SHIFA KHAN
21BBS0255

JAVA PROGRAMMING

LA - 8

Q1 *Design the workshop event table on MYSQL with necessary fields like ame_participant,mobile, college name, participant_id, schedule, time, paid_amount. Craete the table in MYSQL. Write a java program to perform the following actions.*

1) Insert around 10 records into it

2) Take participant_id and perform the update to a particular record

3) Display all the values in the table in a formatted way

CODE:

```
import java.sql.*;


public class JdbcProgram {
   // JDBC URL, username, and password
   static final String JDBC_URL = "jdbc:mysql://localhost:3306";
   static final String USERNAME = "root@localhost";
   static final String PASSWORD = "Shifa@3010";

   static final String DATABASE_NAME = "eventDatabase"; // Name of the database to create

   public static void main(String[] args) {
      try {
         Class.forName("com.mysql.cj.jdbc.Driver");
         Connection connection = DriverManager.getConnection(JDBC_URL, USERNAME, PASSWORD);
         System.out.println("Connected to the database");

         // Create database if not exists
         createDatabase(connection);

         // Use the created database
         connection.setCatalog(DATABASE_NAME);

         // Create table if not exists
         createTable(connection);

         // Clear previous records from table if table already exists
```

```
        clearTable(connection);

        // 1) Insert around 10 records
        insertRecords(connection);

        System.out.println("********Display Records Before Update**********");
        displayRecords(connection);

        // 2) Update a particular record by participant_id
        updateRecord(connection, 70, 150.00);

        // 3) Display all values in the table
        System.out.println("********Display Records After Update**********");
        displayRecords(connection);
    } catch (Exception e) {
        e.printStackTrace();
    }
}

// Method to create database
private static void createDatabase(Connection connection) throws SQLException {
    try (Statement statement = connection.createStatement()) {
        statement.executeUpdate("CREATE DATABASE IF NOT EXISTS " + DATABASE_NAME);
        System.out.println("Database '" + DATABASE_NAME + "' created successfully (if it didn't exist already)");
    }
}

// Method to create table
private static void createTable(Connection connection) throws SQLException {
    String createTableQuery = "CREATE TABLE IF NOT EXISTS workshop_event (" +
            "participant_id INT AUTO_INCREMENT PRIMARY KEY," +
            "name_participant VARCHAR(255)," +
            "mobile VARCHAR(15)," +
            "college_name VARCHAR(255)," +
            "schedule DATE," +
            "time TIME," +
            "paid_amount DECIMAL(10, 2)" +
            ")";
```

```
    try (Statement statement = connection.createStatement()) {
        statement.executeUpdate(createTableQuery);
        System.out.println("Table created successfully (if it didn't exist already)");
    }
}

private static void clearTable(Connection connection) throws SQLException {
    String clearQuery = "DELETE FROM workshop_event";
    try (Statement statement = connection.createStatement()) {
        statement.executeUpdate(clearQuery);
        System.out.println("Cleared existing records from the table");
    }
}

// Method to insert records
private static void insertRecords(Connection connection) throws SQLException {
    String insertQuery = "INSERT INTO workshop_event (name_participant, mobile, college_name, schedule, time, paid_amount)
VALUES (?, ?, ?, ?, ?, ?)";
    try (PreparedStatement preparedStatement = connection.prepareStatement(insertQuery)) {
        String[][] data = {
                {"John Smith", "+447700123456", "University of Cambridge", "2024-05-01", "11:30:00", "1500.00"},
                {"Emma Johnson", "+447700987654", "University of College London", "2024-05-02", "10:45:00", "1800.00"},
                {"james Brown", "+447712345678", "Imperial College of London", "2024-05-03", "09:15:00", "1350.00"},
                {"Sophie Taylor", "+447712345678", "University of Oxford", "2024-05-04", "13:00:00", "1100.00"},
                {"david Wilson", "+447712345678", "London School of Economics", "2024-05-05", "14:30:00", "2000.00"},
                {"Charlotte Evans", "+447700111222", "King's College London", "2024-05-06", "12:00:00", "1750.00"},
                {"Michael Clark", "+447700222333", "University of Manchester", "2024-05-07", "11:00:00", "1600.00"},
                {"Lucy Wright", "+447700333444", "University of Edinburgh", "2024-05-08", "15:45:00", "1900.00"},
                {"Daniel Hughes", "+447700444555", "University of Brsitol ", "2024-05-09", "09:30:00", "1700.00"},
                {"jessica Lee", "+447700555666", "University of Glasgow", "2024-05-10", "16:15:00", "1450.00"}
        };

        for (String[] record : data) {
            for (int i = 0; i < record.length; i++) {
                preparedStatement.setString(i + 1, record[i]);
            }
            preparedStatement.executeUpdate();
        }
```

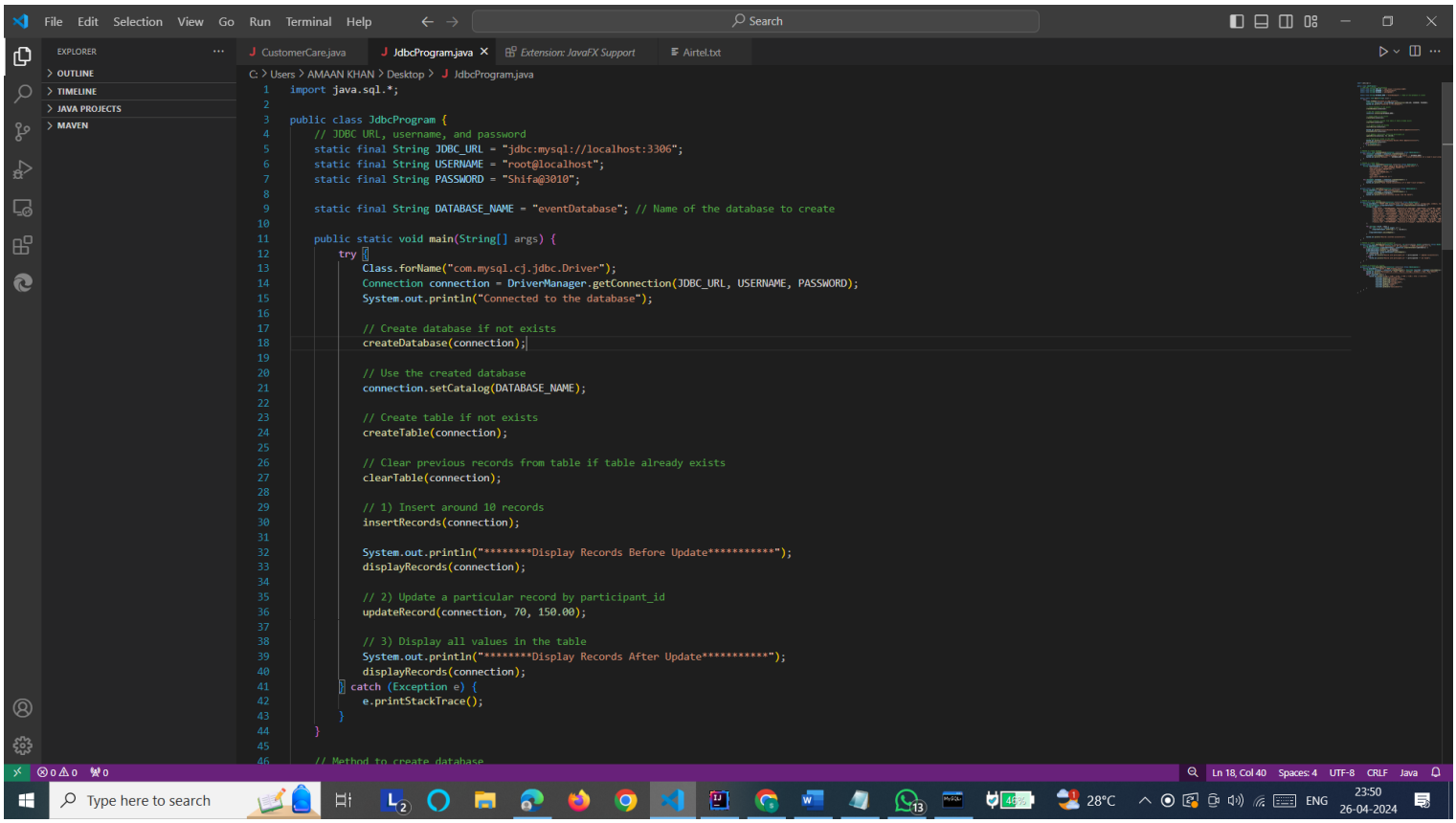
```
            System.out.println("Records inserted successfully");
        }
    }

    // Method to update a record by participant_id
    private static void updateRecord(Connection connection, int participantId, double paidAmount) throws SQLException {
        String updateQuery = "UPDATE workshop_event SET paid_amount = ? WHERE participant_id = ?";
        try (PreparedStatement preparedStatement = connection.prepareStatement(updateQuery)) {
            preparedStatement.setDouble(1, paidAmount);
            preparedStatement.setInt(2, participantId);
            int rowsAffected = preparedStatement.executeUpdate();
            if (rowsAffected > 0) {
                System.out.println("Record with participant_id " + participantId + " updated successfully");
            } else {
                System.out.println("Record with participant_id " + participantId + " not found");
            }
        }
    }

    // Method to display all records
    private static void displayRecords(Connection connection) throws SQLException {
        String selectQuery = "SELECT * FROM workshop_event";
        try (Statement statement = connection.createStatement(); ResultSet resultSet = statement.executeQuery(selectQuery)) {
            System.out.println("Participant ID | Name | Mobile | College | Schedule | Time | Paid Amount");
            System.out.println("-----------------------------------------------------------------");
            while (resultSet.next()) {
                System.out.printf("%-14d | %-20s | %-15s | %-30s | %-10s | %-8s | %-10.2f%n",
                        resultSet.getInt("participant_id"),
                        resultSet.getString("name_participant"),
                        resultSet.getString("mobile"),
                        resultSet.getString("college_name"),
                        resultSet.getDate("schedule"),
                        resultSet.getTime("time"),
                        resultSet.getDouble("paid_amount"));
            }
        }
    }
```

}

OUTPUT:

```
Table created successfully (if it didn't exist already)
Cleared existing records from the table
Records inserted successfully
********Display Records Before Update***********
Participant ID | Name | Mobile | College | Schedule | Time | Paid Amount
---------------------------------------------------------------
77             | John Smith           | +447700123456   | University of Cambridge        | 2024-05-01 | 11:30:00 | 1500.00
78             | Emma Johnson         | +447700987654   | University College London      | 2024-05-02 | 10:45:00 | 1800.00
79             | James Brown          | +447712345678   | Imperial College London        | 2024-05-03 | 09:15:00 | 1350.00
80             | Sophie Taylor        | +447712345678   | University of Oxford           | 2024-05-04 | 13:00:00 | 1100.00
81             | David Wilson         | +447712345678   | London School of Economics     | 2024-05-05 | 14:30:00 | 2000.00
82             | Charlotte Evans      | +447700111222   | King's College London          | 2024-05-06 | 12:00:00 | 1750.00
83             | Michael Clark        | +447700222333   | University of Manchester       | 2024-05-07 | 11:00:00 | 1600.00
84             | Lucy Wright          | +447700333444   | University of Edinburgh        | 2024-05-08 | 15:45:00 | 1900.00
85             | Daniel Hughes        | +447700444555   | University of Bristol          | 2024-05-09 | 09:30:00 | 1700.00
86             | Jessica Lee          | +447700555666   | University of Glasgow          | 2024-05-10 | 16:15:00 | 1450.00
Record with participant_id 70 not found
********Display Records After Update***********
Participant ID | Name | Mobile | College | Schedule | Time | Paid Amount
---------------------------------------------------------------
77             | John Smith           | +447700123456   | University of Cambridge        | 2024-05-01 | 11:30:00 | 1500.00
78             | Emma Johnson         | +447700987654   | University College London      | 2024-05-02 | 10:45:00 | 1800.00
79             | James Brown          | +447712345678   | Imperial College London        | 2024-05-03 | 09:15:00 | 1350.00
80             | Sophie Taylor        | +447712345678   | University of Oxford           | 2024-05-04 | 13:00:00 | 1100.00
81             | David Wilson         | +447712345678   | London School of Economics     | 2024-05-05 | 14:30:00 | 2000.00
82             | Charlotte Evans      | +447700111222   | King's College London          | 2024-05-06 | 12:00:00 | 1750.00
83             | Michael Clark        | +447700222333   | University of Manchester       | 2024-05-07 | 11:00:00 | 1600.00
```

SCREENSHOT:



Q9 *Design a JFX window to support the conference registration process. A conference attendee may have to enter, their name, age, employee ID, organization name, address, mobile number, research area, the title of his paper, and abstract.*

CODE:

```
package org.example.conferenceregistrationformm;


import javafx.application.Application;
import javafx.scene.Scene;
import javafx.stage.Stage;
import javafx.geometry.Insets;
import javafx.scene.control.*;
import javafx.scene.layout.GridPane;



public class ConferenceRegistrationFormm extends Application {

    @Override
    public void start(Stage primaryStage) {
        primaryStage.setTitle("Conference Registration");

        // Creating a GridPane layout
        GridPane grid = new GridPane();
        grid.setPadding(new Insets(20, 20, 20, 20));
        grid.setVgap(10);
        grid.setHgap(10);

        // Labels
        Label nameLabel = new Label("Name:");
        GridPane.setConstraints(nameLabel, 0, 0);

        Label ageLabel = new Label("Age:");
        GridPane.setConstraints(ageLabel, 0, 1);
```

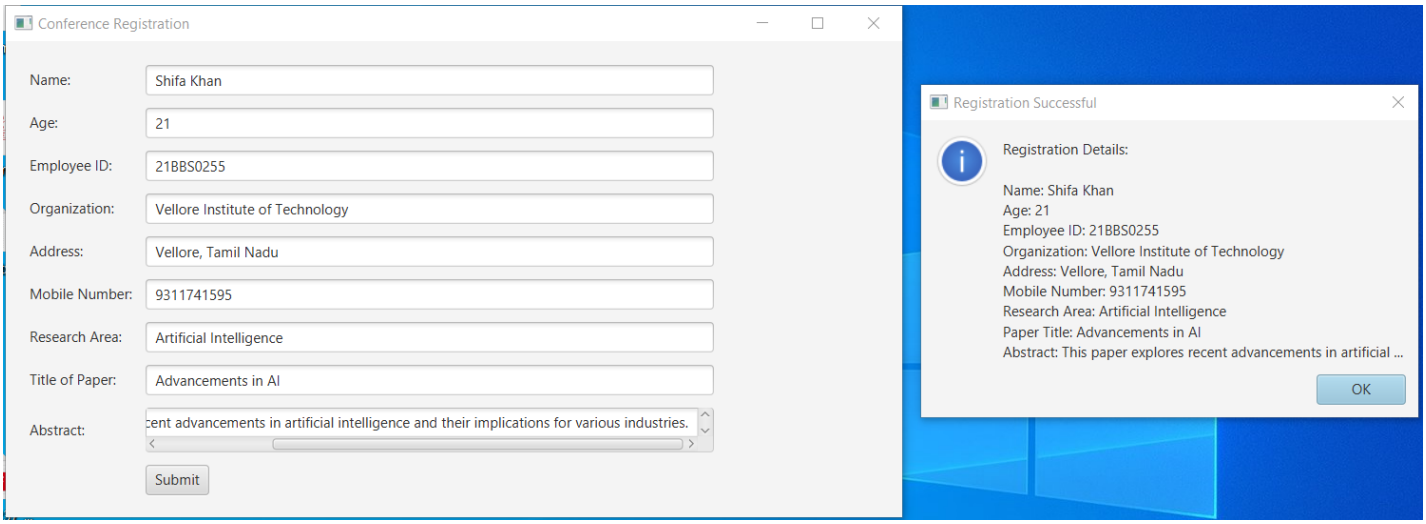
```
Label employeeIdLabel = new Label("Employee ID:");
GridPane.setConstraints(employeeIdLabel, 0, 2);

Label organizationLabel = new Label("Organization:");
GridPane.setConstraints(organizationLabel, 0, 3);

Label addressLabel = new Label("Address:");
GridPane.setConstraints(addressLabel, 0, 4);

Label mobileLabel = new Label("Mobile Number:");
GridPane.setConstraints(mobileLabel, 0, 5);

Label researchAreaLabel = new Label("Research Area:");
GridPane.setConstraints(researchAreaLabel, 0, 6);

Label paperTitleLabel = new Label("Title of Paper:");
GridPane.setConstraints(paperTitleLabel, 0, 7);

Label abstractLabel = new Label("Abstract:");
GridPane.setConstraints(abstractLabel, 0, 8);

// TextFields
TextField nameField = new TextField();
GridPane.setConstraints(nameField, 1, 0);

TextField ageField = new TextField();
GridPane.setConstraints(ageField, 1, 1);

TextField employeeIdField = new TextField();
GridPane.setConstraints(employeeIdField, 1, 2);

TextField organizationField = new TextField();
GridPane.setConstraints(organizationField, 1, 3);
```

```
TextField addressField = new TextField();
GridPane.setConstraints(addressField, 1, 4);

TextField mobileField = new TextField();
GridPane.setConstraints(mobileField, 1, 5);

TextField researchAreaField = new TextField();
GridPane.setConstraints(researchAreaField, 1, 6);

TextField paperTitleField = new TextField();
GridPane.setConstraints(paperTitleField, 1, 7);

TextArea abstractArea = new TextArea();
abstractArea.setPrefRowCount(5);
GridPane.setConstraints(abstractArea, 1, 8);

// Submit Button
Button submitButton = new Button("Submit");
GridPane.setConstraints(submitButton, 1, 9);

// Adding event handler to the Submit button
submitButton.setOnAction(e -> {
    // Retrieve data from text fields
    String name = nameField.getText();
    String age = ageField.getText();
    String employeeId = employeeIdField.getText();
    String organization = organizationField.getText();
    String address = addressField.getText();
    String mobileNumber = mobileField.getText();
    String researchArea = researchAreaField.getText();
    String paperTitle = paperTitleField.getText();
    String abstractText = abstractArea.getText();

    // Construct registration details message
```

```
    String registrationMessage = "Registration Details:\n\n" +
            "Name: " + name + "\n" +
            "Age: " + age + "\n" +
            "Employee ID: " + employeeId + "\n" +
            "Organization: " + organization + "\n" +
            "Address: " + address + "\n" +
            "Mobile Number: " + mobileNumber + "\n" +
            "Research Area: " + researchArea + "\n" +
            "Paper Title: " + paperTitle + "\n" +
            "Abstract: " + abstractText;

    // Show dialog box with registration details
    Alert alert = new Alert(Alert.AlertType.INFORMATION);
    alert.setTitle("Registration Successful");
    alert.setHeaderText(null);
    alert.setContentText(registrationMessage);
    alert.showAndWait();
});

// Adding all elements to the grid
grid.getChildren().addAll(
        nameLabel, nameField,
        ageLabel, ageField,
        employeeIdLabel, employeeIdField,
        organizationLabel, organizationField,
        addressLabel, addressField,
        mobileLabel, mobileField,
        researchAreaLabel, researchAreaField,
        paperTitleLabel, paperTitleField,
        abstractLabel, abstractArea,
        submitButton
);

Scene scene = new Scene(grid, 400, 400);
```

```
        primaryStage.setScene(scene);
        primaryStage.show();
    }

    public static void main(String[] args) {
        launch(args);
    }
}
```

OUTPUT:



SCREENSHOT: