

## **TABLE OF CONTENTS**

<b>Serial No.</b>	<b>Contents</b>	<b>Page No.</b>
1.	Abstract	1
2.	Introduction	2
3.	Problem Statement	3
4.	Hardware and software requirements	4
5.	Module description	5
6.	Use Case diagram	6
7.	Sequence diagram	7
8.	Testing	8-9
9.	Code	10-14
10.	Output	15-16
11.	Conclusion	17
12.	Reference	18

## **Abstract**

The Pizza Ordering System using C# is designed to streamline the process of ordering pizzas, managing orders, and handling transactions. The system will consist of several modules including customer management, menu management, order processing, and payment handling. Customers will be able to browse through the menu, select their desired pizzas, customize toppings, specify delivery or pickup options, and complete their orders. Restaurant staff will have access to an administrative interface where they can manage menus, track orders, update order statuses, and handle payments.

## **Introduction**

In today's fast-paced world, convenience is paramount, and one area where this is evident is in food ordering systems. Pizza, being one of the most popular and versatile dishes worldwide, is often at the forefront of such systems. With the advent of technology, online pizza ordering systems have become increasingly common, offering customers the ease of ordering their favourite pizzas from the comfort of their homes or on-the-go. This project aims to develop a Pizza Ordering System using C#, providing a seamless and user-friendly experience for both customers and restaurant staff.

## **Problem Statement**

The development of a pizza ordering system aims to streamline the process of ordering pizzas by providing customers with an interactive platform that allows for customization of their orders. Customers often seek convenience and personalization when ordering food online, and a system that can accommodate specific preferences such as pizza type, quantity, size, and additional toppings is essential in meeting these demands. The challenge lies in creating an intuitive user interface that simplifies the selection process while also incorporating a dynamic pricing model that adjusts the cost based on the chosen customizations.

## **Hardware and software requirements**

### **Hardware requirements:**

**RAM:** Sufficient RAM at least 4GB to access data in memory, reducing disk I/O and a dual-core processor.

**Storage:** Fast and reliable storage (e.g., SSDs) to ensure quick access to data.

### **Software Requirements:**

**Operating System:** Windows 10 or higher.

**Programming Languages and Frameworks:** C# (for console application), .NET Framework 4.7.

**Development Tools:** Integrated Development Environment (IDE) such as Visual Studio 2022.

## **Module Description**

### **Make Order Module:**

**Functionality:** Allows the user to create and place an order for pizzas with customizable options such as quantity, toppings, and size.

#### **Key Components:**

- Displays the menu of available pizzas along with their prices.
- Prompts the user to select pizzas by ID and specify the quantity.
- Offers the option to add toppings to the selected pizzas, each topping incurring an additional cost.
- Provides a menu for selecting the size of the pizza, with associated costs.
- Calculates the total cost of the order based on the chosen options.
- Stores the order details and total cost for further processing.

### **Delete Order Module:**

**Functionality:** Enables the user to remove an existing order from the system.

#### **Key Components:**

- Displays the list of existing orders with their respective details (pizza, quantity, total cost).
- Prompts the user to input the index of the order they wish to delete.
- Validates the input index and removes the corresponding order from the system.
- Notifies the user upon successful deletion of the order.

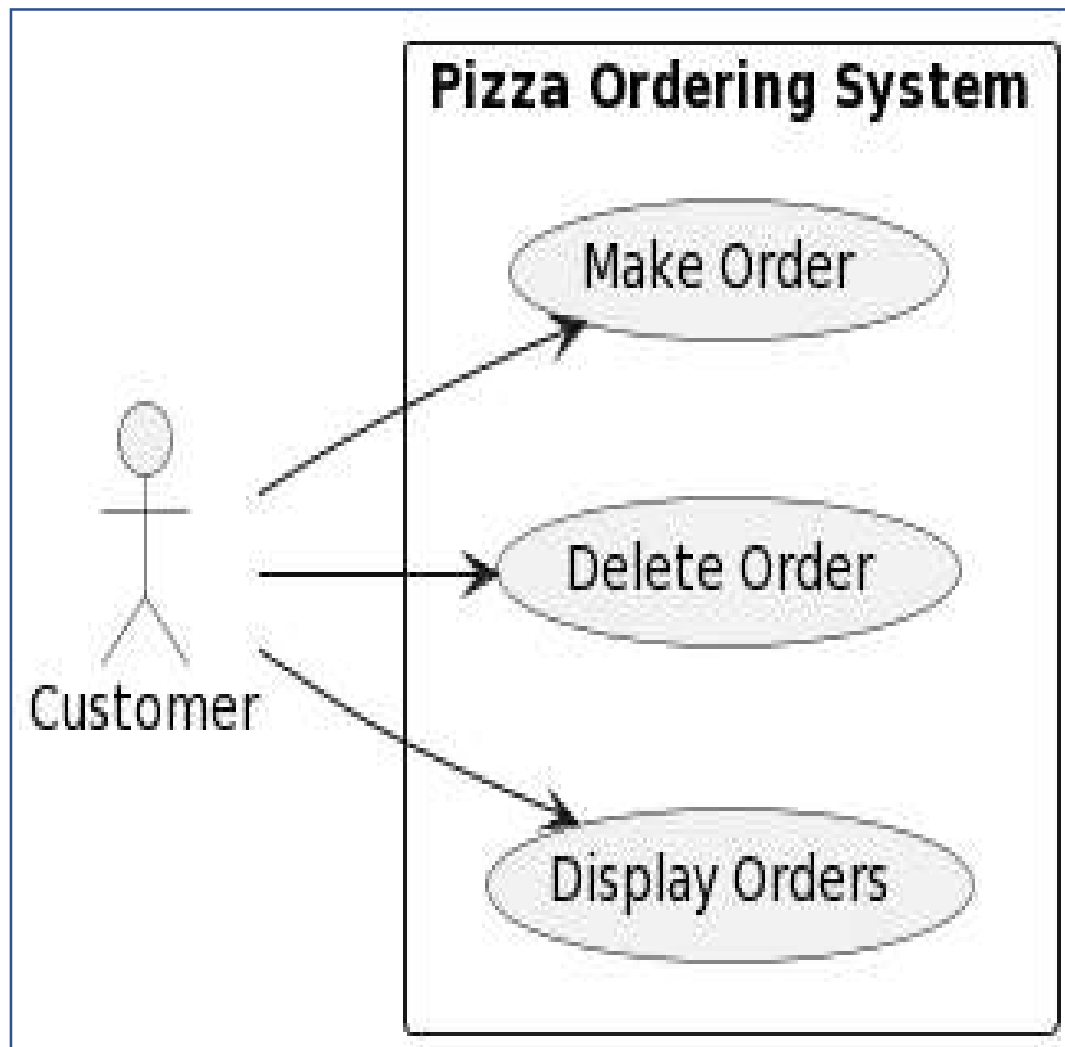
### **Display Order Module:**

**Functionality:** Shows the list of existing orders along with their details, providing visibility into current orders.

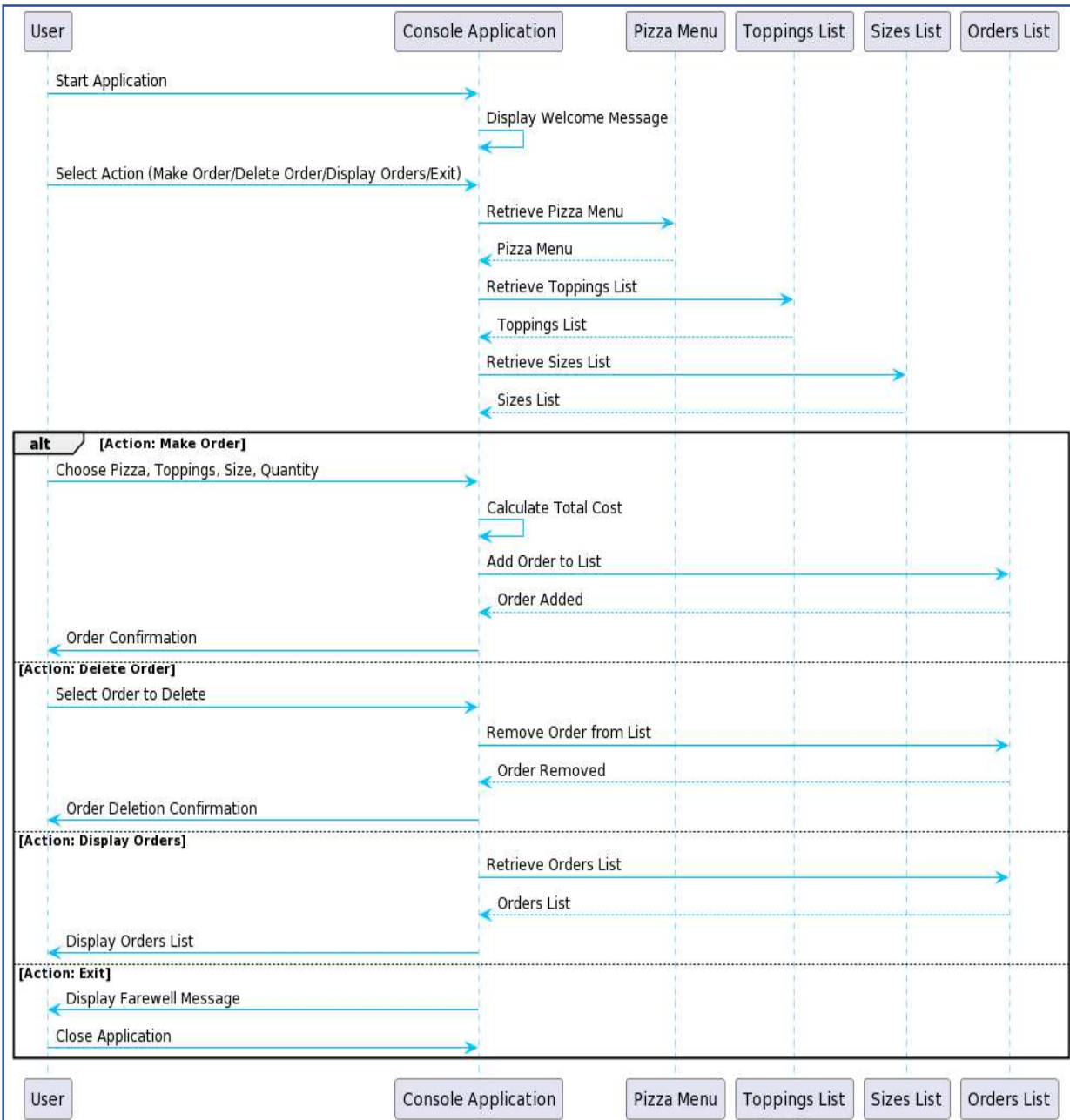
#### **Key Components:**

- Displays a list of all existing orders with relevant information (pizza, quantity, total cost).
- Retrieves order details from the stored data.
- Handles cases where no orders exist, providing appropriate feedback to the user.

## Use Case diagram



## Sequence diagram





## Testing

```
Microsoft Visual Studio Debu x + v
Welcome to Pizza Ordering System!

Menu:
1. Make Order
2. Delete Order
3. Display Orders
4. Exit
Enter your choice: a
Unhandled exception. System.FormatException: Input string was not in a correct format.
   at System.Number.ThrowOverflowOrFormatException(ParsingStatus status, TypeCode type)
   at System.Number.ParseInt32(ReadOnlySpan'1 value, NumberStyles styles, NumberFormatInfo info)
   at System.Int32.Parse(String s)
   at Program.Main(String[] args) in E:\All programs\VSC\ConsoleApp1\Pizza\Program.cs:line 59

E:\All programs\VSC\ConsoleApp1\Pizza\bin\Debug\net5.0\Pizza.exe (process 13760) exited with code 0.
To automatically close the console when debugging stops, enable Tools->Options->Debugging->Automatically close the console when debugging stops.

Press any key to close this window . . .
```

```
Menu:
1. Make Order
2. Delete Order
3. Display Orders
4. Exit
Enter your choice: 1

Menu:
Pizzas:
1. Margherita - Rs899
2. Pepperoni - Rs999
3. Vegetarian - Rs1099
4. Hawaiian - Rs1199

Enter the pizza ID and quantity you want to order (separated by commas):
1,1

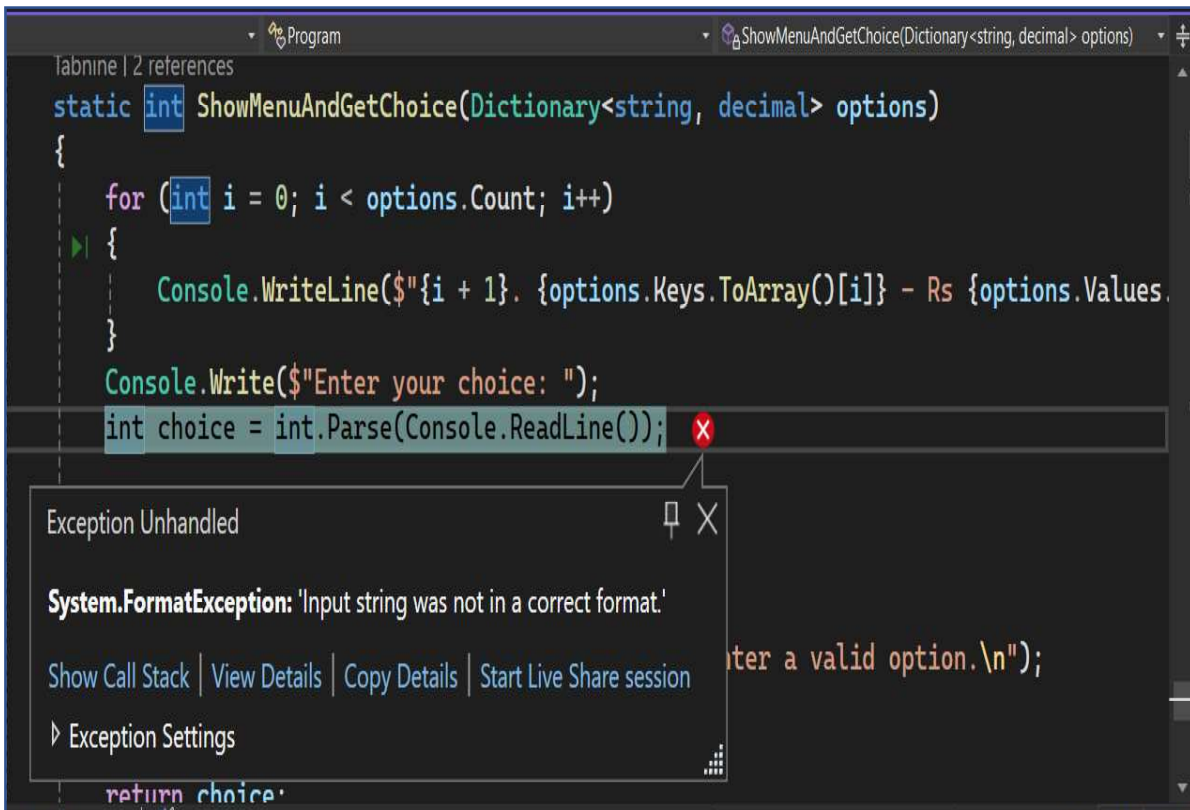
Your Order:
Pizza: Margherita, Quantity: 1
```

```

Your Order:
Pizza: Margherita, Quantity: 1
0. No toppings
1. Extra Cheese
2. Mushrooms
3. Peppers
4. Onions
5. Olives
Choose toppings (separated by commas):
1
Selected Toppings:
- Extra Cheese - Rs50

Choose size:
1. small - Rs 5
2. medium - Rs 20
3. large - Rs 40
Enter your choice: A
Unhandled exception. System.FormatException: Input string was not in a correct format.
   at System.Number.ThrowOverflowOrFormatException(ParsingStatus status, TypeCode type)
   at System.Number.ParseInt32(ReadOnlySpan`1 value, NumberStyles styles, NumberFormatInfo info)
   at System.Int32.Parse(String s)
   at Program.ShowMenuAndGetChoice(Dictionary`2 options) in E:\All programs\VSC\ConsoleApp1\Pizza\Program.cs:line 230
   at Program.MakeOrder(List`1 pizzas, List`1 toppings, Dictionary`2 sizes) in E:\All programs\VSC\ConsoleApp1\Pizza\Program.cs:line 149
   at Program.Main(String[] args) in E:\All programs\VSC\ConsoleApp1\Pizza\Program.cs:line 64

```



```

Program
ShowMenuAndGetChoice(Dictionary<string, decimal> options)
tabnine | 2 references
static int ShowMenuAndGetChoice(Dictionary<string, decimal> options)
{
    for (int i = 0; i < options.Count; i++)
    {
        Console.WriteLine($"{i + 1}. {options.Keys.ToArray()[i]} - Rs {options.Values.
    }
    Console.Write($"Enter your choice: ");
    int choice = int.Parse(Console.ReadLine());
    return choice;
}

```

Exception Unhandled

**System.FormatException:** 'Input string was not in a correct format.'

Show Call Stack | View Details | Copy Details | Start Live Share session

Exception Settings

## Code

```
using System;
using System.Collections.Generic;
using System.Linq;

class Pizza
{
    public string Name { get; set; }
    public decimal Price { get; set; }

    public Pizza(string name, decimal price)
    {
        Name = name;
        Price = price;
    }
}

class Program
{
    static List<string> orders = new List<string>();

    static void Main(string[] args)
    {
        // Creating a menu of pizzas
        List<Pizza> pizzas = new List<Pizza>
        {
            new Pizza("Margherita", 899m),
            new Pizza("Pepperoni", 999m),
            new Pizza("Vegetarian", 1099m),
            new Pizza("Hawaiian", 1199m)
        };

        // Creating a menu of toppings
        List<string> toppings = new List<string>
        {
            "Extra Cheese",
            "Mushrooms",
            "Peppers",
            "Onions",
            "Olives"
        };

        // Creating a menu of sizes
        Dictionary<string, decimal> sizes = new Dictionary<string, decimal>
        {
            { "small", 5m },
            { "medium", 20m },
            { "large", 40m }
        };
    }
}
```

```

Console.WriteLine("\nWelcome to Pizza Ordering System!");
bool exit = false;
while (!exit)
{
    Console.WriteLine("\nMenu:");
    Console.WriteLine("1. Make Order");
    Console.WriteLine("2. Delete Order");
    Console.WriteLine("3. Display Orders");
    Console.WriteLine("4. Exit");
    Console.Write("Enter your choice: ");
    int choice = int.Parse(Console.ReadLine());

    switch (choice)
    {
        case 1:
            MakeOrder(pizzas, toppings, sizes);
            break;
        case 2:
            DisplayOrders(pizzas);

            DeleteOrder();
            break;
        case 3:
            DisplayOrders(pizzas);
            break;
        case 4:
            exit = true;
            break;
        default:
            Console.WriteLine("Invalid choice. Please enter a number from 1 to 4.");
            break;
    }
}

Console.WriteLine("Thank you for ordering from us!");
}

static void MakeOrder(List<Pizza> pizzas, List<string> toppings, Dictionary<string,
decimal> sizes)
{
    // Displaying the menu
    Console.WriteLine("\n\nMenu:");
    Console.WriteLine("Pizzas:");
    for (int i = 0; i < pizzas.Count; i++)
    {
        Console.WriteLine($"{i + 1}. {pizzas[i].Name} - Rs{pizzas[i].Price}");
    }

    // Selecting pizzas to order

```

```

    Console.WriteLine("\nEnter the pizza ID and quantity you want to order (separated by
commas):");
    string input = Console.ReadLine();
    string[] selections = input.Split(',');

    // Confirming the order
    decimal totalCost = 0;
    Console.WriteLine("\nYour Order:");
    for (int i = 0; i < selections.Length; i += 2)
    {
        int index = int.Parse(selections[i].Trim()) - 1;
        int quantity = int.Parse(selections[i + 1].Trim());

        Console.WriteLine($"Pizza: {pizzas[index].Name}, Quantity: {quantity}");
        b:
        // Choosing toppings
        Console.WriteLine("0. No toppings");
        for (int j = 0; j < toppings.Count; j++)
        {
            Console.WriteLine($"{j + 1}. {toppings[j]}");
        }
        Console.WriteLine("Choose toppings (separated by commas):");
        string toppingInput = Console.ReadLine();
        string[] selectedToppings = toppingInput.Split(',');
        //Console.WriteLine("Selected Toppings:");
        string selectedTopping = "";
        foreach (var toppingIndexString in selectedToppings)
        {
            if (!string.IsNullOrEmpty(toppingIndexString) &&
int.TryParse(toppingIndexString.Trim(), out int toppingIndex))
            {
                if (toppingIndex == 0)
                {
                    Console.WriteLine("Selected Toppings:");
                    selectedTopping = "No Toppings";
                    Console.WriteLine($"- {selectedTopping}");
                }
                else if (toppingIndex >= 1 && toppingIndex <= toppings.Count)
                {
                    Console.WriteLine("Selected Toppings:");
                    selectedTopping = toppings[toppingIndex - 1];
                    Console.WriteLine($"- {selectedTopping} - Rs50");
                    totalCost += 50m * quantity;
                }
                else
                {
                    Console.WriteLine("Invalid choice. Please enter a valid option.\n");
                    goto b;
                }
            }
        }
    }

```

```

    }

    // Choosing size
    Console.WriteLine("\nChoose size:");
    int sizeChoice = ShowMenuAndGetChoice(sizes);
    string selectedSize = sizes.Keys.ToArray()[sizeChoice - 1];
    decimal sizeCost = sizes[selectedSize];
    Console.WriteLine($"Selected Size: {selectedSize}");

    totalCost += (pizzas[index].Price + sizeCost) * quantity;
    Console.WriteLine("\n");
    Console.WriteLine($"Pizza: {pizzas[index].Name}, Quantity: {quantity}, Size:
{selectedSize}, Toppings:");
    foreach (var toppingIndexString in selectedToppings)
    {
        if (!string.IsNullOrEmpty(toppingIndexString) &&
int.TryParse(toppingIndexString.Trim(), out int toppingIndex))
        {
            if (toppingIndex >= 1 && toppingIndex <= toppings.Count)
            {
                selectedTopping = toppings[toppingIndex-1];
                Console.WriteLine(selectedTopping + ", ");
            }
        }
    }
}
Console.WriteLine($"Total Cost: Rs.{totalCost} ");
Console.WriteLine("\nYour Order has been placed successfully.");
// Add the order to the orders list
orders.Add($"{selections[0]}, {selections[1]}, {totalCost}");

Console.WriteLine($"Please pay Rs.{totalCost} in cash upon delivery.");
}
static void DeleteOrder()
{
    if (orders.Count == 0)
    {
        Console.WriteLine("No orders exist.");
        return;
    }
    a:
    Console.WriteLine("\nEnter the index of the order to delete:");
    int index = int.Parse(Console.ReadLine());
    if (index >= 1 && index <= orders.Count)
    {
        orders.RemoveAt(index - 1); // Adjust index to match list index
        Console.WriteLine("Order deleted successfully.");
    }
}

```

```

        else
        {
            Console.WriteLine("Invalid index.\n");
            goto a;
        }
    }

static void DisplayOrders(List<Pizza> pizzas)
{
    if (orders.Count == 0)
    {
        Console.WriteLine("No orders exist.");
        return;
    }

    Console.WriteLine("\nOrders:");
    for (int i = 0; i < orders.Count; i++)
    {
        // Splitting the order string to extract pizza ID, quantity, and total cost
        string[] orderParts = orders[i].Split(',');
        int pizzaID = int.Parse(orderParts[0]);
        int quantity = int.Parse(orderParts[1]);
        decimal totalCost = decimal.Parse(orderParts[2]);
        // Getting the pizza name based on pizza ID
        Pizza orderedPizza = pizzas[pizzaID - 1];

        Console.WriteLine($"{i + 1}. Pizza: {orderedPizza.Name}, Quantity: {quantity}, Total
Cost: Rs.{totalCost} ");
    }
}

static int ShowMenuAndGetChoice(Dictionary<string, decimal> options)
{
    for (int i = 0; i < options.Count; i++)
    {
        Console.WriteLine($"{i + 1}. {options.Keys.ToArray()[i]} - Rs
{options.Values.ToArray()[i]}");
    }
    Console.Write("Enter your choice: ");
    int choice = int.Parse(Console.ReadLine());

    // Validate choice
    if (choice < 1 || choice > options.Count)
    {
        Console.WriteLine("Invalid choice. Please enter a valid option.\n");
        return ShowMenuAndGetChoice(options);
    }
    return choice;
}
}

```

## Output

```
Microsoft Visual Studio Debug Console

Welcome to Pizza Ordering System!

Menu:
1. Make Order
2. Delete Order
3. Display Orders
4. Exit
Enter your choice: 1

Menu:
Pizzas:
1. Margherita - Rs899
2. Pepperoni - Rs999
3. Vegetarian - Rs1099
4. Hawaiian - Rs1199

Enter the pizza ID and quantity you want to order (separated by commas):
1,1

Your Order:
Pizza: Margherita, Quantity: 1
0. No toppings
1. Extra Cheese
2. Mushrooms
3. Peppers
4. Onions
5. Olives
Choose toppings (separated by commas):
2
Selected Toppings:
- Mushrooms - Rs50

Choose size:
```

```
Microsoft Visual Studio Debug Console

Choose size:
1. small - Rs 5
2. medium - Rs 20
3. large - Rs 40
Enter your choice: 3
Selected Size: large

Pizza: Margherita, Quantity: 1, Size: large, Toppings:Mushrooms, Total Cost: Rs.989

Your Order has been placed successfully.
Please pay Rs.989 in cash upon delivery.
```



```
Menu:
1. Make Order
2. Delete Order
3. Display Orders
4. Exit
Enter your choice: 3

Orders:
1. Pizza: Margherita, Quantity: 1, Total Cost: Rs.989
2. Pizza: Vegetarian, Quantity: 1, Total Cost: Rs.1219
3. Pizza: Hawaiian, Quantity: 1, Total Cost: Rs.1339

Menu:
1. Make Order
2. Delete Order
3. Display Orders
4. Exit
Enter your choice: 2

Orders:
1. Pizza: Margherita, Quantity: 1, Total Cost: Rs.989
2. Pizza: Vegetarian, Quantity: 1, Total Cost: Rs.1219
3. Pizza: Hawaiian, Quantity: 1, Total Cost: Rs.1339

Enter the index of the order to delete:
2
Order deleted successfully.
```

```
Menu:
1. Make Order
2. Delete Order
3. Display Orders
4. Exit
Enter your choice: 3

Orders:
1. Pizza: Margherita, Quantity: 1, Total Cost: Rs.989
2. Pizza: Hawaiian, Quantity: 1, Total Cost: Rs.1339

Menu:
1. Make Order
2. Delete Order
3. Display Orders
4. Exit
Enter your choice: 4
Thank you for ordering from us!
```

## **Conclusion**

The implementation of the pizza ordering system represents a significant advancement in the way customers interact with food service establishments. By enabling users to select from various pizza options, sizes, and extra toppings with transparent pricing adjustments, the system offers a tailored ordering experience that aligns with modern consumer expectations. The ability to display and delete orders provides both customers and staff with greater control over the transaction process, ensuring that each order is accurate and up-to-date. In conclusion, this innovative approach not only enhances operational efficiency for businesses but also elevates customer satisfaction by offering a convenient and customizable ordering solution. As technology continues to evolve, such systems will become increasingly integral to the competitive landscape of the food service industry.

## **References**

1. C# 9.0 in a Nutshell: “The Definitive Reference” by Joseph Albahari and Ben Albahari.
2. “Head First C#” by Andrew Stellman and Jennifer Greene.
3. <https://itsourcecode.com/free-projects/csharp/pizza-ordering-system-in-c-with-source-code>
4. <https://www.youtube.com/watch?v=dz5-MeOjS0w>
5. <https://github.com/vvijayaraman0822/Pizza-Order-Application>