

**EXPERIENCExchange - TIME-BASED ACCESS
PASSES FOR UNIQUE EXPERIENCES**

A

MINI PROJECT REPORT

for

JAVA - OBJECT ORIENTED PROGRAMMING LAB

Submitted by

DIVYA HIREN SAVLA Roll No. 49

DHARMIK VYAS Roll No. 62



DEPARTMENT OF COMPUTER ENGINEERING

SHAH & ANCHOR KUTCHHI ENGINEERING COLLEGE

(An Autonomous Institute Affiliated to University of Mumbai)

MUMBAI - 400 088, MAHARASHTRA (INDIA)

2024

EXPERIENCEXCHANGE - TIME-BASED ACCESS PASSES FOR UNIQUE EXPERIENCES

Introduction

The modern experience economy has created a demand for unique, curated, and local experiences that go beyond conventional ticketed events. This report presents **ExperienceXchange**, a platform designed to facilitate the buying, selling, and booking of **time-based access passes** for non-ticketed experiences such as workshops, guided tours, classes, and personal coaching sessions. The focus of the platform is on fair pricing, verified hosts, and a transparent booking process.

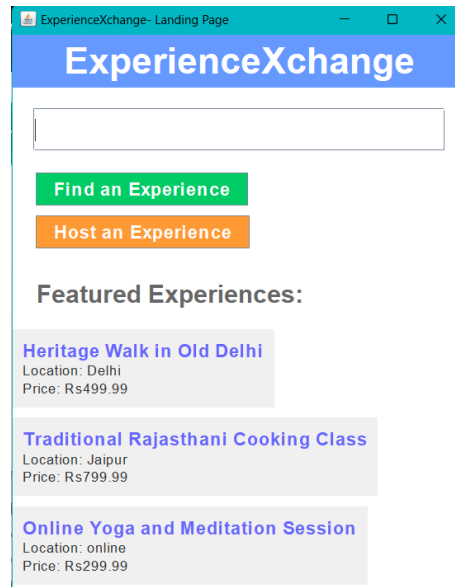
Objectives

The primary objective of **ExperienceXchange** is to provide a user-friendly, secure, and dynamic platform where hosts can offer, and users can discover and book, both local and online experiences. The key goals include:

- Offering a seamless discovery and booking system for unique experiences.
- Providing a robust host verification and management system.
- Ensuring secure payment processing and transparency for users and hosts.

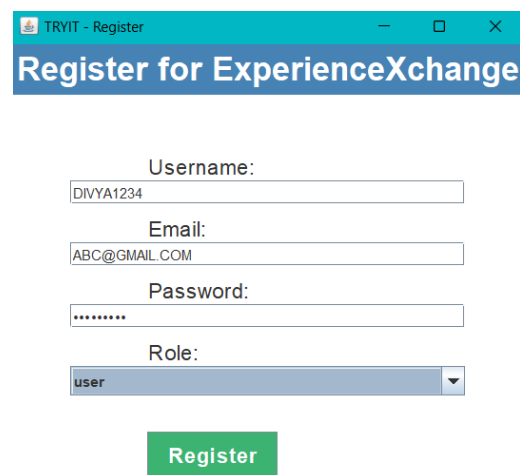
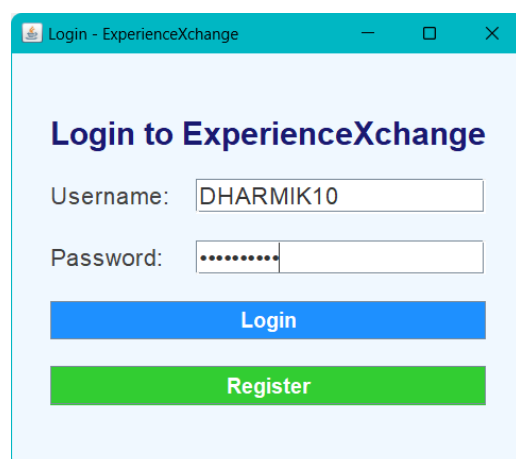
Features and Pages

Landing Page



The landing page serves as the primary gateway, featuring the top experiences and offering easy access to both users and hosts via intuitive buttons like *Find an Experience* and *Host an Experience*.

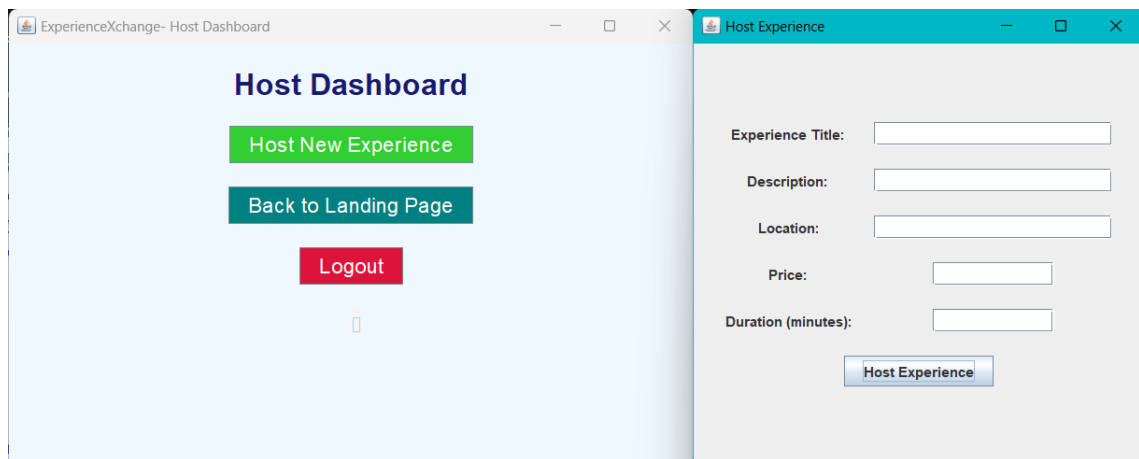
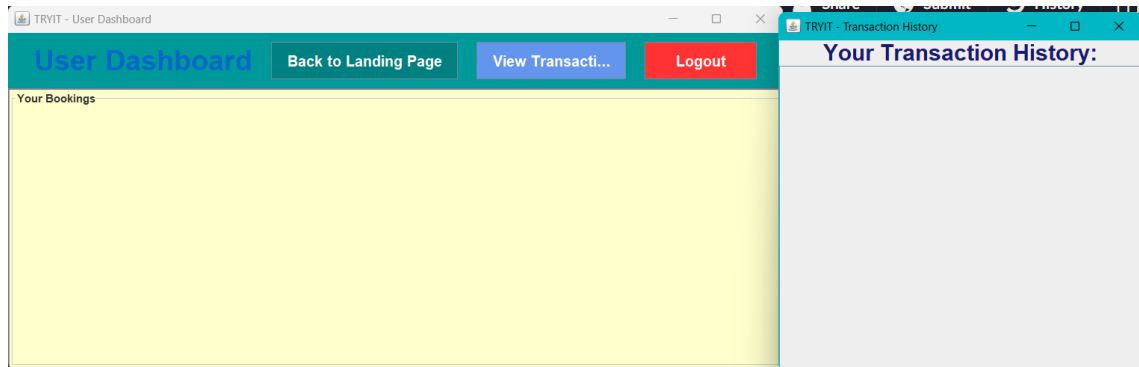
User Authentication



- **Login/Signup Pages:** Users and hosts can sign up using email(Google, Facebook).

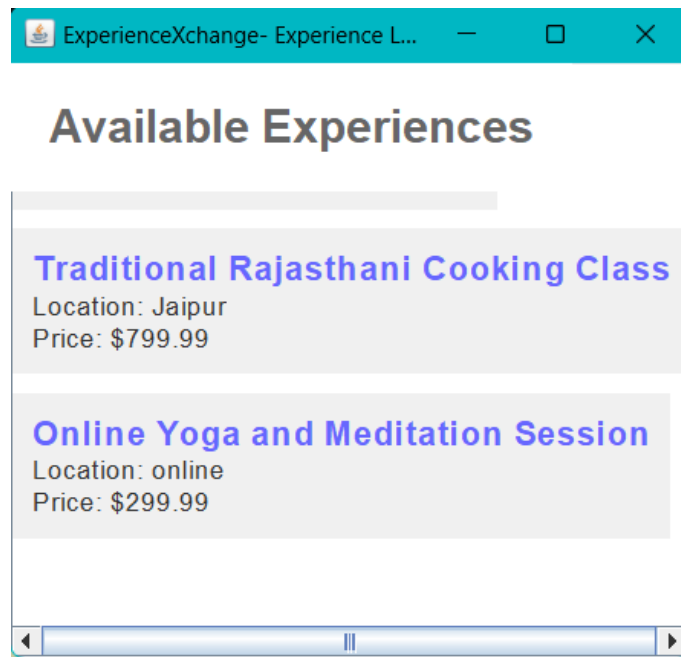
- **User Profiles:** Profiles for both user and hosts to manage bookings, listings, and schedules.

Dashboard



- **User Dashboard:** Displays booked experiences and upcoming schedules.
- **Host Dashboard:** Allows hosts to manage their listed experiences, update availability, and track bookings.

Experience Listing Page



Hosts can list new experiences by adding details like type, location, price, and date/time, along with images or videos.

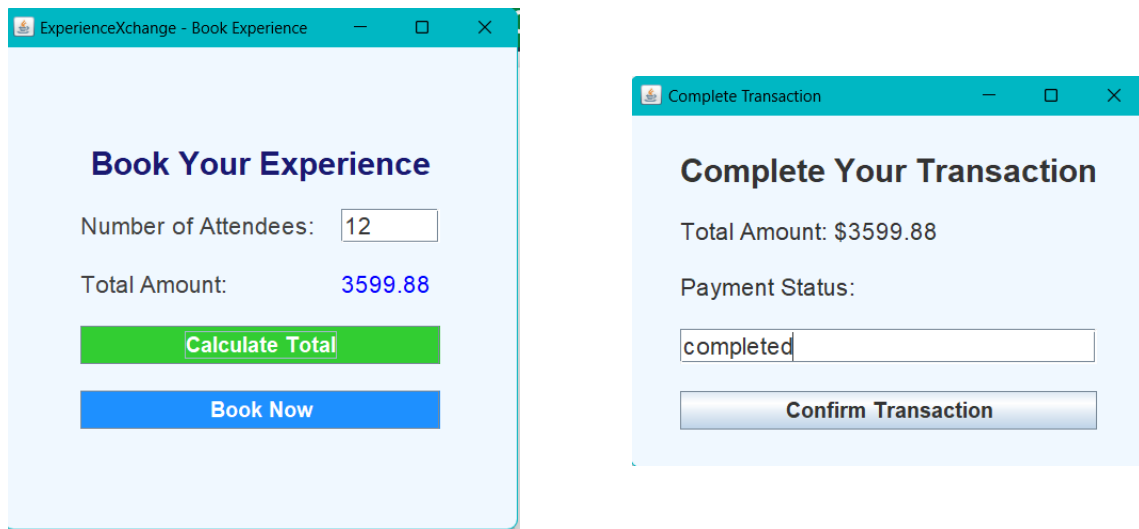
Search and Browse Experiences

Users can explore experiences based on location, type, or date using filters and a dynamic search bar.

Host an Experience Page

Each experience has a dedicated page detailing the activity, date, time, price, location (or virtual link if online), and host information.

Booking and Transaction



Book Your Experience
Number of Attendees:
Total Amount: **3599.88**
Calculate Total
Book Now

Complete Your Transaction
Total Amount: \$3599.88
Payment Status:
Confirm Transaction

A seamless booking process allows users to select the number of attendees and customize the experience (if applicable) before proceeding to a secure checkout.

Transaction History

Both users and hosts can access their transaction history to view past bookings, completed sessions, and payment details.

Database Structure

The platform's database is structured to store user information, experience listings, bookings, and transactions, ensuring an organized and efficient retrieval process.

Users Table

Field	Type	Description
id	INT	Primary key
username	VARCHAR(50)	Unique username
email	VARCHAR(100)	User email
password	VARCHAR(255)	Hashed password
role	ENUM('user', 'host')	Role of the user
date_created	TIMESTAMP	Account creation date

Experiences Table

Field	Type	Description
id	INT	Primary key
title	VARCHAR(100)	Experience name
description	TEXT	Detailed description
location	VARCHAR(100)	Location (or "online" if virtual)
host_id	INT	Foreign key referencing Users.id
price	DECIMAL(10,2)	Price per session
duration	INT	Duration in minutes
date_listed	TIMESTAMP	Date experience was listed

Bookings Table

Field	Type	Description
id	INT	Primary key
experience_id	INT	Foreign key referencing Experiences.id
user_id	INT	Foreign key referencing Users.id
booking_date	TIMESTAMP	Date and time of booking
attendees	INT	Number of attendees
total_amount	DECIMAL(10,2)	Total amount paid

Transactions Table

Field	Type	Description
id	INT	Primary key
booking_id	INT	Foreign key referencing Bookings.id
payment_status	ENUM('pending', 'completed')	Payment status
transaction_date	TIMESTAMP	Date of transaction
amount	DECIMAL(10,2)	Total price

SELECT * FROM users LIMIT... X						
Max. rows: 100 Fetched Rows: 6						
#	id	username	email	password	role	date_created
1	1	RaviKumar	ravikumar@hotmail.com	1234567890	host	2024-10-09 00:35:40.000
2	2	AnitaSharma	anitasharma@yahoo.com	0987654321	user	2024-10-09 00:35:40.000
3	3	SanjayVerma	sanjayverma@gmail.com	12233445566	host	2024-10-09 00:35:40.000
4	4	DIVYA_1234	SDIVYA721.127@GMAIL.COM	1234	user	2024-10-14 23:58:36.000
5	5	DHARMIK10	DHARMIK@XYZ.COM	1234567890	host	2024-10-16 10:55:43.000

SELECT * FROM experiences... X									
Max. rows: 100 Fetched Rows: 3 Matching Rows:									
#	id	title	description	location	host_id	price	duration	date_listed	
1	1	Heritage Walk in Old Delhi	Explore the rich history and hidden gems of O. Delhi		1	499.99	120	2024-10-09 00:35:40.000	
2	2	Traditional Rajasthani Cooking Class	Heritage Walk in Old Delhi	asthani dishes with. Jaipur	3	799.99	180	2024-10-09 00:35:40.000	
3	3	Online Yoga and Meditation Session	Join a virtual session on Yoga and meditation.	online	1	299.99	60	2024-10-09 00:35:40.000	

SELECT * FROM bookings LI... X							
Max. rows: 100 Fetched Rows: 3 Matching Rows:							
#	id	experience_id	user_id	booking_date	attendees	total_amount	
1	1	1	2	2024-10-05 10:00:00.000	1	499.99	
2	2	2	2	2024-10-05 11:00:00.000	2	1599.98	
3	3	3	2	2024-10-05 12:00:00.000	1	299.99	

SELECT * FROM transaction... X						
Max. rows: 100 Fetched Rows: 3 Matching Rows:						
#	id	booking_id	payment_status	transaction_date	amount	
1	1	1	completed	2024-10-05 10:05:00.000	499.99	
2	2	2	completed	2024-10-05 11:10:00.000	1599.98	
3	3	3	pending	2024-10-05 12:10:00.000	299.99	

Implementation code

LoginPage

```

1 package UI;
2
3 import javax.swing.*;
4 import java.awt.*;
5 import java.awt.event.ActionEvent;
6 import java.awt.event.ActionListener;
7 import java.sql.Connection;
8 import java.sql.PreparedStatement;
9 import java.sql.ResultSet;
10
11 public class LoginPage extends JFrame {
12
13     public LoginPage() {
14         initComponents();
15     }
16
17     private void initComponents() {
18         setTitle("Login - ExperienceXchange");
19         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
20
21         // Main Panel
22         JPanel mainPanel = new JPanel();
23         mainPanel.setBackground(new Color(240, 248, 255)); // Light
24             Blue Background
25         mainPanel.setLayout(new GridBagLayout());
26         GridBagConstraints gbc = new GridBagConstraints();
27         gbc.insets = new Insets(10, 10, 10, 10);
28         gbc.fill = GridBagConstraints.HORIZONTAL;
29
30         // Title
31         JLabel titleLabel = new JLabel("Login to ExperienceXchange",
            JLabel.CENTER);
32         titleLabel.setFont(new Font("Arial", Font.BOLD, 24));

```



```

32     titleLabel.setForeground(new Color(25, 25, 112)); //
        Midnight blue
33
34     gbc.gridx = 0;
35     gbc.gridy = 0;
36     gbc.gridwidth = 2;
37     mainPanel.add(titleLabel, gbc);
38
39     // Username Label & Field
40     JLabel usernameLabel = new JLabel("Username:");
41     usernameLabel.setFont(new Font("Arial", Font.PLAIN, 18));
42     usernameLabel.setForeground(Color.DARK_GRAY);
43
44     JTextField usernameField = new JTextField(15);
45     usernameField.setFont(new Font("Arial", Font.PLAIN, 18));
46
47     gbc.gridy = 1;
48     gbc.gridwidth = 1;
49     mainPanel.add(usernameLabel, gbc);
50
51     gbc.gridx = 1;
52     mainPanel.add(usernameField, gbc);
53
54     // Password Label & Field
55     JLabel passwordLabel = new JLabel("Password:");
56     passwordLabel.setFont(new Font("Arial", Font.PLAIN, 18));
57     passwordLabel.setForeground(Color.DARK_GRAY);
58
59     JPasswordField passwordField = new JPasswordField(15);
60     passwordField.setFont(new Font("Arial", Font.PLAIN, 18));
61
62     gbc.gridx = 0;
63     gbc.gridy = 2;
64     mainPanel.add(passwordLabel, gbc);
65
66     gbc.gridx = 1;
67     mainPanel.add(passwordField, gbc);
68
69     // Login Button
70     JButton loginButton = new JButton("Login");
71     loginButton.setFont(new Font("Arial", Font.BOLD, 16));
72     loginButton.setBackground(new Color(30, 144, 255)); //
        Dodger Blue
73     loginButton.setForeground(Color.WHITE);
74     loginButton.addActionListener(new ActionListener() {
75         @Override
76         public void actionPerformed(ActionEvent e) {
77             login(usernameField.getText(), new String(
78                 passwordField.getPassword()));
79         }
80     });
81
82     gbc.gridx = 0;
83     gbc.gridy = 3;
84     gbc.gridwidth = 2;
85     mainPanel.add(loginButton, gbc);
86
87     // Register Button

```

```

87     JButton registerButton = new JButton("Register");
88     registerButton.setFont(new Font("Arial", Font.BOLD, 16));
89     registerButton.setBackground(new Color(50, 205, 50)); //
        Lime Green
90     registerButton.setForeground(Color.WHITE);
91     registerButton.addActionListener(e -> openRegisterPage());
92
93     gbc.gridy = 4; // Positioning the Register button below the
        Login button
94     mainPanel.add(registerButton, gbc);
95
96     add(mainPanel);
97     setSize(400, 350);
98     setLocationRelativeTo(null); // Center the frame
99     setVisible(true);
100 }
101
102 private void login(String username, String password) {
103     try (Connection conn = DatabaseConnection.getConnection())
104     {
105         String sql = "SELECT id, role FROM Users WHERE username
            = ? AND password = ?";
106         PreparedStatement stmt = conn.prepareStatement(sql);
107         stmt.setString(1, username);
108         stmt.setString(2, password); // You should hash
            passwords for security
109         ResultSet rs = stmt.executeQuery();
110
111         if (rs.next()) {
112             int userId = rs.getInt("id");
113             String role = rs.getString("role");
114
115             JOptionPane.showMessageDialog(this, "Login
                Successful!");
116
117             // Open the appropriate dashboard based on the user
                's role
118             if ("user".equalsIgnoreCase(role)) {
119                 new UserDashboard(userId).setVisible(true);
120             } else if ("host".equalsIgnoreCase(role)) {
121                 new HostDashboard(userId).setVisible(true); //
                    Ensure you have a HostDashboard class
122             }
123
124             this.dispose(); // Close the login window
125         } else {
126             JOptionPane.showMessageDialog(this, "Invalid
                username or password.", "Login Error",
                JOptionPane.ERROR_MESSAGE);
127         }
128     } catch (Exception e) {
129         JOptionPane.showMessageDialog(this, "Error during login
            : " + e.getMessage(), "Error", JOptionPane.
            ERROR_MESSAGE);
130     }
131
132 private void openRegisterPage() {

```

```

133         new RegisterPage().setVisible(true); // Open the
           RegisterPage
134     }
135
136     public static void main(String[] args) {
137         SwingUtilities.invokeLater(LoginPage::new);
138     }
139 }
140
141 }

```

RegisterPage

```

1 package UI;
2 import UI.DatabaseConnection;
3 import java.sql.Connection;
4 import java.sql.PreparedStatement;
5 import javax.swing.*;
6 import java.awt.*;
7
8 public class RegisterPage extends JFrame {
9     private JTextField usernameField, emailField;
10    private JPasswordField passwordField;
11    private JComboBox<String> roleComboBox;
12
13    public RegisterPage() {
14        initComponents();
15    }
16
17    private void initComponents() {
18        setTitle("TRYIT - Register");
19        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
20        setSize(500, 600);
21        setLocationRelativeTo(null);
22        setLayout(new BorderLayout());
23
24        JPanel topPanel = new JPanel();
25        topPanel.setBackground(new Color(70, 130, 180));
26        JLabel titleLabel = new JLabel("Register for
27            ExperienceXchange");
28        titleLabel.setFont(new Font("Arial", Font.BOLD, 28));
29        titleLabel.setForeground(Color.WHITE);
30        topPanel.add(titleLabel);
31        add(topPanel, BorderLayout.NORTH);
32
33        JPanel formPanel = new JPanel();
34        formPanel.setLayout(new BoxLayout(formPanel, BoxLayout.
35            Y_AXIS));
36        formPanel.setBackground(Color.WHITE);
37        formPanel.setBorder(BorderFactory.createEmptyBorder(20, 50,
38            20, 50));
39
40        JLabel usernameLabel = new JLabel("Username:");
41        usernameLabel.setFont(new Font("Arial", Font.PLAIN, 18));
42        usernameField = new JTextField(20);
43        usernameField.setMaximumSize(new Dimension(Integer.
44            MAX_VALUE, 40));

```

```

41
42 JLabel emailLabel = new JLabel("Email:");
43 emailLabel.setFont(new Font("Arial", Font.PLAIN, 18));
44 emailField = new JTextField(20);
45 emailField.setMaximumSize(new Dimension(Integer.MAX_VALUE,
46     40));
47
48 JLabel passwordLabel = new JLabel("Password:");
49 passwordLabel.setFont(new Font("Arial", Font.PLAIN, 18));
50 passwordField = new JPasswordField(20);
51 passwordField.setMaximumSize(new Dimension(Integer.
52     MAX_VALUE, 40));
53
54 JLabel roleLabel = new JLabel("Role:");
55 roleLabel.setFont(new Font("Arial", Font.PLAIN, 18));
56 roleComboBox = new JComboBox<>(new String[]{"user", "host"
57     });
58 roleComboBox.setMaximumSize(new Dimension(Integer.MAX_VALUE
59     , 40));
60
61 JButton registerButton = new JButton("Register");
62 registerButton.setFont(new Font("Arial", Font.BOLD, 18));
63 registerButton.setBackground(new Color(60, 179, 113));
64 registerButton.setForeground(Color.WHITE);
65 registerButton.setPreferredSize(new Dimension(120, 40));
66 registerButton.addActionListener(e -> register());
67
68 formPanel.add(Box.createVerticalStrut(30));
69 formPanel.add(usernameLabel);
70 formPanel.add(usernameField);
71 formPanel.add(Box.createVerticalStrut(10));
72 formPanel.add(emailLabel);
73 formPanel.add(emailField);
74 formPanel.add(Box.createVerticalStrut(10));
75 formPanel.add(passwordLabel);
76 formPanel.add(passwordField);
77 formPanel.add(Box.createVerticalStrut(10));
78 formPanel.add(roleLabel);
79 formPanel.add(roleComboBox);
80 formPanel.add(Box.createVerticalStrut(30));
81 formPanel.add(registerButton);
82
83 add(formPanel, BorderLayout.CENTER);
84 pack();
85
86 }
87
88 private void register() {
89     String username = usernameField.getText();
90     String email = emailField.getText();
91     String password = new String(passwordField.getPassword());
92     String role = (String) roleComboBox.getSelectedItem();
93     if (username.isEmpty() || email.isEmpty() || password.
94         isEmpty()) {
95         JOptionPane.showMessageDialog(this, "All fields are
96             required", "Error", JOptionPane.ERROR_MESSAGE);
97         return;
98     }
99 }

```

```

92         try (Connection conn = DatabaseConnection.getConnection())
93         {
94             String sql = "INSERT INTO Users (username, email,
95                 password, role) VALUES (?, ?, ?, ?)";
96             PreparedStatement stmt = conn.prepareStatement(sql);
97             stmt.setString(1, username);
98             stmt.setString(2, email);
99             stmt.setString(3, password);
100             stmt.setString(4, role);
101             stmt.executeUpdate();
102             JOptionPane.showMessageDialog(this, "Registration
103                 successful!");
104             new LoginPage().setVisible(true);
105             this.dispose();
106         } catch (Exception e) {
107             JOptionPane.showMessageDialog(this, "Error: " + e.
108                 getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
109         }
110     }
111 }

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

```

LandingPage

```

1 package UI;
2 // LandingPage.javapackage UI;
3 import UI.DatabaseConnection;
4 import java.sql.Connection;
5 import java.sql.PreparedStatement;
6 import java.sql.ResultSet;
7 import javax.swing.*;
8 import java.awt.*;
9
10 public class LandingPage extends JFrame {
11     private int userId;
12
13     public LandingPage(int userId) {
14         this.userId = userId;
15         initComponents();
16         fetchFeaturedExperiences();
17     }
18
19     private void initComponents() {
20         setTitle("ExperienceXchange - Landing Page");
21         setDefaultCloseOperation(EXIT_ON_CLOSE);
22         setSize(600, 600);
23         setLocationRelativeTo(null);
24         setLayout(new BorderLayout());
25
26         // Header Panel
27         JPanel headerPanel = new JPanel();
28         headerPanel.setBackground(new Color(102, 153, 255));

```

```

29     JLabel logoLabel = new JLabel("ExperienceXchange");
30     logoLabel.setFont(new Font("Arial", Font.BOLD, 36));
31     logoLabel.setForeground(Color.WHITE);
32     headerPanel.add(logoLabel);
33     add(headerPanel, BorderLayout.NORTH);
34
35     // Main Content Panel
36     JPanel mainPanel = new JPanel();
37     mainPanel.setLayout(new BoxLayout(mainPanel, BoxLayout.
38         Y_AXIS));
39     mainPanel.setBackground(Color.WHITE);
40     mainPanel.setBorder(BorderFactory.createEmptyBorder(20, 20,
41         20, 20));
42
43     // Search Field
44     JTextField searchField = new JTextField(20);
45     searchField.setFont(new Font("Arial", Font.PLAIN, 18));
46     searchField.setMaximumSize(new Dimension(Integer.MAX_VALUE,
47         50));
48
49     // Buttons
50     JButton findExperienceButton = new JButton("Find an
51         Experience");
52     findExperienceButton.setFont(new Font("Arial", Font.BOLD,
53         18));
54     findExperienceButton.setBackground(new Color(0, 204, 102));
55     findExperienceButton.setForeground(Color.WHITE);
56     findExperienceButton.addActionListener(e -> new
57         ExperienceListingPage(userId).setVisible(true));
58
59     JButton hostExperienceButton = new JButton("Host an
60         Experience");
61     hostExperienceButton.setFont(new Font("Arial", Font.BOLD,
62         18));
63     hostExperienceButton.setBackground(new Color(255, 153, 51))
64         ;
65     hostExperienceButton.setForeground(Color.WHITE);
66     hostExperienceButton.addActionListener(e -> new
67         RegisterPage().setVisible(true));
68
69     // Featured Experiences Label
70     JLabel featuredLabel = new JLabel("Featured Experiences:");
71     featuredLabel.setFont(new Font("Arial", Font.BOLD, 24));
72     featuredLabel.setForeground(new Color(102, 102, 102));
73
74     // Adding Components to Main Panel
75     mainPanel.add(searchField);
76     mainPanel.add(Box.createVerticalStrut(20));
77     mainPanel.add(findExperienceButton);
78     mainPanel.add(Box.createVerticalStrut(10));
79     mainPanel.add(hostExperienceButton);
80     mainPanel.add(Box.createVerticalStrut(30));
81     mainPanel.add(featuredLabel);
82
83     add(mainPanel, BorderLayout.CENTER);
84     pack();
85 }

```

```

77     private void fetchFeaturedExperiences() {
78         try (Connection conn = DatabaseConnection.getConnection())
79         {
80             String sql = "SELECT title, location, price FROM
81                 Experiences ORDER BY date_listed DESC LIMIT 5";
82             PreparedStatement stmt = conn.prepareStatement(sql);
83             ResultSet rs = stmt.executeQuery();
84
85             // Panel to hold featured experiences
86             JPanel featuredPanel = new JPanel();
87             featuredPanel.setLayout(new BorderLayout(featuredPanel,
88                 BorderLayout.Y_AXIS));
89             featuredPanel.setBackground(Color.WHITE);
90
91             while (rs.next()) {
92                 String title = rs.getString("title");
93                 String location = rs.getString("location");
94                 double price = rs.getDouble("price");
95
96                 // Experience panel
97                 JPanel experiencePanel = new JPanel();
98                 experiencePanel.setBackground(new Color(240, 240,
99                     240));
100                 experiencePanel.setLayout(new BorderLayout(
101                     experiencePanel, BorderLayout.Y_AXIS));
102                 experiencePanel.setBorder(BorderFactory.
103                     createEmptyBorder(10, 10, 10, 10));
104
105                 JLabel titleLabel = new JLabel(title);
106                 titleLabel.setFont(new Font("Arial", Font.BOLD, 18)
107                     );
108                 titleLabel.setForeground(new Color(102, 102, 255));
109
110                 JLabel locationLabel = new JLabel("Location: " +
111                     location);
112                 locationLabel.setFont(new Font("Arial", Font.PLAIN,
113                     14));
114
115                 JLabel priceLabel = new JLabel("Price: Rs" + price)
116                     ;
117                 priceLabel.setFont(new Font("Arial", Font.PLAIN,
118                     14));
119
120                 experiencePanel.add(titleLabel);
121                 experiencePanel.add(locationLabel);
122                 experiencePanel.add(priceLabel);
123                 experiencePanel.setAlignmentX(Component.
124                     LEFT_ALIGNMENT);
125
126                 experiencePanel.addMouseListener(new java.awt.event
127                     .MouseAdapter() {
128                     public void mouseClicked(java.awt.event.
129                         MouseEvent evt) {
130                         new ExperienceListingPage(userId).
131                             setVisible(true);
132                     }
133                 });
134             }
135         }
136     }

```

```

120         featuredPanel.add(experiencePanel);
121         featuredPanel.add(Box.createVerticalStrut(10));
122     }
123
124     // Adding featured experiences to main panel
125     add(featuredPanel, BorderLayout.SOUTH);
126     revalidate();
127 } catch (Exception e) {
128     JOptionPane.showMessageDialog(this, "Error fetching
        experiences: " + e.getMessage(), "Error",
        JOptionPane.ERROR_MESSAGE);
129 }
130 }
131
132 public static void main(String[] args) {
133     SwingUtilities.invokeLater(() -> {
134         try (Connection conn = DatabaseConnection.getConnection
            ()) {
135             String sql = "SELECT id FROM Users WHERE username =
                'your_username'"; // Replace 'your_username'
                with the actual username
136             PreparedStatement stmt = conn.prepareStatement(sql)
                ;
137             ResultSet rs = stmt.executeQuery();
138             if (rs.next()) {
139                 int userId = rs.getInt("id");
140                 new LandingPage(userId).setVisible(true);
141             }
142         } catch (Exception e) {
143             e.printStackTrace();
144         }
145     });
146 }
147 }

```

HostDashboard

```

1 package UI;
2
3 import javax.swing.*;
4 import java.awt.*;
5 import java.sql.*;
6
7 public class HostDashboard extends JFrame {
8     private int hostId;
9     private JPanel experiencePanel; // Declare experiencePanel as
        an instance variable
10
11     public HostDashboard(int hostId) {
12         this.hostId = hostId;
13         initComponents();
14         fetchHostedExperiences();
15     }
16
17     private void initComponents() {
18         setTitle("ExperienceXchange - Host Dashboard");

```



```

19     setSize(600, 600);
20     setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
21     setLocationRelativeTo(null);
22
23     JPanel panel = new JPanel();
24     panel.setLayout(new BoxLayout(panel, BoxLayout.Y_AXIS));
25     panel.setBackground(new Color(240, 248, 255));
26
27     JLabel dashboardLabel = new JLabel("Host Dashboard");
28     dashboardLabel.setFont(new Font("Arial", Font.BOLD, 26));
29     dashboardLabel.setAlignmentX(Component.CENTER_ALIGNMENT);
30     dashboardLabel.setForeground(new Color(25, 25, 112));
31
32     JButton hostExperienceButton = new JButton("Host New
33         Experience");
34     hostExperienceButton.setFont(new Font("Arial", Font.PLAIN,
35         18));
36     hostExperienceButton.setBackground(new Color(50, 205, 50));
37     hostExperienceButton.setForeground(Color.WHITE);
38     hostExperienceButton.setFocusPainted(false);
39     hostExperienceButton.setAlignmentX(Component.
40         CENTER_ALIGNMENT);
41     hostExperienceButton.setPreferredSize(new Dimension(200,
42         50));
43     hostExperienceButton.addActionListener(e -> new
44         HostExperiencePage(hostId).setVisible(true));
45
46     JButton logOutButton = new JButton("Logout");
47     logOutButton.setFont(new Font("Arial", Font.PLAIN, 18));
48     logOutButton.setBackground(new Color(220, 20, 60));
49     logOutButton.setForeground(Color.WHITE);
50     logOutButton.setFocusPainted(false);
51     logOutButton.setAlignmentX(Component.CENTER_ALIGNMENT);
52     logOutButton.setPreferredSize(new Dimension(200, 50));
53     logOutButton.addActionListener(e -> {
54         new LoginPage().setVisible(true);
55         this.dispose();
56     });
57
58     JButton backButton = new JButton("Back to Landing Page");
59     backButton.setFont(new Font("Arial", Font.PLAIN, 18));
60     backButton.setBackground(new Color(0, 128, 128));
61     backButton.setForeground(Color.WHITE);
62     backButton.setFocusPainted(false);
63     backButton.setAlignmentX(Component.CENTER_ALIGNMENT);
64     backButton.setPreferredSize(new Dimension(200, 50));
65     backButton.addActionListener(e -> {
66         new LandingPage(hostId).setVisible(true);
67         this.dispose();
68     });
69
70     // Initialize experiencePanel here
71     experiencePanel = new JPanel();
72     experiencePanel.setLayout(new BoxLayout(experiencePanel,
73         BoxLayout.Y_AXIS));
74     experiencePanel.setBackground(new Color(245, 245, 245));
75     experiencePanel.setBorder(BorderFactory.createTitledBorder(
76         "Your Hosted Experiences"));

```

```

70
71     panel.add(Box.createRigidArea(new Dimension(0, 20)));
72     panel.add(dashboardLabel);
73     panel.add(Box.createRigidArea(new Dimension(0, 20)));
74     panel.add(hostExperienceButton);
75     panel.add(Box.createRigidArea(new Dimension(0, 20)));
76     panel.add(backButton);
77     panel.add(Box.createRigidArea(new Dimension(0, 20)));
78     panel.add(logOutButton);
79     panel.add(Box.createRigidArea(new Dimension(0, 20)));
80     panel.add(experiencePanel); // Add experiencePanel to the
        main panel
81
82     add(panel);
83 }
84
85 private void fetchHostedExperiences() {
86     try (Connection conn = DatabaseConnection.getConnection())
87     {
88         String sql = "SELECT title, date_listed, price FROM
            Experiences WHERE host_id = ?";
89         PreparedStatement stmt = conn.prepareStatement(sql);
90         stmt.setInt(1, hostId);
91         ResultSet rs = stmt.executeQuery();
92
93         // Clear previous experiences
94         experiencePanel.removeAll();
95
96         while (rs.next()) {
97             String title = rs.getString("title");
98             String dateListed = rs.getString("date_listed");
99             double price = rs.getDouble("price");
100
101             JLabel experienceLabel = new JLabel("Experience: "
                + title + " | Listed on: " + dateListed + " |
                Price: $" + price);
102             experienceLabel.setFont(new Font("Arial", Font.
                PLAIN, 16));
103             experienceLabel.setForeground(new Color(25, 25,
                112));
104             experiencePanel.add(experienceLabel);
105         }
106
107         // Revalidate and repaint to update the UI
108         experiencePanel.revalidate();
109         experiencePanel.repaint();
110
111     } catch (Exception e) {
112         JOptionPane.showMessageDialog(this, "Error fetching
            hosted experiences: " + e.getMessage(), "Error",
            JOptionPane.ERROR_MESSAGE);
113     }
114 }
115
116 public static void main(String[] args) {
117     // Replace with actual host ID after login
    SwingUtilities.invokeLater(() -> {

```

```

118         int hostId = 1; // Example host ID for testing; replace
119             with actual retrieval from the database
120         new HostDashboard(hostId).setVisible(true);
121     });
122 }

```

UserDashboard

```

1 package UI;
2 import UI.DatabaseConnection;
3 import java.awt.*;
4 import java.sql.Connection;
5 import java.sql.PreparedStatement;
6 import java.sql.ResultSet;
7 import javax.swing.*;
8
9 public class UserDashboard extends JFrame {
10     private int userId;
11
12     public UserDashboard(int userId) {
13         this.userId = userId;
14         initComponents();
15         fetchUserBookings();
16     }
17
18     private void initComponents() {
19         setTitle("TRYIT - User Dashboard");
20         getContentPane().setBackground(new Color(255, 250, 240));
21
22         JLabel dashboardLabel = new JLabel("User Dashboard");
23         dashboardLabel.setFont(new Font("Arial", Font.BOLD, 30));
24         dashboardLabel.setForeground(new Color(0, 102, 204));
25
26         JButton logOutButton = new JButton("Logout");
27         logOutButton.setFont(new Font("Arial", Font.BOLD, 16));
28         logOutButton.setBackground(new Color(255, 51, 51));
29         logOutButton.setForeground(Color.WHITE);
30         logOutButton.setFocusPainted(false);
31         logOutButton.setPreferredSize(new Dimension(120, 40));
32         logOutButton.addActionListener(e -> {
33             new LoginPage().setVisible(true);
34             this.dispose();
35         });
36
37         JButton backButton = new JButton("Back to Landing Page");
38         backButton.setFont(new Font("Arial", Font.BOLD, 16));
39         backButton.setBackground(new Color(0, 128, 128));
40         backButton.setForeground(Color.WHITE);
41         backButton.setFocusPainted(false);
42         backButton.setPreferredSize(new Dimension(200, 40));
43         backButton.addActionListener(e -> new LandingPage(userId).
44             setVisible(true));
45
46         // Assuming this is in a button click event or similar
47         JButton viewTransactionsButton = new JButton("View Transaction
48             History");

```

```

47
48
49     viewTransactionsButton.setFont(new Font("Arial", Font.BOLD,
50         16));
51     viewTransactionsButton.setBackground(new Color(100, 149,
52         237));
53     viewTransactionsButton.setForeground(Color.WHITE);
54     viewTransactionsButton.setFocusPainted(false);
55     viewTransactionsButton.setPreferredSize(new Dimension(160,
56         40));
57     viewTransactionsButton.addActionListener(e -> new
58         TransactionHistoryPage(userId).setVisible(true));
59
60     JPanel bookingsPanel = new JPanel();
61     bookingsPanel.setBackground(new Color(255, 255, 204));
62     bookingsPanel.setLayout(new BoxLayout(bookingsPanel,
63         BoxLayout.Y_AXIS));
64     bookingsPanel.setBorder(BorderFactory.createTitledBorder("
65         Your Bookings"));
66
67     setLayout(new BorderLayout());
68     JPanel headerPanel = new JPanel();
69     headerPanel.setBackground(new Color(0, 153, 153));
70     headerPanel.setLayout(new FlowLayout(FlowLayout.CENTER, 20,
71         10));
72     headerPanel.add(dashboardLabel);
73     headerPanel.add(backButton);
74     headerPanel.add(viewTransactionsButton);
75     headerPanel.add(logOutButton);
76
77     JScrollPane scrollPane = new JScrollPane(bookingsPanel);
78     scrollPane.setPreferredSize(new Dimension(600, 300));
79     scrollPane.setBorder(BorderFactory.createLineBorder(Color.
80         GRAY, 2));
81
82     add(headerPanel, BorderLayout.NORTH);
83     add(scrollPane, BorderLayout.CENTER);
84     pack();
85     setLocationRelativeTo(null);
86     setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
87     setVisible(true);
88 }
89
90 // In the UserDashboard class, add:
91 private void fetchUserBookings() {
92     try (Connection conn = DatabaseConnection.getConnection()) {
93         String sql = "SELECT Experiences.title, Bookings.attendees,
94             Bookings.total_amount FROM Bookings " +
95             "JOIN Experiences ON Bookings.experience_id =
96             Experiences.id " +
97             "WHERE Bookings.user_id = ?";
98         PreparedStatement stmt = conn.prepareStatement(sql);
99         stmt.setInt(1, userId);
100         ResultSet rs = stmt.executeQuery();
101
102         while (rs.next()) {

```

```

95         String experienceTitle = rs.getString("title");
96         int attendees = rs.getInt("attendees");
97         double totalAmount = rs.getDouble("total_amount");
98
99         // Add booking details to the dashboard
100     }
101     } catch (Exception e) {
102         JOptionPane.showMessageDialog(this, "Error fetching
103         bookings: " + e.getMessage());
104     }
105 }
106
107 public static void main(String[] args) {
108     SwingUtilities.invokeLater(() -> {
109         try (Connection conn = DatabaseConnection.getConnection
110             ()) {
111             String sql = "SELECT id FROM Users WHERE username =
112                 ''"; // Replace 'your_username' with the actual
113                 username
114             PreparedStatement stmt = conn.prepareStatement(sql)
115                 ;
116             ResultSet rs = stmt.executeQuery();
117             if (rs.next()) {
118                 int userId = rs.getInt("id");
119                 new UserDashboard(userId).setVisible(true);
120             }
121             } catch (Exception e) {
122                 e.printStackTrace();
123             }
124         });
125     }}

```

ExperienceListingPage

```

1 package UI;
2 import UI.DatabaseConnection;
3 import java.sql.Connection;
4 import java.sql.PreparedStatement;
5 import java.sql.ResultSet;
6 import javax.swing.*;
7 import java.awt.*;
8
9 public class ExperienceListingPage extends JFrame {
10     private int userId;
11
12     public ExperienceListingPage(int userId) {
13         this.userId = userId;
14         initComponents();
15         fetchExperiences();
16     }
17
18     private void initComponents() {
19         setTitle("ExperienceXchange- Experience Listing");
20         setSize(500, 600);
21         setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);

```

```

22     setLocationRelativeTo(null);
23     setLayout(new BorderLayout());
24
25     // Main Content Panel
26     JPanel mainPanel = new JPanel();
27     mainPanel.setLayout(new BoxLayout(mainPanel, BoxLayout.
        Y_AXIS));
28     mainPanel.setBackground(Color.WHITE);
29     mainPanel.setBorder(BorderFactory.createEmptyBorder(20, 20,
        20, 20));
30
31     // Title Label
32     JLabel titleLabel = new JLabel("Available Experiences");
33     titleLabel.setFont(new Font("Arial", Font.BOLD, 24));
34     titleLabel.setForeground(new Color(102, 102, 102));
35     mainPanel.add(titleLabel);
36
37     add(mainPanel, BorderLayout.CENTER);
38     pack();
39 }
40
41 private void fetchExperiences() {
42     try (Connection conn = DatabaseConnection.getConnection()) {
43         String sql = "SELECT id, title, location, price FROM
            Experiences ORDER BY date_listed DESC";
44         PreparedStatement stmt = conn.prepareStatement(sql);
45         ResultSet rs = stmt.executeQuery();
46
47         // Panel to hold experience listings
48         JPanel experiencePanel = new JPanel();
49         experiencePanel.setLayout(new BoxLayout(experiencePanel,
            BoxLayout.Y_AXIS));
50         experiencePanel.setBackground(Color.WHITE);
51
52         while (rs.next()) {
53             int experienceId = rs.getInt("id");
54             String title = rs.getString("title");
55             String location = rs.getString("location");
56             double price = rs.getDouble("price");
57
58             // Individual experience panel
59             JPanel singleExperiencePanel = new JPanel();
60             singleExperiencePanel.setBackground(new Color(240, 240,
                240));
61             singleExperiencePanel.setLayout(new BoxLayout(
                singleExperiencePanel, BoxLayout.Y_AXIS));
62             singleExperiencePanel.setBorder(BorderFactory.
                createEmptyBorder(10, 10, 10, 10));
63
64             JLabel experienceTitle = new JLabel(title);
65             experienceTitle.setFont(new Font("Arial", Font.BOLD,
                18));
66             experienceTitle.setForeground(new Color(102, 102, 255))
                ;
67
68             JLabel experienceLocation = new JLabel("Location: " +
                location);

```

```

69         experienceLocation.setFont(new Font("Arial", Font.PLAIN
70             , 14));
71         JLabel experiencePrice = new JLabel("Price: $" + price)
72             ;
73         experiencePrice.setFont(new Font("Arial", Font.PLAIN,
74             14));
75         singleExperiencePanel.add(experienceTitle);
76         singleExperiencePanel.add(experienceLocation);
77         singleExperiencePanel.add(experiencePrice);
78         singleExperiencePanel.setAlignmentX(Component.
79             LEFT_ALIGNMENT);
80
81         singleExperiencePanel.addMouseListener(new java.awt.
82             event.MouseAdapter() {
83             public void mouseClicked(java.awt.event.MouseEvent
84                 evt) {
85                 new BookingPage(userId, experienceId, price).
86                     setVisible(true);
87             }
88         });
89
90         experiencePanel.add(singleExperiencePanel);
91         experiencePanel.add(Box.createVerticalStrut(10));
92     }
93
94     add(new JScrollPane(experiencePanel), BorderLayout.CENTER);
95     revalidate();
96 } catch (Exception e) {
97     JOptionPane.showMessageDialog(this, "Error fetching
98         experiences: " + e.getMessage(), "Error", JOptionPane.
99         ERROR_MESSAGE);
100 }
101
102 public static void main(String[] args) {
103     SwingUtilities.invokeLater(() -> {
104         new ExperienceListingPage(1).setVisible(true); //
105             Example usage with a dummy user ID
106     });
107 }

```

HostExperiencePage

```

1 package UI;
2
3 import java.awt.*;
4 import java.math.BigDecimal;
5 import java.sql.*;
6 import javax.swing.*;
7
8 public class HostExperiencePage extends JFrame {
9     private JTextField titleField, descriptionField, locationField,
10         priceField, durationField;

```

```

10     private int hostId;
11
12     public HostExperiencePage(int hostId) {
13         this.hostId = hostId;
14         initComponents();
15     }
16
17     private void initComponents() {
18         setTitle("Host Experience");
19
20         // Create the main panel
21         JPanel mainPanel = new JPanel(new GridBagLayout());
22         GridBagConstraints gbc = new GridBagConstraints();
23         gbc.insets = new Insets(10, 10, 10, 10);
24
25         // Title field
26         JLabel titleLabel = new JLabel("Experience Title:");
27         titleField = new JTextField(20);
28         gbc.gridx = 0;
29         gbc.gridy = 0;
30         mainPanel.add(titleLabel, gbc);
31         gbc.gridx = 1;
32         mainPanel.add(titleField, gbc);
33
34         // Description field
35         JLabel descriptionLabel = new JLabel("Description:");
36         descriptionField = new JTextField(20);
37         gbc.gridx = 0;
38         gbc.gridy = 1;
39         mainPanel.add(descriptionLabel, gbc);
40         gbc.gridx = 1;
41         mainPanel.add(descriptionField, gbc);
42
43         // Location field
44         JLabel locationLabel = new JLabel("Location:");
45         locationField = new JTextField(20);
46         gbc.gridx = 0;
47         gbc.gridy = 2;
48         mainPanel.add(locationLabel, gbc);
49         gbc.gridx = 1;
50         mainPanel.add(locationField, gbc);
51
52         // Price field
53         JLabel priceLabel = new JLabel("Price:");
54         priceField = new JTextField(10);
55         gbc.gridx = 0;
56         gbc.gridy = 3;
57         mainPanel.add(priceLabel, gbc);
58         gbc.gridx = 1;
59         mainPanel.add(priceField, gbc);
60
61         // Duration field
62         JLabel durationLabel = new JLabel("Duration (minutes):");
63         durationField = new JTextField(10);
64         gbc.gridx = 0;
65         gbc.gridy = 4;
66         mainPanel.add(durationLabel, gbc);
67         gbc.gridx = 1;

```



```

68         mainPanel.add(durationField, gbc);
69
70         // Submit button
71         JButton submitButton = new JButton("Host Experience");
72         submitButton.addActionListener(e -> createExperience());
73         gbc.gridx = 0;
74         gbc.gridy = 5;
75         gbc.gridwidth = 2;
76         mainPanel.add(submitButton, gbc);
77
78         add(mainPanel);
79         setSize(400, 400);
80         setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
81         setLocationRelativeTo(null);
82     }
83
84     private void createExperience() {
85         try {
86             String title = titleField.getText();
87             String description = descriptionField.getText();
88             String location = locationField.getText();
89             BigDecimal price = new BigDecimal(priceField.getText());
90             ;
91             int duration = Integer.parseInt(durationField.getText());
92
93             try (Connection conn = DatabaseConnection.getConnection()) {
94                 String sql = "INSERT INTO Experiences (title,
95                             description, location, host_id, price, duration)
96                             VALUES (?, ?, ?, ?, ?, ?)";
97                 PreparedStatement stmt = conn.prepareStatement(sql);
98                 ;
99                 stmt.setString(1, title);
100                stmt.setString(2, description);
101                stmt.setString(3, location);
102                stmt.setInt(4, hostId);
103                stmt.setBigDecimal(5, price);
104                stmt.setInt(6, duration);
105                stmt.executeUpdate();
106                JOptionPane.showMessageDialog(this, "Experience
107                hosted successfully!");
108                this.dispose();
109            }
110        } catch (Exception e) {
111            JOptionPane.showMessageDialog(this, "Error: " + e.
112                getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
113        }
114    }
115 }

```

TransactionConfirmationPage

```

1 package UI;
2
3 import javax.swing.*;

```

```

4 import java.awt.*;
5 import java.sql.Connection;
6 import java.sql.PreparedStatement;
7
8 public class TransactionConfirmationPage extends JFrame {
9
10     private int bookingId; // To link the transaction to the
        booking
11     private double totalAmount; // The total amount for the
        transaction
12
13     public TransactionConfirmationPage(int bookingId, double
        totalAmount) {
14         this.bookingId = bookingId;
15         this.totalAmount = totalAmount;
16         initComponents();
17     }
18
19     private void initComponents() {
20         setTitle("Complete Transaction");
21         setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
22         setSize(400, 300);
23         setLocationRelativeTo(null); // Center the frame
24
25         JPanel mainPanel = new JPanel();
26         mainPanel.setLayout(new GridBagLayout());
27         mainPanel.setBackground(new Color(240, 248, 255)); // Light
            Blue Background
28         GridBagConstraints gbc = new GridBagConstraints();
29         gbc.insets = new Insets(10, 10, 10, 10);
30         gbc.fill = GridBagConstraints.HORIZONTAL;
31
32         // Title
33         JLabel titleLabel = new JLabel("Complete Your Transaction",
            JLabel.CENTER);
34         titleLabel.setFont(new Font("Arial", Font.BOLD, 24));
35         mainPanel.add(titleLabel, gbc);
36
37         // Total Amount Label
38         JLabel amountLabel = new JLabel("Total Amount: $" + String.
            format("%.2f", totalAmount));
39         amountLabel.setFont(new Font("Arial", Font.PLAIN, 18));
40         gbc.gridy = 1;
41         mainPanel.add(amountLabel, gbc);
42
43         // Payment Status Label
44         JLabel paymentStatusLabel = new JLabel("Payment Status:");
45         paymentStatusLabel.setFont(new Font("Arial", Font.PLAIN,
            18));
46         gbc.gridy = 2;
47         mainPanel.add(paymentStatusLabel, gbc);
48
49         // Payment Status (Placeholder for now)
50         JTextField paymentStatusField = new JTextField("completed",
            15);
51         paymentStatusField.setFont(new Font("Arial", Font.PLAIN,
            18));
52         gbc.gridy = 3;

```

```

53         mainPanel.add(paymentStatusField, gbc);
54
55         // Confirm Button
56         JButton confirmButton = new JButton("Confirm Transaction");
57         confirmButton.setFont(new Font("Arial", Font.BOLD, 16));
58         confirmButton.addActionListener(e -> completeTransaction());
59
60         gbc.gridy = 4;
61         mainPanel.add(confirmButton, gbc);
62
63         add(mainPanel);
64         setVisible(true);
65     }
66
67     private void completeTransaction() {
68         try (Connection conn = DatabaseConnection.getConnection()) {
69             String sql = "INSERT INTO Transactions (booking_id,
70                 payment_status, amount) VALUES (?, 'completed', ?)";
71             PreparedStatement stmt = conn.prepareStatement(sql);
72             stmt.setInt(1, bookingId);
73             stmt.setDouble(2, totalAmount);
74             stmt.executeUpdate();
75
76             JOptionPane.showMessageDialog(this, "Transaction Completed
77                 Successfully!");
78             dispose(); // Close the transaction confirmation window
79         } catch (Exception e) {
80             JOptionPane.showMessageDialog(this, "Error completing
81                 transaction: " + e.getMessage(), "Error", JOptionPane.
82                 ERROR_MESSAGE);
83         }
84     }
85 }

```

TransactionHistoryPage

```

1 package UI;
2
3 import java.awt.*;
4 import java.sql.*;
5 import javax.swing.*;
6
7 public class TransactionHistoryPage extends JFrame {
8     private int userId; // Actual user ID
9
10    public TransactionHistoryPage(int userId) {
11        this.userId = userId; // Store the user ID
12        initComponents();
13        fetchTransactionHistory(); // Fetch transaction history for
14            the user
15    }
16
17    private void initComponents() {
18        setTitle("TRYIT - Transaction History");
19        setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
20    }
21 }

```

```

19     setSize(600, 500);
20     setLocationRelativeTo(null);
21
22     JPanel mainPanel = new JPanel(new BorderLayout());
23     JLabel historyLabel = new JLabel("Your Transaction History:");
24     historyLabel.setFont(new Font("Arial", Font.BOLD, 24));
25     historyLabel.setHorizontalAlignment(SwingConstants.CENTER);
26     historyLabel.setForeground(new Color(25, 25, 112));
27
28     mainPanel.add(historyLabel, BorderLayout.NORTH);
29
30     JPanel transactionPanel = new JPanel();
31     transactionPanel.setLayout(new BoxLayout(transactionPanel,
32         BoxLayout.Y_AXIS));
33     JScrollPane scrollPane = new JScrollPane(transactionPanel);
34     mainPanel.add(scrollPane, BorderLayout.CENTER);
35
36     add(mainPanel);
37
38     private void fetchTransactionHistory() {
39         try (Connection conn = DatabaseConnection.getConnection())
40         {
41             String sql = "SELECT Experiences.title, Transactions.
42                 transaction_date, Transactions.amount, Transactions.
43                 payment_status " +
44                 "FROM Transactions " +
45                 "JOIN Bookings ON Transactions.booking_id
46                 " +
47                 "JOIN Experiences ON Bookings.
48                 experience_id " +
49                 "WHERE Bookings.user_id = ?";
50             PreparedStatement stmt = conn.prepareStatement(sql);
51             stmt.setInt(1, userId); // Set the actual user ID
52             ResultSet rs = stmt.executeQuery();
53
54             JPanel transactionPanel = (JPanel) ((JScrollPane)
55                 getContentPane().getComponent(1)).getViewport().
56                 getView();
57             transactionPanel.removeAll(); // Clear previous entries
58
59             if (!rs.next()) {
60                 JLabel noTransactionsLabel = new JLabel("No
61                     transactions found.");
62                 noTransactionsLabel.setFont(new Font("Arial", Font.
63                     ITALIC, 16));
64                 noTransactionsLabel.setForeground(new Color(255, 0,
65                     0));
66                 transactionPanel.add(noTransactionsLabel);
67             } else {
68                 do {
69                     String title = rs.getString("title");
70                     String date = rs.getString("transaction_date");
71                     double amount = rs.getDouble("amount");
72                     String status = rs.getString("payment_status");

```

```

64         JPanel itemPanel = new JPanel(new BorderLayout
65             ());
66         itemPanel.setBorder(BorderFactory.
67             createMatteBorder(0, 0, 1, 0, Color.
68                 LIGHT_GRAY));
69         itemPanel.setBackground(new Color(245, 245,
70             245));
71
72         JLabel titleLabel = new JLabel("Experience: " +
73             title);
74         titleLabel.setFont(new Font("Arial", Font.BOLD,
75             16));
76         titleLabel.setForeground(new Color(0, 100, 0));
77
78         JLabel detailsLabel = new JLabel("Date: " +
79             date + " | Amount: $" + amount + " | Status:
80             " + status);
81         detailsLabel.setFont(new Font("Arial", Font.
82             PLAIN, 14));
83         detailsLabel.setForeground(new Color(105, 105,
84             105));
85
86         itemPanel.add(titleLabel, BorderLayout.NORTH);
87         itemPanel.add(detailsLabel, BorderLayout.CENTER
88             );
89         transactionPanel.add(itemPanel);
90     } while (rs.next());
91 }
92
93 transactionPanel.revalidate(); // Refresh the panel
94 transactionPanel.repaint(); // Update the UI
95 } catch (Exception e) {
96     JOptionPane.showMessageDialog(this, "Error fetching
97         transaction history: " + e.getMessage(), "Error",
98         JOptionPane.ERROR_MESSAGE);
99 }
100 }
101 }
102 }

```

BookingPage

```

1 package UI;
2
3 import UI.DatabaseConnection;
4 import java.awt.*;
5 import java.sql.Connection;
6 import java.sql.PreparedStatement;
7 import java.sql.ResultSet;
8 import javax.swing.*;
9
10 public class BookingPage extends JFrame {
11
12     private int userId;
13     private int experienceId;
14     private JTextField attendeesField;
15     private JLabel totalAmountLabel;

```

```

16     private double experiencePrice;
17
18     public BookingPage(int userId, int experienceId, double
19         experiencePrice) {
20         this.userId = userId;
21         this.experienceId = experienceId;
22         this.experiencePrice = experiencePrice;
23         initComponents();
24     }
25
26     private void initComponents() {
27         setTitle("ExperienceXchange - Book Experience");
28
29         // Set main panel layout and background color
30         JPanel mainPanel = new JPanel();
31         mainPanel.setBackground(new Color(240, 248, 255)); // Light
32             blue background
33         mainPanel.setLayout(new GridBagLayout());
34         GridBagConstraints gbc = new GridBagConstraints();
35         gbc.insets = new Insets(10, 10, 10, 10);
36         gbc.fill = GridBagConstraints.HORIZONTAL;
37
38         // Heading
39         JLabel headerLabel = new JLabel("Book Your Experience",
40             JLabel.CENTER);
41         headerLabel.setFont(new Font("Arial", Font.BOLD, 24));
42         headerLabel.setForeground(new Color(25, 25, 112)); //
43             Midnight blue color
44         gbc.gridx = 0;
45         gbc.gridy = 0;
46         gbc.gridwidth = 2;
47         mainPanel.add(headerLabel, gbc);
48
49         // Attendees label and field
50         JLabel attendeesLabel = new JLabel("Number of Attendees:");
51         attendeesLabel.setFont(new Font("Arial", Font.PLAIN, 18));
52         attendeesLabel.setForeground(Color.DARK_GRAY);
53         gbc.gridx = 1;
54         gbc.gridwidth = 1;
55         mainPanel.add(attendeesLabel, gbc);
56
57         attendeesField = new JTextField(5);
58         attendeesField.setFont(new Font("Arial", Font.PLAIN, 18));
59         gbc.gridx = 1;
60         mainPanel.add(attendeesField, gbc);
61
62         // Total amount label
63         JLabel amountLabel = new JLabel("Total Amount:");
64         amountLabel.setFont(new Font("Arial", Font.PLAIN, 18));
65         amountLabel.setForeground(Color.DARK_GRAY);
66         gbc.gridx = 0;
67         gbc.gridy = 2;
68         mainPanel.add(amountLabel, gbc);
69
70         totalAmountLabel = new JLabel("0.00");
71         totalAmountLabel.setFont(new Font("Arial", Font.PLAIN, 18));
72         ;
73         totalAmountLabel.setForeground(Color.BLUE);

```

```

69         gbc.gridx = 1;
70         mainPanel.add(totalAmountLabel, gbc);
71
72         // Calculate button
73         JButton calculateButton = new JButton("Calculate Total");
74         calculateButton.setFont(new Font("Arial", Font.BOLD, 16));
75         calculateButton.setBackground(new Color(50, 205, 50)); //
            Lime green background
76         calculateButton.setForeground(Color.WHITE);
77         calculateButton.addActionListener(e -> calculateTotal());
78         gbc.gridx = 0;
79         gbc.gridy = 3;
80         gbc.gridwidth = 2;
81         mainPanel.add(calculateButton, gbc);
82
83         // Book button
84         JButton bookButton = new JButton("Book Now");
85         bookButton.setFont(new Font("Arial", Font.BOLD, 16));
86         bookButton.setBackground(new Color(30, 144, 255)); //
            Dodger blue background
87         bookButton.setForeground(Color.WHITE);
88         bookButton.addActionListener(e -> bookExperience());
89         gbc.gridy = 4;
90         mainPanel.add(bookButton, gbc);
91
92         // Add padding around the content
93         mainPanel.setBorder(BorderFactory.createEmptyBorder(20, 20,
            20, 20));
94
95         // Set layout and window properties
96         add(mainPanel);
97         setSize(400, 400);
98         setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
99         setLocationRelativeTo(null); // Center on screen
100        setVisible(true);
101    }
102
103    private void calculateTotal() {
104        try {
105            int attendees = Integer.parseInt(attendeesField.getText
                ());
106            double totalAmount = attendees * experiencePrice;
107            totalAmountLabel.setText(String.format("%.2f",
                totalAmount));
108        } catch (NumberFormatException e) {
109            JOptionPane.showMessageDialog(this, "Please enter a
                valid number of attendees.", "Input Error",
                JOptionPane.ERROR_MESSAGE);
110        }
111    }
112
113    private void bookExperience() {
114        try {
115            int attendees = Integer.parseInt(attendeesField.getText());
116            double totalAmount = attendees * experiencePrice;
117
118            try (Connection conn = DatabaseConnection.getConnection())
            {

```

```

119         String sql = "INSERT INTO Bookings (experience_id,
120             user_id, attendees, total_amount) VALUES (?, ?, ?,
121             ?)";
122         PreparedStatement stmt = conn.prepareStatement(sql,
123             PreparedStatement.RETURN_GENERATED_KEYS);
124         stmt.setInt(1, experienceId);
125         stmt.setInt(2, userId);
126         stmt.setInt(3, attendees);
127         stmt.setDouble(4, totalAmount);
128         stmt.executeUpdate();
129
130         // Get the generated booking ID
131         ResultSet generatedKeys = stmt.getGeneratedKeys();
132         if (generatedKeys.next()) {
133             int bookingId = generatedKeys.getInt(1);
134             // Open transaction confirmation page
135             new TransactionConfirmationPage(bookingId,
136                 totalAmount).setVisible(true);
137             JOptionPane.showMessageDialog(this, "Booking
138                 Successful! Please confirm the transaction.");
139             this.dispose(); // Close the booking page
140         }
141     } catch (Exception e) {
142         JOptionPane.showMessageDialog(this, "Error booking
143             experience: " + e.getMessage(), "Error", JOptionPane.
144             ERROR_MESSAGE);
145     }
146 } catch (NumberFormatException e) {
147     JOptionPane.showMessageDialog(this, "Please enter a valid
148         number of attendees.", "Input Error", JOptionPane.
149         ERROR_MESSAGE);
150 }
151 }
152 }
153 }
154 }

```

DatabaseConnection

```

1 package UI;
2 import java.sql.Connection;
3 import java.sql.DriverManager;
4 import java.sql.SQLException;
5 /**
6  *
7  * @author sdivy
8  */
9 public class DatabaseConnection {
10     private static final String URL = "jdbc:mysql://localhost:3306/
11         test";
12     private static final String USERNAME = "root"; // Replace with
13         your MySQL username
14     private static final String PASSWORD = ""; // Replace with your
15         MySQL password
16
17     public static Connection getConnection() throws SQLException {
18         Connection connection = null;
19     }
20 }

```



```
16         try {
17             connection = DriverManager.getConnection(URL, USERNAME,
18                 PASSWORD);
19             System.out.println("Database connected!");
20         } catch (SQLException e) {
21             System.err.println("Database connection failed: " + e.
22                 getMessage());
23             throw e;
24         }
25     }
    return connection;
}
```

Conclusion

ExperienceXchange offers a comprehensive platform that connects users with unique experiences, both online and in-person. By focusing on an intuitive user interface, seamless booking and payment processes, and a robust database structure, the platform provides a complete solution for experience seekers and hosts alike.

Bibliography

- [1] Oracle, *Creating a GUI with Swing*, 2024. Accessed: 2024-10-19. Available: <https://docs.oracle.com/javase/tutorial/uiswing/>.
- [2] Oracle, *JDBC Basics*, 2024. Accessed: 2024-10-19. Available: <https://docs.oracle.com/javase/tutorial/jdbc/>.
- [3] Stripe, *Stripe Java API*, 2024. Accessed: 2024-10-19. Available: <https://stripe.com/docs/api/java>.
- [4] Spring, *Spring Boot Documentation*, 2024. Accessed: 2024-10-19. Available: <https://spring.io/projects/spring-boot>.
- [5] Apache, *Apache Tomcat Documentation*, 2024. Accessed: 2024-10-19. Available: <https://tomcat.apache.org/tomcat-9.0-doc/>.