Object-Oriented Programming Assignment Report

BTech 4th Semester

2024 - 2025

Project Title: Smart City Application

Submitted To: Mr. Aryan Assistant Professor (SoCS)

Submitted By: Anurag Pandey 500120453

Smart City Application

Abstract

The rapid urbanization and technological advancements have led to the need for smart solutions to improve city infrastructure and enhance the overall quality of life. This project, **Smart City Application**, is designed to serve as an interactive guide for new visitors by providing crucial information about the city, including hotels, restaurants, shopping malls, tourist places, and transportation options. The application is developed using **Java Swing for the UI and MySQL for the database**, ensuring efficient data management and user-friendly navigation.

The system allows users to search for places based on location and displays relevant details such as contact information, pricing, opening hours, and ratings. The transportation module enhances user convenience by providing an interactive **cab-like booking system**. The project follows object-oriented programming (OOP) principles to ensure modularity and scalability.

By integrating technology with urban infrastructure, this application aims to streamline city exploration, making it easier for residents and visitors to access essential services. The future scope includes enhancing the application with real-time updates, GPS-based tracking, and additional smart city functionalities.

Acknowledgment

I would like to express my heartfelt gratitude to my **instructor**, whose invaluable guidance and constructive feedback have been instrumental in the successful completion of this project. Their insights into **Java programming, database management, and UI design** have significantly contributed to my learning experience.

I am also thankful to my **peers and classmates** for their continuous support and encouragement throughout the development process. Their suggestions helped in refining the application and making it more user-friendly.

Finally, I extend my appreciation to my **family and friends**, whose unwavering motivation kept me dedicated to completing this project efficiently. This journey has been a great learning experience, and I look forward to applying these skills in future smart city initiatives.

Table of Contents

Sr. No	Title
1	Introduction
2	Flow Chart
3	UML Diagram
4	Software Requirements
5	Project Directory Structure
6	Code Snippets
7	Database Schema and Tables
8	Output Screenshots
9	Project Summary
10	Important Learnings

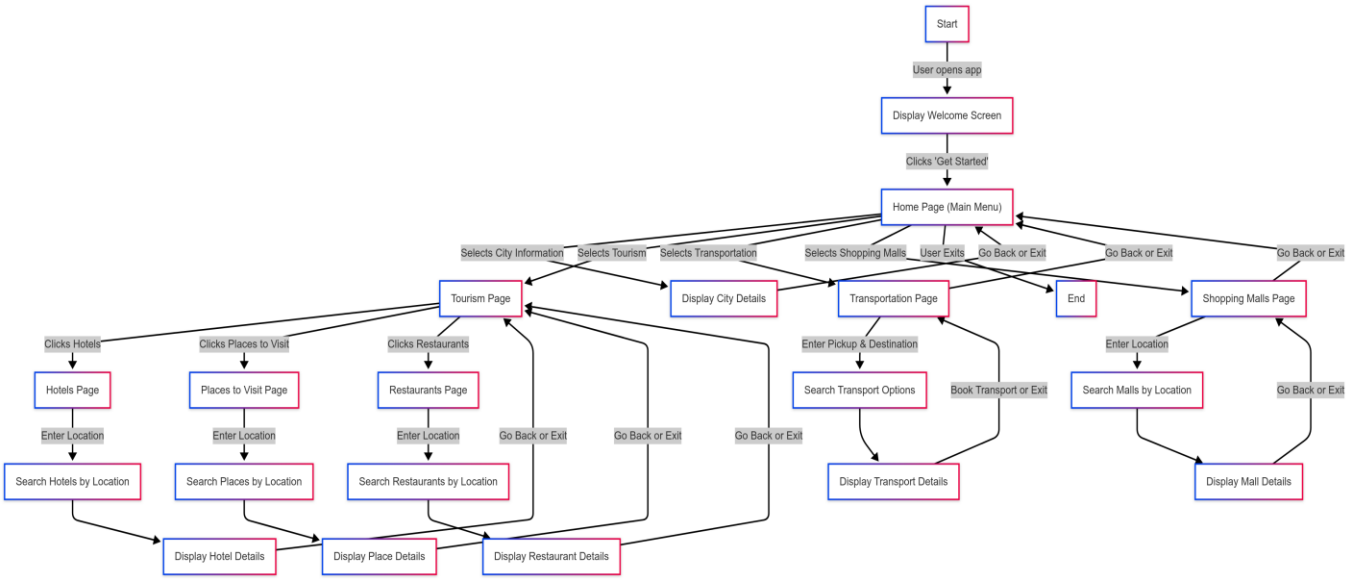
1. Introduction

The Smart City Application is a Java Swing-based desktop application that serves as a digital city guide. It provides users with easy access to essential city services, making their navigation seamless.

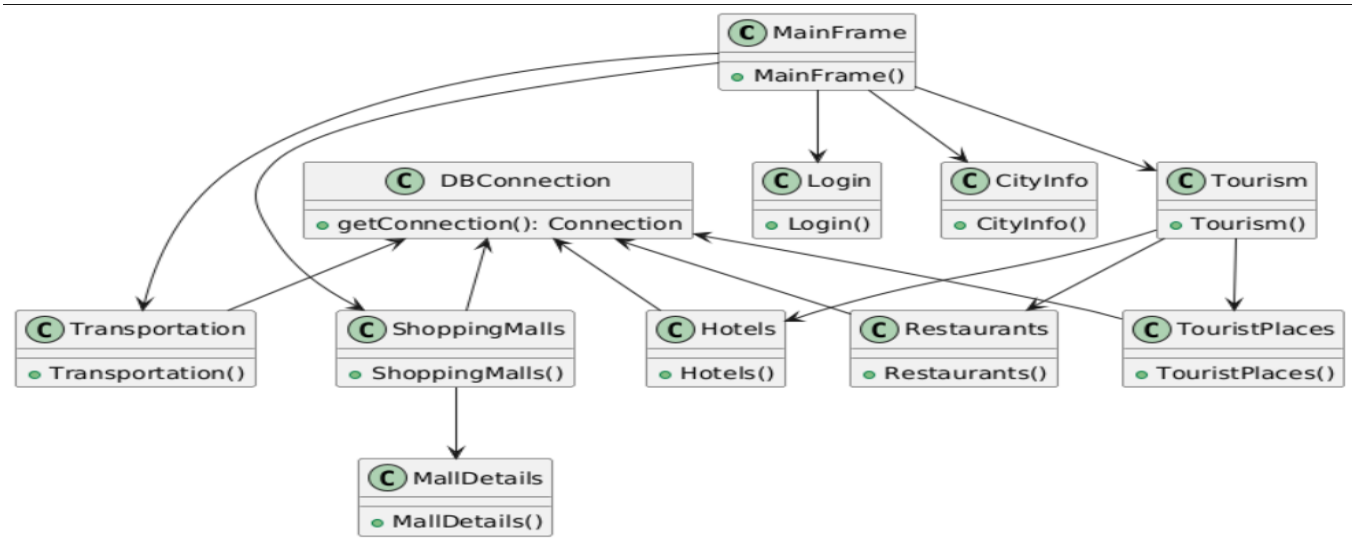
Key Features:

• City Information: Overview of major landmarks and facilities.
• Tourism Guide: Information about hotels, restaurants, and tourist attractions.
• Transportation System: Allows users to find and book rides from different locations.
• shopping malls: Provides details of malls, including their opening and closing times.
• User-Friendly Interface: Interactive UI designed to enhance the user experience.

2. Flow Chart



3. UML Diagram

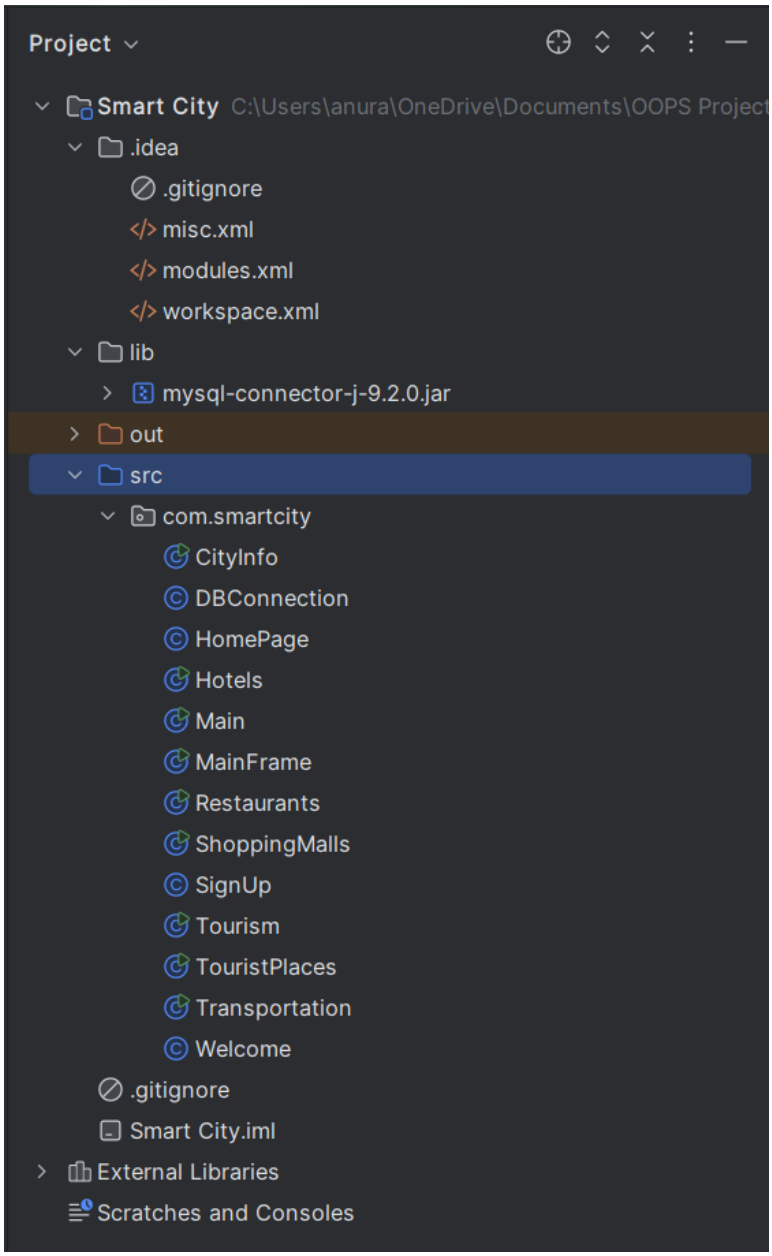


4. Software Requirements

Hardware Requirements:
• Processor: Intel Core i3 or above
• RAM: Minimum 4GB
• Storage: Minimum 500MB

Software Requirements:
• Programming Language: Java (Swing for UI, JDBC for database connectivity)
• Database: MySQL
• IDE Used: IntelliJ IDEA Community Edition
• Libraries: Swing, AWT, MySQL Connector/J
• Operating System: Windows/Linux

5. Project Directory Structure



6. Database Schema and Tables

Query 1

Limit to 1000 rows

```sql
use smartcity;
CREATE TABLE users (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(100),
    username VARCHAR(50) UNIQUE NOT NULL,
    password VARCHAR(50) NOT NULL,
    dob VARCHAR(20),
    country VARCHAR(50),
    state VARCHAR(50),
    pincode VARCHAR(10),
    mobile VARCHAR(15) UNIQUE
);
select * from users;
show tables;
select * from city_information;
select * from tourism;
select * from hotels;
select * from restaurants;
select * from transportation;
select * from shopping_mall;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Co

Tables_in_smartcity
city_information
hotels
restaurants
shopping_mall
tourism
transportation
user_feedback
users

7. Code Snippets

Main.java

```java
package com.smartcity;

public class Main {
    public static void main(String[] args) {
        MainFrame mainFrame = new MainFrame(); // Create MainFrame instance
        new Welcome(mainFrame); // Start with Welcome page
    }
}
```

CheckBudget.java | HomePage.java | MainFrame.java | Signup.java | Update.java | ViewExpenses.java | Welcome.java

```java
package com.expensetracker;

import javax.swing.*;

public class MainFrame extends JFrame {  4 usages
    public MainFrame() {  2 usages
        setTitle("Expense Tracker");
        setSize( width: 500,  height: 400);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
    }
}
```

Main.java | MainFrame.java | Welcome.java

```java
package com.smartcity;

import javax.swing.*;
import java.awt.*;
import java.sql.*;
import java.util.logging.Level;
import java.util.logging.Logger;

public class Welcome extends JFrame {  3 usages
    private JPanel welcomePanel;  6 usages
    private JPanel loginPanel;  10 usages
    private final MainFrame mainFrame;  2 usages

    public Welcome(MainFrame mainFrame) {  2 usages
        this.mainFrame = mainFrame;
        setTitle("Smart City");
        setSize( width: 500,  height: 300);
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setLocationRelativeTo(null);
        setLayout(new CardLayout());

        // Create Welcome and Login Panels
        createWelcomePanel();
        createLoginPanel();

        // Initially show the Welcome Panel
        setContentPane(welcomePanel);
        setVisible(true);
    }

    private void createWelcomePanel() {  1 usage
        welcomePanel = new JPanel(new BorderLayout());
        welcomePanel.setBackground(new Color( r: 220,  g: 235,  b: 255)); // Soft light blue


        // **Title: "Welcome to Smart City"**
        JLabel headline = new JLabel( text: "Welcome to Smart City", SwingConstants.CENTER);
        headline.setFont(new Font( name: "Arial", Font.BOLD,  size: 26));
        headline.setForeground(new Color( r: 0,  g: 76,  b: 153)); // Dark blue for elegance

        // **Subheading**
        JLabel subheading = new JLabel( text: "Your Guide to Explore and Navigate", SwingConstants.CENTER);
        subheading.setFont(new Font( name: "Arial", Font.PLAIN,  size: 16));
        subheading.setForeground(new Color( r: 80,  g: 80,  b: 80)); // Dark gray for readability

        // **Button Panel for Centering**
        JPanel buttonPanel = new JPanel();
        buttonPanel.setBackground(new Color( r: 220,  g: 235,  b: 255)); // Same as main panel
        JButton getStartedButton = new JButton( text: "Get Started");
        getStartedButton.setFont(new Font( name: "Arial", Font.BOLD,  size: 16));
        getStartedButton.setBackground(new Color( r: 0,  g: 153,  b: 76)); // Green
        getStartedButton.setForeground(Color.WHITE);
        getStartedButton.setFocusPainted(false);
        getStartedButton.setPreferredSize(new Dimension( width: 150,  height: 40)); // Proper size

        // **Hover Effect**
        getStartedButton.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseEntered(java.awt.event.MouseEvent evt) {
                getStartedButton.setBackground(new Color( r: 0,  g: 204,  b: 102)); // Lighter green
            }
            public void mouseExited(java.awt.event.MouseEvent evt) {
                getStartedButton.setBackground(new Color( r: 0,  g: 153,  b: 76));
            }
        });
```

```
        buttonPanel.add(getStartedButton);

        // **Adding Components**
        welcomePanel.add(subheading, BorderLayout.NORTH);
        welcomePanel.add(headline, BorderLayout.CENTER);
        welcomePanel.add(buttonPanel, BorderLayout.SOUTH);

        // **Button Click Action: Switch to Login**
        getStartedButton.addActionListener( ActionEvent e -> {
            setContentPane(loginPanel);
            revalidate();
            repaint();
        });
    }

    private void createLoginPanel() {  1 usage
        loginPanel = new JPanel(new GridBagLayout());
        loginPanel.setBackground(Color.WHITE); // White background for contrast

        GridBagConstraints gbc = new GridBagConstraints();
        gbc.insets = new Insets( top: 10,  left: 10,  bottom: 10,  right: 10); // Padding

        // Title
        JLabel titleLabel = new JLabel( text: "Login to Smart City");
        titleLabel.setFont(new Font( name: "Arial", Font.BOLD,  size: 20));
        gbc.gridx = 0;
        gbc.gridy = 0;
        gbc.gridwidth = 2;
        loginPanel.add(titleLabel, gbc);


        // Username Label
        gbc.gridwidth = 1;
        gbc.gridx = 0;
        gbc.gridy = 1;
        loginPanel.add(new JLabel( text: "Username:"), gbc);

        // Username Field
        JTextField userField = new JTextField( columns: 15);
        gbc.gridx = 1;
        loginPanel.add(userField, gbc);

        // Password Label
        gbc.gridx = 0;
        gbc.gridy = 2;
        loginPanel.add(new JLabel( text: "Password:"), gbc);

        // Password Field
        JPasswordField passField = new JPasswordField( columns: 15);
        gbc.gridx = 1;
        loginPanel.add(passField, gbc);

        // Sign In Button
        JButton loginButton = new JButton( text: "Sign In");
        loginButton.setBackground(new Color( r: 0,  g: 102,  b: 204)); // Blue color
        loginButton.setForeground(Color.WHITE);
        loginButton.setFocusPainted(false);
        gbc.gridx = 0;
        gbc.gridy = 3;
        gbc.gridwidth = 2;
        loginPanel.add(loginButton, gbc);
```

```
125
126         // Sign Up Button
127         JButton signupButton = new JButton( text: "Sign Up");
128         signupButton.setBackground(new Color( r: 0,  g: 204,  b: 102)); // Green color
129         signupButton.setForeground(Color.WHITE);
130         signupButton.setFocusPainted(false);
131         gbc.gridy = 4;
132         loginPanel.add(signupButton, gbc);
133
134         // Sign In Action
135         loginButton.addActionListener( ActionEvent e -> {
136             String username = userField.getText();
137             String password = new String(passField.getPassword());
138
139             if (validateUser(username, password)) {
140                 JOptionPane.showMessageDialog( parentComponent: null,  message: "Login Successful!");
141                 dispose(); // Close Welcome Window
142                 mainFrame.showMainUI(); // Open MainFrame
143             } else {
144                 JOptionPane.showMessageDialog( parentComponent: null,  message: "Incorrect Username or Password!",  title: "
145             }
146         });
147
148         // Open SignUp Window
149         signupButton.addActionListener( ActionEvent e -> new SignUp());
150     }
151
152
153
154     private boolean validateUser(String username, String password) {  1 usage
```

```
154     private boolean validateUser(String username, String password) {  1 usage
155         try (Connection con = DBConnection.getConnection()) {
156             assert con != null;
157             try (PreparedStatement pst = con.prepareStatement( sql: "SELECT * FROM users WHERE username=? AND password
158                 pst.setString( parameterIndex: 1, username);
159                 pst.setString( parameterIndex: 2, password);
160                 ResultSet rs = pst.executeQuery();
161                 return rs.next(); // Returns true if user exists
162             }
163         } catch (SQLException ex) {
164             Logger.getLogger(Welcome.class.getName()).log(Level.SEVERE,  msg: "Database error", ex);
165             return false;
166         }
167     }
168 }
```

HomePage.java

0.3 KBps

```java
package com.smartcity;

import javax.swing.*;
import java.awt.*;

public class HomePage extends JPanel {  2 usages
    public HomePage(MainFrame mainFrame) {  1 usage
        setLayout(new GridBagLayout());
        setBackground(new Color( r: 230,  g: 245,  b: 255)); // Light background

        // **Card Panel for Centered Content**
        JPanel cardPanel = new JPanel() {
            @Override
            protected void paintComponent(Graphics g) {
                super.paintComponent(g);
                Graphics2D g2d = (Graphics2D) g;
                int width = getWidth();
                int height = getHeight();
                Color color1 = new Color( r: 180,  g: 220,  b: 255);
                Color color2 = new Color( r: 120,  g: 180,  b: 255);
                GradientPaint gp = new GradientPaint( x1: 0,  y1: 0, color1, width, height, color2);
                g2d.setPaint(gp);
                g2d.fillRoundRect( x: 0,  y: 0, width, height,  arcWidth: 30,  arcHeight: 30); // Rounded Corners
            }
        };
        cardPanel.setPreferredSize(new Dimension( width: 450,  height: 300));
        cardPanel.setLayout(new GridBagLayout());
        cardPanel.setOpaque(false);

        GridBagConstraints gbc = new GridBagConstraints();
        gbc.insets = new Insets( top: 10,  left: 10,  bottom: 10,  right: 10);
        gbc.fill = GridBagConstraints.BOTH;

        // **Title**
        JLabel titleLabel = new JLabel( text: "Smart City Guide", SwingConstants.CENTER);
        titleLabel.setFont(new Font( name: "Arial", Font.BOLD,  size: 26));
        titleLabel.setForeground(new Color( r: 10,  g: 10,  b: 70));
        gbc.gridx = 0;
        gbc.gridy = 0;
        gbc.gridwidth = 2;
        cardPanel.add(titleLabel, gbc);

        // **Buttons with Styling**
        JButton cityInfoButton = createStyledButton( text: "City Information");
        JButton tourismButton = createStyledButton( text: "Tourism");
        JButton transportButton = createStyledButton( text: "Transportation");
        JButton shoppingButton = createStyledButton( text: "Shopping Malls");

        gbc.gridwidth = 1;
        gbc.gridy = 1;
        gbc.gridx = 0;
        cardPanel.add(cityInfoButton, gbc);

        gbc.gridx = 1;
        cardPanel.add(tourismButton, gbc);

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

        gbc.gridx = 1;
        cardPanel.add(shoppingButton, gbc);

        add(cardPanel);
```

```
        // **Button Actions**
        cityInfoButton.addActionListener( ActionEvent e -> new CityInfo().setVisible(true));
        tourismButton.addActionListener( ActionEvent e -> new Tourism().setVisible(true));
        transportButton.addActionListener( ActionEvent e -> new Transportation().setVisible(true));
        shoppingButton.addActionListener( ActionEvent e -> new ShoppingMalls().setVisible(true));
    }

    // **Create a Styled Button**
    private JButton createStyledButton(String text) {  4 usages
        JButton button = new JButton(text);
        button.setFont(new Font( name: "Arial", Font.BOLD,  size: 16));
        button.setBackground(new Color( r: 0,  g: 102,  b: 204));
        button.setForeground(Color.WHITE);
        button.setFocusPainted(false);
        button.setBorder(BorderFactory.createEmptyBorder( top: 10,  left: 20,  bottom: 10,  right: 20));
        button.setPreferredSize(new Dimension( width: 180,  height: 45));
        button.setCursor(new Cursor(Cursor.HAND_CURSOR));

        // **3D Button Effect**
        button.setBorder(BorderFactory.createRaisedSoftBevelBorder());

        // **Hover Effect**
        button.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseEntered(java.awt.event.MouseEvent evt) {
                button.setBackground(new Color( r: 0,  g: 153,  b: 255));
                button.setBorder(BorderFactory.createLoweredSoftBevelBorder());
            }
            public void mouseExited(java.awt.event.MouseEvent evt) {
                button.setBackground(new Color( r: 0,  g: 102,  b: 204));
                button.setBorder(BorderFactory.createRaisedSoftBevelBorder());
            }
        });

        return button;
    }
}
```

CityInfo.java ×

4.1 KBps

1

```
package com.smartcity;

import javax.swing.*;
import javax.swing.border.EmptyBorder;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;

public class CityInfo extends JFrame {
    private final JTextField searchField;  5 usages
    private final JLabel cityNameLabel;  5 usages
    private final JTextPane infoPane;  8 usages
```

```java

    public CityInfo() {  2 usages
        setTitle("City Information");
        setSize( width: 850,  height: 550);
        setLocationRelativeTo(null);
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setLayout(new BorderLayout());

        // Header Panel (Title)
        JPanel headerPanel = new JPanel();
        headerPanel.setBackground(new Color( r: 30,  g: 144,  b: 255)); // Dodger Blue
        headerPanel.setPreferredSize(new Dimension(getWidth(),  height: 60));

        cityNameLabel = new JLabel( text: "CITY INFORMATION", SwingConstants.CENTER);
        cityNameLabel.setFont(new Font( name: "Arial", Font.BOLD,  size: 26));
        cityNameLabel.setForeground(Color.WHITE);
        headerPanel.add(cityNameLabel);
        add(headerPanel, BorderLayout.NORTH);

        // Main Content Panel
        JPanel mainPanel = new JPanel(new BorderLayout());
        mainPanel.setBorder(new EmptyBorder( top: 20,  left: 20,  bottom: 20,  right: 20));
        add(mainPanel);

        // Info Pane (City Details)
        infoPane = new JTextPane();
        infoPane.setContentType("text/html");
        infoPane.setEditable(false);
        infoPane.setFont(new Font( name: "Arial", Font.PLAIN,  size: 16));
        infoPane.setBorder(BorderFactory.createLineBorder(Color.LIGHT_GRAY,  thickness: 1));
        JScrollPane scrollPane = new JScrollPane(infoPane);
        mainPanel.add(scrollPane, BorderLayout.CENTER);

        // Search Panel
        JPanel searchPanel = new JPanel(new FlowLayout(FlowLayout.CENTER,  hgap: 10,  vgap: 10));
        searchPanel.setBackground(new Color( r: 245,  g: 245,  b: 245)); // Light Gray Background

        searchField = new JTextField( columns: 20);
        searchField.setFont(new Font( name: "Arial", Font.PLAIN,  size: 14));
        searchField.setBorder(BorderFactory.createCompoundBorder(
                BorderFactory.createLineBorder(new Color( r: 100,  g: 149,  b: 237),  thickness: 2), // Cornflower Blue Bor
                BorderFactory.createEmptyBorder( top: 5,  left: 10,  bottom: 5,  right: 10)
        ));

        JButton searchButton = new JButton( text: "Search");
        searchButton.setFont(new Font( name: "Arial", Font.BOLD,  size: 14));
        searchButton.setBackground(new Color( r: 0,  g: 123,  b: 255));
        searchButton.setForeground(Color.WHITE);
        searchButton.setFocusPainted(false);
        searchButton.setBorder(BorderFactory.createEmptyBorder( top: 8,  left: 15,  bottom: 8,  right: 15));

        searchPanel.add(new JLabel( text: "Enter City: "));
        searchPanel.add(searchField);
        searchPanel.add(searchButton);
        mainPanel.add(searchPanel, BorderLayout.SOUTH);

        // Search Action
        searchButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                searchCity();
            }
        });
```

```
80      }
81
82      private void searchCity() {  1 usage
83          String cityName = searchField.getText().trim();
84          if (cityName.isEmpty()) {
85              JOptionPane.showMessageDialog( parentComponent: this,  message: "Please enter a city name.",  title: "Error", JO
86              return;
87          }
88
89          try (Connection conn = DBConnection.getConnection()) {
90              assert conn != null;
91              try (PreparedStatement stmt = conn.prepareStatement( sql: "SELECT description, population, emergency_conta
92
93                  stmt.setString( parameterIndex: 1, cityName);
94                  ResultSet rs = stmt.executeQuery();
95
96                  if (rs.next()) {
97                      String description = rs.getString( columnLabel: "description");
98                      int population = rs.getInt( columnLabel: "population");
99                      String emergencyContacts = rs.getString( columnLabel: "emergency_contacts");
100
101                     // Update Title & Text
102                     cityNameLabel.setText(cityName.toUpperCase());
103                     String displayText = "<html><div style='text-align:left;'>"
104                             + "<h2 style='color:#1E90FF;'>Population: " + population + "</h2>"
105                             + "<p style='font-size:16px;'>" + description + "</p>"
106                             + "<div style='background-color:#FFD700; padding:10px; border-radius:5px;'>"
107                             + "<b>Emergency Contacts:</b> " + emergencyContacts + "</div>"
108                             + "</div></html>";
109
110                     infoPane.setText(displayText);
111                 } else {
112                     infoPane.setText("<html><h3 style='color:red;'>No information found for: " + cityName + "</h3></
113                 }
114
115             }
116         } catch (SQLException ex) {
117             JOptionPane.showMessageDialog( parentComponent: this,  message: "Database error occurred. Please try again.",
118         }
119     }
120
121     public static void main(String[] args) {
122         SwingUtilities.invokeLater(() -> new CityInfo().setVisible(true));
123     }
124 }
125
```

Tourism.java

0.1 KBps

```
1   package com.smartcity;
2
3   import javax.swing.*;
4   import java.awt.*;
5   import java.awt.event.ActionEvent;
6   import java.awt.event.ActionListener;
7
8   public class Tourism extends JFrame {
9       public Tourism() {  2 usages
10          setTitle("Tourism - Smart City");
11          setSize( width: 600,  height: 400);
12          setLocationRelativeTo(null);
13          setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
14          setLayout(new BorderLayout());
15
```

5

```java
        // Header Panel
        JPanel headerPanel = new JPanel();
        headerPanel.setBackground(new Color( r: 0,  g: 102,  b: 204));
        JLabel titleLabel = new JLabel( text: "Explore Tourism");
        titleLabel.setFont(new Font( name: "Arial", Font.BOLD,  size: 26));
        titleLabel.setForeground(Color.WHITE);
        headerPanel.add(titleLabel);
        add(headerPanel, BorderLayout.NORTH);

        // Center Panel with Buttons
        JPanel centerPanel = new JPanel();
        centerPanel.setLayout(new GridLayout( rows: 3,  cols: 1,  hgap: 10,  vgap: 10));
        centerPanel.setBorder(BorderFactory.createEmptyBorder( top: 30,  left: 50,  bottom: 30,  right: 50));

        JButton btnHotels = createStyledButton( text: "Hotels");
        JButton btnPlaces = createStyledButton( text: "Places to Visit");
        JButton btnRestaurants = createStyledButton( text: "Restaurants");

        centerPanel.add(btnHotels);
        centerPanel.add(btnPlaces);
        centerPanel.add(btnRestaurants);

        add(centerPanel, BorderLayout.CENTER);

        // Button Actions
        btnHotels.addActionListener( ActionEvent e -> new Hotels().setVisible(true));
        btnPlaces.addActionListener( ActionEvent e -> new TouristPlaces().setVisible(true));
        btnRestaurants.addActionListener( ActionEvent e -> new Restaurants().setVisible(true));
    }

    // Method to create styled buttons
    private JButton createStyledButton(String text) {  3 usages
        JButton button = new JButton(text);
        button.setFont(new Font( name: "Arial", Font.BOLD,  size: 16));
        button.setBackground(new Color( r: 0,  g: 153,  b: 255));
        button.setForeground(Color.WHITE);
        button.setFocusPainted(false);
        button.setBorder(BorderFactory.createEmptyBorder( top: 10,  left: 20,  bottom: 10,  right: 20));

        // Hover Effect
        button.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseEntered(java.awt.event.MouseEvent evt) {
                button.setBackground(new Color( r: 0,  g: 102,  b: 204));
            }
            public void mouseExited(java.awt.event.MouseEvent evt) {
                button.setBackground(new Color( r: 0,  g: 153,  b: 255));
            }
        });

        return button;
    }

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

Hotels.java

```java
package com.smartcity;

import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

public class Hotels extends JFrame {
    private final JTextField searchField;  3 usages
    private final DefaultTableModel model;  5 usages

    public Hotels() {  2 usages
        setTitle("Hotel Search");
        setSize( width: 700,  height: 400);
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setLocationRelativeTo(null);

        // Panel for Search
        JPanel searchPanel = new JPanel();
        searchPanel.add(new JLabel( text: "Enter Location: "));
        searchField = new JTextField( columns: 15);
        searchPanel.add(searchField);
        JButton searchButton = new JButton( text: "Search");
        searchPanel.add(searchButton);

        // Table Model Setup
        model = new DefaultTableModel();
        model.setColumnIdentifiers(new String[]{"Hotel Name", "Location", "Contact", "Price Range", "Rating"});
        JTable table = new JTable(model);
        JScrollPane scrollPane = new JScrollPane(table);

        // Layout
        setLayout(new BorderLayout());
        add(searchPanel, BorderLayout.NORTH);
        add(scrollPane, BorderLayout.CENTER);

        // Action Listener for Search Button
        searchButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                searchHotels();
            }
        });
    }

    private void searchHotels() {  1 usage
        String location = searchField.getText().trim();

        if (location.isEmpty()) {
            JOptionPane.showMessageDialog( parentComponent: this,  message: "Please enter a location!",  title: "Warning", JO
            return;
        }

        try {
            Connection con = DBConnection.getConnection();
            String sql = "SELECT hotel_name, location, contact, price_range, rating FROM hotels WHERE location LIKE ?"
            assert con != null;
            PreparedStatement pst = con.prepareStatement(sql);
            pst.setString( parameterIndex: 1,  x: "%" + location + "%");

            ResultSet rs = pst.executeQuery();
            model.setRowCount(0); // Clear previous results
```

```
67
68              boolean found = false;
69              while (rs.next()) {
70                  model.addRow(new Object[]{
71                          rs.getString( columnLabel: "hotel_name"),
72                          rs.getString( columnLabel: "location"),
73                          rs.getString( columnLabel: "contact"),
74                          rs.getString( columnLabel: "price_range"),
75                          rs.getFloat( columnLabel: "rating")
76                  });
77                  found = true;
78              }
79
80              if (!found) {
81                  JOptionPane.showMessageDialog( parentComponent: this,  message: "No hotels found in this location.",  title:
82              }
83
84              rs.close();
85              pst.close();
86              con.close();
87          } catch (Exception ex) {
88              JOptionPane.showMessageDialog( parentComponent: this,  message: "Error fetching data: " + ex.getMessage(),  titl
89          }
90      }
91
92      public static void main(String[] args) {
93          SwingUtilities.invokeLater(() -> new Hotels().setVisible(true));
94      }
95  }
96
```

TouristPlaces.java

3.2 KBps

```
1   package com.smartcity;
2
3   import javax.swing.*;
4   import javax.swing.table.DefaultTableModel;
5   import java.awt.*;
6   import java.awt.event.ActionEvent;
7   import java.awt.event.ActionListener;
8   import java.sql.Connection;
9   import java.sql.PreparedStatement;
10  import java.sql.ResultSet;
11
12  public class TouristPlaces extends JFrame {
13      private final JTextField searchField;  3 usages
14      private final DefaultTableModel model;  5 usages
15
16      public TouristPlaces() {  2 usages
17          setTitle("Tourist Places");
18          setSize( width: 800,  height: 400);
19          setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
20          setLocationRelativeTo(null);
21
22          // Search Panel
23          JPanel searchPanel = new JPanel();
24          searchPanel.add(new JLabel( text: "Enter Location: "));
25          searchField = new JTextField( columns: 15);
26          searchPanel.add(searchField);
27          JButton searchButton = new JButton( text: "Search");
28          searchPanel.add(searchButton);
29
30          // Table Setup
31          model = new DefaultTableModel();
32          model.setColumnIdentifiers(new String[]{"Place Name", "Location", "Description", "Entry Fee", "Opening Hours"
33          JTable table = new JTable(model);
34          JScrollPane scrollPane = new JScrollPane(table);
```

```
35
36          // Layout
37          setLayout(new BorderLayout());
38          add(searchPanel, BorderLayout.NORTH);
39          add(scrollPane, BorderLayout.CENTER);
40
41          // Search Button Action
42          searchButton.addActionListener(new ActionListener() {
43              @Override
44              public void actionPerformed(ActionEvent e) {
45                  searchTouristPlaces();
46              }
47          });
48      }
49
50      private void searchTouristPlaces() {  1 usage
51          String location = searchField.getText().trim();
52
53          if (location.isEmpty()) {
54              JOptionPane.showMessageDialog( parentComponent: this,  message: "Please enter a location!",  title: "Warning", JO
55              return;
56          }
57
58          try {
59              Connection con = DBConnection.getConnection();
60              String sql = "SELECT place_name, location, description, entry_fee, opening_hours FROM tourism WHERE locat
61              assert con != null;
62              PreparedStatement pst = con.prepareStatement(sql);
63              pst.setString( parameterIndex: 1,  x: "%" + location + "%");
64
65              ResultSet rs = pst.executeQuery();
66              model.setRowCount(0); // Clear previous results
67
68              boolean found = false;
69              while (rs.next()) {
70                  model.addRow(new Object[]{
71                          rs.getString( columnLabel: "place_name"),
72                          rs.getString( columnLabel: "location"),
73                          rs.getString( columnLabel: "description"),
74                          rs.getDouble( columnLabel: "entry_fee"),
75                          rs.getString( columnLabel: "opening_hours")
76                  });
77                  found = true;
78              }
79
80              if (!found) {
81                  JOptionPane.showMessageDialog( parentComponent: this,  message: "No tourist places found in this location.
82              }
83
84              rs.close();
85              pst.close();
86              con.close();
87          } catch (Exception ex) {
88              JOptionPane.showMessageDialog( parentComponent: this,  message: "Error fetching data: " + ex.getMessage(),  tit
89          }
90      }
91
92      public static void main(String[] args) {
93          SwingUtilities.invokeLater(() -> new TouristPlaces().setVisible(true));
94      }
95  }
96
```

Restaurants.java

```java
package com.smartcity;

import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;

public class Restaurants extends JFrame {
    private final JTextField searchField;  3 usages
    private final DefaultTableModel model;  5 usages

    public Restaurants() {  2 usages
        setTitle("Restaurants");
        setSize( width: 800,  height: 400);
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setLocationRelativeTo(null);

        // Search Panel
        JPanel searchPanel = new JPanel();
        searchPanel.add(new JLabel( text: "Enter Location: "));
        searchField = new JTextField( columns: 15);
        searchPanel.add(searchField);
        JButton searchButton = new JButton( text: "Search");
        searchPanel.add(searchButton);

        // Table Setup
        model = new DefaultTableModel();
        model.setColumnIdentifiers(new String[]{"Restaurant Name", "Location", "Cuisine", "Price Range", "Contact"});
        JTable table = new JTable(model);
        JScrollPane scrollPane = new JScrollPane(table);

        // Layout
        setLayout(new BorderLayout());
        add(searchPanel, BorderLayout.NORTH);
        add(scrollPane, BorderLayout.CENTER);

        // Search Button Action
        searchButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                searchRestaurants();
            }
        });
    }

    private void searchRestaurants() {  1 usage
        String location = searchField.getText().trim();

        if (location.isEmpty()) {
            JOptionPane.showMessageDialog( parentComponent: this,  message: "Please enter a location!",  title: "Warning", JO
            return;
        }

        try {
            Connection con = DBConnection.getConnection();
            String sql = "SELECT restaurant_name, location, cuisine, price_range, contact FROM restaurants WHERE loca
            PreparedStatement pst = con.prepareStatement(sql);
            pst.setString( parameterIndex: 1,  x: "%" + location + "%");

            ResultSet rs = pst.executeQuery();
            model.setRowCount(0); // Clear previous results
```

```
            boolean found = false;
            while (rs.next()) {
                model.addRow(new Object[]{
                        rs.getString( columnLabel: "restaurant_name"),
                        rs.getString( columnLabel: "location"),
                        rs.getString( columnLabel: "cuisine"),
                        rs.getString( columnLabel: "price_range"),
                        rs.getString( columnLabel: "contact")
                });
                found = true;
            }

            if (!found) {
                JOptionPane.showMessageDialog( parentComponent: this,  message: "No restaurants found in this location.",
            }

            rs.close();
            pst.close();
            con.close();
        } catch (Exception ex) {
            JOptionPane.showMessageDialog( parentComponent: this,  message: "Error fetching data: " + ex.getMessage(),  tit
        }
    }

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

Transportation.java ×

0.5 KBps

```
package com.smartcity;

import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.util.Objects;

public class Transportation extends JFrame {
    private final JComboBox<String> pickupField;  4 usages
    private final JComboBox<String> destinationField;  4 usages
    private final JPanel transportPanel;  8 usages

    public Transportation() {  2 usages
        setTitle("Find Your Ride");
        setSize( width: 600,  height: 500);
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setLocationRelativeTo(null);
        setLayout(new BorderLayout());

        // Title Label
        JLabel titleLabel = new JLabel( text: "Find Your Transportation", SwingConstants.CENTER);
        titleLabel.setFont(new Font( name: "Arial", Font.BOLD,  size: 24));
        titleLabel.setBorder(BorderFactory.createEmptyBorder( top: 10,  left: 0,  bottom: 10,  right: 0));
        add(titleLabel, BorderLayout.NORTH);

        // Input Panel
        JPanel inputPanel = new JPanel(new GridBagLayout());
        inputPanel.setBorder(BorderFactory.createEmptyBorder( top: 10,  left: 20,  bottom: 10,  right: 20));
        GridBagConstraints gbc = new GridBagConstraints();
        gbc.gridx = 0;
```

```
        gbc.gridy = 0;
        gbc.anchor = GridBagConstraints.WEST;
        gbc.insets = new Insets( top: 5,  left: 5,  bottom: 5,  right: 5);

        inputPanel.add(new JLabel( text: "🚉 Pickup Location:"), gbc);
        gbc.gridx = 1;
        pickupField = new JComboBox<>(new String[]{"Select", "Central Station", "Airport", "Metro Station", "Railway
        pickupField.setPreferredSize(new Dimension( width: 200,  height: 30));
        inputPanel.add(pickupField, gbc);

        gbc.gridx = 0;
        gbc.gridy = 1;
        inputPanel.add(new JLabel( text: "🎯 Destination:"), gbc);
        gbc.gridx = 1;
        destinationField = new JComboBox<>(new String[]{"Select", "Downtown", "City Center", "TechVille", "Springfie
        destinationField.setPreferredSize(new Dimension( width: 200,  height: 30));
        inputPanel.add(destinationField, gbc);

        // Find Ride Button
        JButton findRideBtn = new JButton( text: "Find Ride");
        findRideBtn.setFont(new Font( name: "Arial", Font.BOLD,  size: 16));
        findRideBtn.setBackground(new Color( r: 30,  g: 144,  b: 255)); // Blue color
        findRideBtn.setForeground(Color.WHITE);
        findRideBtn.setFocusPainted(false);
        findRideBtn.setPreferredSize(new Dimension( width: 200,  height: 40));
        findRideBtn.setBorder(BorderFactory.createLineBorder(Color.BLACK,  thickness: 2));

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

        add(inputPanel, BorderLayout.CENTER);

        // Transport Panel
        transportPanel = new JPanel();
        transportPanel.setLayout(new GridLayout( rows: 0,  cols: 1,  hgap: 10,  vgap: 10));
        transportPanel.setBorder(BorderFactory.createTitledBorder("🚖 Available Transport Options"));
        JScrollPane scrollPane = new JScrollPane(transportPanel);
        scrollPane.setPreferredSize(new Dimension( width: 580,  height: 200));
        add(scrollPane, BorderLayout.SOUTH);

        // Button Action
        findRideBtn.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String pickup = (String) pickupField.getSelectedItem();
                String destination = (String) destinationField.getSelectedItem();
                assert pickup != null;
                if (pickup.equals("Select") || Objects.equals(destination,  b: "Select")) {
                    JOptionPane.showMessageDialog( parentComponent: null,  message: "Please select both pickup and destin
                } else {
                    fetchTransportOptions(pickup, destination);
                }
            }
        });

        setVisible(true);
    }

    private void fetchTransportOptions(String pickup, String destination) {  1 usage
        transportPanel.removeAll();
        try {
            Connection conn = DBConnection.getConnection();
            String query = "SELECT * FROM transportation WHERE pickup_point = ? AND destination = ?";
```

```
12  public class Transportation extends JFrame {                                    ⚠1 ⚠1 ✓1
94      private void fetchTransportOptions(String pickup, String destination) {  1 usage
99              assert conn != null;
100             PreparedStatement pstmt = conn.prepareStatement(query);
101             pstmt.setString( parameterIndex: 1, pickup);
102             pstmt.setString( parameterIndex: 2, destination);
103             ResultSet rs = pstmt.executeQuery();
104
105             boolean dataFound = false;
106
107             while (rs.next()) {
108                 dataFound = true;
109                 String transportType = rs.getString( columnLabel: "transport_type");
110                 double fare = rs.getDouble( columnLabel: "fare");
111                 String bookingMethod = rs.getString( columnLabel: "booking_method");
112
113                 // Transport Option Card
114                 JPanel transportCard = new JPanel(new BorderLayout());
115                 transportCard.setBorder(BorderFactory.createLineBorder(Color.GRAY,  thickness: 1));
116                 transportCard.setBackground(new Color( r: 245,  g: 245,  b: 245));
117                 transportCard.setPreferredSize(new Dimension( width: 550,  height: 50));
118
119                 JLabel transportLabel = new JLabel( text: "🚖 " + transportType, SwingConstants.LEFT);
120                 transportLabel.setFont(new Font( name: "Arial", Font.BOLD,  size: 14));
121
122                 JLabel fareLabel = new JLabel( text: "💰 Fare: $" + fare, SwingConstants.CENTER);
123                 JLabel bookingLabel = new JLabel( text: "📅 Booking: " + bookingMethod, SwingConstants.RIGHT);
124
125                 transportCard.add(transportLabel, BorderLayout.WEST);
126                 transportCard.add(fareLabel, BorderLayout.CENTER);
127                 transportCard.add(bookingLabel, BorderLayout.EAST);
128
129                 transportPanel.add(transportCard);
130             }
131
132             if (!dataFound) {
133                 JOptionPane.showMessageDialog( parentComponent: null,  message: "No transport options available for the s
134             }
135
136             transportPanel.revalidate();
137             transportPanel.repaint();
138
139             conn.close();
140         } catch (Exception ex) {
141             JOptionPane.showMessageDialog( parentComponent: null,  message: "Error fetching data: " + ex.getMessage());
142         }
143     }
144
145     public static void main(String[] args) {
146         new Transportation();
147     }
148 }
149
```

ShoppingMalls.java

0.9 KBps

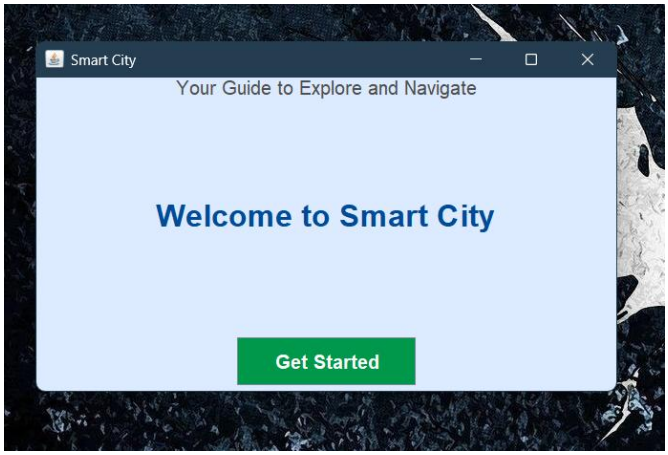
```java
package com.smartcity;

import javax.swing.*;
import javax.swing.table.DefaultTableModel;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.sql.*;

public class ShoppingMalls extends JFrame {
    private final JTextField searchField;  5 usages
    private final DefaultTableModel tableModel;  4 usages

    public ShoppingMalls() {  2 usages
        setTitle("Shopping Malls");
        setSize( width: 700,  height: 450);
        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
        setLayout(new BorderLayout());
        getContentPane().setBackground(new Color( r: 240,  g: 248,  b: 255)); // Light blue background

        // Search Panel
        JPanel searchPanel = new JPanel();
        searchPanel.setBackground(new Color( r: 200,  g: 220,  b: 240));
        searchPanel.setBorder(BorderFactory.createEmptyBorder( top: 10,  left: 10,  bottom: 10,  right: 10));
        searchPanel.setLayout(new FlowLayout());

        JLabel searchLabel = new JLabel( text: "Search by Location:");
        searchLabel.setFont(new Font( name: "Arial", Font.BOLD,  size: 14));
        searchField = new JTextField( columns: 15);
        searchField.setFont(new Font( name: "Arial", Font.PLAIN,  size: 14));
        searchField.setBorder(BorderFactory.createLineBorder(Color.GRAY,  thickness: 1));

        JButton searchButton = new JButton( text: "Search");
        searchButton.setFont(new Font( name: "Arial", Font.BOLD,  size: 14));
        searchButton.setBackground(new Color( r: 70,  g: 130,  b: 180)); // Steel Blue
        searchButton.setForeground(Color.WHITE);
        searchButton.setFocusPainted(false);
        searchButton.setBorder(BorderFactory.createEmptyBorder( top: 5,  left: 10,  bottom: 5,  right: 10));

        searchPanel.add(searchLabel);
        searchPanel.add(searchField);
        searchPanel.add(searchButton);
        add(searchPanel, BorderLayout.NORTH);

        // Table Panel
        tableModel = new DefaultTableModel(new String[]{"Mall Name", "Location", "Opening Time", "Closing Time"},  row
        JTable table = new JTable(tableModel);
        table.setRowHeight(25);
        table.getTableHeader().setFont(new Font( name: "Arial", Font.BOLD,  size: 14));
        table.getTableHeader().setBackground(new Color( r: 70,  g: 130,  b: 180));
        table.getTableHeader().setForeground(Color.WHITE);
        table.setFont(new Font( name: "Arial", Font.PLAIN,  size: 14));
        table.setSelectionBackground(new Color( r: 173,  g: 216,  b: 230));

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

        // Search action
        searchButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                String location = searchField.getText().trim();
                searchMalls(location);
            }
        });

        setVisible(true);
```

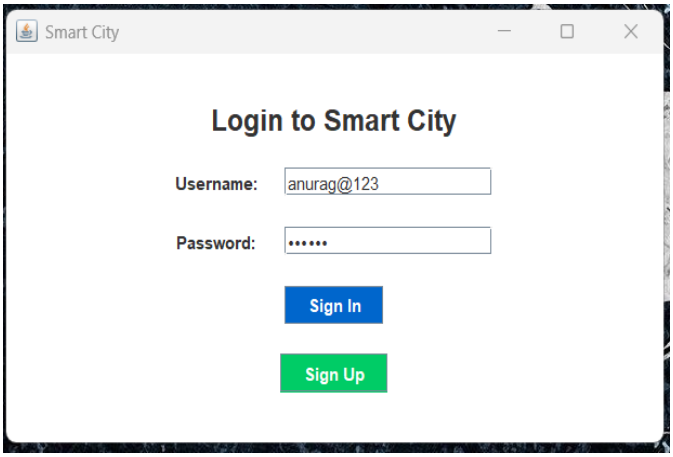
```
66          setVisible(true);
67      }
68
69      private void searchMalls(String location) {  1 usage
70          tableModel.setRowCount(0); // Clear previous results
71
72          try (Connection conn = DBConnection.getConnection()) {
73              assert conn != null;
74              try (PreparedStatement stmt = conn.prepareStatement( sql: "SELECT mall_name, location, opening_time, closin
75
76                  stmt.setString( parameterIndex: 1,  x: "%" + location + "%");
77                  ResultSet rs = stmt.executeQuery();
78
79                  while (rs.next()) {
80                      tableModel.addRow(new Object[]{rs.getString( columnLabel: "mall_name"), rs.getString( columnLabel: "loc
81                  }
82              }
83          } catch (SQLException ex) {
84              JOptionPane.showMessageDialog( parentComponent: this,  message: "Database Error: " + ex.getMessage(),  title: "Er
85          }
86      }
87
88      public static void main(String[] args) {
89          new ShoppingMalls();
90      }
91  }
92
```

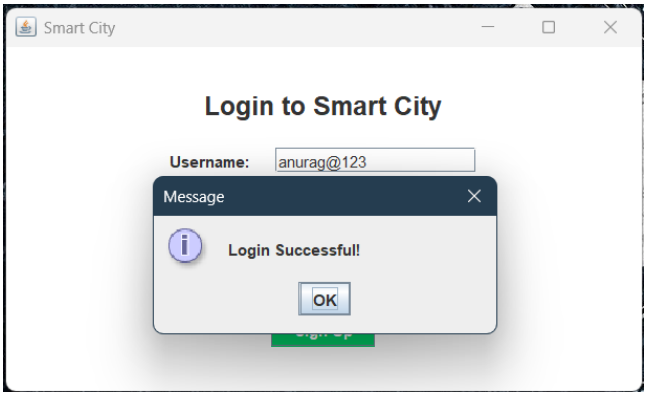
DBConnection.java

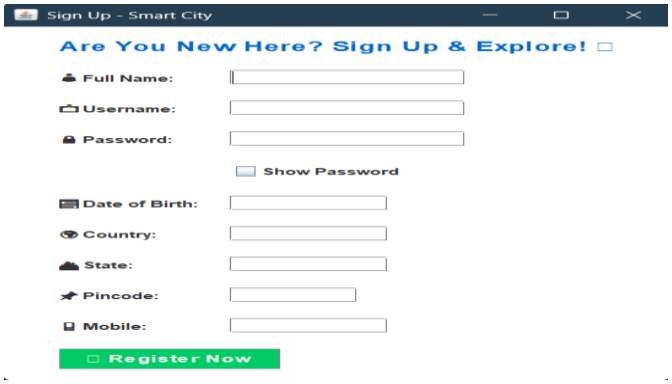
```
1   package com.smartcity;
2
3   import java.sql.Connection;
4   import java.sql.DriverManager;
5   import java.sql.SQLException;
6   import java.util.logging.Level;
7   import java.util.logging.Logger;
8
9   public class DBConnection {  9 usages
10      private static final String URL = "jdbc:mysql://localhost:3306/smartcity";  1 usage
11      private static final String USER = "root";  1 usage
12      private static final String PASSWORD = "614599";  1 usage
13
14      private static final Logger LOGGER = Logger.getLogger(DBConnection.class.getName());  1 usage
15
16      public static Connection getConnection() {  8 usages
17          try {
18              // Load MySQL JDBC Driver
19              Class.forName( className: "com.mysql.cj.jdbc.Driver");
20
21              // Establish connection
22              return DriverManager.getConnection(URL, USER, PASSWORD);
23          } catch (ClassNotFoundException | SQLException e) {
24              LOGGER.log(Level.SEVERE,  msg: "Database connection failed!", e);
25              return null; // Return null if connection fails
26          }
27      }
28  }
29
```

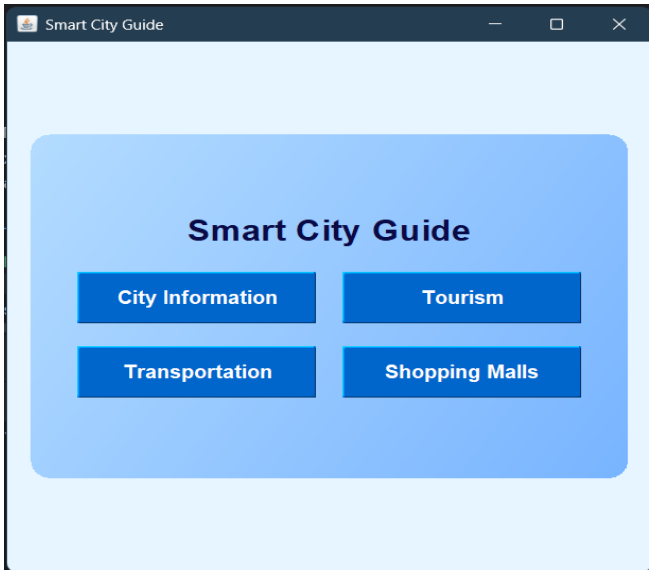
8. Output Screenshots

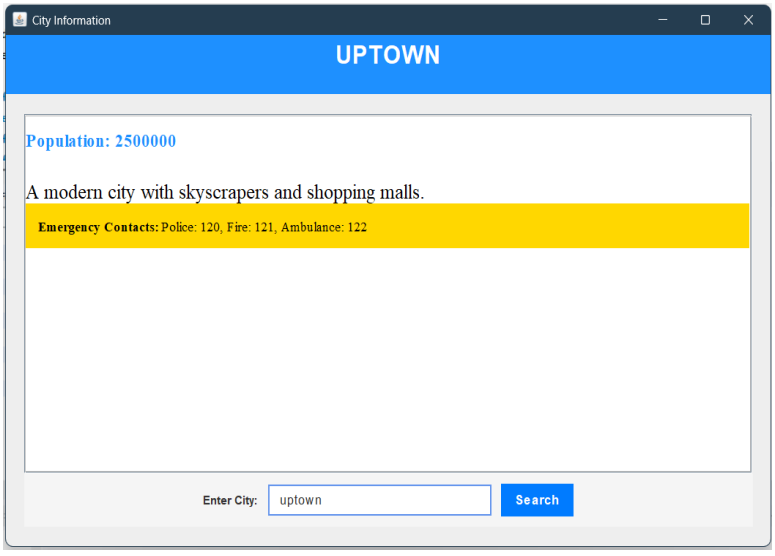


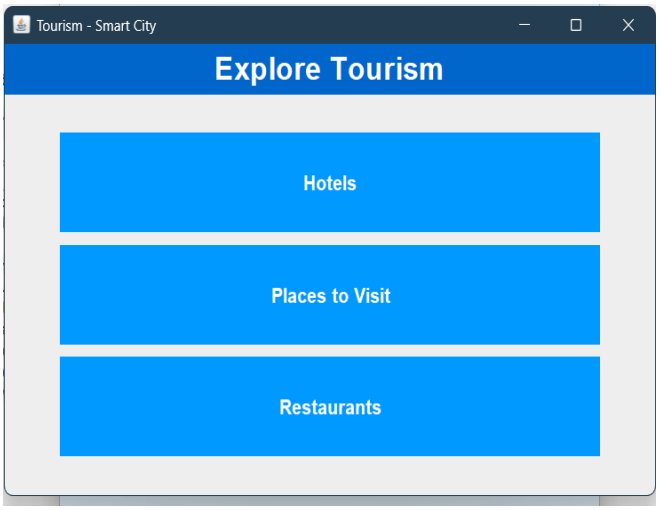


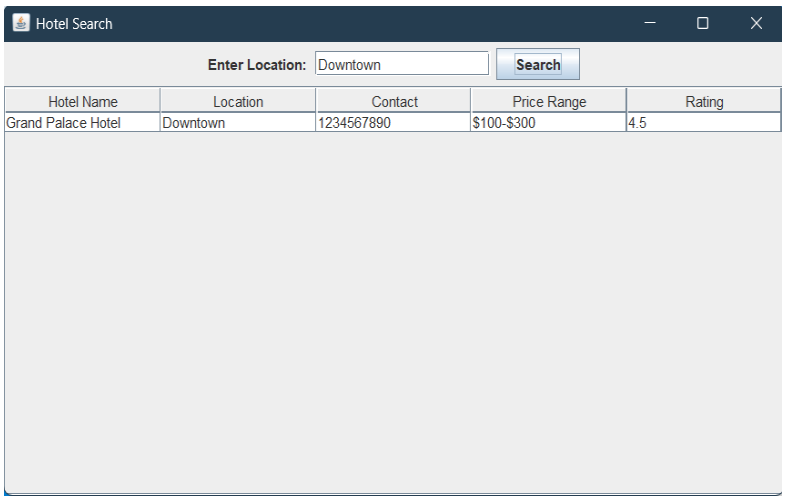


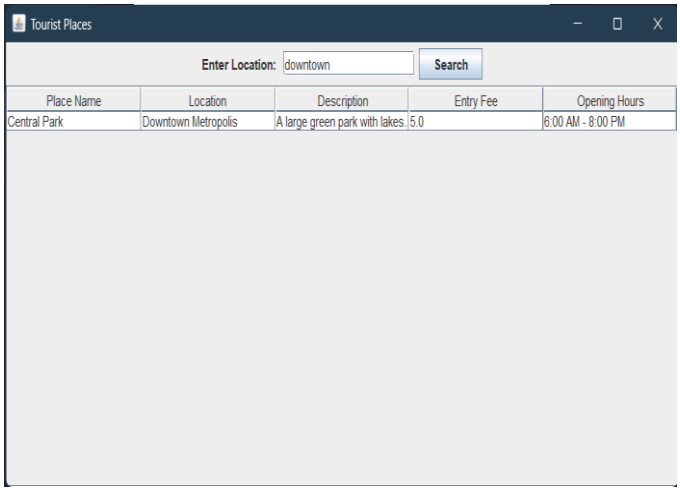


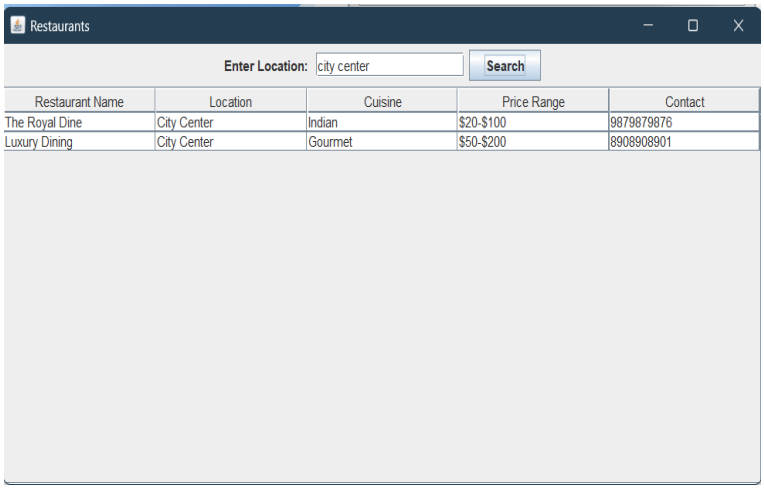


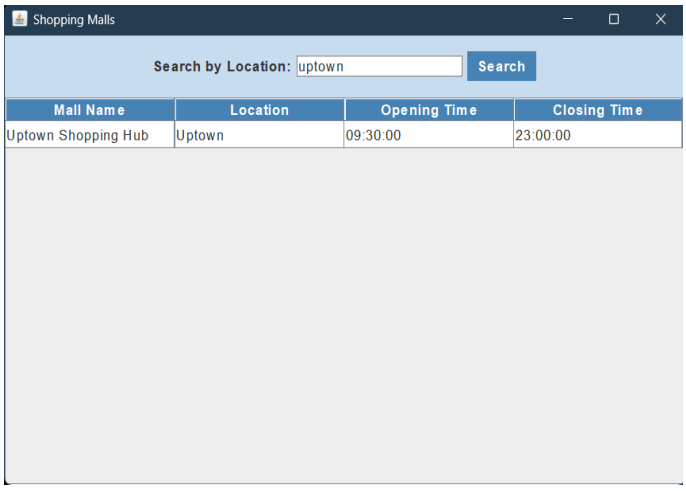


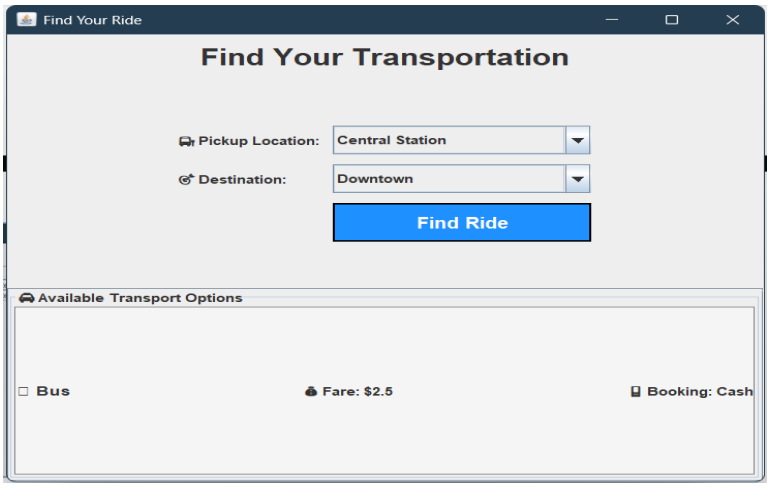

Tourism - Smart City
Explore Tourism
Hotels
Places to Visit
Restaurants


Hotel Search
Enter Location: Downtown
Search
Hotel Name | Location | Contact | Price Range | Rating
Grand Palace Hotel | Downtown | 1234567890 | $100-$300 | 4.5


Tourist Places
Enter Location: downtown
Search
Place Name | Location | Description | Entry Fee | Opening Hours
Central Park | Downtown Metropolis | A large green park with lakes. | 5.0 | 6:00 AM - 8:00 PM


Restaurants
Enter Location: city center
Search
Restaurant Name | Location | Cuisine | Price Range | Contact
The Royal Dine | City Center | Indian | $20-$100 | 9879879876
Luxury Dining | City Center | Gourmet | $50-$200 | 8908908901


Shopping Malls
Search by Location: uptown
Search
Mall Name | Location | Opening Time | Closing Time
Uptown Shopping Hub | Uptown | 09:30:00 | 23:00:00


Find Your Ride
Find Your Transportation
Pickup Location: Central Station
Destination: Downtown
Find Ride
Available Transport Options
Bus
Fare: $2.5
Booking: Cash

9. Project Summary

The Smart City Application successfully integrates city information, tourism, transportation, and shopping services into a single platform. The application is built using Java Swing and MySQL, allowing users to search for places dynamically. The enhanced UI design, especially in the Transportation section, provides an experience similar to modern ride-booking apps.

10. Important Learnings

During the development of this project, I gained valuable insights into:
Java Swing UI Development
MySQL Database Integration
JDBC for Data Fetching
Object-Oriented Programming (OOP) Concepts
UI/UX Enhancement Techniques
Designing an Interactive Transportation System

Conclusion

The **Smart City Guide Application** successfully provides an interactive platform for users to explore essential city services, including **tourism, transportation, hotels, restaurants, and shopping malls**. Through an intuitive Java Swing interface and a well-structured MySQL database, the application ensures smooth navigation and efficient data retrieval. Users can search for locations, view important details, and make informed decisions based on real-time information.

This project not only enhanced our understanding of **Object-Oriented Programming (OOP) in Java** but also provided hands-on experience in **database integration, UI design, and application development**. Future improvements may include **real-time booking options, user authentication, and advanced search features** to further enhance user engagement. Overall, this project serves as a solid foundation for building smart city solutions and improving urban navigation experiences.