

Relational Databases with MySQL Week 10 Assignment

Points possible: 70

Category	Criteria	% of Grade
Functionality	Does the code work?	25
Organization	Is the code clean and organized? Proper use of white space, syntax, and consistency are utilized. Names and comments are concise and clear.	25
Creativity	Student solved the problems presented in the assignment using creativity and out of the box thinking.	25
Completeness	All requirements of the assignment are complete.	25

Instructions: In Eclipse, or an IDE of your choice, write the code that accomplishes the objectives listed below. Ensure that the code compiles and runs as directed. Take screenshots of the code and of the running program (make sure to get screenshots of all required functionality) and paste them in this document where instructed below. Create a new repository on GitHub for this week's assignments and push this document to the repository. Additionally, push an .sql file with all your queries and your Java project code to the same repository. Add the URL for this week's repository to this document where instructed and submit this document to your instructor when complete.

Coding Steps:

In this week's coding activity, you will create a menu driven application backed by a MySQL database.

To start, choose one item that you like. It could be vehicles, sports, foods, etc....

Create a new Java project in Eclipse.

Create a SQL script in the project to create a database with one table. The table should be the item you picked.

Write a Java menu driven application that allows you to perform all four CRUD operations on your table.

Tips:

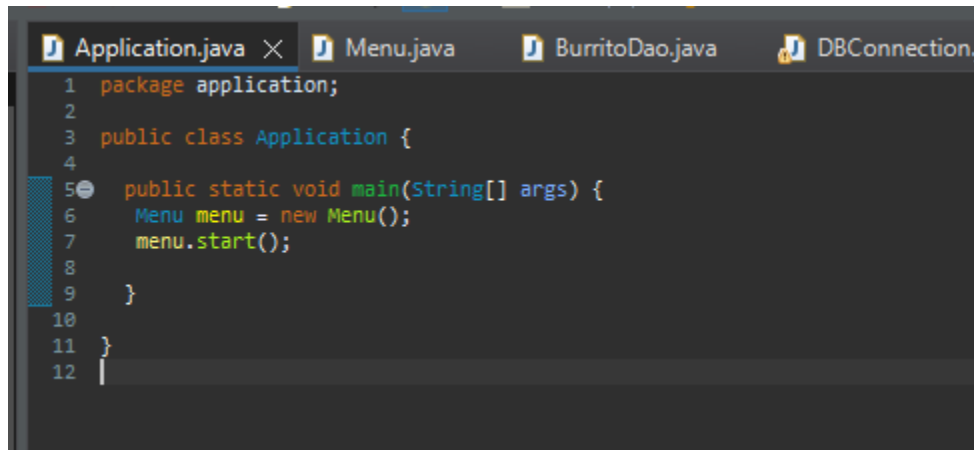
The application does not need to be as complex as the example in the video curriculum.

You need an option for each of the CRUD operations (Create, Read, Update, and Delete).

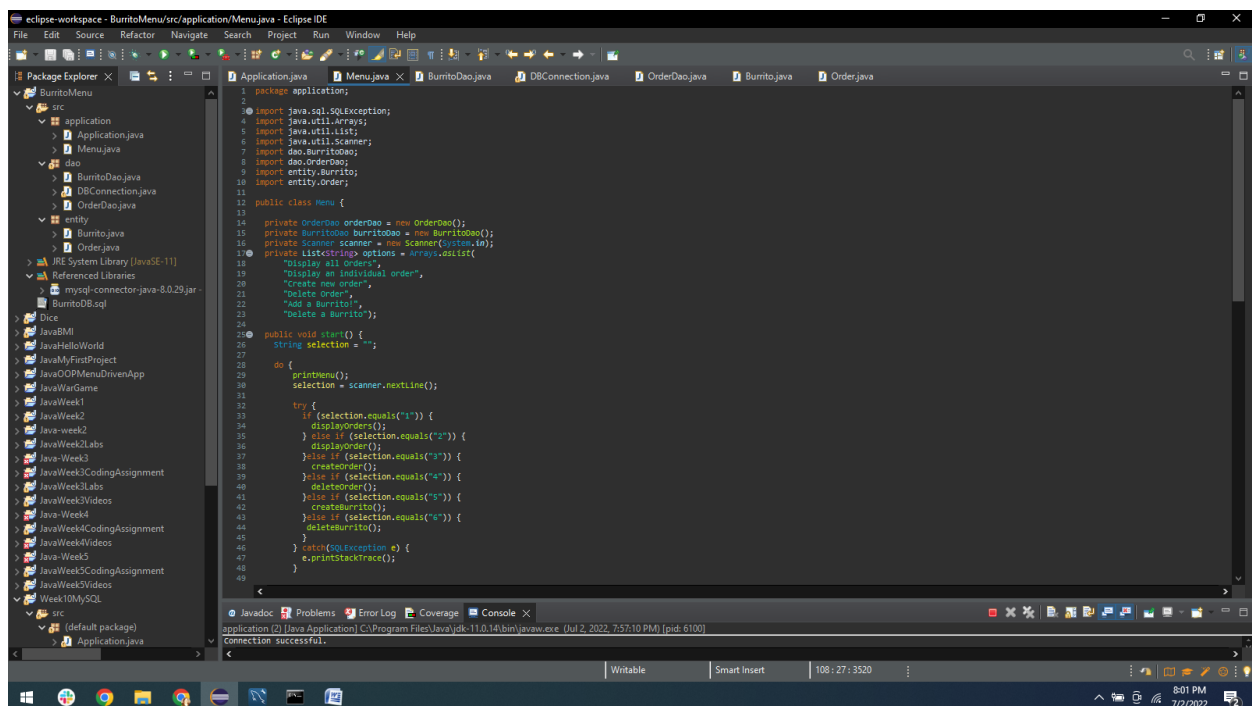
Remember that `PreparedStatement.executeQuery()` is only for Reading data and `.executeUpdate()` is used for Creating, Updating, and Deleting data.

Remember that both parameters on `PreparedStatements` and the `ResultSet` columns are based on indexes that start with 1, not 0.

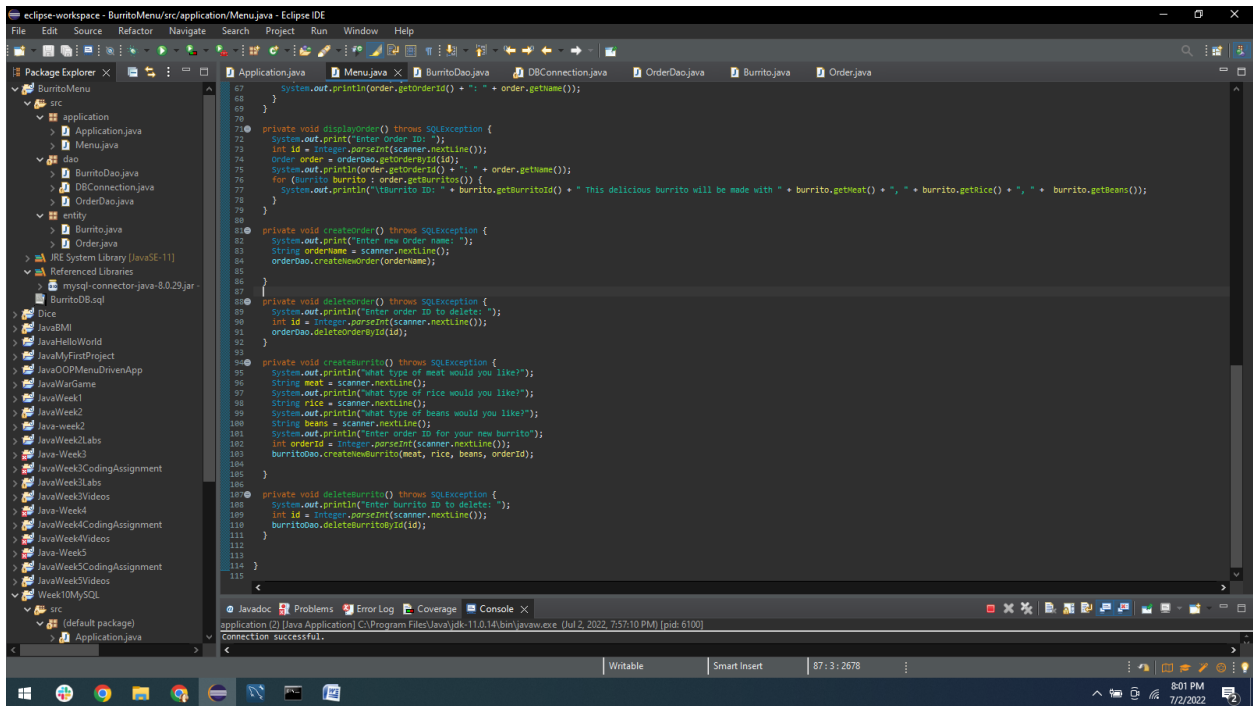
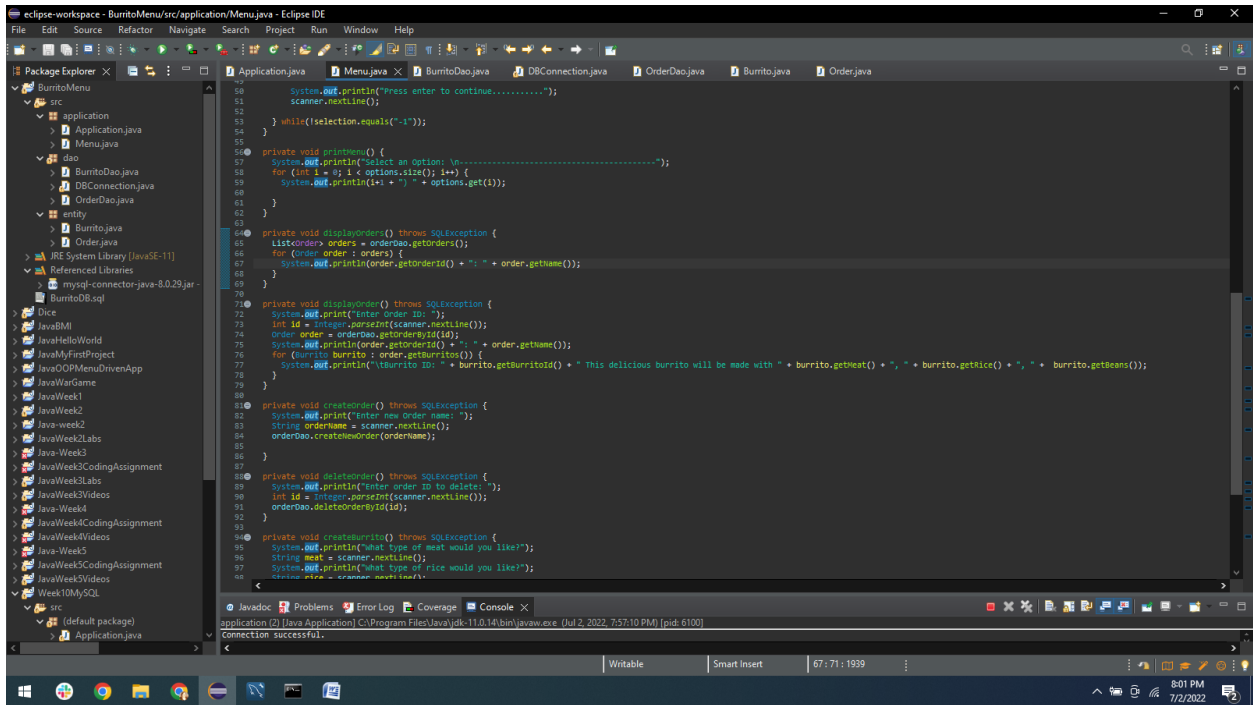
Screenshots of Code:

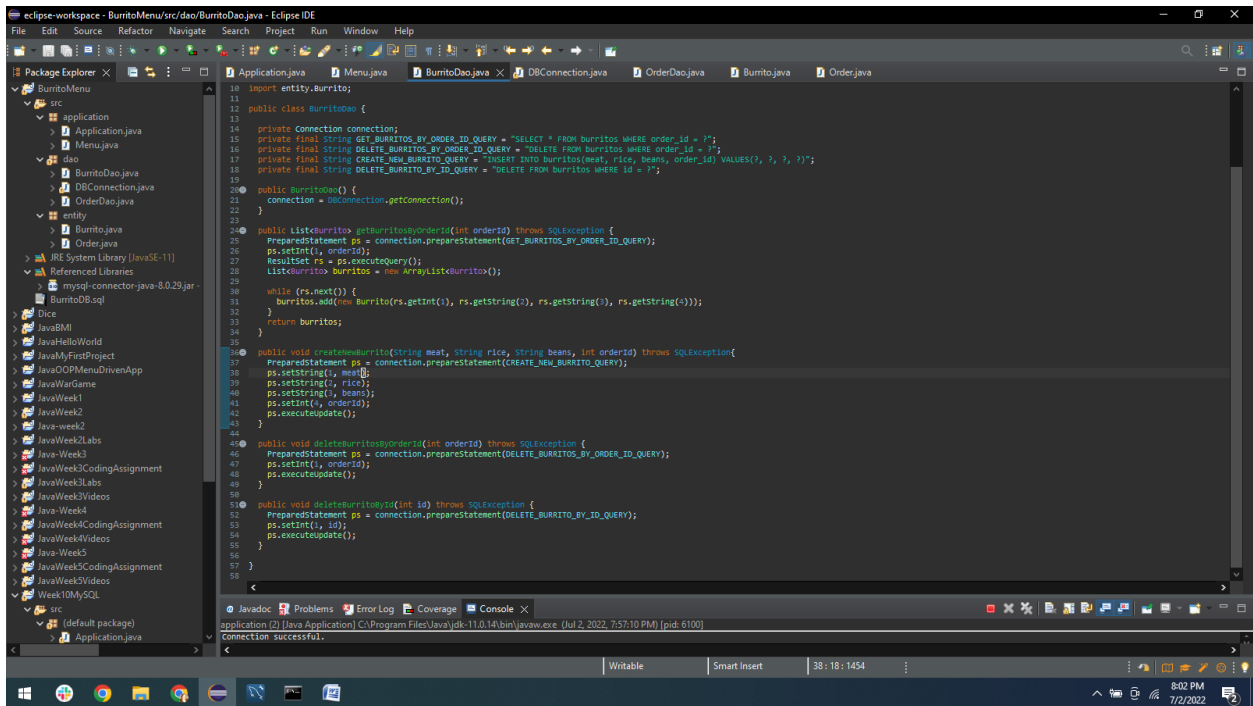
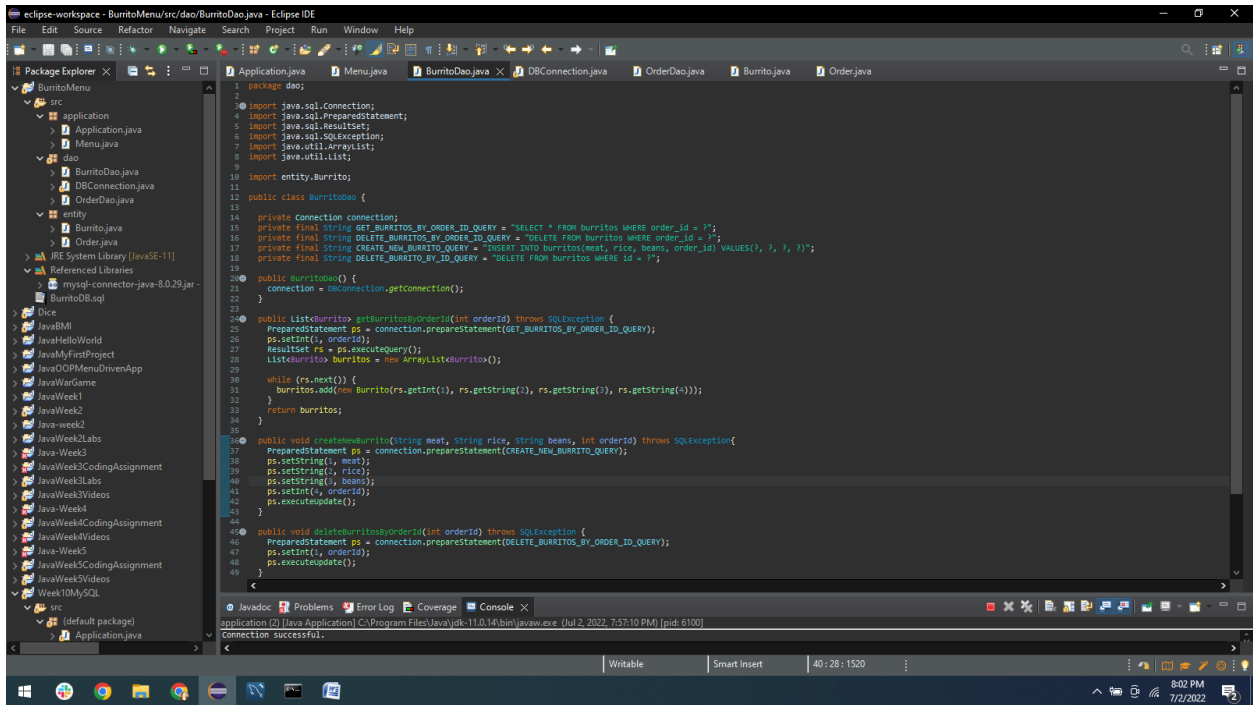


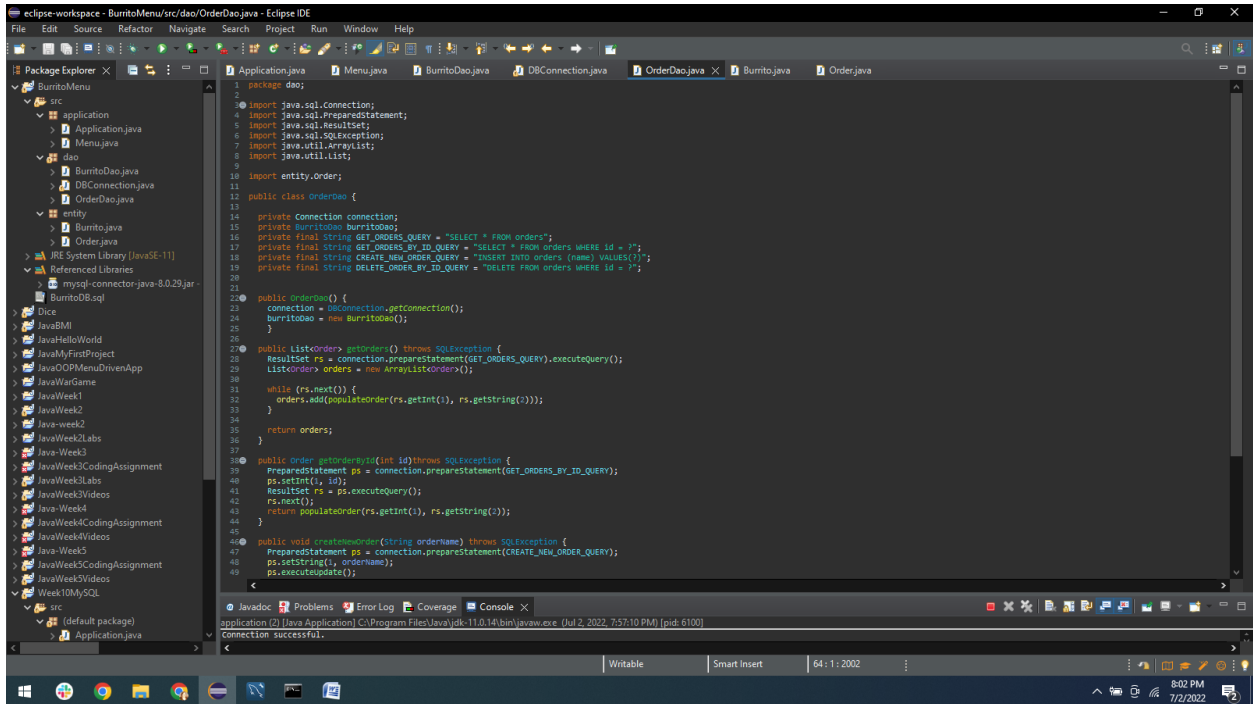
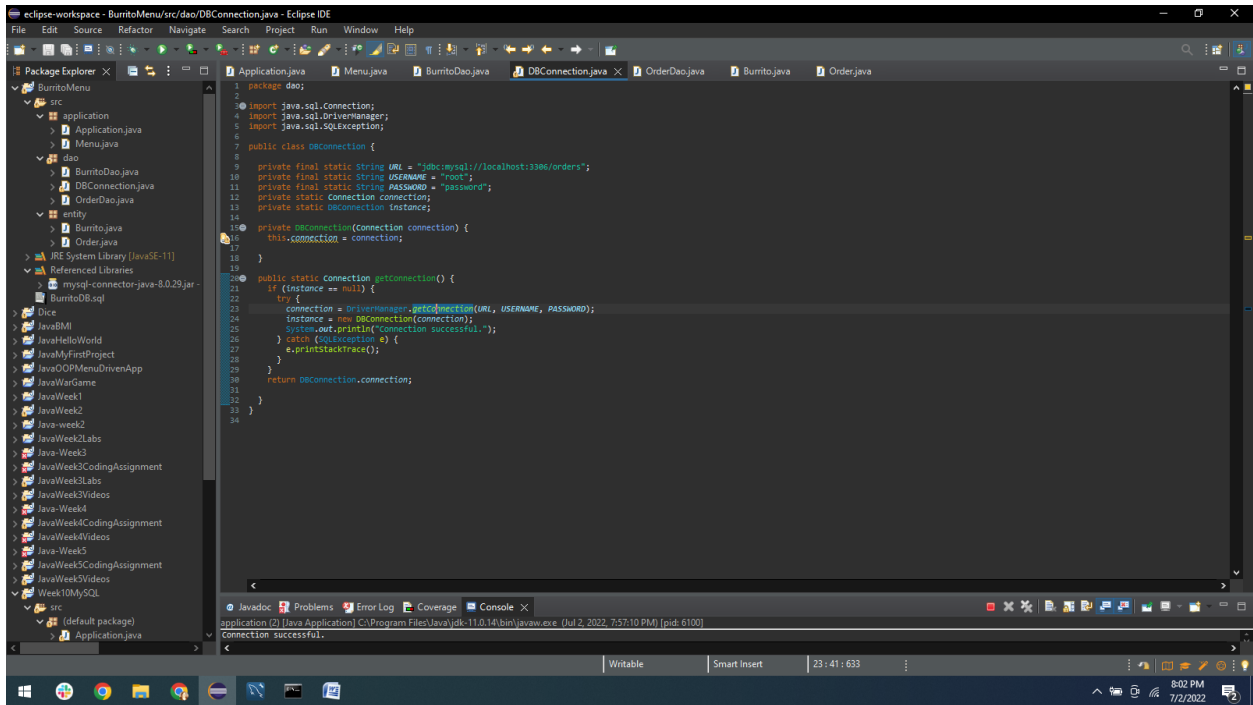
```
1 package application;
2
3 public class Application {
4
5     public static void main(String[] args) {
6         Menu menu = new Menu();
7         menu.start();
8     }
9 }
10
11 }
12 |
```

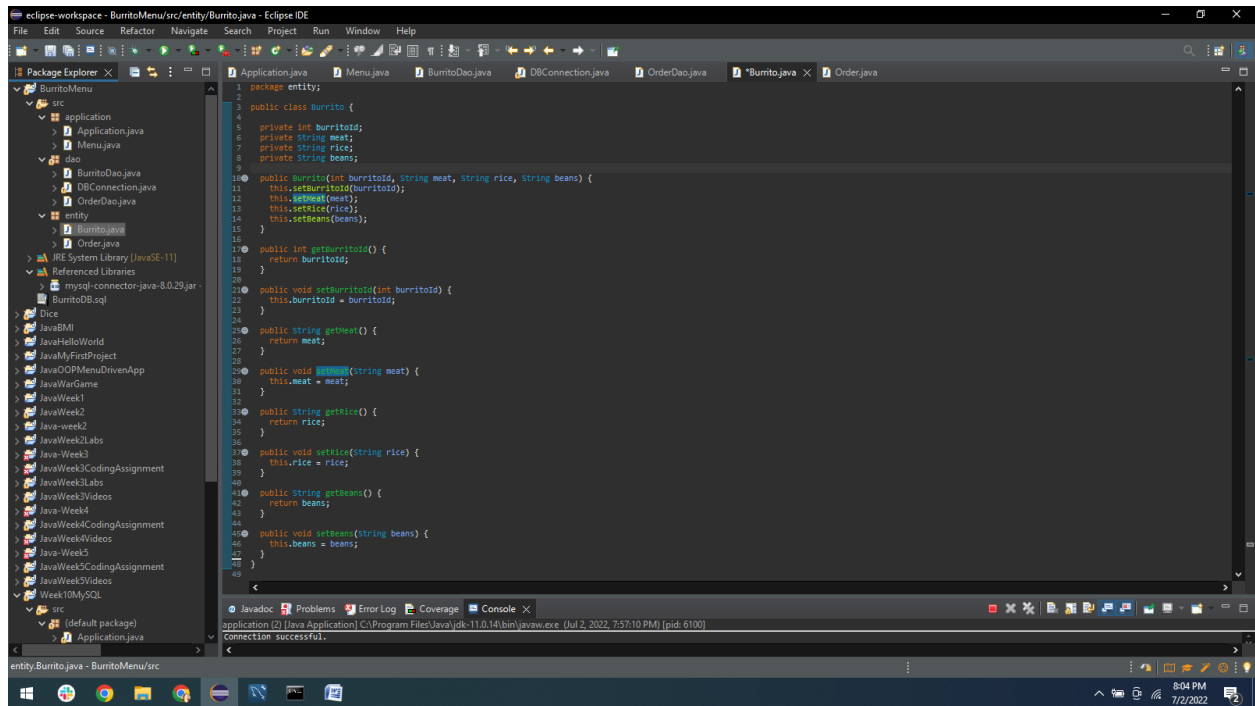
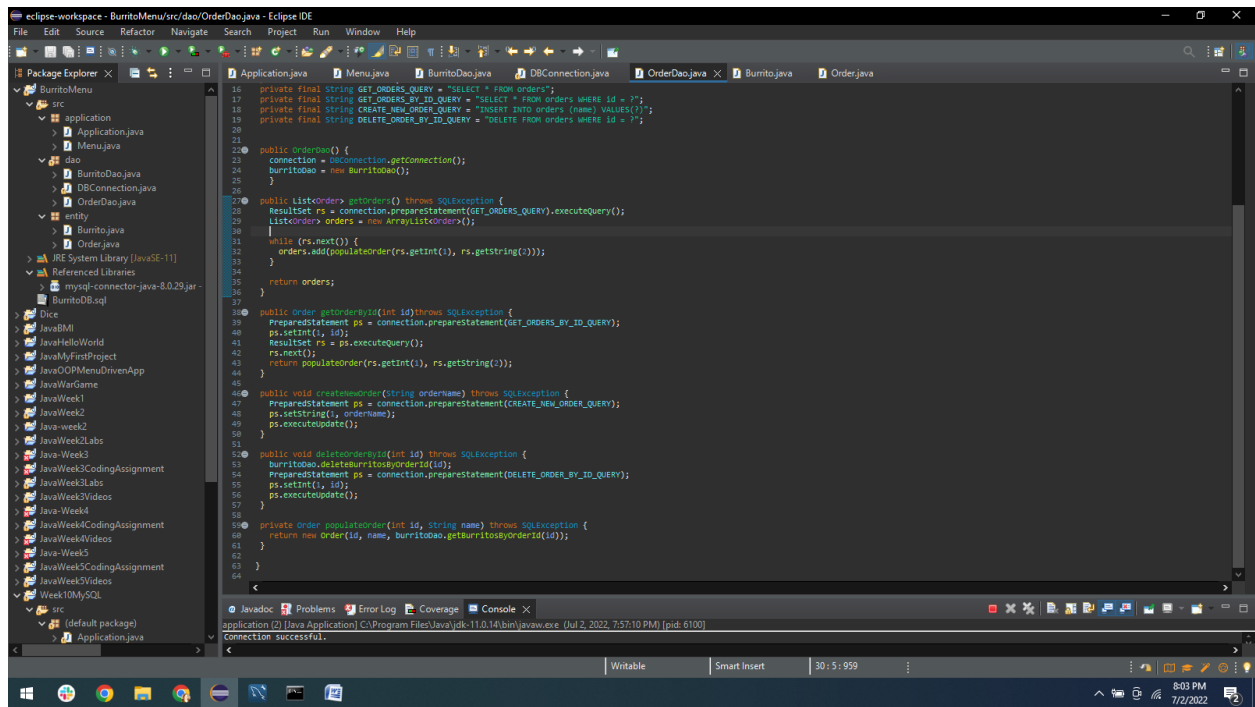


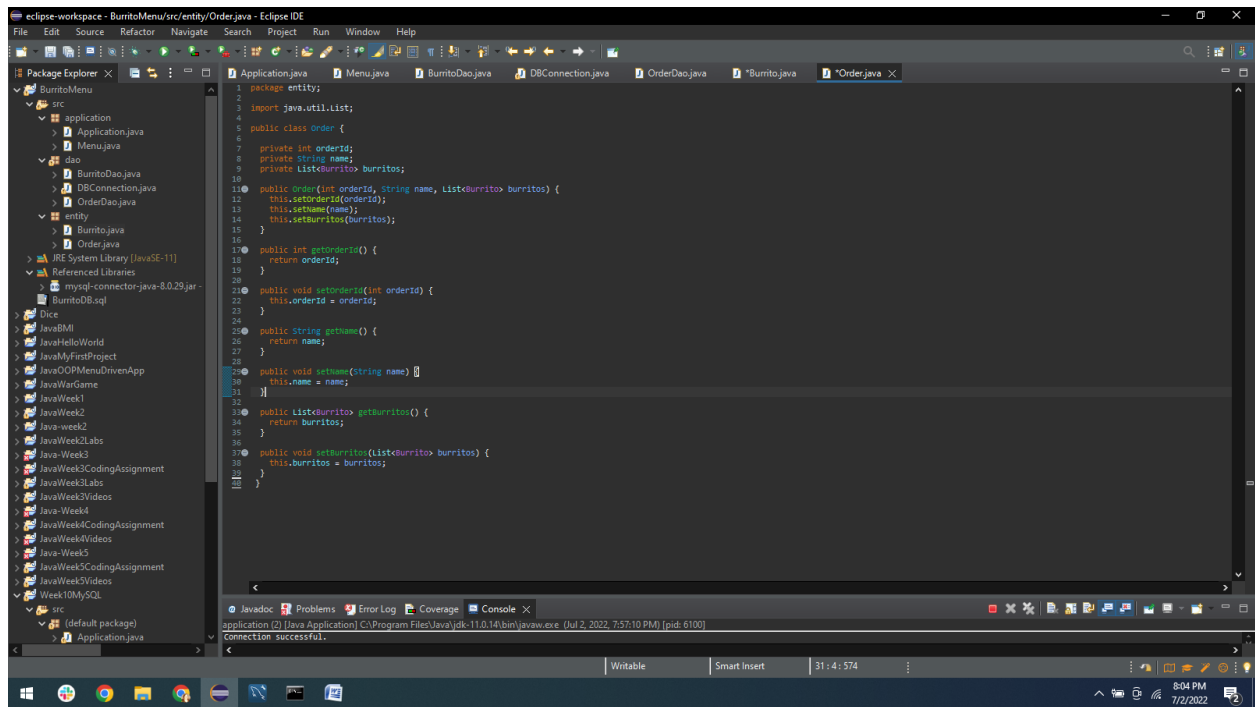
```
1 package application;
2
3 import java.sql.SQLException;
4 import java.util.ArrayList;
5 import java.util.Scanner;
6 import java.util.List;
7 import dao.BurritoDao;
8 import dao.OrderDao;
9 import entity.Burrito;
10 import entity.Order;
11
12 public class Menu {
13
14     private OrderDao orderDao = new OrderDao();
15     private BurritoDao burritoDao = new BurritoDao();
16     private Scanner scanner = new Scanner(System.in);
17     private List<String> options = Arrays.asList(
18         "Display all orders",
19         "Display an individual order",
20         "Create new order",
21         "Delete order",
22         "Add a burrito",
23         "Delete a burrito"
24     );
25
26     public void start() {
27         String selection = "";
28         do {
29             printMenu();
30             selection = scanner.nextLine();
31             try {
32                 if (selection.equals("1")) {
33                     displayOrders();
34                 } else if (selection.equals("2")) {
35                     displayOrder();
36                 } else if (selection.equals("3")) {
37                     createOrder();
38                 } else if (selection.equals("4")) {
39                     deleteOrder();
40                 } else if (selection.equals("5")) {
41                     createBurrito();
42                 } else if (selection.equals("6")) {
43                     deleteBurrito();
44                 }
45             } catch (SQLException e) {
46                 e.printStackTrace();
47             }
48         } while (!selection.equals("q"));
49     }
50
51     private void printMenu() {
52         for (String option : options) {
53             System.out.println(option);
54         }
55     }
56
57     private void displayOrders() {
58         List<Order> orders = orderDao.getAll();
59         for (Order order : orders) {
60             System.out.println(order);
61         }
62     }
63
64     private void displayOrder() {
65         System.out.println("Enter order ID:");
66         int id = scanner.nextInt();
67         Order order = orderDao.findById(id);
68         if (order != null) {
69             System.out.println(order);
70         } else {
71             System.out.println("Order not found");
72         }
73     }
74
75     private void createOrder() {
76         System.out.println("Enter order details:");
77         System.out.println("Enter quantity:");
78         int quantity = scanner.nextInt();
79         System.out.println("Enter burrito ID:");
80         int burritoId = scanner.nextInt();
81         Order order = new Order(quantity, burritoId);
82         orderDao.create(order);
83     }
84
85     private void deleteOrder() {
86         System.out.println("Enter order ID to delete:");
87         int id = scanner.nextInt();
88         Order order = orderDao.findById(id);
89         if (order != null) {
90             orderDao.delete(order);
91         } else {
92             System.out.println("Order not found");
93         }
94     }
95
96     private void createBurrito() {
97         System.out.println("Enter burrito details:");
98         System.out.println("Enter name:");
99         String name = scanner.nextLine();
100        System.out.println("Enter price:");
101        double price = scanner.nextDouble();
102        Burrito burrito = new Burrito(name, price);
103        burritoDao.create(burrito);
104    }
105
106    private void deleteBurrito() {
107        System.out.println("Enter burrito ID to delete:");
108        int id = scanner.nextInt();
109        Burrito burrito = burritoDao.findById(id);
110        if (burrito != null) {
111            burritoDao.delete(burrito);
112        } else {
113            System.out.println("Burrito not found");
114        }
115    }
116}
```



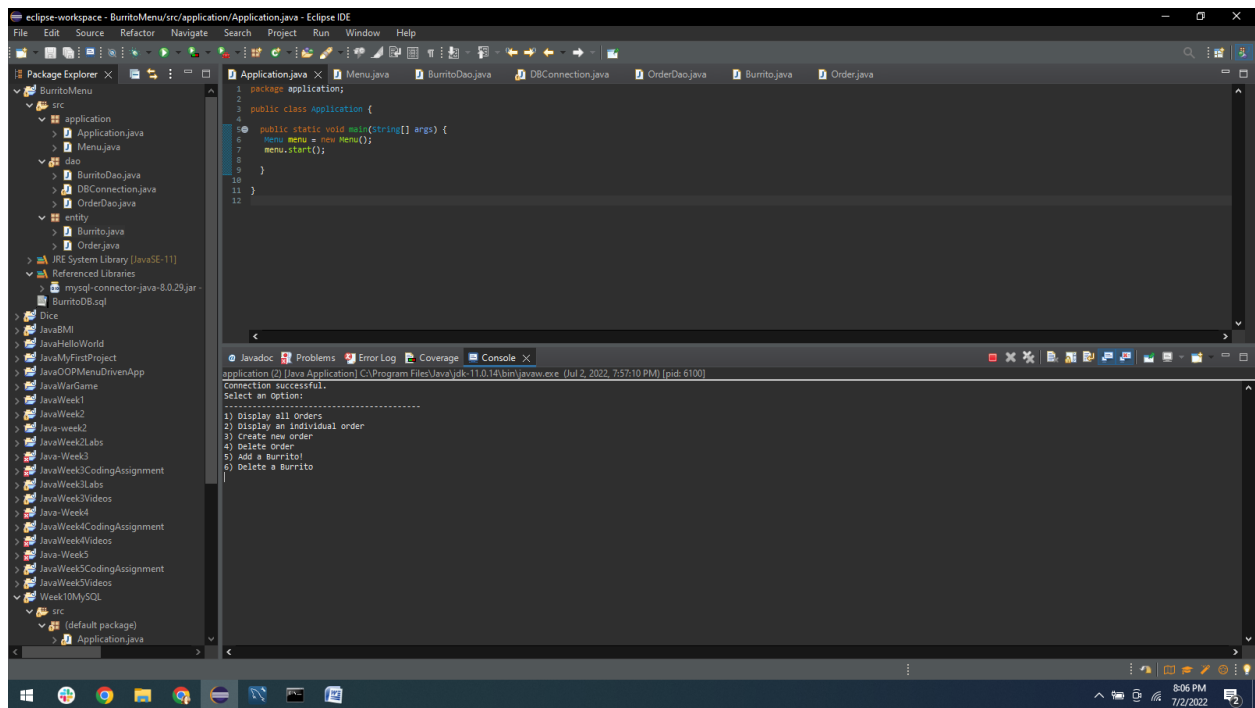


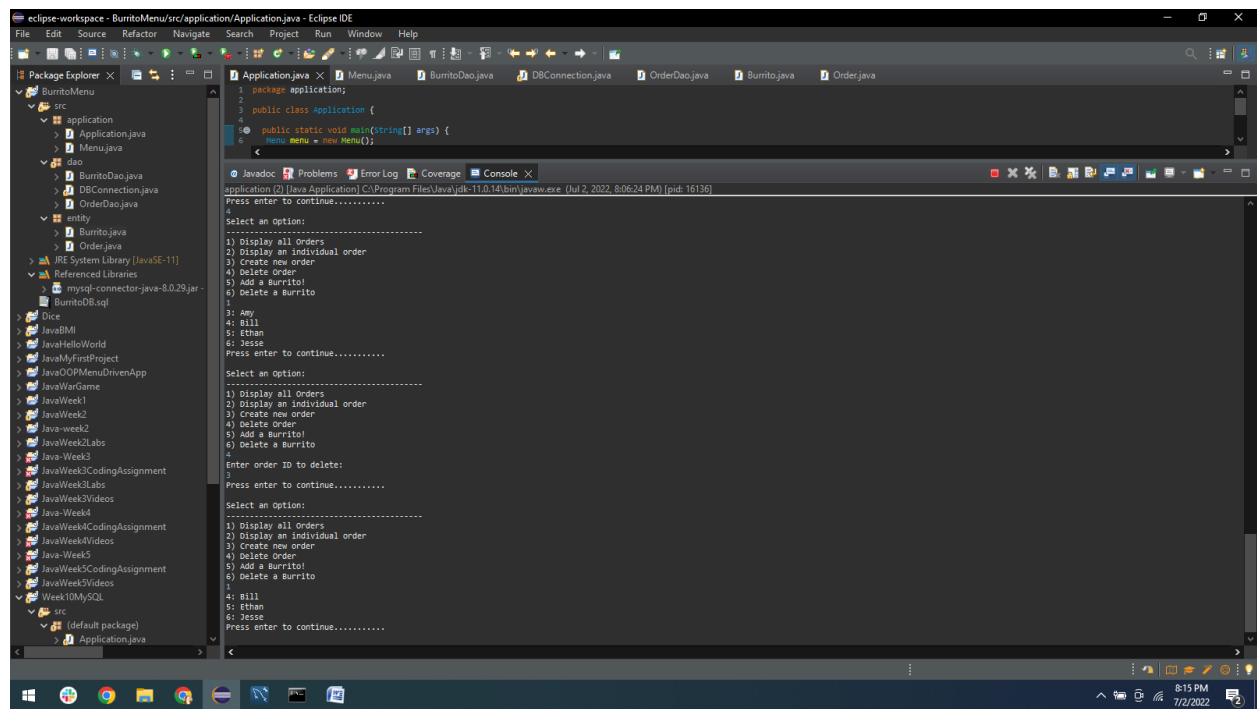






Screenshots of Running Application:





URL to GitHub Repository: <https://github.com/CodingVegas/week10SQLBurritoOrder>