Restaurant Management System

Session: 2023 – 2027

Submitted by:

Mohammad Bilal

2023-CS-168

Supervised by:

Sir Laeeq uz Zaman

Khan Niazi

Department of Computer Science

University of Engineering and Technology

Lahore, Pakistan

Table of Contents

1. Project Description:

1.1 Purpose:

The purpose of the Restaurant Management System is to streamline the operations of the any booming restaurant that is in need to expand its operations. It aims to provide an efficient and user-friendly interface for placing orders, managing orders, inventory, and other related activities.

1.2 Scope:

The system is designed to cater specifically to the needs of the restaurants focusing on order processing, inventory management, and overall restaurant administration.

1.3 Objectives:

- Automate order placement and processing.
- Manage inventory efficiently.
- Provide a user-friendly graphical interface for both customers and staff.

1.4 Major Features:

- User-friendly GUI for order placement and processing.
- Inventory management for tracking product availability.
- Role-based access control for secure operations.
- Real-time order tracking.
- Reservation management.
- Forgot password functionality.
- Messaging and notification system.
- Allow customers to upload profile
- Interactive and Responsive UI.
- Incentives for customers to attract them to this app.
- Generating reports for sales, order history, users and many more.
- Customer’s cart info is saved even after closing app.
- Implemented complete functionality with both DB and FH.
- Complete implementation of OOP Features.

2. Technical Details:

2.1 Development Environment:

- Integrated Development Environment (IDE): Visual Studio
- Version Control: Git

2.2 Programming Languages:

Programming languages used are C# and SQL.

2.3 Framework:

.NET Framework

2.4 Database:

SQL Server for data storage and retrieval.

3. User Guide:

3.1 Getting Started:

On launching the application, the customers lands on homepage and can navigate to Sign in or Sign-up page, after successfully signing up, he can sign in using valid credentials. From here on out the user will explore the options and features available.

3.2 User Roles and Permissions:

Administrator:

Full access to all the management features.

Staff:

Limited access for order processing and products management. They include *Chefs* and *Waiters.*

Customer:

Limited access to many features including placing orders, making reservations, giving feedback etc.

3.3 Functionality Overview:

3.3.1 Admin:

- CRUD Operations for Customers and Employees.
- Managing Tables.
- Add new Admin.
- View Analytics.
- Notify Users and Reply to their messages.
- View User Reviews.
- Change Username/Password.

- Delete Account.
- Log Out.

3.3.2 Customer:

- Can promote to VIP to access more features and get rewards.
- Place and track orders.
- View order history and details.
- Make a Reservation.
- Can upload Profile.
- Provide feedback.
- Messaging System.
- Change Username/Password.
- Delete Account.
- Log Out.

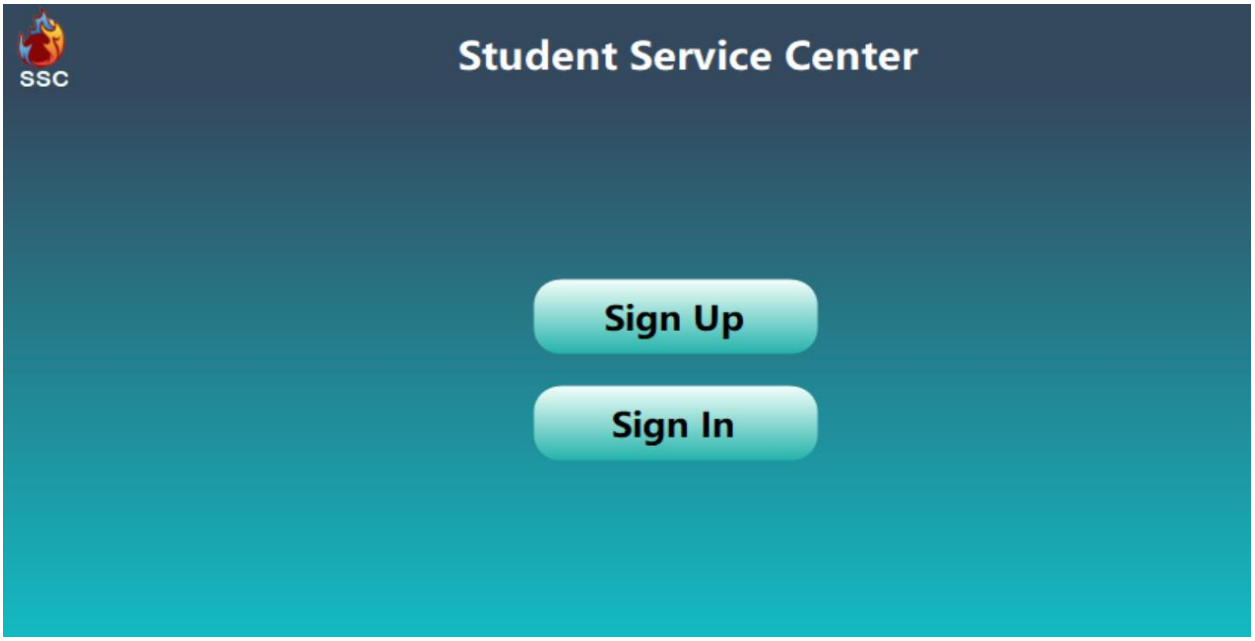
3.3.2 Staff (Waiters and Chefs):

- Table Reservation Management.
- Order Processing.
- Products and Deals Management.
- Order Status Updates.
- Log Out

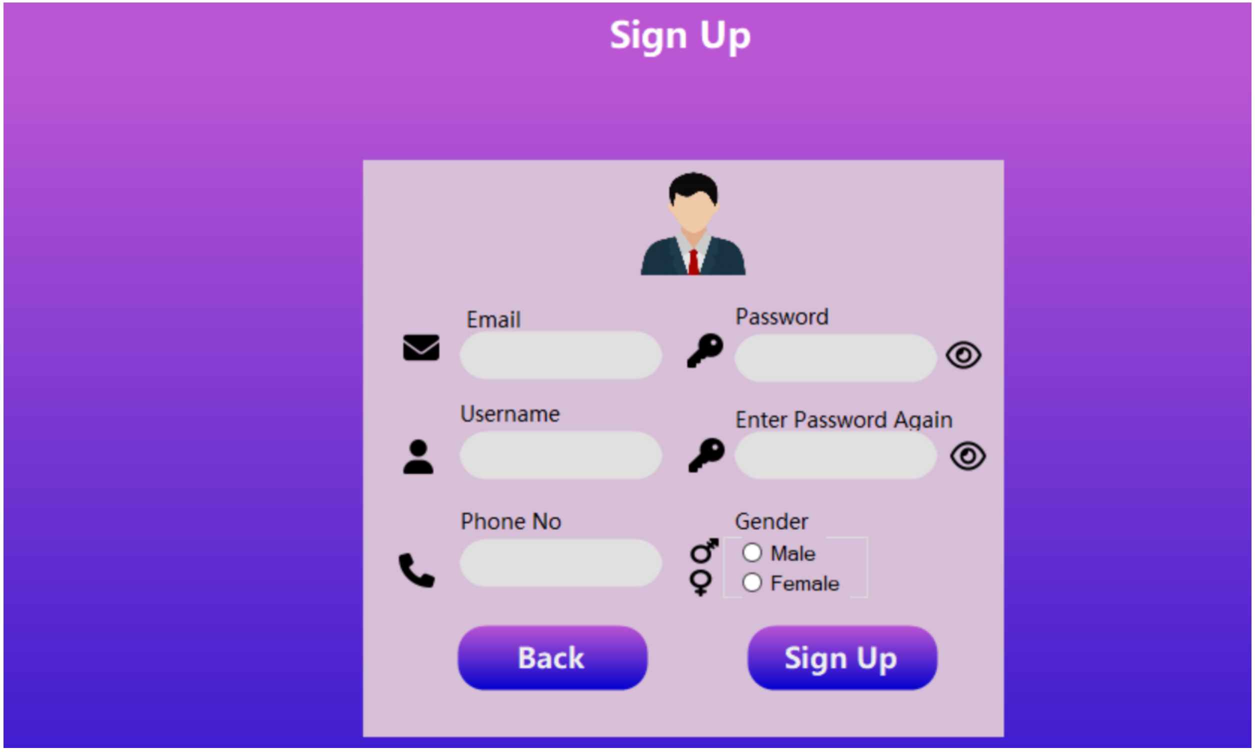
4. Wireframes:

4.1 Homepage:

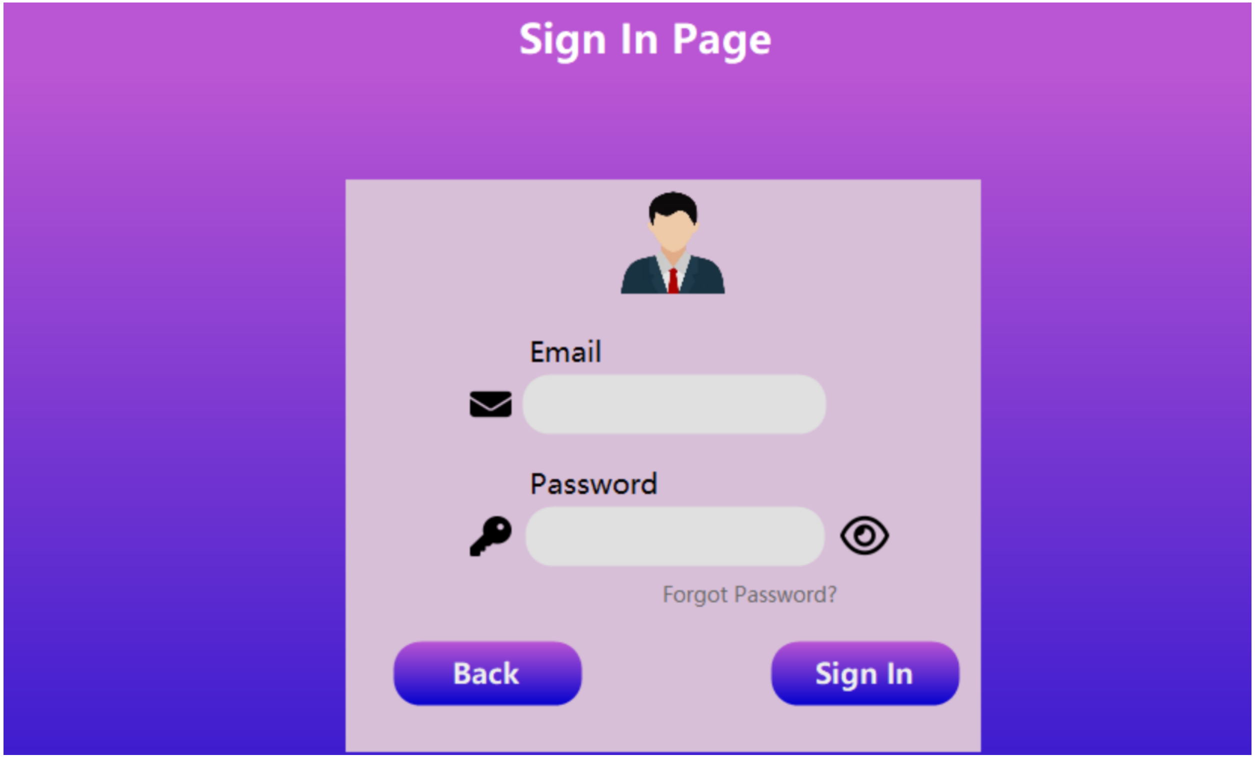
This is the landing page for all users.



4.2 Sign Up Page:

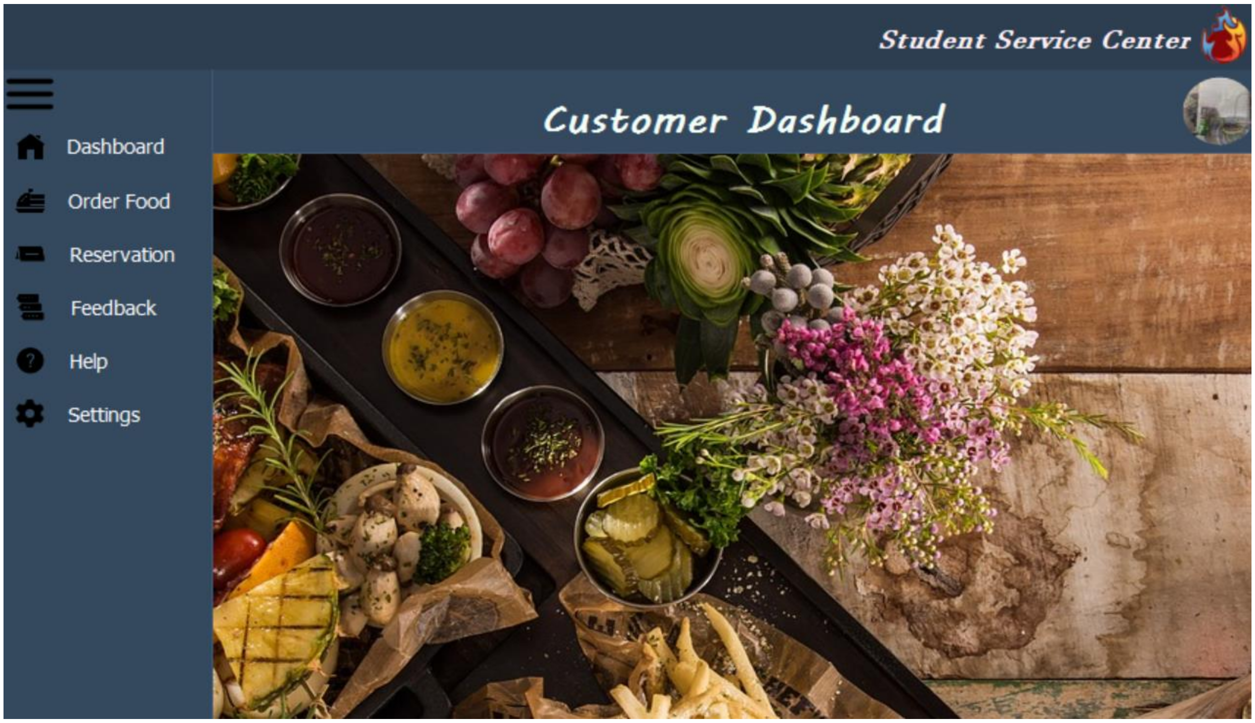


4.3 Sign-In Page:

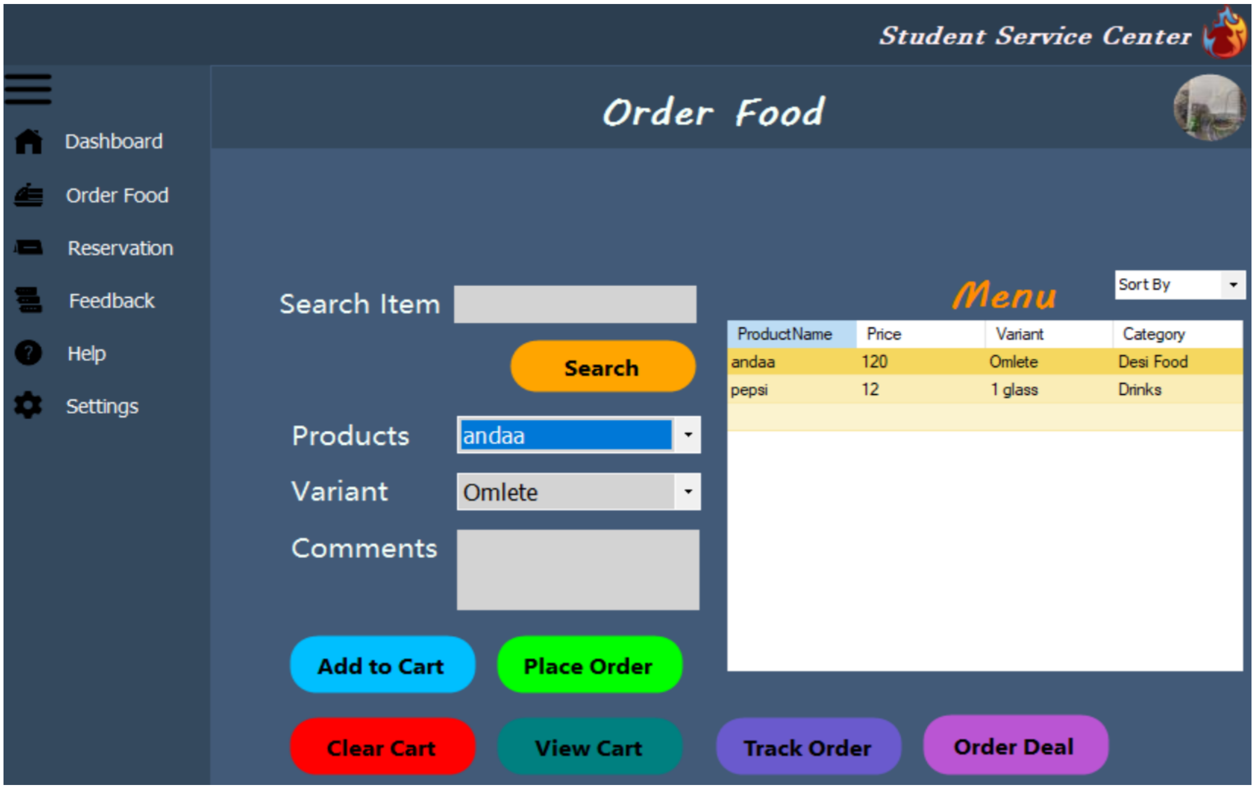


4.4 Customer:

4.4.1 Landing Page:



4.4.2 Order Food:

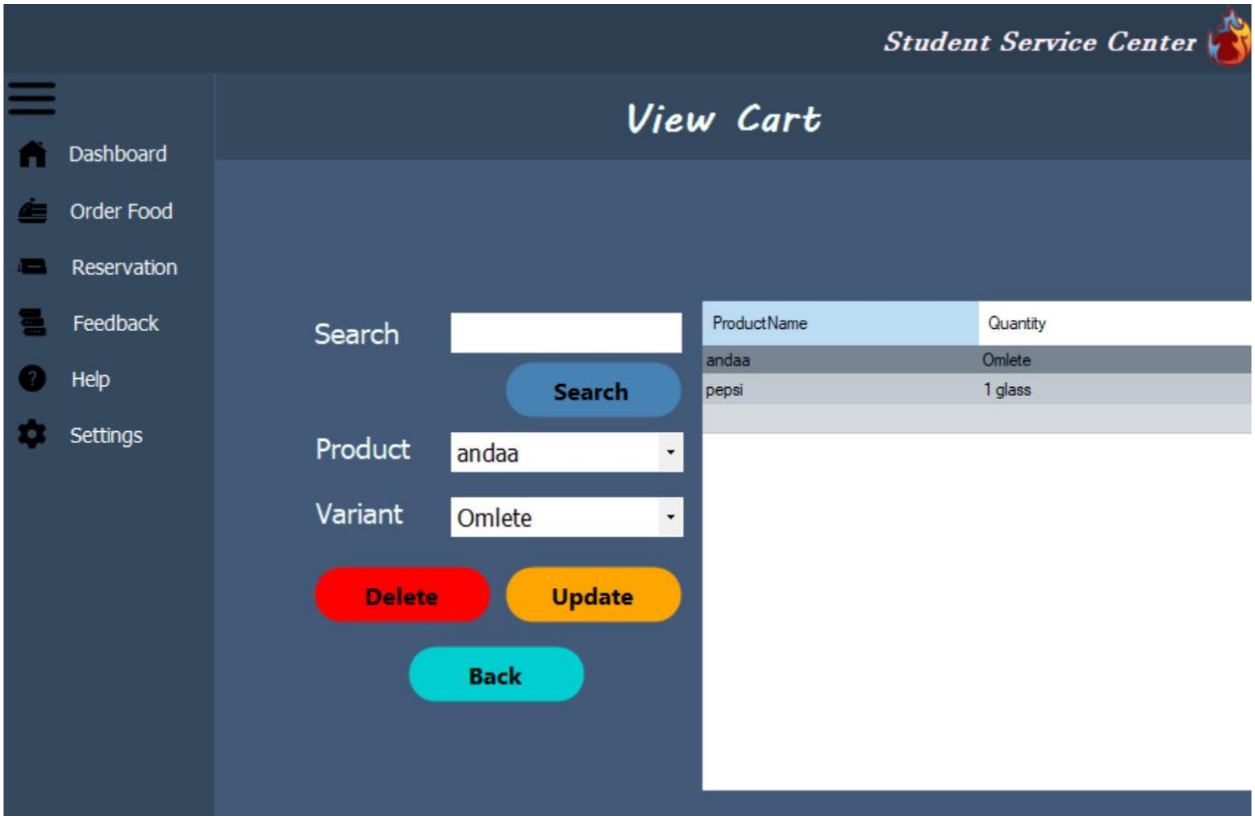


Products: andaa

Variant: Omlete

ProductName	Price	Variant	Category
andaa	120	Omlete	Desi Food
pepsi	12	1 glass	Drinks

4.4.3 View Cart:

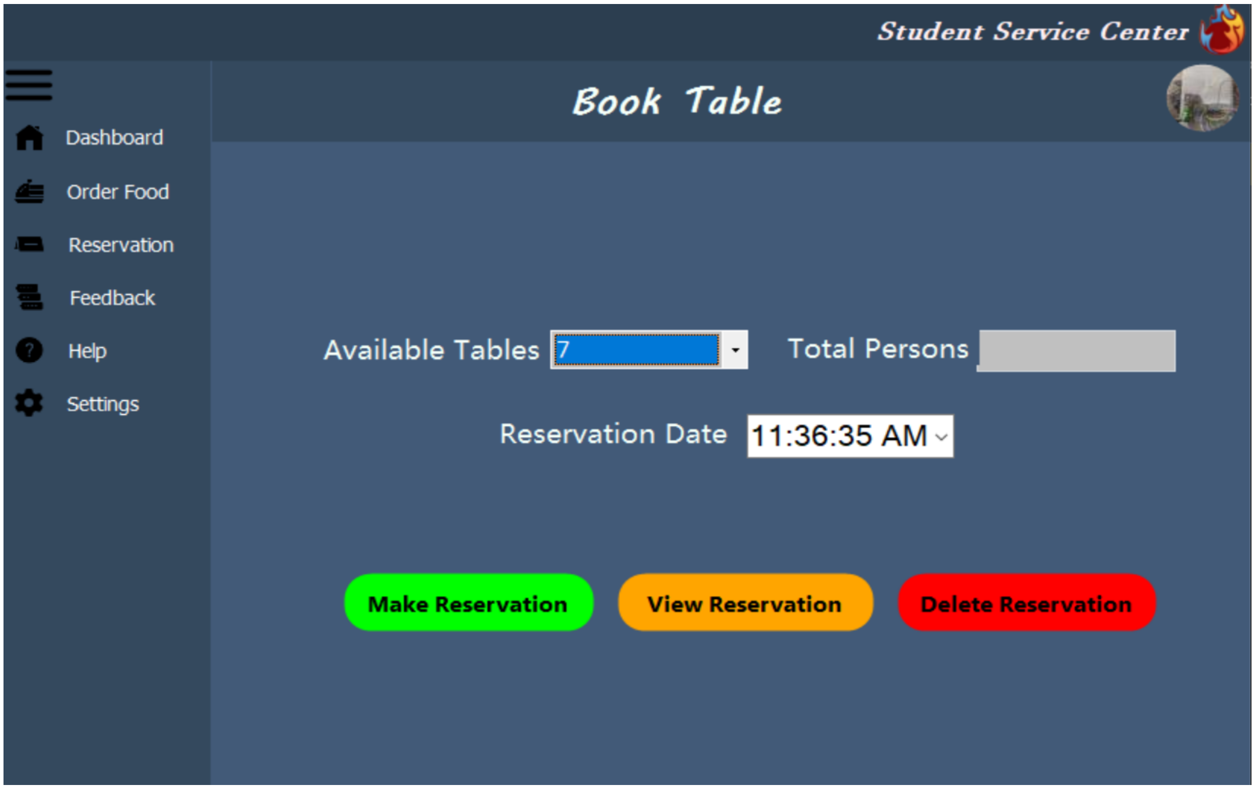


Product: andaa

Variant: Omlete

ProductName	Quantity
andaa	Omlete
pepsi	1 glass

4.4.4 Book Table:

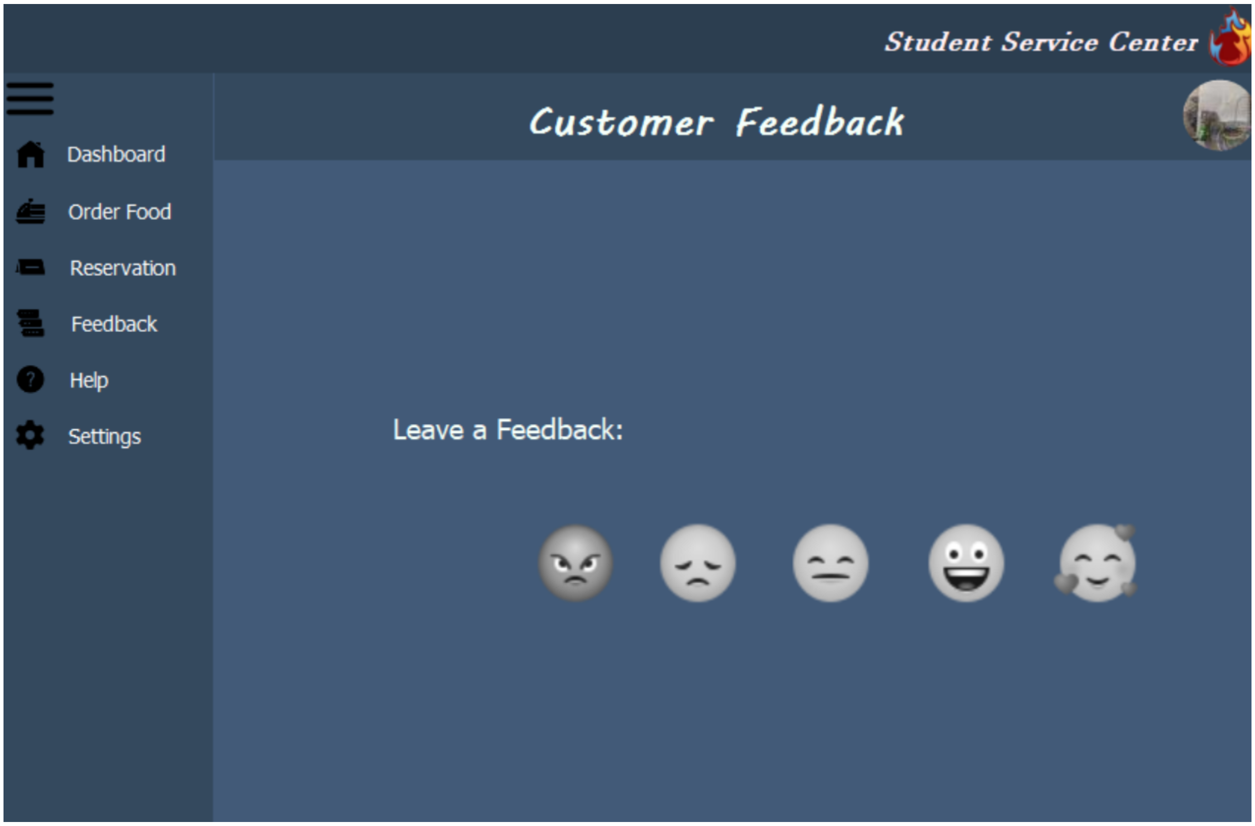


Available Tables: 7

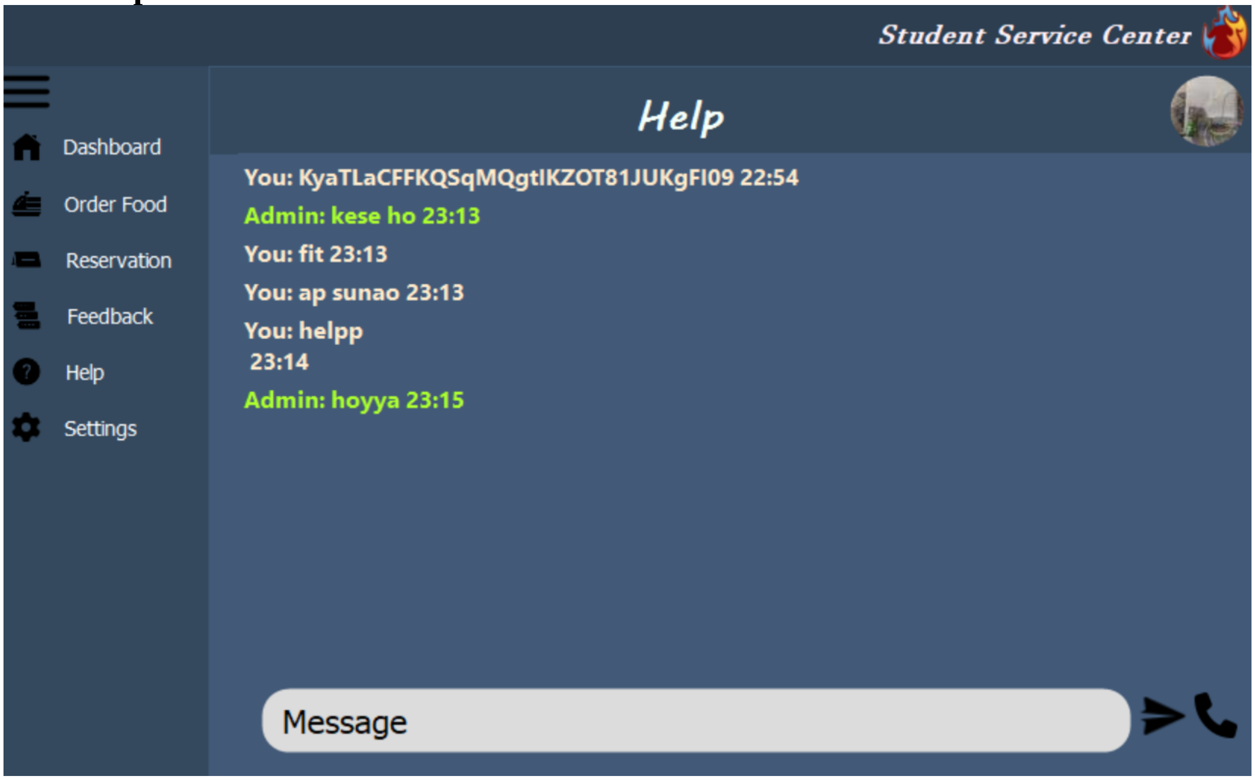
Total Persons:

Reservation Date: 11:36:35 AM

4.4.5 Feedback:



4.4.6 Help:



You: KyaTLaCFFKQSqMQgtIKZOT81JUKgFI09 22:54

Admin: kese ho 23:13

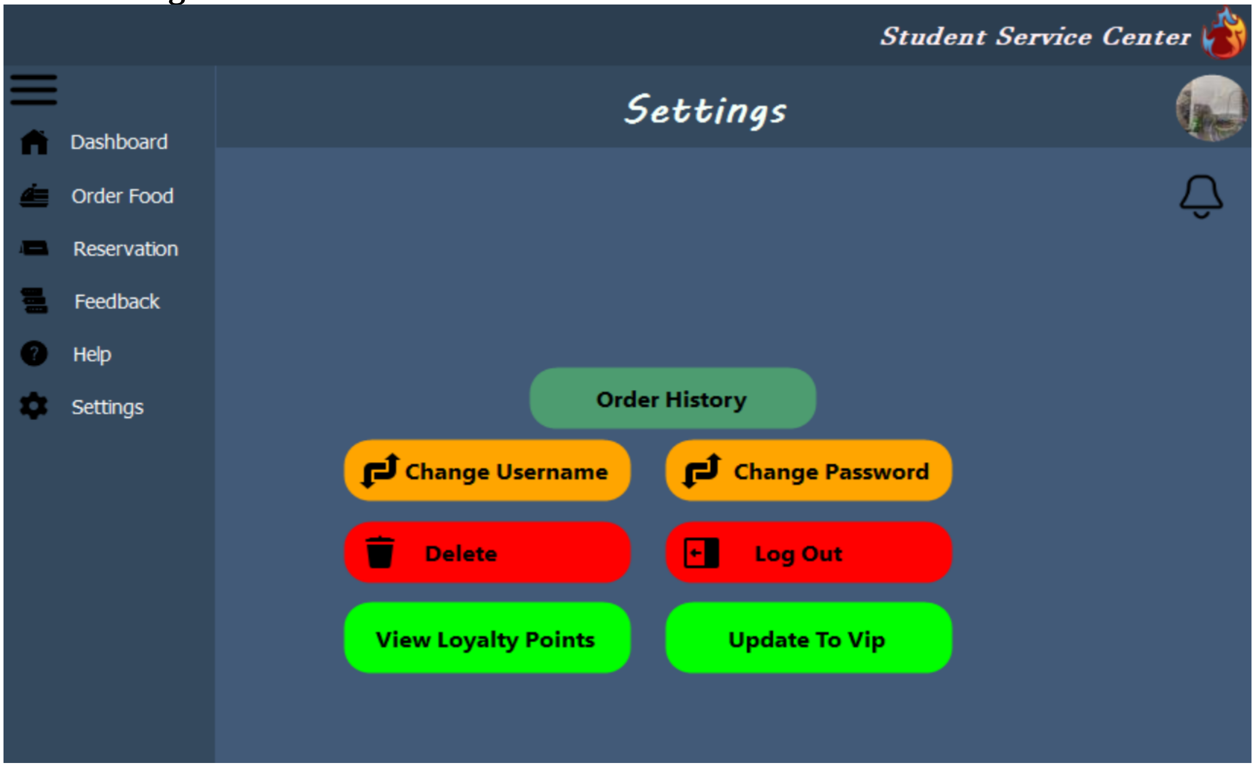
You: fit 23:13

You: ap sunao 23:13

You: helpp
23:14

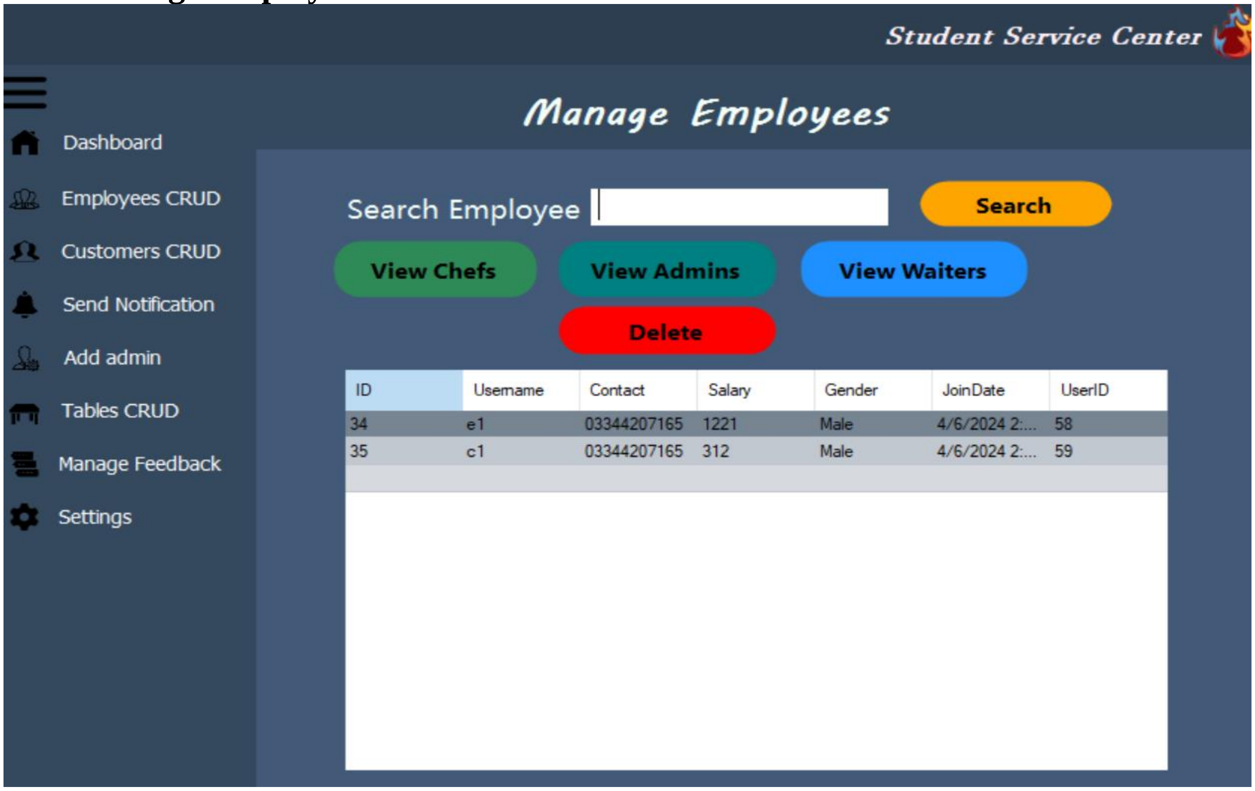
Admin: hoyya 23:15

4.4.7 Settings:



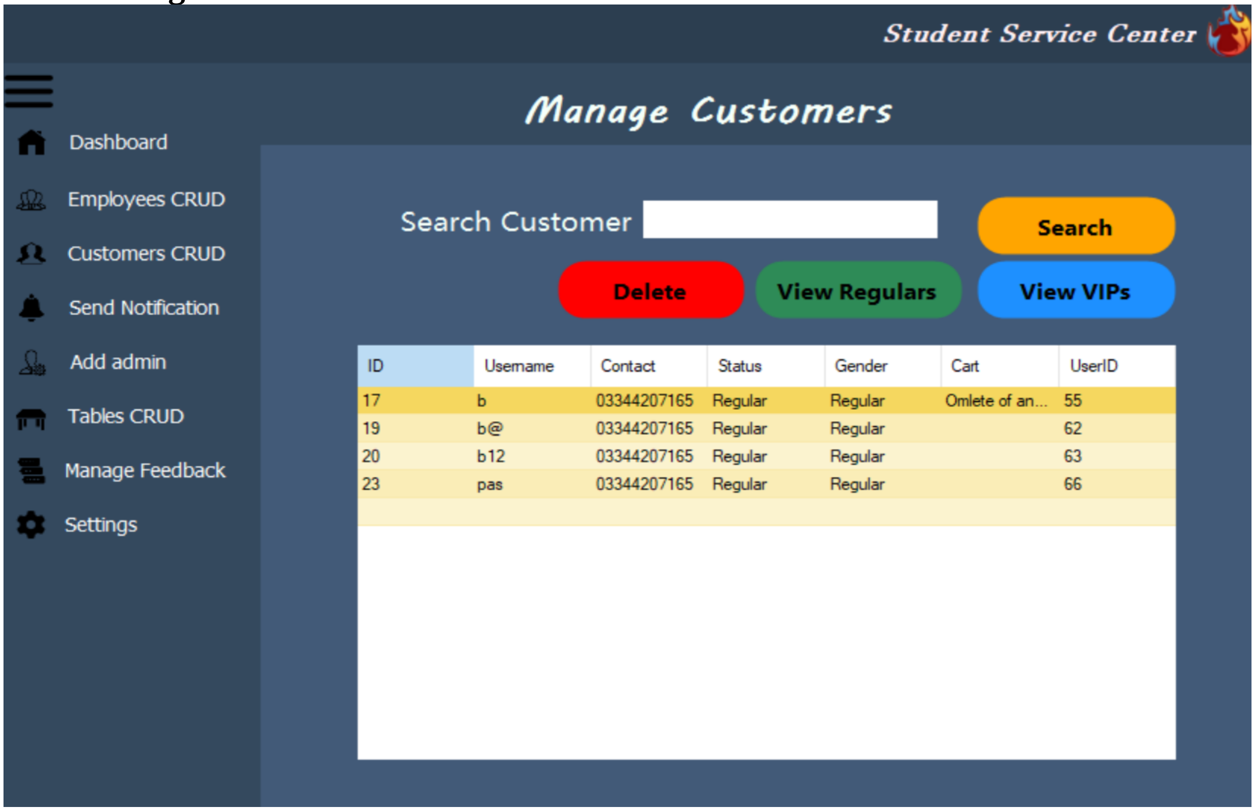
4.5 Admin:

4.5.1 Manage Employees:



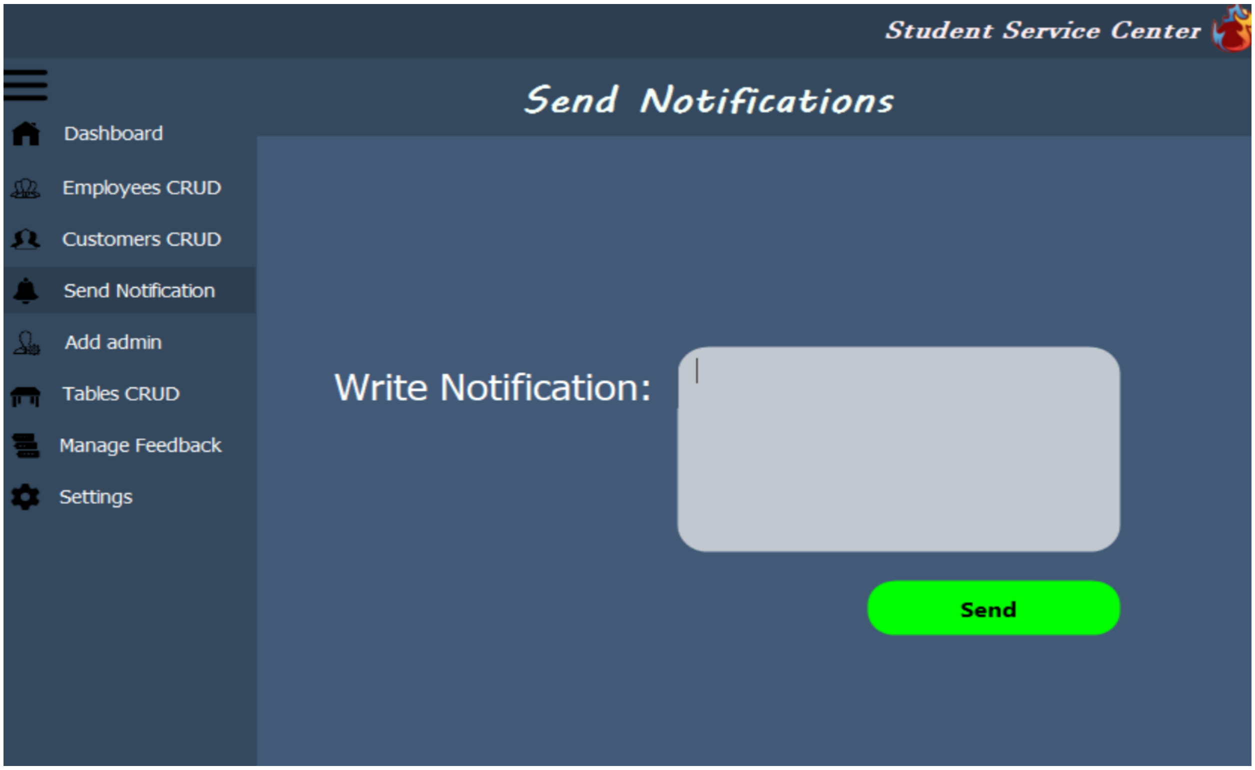
ID	Username	Contact	Salary	Gender	JoinDate	UserID
34	e1	03344207165	1221	Male	4/6/2024 2:...	58
35	c1	03344207165	312	Male	4/6/2024 2:...	59

4.5.2 Manage Customers:

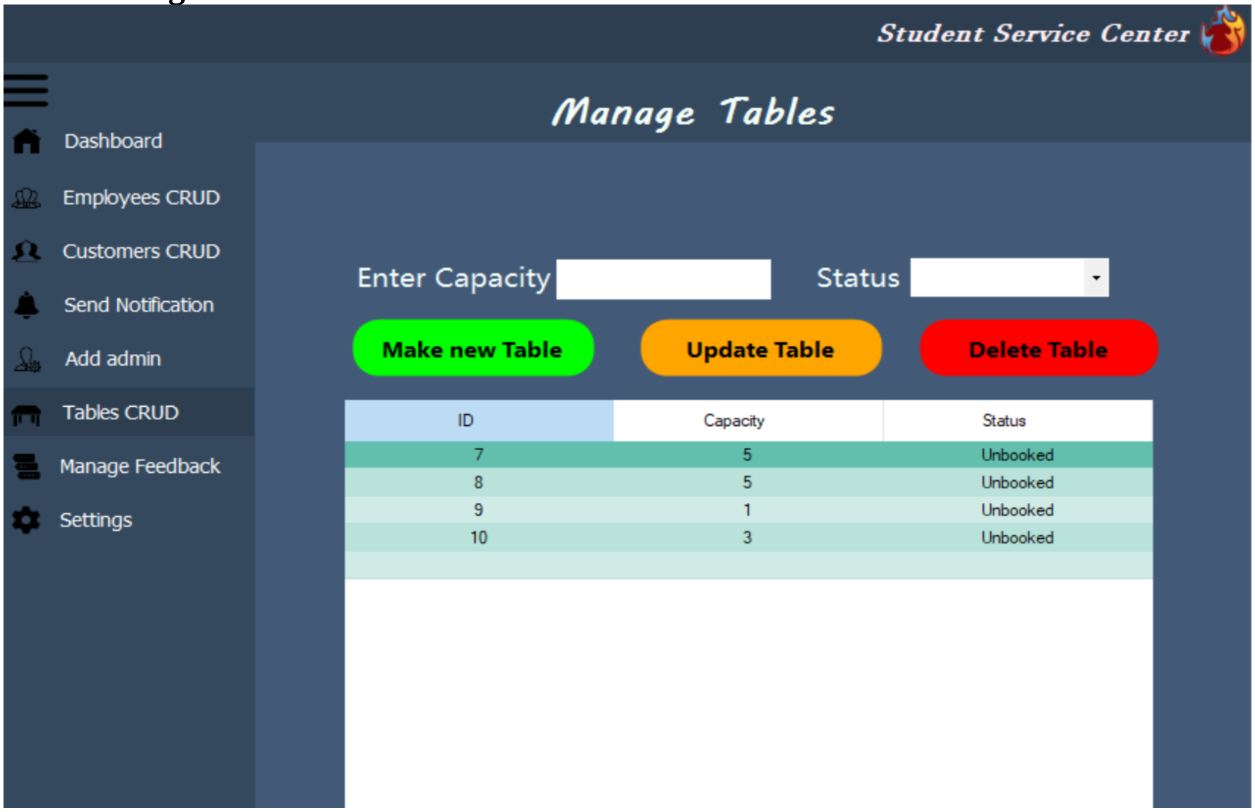


ID	Username	Contact	Status	Gender	Cart	UserID
17	b	03344207165	Regular	Regular	Omlete of an...	55
19	b@	03344207165	Regular	Regular		62
20	b12	03344207165	Regular	Regular		63
23	pas	03344207165	Regular	Regular		66

4.5.3 Send Notifications:

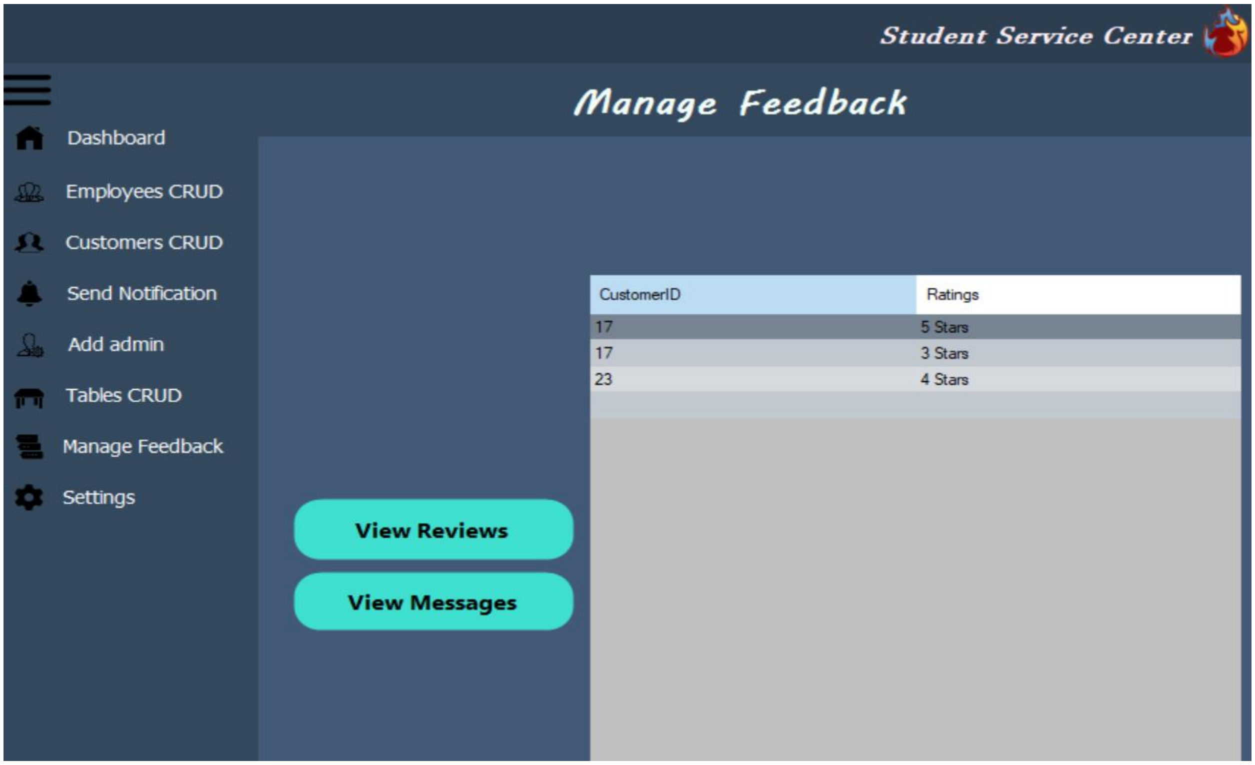


4.5.4 Manage Tables:



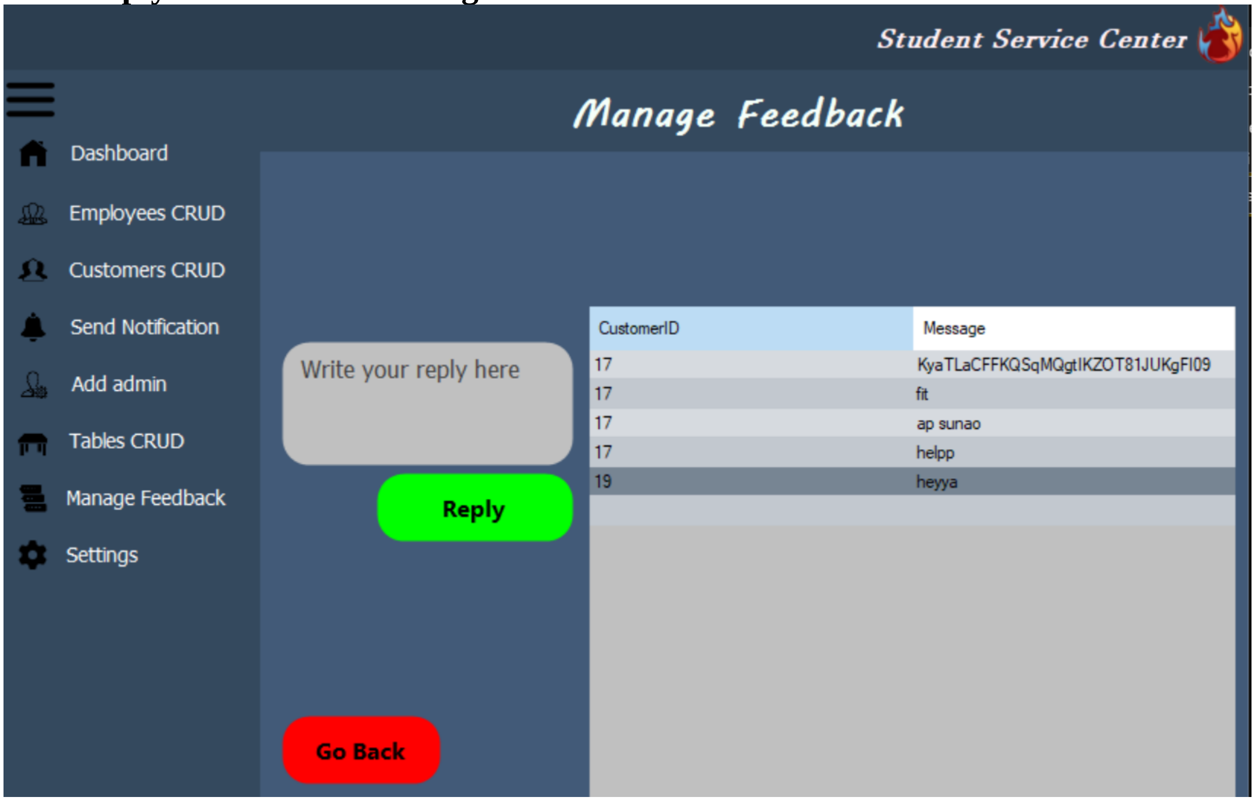
ID	Capacity	Status
7	5	Unbooked
8	5	Unbooked
9	1	Unbooked
10	3	Unbooked

4.5.5 View Feedback:



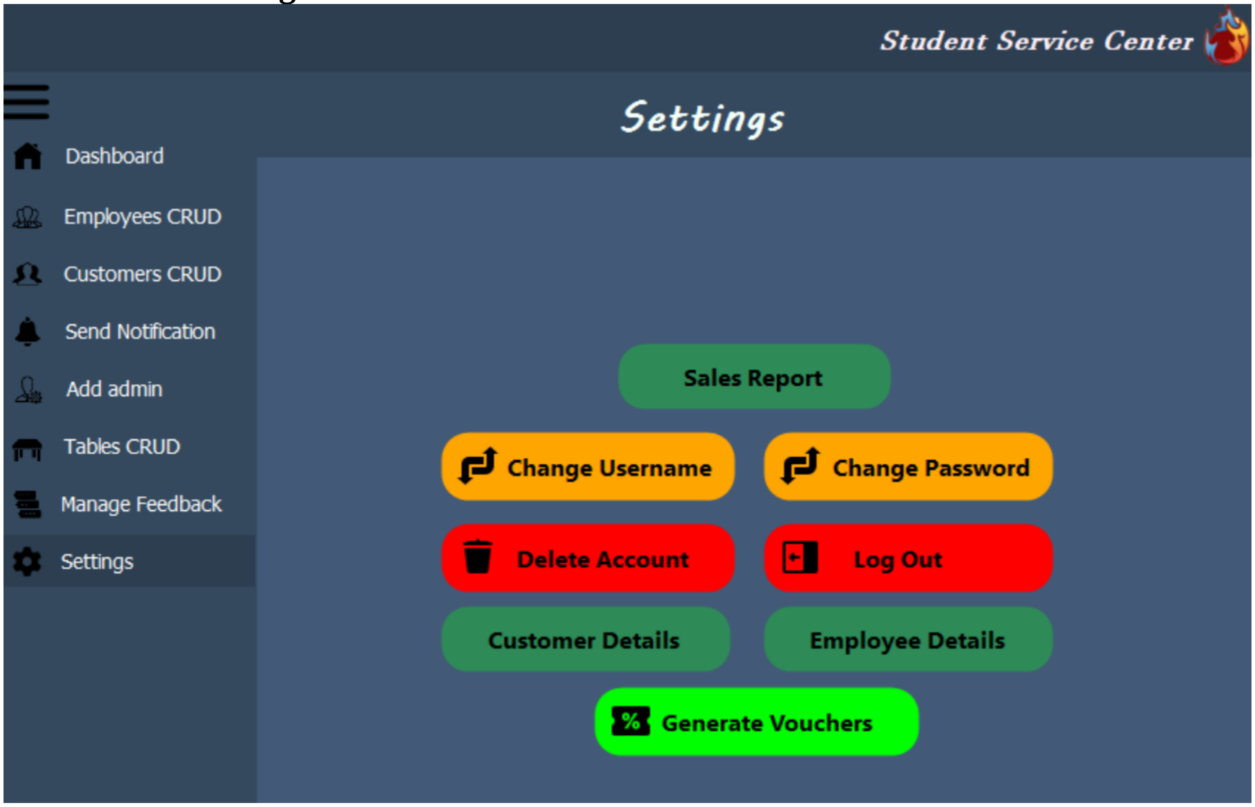
CustomerID	Ratings
17	5 Stars
17	3 Stars
23	4 Stars

4.5.6 Reply to Customer Messages:



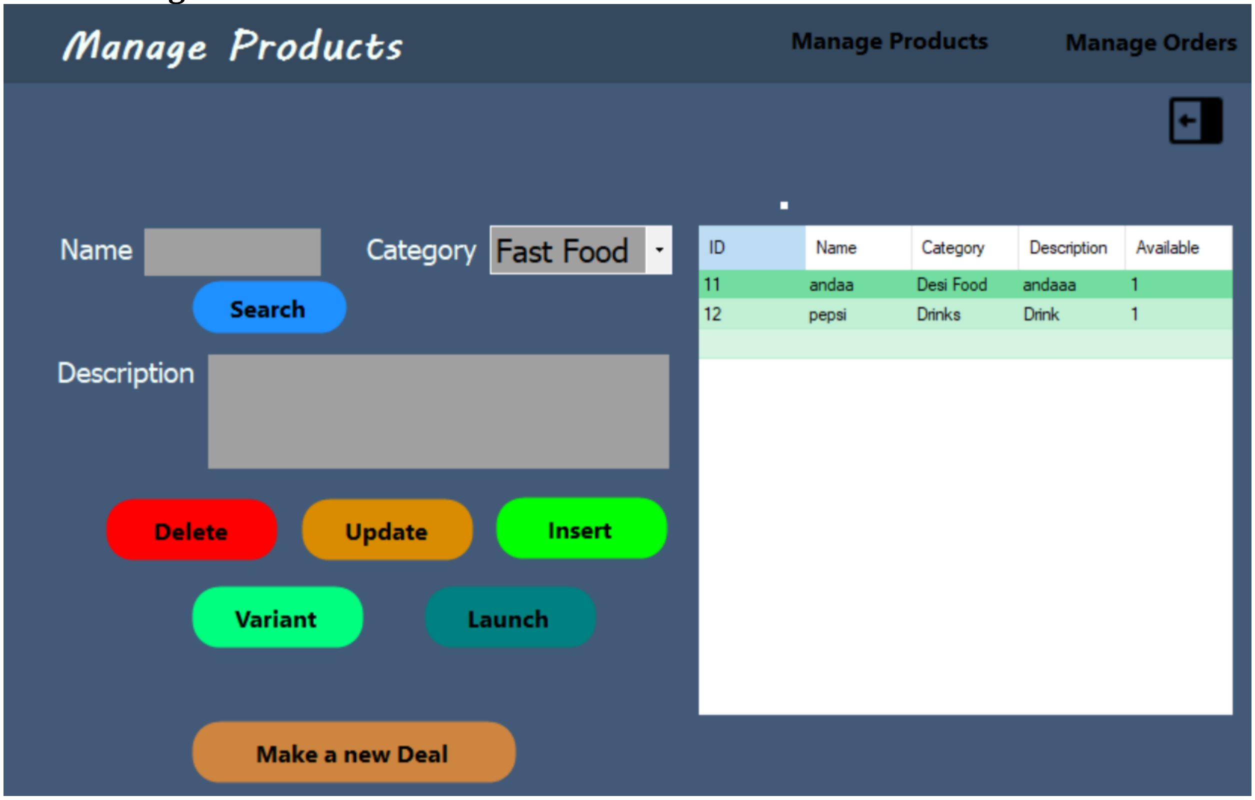
CustomerID	Message
17	KyaTLaCFFKQSqMQgtIKZOT81JUKgFI09
17	fit
17	ap sunao
17	helpp
19	heyya

4.5.7 Admin Settings:



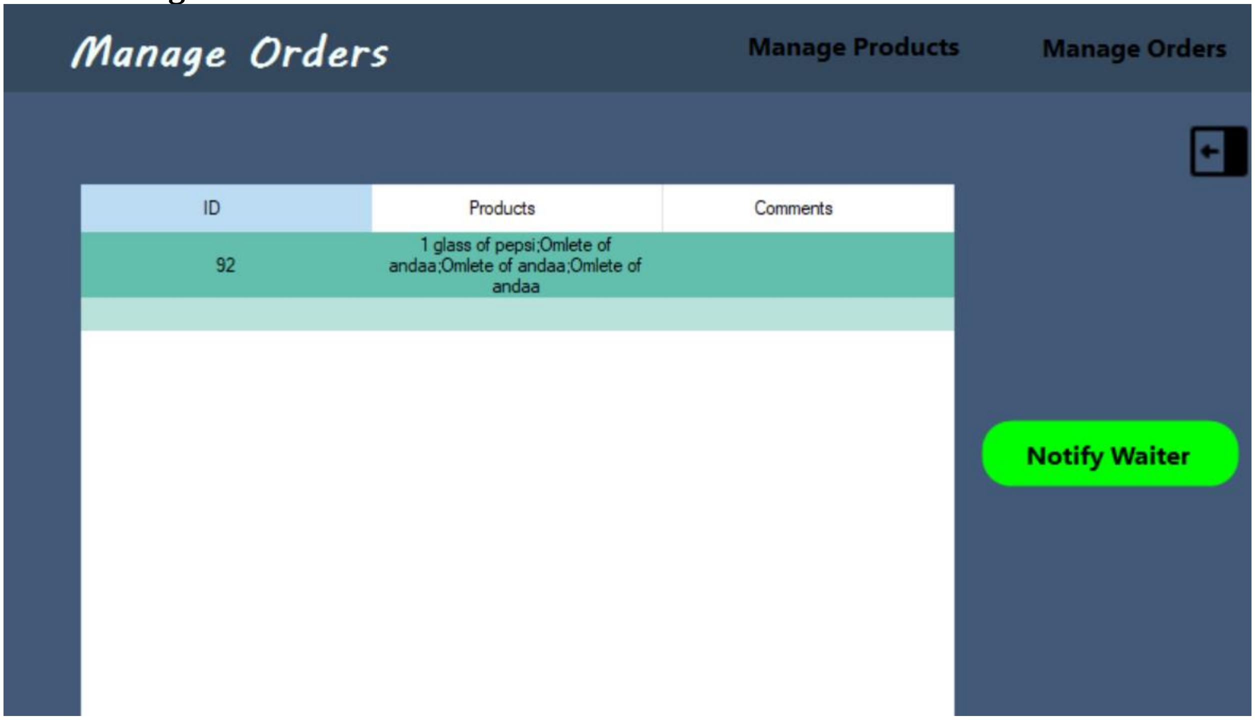
4.6 Chef:

4.6.1 Manage Products:



ID	Name	Category	Description	Available
11	andaa	Desi Food	andaaa	1
12	pepsi	Drinks	Drink	1

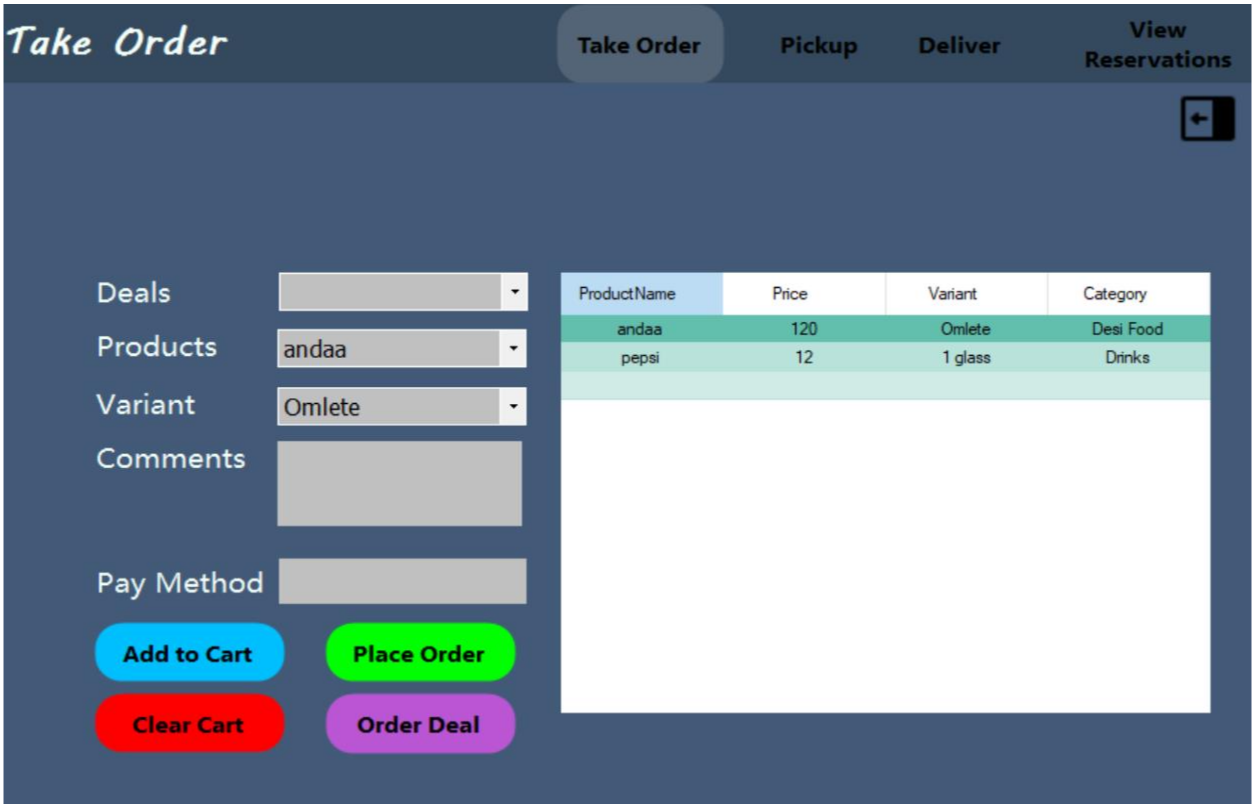
4.6.2 Manage Orders:



ID	Products	Comments
92	1 glass of pepsi;Omlete of andaa;Omlete of andaa;Omlete of andaa	

4.7 Waiter:

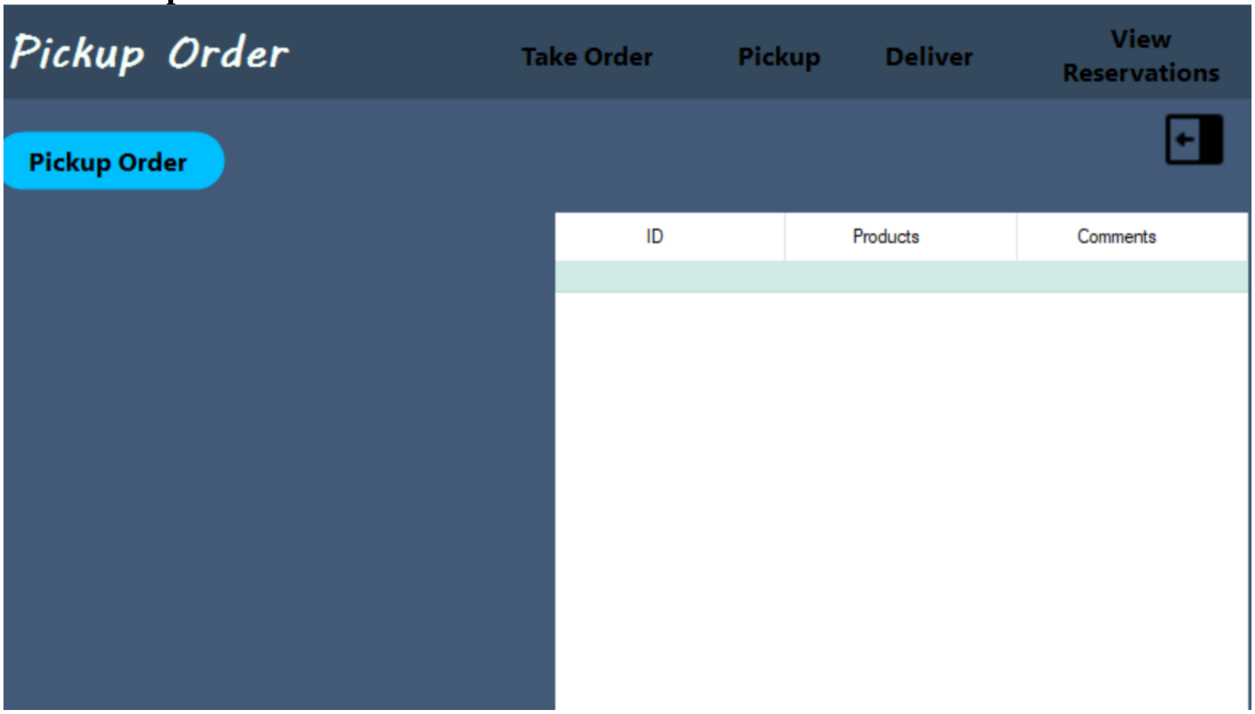
4.7.1: Take Order:



Deals	
Products	andaa
Variant	Omlete
Comments	
Pay Method	

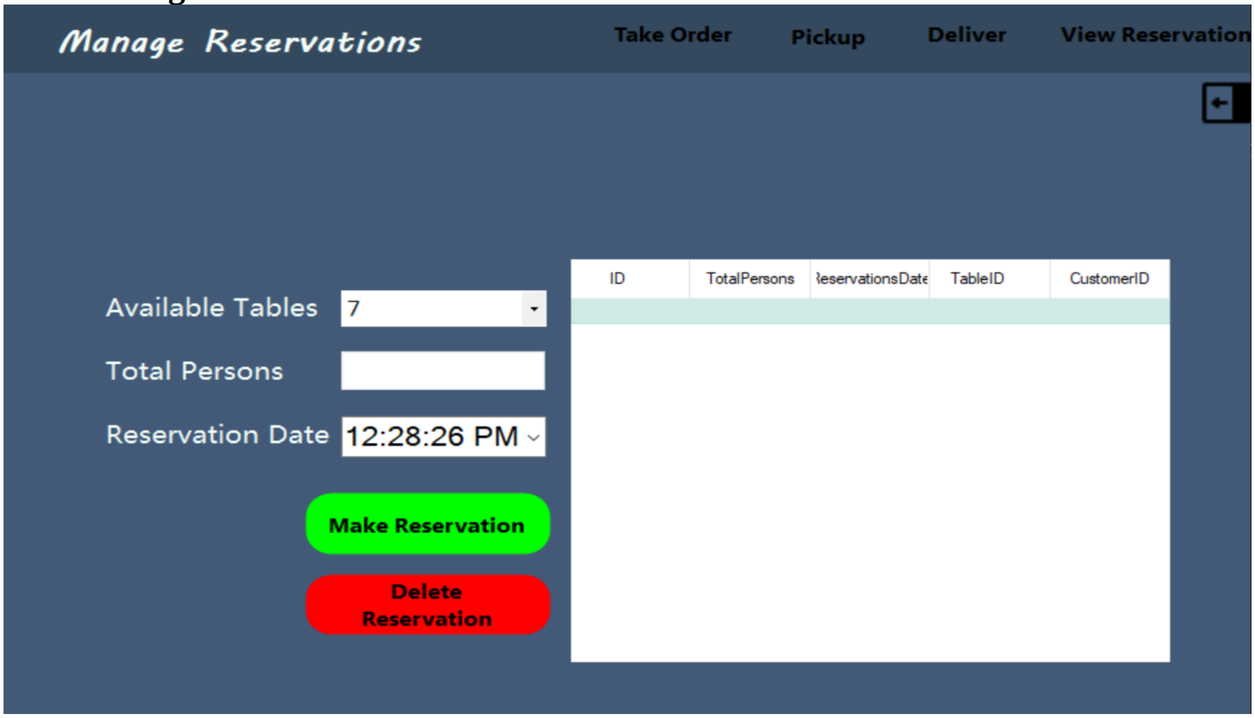
ProductName	Price	Variant	Category
andaa	120	Omlete	Desi Food
pepsi	12	1 glass	Drinks

4.7.2 Pickup and Deliver Order:



ID	Products	Comments

4.7.3 Manage Reservations:



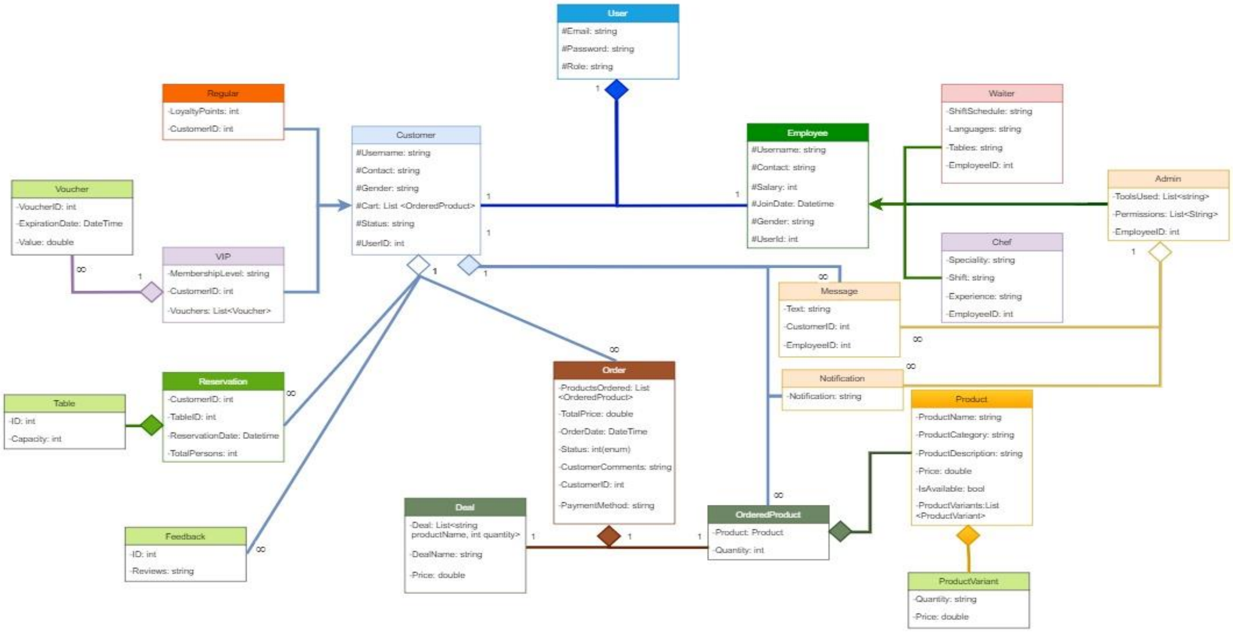
Available Tables 7

Total Persons

Reservation Date 12:28:26 PM

ID	TotalPersons	ReservationsDate	TableID	CustomerID

5. CRC Diagram:



6. Code:

```
    public class Feedback
    {
        private string Review;
        private int CustomerID;

        public Feedback(string review, int customerID)
        {
            Review = review;
            CustomerID = customerID;
        }

        public string GetReview()
        {
            return Review;
        }

        public int GetCustomerID()
        {
            return CustomerID;
        }

        public void SetCustomerID(int id)
        {
            CustomerID = id;
        }
        public void SetReview(string review)
        {
            Review = review;
        }
    }


public interface IFeedbackDL
    {
        void SaveReview(Feedback feedback);
        List<Feedback> GetReviews();
    }

    public class FeedbackDBDL : IFeedbackDL
    {
        public void SaveReview(Feedback feedback)
        {
            using (SqlConnection connection = UtilityFunctions.GetSqlConnection())
            {
                connection.Open();
                SqlCommand command = new SqlCommand("INSERT INTO Feedback(Reviews,
CustomerID) VALUES (@Reviews, @CustomerID)", connection);
                command.Parameters.AddWithValue("@Reviews", feedback.GetReview());
                command.Parameters.AddWithValue("@CustomerID", feedback.GetCustomerID());
                command.ExecuteNonQuery();
            }
        }

        public List<Feedback> GetReviews()
        {
            using (SqlConnection connection = UtilityFunctions.GetSqlConnection())
            {
                connection.Open();
                SqlCommand command = new SqlCommand("SELECT * FROM Feedback",
connection);
                SqlDataReader reader = command.ExecuteReader();
                List<Feedback> feedbacks = new List<Feedback>();
                while (reader.Read())
                {
                    feedbacks.Add(new Feedback(reader.GetString(1), reader.GetInt32(2)));
                }
                return feedbacks;
            }
        }
    }
    public class FeedbackFHDL : IFeedbackDL
    {
        public void SaveReview(Feedback feedback)
        {
            string path = UtilityFunctions.GetPath("Feedbacks.txt");
            int id = UtilityFunctions.AssignID(path);
            using (StreamWriter writer = File.AppendText(path))
            {
                writer.WriteLine($"{id},{feedback.GetReview()}, {feedback.GetCustomerID()}");
            }
        }

        public List<Feedback> GetReviews()
        {
            List<Feedback> feedbacks = new List<Feedback>();
            string path = UtilityFunctions.GetPath("Feedbacks.txt");

            if (File.Exists(path))
            {
                string[] lines = File.ReadAllLines(path);
                foreach (string line in lines)
                {
                    string[] parts = line.Split(',');
                    if (parts.Length == 3)
                    {
                        string review = parts[1].Trim();
                        int customerID = Convert.ToInt32(parts[2].Trim());

                        Feedback feedback = new Feedback(review, customerID);
                        feedbacks.Add(feedback);
                    }
                }
            }
            return feedbacks;
        }
    }

public class FeedbackUI
    {
        public static int TakeFeedback()
        {
            Console.ForegroundColor = ConsoleColor.DarkBlue;
            Console.WriteLine("\n\t\t\t\t\t\t\t5 Stars");
            Console.WriteLine("\t\t\t\t\t\t\t4 Stars");
            Console.WriteLine("\t\t\t\t\t\t\t3 Stars");
            Console.WriteLine("\t\t\t\t\t\t\t2 Stars");
            Console.WriteLine("\t\t\t\t\t\t\t1 Stars");
            Console.WriteLine("\t\t\t\t\t\t\tLog Out");
            Console.ResetColor();
            return ConsoleUtility.MovementOfArrow(53, 7, 1, 6);
        }

        public static void PrintReviews()
        {
            Console.ForegroundColor = ConsoleColor.DarkBlue;
            foreach(var feedback in ObjectHandler.GetFeedbackDL().GetReviews())
            {
                Console.WriteLine("\t\t\t\t\t\t\tFeedback: " + feedback.GetReview() + " Customer ID: "+ feedback.GetCustomerID());
            }
            Console.ForegroundColor = ConsoleColor.DarkBlue;
        }
    }
```

7. GitHub Link:

https://github.com/mb0227/Business-App-Project-2nd-Semester