add(thaiPanel);

// Confirm button
JButton button1 = new JButton("Confirm");
button1.setAlignmentX(Component.CENTER_ALIGNMENT);
button1.addActionListener(e -> {
// Perform validation for each panel and calculate total cost
boolean isValid = validateGuestCount(poolPanel, totalGuests, "Pool", 500.00) &&
validateGuestCount(gymPanel, totalGuests, "Gym", 500.00) &&
validateGuestCount(footSpaPanel, totalGuests, "Foot Spa", 850.00) &&
validateGuestCount(facialPanel, totalGuests, "Facial Massage", 1045.00) &&
validateGuestCount(thaiPanel, totalGuests, "Thai Massage", 1540.00);

if (isValid) {
// Calculate the total cost for each amenity
double totalPool = calculateTotalCost(poolPanel, 500.00);
double totalGym = calculateTotalCost(gymPanel, 500.00);
double totalFootSpa = calculateTotalCost(footSpaPanel, 850.00);
double totalFacialMassage = calculateTotalCost(facialPanel, 1045.00);
double totalThaiMassage = calculateTotalCost(thaiPanel, 1540.00);

// Display the total cost summary
String message = "Availed Amenities Data and Prices:\n\n";
message += "POOL: Total: Php " + totalPool + "\n";
message += "GYM: Total: Php " + totalGym + "\n";
message += "Foot Spa: Total: Php " + totalFootSpa + "\n";
message += "Facial Massage: Total: Php " + totalFacialMassage + "\n";
message += "Thai Massage: Total: Php " + totalThaiMassage + "\n\n";

// Show the validation message
int confirm = JOptionPane.showConfirmDialog(frame, message + "\nProceed to the next page?",
"Validation Successful", JOptionPane.OK_CANCEL_OPTION);
if (confirm == JOptionPane.OK_OPTION) {
// Navigate to the seventh page
navigateToSeventhhPage();
}
```

```
}
});
frame.add(button1);

// Set default close operation and visibility
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.setVisible(true);
}

private JPanel createCheckboxWithQuestions(String title, String[] questions, double pricePerUnit) {
JPanel mainPanel = new JPanel();
mainPanel.setLayout(new BoxLayout(mainPanel, BoxLayout.Y_AXIS)); // Set vertical BoxLayout
mainPanel.setAlignmentX(Component.LEFT_ALIGNMENT);

JPanel checkboxPanel = new JPanel(new FlowLayout(FlowLayout.LEFT, 0, 0)); // Left-align checkbox
JCheckBox checkBox = new JCheckBox(title);
checkboxPanel.add(checkBox);
mainPanel.add(checkboxPanel);

JPanel questionsPanel = new JPanel();
questionsPanel.setLayout(new GridLayout(questions.length, 2, 5, 5)); // Rows, columns, hgap, vgap
questionsPanel.setVisible(false); // Start with questions hidden

// Add labels and text fields for the questions
for (String question : questions) {
JLabel label = new JLabel(question);
JTextField textField = new JTextField();
questionsPanel.add(label);
questionsPanel.add(textField);
}

// Wrap the questions panel in a small gap
JPanel wrappedPanel = new JPanel();
wrappedPanel.setLayout(new BorderLayout());
wrappedPanel.add(questionsPanel, BorderLayout.CENTER);
wrappedPanel.setBorder(BorderFactory.createEmptyBorder(5, 20, 10, 10)); // Spacing around questions

mainPanel.add(wrappedPanel);

// Show/hide questions panel based on checkbox selection
checkBox.addActionListener(e -> questionsPanel.setVisible(checkBox.isSelected()));
```

```
return mainPanel;
}

private boolean validateGuestCount(JPanel panel, int totalGuests, String amenityName, double
pricePerUnit) {
JCheckBox checkBox = (JCheckBox) ((JPanel) panel.getComponent(0)).getComponent(0);
if (!checkBox.isSelected()) {
return true; // Skip validation if checkbox is not selected
}

JPanel wrappedPanel = (JPanel) panel.getComponent(1);
JPanel questionsPanel = (JPanel) wrappedPanel.getComponent(0);

// Correct indexing to get the right text fields
JTextField guestCountField = (JTextField) questionsPanel.getComponent(1); // Guest count field
JTextField daysOrSessionsField = (JTextField) questionsPanel.getComponent(3); // Days or sessions field
JTextField elderlyPWDField = (JTextField) questionsPanel.getComponent(5); // Elderly/PWD field

try {
// Validate and parse guest count
String guestCountText = guestCountField.getText().trim();
if (guestCountText.isEmpty() || !guestCountText.matches("\\d+")) {
JOptionPane.showMessageDialog(panel, "Please enter a valid number for the " + amenityName + "
guest count.",
"Validation Error", JOptionPane.ERROR_MESSAGE);
return false;
}
int guestCount = Integer.parseInt(guestCountText);
if (guestCount > totalGuests) {
JOptionPane.showMessageDialog(panel, "The number of guests availing the " + amenityName +
" exceeds the total number of guests (" + totalGuests + ").", "Validation Error",
JOptionPane.ERROR_MESSAGE);
return false;
}

// Validate and parse days or sessions count
String daysOrSessionsText = daysOrSessionsField.getText().trim();
if (daysOrSessionsText.isEmpty() || !daysOrSessionsText.matches("\\d+")) {
JOptionPane.showMessageDialog(panel, "Please enter a valid number for the " + amenityName + "
days/sessions.",
"Validation Error", JOptionPane.ERROR_MESSAGE);
return false;
```

```
}

// Validate and parse elderly/PWD count (allowing 0)
String elderlyPWDText = elderlyPWDField.getText().trim();
if (!elderlyPWDText.matches("\\d+") || Integer.parseInt(elderlyPWDText) <= 0) {
JOptionPane.showMessageDialog(panel, "Please enter a valid number (including 0) for elderly/PWD
guests.",
"Validation Error", JOptionPane.ERROR_MESSAGE);
return false;
}

} catch (NumberFormatException e) {
JOptionPane.showMessageDialog(panel, "Invalid input for " + amenityName + " guest count.",
"Validation Error", JOptionPane.ERROR_MESSAGE);
return false;
}

return true;
}

private double calculateTotalCost(JPanel panel, double pricePerUnit) {
JCheckBox checkBox = (JCheckBox) ((JPanel) panel.getComponent(0)).getComponent(0);
if (!checkBox.isSelected()) {
return 0.0; // Skip if not selected
}

JPanel wrappedPanel = (JPanel) panel.getComponent(1);
JPanel questionsPanel = (JPanel) wrappedPanel.getComponent(0);

// Correct component indexing
JTextField guestCountField = (JTextField) questionsPanel.getComponent(1); // Guest count field
JTextField daysOrSessionsField = (JTextField) questionsPanel.getComponent(3); // Days or sessions field

try {
int guestCount = Integer.parseInt(guestCountField.getText().trim());
int daysOrSessions = Integer.parseInt(daysOrSessionsField.getText().trim());

return guestCount * daysOrSessions * pricePerUnit;
} catch (NumberFormatException e) {
return 0.0;
}
}
```

```
// Method to navigate to the seventh page
private void navigateToSeventhhPage() {
new SeventhPage().setVisible(true); // Open SeventhPage
dispose(); // Close current window (SixthPage)
}

public static void main(String[] args) {
SwingUtilities.invokeLater(SixthPage::new);
}
}

class SeventhPage extends javax.swing.JFrame {

public SeventhPage() {
initComponents();
}

private void initComponents() {
// Initialize labels, text fields, and button
jLabel1 = new JLabel("Reservation Summary");
jLabel2 = new JLabel("Check in:");
jLabel3 = new JLabel("Check out:");
jLabel4 = new JLabel("No. of Adults:");
jLabel5 = new JLabel("No. of Children:");
jLabel6 = new JLabel("Amenities availed:");
jLabel7 = new JLabel("Add-Ons availed:");
jLabel8 = new JLabel("Room Type:");
jLabel9 = new JLabel("Destination:");
jLabel10 = new JLabel("Email:");
jLabel11 = new JLabel("Age:");
jLabel12 = new JLabel("Contact No:");
jLabel13 = new JLabel("TOTAL COST");

jTextField1 = new JTextField(); // Check in
jTextField2 = new JTextField(); // Check out
jTextField3 = new JTextField(); // No. of Adults
jTextField4 = new JTextField(); // No. of Children
jTextField5 = new JTextField(); // Amenities availed
jTextField6 = new JTextField(); // Add-Ons availed
jTextField7 = new JTextField(); // Room Type
jTextField8 = new JTextField(); // Destination
```

```
jTextField9 = new JTextField(); // Email
jTextField10 = new JTextField(); // Age
jTextField11 = new JTextField(); // Contact No
jTextField12 = new JTextField(); // TOTAL COST

jButton1 = new JButton("CONFIRM");
jButton1.setFont(new java.awt.Font("Segoe UI", 1, 14));
jButton1.addActionListener(evt -> jButton1ActionPerformed(evt));

// Set the default texts on the JTextFields
jTextField1.setText(""); // Check in
jTextField2.setText(""); // Check out
jTextField3.setText("1"); // No. of Adults
jTextField4.setText("1"); // No. of Children
jTextField5.setText("1"); // Amenities availed
jTextField6.setText("4"); // Add-Ons availed
jTextField7.setText("Standard"); // Room Type
jTextField8.setText(""); // Destination
jTextField9.setText(""); // Email
jTextField12.setText(""); // TOTAL COST

// Set the layout and positioning of components
javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);

layout.setHorizontalGroup(
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addGroup(layout.createSequentialGroup()
.addGap(30, 30, 30)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addComponent(jLabel2)
.addComponent(jLabel3)
.addComponent(jLabel4)
.addComponent(jLabel5)
.addComponent(jLabel6)
.addComponent(jLabel7)
.addComponent(jLabel8)
.addComponent(jLabel9))
.addGap(41, 41, 41)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING, false)
.addComponent(jTextField6, javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 108, Short.MAX_VALUE)
```

```
.addComponent(jTextField5, javax.swing.GroupLayout.Alignment.LEADING)
.addComponent(jTextField1, javax.swing.GroupLayout.Alignment.LEADING)
.addComponent(jTextField2, javax.swing.GroupLayout.Alignment.LEADING)
.addComponent(jTextField3, javax.swing.GroupLayout.Alignment.LEADING)
.addComponent(jTextField4, javax.swing.GroupLayout.Alignment.LEADING)
.addComponent(jTextField7)
.addComponent(jTextField8))
.addGap(112, 112, 112)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addComponent(jLabel10)
.addComponent(jLabel11)
.addComponent(jLabel12)
.addComponent(jLabel13))
.addGap(31, 31, 31)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING, false)
.addComponent(jTextField9, javax.swing.GroupLayout.DEFAULT_SIZE, 116, Short.MAX_VALUE)
.addComponent(jTextField10)
.addComponent(jTextField11)
.addComponent(jTextField12))
.addContainerGap(44, Short.MAX_VALUE))
.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
.addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
.addComponent(jLabel1)
.addGap(240, 240, 240))
.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
.addComponent(jButton1)
.addGap(271, 271, 271))))
);

layout.setVerticalGroup(
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addGroup(layout.createSequentialGroup()
.addGap(18, 18, 18)
.addComponent(jLabel1)
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
.addComponent(jLabel2)
.addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
.addComponent(jLabel10)
```

```
.addComponent(jTextField9, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
.addGap(18, 18, 18)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
.addComponent(jLabel3)
.addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
.addComponent(jLabel11)
.addComponent(jTextField10, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
.addGap(18, 18, 18)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
.addComponent(jLabel4)
.addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
.addComponent(jLabel12)
.addComponent(jTextField11, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
.addGap(18, 18, 18)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
.addComponent(jLabel5)
.addComponent(jTextField4, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
.addGap(18, 18, 18)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
.addComponent(jLabel6)
.addComponent(jTextField5, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
.addGap(18, 18, 18)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
.addComponent(jLabel7)
.addComponent(jTextField6, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
.addGap(18, 18, 18)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
.addComponent(jLabel8)
.addComponent(jTextField7, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
.addComponent(jLabel13)
.addComponent(jTextField12, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
.addGap(18, 18, 18)
```

```
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
.addComponent(jLabel9)
.addComponent(jTextField8, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED, 49, Short.MAX_VALUE)
.addComponent(jButton1)
.addGap(16, 16, 16))
);

pack();
}

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
// Open the PaymentPage when the button is clicked
PaymentPage jf2 = new PaymentPage();
jf2.show();
dispose();
}

public static void main(String args[]) {
java.awt.EventQueue.invokeLater(() -> {
new SeventhPage().setVisible(true);
});
}

// Variables declaration
private javax.swing.JButton jButton1;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel10;
private javax.swing.JLabel jLabel11;
private javax.swing.JLabel jLabel12;
private javax.swing.JLabel jLabel13;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JLabel jLabel6;
private javax.swing.JLabel jLabel7;
private javax.swing.JLabel jLabel8;
private javax.swing.JLabel jLabel9;
private javax.swing.JTextField jTextField1;
private javax.swing.JTextField jTextField10;
```

```
private javax.swing.JTextField jTextField11;
private javax.swing.JTextField jTextField12;
private javax.swing.JTextField jTextField2;
private javax.swing.JTextField jTextField3;
private javax.swing.JTextField jTextField4;
private javax.swing.JTextField jTextField5;
private javax.swing.JTextField jTextField6;
private javax.swing.JTextField jTextField7;
private javax.swing.JTextField jTextField8;
private javax.swing.JTextField jTextField9;
}

class PaymentPage extends javax.swing.JFrame {

// Constructor for initializing the form
public PaymentPage() {
initComponents();
}

@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">
private void initComponents() {

jLabel1 = new javax.swing.JLabel();
jButton1 = new javax.swing.JButton();
jButton2 = new javax.swing.JButton();

setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

jLabel1.setFont(new java.awt.Font("Segoe UI", 1, 18)); // NOI18N
jLabel1.setText("MODES OF PAYMENT");

jButton1.setFont(new java.awt.Font("Segoe UI", 1, 14)); // NOI18N
jButton1.setText("CASH");
jButton1.addActionListener(new java.awt.event.ActionListener() {
public void actionPerformed(java.awt.event.ActionEvent evt) {
navigateToCashPage(); // Navigate to CashPage when clicked
}
});

jButton2.setFont(new java.awt.Font("Segoe UI", 1, 14)); // NOI18N
jButton2.setText("CARD");
```

```
jButton2.addActionListener(new java.awt.event.ActionListener() {
public void actionPerformed(java.awt.event.ActionEvent evt) {
navigateToCardPage(); // Navigate to CardPage when clicked
}
});

javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
getContentPane().setLayout(layout);
layout.setHorizontalGroup(
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
.addContainerGap(258, Short.MAX_VALUE)
.addComponent(jLabel1)
.addGap(222, 222, 222))
.addGroup(layout.createSequentialGroup()
.addGap(61, 61, 61)
.addComponent(jButton1)
.addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
.addComponent(jButton2)
.addGap(48, 48, 48))
);
layout.setVerticalGroup(
layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addGroup(layout.createSequentialGroup()
.addGap(24, 24, 24)
.addComponent(jLabel1)
.addGap(74, 74, 74)
.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
.addComponent(jButton1)
.addComponent(jButton2))
.addContainerGap(268, Short.MAX_VALUE))
);

pack();
}// </editor-fold>

// Method to navigate to CashPage
private void navigateToCashPage() {
new CashPage().setVisible(true); // Open CashPage
dispose(); // Close current window (PaymentPage)
}
```

```
// Method to navigate to CardPage
private void navigateToCardPage() {
new CardPage().setVisible(true); // Open CardPage
dispose(); // Close current window (PaymentPage)
}

public static void main(String args[]) {
// Set the Nimbus look and feel
try {
for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {
if ("Nimbus".equals(info.getName())) {
javax.swing.UIManager.setLookAndFeel(info.getClassName());
break;
}
}
} catch (ClassNotFoundException | InstantiationException | IllegalAccessException |
javax.swing.UnsupportedLookAndFeelException ex) {
java.util.logging.Logger.getLogger(PaymentPage.class.getName()).log(java.util.logging.Level.SEVERE, null,
ex);
}

/* Create and display the form */
java.awt.EventQueue.invokeLater(new Runnable() {
public void run() {
new PaymentPage().setVisible(true);
}
});
}

// Variables declaration - do not modify
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton2;
private javax.swing.JLabel jLabel1;
// End of variables declaration
}

class CardPage extends JFrame {

// Declare components
private JTextField cardNumberField;
private JLabel instructionLabel;
```

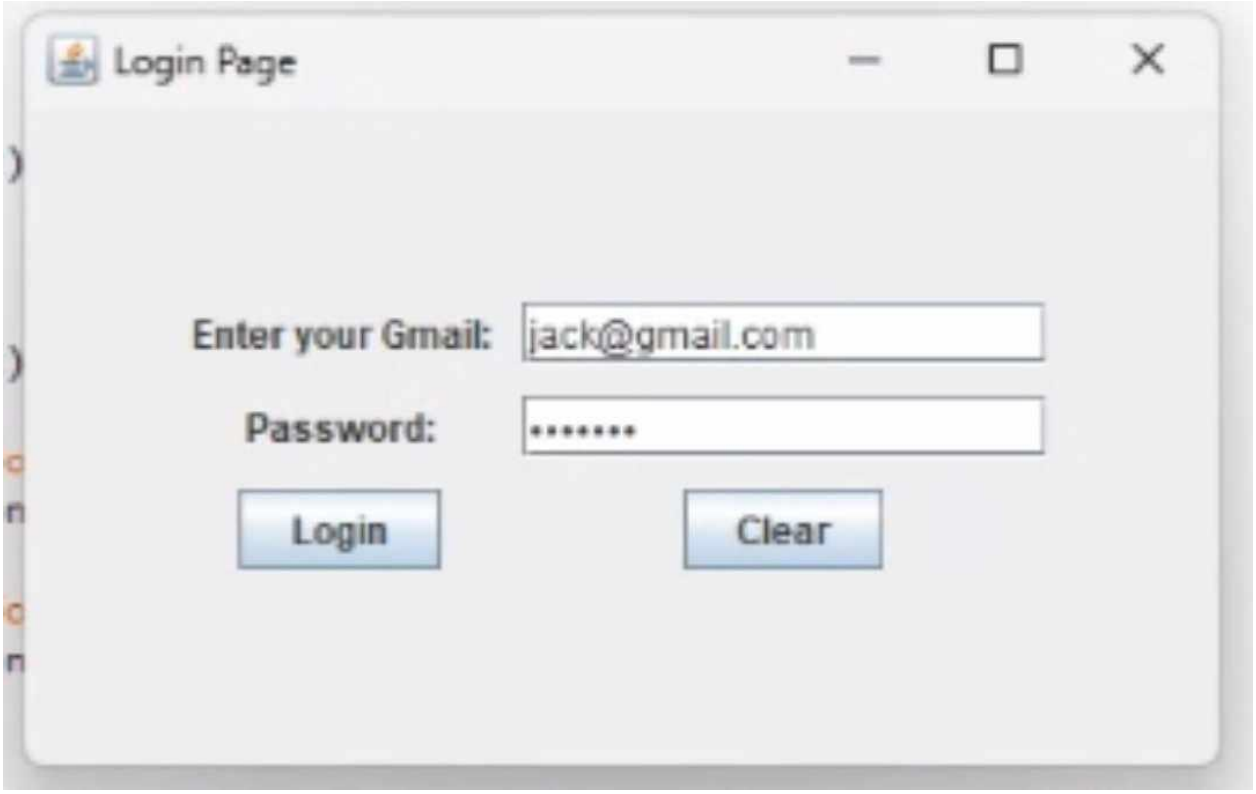
```
private JButton confirmButton;

public CardPage() {
// Initialize components
cardNumberField = new JTextField();
instructionLabel = new JLabel("Please enter your card number here:");
confirmButton = new JButton("CONFIRM");

// Set up the frame
setTitle("Card Payment Page");
setSize(350, 200);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setLayout(null); // Using null layout for manual positioning

// Position the components
instructionLabel.setBounds(20, 20, 300, 30);
cardNumberField.setBounds(20, 60, 200, 30);
confirmButton.setBounds(100, 100, 120, 30);

// Add components to the frame
add(instructionLabel);
add(cardNumberField);
add(confirmButton);

// Add action listener to the confirm button
confirmButton.addActionListener(e -> handleCardNumber());
}

private void handleCardNumber() {
String cardNumber = cardNumberField.getText().trim();

// Simple validation for card number (check if it's digits and has a length of 16)
if (cardNumber.matches("\\d{16}")) {
JOptionPane.showMessageDialog(this, "Card Number Accepted.");

// Now ask if the user wants to book again
int option = JOptionPane.showConfirmDialog(this, "Would you like to book again?",
"Booking Confirmation", JOptionPane.YES_NO_OPTION);
if (option == JOptionPane.YES_OPTION) {
// If the user chooses "Yes", open the main menu (HotelStarPlatinum)
new HotelStarPlatinumCase().setVisible(true);
dispose(); // Close the current window (CardPage)
```

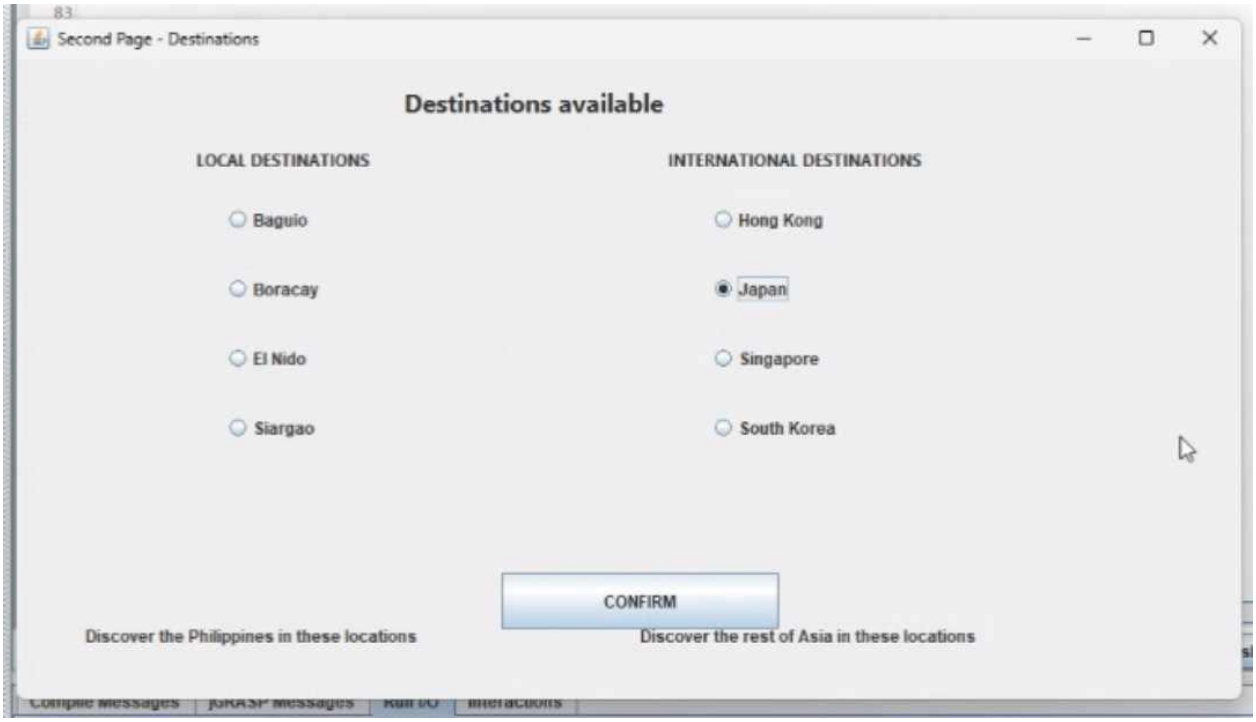
```
} else {
// If the user chooses "No", exit the program
System.exit(0);
}
} else {
JOptionPane.showMessageDialog(this, "Invalid card number. Please enter a 16-digit number.");
}
}

public static void main(String[] args) {
// Run the application
SwingUtilities.invokeLater(() -> {
new CardPage().setVisible(true);
});
}
}

class CashPage extends JFrame {

// Declare components
private JTextField paymentField;
private JLabel instructionLabel, advisoryLabel, noteLabel;
private JButton confirmButton;

public CashPage() {
// Initialize the components
paymentField = new JTextField();
instructionLabel = new JLabel("Please enter the amount of your pay here");
advisoryLabel = new JLabel("Advisory:");
noteLabel = new JLabel("If you enter a number less than your required payment, it will not allow.");
confirmButton = new JButton("CONFIRM");

// Set up the frame
setTitle("Cash Payment Page");
setSize(350, 250);
setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
setLayout(null); // Set layout to null for manual positioning

// Position the components
instructionLabel.setBounds(20, 20, 300, 30);
paymentField.setBounds(20, 60, 120, 30);
advisoryLabel.setBounds(20, 100, 80, 30);
```

```
noteLabel.setBounds(20, 130, 300, 30);
confirmButton.setBounds(100, 170, 120, 30);

// Add components to the frame
add(instructionLabel);
add(paymentField);
add(advisoryLabel);
add(noteLabel);
add(confirmButton);

// Add action listener to the confirm button
confirmButton.addActionListener(e -> handlePayment());
}

private void handlePayment() {
try {
// Parse the input amount
double amount = Double.parseDouble(paymentField.getText());

// Assuming the required payment is Php 1000.00
double requiredAmount = 1000.00;

// Check if the entered amount is valid
if (amount >= requiredAmount) {
// Show confirmation message for payment
JOptionPane.showMessageDialog(this, "Payment Confirmed. Thank you!");

// Now ask if the user wants to book again
int option = JOptionPane.showConfirmDialog(this, "Would you like to book again?",
"Booking Confirmation", JOptionPane.YES_NO_OPTION);
if (option == JOptionPane.YES_OPTION) {
// If the user chooses "Yes", open the main menu (HotelStarPlatinum)
new HotelStarPlatinumCase().setVisible(true);
dispose(); // Close the current window (CashPage)
} else {
// If the user chooses "No", exit the program
System.exit(0);
}
} else {
JOptionPane.showMessageDialog(this, "Insufficient amount. Please enter a valid amount.");
}
} catch (NumberFormatException e) {
```

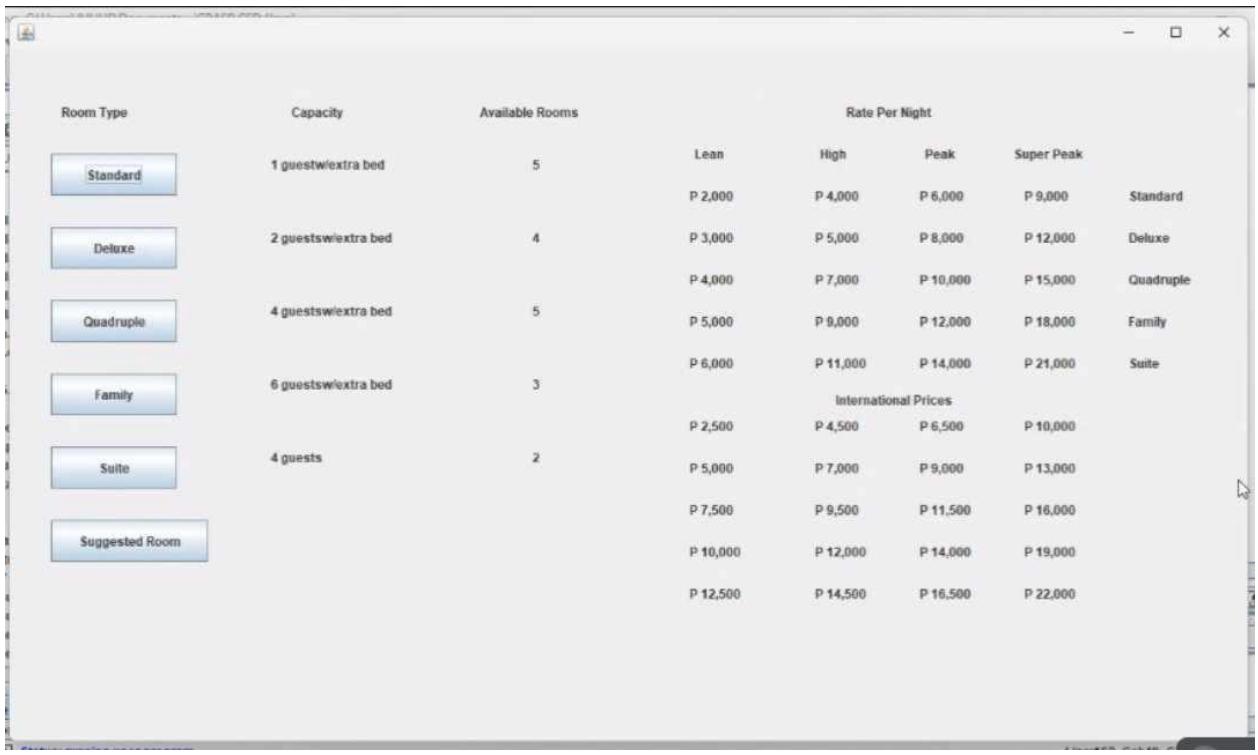
```
// Handle invalid input
JOptionPane.showMessageDialog(this, "Please enter a valid number.");
}
}

public static void main(String[] args) {
// Run the application
SwingUtilities.invokeLater(() -> {
new CashPage().setVisible(true);
});
}
}
```

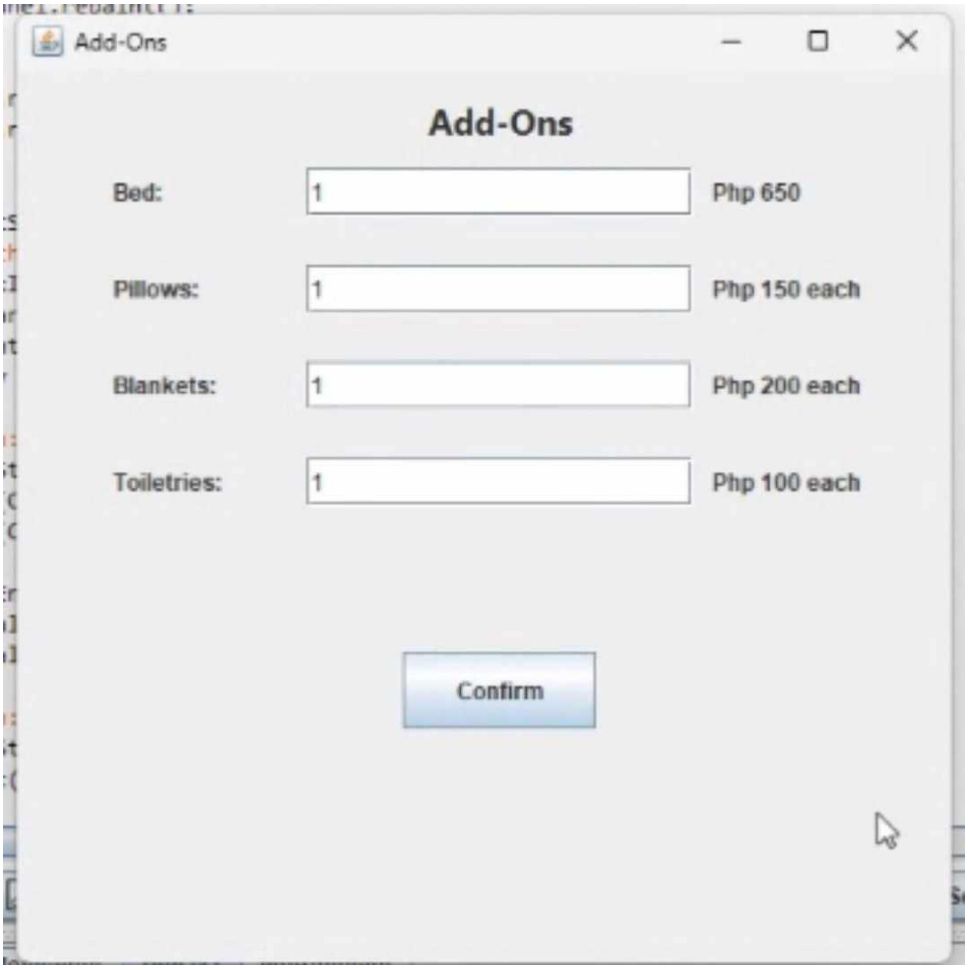

Star Platinum Hotel
WELCOME TO STAR PLATINUM HOTEL
CLICK HERE TO START BOOKING

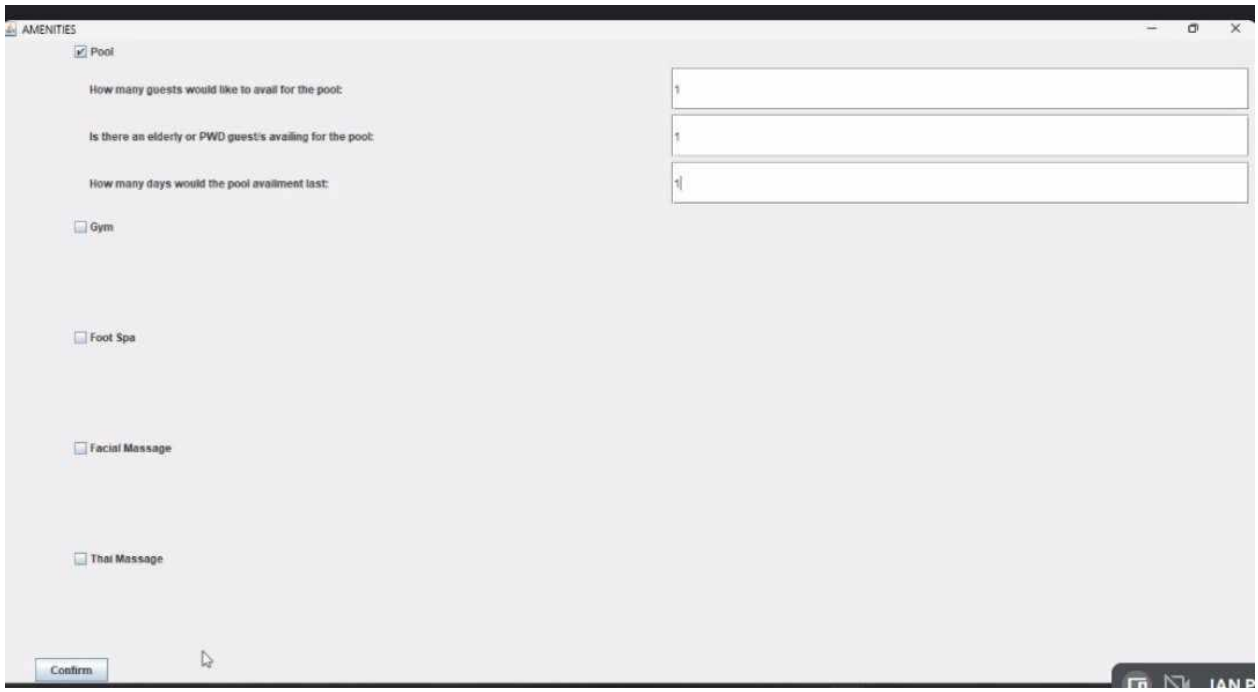

Login Page
Enter your Gmail:
jack@gmail.com
Password:
Login
Clear

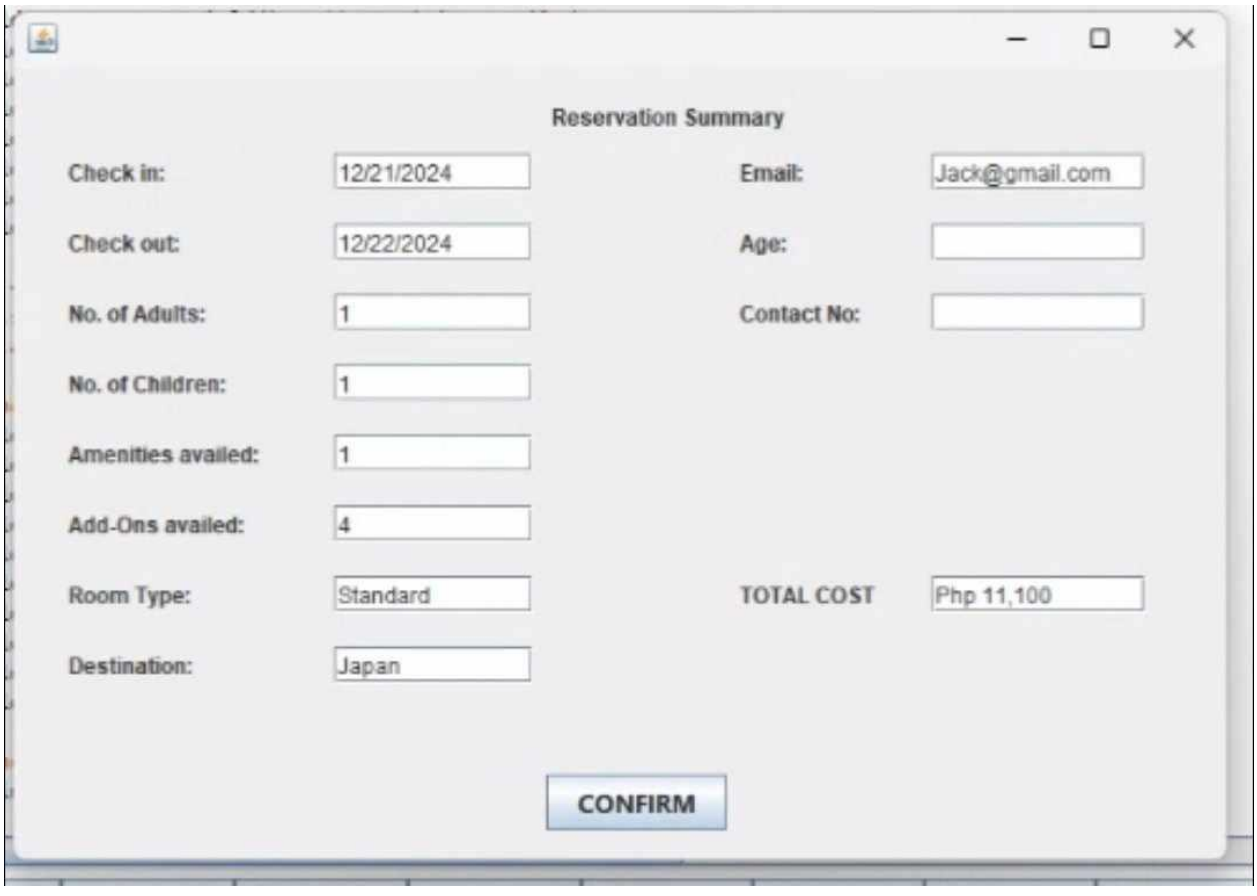

Second Page - Destinations
Destinations available
LOCAL DESTINATIONS
INTERNATIONAL DESTINATIONS
Baguio
Boracay
El Nido
Siargao
Hong Kong
Japan
Singapore
South Korea
CONFIRM
Discover the Philippines in these locations
Discover the rest of Asia in these locations

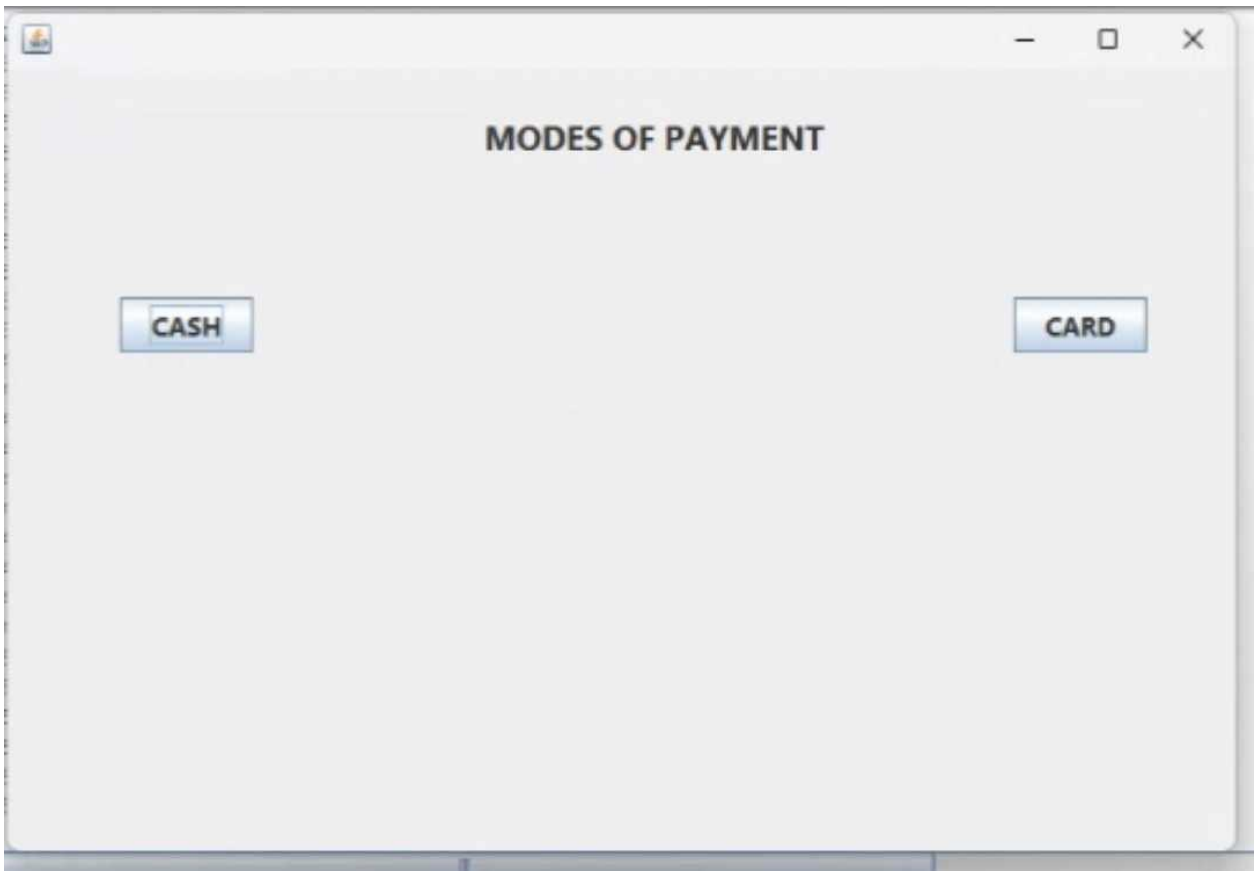

Third Page - Booking Details
Check-in Date (MM/dd/yyyy):
12/21/2024
Check-out Date (MM/dd/yyyy):
12/22/2024
Number of Adults:
1
Number of Children:
1
Age of Child/ren:
Age of child #1:
1
Confirm Booking

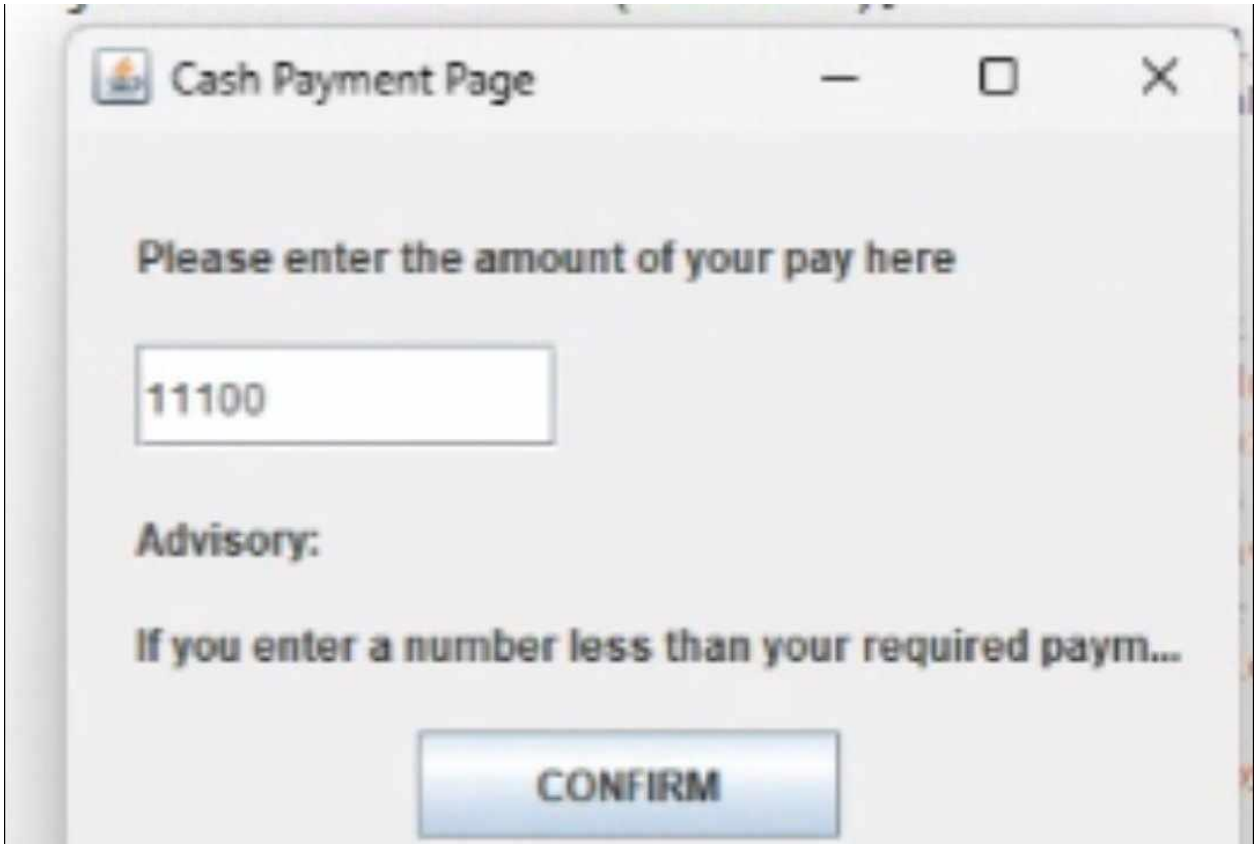

Room Type
Capacity
Available Rooms
Rate Per Night
Lean
High
Peak
Super Peak
Standard
1 guestw/extra bed
5
P 2,000
P 4,000
P 6,000
P 9,000
Standard
Deluxe
2 guestsw/extra bed
4
P 3,000
P 5,000
P 8,000
P 12,000
Deluxe
P 4,000
P 7,000
P 10,000
P 15,000
Quadruple
Quadruple
4 guestsw/extra bed
5
P 5,000
P 9,000
P 12,000
P 18,000
Family
P 6,000
P 11,000
P 14,000
P 21,000
Suite
Family
6 guestsw/extra bed
3
International Prices
P 2,500
P 4,500
P 6,500
P 10,000
Suite
4 guests
2
P 5,000
P 7,000
P 9,000
P 13,000
P 7,500
P 9,500
P 11,500
P 16,000
Suggested Room
P 10,000
P 12,000
P 14,000
P 19,000
P 12,500
P 14,500
P 16,500
P 22,000

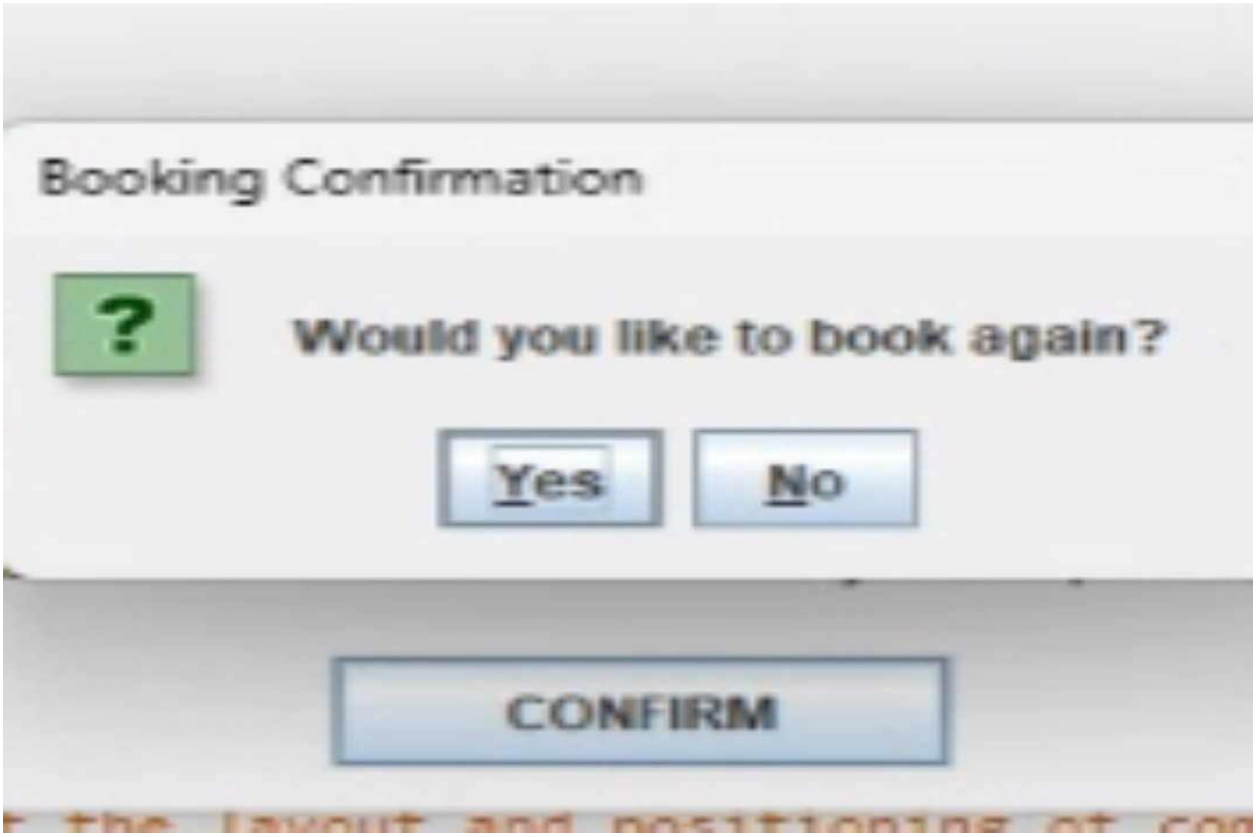

Add-Ons
Add-Ons
Bed:
1
Php 650
Pillows:
1
Php 150 each
Blankets:
1
Php 200 each
Toiletries:
1
Php 100 each
Confirm

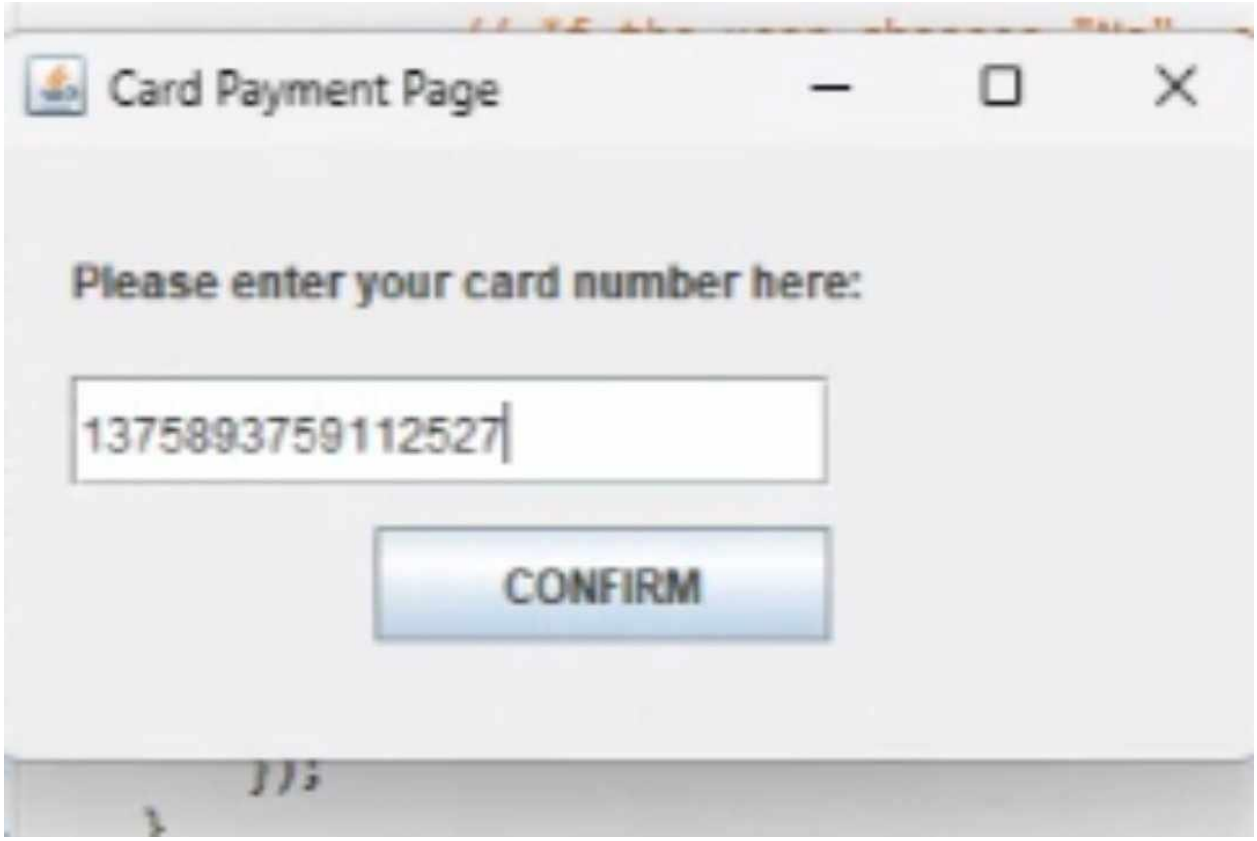

AMENITIES
Pool
How many guests would like to avail for the pool:
1
Is there an elderly or PWD guest/s availing for the pool:
1
How many days would the pool availment last:
1
Gym
Foot Spa
Facial Massage
Thai Massage
Confirm

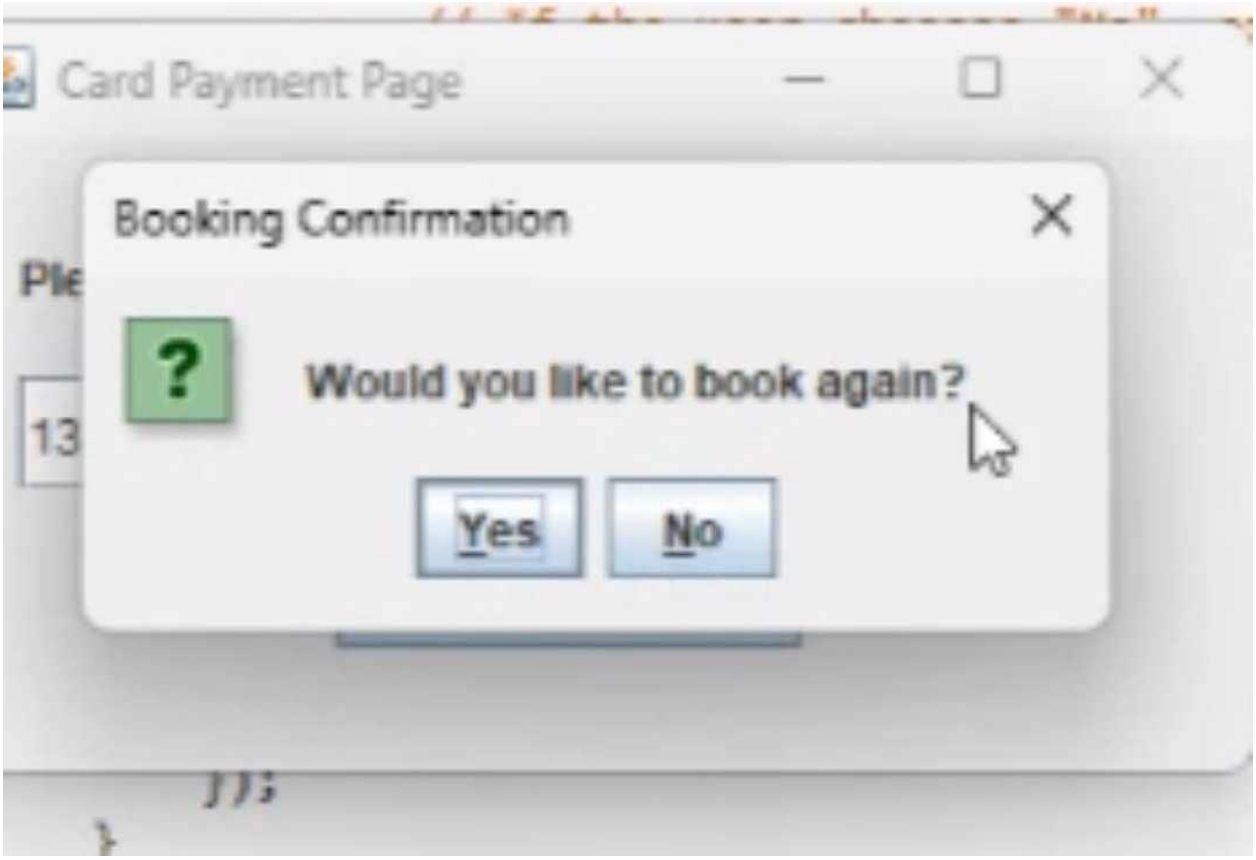

Reservation Summary
Check in:
12/21/2024
Email:
Jack@gmail.com
Check out:
12/22/2024
Age:
No. of Adults:
1
Contact No:
No. of Children:
1
Amenities availed:
1
Add-Ons availed:
4
Room Type:
Standard
TOTAL COST
Php 11,100
Destination:
Japan
CONFIRM


MODES OF PAYMENT
CASH
CARD


Cash Payment Page
Please enter the amount of your pay here
11100
Advisory:
If you enter a number less than your required paym...
CONFIRM


Booking Confirmation
Would you like to book again?
Yes
No
CONFIRM


Card Payment Page
Please enter your card number here:
1375893759112527
CONFIRM


Card Payment Page
Booking Confirmation
Would you like to book again?
Yes
No