UNIVERSITY OF NORTH TEXAS

COMPUTER SCIENCE DEPARTMENT

CSCE5350/FUNDAMENTALS OF DATABASE SYSTEMS

RESTAURANT MANAGMENT SYSTEM

PROJECT FINAL REPORT

Project Description
ER Model
SQL Statements – Create Tables
SQL Statements – Insert Data
Programming Languages
Software Platforms & Tools
Installation & Execution of Project Code
Application User Flows – Screenshots
Future Implementations

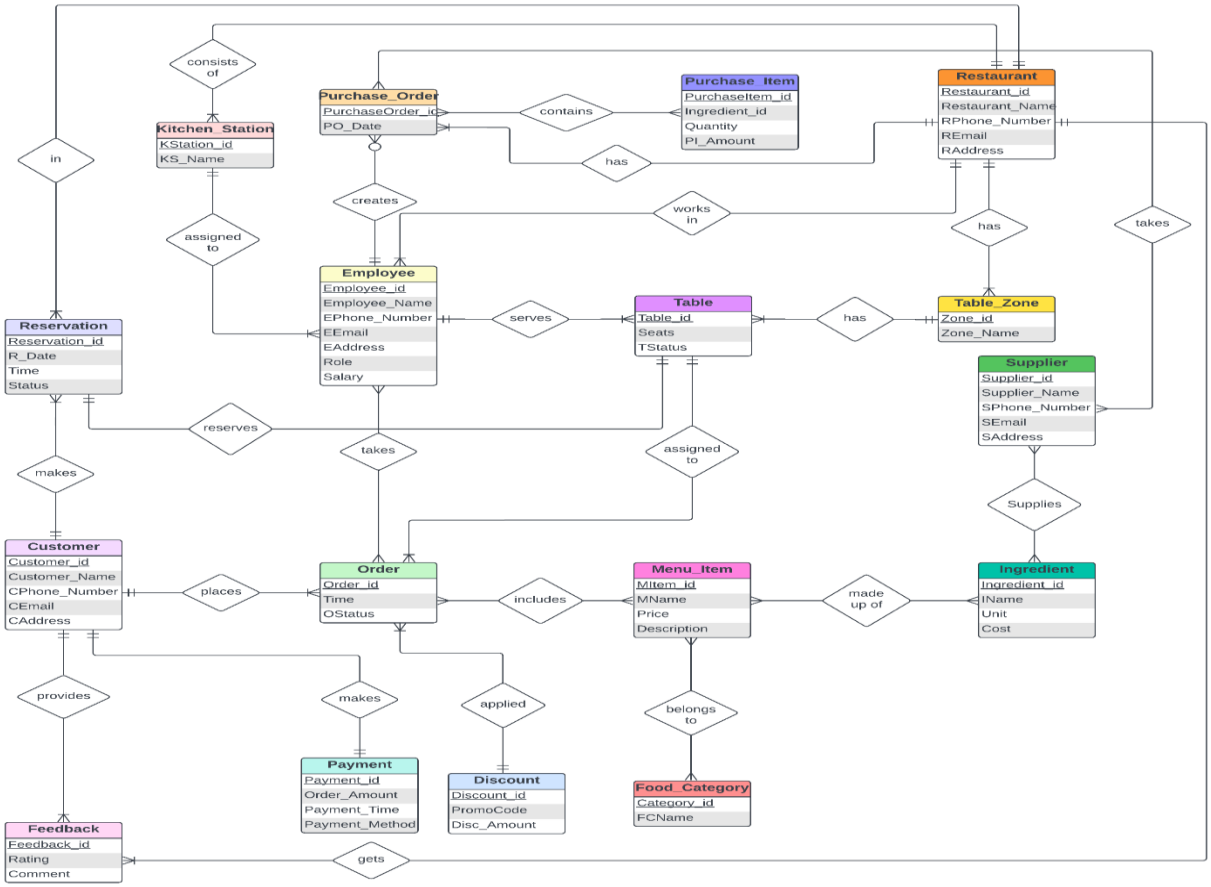
GROUP 9

Harini Kamarthy
Vaishnavi Gunna
Sai Sneha Palle
Sharan Kumar Pallapu
Greeshma Sangaraju
Sahit Reddy Chinthakuntla

Project Description:

Restaurant Management System is a desktop software application designed that helps restaurant owners manage their business operations by automating various tasks such as order management, handling payments, scheduling reservations, storing customer feedbacks, employee management. The system is designed to give best customer experience while streamlining restaurant operations. System is built using Python and CustomTkinter, a custom-built GUI toolkit that extends the functionality of the standard Tkinter library. This application offers an intuitive and user-friendly interface that is easy to navigate. Main Window have two options which provides Employee and Customer flows separately. Here we have concentrated more on Customer flows as it is mainly designed for restaurant to make customer experience as hassle-free. Employee flow can only provide the view of the whole restaurant related data. If Customer is a new person, it grabs his/her data and stores in database. If Customer is an existing one, then it allows to make Reservations, Order the food & make payment, provide the Feedback.

ER Model:



Relational Schema is clearly available in our Project Proposal document, which will also be attached in this project submission files.

SQL Statements – Create Tables

Customer

```
CREATE TABLE `customer` (
  `Customer_id` int NOT NULL,
  `Customer_Name` varchar(50) DEFAULT NULL,
  `CPhone_Number`  varchar(20) DEFAULT NULL,
  `CEmail` varchar(50) DEFAULT NULL,
  `CAddress` varchar(100) DEFAULT NULL,
  PRIMARY KEY (`Customer_id`)
)
```

Discount

```
CREATE TABLE `discount` (
  `Discount_id` int NOT NULL,
  `PromoCode` varchar(20) NOT NULL,
  `Disc_Amount` decimal(10,2) NOT NULL,
  PRIMARY KEY (`Discount_id`)
)
```

Employee

```
CREATE TABLE `employee` (
  `Employee_id` int NOT NULL,
  `Employee_Name` varchar(50) DEFAULT NULL,
  `EPhone_Number`  varchar(20) DEFAULT NULL,
  `EEmail` varchar(50) DEFAULT NULL,
  `EAddress` varchar(100) DEFAULT NULL,
  `Role` varchar(10) DEFAULT NULL,
  `Salary` decimal(10,2) DEFAULT NULL,
  `KStation_id` int DEFAULT NULL,
  `Restaurant_id` int DEFAULT NULL,
  PRIMARY KEY (`Employee_id`),
  KEY `KStation_id` (`KStation_id`),
  KEY `Restaurant_id` (`Restaurant_id`),
  CONSTRAINT `employee_ibfk_1` FOREIGN KEY (`KStation_id`) REFERENCES `kitchen_station`
(`KStation_id`),
  CONSTRAINT `employee_ibfk_2` FOREIGN KEY (`Restaurant_id`) REFERENCES `restaurant`
(`Restaurant_id`)
)
```

Employee_Order

```
CREATE TABLE `employee` (
  `Employee_id` int NOT NULL,
  `Employee_Name` varchar(50) DEFAULT NULL,
```

```
  `EPhone_Number` varchar(20) DEFAULT NULL,
  `EEmail` varchar(50) DEFAULT NULL,
  `EAddress` varchar(100) DEFAULT NULL,
  `Role` varchar(10) DEFAULT NULL,
  `Salary` decimal(10,2) DEFAULT NULL,
  `KStation_id` int DEFAULT NULL,
  `Restaurant_id` int DEFAULT NULL,
  PRIMARY KEY (`Employee_id`),
  KEY `KStation_id` (`KStation_id`),
  KEY `Restaurant_id` (`Restaurant_id`),
  CONSTRAINT `employee_ibfk_1` FOREIGN KEY (`KStation_id`) REFERENCES `kitchen_station`
(`KStation_id`),
  CONSTRAINT `employee_ibfk_2` FOREIGN KEY (`Restaurant_id`) REFERENCES `restaurant`
(`Restaurant_id`)
)
```

Feedback

```
CREATE TABLE `feedback` (
  `Feedback_id` int NOT NULL,
  `Customer_id` int DEFAULT NULL,
  `Restaurant_id` int DEFAULT NULL,
  `Rating` int DEFAULT NULL,
  `Comment` text,
  PRIMARY KEY (`Feedback_id`),
  KEY `Customer_id` (`Customer_id`),
  KEY `Restaurant_id` (`Restaurant_id`),
  CONSTRAINT `feedback_ibfk_1` FOREIGN KEY (`Customer_id`) REFERENCES `customer`
(`Customer_id`),
  CONSTRAINT `feedback_ibfk_2` FOREIGN KEY (`Restaurant_id`) REFERENCES `restaurant`
(`Restaurant_id`)
)
```

Food_Category

```
CREATE TABLE `food_category` (
  `Category_id` int NOT NULL,
  `FCName` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`Category_id`)
)
```

Ingredient

```
CREATE TABLE `ingredient` (
  `Ingredient_id` int NOT NULL,
  `IName` varchar(255) DEFAULT NULL,
  `Unit` varchar(50) DEFAULT NULL,
```

```
  `Cost` decimal(10,2) DEFAULT NULL,
  PRIMARY KEY (`Ingredient_id`)
)
```

Kitchen_Station

```
CREATE TABLE `kitchen_station` (
  `KStation_id` int NOT NULL,
  `KS_Name` varchar(50) NOT NULL,
  `Restaurant_id` int NOT NULL,
  PRIMARY KEY (`KStation_id`),
  KEY `Restaurant_id` (`Restaurant_id`),
  CONSTRAINT `kitchen_station_ibfk_1` FOREIGN KEY (`Restaurant_id`) REFERENCES `restaurant`
(`Restaurant_id`)
)
```

Menu_Item

```
CREATE TABLE `menu_item` (
  `MItem_id` int NOT NULL,
  `MName` varchar(255) NOT NULL,
  `Price` decimal(10,2) NOT NULL,
  `Description` text,
  PRIMARY KEY (`MItem_id`)
)
```

MenuItem_Category

```
CREATE TABLE `menuitem_category` (
  `MItem_id` int NOT NULL,
  `Category_id` int NOT NULL,
  PRIMARY KEY (`MItem_id`,`Category_id`),
  KEY `Category_id` (`Category_id`),
  CONSTRAINT `menuitem_category_ibfk_1` FOREIGN KEY (`MItem_id`) REFERENCES
`menu_item` (`MItem_id`),
  CONSTRAINT `menuitem_category_ibfk_2` FOREIGN KEY (`Category_id`) REFERENCES
`food_category` (`Category_id`)
)
```

MenuItem_Ingredients

```
CREATE TABLE `menuitem_ingredients` (
  `MItem_id` int NOT NULL,
  `Ingredient_id` int NOT NULL,
  PRIMARY KEY (`MItem_id`,`Ingredient_id`),
  KEY `Ingredient_id` (`Ingredient_id`),
  CONSTRAINT `menuitem_ingredients_ibfk_1` FOREIGN KEY (`MItem_id`) REFERENCES
`menu_item` (`MItem_id`),
```

```
  CONSTRAINT `menuitem_ingredients_ibfk_2` FOREIGN KEY (`Ingredient_id`) REFERENCES `ingredient` (`Ingredient_id`)
)
```

Order

```
CREATE TABLE `order` (
  `Order_id` int NOT NULL,
  `Table_id` int DEFAULT NULL,
  `Customer_id` int DEFAULT NULL,
  `Discount_id` int DEFAULT NULL,
  `Time` datetime DEFAULT NULL,
  `Status` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`Order_id`),
  KEY `Table_id` (`Table_id`),
  KEY `Customer_id` (`Customer_id`),
  KEY `Discount_id` (`Discount_id`),
  CONSTRAINT `order_ibfk_1` FOREIGN KEY (`Table_id`) REFERENCES `table` (`Table_id`),
  CONSTRAINT `order_ibfk_2` FOREIGN KEY (`Customer_id`) REFERENCES `customer` (`Customer_id`),
  CONSTRAINT `order_ibfk_3` FOREIGN KEY (`Discount_id`) REFERENCES `discount` (`Discount_id`)
)
```

Order_Menu

```
CREATE TABLE `order_menu` (
  `Order_id` int NOT NULL,
  `MItem_id` int NOT NULL,
  PRIMARY KEY (`Order_id`,`MItem_id`),
  KEY `MItem_id` (`MItem_id`),
  CONSTRAINT `order_menu_ibfk_1` FOREIGN KEY (`Order_id`) REFERENCES `order` (`Order_id`),
  CONSTRAINT `order_menu_ibfk_2` FOREIGN KEY (`MItem_id`) REFERENCES `menu_item` (`MItem_id`)
)
```

Payment

```
CREATE TABLE `payment` (
  `Payment_id` int NOT NULL,
  `Customer_id` int DEFAULT NULL,
  `Order_id` int DEFAULT NULL,
  `Order_Amount` decimal(10,2) DEFAULT NULL,
  `Payment_Time` datetime DEFAULT NULL,
  `Payment_Method` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`Payment_id`),
  KEY `Order_id` (`Order_id`),
```

```
  KEY `Customer_id` (`Customer_id`),
  CONSTRAINT `payment_ibfk_1` FOREIGN KEY (`Order_id`) REFERENCES `order` (`Order_id`),
  CONSTRAINT `payment_ibfk_2` FOREIGN KEY (`Customer_id`) REFERENCES `customer` (`Customer_id`)
)
```

Purchase_Item

```
CREATE TABLE `purchase_item` (
  `PurchaseItem_id` int NOT NULL,
  `Ingredient_id` int DEFAULT NULL,
  `Quantity` int DEFAULT NULL,
  `PI_Amount` decimal(10,2) DEFAULT NULL,
  PRIMARY KEY (`PurchaseItem_id`),
  KEY `Ingredient_id` (`Ingredient_id`),
  CONSTRAINT `purchase_item_ibfk_1` FOREIGN KEY (`Ingredient_id`) REFERENCES `ingredient` (`Ingredient_id`)
)
```

Purchase_Order

```
CREATE TABLE `purchase_order` (
  `PurchaseOrder_id` int NOT NULL,
  `PO_Date` date DEFAULT NULL,
  `Restaurant_id` int DEFAULT NULL,
  `Employee_id` int DEFAULT NULL,
  `Supplier_id` int DEFAULT NULL,
  PRIMARY KEY (`PurchaseOrder_id`),
  KEY `Restaurant_id` (`Restaurant_id`),
  KEY `Employee_id` (`Employee_id`),
  KEY `Supplier_id` (`Supplier_id`),
  CONSTRAINT `purchase_order_ibfk_1` FOREIGN KEY (`Restaurant_id`) REFERENCES `restaurant` (`Restaurant_id`),
  CONSTRAINT `purchase_order_ibfk_2` FOREIGN KEY (`Employee_id`) REFERENCES `employee` (`Employee_id`),
  CONSTRAINT `purchase_order_ibfk_3` FOREIGN KEY (`Supplier_id`) REFERENCES `supplier` (`Supplier_id`)
)
```

PurchaseOrder_Items

```
CREATE TABLE `purchaseorder_items` (
  `PurchaseOrder_id` int NOT NULL,
  `PurchaseItem_id` int NOT NULL,
  `PStatus` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`PurchaseOrder_id`,`PurchaseItem_id`),
  KEY `PurchaseItem_id` (`PurchaseItem_id`),
```

```
  CONSTRAINT `purchaseorder_items_ibfk_1` FOREIGN KEY (`PurchaseOrder_id`) REFERENCES `purchase_order` (`PurchaseOrder_id`),
  CONSTRAINT `purchaseorder_items_ibfk_2` FOREIGN KEY (`PurchaseItem_id`) REFERENCES `purchase_item` (`PurchaseItem_id`)
)
```

Reservation

```
CREATE TABLE `reservation` (
  `Reservation_id` int NOT NULL,
  `Customer_id` int DEFAULT NULL,
  `Restaurant_id` int DEFAULT NULL,
  `Table_id` int DEFAULT NULL,
  `R_Date` date DEFAULT NULL,
  `Time` time DEFAULT NULL,
  `Status` varchar(50) DEFAULT NULL,
  PRIMARY KEY (`Reservation_id`),
  KEY `Customer_id` (`Customer_id`),
  KEY `Restaurant_id` (`Restaurant_id`),
  KEY `Table_id` (`Table_id`),
  CONSTRAINT `reservation_ibfk_1` FOREIGN KEY (`Customer_id`) REFERENCES `customer` (`Customer_id`),
  CONSTRAINT `reservation_ibfk_2` FOREIGN KEY (`Restaurant_id`) REFERENCES `restaurant` (`Restaurant_id`),
  CONSTRAINT `reservation_ibfk_3` FOREIGN KEY (`Table_id`) REFERENCES `table` (`Table_id`)
)
```

Restaurant

```
CREATE TABLE `restaurant` (
  `Restaurant_id` int NOT NULL,
  `Restaurant_Name` varchar(50) DEFAULT NULL,
  `RPhone_Number`  varchar(20) DEFAULT NULL,
  `REmail` varchar(50) DEFAULT NULL,
  `RAddress` varchar(100) DEFAULT NULL,
  PRIMARY KEY (`Restaurant_id`)
)
```

Supplier

```
CREATE TABLE `supplier` (
  `Supplier_id` int NOT NULL,
  `Supplier_Name` varchar(255) DEFAULT NULL,
  `SPhone_Number`  varchar(20) DEFAULT NULL,
  `SEmail` varchar(255) DEFAULT NULL,
  `SAddress` varchar(255) DEFAULT NULL,
  PRIMARY KEY (`Supplier_id`)
```

```
)
```

Supplier_Ingredients

```
CREATE TABLE `supplier_ingredients` (
  `Supplier_id` int NOT NULL,
  `Ingredient_id` int NOT NULL,
  PRIMARY KEY (`Supplier_id`,`Ingredient_id`),
  KEY `Ingredient_id` (`Ingredient_id`),
  CONSTRAINT `supplier_ingredients_ibfk_1` FOREIGN KEY (`Supplier_id`) REFERENCES `supplier`
(`Supplier_id`),
  CONSTRAINT `supplier_ingredients_ibfk_2` FOREIGN KEY (`Ingredient_id`) REFERENCES
`ingredient` (`Ingredient_id`)
)
```

SupplierPurchase_Order

```
CREATE TABLE `supplierpurchase_order` (
  `Supplier_id` int NOT NULL,
  `PurchaseOrder_id` int NOT NULL,
  PRIMARY KEY (`Supplier_id`,`PurchaseOrder_id`),
  KEY `PurchaseOrder_id` (`PurchaseOrder_id`),
  CONSTRAINT `supplierpurchase_order_ibfk_1`  FOREIGN KEY (`Supplier_id`) REFERENCES
`supplier` (`Supplier_id`),
  CONSTRAINT `supplierpurchase_order_ibfk_2`  FOREIGN KEY (`PurchaseOrder_id`) REFERENCES
`purchase_order` (`PurchaseOrder_id`)
)
```

Table

```
CREATE TABLE `table` (
  `Table_id` int NOT NULL,
  `Zone_id` int DEFAULT NULL,
  `Employee_id` int DEFAULT NULL,
  `Seats` int DEFAULT NULL,
  `TStatus` varchar(20) DEFAULT NULL,
  PRIMARY KEY (`Table_id`),
  KEY `Zone_id` (`Zone_id`),
  KEY `Employee_id` (`Employee_id`),
  CONSTRAINT `table_ibfk_1` FOREIGN KEY (`Zone_id`) REFERENCES `table_zone` (`Zone_id`),
  CONSTRAINT `table_ibfk_2` FOREIGN KEY (`Employee_id`) REFERENCES `employee`
(`Employee_id`)
)
```

Table_Zone

```
CREATE TABLE `table_zone` (
  `Zone_id` int NOT NULL,
```

```
  `Zone_Name` varchar(50) NOT NULL,
  `Restaurant_id` int NOT NULL,
  PRIMARY KEY (`Zone_id`),
  KEY `Restaurant_id` (`Restaurant_id`),
  CONSTRAINT `table_zone_ibfk_1` FOREIGN KEY (`Restaurant_id`) REFERENCES `restaurant`
(`Restaurant_id`)
)
```

SQL Statements – Insert Data

Customer

```
INSERT INTO customer (Customer_id, Customer_Name, CPhone_Number,  CEmail, CAddress)
VALUES
   (1, 'John Doe', '123-456-7890', 'johndoe@example.com', '123 Main St'),
   (2, 'Jane Smith', '555-555-5555', 'janesmith@example.com', '456 Elm St'),
   (3, 'Bob Johnson', '999-999-9999', 'bob@example.com', '789 Oak St'),
   (4, 'Alice Brown', '111-111-1111', 'alice@example.com', '456 Maple St'),
   (5, 'Tom Wilson', '222-222-2222', 'tom@example.com', '789 Pine St'),
   (6, 'Samantha Lee', '333-333-3333', 'samantha@example.com', '123 Cedar St');
```

Restaurant

```
INSERT INTO restaurant (Restaurant_id, Restaurant_Name, RPhone_Number, REmail, RAddress)
VALUES
   (1, 'The Best Burger', '123-456-7890', 'info@thebestburger.com', '456 Main St'),
   (2, 'Pizza Palace', '555-555-5555', 'contact@pizzapalace.com', '789 Elm St'),
   (3, 'Taco Time', '999-999-9999', 'info@tacotime.com', '123 Oak St');
```

Kitchen_Station

```
INSERT INTO kitchen_station (KStation_id, KS_Name, Restaurant_id)
VALUES (1, 'Grill Station', 1), (2, 'Pizza Station', 2), (3, 'Taco Station', 3);
```

Employee

```
INSERT INTO employee (Employee_id, Employee_Name, EPhone_Number, EEmail, EAddress,
Role, Salary, KStation_id, Restaurant_id)
VALUES
   -- Restaurant 1
   (1, 'John Smith', '123-456-7890', 'john.smith@restaurant1.com', '123 Main St', 'Manager',
50000, NULL, 1),
   (2, 'Jane Doe', '555-555-5555', 'jane.doe@restaurant1.com', '456 Elm St', 'Server', 20000,
NULL, 1),
   (3, 'Samantha Brown', '999-999-9999', 'samantha.brown@restaurant1.com', '789 Oak St',
'Chef', 60000, 1, 1),
   (4, 'Michael Johnson', '111-111-1111', 'michael.johnson@restaurant1.com', '1010 Broad St',
'Host', 18000, NULL, 1),
```

(5, 'Sarah Lee', '222-222-2222', 'sarah.lee@restaurant1.com', '1313 5th Ave', 'Chef', 60000, 2, 1),
(6, 'Peter Green', '333-333-3333', 'peter.green@restaurant1.com', '1212 6th Ave', 'Chef', 60000, 3, 1),
(7, 'Alex Davis', '444-444-4444', 'alex.davis@restaurant2.com', '222 Main St', 'Manager', 50000, NULL, 1),
(8, 'Emily Wilson', '777-777-7777', 'emily.wilson@restaurant2.com', '333 Elm St', 'Server', 20000, NULL, 1),
(9, 'Olivia Lee', '999-999-9999', 'olivia.lee@restaurant2.com', '555 Broad St', 'Host', 18000, NULL, 1),
(10, 'Jessica Smith', '666-666-6666', 'jessica.smith@restaurant3.com', '444 Elm St', 'Server', 20000, NULL, 1);

Table_Zone

```
INSERT INTO table_zone (Zone_id, Zone_Name, Restaurant_id)
VALUES
  -- Restaurant 1
  (1, 'Zone 1', 1), (2, 'Zone 2', 1), (3, 'Zone 3', 1);
```

Table

```
INSERT INTO `Table` (Table_id, Zone_id, Employee_id, Seats, TStatus)
VALUES
  (1, 1, 1, 4, 'Available'), (2, 1, 2, 2, 'Available'), (3, 1, 3, 6, 'Occupied'), (4, 2, 4, 4, 'Available'),
  (5, 2, 5, 2, 'Occupied'), (6, 2, 7, 6, 'Available'), (7, 3, 6, 4, 'Available'), (8, 3, 8, 2, 'Available'),
  (9, 3, 9, 6, 'Available');
```

Reservation

```
INSERT INTO reservation (Reservation_id, Customer_id, Restaurant_id, Table_id, R_Date, Time, Status)
VALUES
  (1, 1, 1, 1, '2023-04-03', '18:30:00', 'Confirmed'), (2, 2, 1, 2, '2023-04-03', '19:00:00', 'Confirmed'), (3, 3, 1, 4, '2023-04-03', '19:30:00', 'Confirmed'), (4, 4, 1, 5, '2023-04-03', '20:00:00', 'Confirmed'), (5, 5, 1, 7, '2023-04-03', '20:30:00', 'Confirmed'), (6, 6, 1, 8, '2023-04-03', '21:00:00', 'Confirmed');
```

Discount

```
INSERT INTO discount (Discount_id, PromoCode, Disc_Amount)
VALUES
  (1, 'SUMMER10', 10.00), (2, 'FALL20', 20.00), (3, 'WINTER30', 30.00);
```

Order

```
INSERT INTO `order` (Order_id, Table_id, Customer_id, Discount_id, Time, Status)
VALUES
  (1, 1, 1, NULL, '2023-04-02 18:00:00', 'Close'), (2, 2, 2, NULL, '2023-04-02 18:15:00', 'Close'),
```

(3, 3, 3, NULL, '2023-04-02 18:30:00', 'Close'), (4, 4, 4, NULL, '2023-04-02 18:45:00', 'Close'),
(5, 5, 5, NULL, '2023-04-02 19:00:00', 'Close'), (6, 6, 6, NULL, '2023-04-02 19:15:00', 'Close'),
(7, 7, 1, 1, '2023-04-02 19:30:00', 'Open'), (8, 8, 2, 2, '2023-04-02 19:45:00', 'Open'),
(9, 9, 3, 3, '2023-04-02 20:00:00', 'Open');

Payment

INSERT INTO payment (Payment_id, Customer_id, Order_id, Order_Amount, Payment_Time, Payment_Method)
VALUES
(1, 1, 1, 50.00, '2023-04-02 19:30:00', 'Cash'), (2, 2, 2, 35.00, '2023-04-02 20:00:00', 'Credit Card'), (3, 3, 3, 75.00, '2023-04-02 20:30:00', 'Debit Card'), (4, 4, 4, 42.00, '2023-04-02 21:00:00', 'Others');

Feedback

INSERT INTO feedback (Feedback_id, Customer_id, Restaurant_id, Rating, Comment)
VALUES
(1, 1, 1, 4, 'The food was great, but the service could have been better.'),
(2, 2, 1, 5, 'The pizza was amazing, and the service was top-notch!'),
(3, 3, 1, 3, 'The tacos were average, and the service was slow.'),
(4, 4, 1, 5, 'The burgers were delicious, and the service was friendly.'),
(5, 5, 1, 4, 'The pasta was good, but the service was a bit slow.'),
(6, 6, 1, 2, 'The food was terrible, and the service was rude.');

Menu_Item

INSERT INTO menu_item (MItem_id, MName, Price, Description) VALUES
(1, 'Classic Burger', 8.99, 'A juicy beef patty topped with lettuce, tomato, onion, and pickles'),
(2, 'Chicken Caesar Salad', 10.99, 'Romaine lettuce, grilled chicken, croutons, and Parmesan cheese'),
(3, 'Margherita Pizza', 12.99, 'Tomato sauce, fresh mozzarella, and basil on a thin crust'),
(4, 'BBQ Ribs Pasta', 16.99, 'Slow-cooked and basted in our signature BBQ sauce'),
(5, 'Fish and Chips Sides', 14.99, 'Beer-battered cod served with fries and tartar sauce'),
(6, 'Spaghetti Bolognese Pasta', 11.99, 'Spaghetti with a meaty tomato sauce and Parmesan cheese'),
(7, 'Avacado Burger', 10.99, 'Tasty burger with avocado and spicy sauce'),
(8, 'Healthy Salad', 9.99, 'Healthy salad with mixed greens and veggies'),
(9, 'Chicken Quinoa Salad', 11.99, 'Fresh salad with grilled chicken and quinoa'),
(10, 'Classic Pizza', 12.99, 'Classic pizza with pepperoni and mushrooms'),
(11, 'Gourmet pizza', 13.99, 'Gourmet pizza with prosciutto and arugula'),
(12, 'Traditional pasta', 11.99, 'Traditional pasta with meat sauce'),
(13, 'French Fries Sides', 3.99, 'Crispy french fries'),
(14, 'Onion Rings Sides', 4.99, 'Golden Onion Rings'),
(15, 'Soda Drink', 2.99, 'Refreshing soda'),
(16, 'Lemon Tea Drink', 3.99, 'Iced tea with lemon');

Food_Category

INSERT INTO food_category (Category_id, FCName) VALUES
(1, 'Burger'), (2, 'Salad'), (3, 'Pizza'), (4, 'Drink'), (5, 'Sides'), (6, 'Pasta');

Ingredient

INSERT INTO ingredient (Ingredient_id, IName, Unit, Cost) VALUES
(1, 'Ground Beef', 'lb', 3.99), (2, 'Chicken Breast', 'lb', 4.99), (3, 'Romaine Lettuce', 'head', 1.99), (4, 'Tomatoes', 'lb', 2.99), (5, 'Onions', 'lb', 1.49), (6, 'Pickles', 'jar', 2.49), (7, 'Pizza Dough', 'lb', 2.99), (8, 'Mozzarella Cheese', 'lb', 5.99), (9, 'Basil', 'bunch', 1.99), (10, 'Pork Ribs', 'lb', 5.99), (11, 'Cod Fillets', 'lb', 7.99), (12, 'Potatoes', 'lb', 0.99), (13, 'Flour', 'lb', 1.99), (14, 'Tomato Sauce', 'can', 1.49), (15, 'Ground Pork', 'lb', 2.99);

Supplier

INSERT INTO supplier (Supplier_id, Supplier_Name, SPhone_Number, SEmail, SAddress) VALUES
(1, 'FreshMeat Inc.', '555-1234', 'info@freshmeat.com', '123 Main St'),
(2, 'Greens Galore', '555-5678', 'info@greensgalore.com', '456 Elm St'),
(3, 'Pizzaiolo Supplies', '555-9999', 'info@pizzaiolosupplies.com', '789 Oak St');

Purchase_Order

INSERT INTO purchase_order (PurchaseOrder_id, PO_Date, Restaurant_id, Employee_id, Supplier_id) VALUES
(1, '2023-04-03', 1, 1, 1), (2, '2023-04-03', 1, 7, 2), (3, '2023-04-03', 1, 1, 3);

Purchase_Item

INSERT INTO purchase_item (PurchaseItem_id, Ingredient_id, Quantity, PI_Amount)
VALUES
(1, 2, 5, 10.50), (2, 1, 2, 6.00), (3, 3, 3, 8.25), (4, 4, 1, 3.50), (5, 5, 4, 12.00), (6, 6, 2, 7.00), (7, 7, 3, 9.75), (8, 8, 1, 4.00), (9, 9, 4, 11.50), (10, 10, 2, 6.00), (11, 11, 3, 8.25), (12, 12, 1, 3.50), (13, 13, 4, 12.00), (14, 14, 2, 7.00), (15, 15, 3, 9.75);

Employee_Order

INSERT INTO Employee_Order (Employee_id, Order_id, EO_Date)
VALUES (1, 1, '2023-04-02'), (2, 1, '2023-04-02'), (3, 2, '2023-04-02');

Order_Menu

INSERT INTO Order_Menu (Order_id, MItem_id)
VALUES (1, 1), (1, 2), (2, 3);

MenuItem_Ingredients

INSERT INTO MenuItem_Ingredients (MItem_id, Ingredient_id)
VALUES (1, 1), (1, 2), (2, 3), (3, 4);

MenuItem_Category

INSERT INTO MenuItem_Category (MItem_id, Category_id)

VALUES (1, 1), (2, 2), (3, 3);

Supplier_Ingredients
INSERT INTO Supplier_Ingredients (Supplier_id, Ingredient_id)
VALUES (1, 1), (1, 2), (2, 3), (3, 4);

PurchaseOrder_Items
INSERT INTO PurchaseOrder_Items (PurchaseOrder_id, PurchaseItem_id, PStatus) VALUES (1, 1, 'Ordered'), (1, 2, 'Delivered'), (2, 3, 'Ordered');

SupplierPurchase_Order
INSERT INTO SupplierPurchase_Order (Supplier_id, PurchaseOrder_id)
VALUES (1, 1), (2, 2), (3, 3);

Programming Languages

- Python
- SQL

Software Platforms & Tools

- My SQL WorkBench
- Microsoft Visual Studio / Jupyter NoteBook / Anaconda
- CustomTkinter GUI

Installation & Execution of Project Code

Pre-requisites:

- Install MySQL Workbench, Python Libraries, Jupyter Notebook, CustomTkinter Packages

Setup to run the project:

- Unzip the file 'Restaurant Project – Harini Kamarthy.zip'
- Database files are exported to 'Restaurant Project – Harini Kamarthy/Database' folder
- Create a database in MySQL Workbench and import these Database files into it

Steps to execute the project/application:

- Open command prompt & Navigate to "Restaurant Project – Harini Kamarthy/UI Code" folder.
- Give command "jupyter notebook".
- From the opened weblink, open file "Restaurant Project UI" file.
- Tap on the code cell & click on "Run" button.

Code used to connect to the Database

- Imported the package, **mysql.connector** into the code file

- Piece of code ---

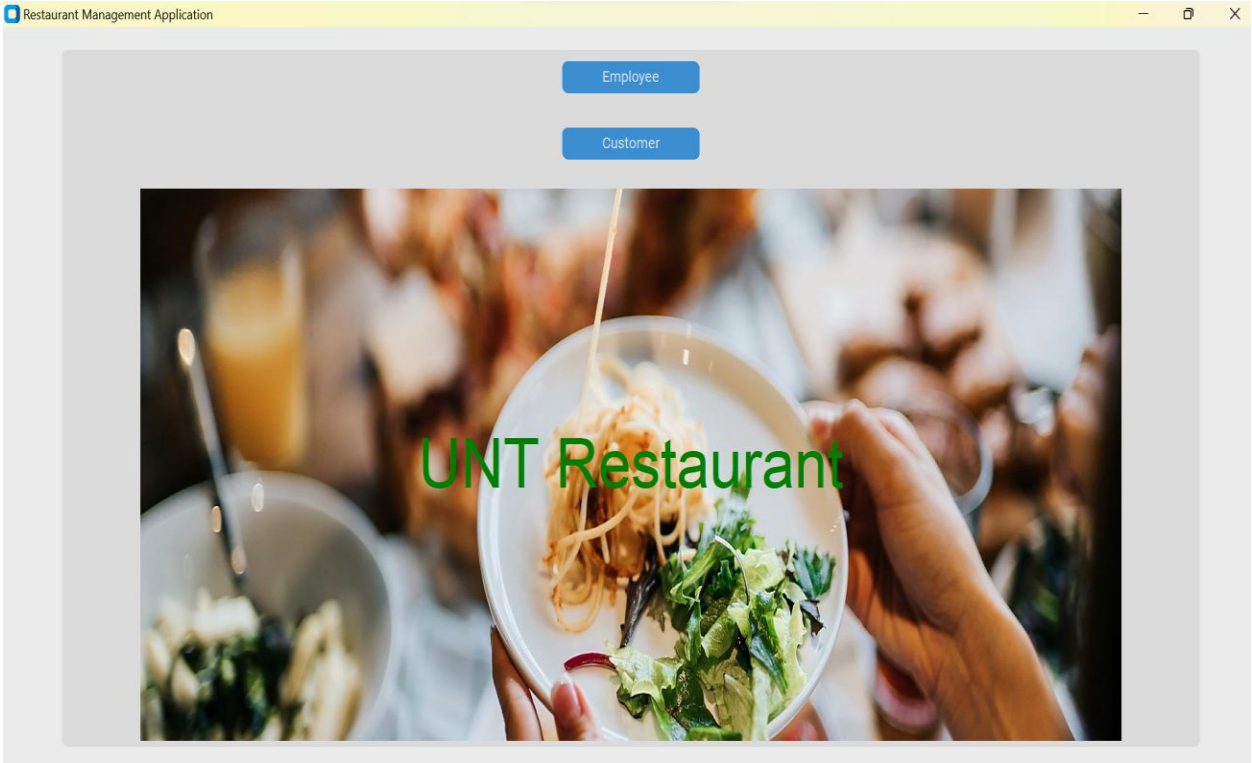
```
# Connect to the database
db = mysql.connector.connect(
        host="127.0.0.1",
        user="root",
        password="Harini@1234",
        database="restaurant_project"
)
```

```
# Connect to the database
db = mysql.connector.connect(
    host="127.0.0.1",
    user="root",
    password="Harini@1234",
    database="restaurant_project"
)
```

Application User Flows – Screenshots

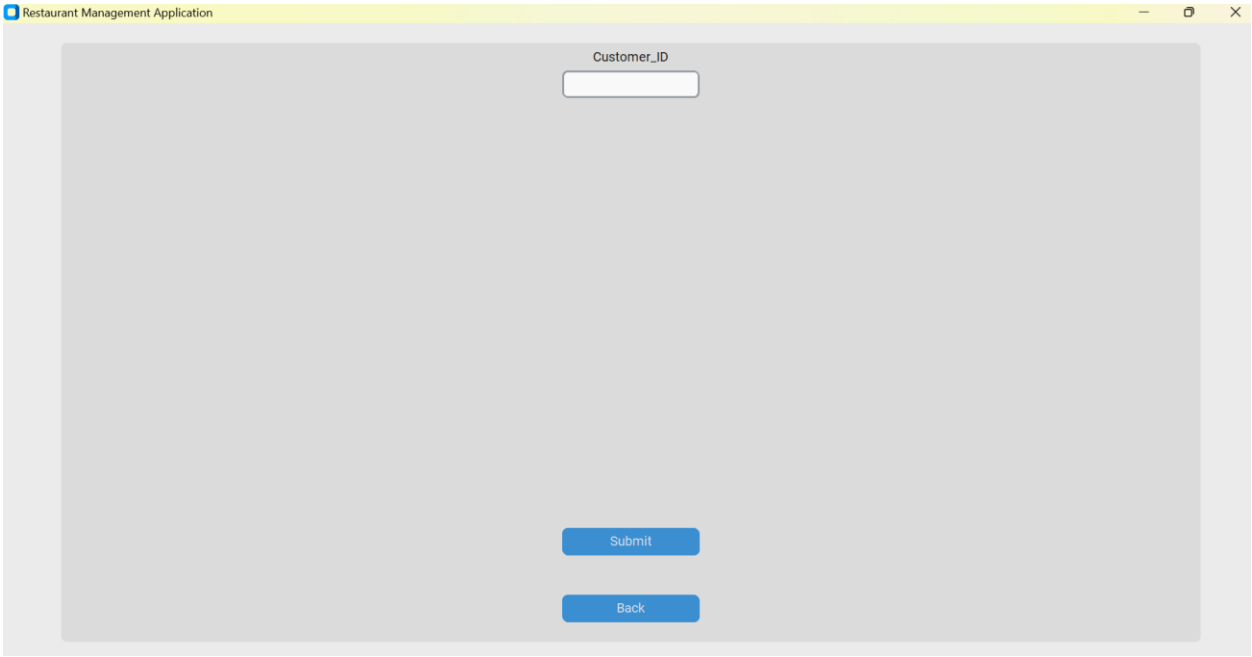
Main Screen:

On Main Screen, we have two options (Buttons) Employee and Customer for their respective flows.



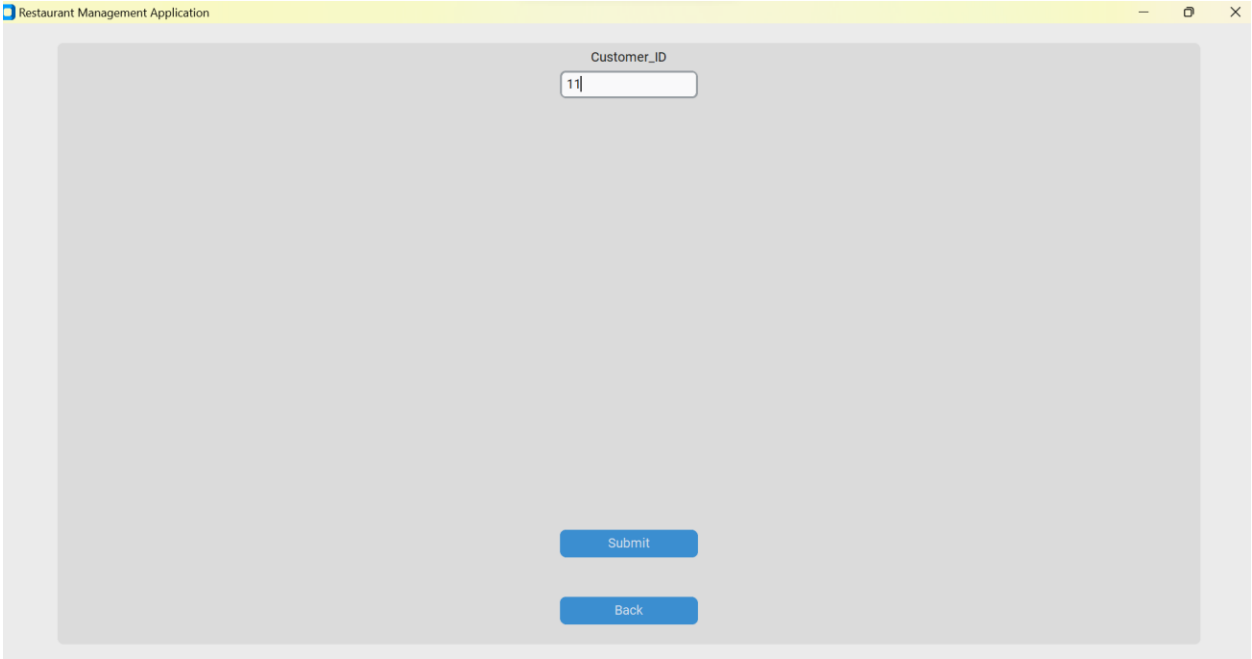
Customer Screen:

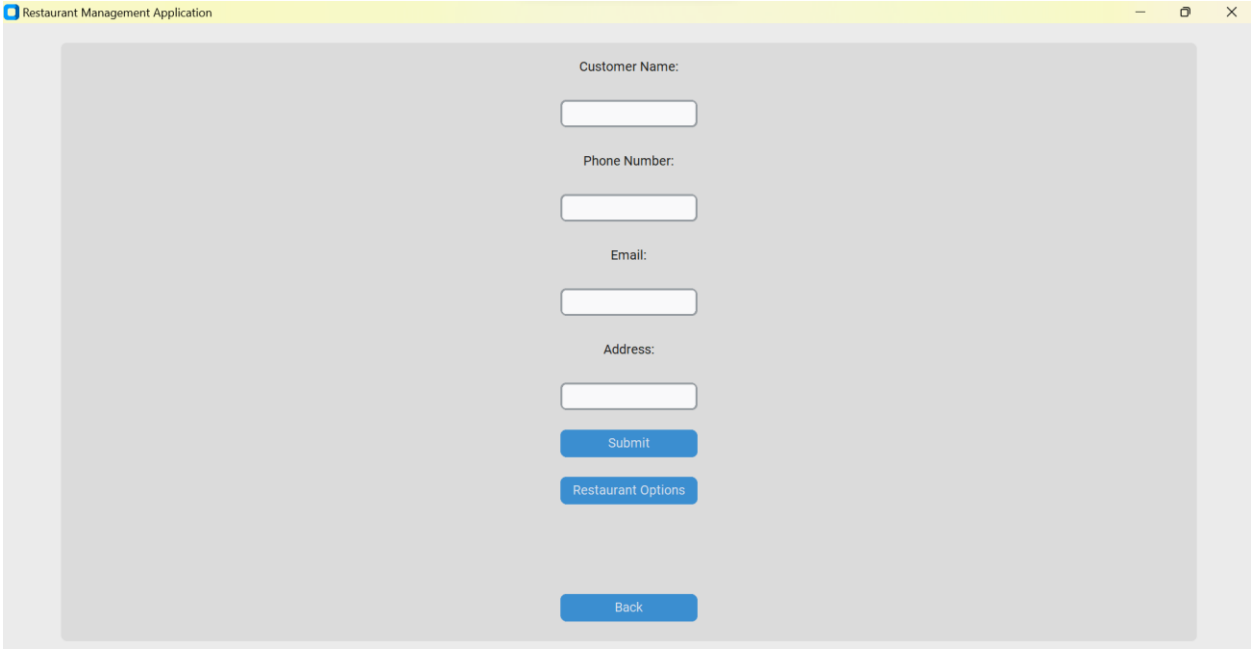
As I mentioned before, here we have implemented more on Customer flow. Clicking on Customer Button, user navigates to Customer Screen where he can enter customer id to verify if he is a new customer or an existing customer.



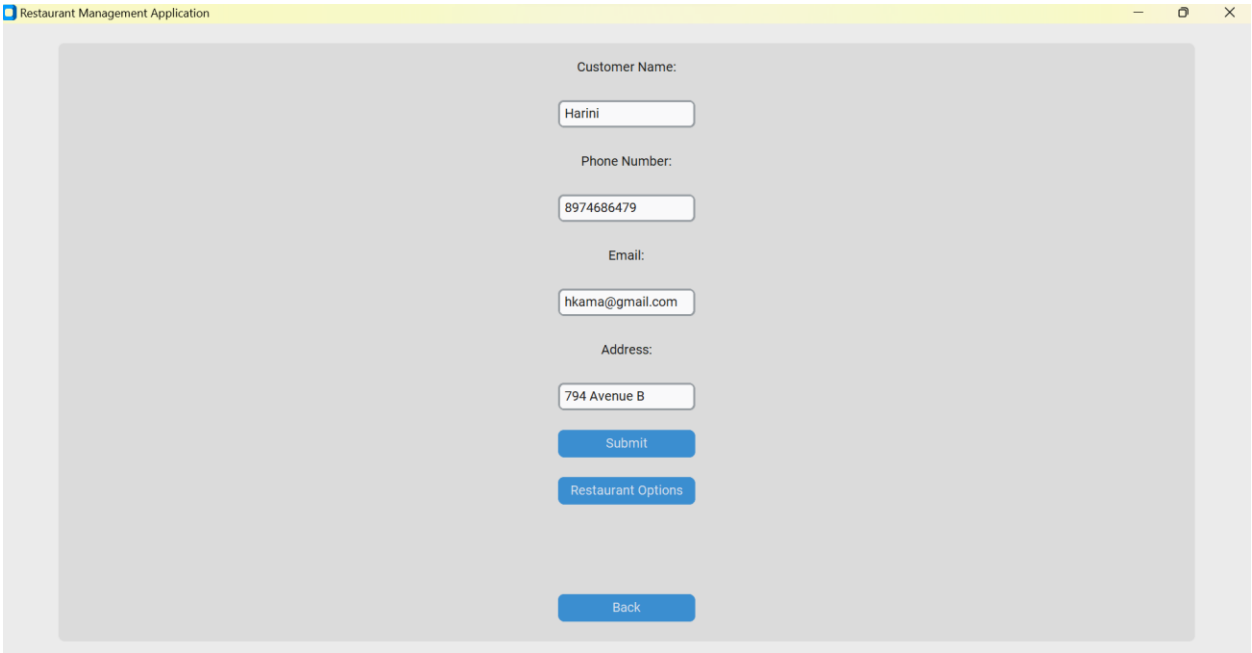
New Customer flow:

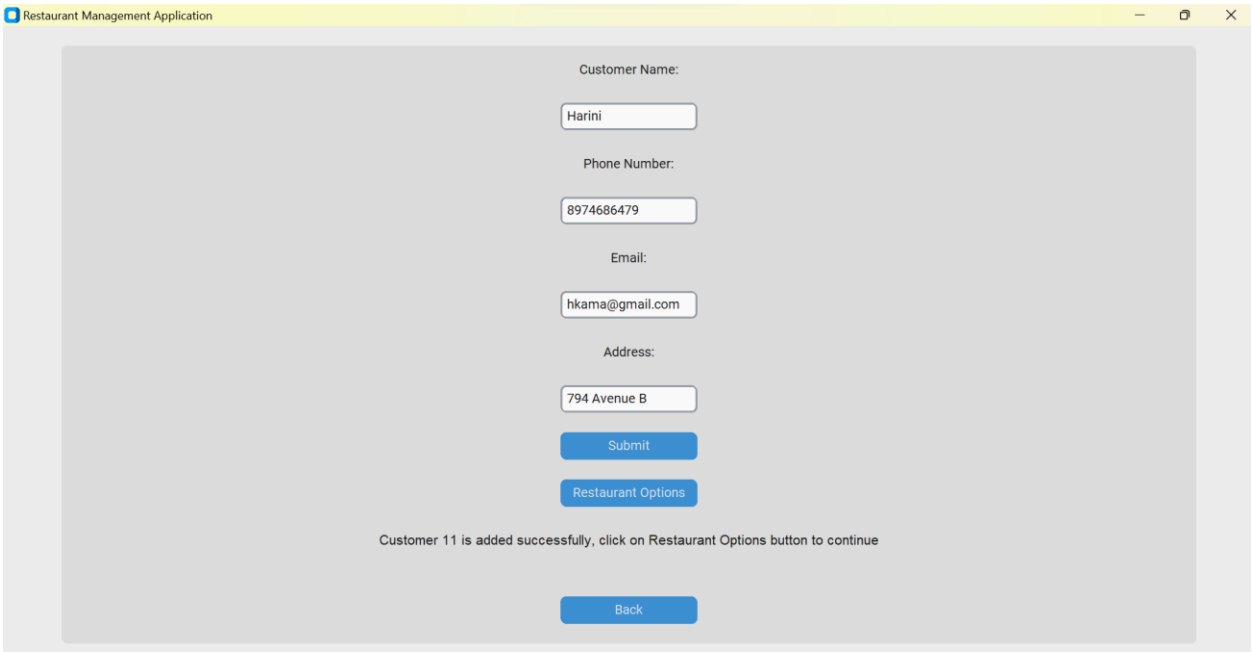
If Customer id entered is new and not available in Customer table data in database, then user is navigated to Customer Details Entry Screen, after clicking on Submit button.



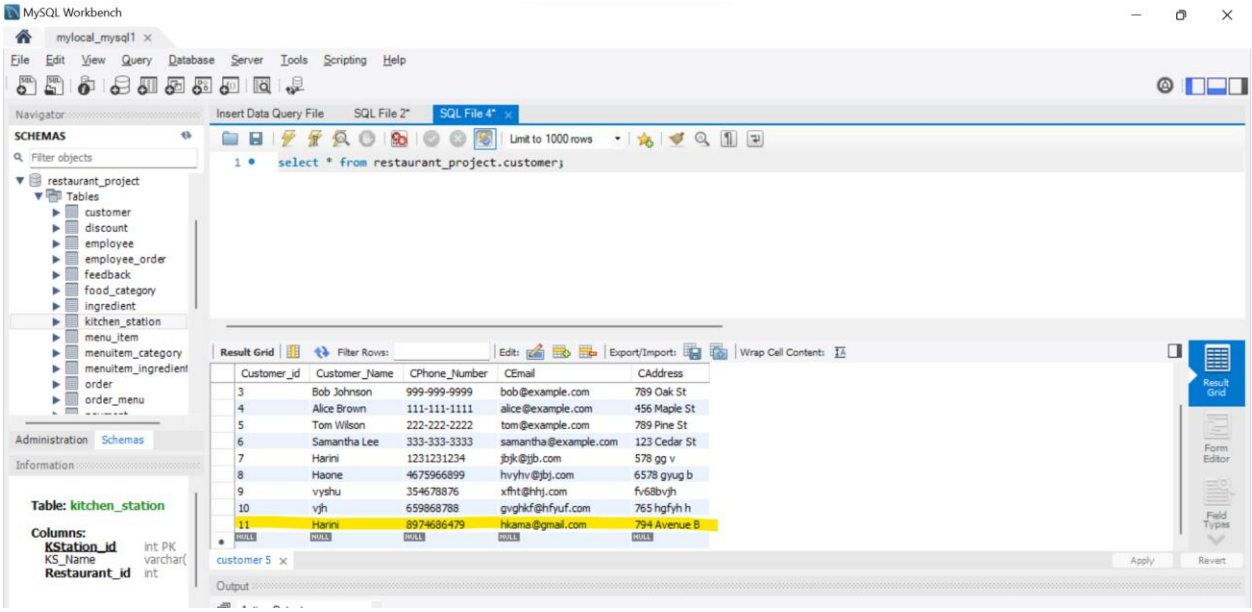


After user enter all customer details and click on Submit button, Customer data saved success message is displayed asking the user to click on “Restaurant Options” to move further. Also, Clicking on Submit button, new customer data is saved, updating the “Customer” table in the database.





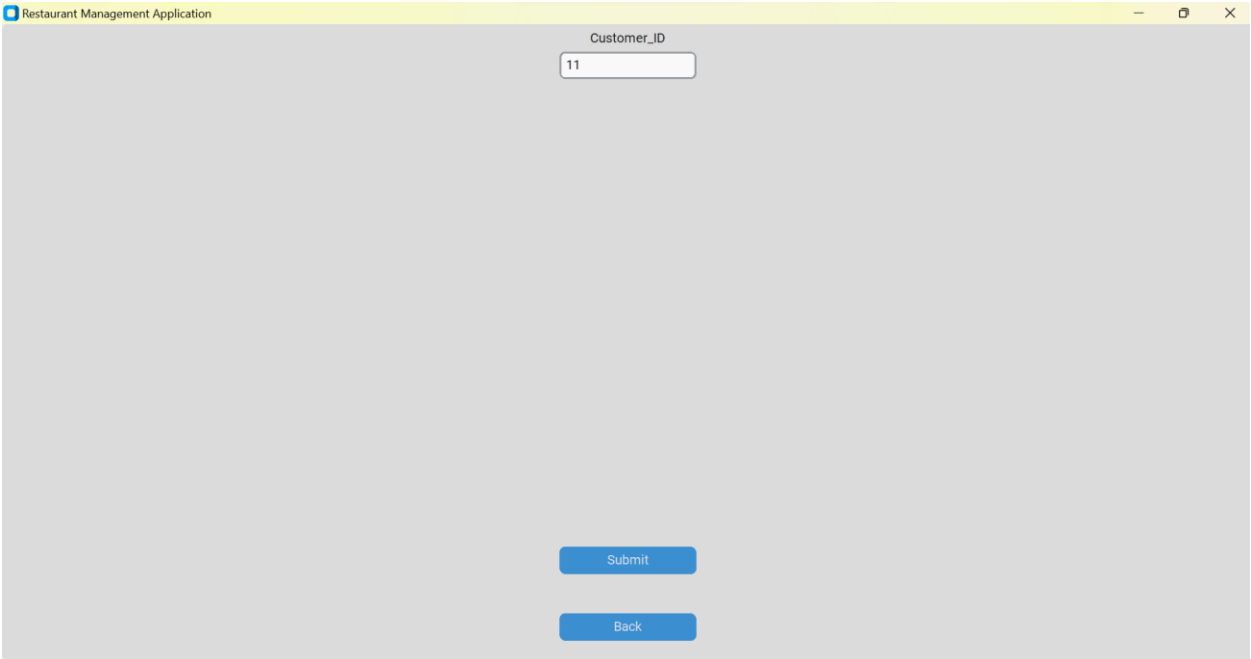
Below is the screenshot of the updated table in the database in MySQL Workbench.



Clicking on “Restaurant Options” button, user navigates to Restaurant Options Screen where he/she can have options to make reservations, order the food, and provide the feedback.

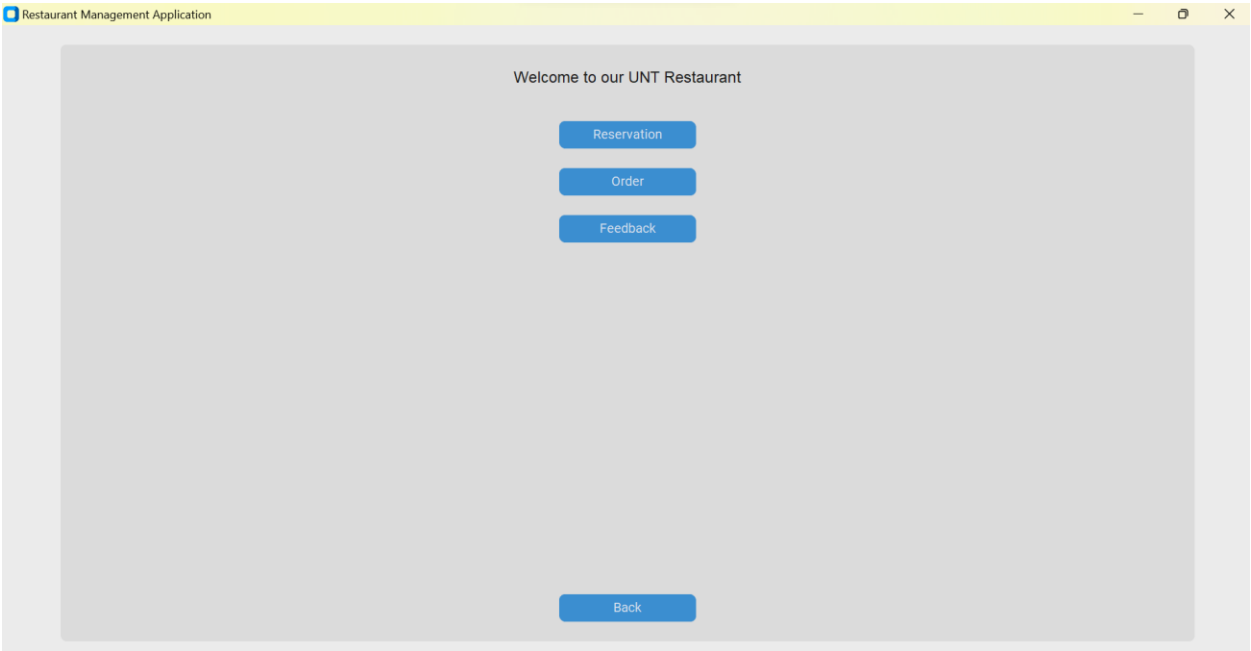
Existing Customer Flow:

On Customer Screen when we enter id of an existing customer, user navigates to Restaurant Options Screen.



Restaurant Options Screen:

Here as said before, we have 3 buttons “Reservation”, “Order”, “Feedback”.



Reservation Flow – Table Zones Screen:

By clicking on "Reservation" button, user is navigated to Table Zones Screen, where we have the options of Zones available in Restaurant to choose Tables.

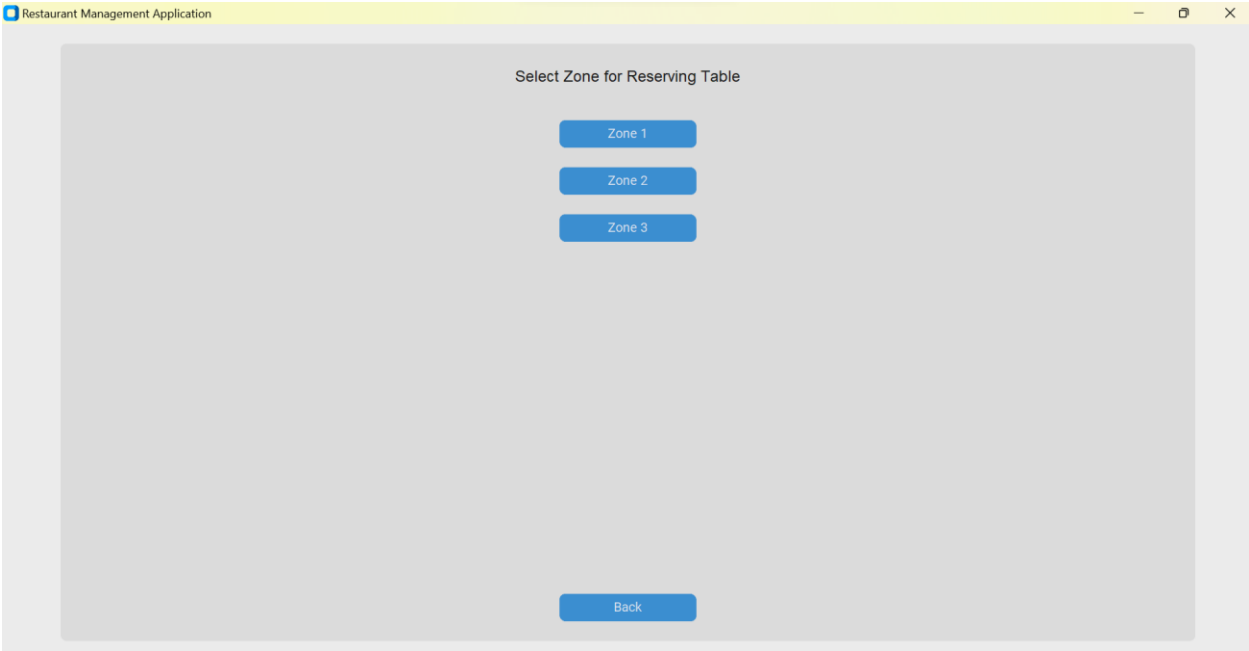
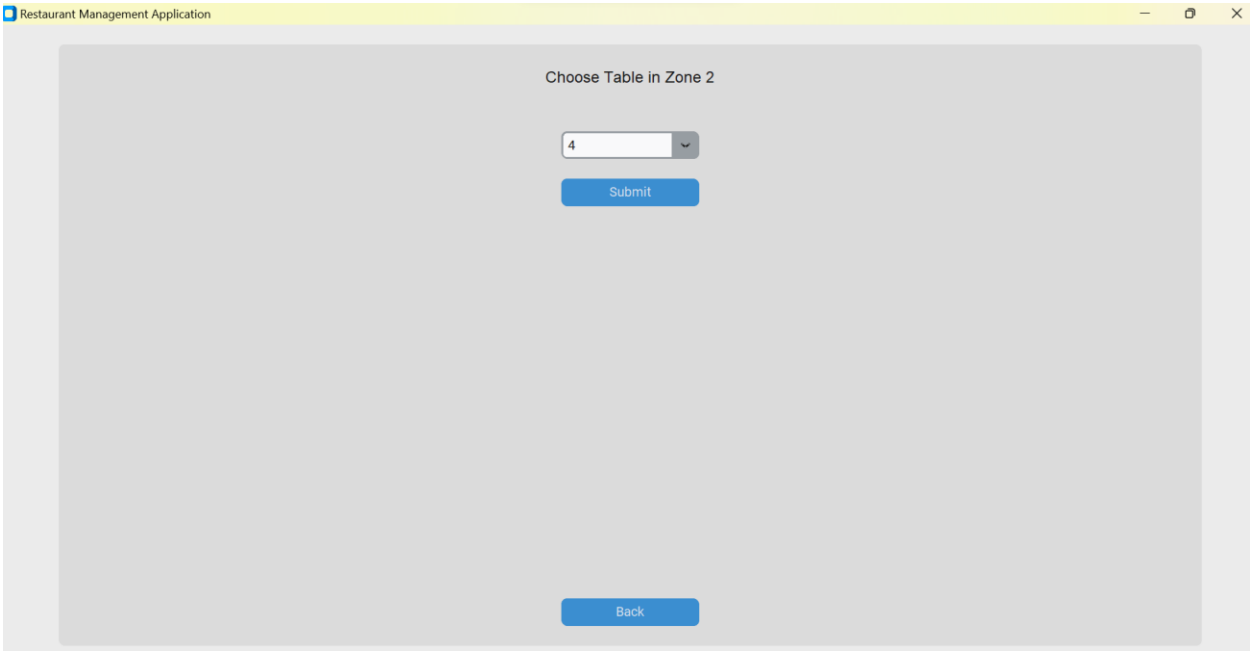


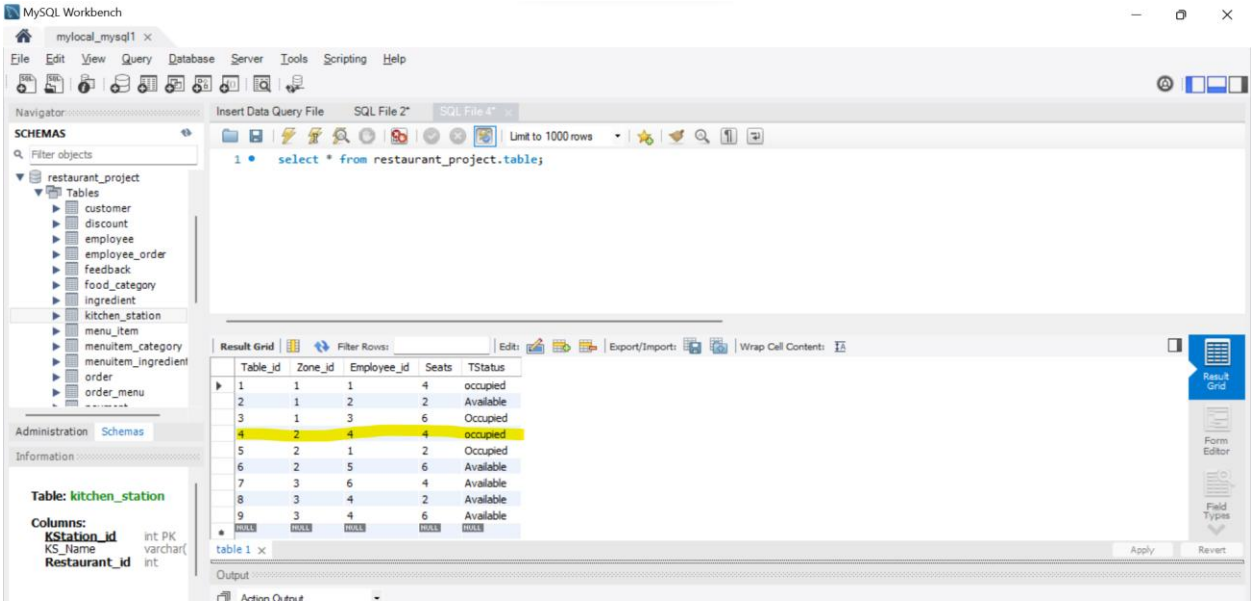
Table Selection Screen:

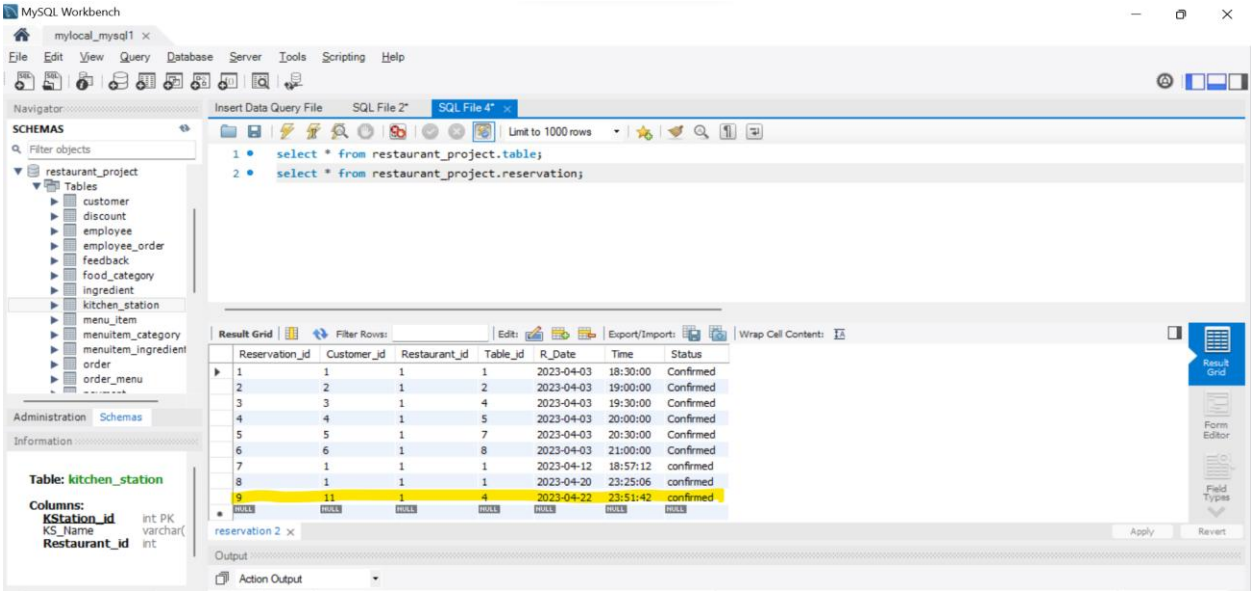
Clicking on one of the Zone buttons, user is navigated to Table Selection Screen, where it displays the dropdown box with the list of available tables (dynamic values) to reserve.



Clicking on Submit button after choosing a table, user is navigated back to Reservation Options Screen, updating the “Table” & “Reservation” tables in the database.

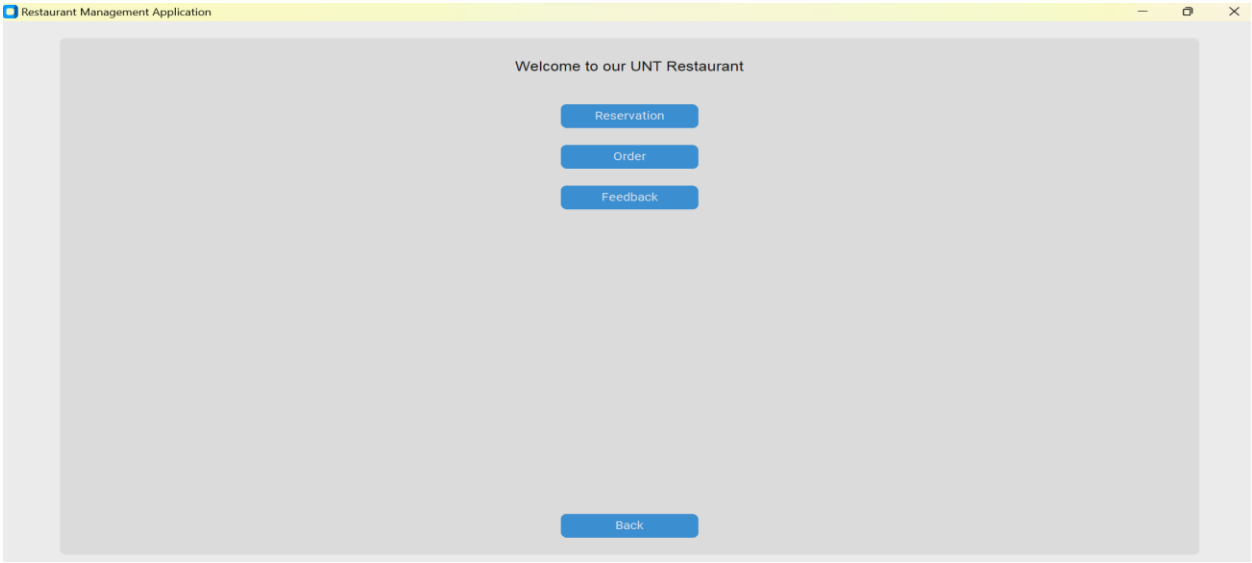
Below are the screenshots of the updated tables in the database in MySQL Workbench.

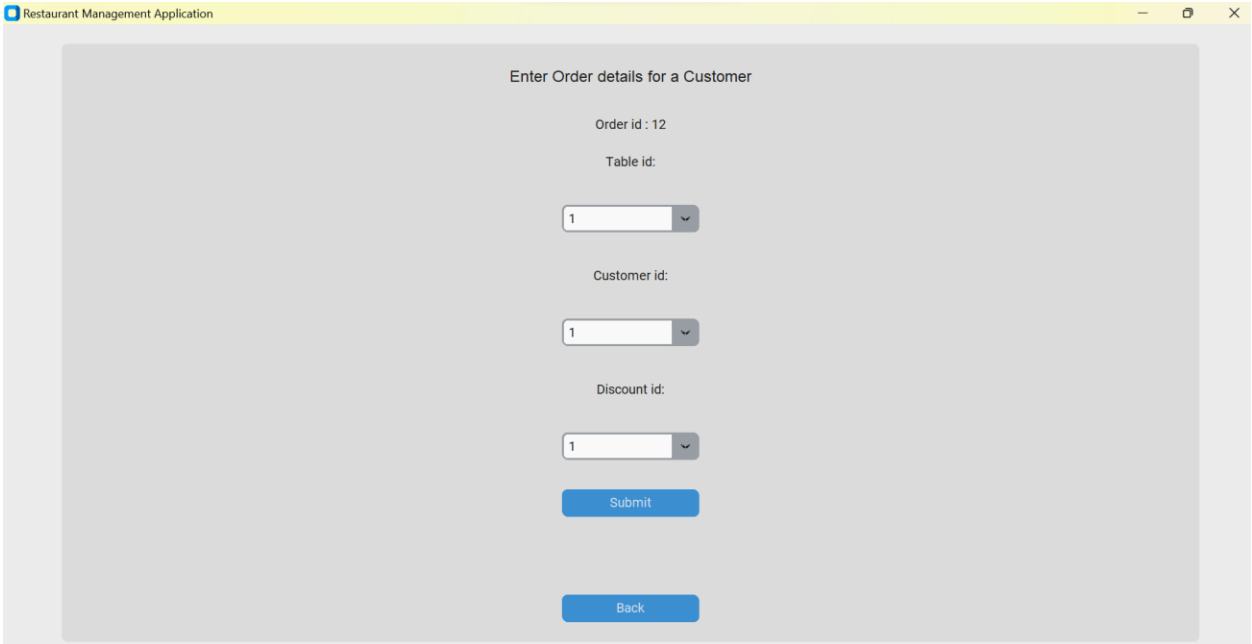


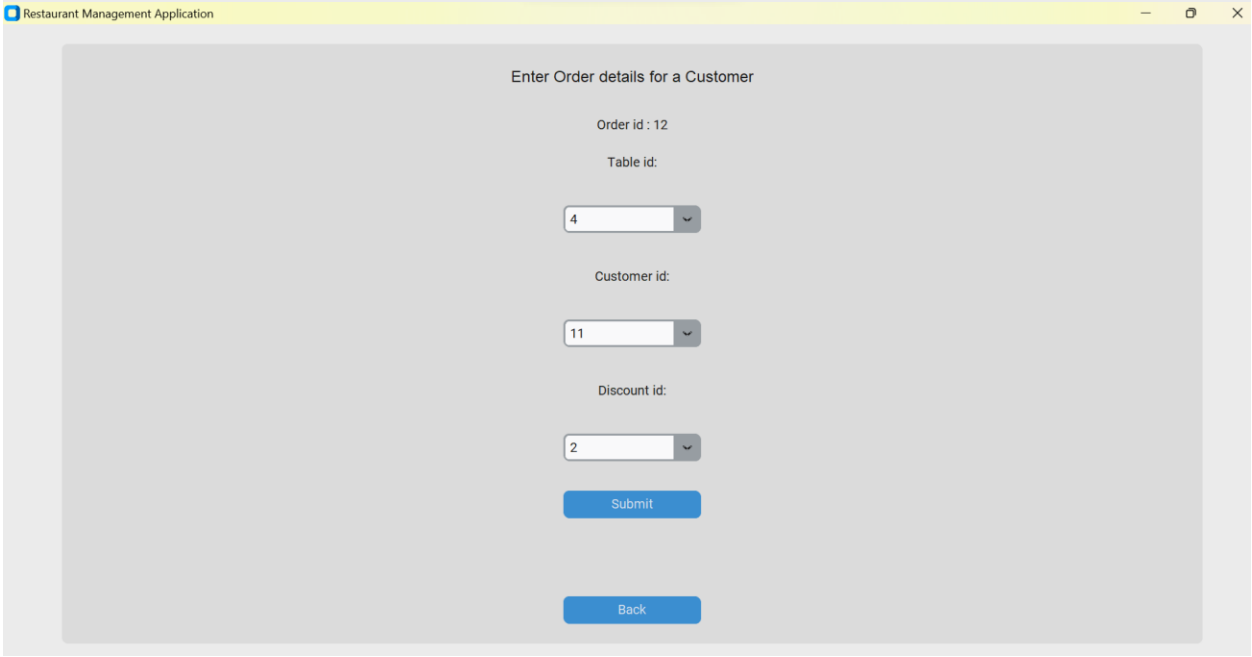


Order Flow – Order Entry Details Screen:

On Clicking on "Order" button, user is navigated to Order Details Entry Screen, where Order id is displayed dynamically, and is displayed with dropdown boxes (dynamic values from database tables) for selecting details of Table id, Customer id, Discount id, and a Submit button.

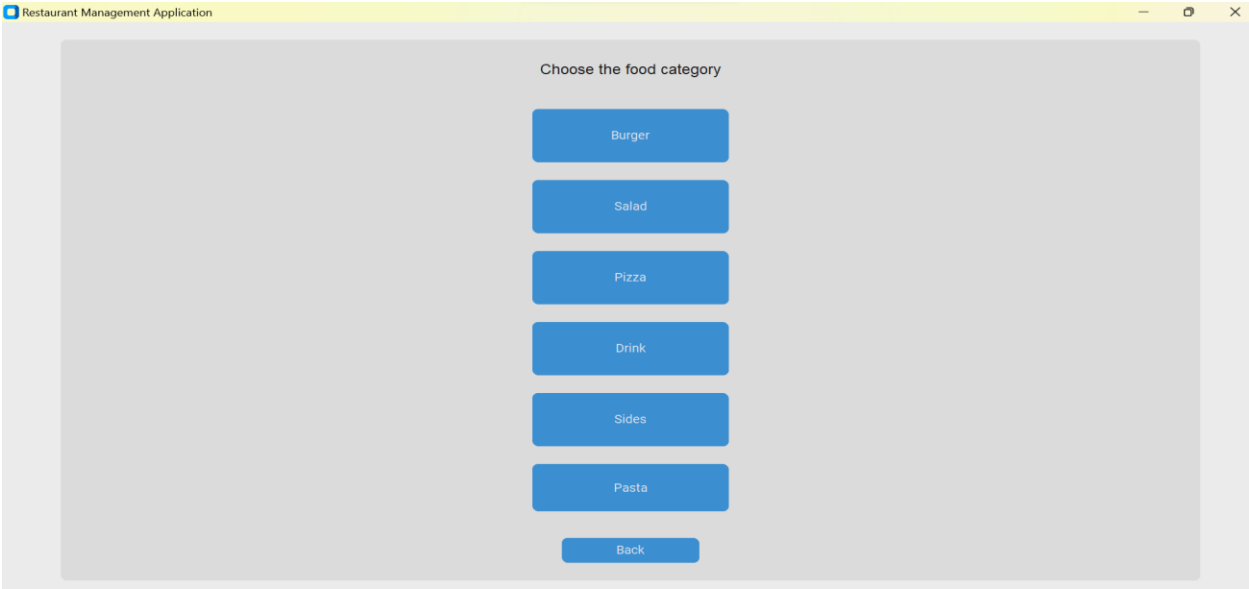






Clicking on Submit button navigates to Food Category Options Screen.

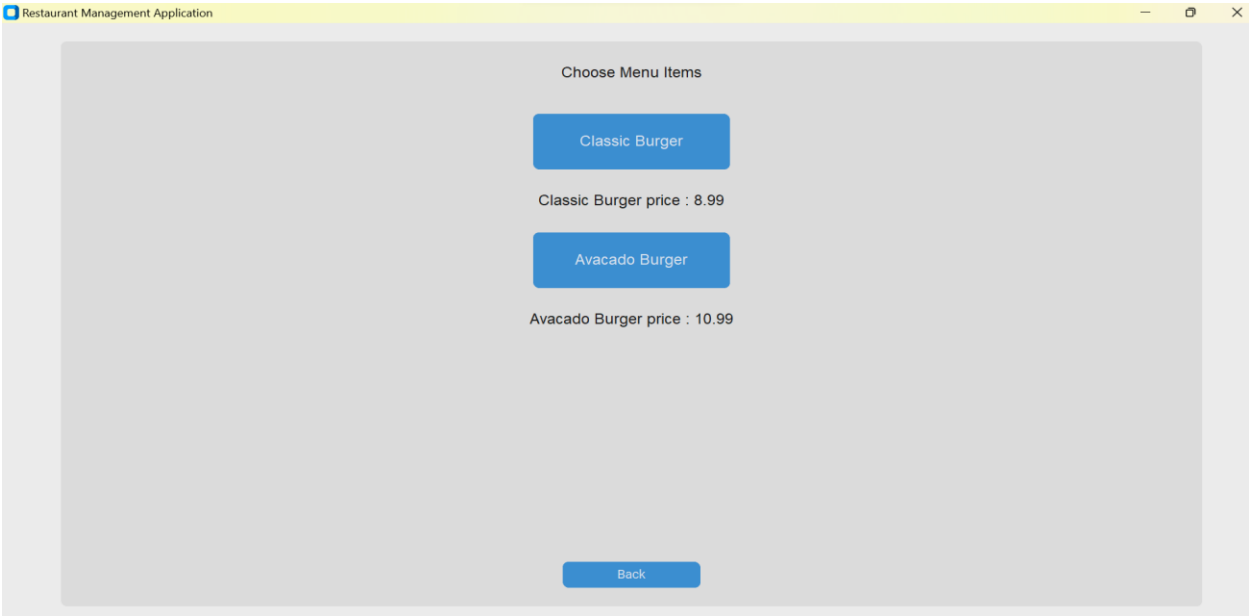
Food Category Options Screen:
On this screen user is displayed with all the available food categories (dynamic, capturing from backend database table) as buttons.



On clicking on any of the food category buttons, user is navigated to related menu items screen to choose. Here for example, I'm clicking on "Burger" button.

Menu Items Selection Screen:

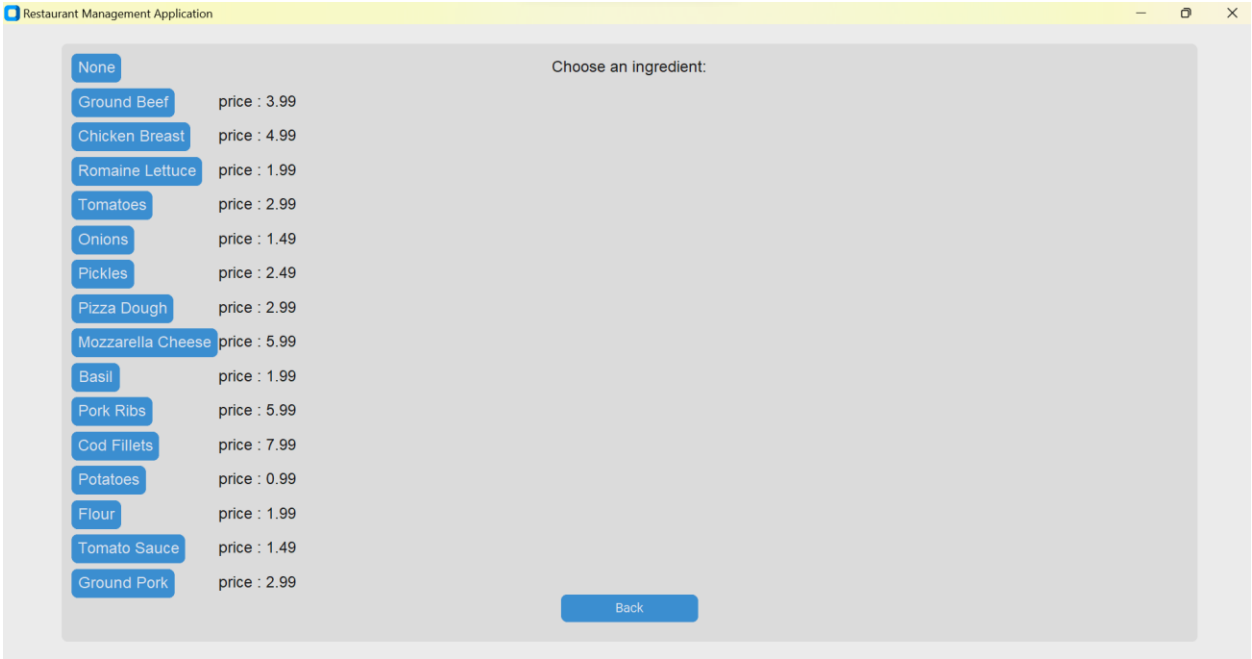
On this screen, the user is displayed with all the available menu items buttons (dynamic, capturing from backend database table) related to the selected food category as buttons, along with their prices. (Example, Burger category menu items screen)



On selecting one item and clicking on it, navigates the user to Ingredients Screen to choose for the selected menu item. Here for example, I'm clicking on "Classic Burger" of price $8.99.

Ingredients Selection Screen:

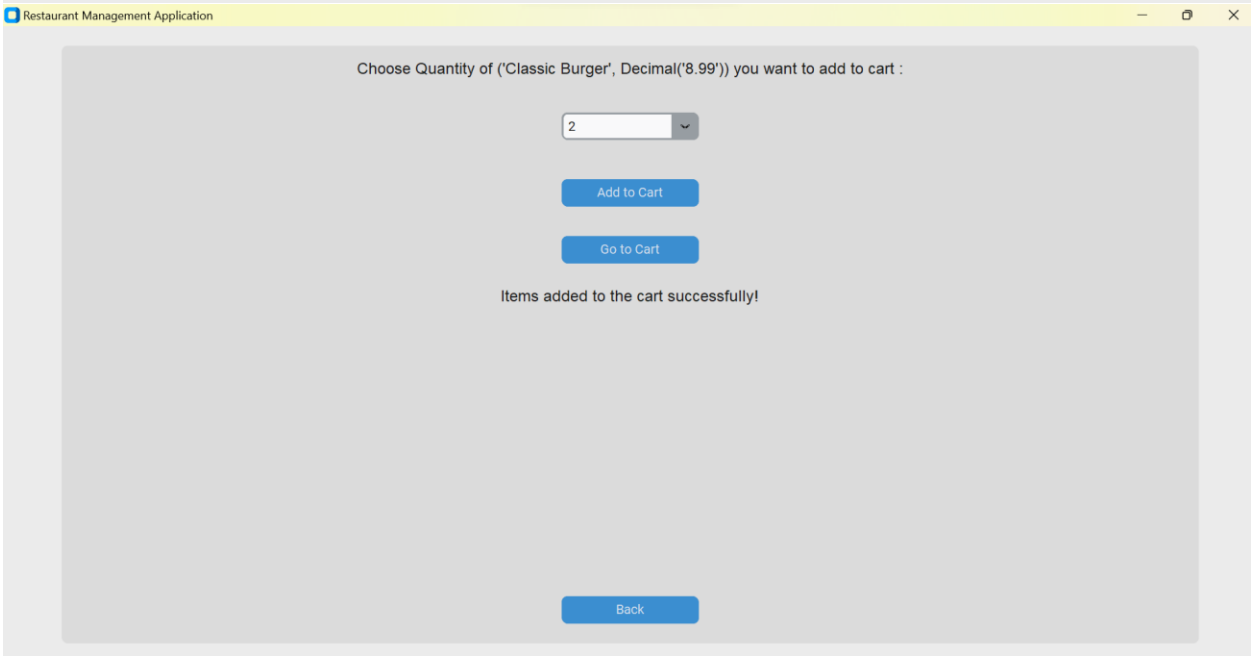
On this screen, the user is displayed with all the available ingredients options buttons (dynamic, capturing from backend database table) to add on the selected menu item, along with the price of each ingredient. (None option button is also displayed, so that if user doesn't want to add on any ingredient, he can choose None as the option.



On selecting one ingredient and clicking on it, navigates the user to Quantity selection Screen. Here for example, I'm clicking on "Chicken Breast" of price $4.99.

Quantity Selection Screen:

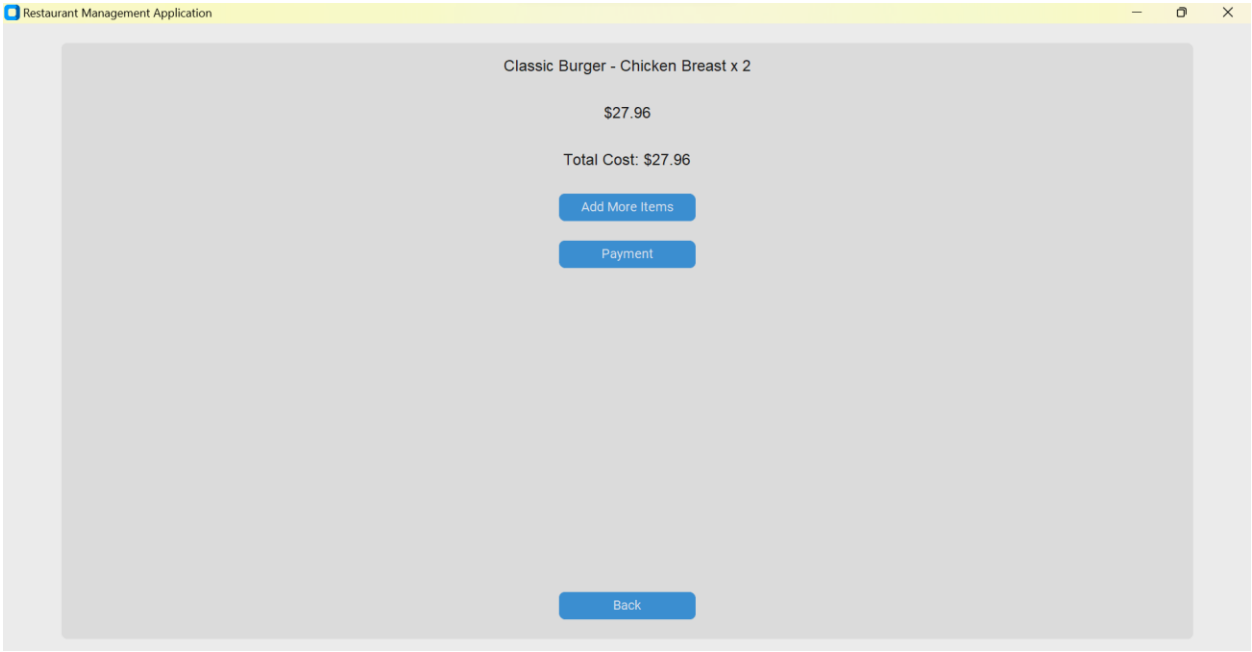
On this screen, we have an option to choose the number of items selected, from the dropdown list (with 1-10 options) to add to cart. After choosing the quantity, click on "Add to cart" button, which will add the items to cart list and display the same message on the screen confirming the user. After then, clicking on "Go to cart" button, user navigates to Cart Screen. (For example, I have chosen 2 as quantity)



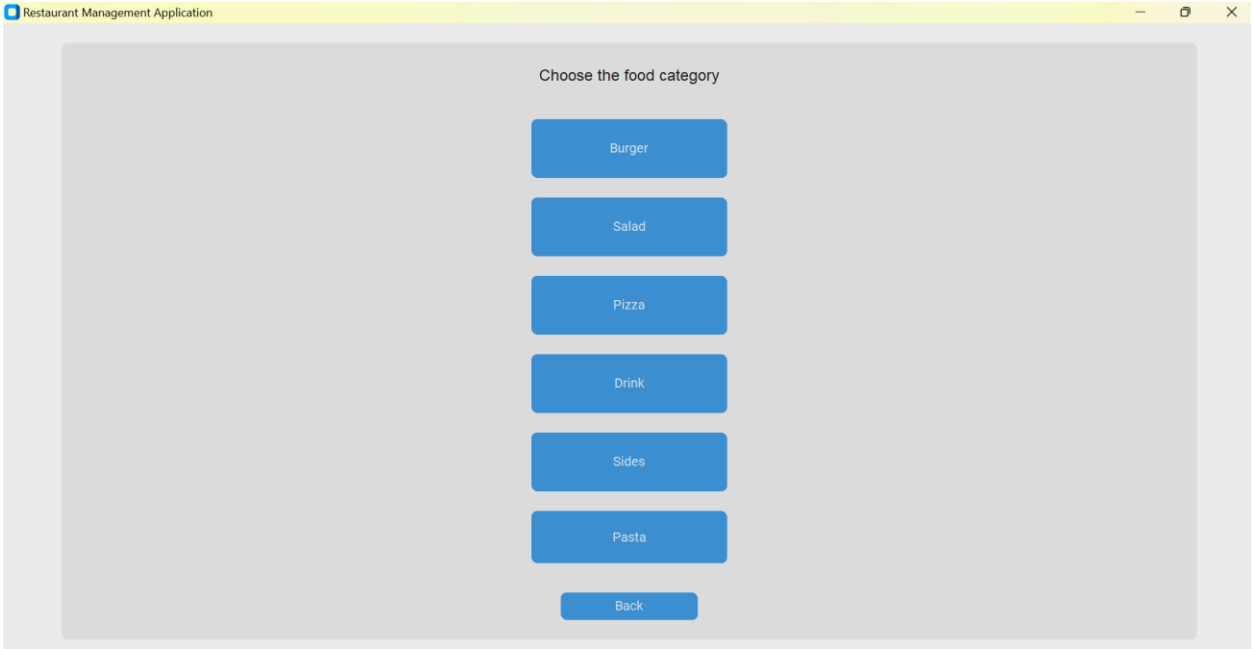
Cart Screen:

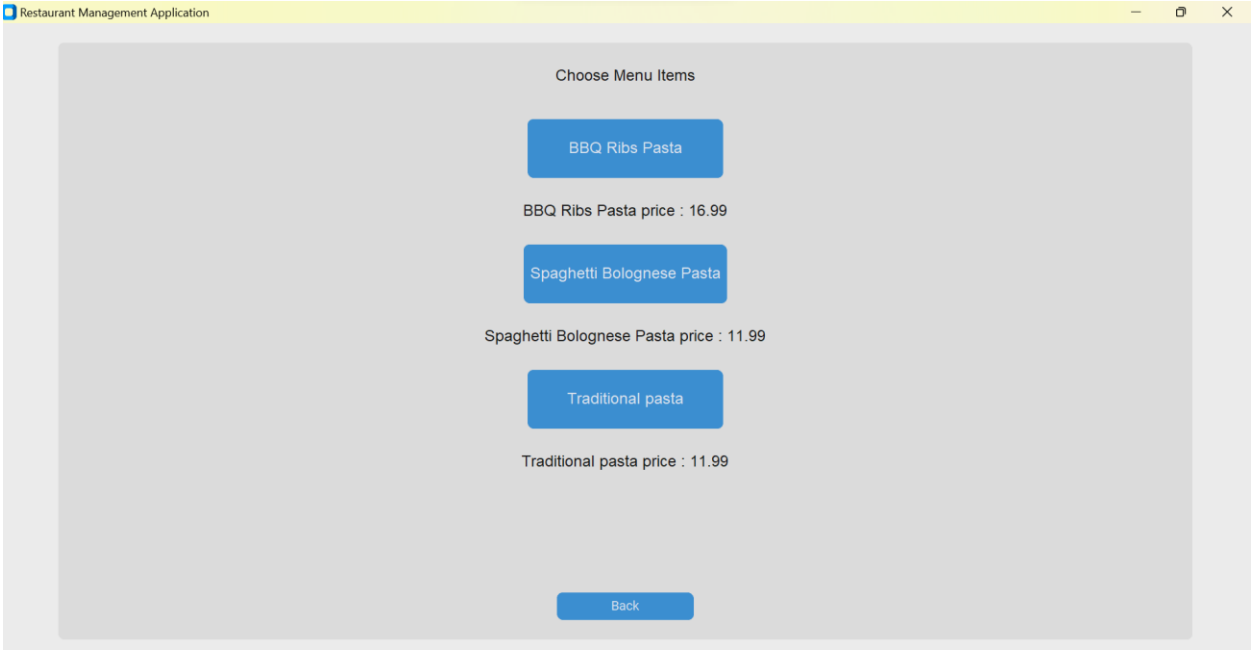
Cart Screen displays the list of items added to cart, along with their ingredients, quantity, and their total price. It also provides the option to add more items with the button "Add More Items", on clicking on it, navigates back to Food Category Screen. If the user is done with the selection of items into cart, it also provides button for making "Payment".

(Total price here is – (8.99+4.99)*2 = $27.96)

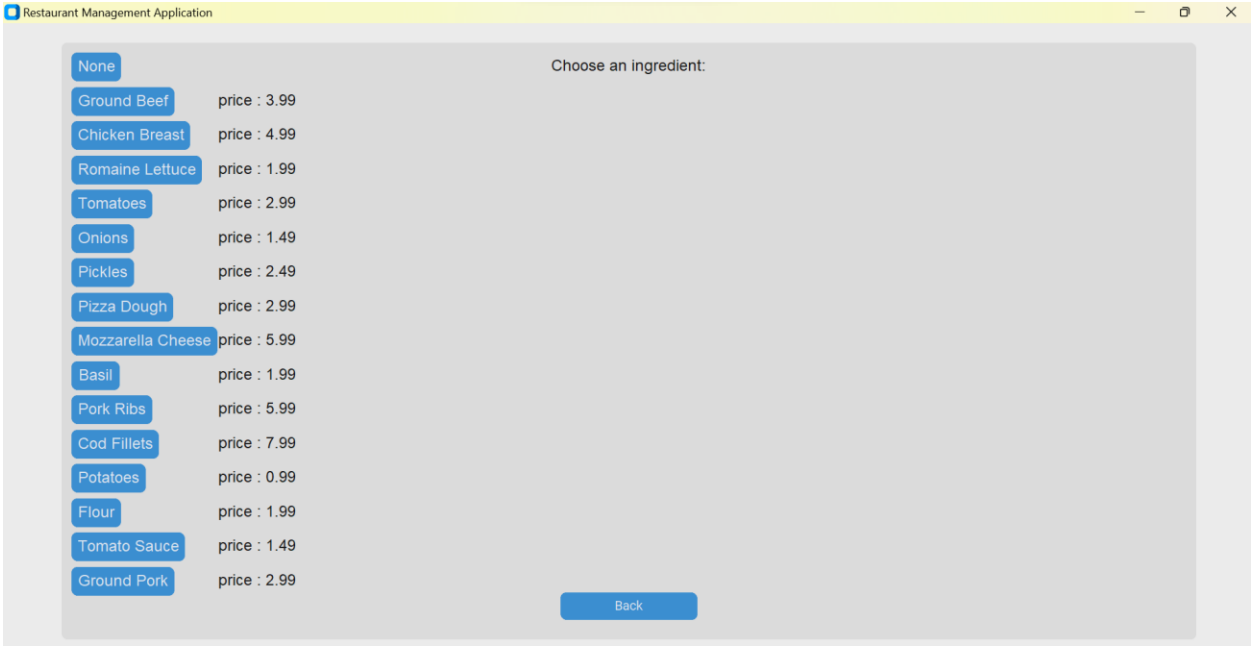


Let’s go with the add more items flow and add one more item.

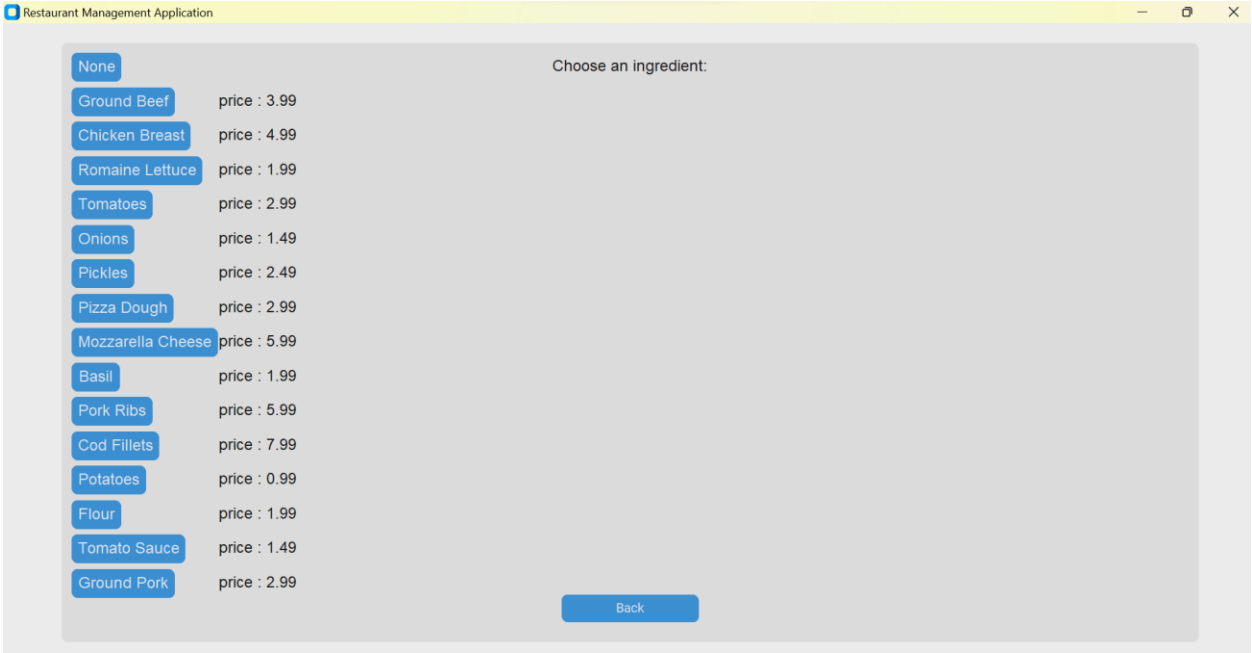


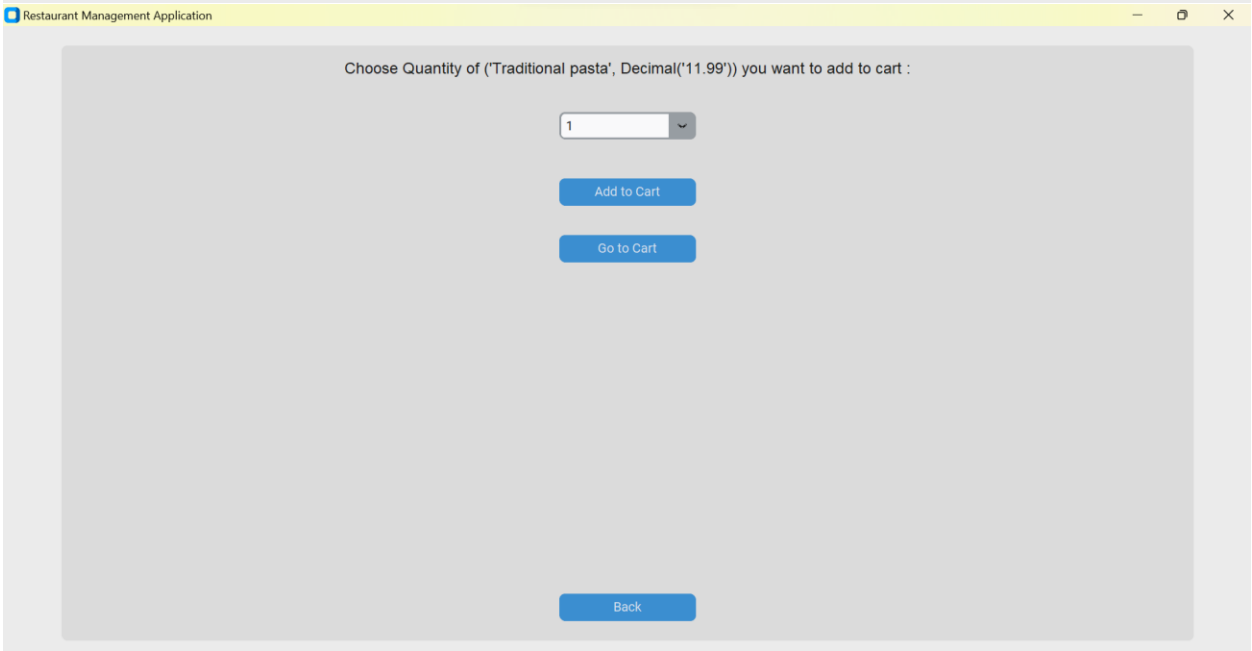


Selecting “Traditional Pasta” of price $11.99



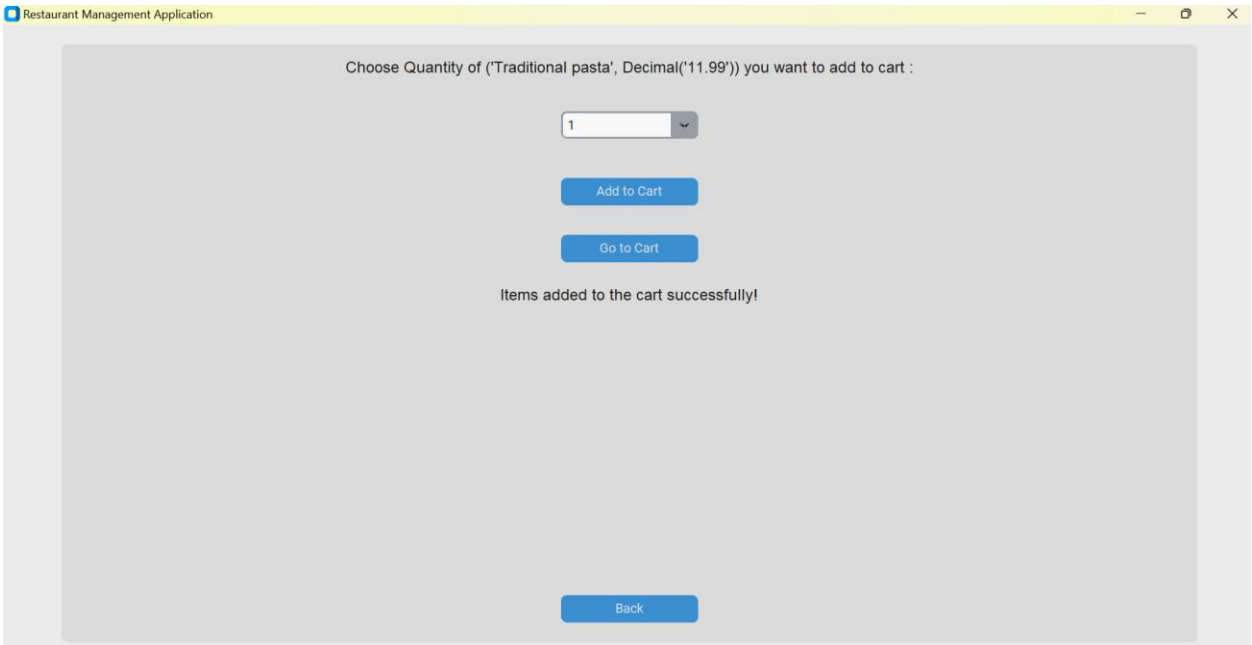
Selecting “None” as ingredient

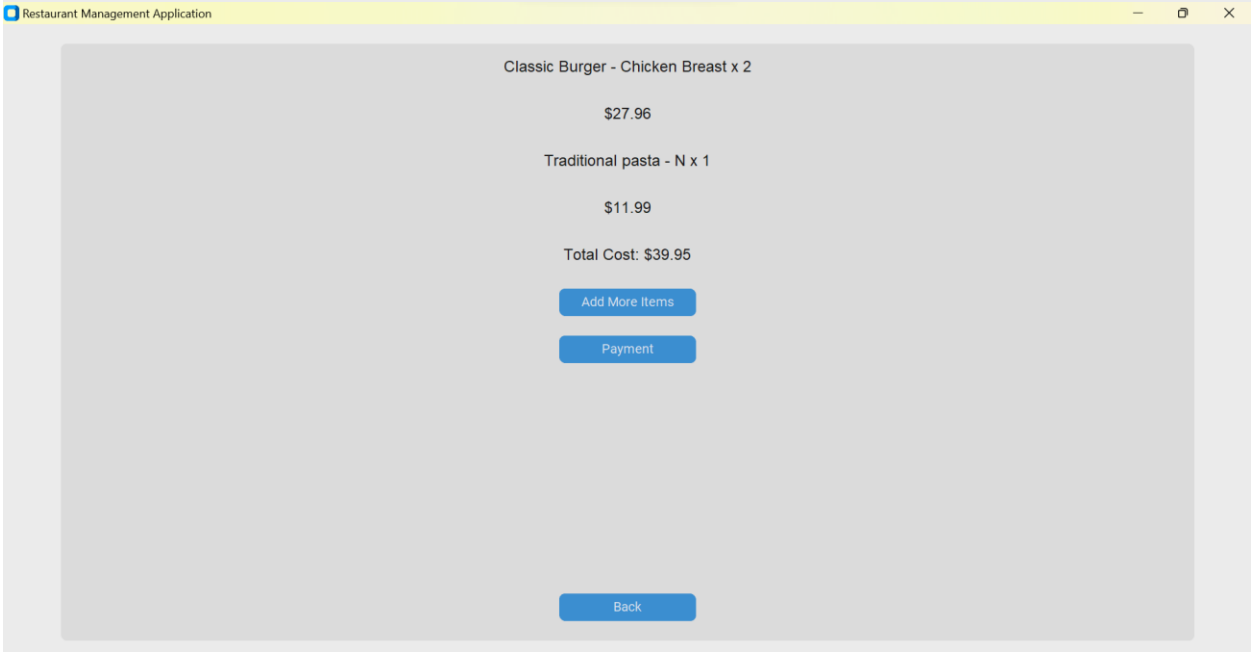




Selecting quantity as 1

And add it to Cart





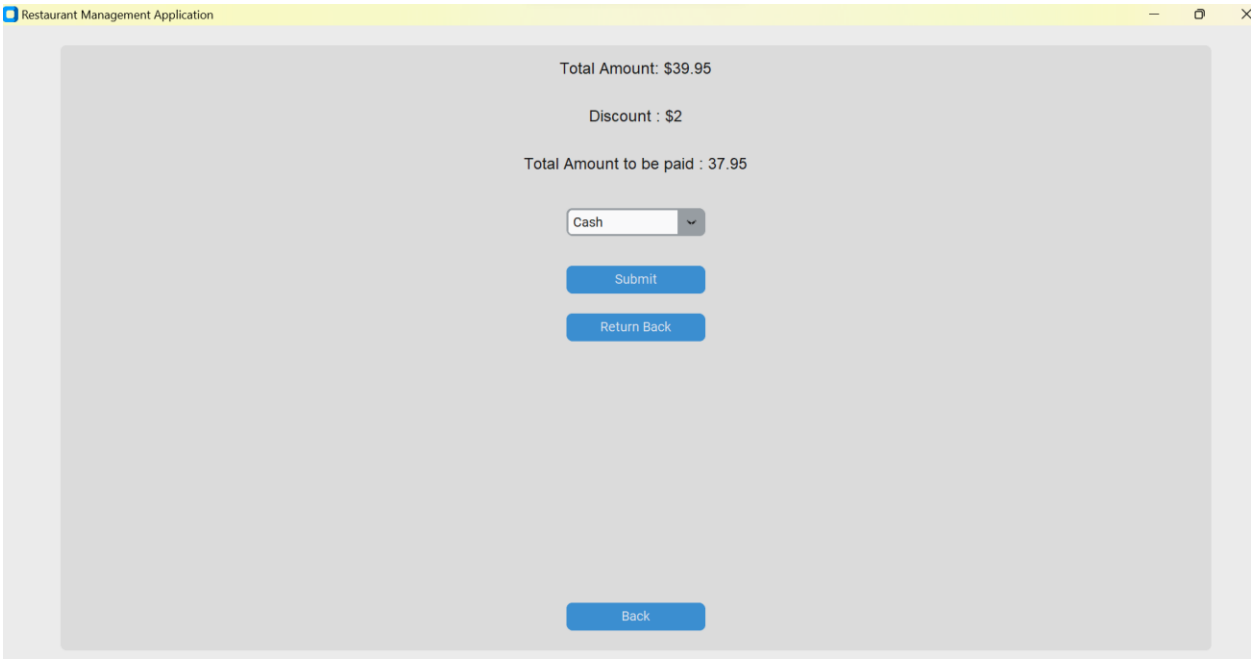
As we have selected None as ingredient and quantity as 1, item cost is $11.99, and this price is added to the previous items cost and is displayed as Total Cost (Here it is $27.96 + $11.99 = $39.95)

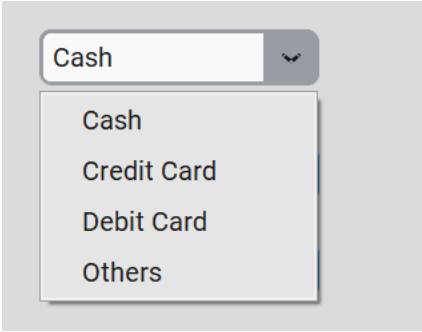
After clicking on “Payment” button it navigates to Payment Screen.

Payment Screen:

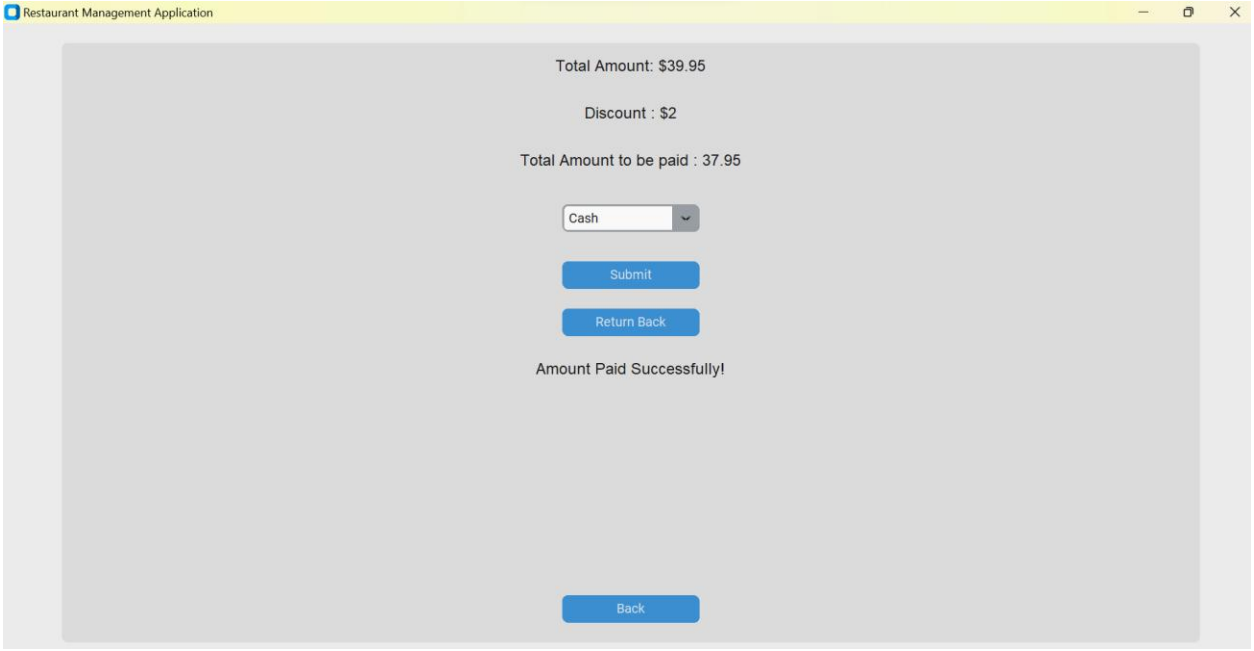
On Payment Screen, total amount of items is displayed and the discount which we have chosen on Order Details Entry Screen is also displayed. Considering these 2 values, user is displayed with the total amount to be paid. (Total amount of items – Discount = $39.95 - $2.00 = 37.95)

Also, a dropdown box with the list of payment options (Cash, Credit Card, Debit Card, Others) is displayed to choose the mode of payment.

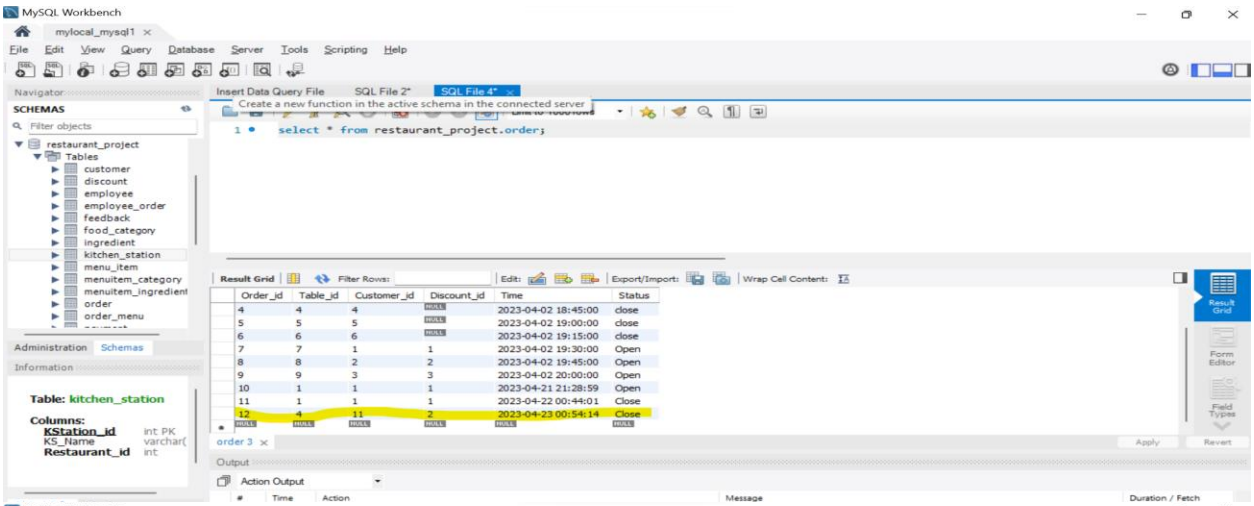


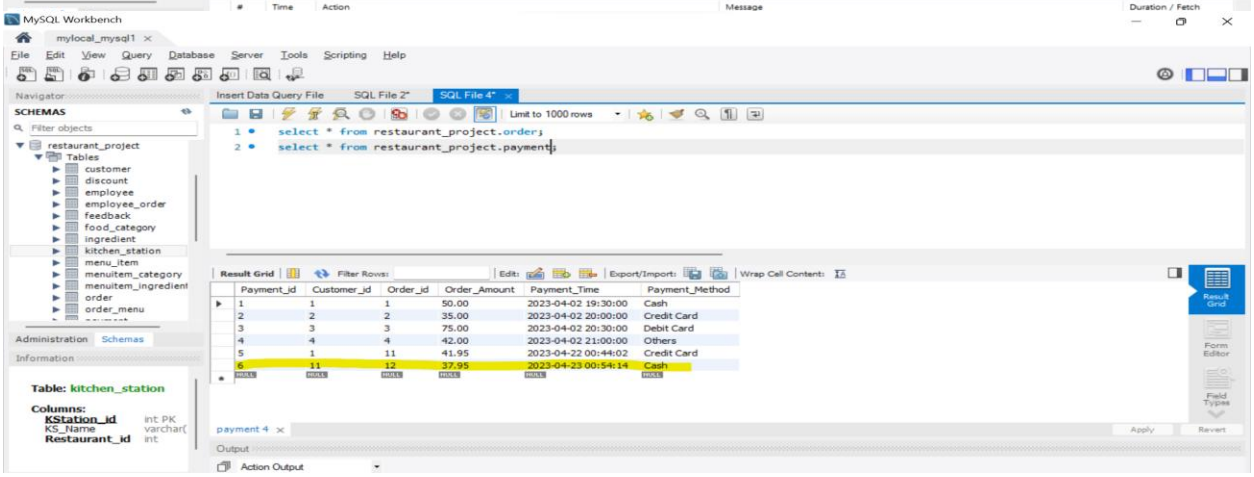


After clicking on Submit button, user is displayed with the message as “Amount Paid Successfully!” and updates the “Order” and “Payment” table in the database.

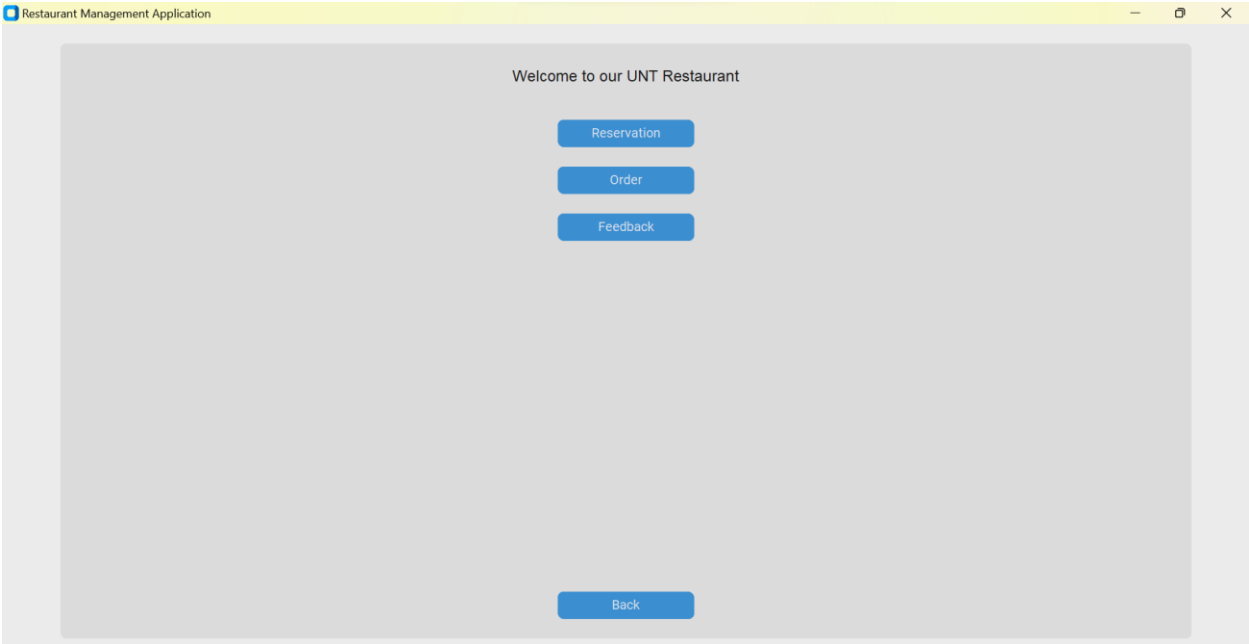


Below are the screenshots of the updated tables in the database in MySQL Workbench.



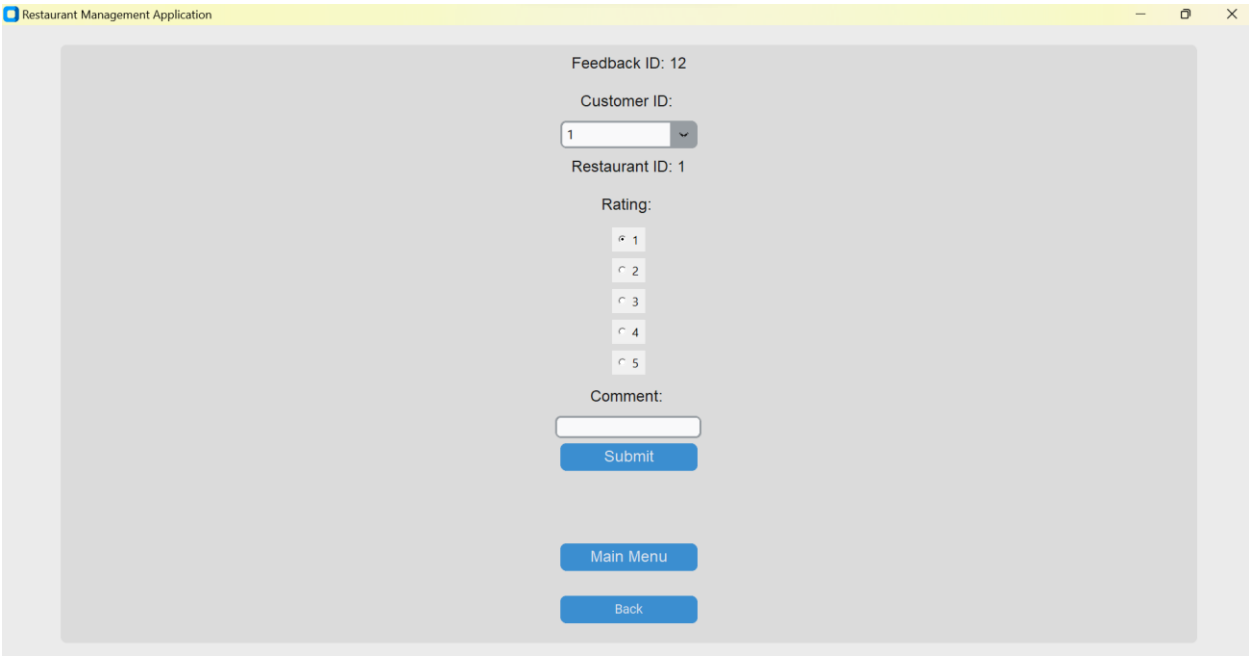


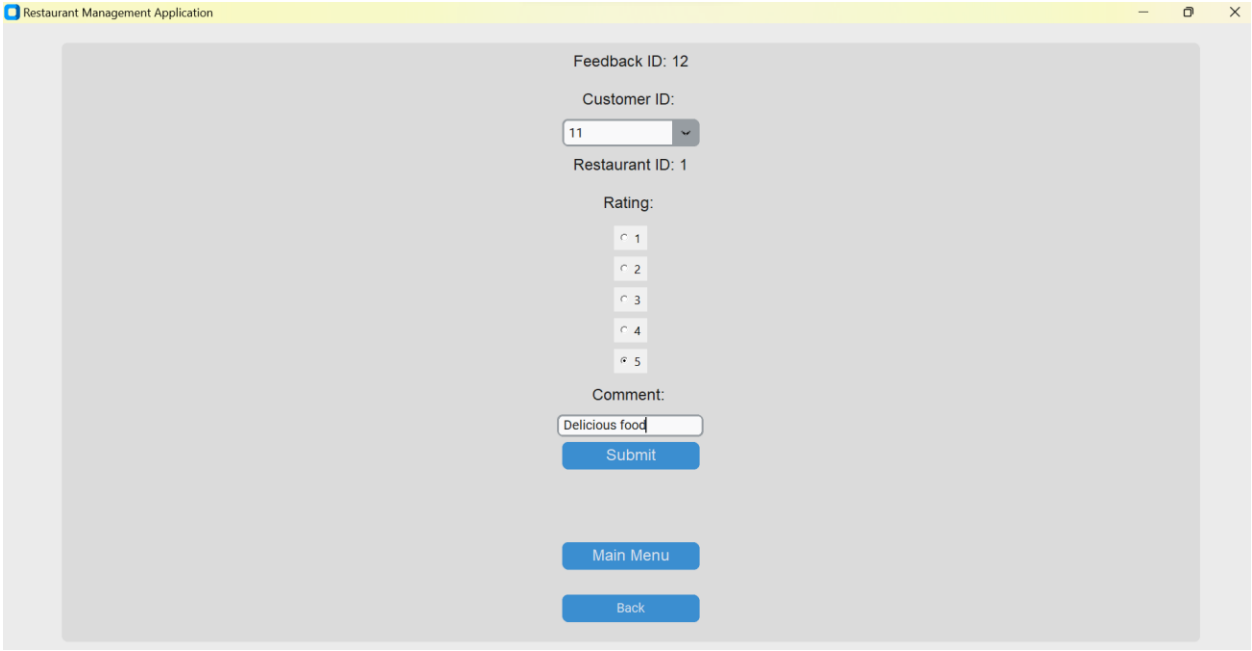
Clicking on “Return Back” button, user is navigated to “Restaurant Options” Screen



Feedback Screen:

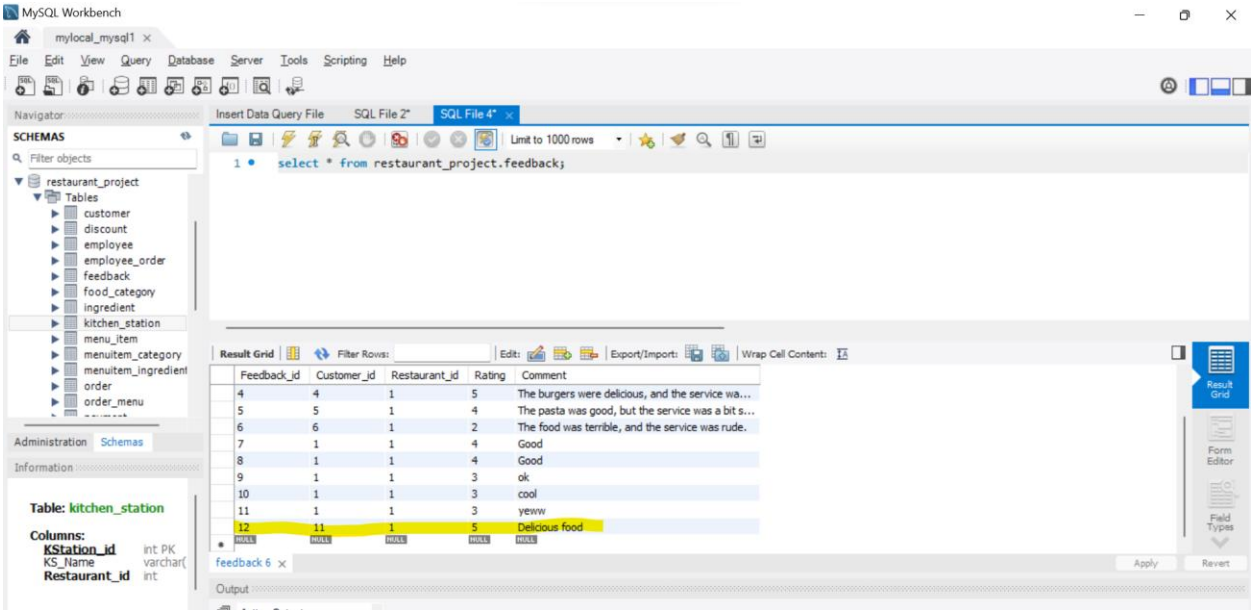
Click on “Feedback” button on Restaurant Options Screen, navigates user to “Feedback Screen”. Feedback Screen is displayed with dynamic Feedback id, Customer id dropdown box to choose with the list of available customers, Restaurant ID, Rating Radio Buttons to choose (range from 1-5), Comment entry box.





On clicking on Submit button, user is displayed with the message as “Feedback stored successfully!”, also updating the Feedback table in the database.

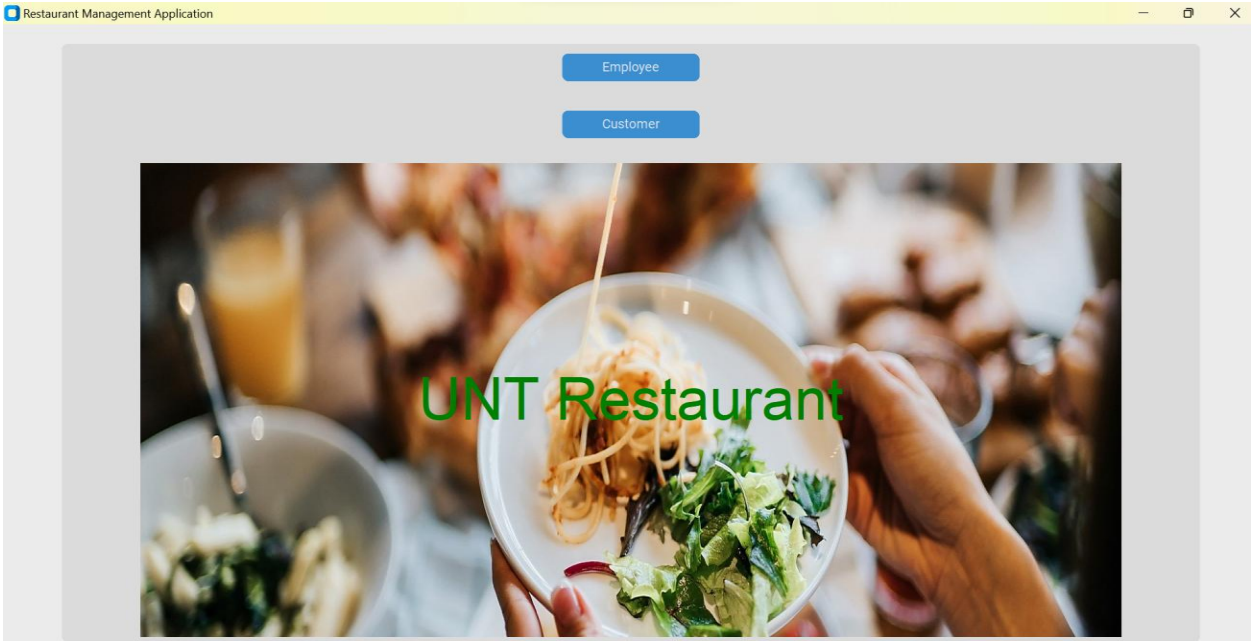
Below is the screenshot of the updated table in the database in MySQL Workbench.

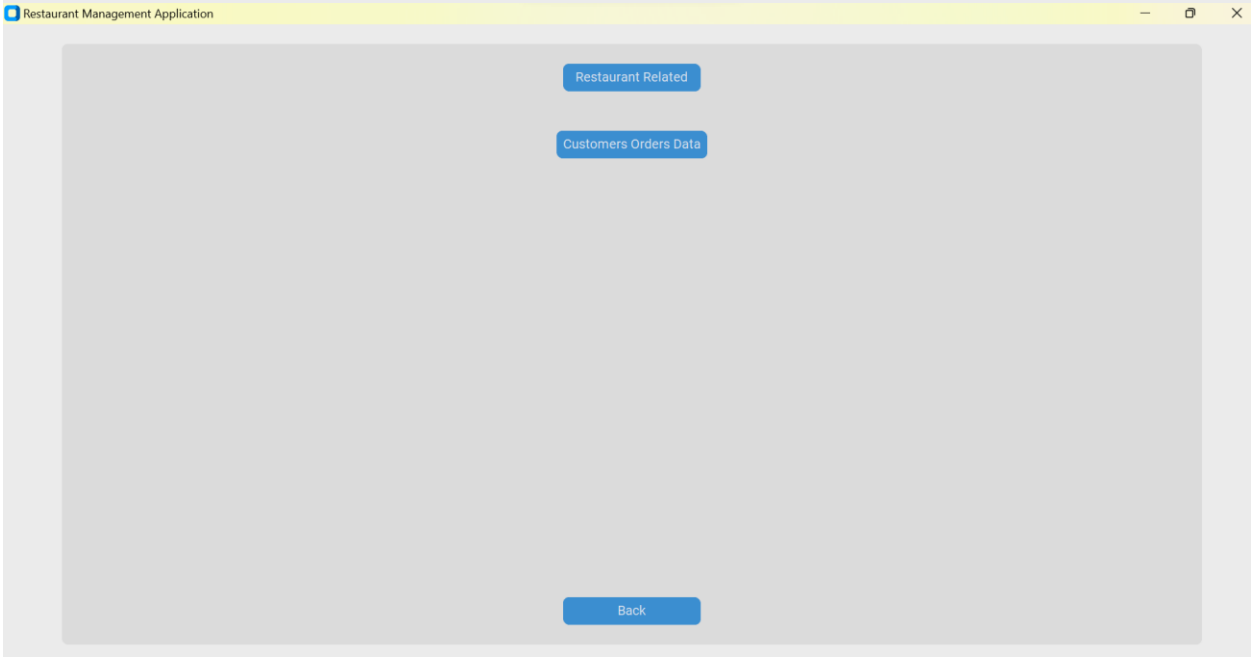


Clicking on “Main Menu” button, user is navigated back to Restaurant Options Screen.

Employee Flow Screens:

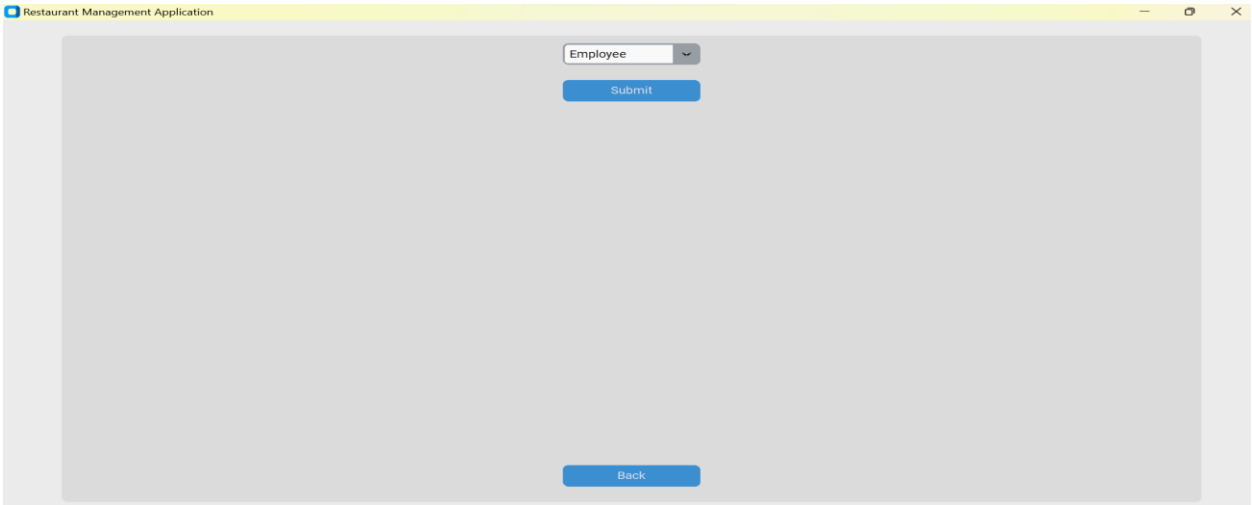
As discussed before, because of critical time constraints and tasks, in this project we have concentrated much on Customer flows than Employee flows. In Employee flow, we have just implemented the screens to display the "Restaurant Related" and "Customers Orders Data" tables.

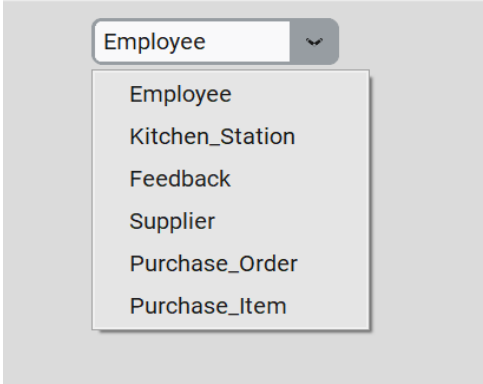


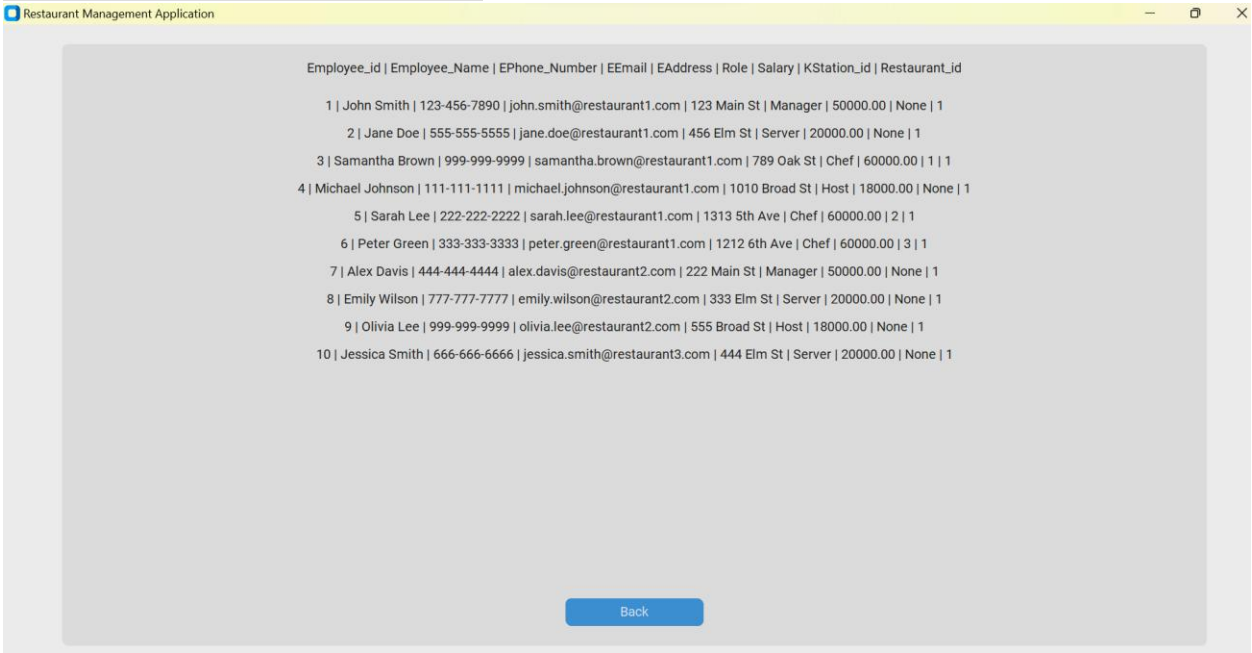


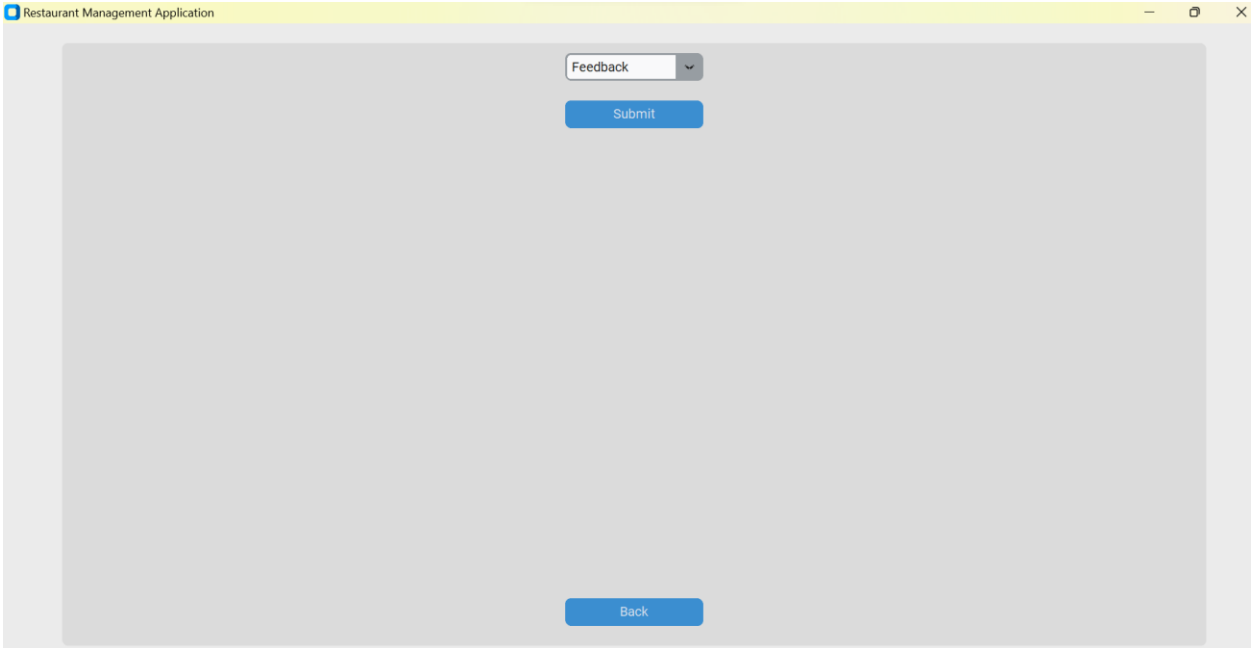
Restaurant Related Screen:

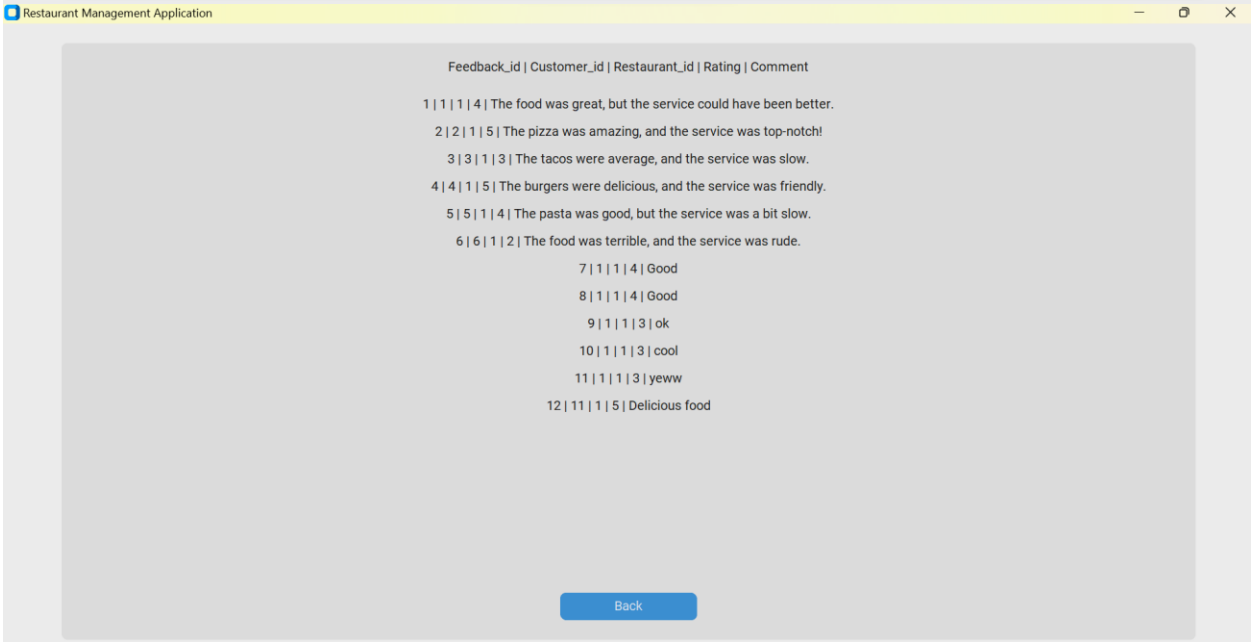
Displays the dropdown box with the list of tables. Here for example, I'm choosing "Employee" and "Feedback" tables to display the data to the user.


Restaurant Management Application
Employee
Submit
Back


Employee
Employee
Kitchen_Station
Feedback
Supplier
Purchase_Order
Purchase_Item

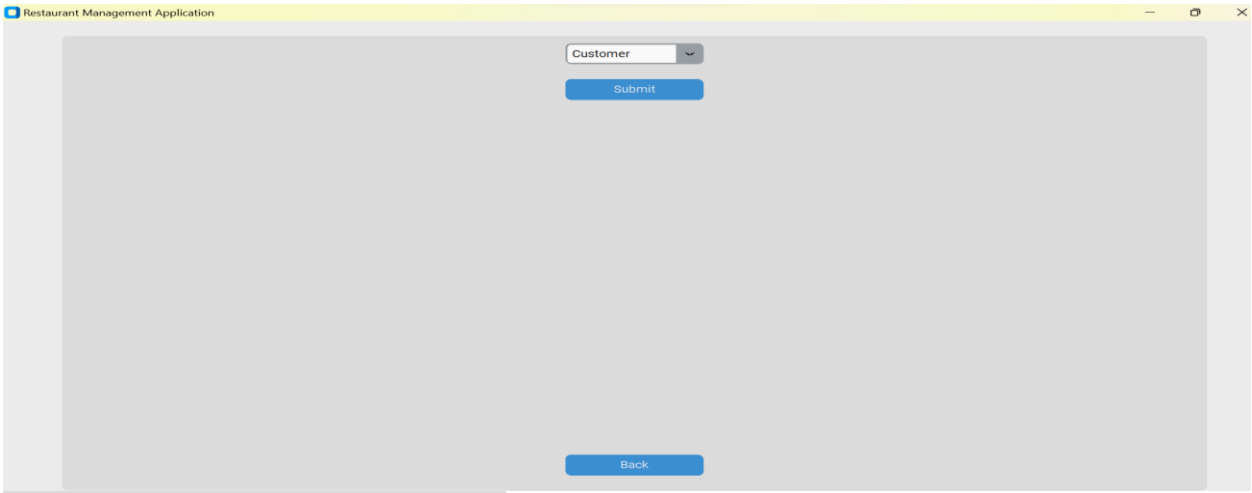

Restaurant Management Application
Employee_id | Employee_Name | EPhone_Number | EEmail | EAddress | Role | Salary | KStation_id | Restaurant_id
1 | John Smith | 123-456-7890 | john.smith@restaurant1.com | 123 Main St | Manager | 50000.00 | None | 1
2 | Jane Doe | 555-555-5555 | jane.doe@restaurant1.com | 456 Elm St | Server | 20000.00 | None | 1
3 | Samantha Brown | 999-999-9999 | samantha.brown@restaurant1.com | 789 Oak St | Chef | 60000.00 | 1 | 1
4 | Michael Johnson | 111-111-1111 | michael.johnson@restaurant1.com | 1010 Broad St | Host | 18000.00 | None | 1
5 | Sarah Lee | 222-222-2222 | sarah.lee@restaurant1.com | 1313 5th Ave | Chef | 60000.00 | 2 | 1
6 | Peter Green | 333-333-3333 | peter.green@restaurant1.com | 1212 6th Ave | Chef | 60000.00 | 3 | 1
7 | Alex Davis | 444-444-4444 | alex.davis@restaurant2.com | 222 Main St | Manager | 50000.00 | None | 1
8 | Emily Wilson | 777-777-7777 | emily.wilson@restaurant2.com | 333 Elm St | Server | 20000.00 | None | 1
9 | Olivia Lee | 999-999-9999 | olivia.lee@restaurant2.com | 555 Broad St | Host | 18000.00 | None | 1
10 | Jessica Smith | 666-666-6666 | jessica.smith@restaurant3.com | 444 Elm St | Server | 20000.00 | None | 1
Back

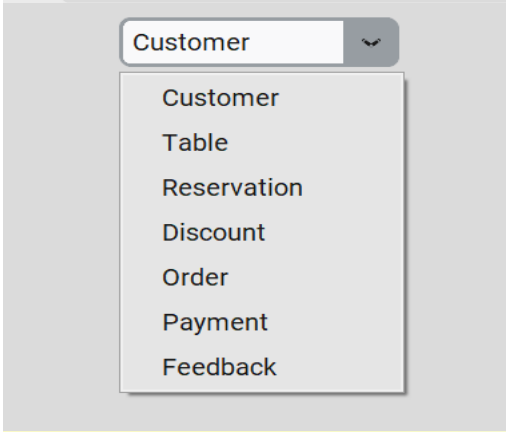


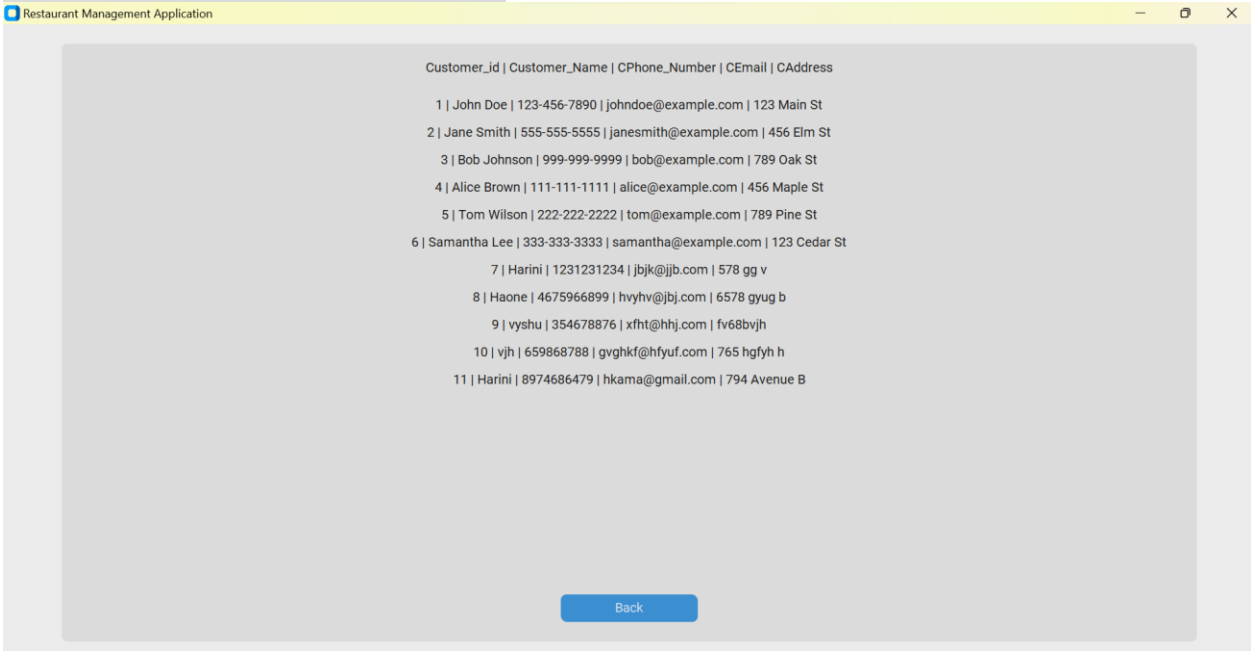


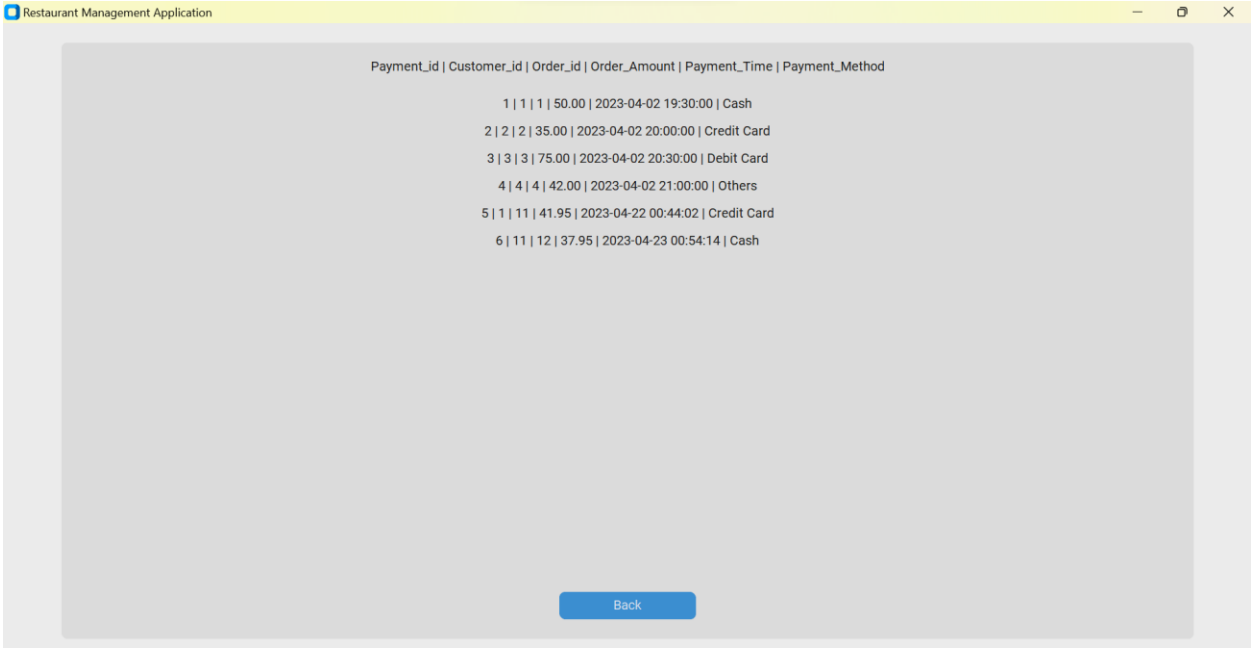
Customers Orders Data Screen:

Displays the dropdown box with the list of tables. Here for example, I'm choosing "Customer" and "Payment" tables to display the data to the user.


Restaurant Management Application
Customer
Submit
Back


Customer
Customer
Table
Reservation
Discount
Order
Payment
Feedback


Restaurant Management Application
Customer_id | Customer_Name | CPhone_Number | CEmail | CAddress
1 | John Doe | 123-456-7890 | johndoe@example.com | 123 Main St
2 | Jane Smith | 555-555-5555 | janesmith@example.com | 456 Elm St
3 | Bob Johnson | 999-999-9999 | bob@example.com | 789 Oak St
4 | Alice Brown | 111-111-1111 | alice@example.com | 456 Maple St
5 | Tom Wilson | 222-222-2222 | tom@example.com | 789 Pine St
6 | Samantha Lee | 333-333-3333 | samantha@example.com | 123 Cedar St
7 | Harini | 1231231234 | jbjk@jjb.com | 578 gg v
8 | Haone | 4675966899 | hvyhv@jbj.com | 6578 gyug b
9 | vyshu | 354678876 | xfht@hhj.com | fv68bvjh
10 | vjh | 659868788 | gvghkf@hfyuf.com | 765 hgfyh h
11 | Harini | 8974686479 | hkama@gmail.com | 794 Avenue B
Back



Future Implementations:

In this Restaurant Management System Application, as future enhancement, we can implement Employee Management and Supplier Inventory Management related screens, where Owner or Manager of the Restaurant can add, modify and delete the data related to Employees, Suppliers, Supplier Orders, Food related Menu.