# Java Project On Restaurant Ordering System Using Java As Front End And Mysql As Backend

**Name:-** Kovuru mahesh naidu

**Regno:-** 192224085

**Java code :-**

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.PreparedStatement;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.Scanner;

import java.time.LocalDate;

import java.time.format.DateTimeFormatter;

public class RSSO {

private static final String DB\_URL = "jdbc:mysql://localhost:3306/restaurant";

private static final String DB\_USER = "root";

private static final String DB\_PASSWORD = "mahi";

public static void main(String[] args) {

try {

Connection connection = DriverManager.getConnection(DB\_URL, DB\_USER, DB\_PASSWORD);

Scanner scanner = new Scanner(System.in);

int choice;

do {

System.out.println("\n1. Insert Order");

System.out.println("2. Retrieve Orders");

System.out.println("3. Update Order");

System.out.println("4. Delete Order");

System.out.println("5. Exit");

System.out.print("Enter your choice: ");

choice = scanner.nextInt();

scanner.nextLine(); // Consume newline character

switch (choice) {

case 1:

insertOrder(connection, scanner);

break;

case 2:

retrieveOrders(connection);

break;

case 3:

updateOrder(connection, scanner);

break;

case 4:

deleteOrder(connection, scanner);

break;

case 5:

System.out.println("Exiting...");

break;

default:

System.out.println("Invalid choice. Please try again.");

}

} while (choice != 5);

scanner.close();

connection.close();

} catch (SQLException e) {

System.out.println("Error: " + e.getMessage());

} catch (Exception e) {

System.out.println("Error: " + e.getMessage());

}

}

private static void insertOrder(Connection connection, Scanner scanner) {

try {

System.out.print("Enter customer name: ");

String customerName = scanner.nextLine();

System.out.print("Enter order date (dd/MM/yyyy) or enter 'today' for today's date: ");

String orderDateStr = scanner.nextLine();

// If order date is blank or 'today', use current date

if (orderDateStr.isEmpty() || orderDateStr.equalsIgnoreCase("today")) {

LocalDate currentDate = LocalDate.now();

DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd/MM/yyyy");

orderDateStr = currentDate.format(formatter);

}

// Convert orderDateStr to java.sql.Date

DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd/MM/yyyy");

LocalDate parsedDate = LocalDate.parse(orderDateStr, formatter);

java.sql.Date orderDate = java.sql.Date.valueOf(parsedDate);

System.out.print("Enter total amount: ");

double totalAmount = scanner.nextDouble();

String insertQuery = "INSERT INTO restaurant\_order (customer\_name, order\_date, total\_amount) VALUES (?, ?, ?)";

PreparedStatement insertStatement = connection.prepareStatement(insertQuery);

insertStatement.setString(1, customerName);

insertStatement.setDate(2, orderDate);

insertStatement.setDouble(3, totalAmount);

int rowsInserted = insertStatement.executeUpdate();

if (rowsInserted > 0) {

System.out.println("New order inserted.");

} else {

System.out.println("No new order inserted.");

}

} catch (Exception e) {

System.out.println("Error: " + e.getMessage());

}

}

private static void retrieveOrders(Connection connection) {

try {

String selectQuery = "SELECT \* FROM restaurant\_order";

Statement selectStatement = connection.createStatement();

ResultSet resultSet = selectStatement.executeQuery(selectQuery);

System.out.println("\nOrder ID\tCustomer Name\tOrder Date\tTotal Amount");

while (resultSet.next()) {

int id = resultSet.getInt("id");

String customerName = resultSet.getString("customer\_name");

java.sql.Date orderDate = resultSet.getDate("order\_date");

double totalAmount = resultSet.getDouble("total\_amount");

System.out.println(id + "\t\t" + customerName + "\t\t" + orderDate + "\t\t" + totalAmount);

}

} catch (Exception e) {

System.out.println("Error: " + e.getMessage());

}

}

private static void updateOrder(Connection connection, Scanner scanner) {

try {

System.out.print("Enter order ID to update: ");

int orderId = scanner.nextInt();

System.out.print("Enter new total amount: ");

double newTotalAmount = scanner.nextDouble();

String updateQuery = "UPDATE restaurant\_order SET total\_amount = ? WHERE id = ?";

PreparedStatement updateStatement = connection.prepareStatement(updateQuery);

updateStatement.setDouble(1, newTotalAmount);

updateStatement.setInt(2, orderId);

int rowsUpdated = updateStatement.executeUpdate();

if (rowsUpdated > 0) {

System.out.println("Order updated.");

} else {

System.out.println("No order updated.");

}

} catch (Exception e) {

System.out.println("Error: " + e.getMessage());

}

}

private static void deleteOrder(Connection connection, Scanner scanner) {

try {

System.out.print("Enter order ID to delete: ");

int orderId = scanner.nextInt();

String deleteQuery = "DELETE FROM restaurant\_order WHERE id = ?";

PreparedStatement deleteStatement = connection.prepareStatement(deleteQuery);

deleteStatement.setInt(1, orderId);

int rowsDeleted = deleteStatement.executeUpdate();

if (rowsDeleted > 0) {

System.out.println("Order deleted.");

} else {

System.out.println("No order deleted.");

}

} catch (Exception e) {

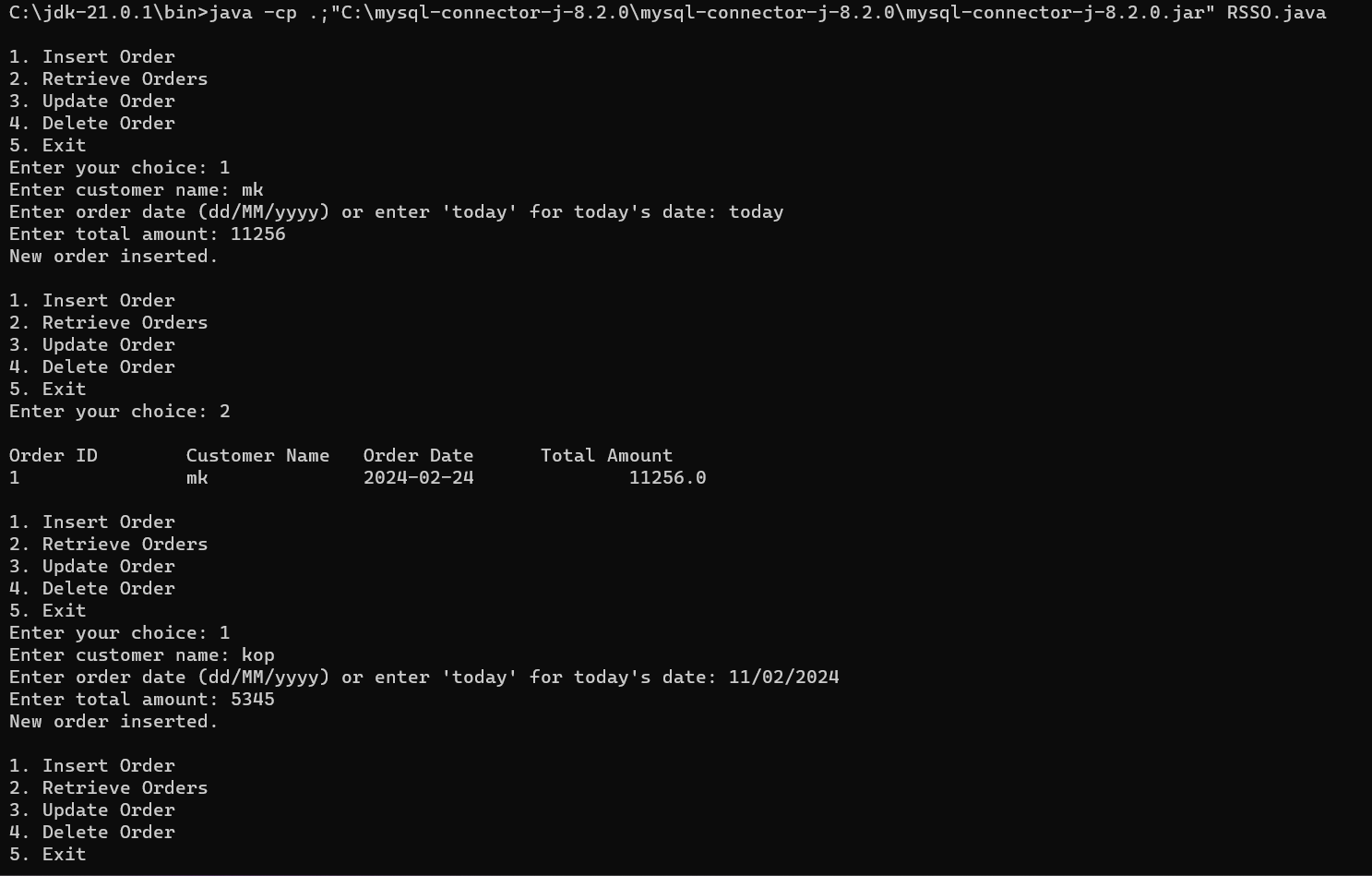
System.out.println("Error: " + e.getMessage());

}

}

}

**Output:-**



A screenshot of a computer

Description automatically generated

