

LEGENDRY SANNINS MART

- MUHAIB SHAMSHER (CR-22029)
- MUNEEZA BADAR (CR-22022)
- MUHAMMAD HAMZA (CR-22048)

COURSE CODE: CT-260.

INSTRUCTOR: PROF.DR SHARIQ MAHMOOD KHAN
CYBERSECURITY

OBJECT ORIENTED PROGRAM

PROBLEM STATEMENT:

A system that allows customers to browse and purchase grocery items from the comfort of their homes.

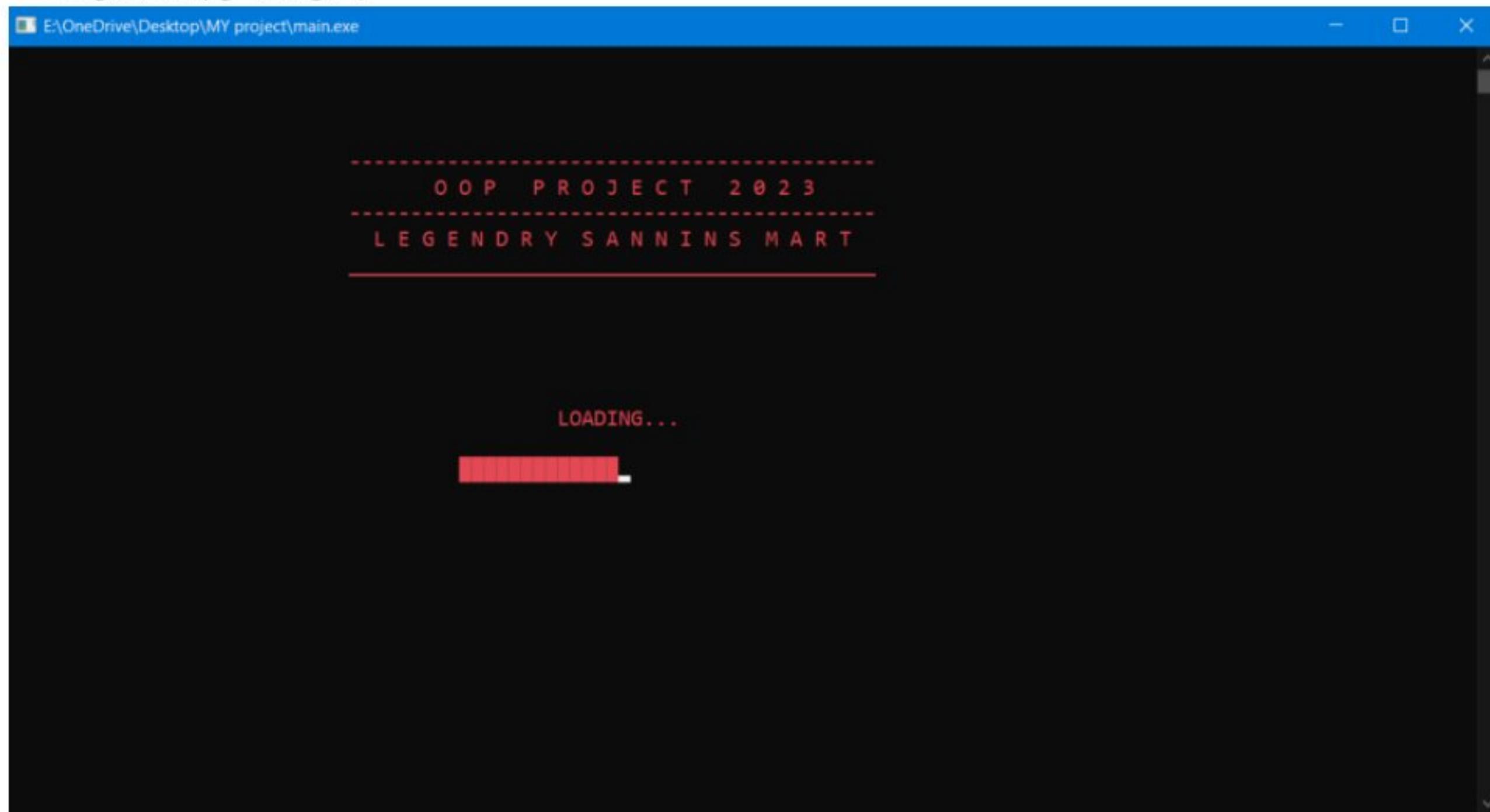
PROJECT DISCRIPTION:

The C++ OOP Online Grocery Mart project is a user-friendly and secure online platform for grocery shopping. The system incorporates key features such as a loading page, registration page, login page, menu page, and validation mechanisms. Upon launching the application, users are greeted with an engaging loading page. New users can register by providing their personal information through the registration page, which includes validation to ensure accurate data entry. Registered users can then securely log in through the login page using their username and password. After successful login, users are directed to the menu page where they can browse various grocery categories and view product details. Throughout the system, input validation is implemented to ensure data integrity and prevent errors. This project aims to provide a seamless and efficient online shopping experience for customers.

STEPS OF SOLUTION:

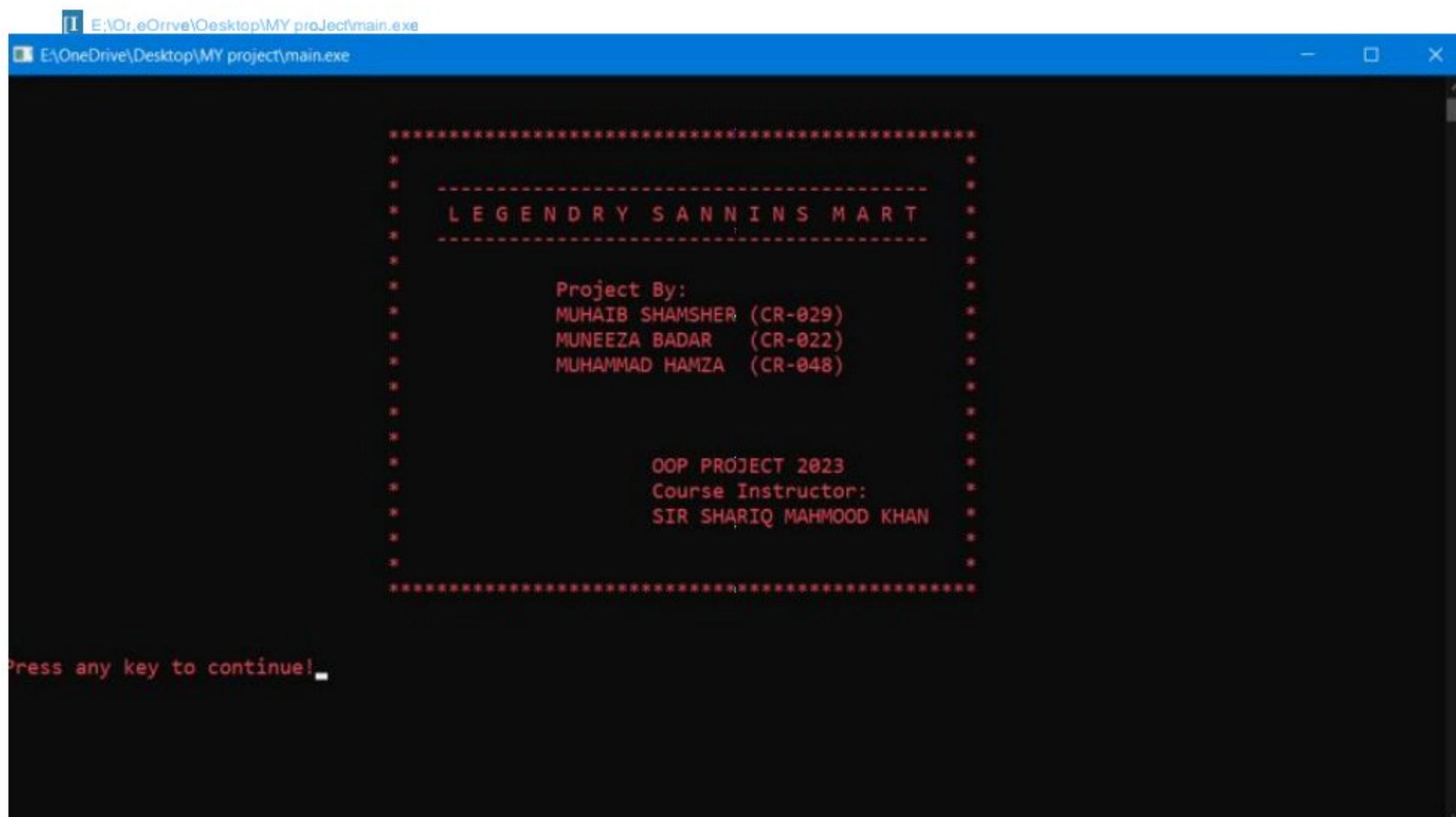
FEATURES OF PROJECT:

LOADING PAGE:



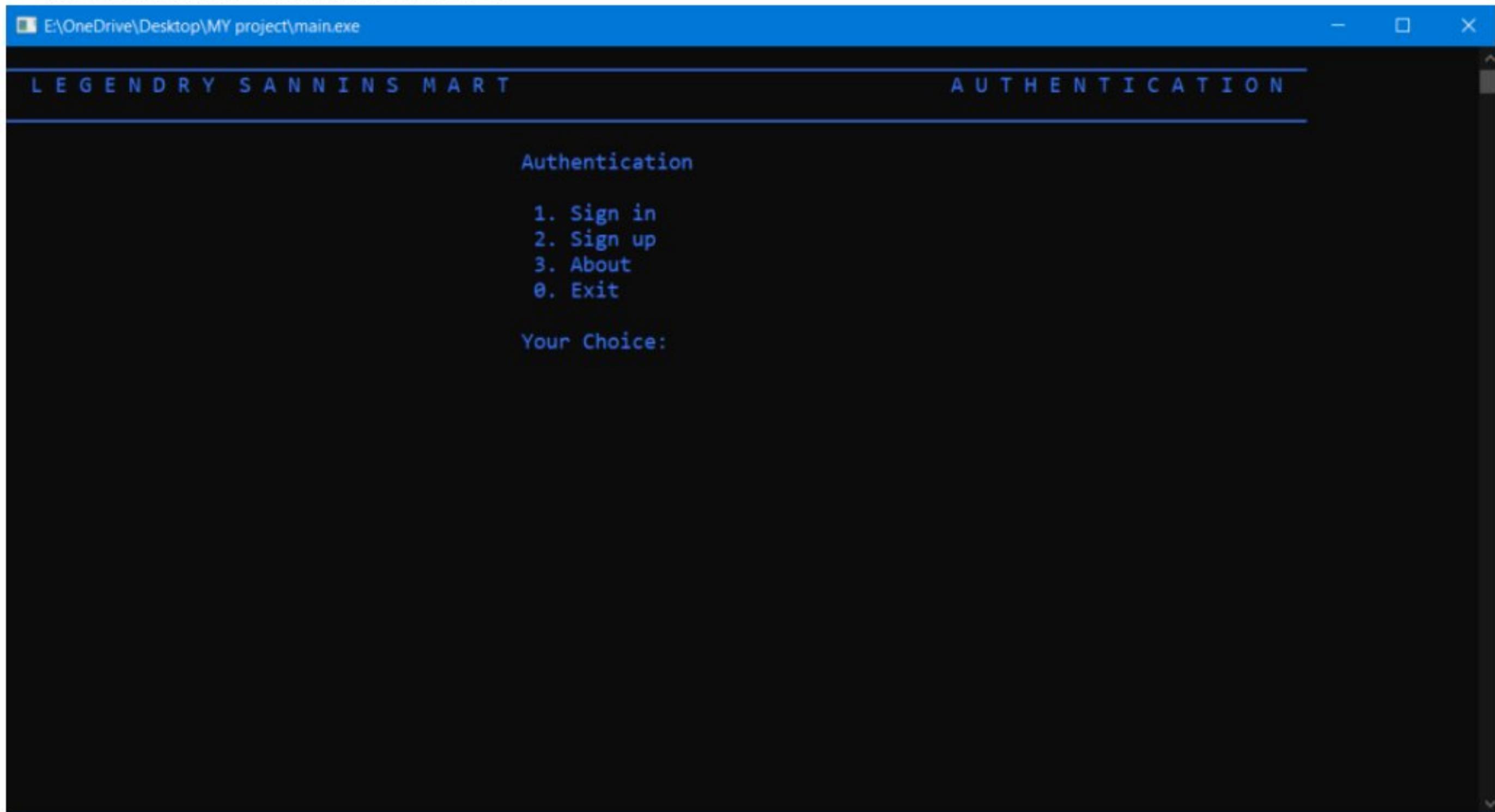
The system greets the user with a visually appealing loading page upon launching the application. It displays a progress indicator or animation to indicate that the system is initializing.

MAIN PAGE:



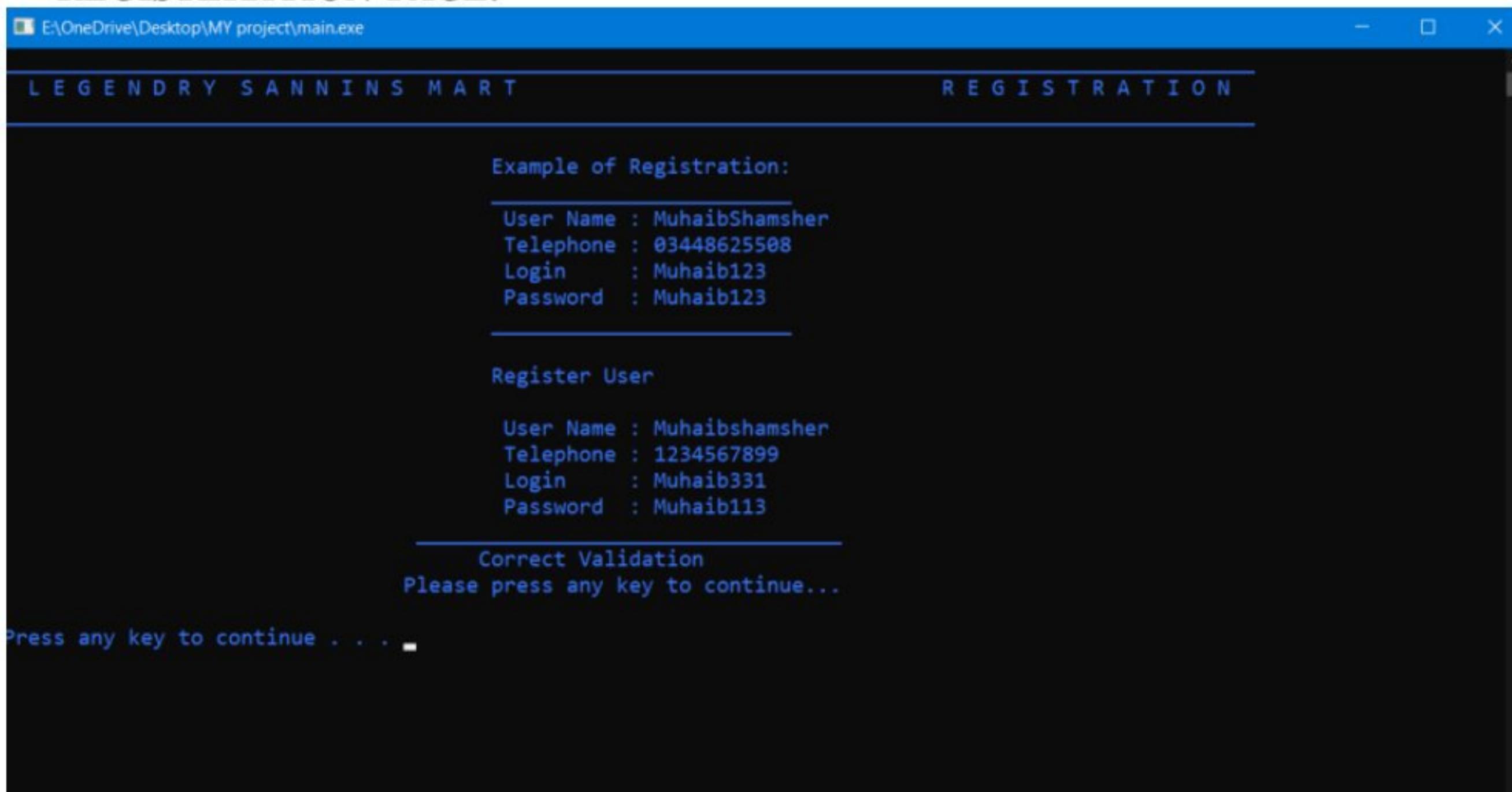
After loading it shows main page and ask user to press any key to continue.

AUTHENTICATION PAGE:

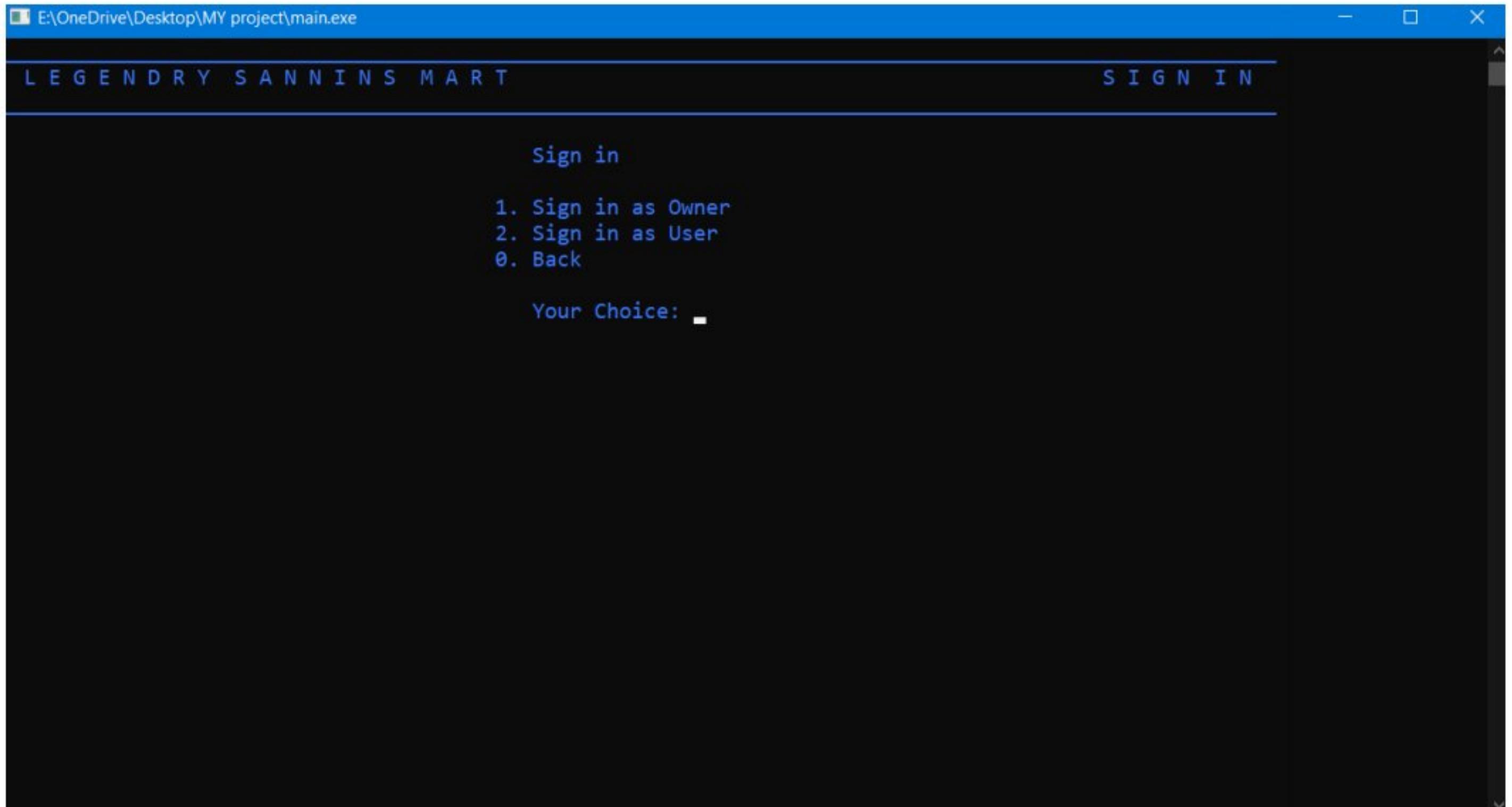


Authentication page ask user weather his account is authenticated or want to register himself. About section will open the main page.

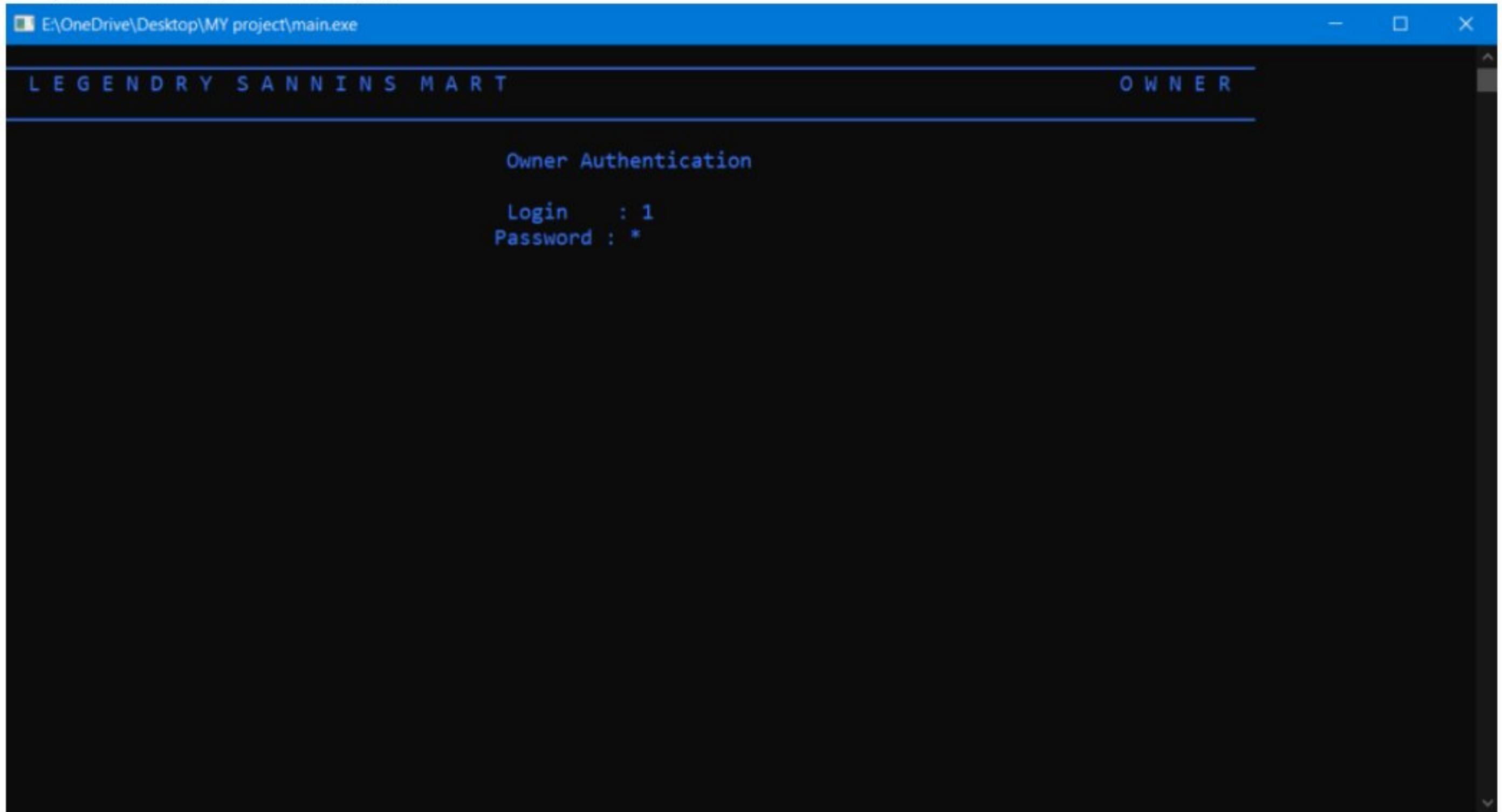
REGISTERATION PAGE:



Customers can register for an account by providing their necessary details such as name, contact number, and password. The registration page should include validation checks to ensure the accuracy and completeness of user input.

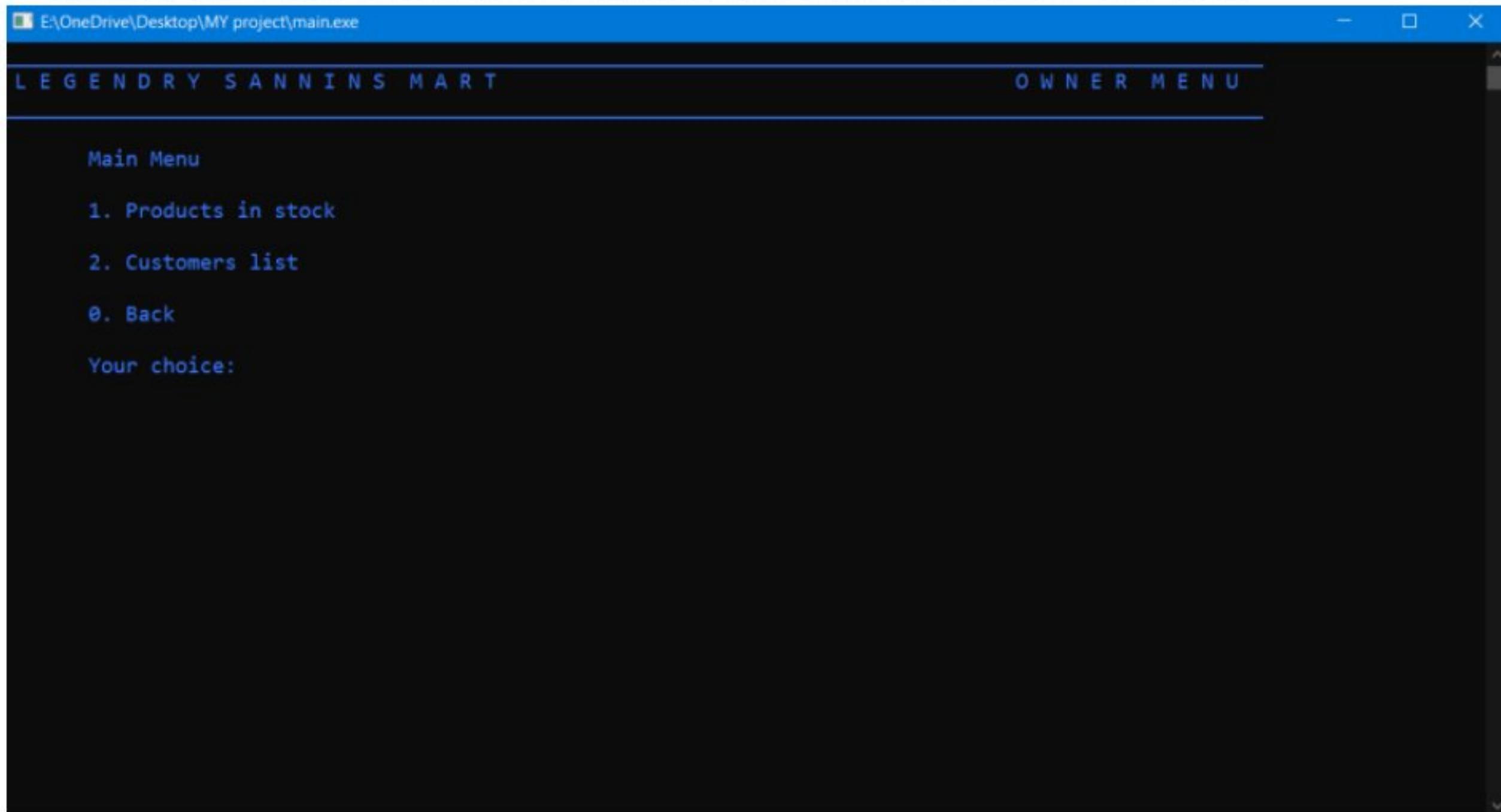
SIGN IN PAGE:

This page ask to sign in as owner or user and you also have a choice to go back.

SIGN IN AS OWNER:

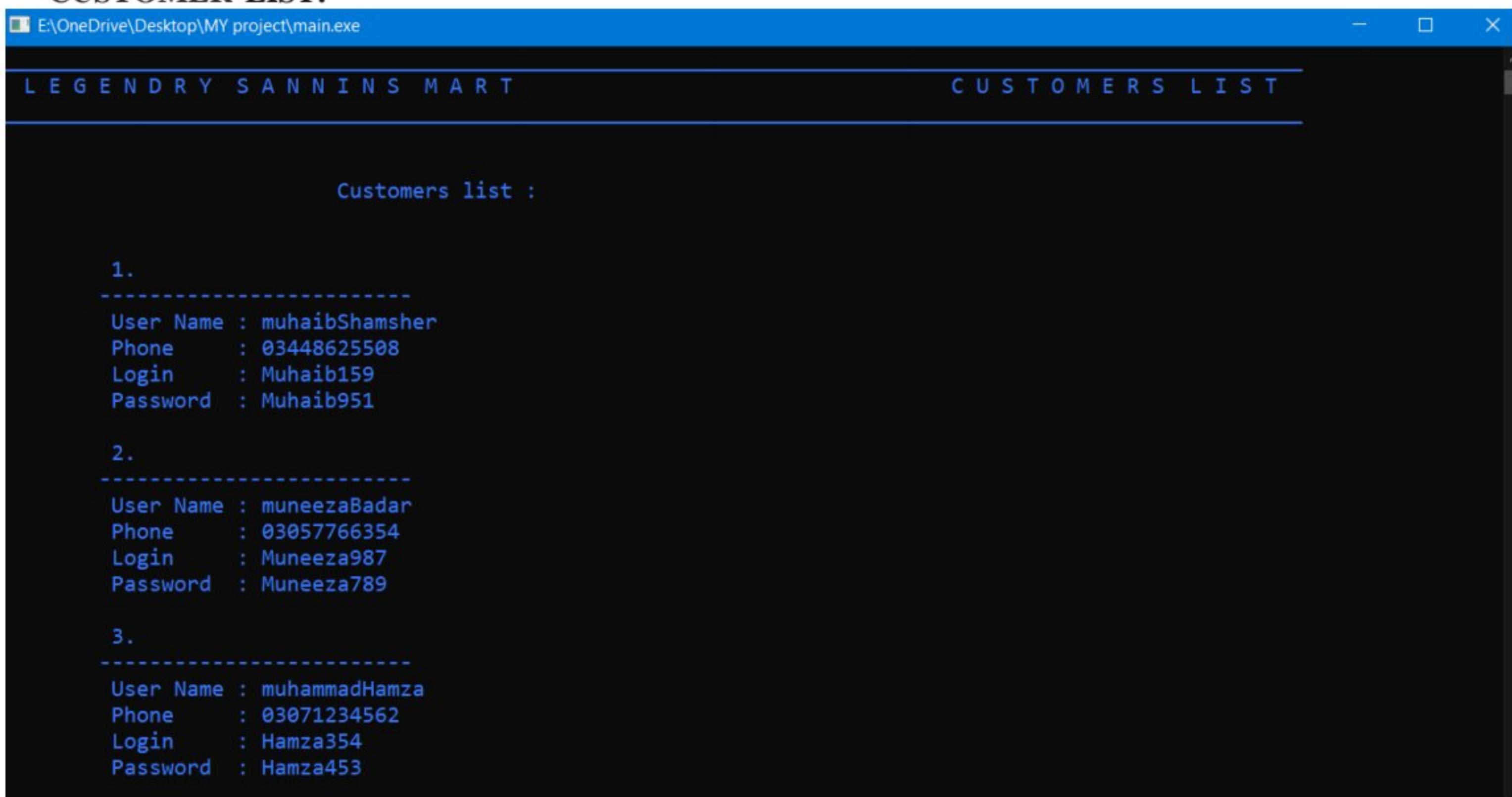
The project includes a separate login page for the owner/administrator of the online grocery mart to access the management features.

OWNER'S PAGE FOR CUSTOMER LIST AND PRODUCT STOCK:



After successful owner login, a page is provided to display the list of registered customers and their relevant information.

CUSTOMER LIST:



The owner can view and manage customer accounts, such as updating customer details or deactivating accounts.

PRODUCT STOCK:

LEGENDRY SANNINS MART		PRODUCTS IN STOCK	
Products List	Category	Price	In Stock
1. Potatoes	Vegetables & Fruits	65	25
2. Carrots	Vegetables & Fruits	70	25
3. Onion	Vegetables & Fruits	120	25
4. Water	Water & Beverages	60	100
5. Pepsi	Water & Beverages	75	100
6. Coca Cola	Water & Beverages	78	100
7. Pizza	Fast Food Products	1800	20
8. Burger	Fast Food Products	900	20
9. Potatoe Fries	Fast Food Products	230	20

0. Back
Make changes in: -

The page also presents the current stock of grocery items available for purchase.

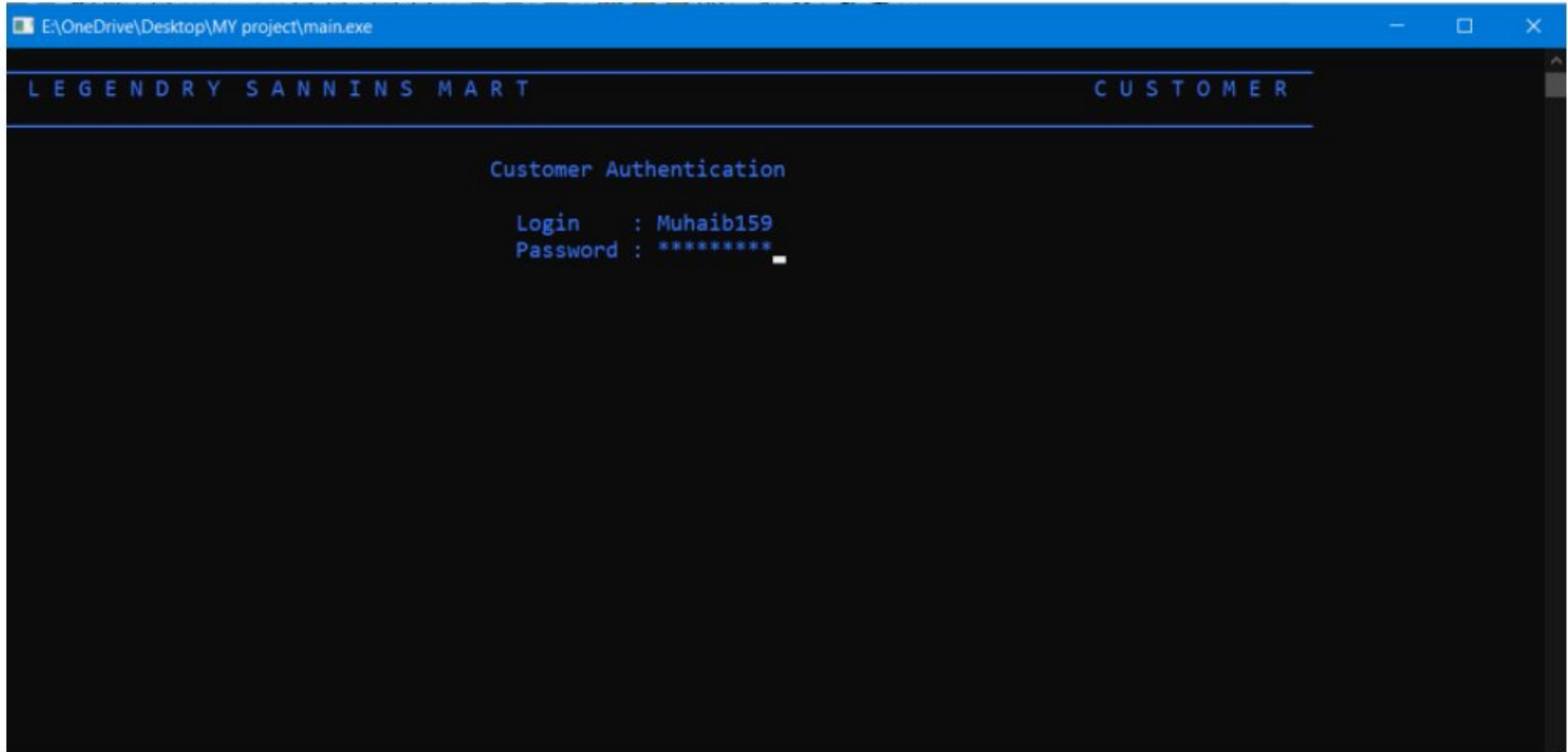
OWNER'S PAGE TO MAKE CHANGES IN PRODUCT DETAILS:

LEGENDRY SANNINS MART			
Products List	Category	Price	In Stock
1. Potatoes	Vegetables & Fruits	65	25

1. Change the price
2. Change the quantity in storage
3. Change the name of product
4. Change the class of product
0. Go back

Your choice :

The owner has the authority to modify product information, such as changing the price, quantity, name, or category of grocery items.

LOGIN AS A CUSTOMER:

Registered customers can log in using their username and password. The login page includes appropriate validation and error handling to ensure a secure login process.

CATEGORIES PAGE:

PRODUCT 1	
Products List	Category
1. Potatoes	Vegetables & Fruits
2. Carrots	Vegetables & Fruits
3. Onion	Vegetables & Fruits
0. Back	120
Your choice:	In Stock
	25
	25
	25

This page allows customer to choose their categories for shopping.

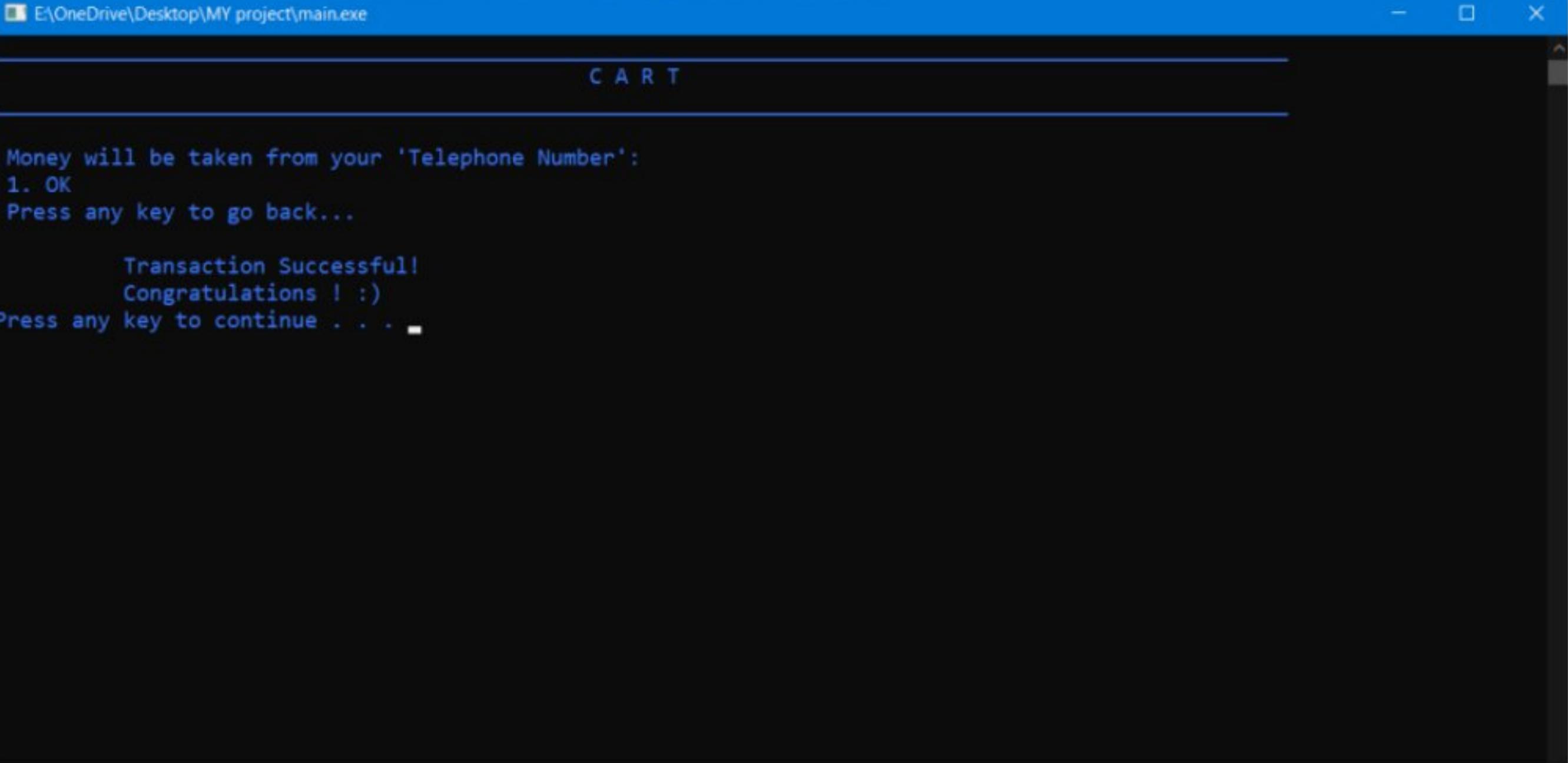
MENU PAGE:

LEGENDRY SANNINS MART		PRODUCT 3	
Products List	Category	Price	In Stock
1. Pizza	Fast Food Products	1800	17
2. Burger	Fast Food Products	900	20
3. Potatoe Fries	Fast Food Products	230	20
0. Back			
Your choice: -			

LEGENDRY SANNINS MART		PRODUCT 2	
Products List	Category	Price	In Stock
4. Water	Water & Beverages	60	100
5. Pepsi	Water & Beverages	75	100
6. Coca Cola	Water & Beverages	78	100
0. Back			
Your choice: -			

LEGENDRY SANNINS MART		PRODUCT 1	
Products List	Category	Price	In Stock
1. Potatoes	Vegetables & Fruits	66	25
2. Carrots	Vegetables & Fruits	70	25
3. Onion	Vegetables & Fruits	120	25
0. Back			
Your choice: -			

Upon successful login, customers are presented with a menu page displaying various grocery item categories such as fast food, vegetables, water & beverages. Customers can browse through different categories and select items to view their details and add them to the shopping cart.

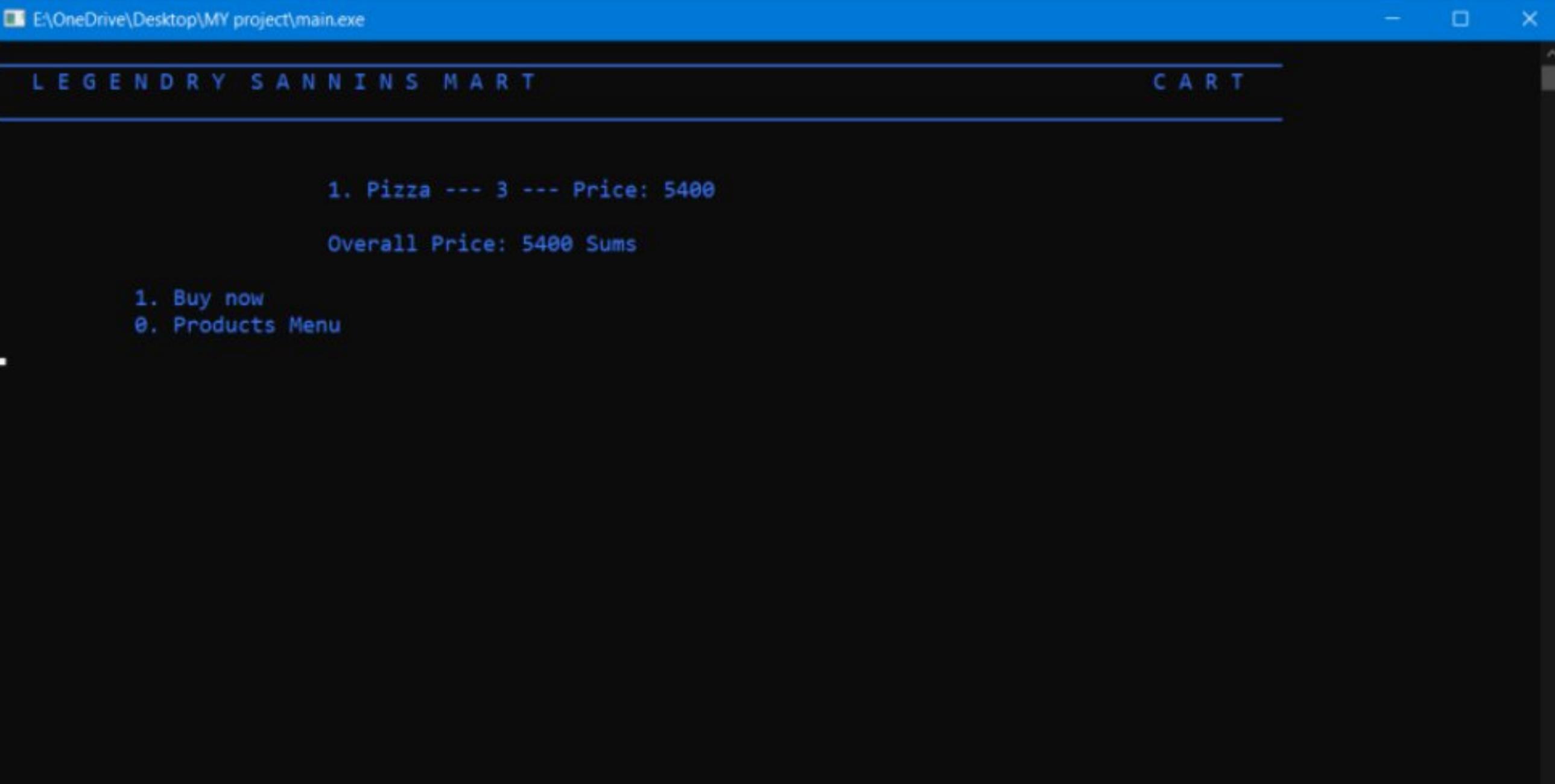
CART:

```
E:\OneDrive\Desktop\MY project\main.exe
C A R T

Money will be taken from your 'Telephone Number':
1. OK
Press any key to go back...

Transaction Successful!
Congratulations ! :)
Press any key to continue . . .
```

A shopping cart feature that allows customers to add items, update quantities, and remove products before proceeding to checkout. The cart will store selected items and calculate the total amount to be paid.

TRANSACTION PAGE:

```
E:\OneDrive\Desktop\MY project\main.exe
LEGENDRY SANNINS MART          C A R T

1. Pizza --- 3 --- Price: 5400
Overall Price: 5400 Sums

1. Buy now
0. Products Menu
```

A secure transaction page that supports telephone number payment method, and ask customer that he is ok with the method or not, customer can go back to the cart .If the customer allow the transaction method then the screen will prompt "transaction successful" message.

CODE SCREENSHOT:

HEADER FILES:

LOADING PAGE:

```
1 #include<iostream>
2 #include <ctime>      // Time
3 #include <windows.h> // Loading
4 #pragma once
5
6 using namespace std;
7
8 void gotoXY(int m, int n) {
9     // 'COORD' is a built in function for positioning the objects
10    COORD d;
11    d.X = m;
12    d.Y = n;
13    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), d);
14 }
15
16 void Loading()
17 {
18     cout << "\n\n\n\n";
19     cout << "\t\t\t----- \n";
20     cout << "\t\t\tO O P   P R O J E C T   2 0 2 3   \n";
21     cout << "\t\t\t----- \n";
22     cout << "\t\t\tL E G E N D R Y   S A N N I N S   M A R T   \n";
23     cout << "\t\t\t----- \n";
24
25     char a = 219;
26     gotoXY(45, 14);
27
28     cout << "LOADING... " << endl;
29
30     gotoXY(37, 16);
31     for (int r = 1; r <= 26; r++)
32     {
33         //for speed
34         for (int speed = 0; speed <= 99000000; speed++)
35             cout << a;
36     }
37     cout << endl;
38 }
```

PASSWORD:

```
1 #pragma once
2 #include <iostream>
3 #include <string>
4 #include <bits/stdc++.h>
5 #include <conio.h>
6
7 using namespace std;
8
9 // Global variables
10
11
12 string Function_Password_Val()
13 {
14     string Password_Val = "";
15
16     cout << "\t\t\t\t\t\tPassword : ";
17     char ch;
18     while ((ch = getch()) != 13)
19     {
20         if(ch!=8){
21             Password_Val += ch;
22             cout<<"*";
23         }
24         else if(ch == 8 ){
25             putchar('\b');
26             putchar(' ');
27             putchar('\b');
28             Password_Val = Password_Val.substr(0,Password_Val.length()-1);
29         }
30     }
31
32     return Password_Val;
33 }
```

PERSON:

```
 1 #pragma once
 2 #include <iostream>
 3 using namespace std;
 4
 5 class Person {
 6 protected:
 7     string name;
 8     string tell;
 9
10 public:
11     Person(string name, string tell) {
12         this->name = name;
13         this->tell = tell;
14     }
15
16     virtual void Display() = 0;
17 };
18
19 class Security : public Person {
20 protected:
21     string login;
22     string password;
23
24 public:
25     Security(string name, string tell, string login, string password) : Person(name, tell) {
26         this->login = login;
27         this->password = password;
28     }
29 };
30
31 class User : public Security {
32 public:
33     // Product brought by the
34     int Product1_1_User = 0, Product1_2_User = 0, Product1_3_User = 0;
35     int Product2_1_User = 0, Product2_2_User = 0, Product2_3_User = 0;
36     int Product3_1_User = 0, Product3_2_User = 0, Product3_3_User = 0;
37
38     User(string name, string tell, string login, string password) : Security(name, tell, login, password) {}
39
40     string getName() {
41         return name;
42     }
43
44     string getTell() {
45         return tell;
46     }
47
48     string getLogin() {
49         return login;
50     }
51
52     string getPassword() {
53         return password;
54     }
55
56     void Display() {
57         cout << "\t\tName : " << name << endl;
58         cout << "\t\tTell : " << tell << endl;
59         cout << "\t\tLogin : " << login << endl;
60         cout << "\t\tPassword : " << password << endl;
61     }
62 };
```

MENU:

```
1 #include<iostream>
2 #include<conio.h>
3 using namespace std;
4
5 #include "LoadingPage.h"
6
7 void welcome()
8 {
9     cout<<endl<<endl;
10    cout<<"\t\t\t\t*****\n";
11    cout<<"\t\t\t\t*\n";
12    cout<<"\t\t\t\t-----\n";
13    cout<<"\t\t\t\t L E G E N D R Y S A N N I N S M A R T \n";
14    cout<<"\t\t\t\t-----\n";
15    cout<<"\t\t\t\t*\n";
16    cout<<"\t\t\t\t Project By:\n";
17    cout<<"\t\t\t\t MUHAIB SHAMSHER (CR-029)\n";
18    cout<<"\t\t\t\t MUNEEZA BADAR (CR-022)\n";
19    cout<<"\t\t\t\t MUHAMMAD HAMZA (CR-048)\n";
20    cout<<"\t\t\t\t *\n";
21    cout<<"\t\t\t\t *\n";
22    cout<<"\t\t\t\t *\n";
23    cout<<"\t\t\t\t OOP PROJECT 2023\n";
24    cout<<"\t\t\t\t Course Instructor:\n";
25    cout<<"\t\t\t\t SIR SHARIQ MAHMOOD KHAN\n";
26    cout<<"\t\t\t\t *\n";
27    cout<<"\t\t\t\t *\n";
28    cout<<"\t\t\t\t *****\n";
29 }
30
31
32 void clear()
33 {
34     cout << "Press any key to continue!";
35     getch();
36     system("CLS");
37 }
38
39
40 void start() {
41     Loading();
42     system("cls");
43     welcome();
44     clear();
45 }
```

PRODUCT:

```
1  #pragma once
2  #include <iostream>
3  #include <string>
4  #include<iomanip>
5
6  using namespace std;
7
8  int position1 = 1; // for numbering the products
9
10 class Products {
11     public:
12         string name;
13         string class_of_products;
14         double price;
15
16     public:
17         Products(string name, string class_of_products, double price) {
18             this->name = name;
19             this->class_of_products = class_of_products;
20             this->price = price;
21         }
22
23         virtual void Display() = 0;
24     };
25
26
27 class Class_Of_Products1 : public Products {
28     public:
29         int quantity;
30
31     public:
32         Class_Of_Products1(string name, string class_of_products, double price, int quantity):
33             Products(name, class_of_products, price)
34         {
35             this->quantity = quantity;
36         }
37
38         string getName() {
39             return name;
40         }
41
42         string getClassOfProducts() {
43             return class_of_products;
44         }
45
46         double getPrice() {
47             return price;
48         }
49
50         int getQuantity() {
51             return quantity;
52         }
53
54         void Display() {
55             cout << " " << position1 << ". " << left << setw(32)
56                         << name << left << setw(27)
57                         << class_of_products << left << setw(18)
58                         << price << quantity << endl;
59
60             position1++;
61         }
62     };
63 }
```

```
64
65 class Class_Of_Products2 : public Products {
66     public:
67         int quantity;
68
69     public:
70         Class_Of_Products2(string name, string class_of_products, double price, int quantity) : Products(name, class_of_products, price) {
71             this->quantity = quantity;
72         }
73
74         string getName() {
75             return name;
76         }
77
78         string getClassOfProducts() {
79             return class_of_products;
80         }
81
82         double getPrice() {
83             return price;
84         }
85
86         int getQuantity() {
87             return quantity;
88         }
89
90         void Display() {
91             cout << " " << position1 << ". " << left << setw(32) << name << left << setw(27) << class_of_products << left << setw(18) << price << quantity << endl;
92             position1++;
93         }
94     };
95
96
97 class Class_Of_Products3 : public Products {
98     public:
99         int quantity;
100
101    public:
102        Class_Of_Products3(string name, string class_of_products, double price, int quantity) : Products(name, class_of_products, price) {
103            this->quantity = quantity;
104        }
105
106        string getName() {
107            return name;
108        }
109
110        string getClassOfProducts() {
111            return class_of_products;
112        }
113
114        double getPrice() {
115            return price;
116        }
117
118        int getQuantity() {
119            return quantity;
120        }
121
122        void Display() {
123            cout << " " << position1 << ". " << left << setw(32) << name << left << setw(27) << class_of_products << left << setw(18) << price << quantity << endl;
124            position1++;
125        }
126    };
127 }
```

VALIDATION:

```

1 #include <iostream>
2 #include <string>
3 #include <ctype.h>
4 using namespace std;
5
6 // Global Identifiers for validation
7 int Num_Upper = 0, Num_Lower = 0, Num_Number = 0, validation = 0;
8
9 // Identifiers for File Handling
10 string Name_Memory, Login_Memory, Password_Memory, TellNum_Memory;
11
12
13 class Validation {
14 private:
15     string Name, Login, Password, TellNum;
16
17 public:
18     Validation(): Name(" "), Login(" "), Password(" "), TellNum(" ") {}
19
20     void Display() {
21         cout << "\t\tUser Information :" << endl << endl;
22         cout << "\t\tUser Name : " << Name << endl;
23         cout << "\t\tTelephone : " << TellNum << endl;
24         cout << "\t\tLogin : " << Login << endl;
25         cout << "\t\tPassword : " << Password << endl;
26     }
27
28     void SetUser() {
29         cout << "\t\t\tRegister User " << endl << endl;
30         cout << "\t\t\tUser Name : " ; cin >> Name;
31         cout << "\t\t\tTelephone : " ; cin >> TellNum;
32         cout << "\t\t\tLogin : " ; cin >> Login;
33         cout << "\t\t\tPassword : " ; cin >> Password;
34
35         // Info sended to Memory
36         Name_Memory = Name;
37         Password_Memory = Password;
38         Login_Memory = Login;
39         TellNum_Memory = TellNum;
40     }
41
42     //Validation Check for Password.
43     friend void ValidationPassword(Validation User);
44
45     //Validation Check for Name
46     friend void ValidationName(Validation User);
47
48     //Validation Check for TellNum
49     friend void ValidationTellNum(Validation User);
50 };

```

```
51
52
53 //Validation Check for Password.
54 void ValidationPassword(Validation User)
55 {
56     if (User.Password.length() >= 6 && User.Password.length() <= 15)
57     {
58         for (int i = 0; i < User.Password.length(); i++) {
59             if (isupper(User.Password[i])) {
60                 Num_Upper += 1;
61             }
62             if (islower(User.Password[i])) {
63                 Num_Lower += 1;
64             }
65             if (isdigit(User.Password[i])) {
66                 Num_Number += 1;
67             }
68         }
69
70         if (Num_Upper >= 1 && Num_Upper <= 10 && Num_Lower >= 4 && Num_Lower <= 10 && Num_Number >= 2 && Num_Number <= 10) {
71             validation++;
72             Num_Upper = 0; Num_Lower = 0; Num_Number = 0;
73         }
74     }
75     else {
76         Num_Upper = 0; Num_Lower = 0; Num_Number = 0;
77     }
78 }
79
80 else {
81     Num_Upper = 0; Num_Lower = 0; Num_Number = 0;
82 }
83 }
84
85 //Validation Check for Name
86 void ValidationName(Validation User)
87 {
88     if (User.Name.length() >= 1 && User.Name.length() <= 15)
89     {
90         for (int i = 0; i < User.Name.length(); i++) {
91             if (isupper(User.Name[i])) {
92                 Num_Upper += 1;
93             }
94             if (islower(User.Name[i])) {
95                 Num_Lower += 1;
96             }
97             if (isdigit(User.Name[i])) {
98                 Num_Number += 1;
99             }
100        }
101    }
```

```
101     if (Num_Upper <= 1 && Num_Lower >= 1 && Num_Lower <= 14 && Num_Number == 0) {
102         validation++;
103         Num_Upper = 0; Num_Lower = 0; Num_Number = 0;
104     }
105
106     else {
107         Num_Upper = 0; Num_Lower = 0; Num_Number = 0;
108     }
109 }
110
111 else {
112     Num_Upper = 0; Num_Lower = 0; Num_Number = 0;
113 }
114
115 }
116
117 //Validation Check for TellNum
118 void ValidationTellNum(Validation User)
119 {
120     if (User.TellNum.length() >= 9 && User.TellNum.length() <= 12) {
121         for (int i = 0; i < User.TellNum.length(); i++) {
122             if (isupper(User.TellNum[i])) {
123                 Num_Upper += 1;
124             }
125             if (islower(User.TellNum[i])) {
126                 Num_Lower += 1;
127             }
128             if (isdigit(User.TellNum[i])) {
129                 Num_Number += 1;
130             }
131         }
132
133         if (Num_Upper == 0 && Num_Lower == 0 && Num_Number >= 9 && Num_Number <= 12) {
134             validation++;
135             Num_Upper = 0; Num_Lower = 0; Num_Number = 0;
136         }
137     }
138
139     else {
140         Num_Upper = 0; Num_Lower = 0; Num_Number = 0;
141     }
142 }
```

MAIN CODE:

```

1 #include <iostream>
2 #include <string>
3 #include <fstream>
4 using namespace std;
5
6 // Header Files
7 #include "menu.h"
8 #include "Validation.h"
9 #include "Sign_Password.h"
10 #include "Products.h"
11 #include "person.h"
12
13 // Classes
14 Validation User_Validation;
15 User user(Name_Memory, TellNum_Memory, Login_Memory, Password_Memory);
16
17 Class_Of_Products1 Product1_1("Potatoes", "Vegetables & Fruits", 65.0, 25);
18 Class_Of_Products1 Product1_2("Carrots", "Vegetables & Fruits", 70.0, 25);
19 Class_Of_Products1 Product1_3("Onion", "Vegetables & Fruits", 120.0, 25);
20
21 Class_Of_Products2 Product2_1("Water", "Water & Beverages", 60.0, 100);
22 Class_Of_Products2 Product2_2("Pepsi", "Water & Beverages", 75.0, 100);
23 Class_Of_Products2 Product2_3("Coca Cola", "Water & Beverages", 78.0, 100);
24
25 Class_Of_Products3 Product3_1("Pizza", "Fast Food Products", 1800.0, 20);
26 Class_Of_Products3 Product3_2("Burger", "Fast Food Products", 900.0, 20);
27 Class_Of_Products3 Product3_3("Potatoe Fries", "Fast Food Products", 230.0, 20);
28
29
30 class LegendrySanninsMart {
31     private:
32         // Password and login for Owner
33         string Owner_Login_Sign = "1";
34         string Owner_Password_Sign = "1";
35
36     protected:
37         // Functions
38         void Sign_in();
39         void About();
40         void Owner_Main_Menu();
41         void Owner_Products_Stotage();
42         void Modify_Info_Products(int take_product_location);
43         void Owner_Customers_List();
44         void User_Main_Menu();
45         void Table_For_Increasing_And_Decreasing();
46         void Class_Of_Products1_Menu();
47         void Class_Of_Products2_Menu();
48         void Class_Of_Products3_Menu();
49         void Cart_Check();
50
51     public:
52         void General_Menu();
53 };
54

```

```
65
66 void LegendarySanninsMart :: General_Menu()
67 {
68     for (int i = 0; i < 1000; i++)
69     {
70         system("cls");
71         cout << " _____ AUTHENTICATION \n";
72         cout << " LEGENDRY SANNINS MART \n";
73         cout << " _____ \n";
74         cout << "\t\t\t Authentication \n";
75         |<< endl;
76         cout << "\t\t\t 1. Sign in" << endl;
77         cout << "\t\t\t 2. Sign up" << endl;
78         cout << "\t\t\t 3. About" << endl;
79         cout << "\t\t\t 0. Exit" << endl
80         |<< endl;
81         cout << "\t\t\t Your Choice: ";
82
83         switch (getch())
84         {
85             case 49:
86                 // Sign in
87                 system("cls");
88                 Sign_in();
89                 system("pause");
90                 break;
91
92             case 50:
93                 // Register
94                 // Set details with validation
95                 for (int i = 0; i != 1)
96                 {
97                     system("cls");
98                     cout << " _____ REGISTRATION \n";
99                     cout << " LEGENDRY SANNINS MART \n";
100                     cout << " _____ \n";
101                     cout << "\t\t\t Example of Registration: " << endl;
102                     cout << "\t\t\t _____" << endl;
103                     cout << "\t\t\t User Name : MuhaibShamsher " << endl;
104                     cout << "\t\t\t Telephone : 03448625508 " << endl;
105                     cout << "\t\t\t Login : Muhaib123 " << endl;
106                     cout << "\t\t\t Password : Muhaib123 " << endl;
107                     cout << "\t\t\t _____" << endl
108                     |<< endl;
109
110                     User_Validation.SetUser();
111                     ValidationPassword(User_Validation);
112                     ValidationName(User_Validation);
113                     ValidationTellNum(User_Validation);
114
115         }
116     }
117 }
```



```

183 void LegendrySanninsMart :: About()
184 {
185     cout << endl
186     << endl;
187     cout << "\t\t\t\t*****\n";
188     cout << "\t\t\t\t"           "\n";
189     cout << "\t\t\t\t-----\n";
190     cout << "\t\t\t\t" LEGENDRY SANNINS MART "\n";
191     cout << "\t\t\t\t-----\n";
192     cout << "\t\t\t\t"           "\n";
193     cout << "\t\t\t\t"           "\n";
194     cout << "\t\t\t\t" Project By:           "\n";
195     cout << "\t\t\t\t" MUHAIB SHAMSHER (CR-029) "\n";
196     cout << "\t\t\t\t" MUNEEZA BADAR (CR-022) "\n";
197     cout << "\t\t\t\t" MUHAMMAD HAMZA (CR-048) "\n";
198     cout << "\t\t\t\t"           "\n";
199     cout << "\t\t\t\t"           "\n";
200     cout << "\t\t\t\t" OOP PROJECT 2023 "\n";
201     cout << "\t\t\t\t" Course Instructor: "\n";
202     cout << "\t\t\t\t" SIR SHARIQ MAHMOOD KHAN "\n";
203     cout << "\t\t\t\t"           "\n";
204     cout << "\t\t\t\t"           "\n";
205     cout << "\t\t\t\t*****\n";
206
207     cout << "\n\t\t\t\tPress any key to go back to Menu \n\n";
208     system("pause");
209 }
210 // DONE
211
212 void LegendrySanninsMart :: Sign_in()
213 {
214     for (int i = 0; i < 1000; i++)
215     {
216         string Login_Sign, Password_Sign;
217
218         system("cls");
219         cout << "_____\n";
220         cout << " LEGENDRY SANNINS MART           SIGN IN \n";
221         cout << "_____\n";
222         cout << "\t\t\t\t Sign in\n\n";
223         cout << "\t\t\t\t1. Sign in as Owner" << endl;
224         cout << "\t\t\t\t2. Sign in as User" << endl;
225         cout << "\t\t\t\t0. Back" << endl;
226         cout << endl;
227         cout << "\t\t\t\t Your Choice: ";
228
229         switch (getch())
230         {
231             case 49:
232                 // Sign in as Owner
233                 system("cls");
234                 cout << "_____\n";
235                 cout << " LEGENDRY SANNINS MART           OWNER \n";
236                 cout << "_____\n";
237
238                 cout << "\t\t\t\t1. Owner Authentication\n\n";
239                 cout << "\t\t\t\tLogin   : ";
240                 cin >> Login_Sign;
241
242                 Password_Sign = Function_Password_Val(); // the function will take Password as input
243
244                 if (Login_Sign == Owner_Login_Sign && Password_Sign == Owner_Password_Sign)
245                 {
246                     // Owner's Menu
247                     Owner_Main_Menu();
248                     system("pause");
249                 }

```

```

253     // If Login hasn't Registered
254     cout << endl
255     |   << endl;
256     cout << "\t\t\t Your Login and Password are Invalid." << endl;
257     cout << "\t\t\t Please press any key to go back to 'Sign in' Menu.\n";
258     cout << endl
259     |   << endl;
260     system("pause");
261     Sign_in();
262 }
263 break;
264
265 case 50:
266 {
267     // Sign in as User
268     system("cls");
269     cout << "
270     L E G E N D R Y S A N N I N S M A R T
271     C U S T O M E R
272     \n\n";
273     cout << "\t\t\t Customer Authentication\n\n";
274     cout << "\t\t\t\tLogin : ";
275     cin >> Login_Sign;
276
277     Password_Sign = Function_Password_Val();
278
279     // File Handling For User Info
280     ifstream Search;
281     Search.open("User_Info.txt");
282
283     while (Search)
284     {
285         Search >> Name_Memory;
286         Search >> TellNum_Memory;
287         Search >> Login_Memory;
288         Search >> Password_Memory;
289         if (Login_Sign == Login_Memory && Password_Sign == Password_Memory)
290         {
291             User user(Name_Memory, TellNum_Memory, Login_Memory, Password_Memory);
292             User_Main_Menu();
293         }
294     }
295     Search.close();
296     // End of the File Handling
297
298     cout << "\n\n\t\t Your Login and Password are Invalid." << endl;
299     cout << "\t\t\t Please press any key to go back to 'Sign in' Menu.\n\n"
300     |   << endl;
301     system("pause");
302     Sign_in();
303 }
304 break;
305
306 case 48:
307     // Back
308     system("cls");
309     i = 1000;
310     General_Menu();
311     break;
312
313 default:
314     cout << "\n\n\t\t Your choice is not available in Menu." << endl;
315     cout << "\t\t\t Please enter correct keys.\n"
316     |   << endl;
317     system("pause");
318     break;
319 }
320 }

```

```
322
323 void LegendarySanninsMart :: Owner_Main_Menu()
324 {
325     system("cls");
326     cout << endl
327         << endl;
328
329     for (int k = 0; k < 1000; k++)
330     {
331         system("cls");
332         cout << " _____ OWNER MENU _____ \n";
333         cout << " L E G E N D R Y S A N N I N S M A R T \n";
334         cout << " _____ \n";
335
336         cout << "      Main Menu\n\n";
337         cout << "      1. Products in stock \n\n";
338         cout << "      2. Customers list \n\n";
339         cout << "      0. Back\n\n";
340         cout << "      Your choice: ";
341
342     switch (getch())
343     {
344     case 49:
345         Owner_Products_Stotage();
346         break;
347
348     case 50:
349         Owner_Customers_List();
350         system("pause");
351         break;
352
353     case 48:
354         // Back to Menu
355         system("cls");
356         k = 1000;
357         Sign_in();
358         break;
359
360     default:
361         cout << "\t\t\tYour choice is not available in Menu." << endl;
362         cout << "\t\t\tPlease enter correct keys.\n"
363             << endl;
364         system("pause");
365         break;
366     }
367 }
368 } // DONE
```

```
370
371 void LegendarySanninsMart :: Owner_Products_Stotage()
372 {
373     for (int i = 0; i < 1000; i++)
374     {
375         system("cls");
376         cout << " _____\n";
377         cout << " L E G E N D R Y   S A N N I N S   M A R T           P R O D U C T S   I N   S T O C K\n";
378         cout << " _____\n";
379
380         cout << "    Products List   \t\t Category\t\t\t Price\t\t\t In Stock\n";
381         Product1_1.Display();
382         Product1_2.Display();
383         Product1_3.Display();
384         Product2_1.Display();
385         Product2_2.Display();
386         Product2_3.Display();
387         Product3_1.Display();
388         Product3_2.Display();
389         Product3_3.Display();
390
391         position1 = 1; // numbering of products return to 1
392
393         cout << endl;
394         | | << endl;
395         cout << " 0. Back\n";
396         cout << "  Make changes in: ";
397         switch (getch())
398         {
399             case '1':
400                 Modify_Info_Products(1);
401                 break;
402
403             case '2':
404                 Modify_Info_Products(2);
405                 break;
406
407             case '3':
408                 Modify_Info_Products(3);
409                 break;
410
411             case '4':
412                 Modify_Info_Products(4);
413                 break;
414
415             case '5':
416                 Modify_Info_Products(5);
417                 break;
418
419             case '6':
420                 Modify_Info_Products(6);
421                 break;
422
423 }
```

```

422     case '7':
423         Modify_Info_Products(7);
424         break;
425
426     case '8':
427         Modify_Info_Products(8);
428         break;
429
430     case '9':
431         Modify_Info_Products(9);
432         break;
433
434     case '0':
435         // Back to Menu
436         system("cls");
437         i = 1000;
438         Owner_Main_Menu();
439         break;
440
441     default:
442         cout << "\n\t\tYour choice is not available in Menu" << endl;
443         cout << "\t\tPlease press any keyboard to continue program\n"
444             << endl;
445         system("pause");
446         break;
447     }
448 }
449
450 }
451 // DONE
452
453 void LegendrySanninsMart :: Modify_Info_Products(int take_product_location)
454 {
455     for (int j = 0; j < 1000; j++)
456     {
457         system("cls");
458
459         cout << "_____\n";
460         cout << " LEGENDRY SANNINS MART\n";
461         cout << "_____\n\n";
462
463         cout << "    Products List \t\t Category\t\t Price\t\t In Stock\n";
464
465         if (take_product_location == 1)
466             Product1_1.Display();
467
468         else if (take_product_location == 2)
469             Product1_2.Display();
470
471         else if (take_product_location == 3)
472             Product1_3.Display();
473
474         else if (take_product_location == 4)
475             Product2_1.Display();
476
477         else if (take_product_location == 5)
478             Product2_2.Display();
479
480         else if (take_product_location == 6)
481             Product2_3.Display();
482
483         else if (take_product_location == 7)
484             Product3_1.Display();
485
486         else if (take_product_location == 8)
487             Product3_2.Display();
488     }
489 }
```

```
488
489     else if (take_product_location == 9)
490         Product3_3.Display();
491
492     position1 = 1;
493     cout << "----- \n\n";
494     cout << " 1. Change the price \n";
495     cout << " 2. Change the quantity in storage\n";
496     cout << " 3. Change the name of product\n";
497     cout << " 4. Change the class of product\n";
498     cout << " 0. Go back \n\n";
499     cout << "    Your choice : \n";
500
501     switch (getch())
502     {
503         // FOR CHANGING A PRICE
504         case 49:
505             cout << "    Enter a new price: ";
506             double change_price;
507             cin >> change_price;
508
509             if (change_price >= 0)
510             {
511                 if (take_product_location == 1)
512                     Product1_1.price = change_price;
513
514                 else if (take_product_location == 2)
515                     Product1_2.price = change_price;
516
517                 else if (take_product_location == 3)
518                     Product1_3.price = change_price;
519
520                 else if (take_product_location == 4)
521                     Product2_1.price = change_price;
522
523                 else if (take_product_location == 5)
524                     Product2_2.price = change_price;
525
526                 else if (take_product_location == 6)
527                     Product2_3.price = change_price;
528
529                 else if (take_product_location == 7)
530                     Product3_1.price = change_price;
531
532                 else if (take_product_location == 8)
533                     Product3_2.price = change_price;
534
535                 else if (take_product_location == 9)
536                     Product3_3.price = change_price;
537
538                 cout << "    Successfully changed!\n";
539                 Sleep(0700);
540                 Sleep(0700);
541             }
542
543             else
544             {
545                 cout << "    Price cannot be negative! Please check one more time.\n";
546                 Sleep(0700);
547                 Sleep(0700);
548             }
549             break;
550 }
```

```
558 // FOR CHANGING THE QUANTITY IN STORAGE
559 case 58:
560     cout << "    Enter a new quantity in storage: ";
561     int change_Quantity;
562     cin >> change_Quantity;
563
564     if (change_Quantity > 0)
565     {
566         if (take_product_location == 1)
567             Product1_1.quantity = change_Quantity;
568
569         else if (take_product_location == 2)
570             Product1_2.quantity = change_Quantity;
571
572         else if (take_product_location == 3)
573             Product1_3.quantity = change_Quantity;
574
575         else if (take_product_location == 4)
576             Product2_1.quantity = change_Quantity;
577
578         else if (take_product_location == 5)
579             Product2_2.quantity = change_Quantity;
580
581         else if (take_product_location == 6)
582             Product2_3.quantity = change_Quantity;
583
584         else if (take_product_location == 7)
585             Product3_1.quantity = change_Quantity;
586
587         else if (take_product_location == 8)
588             Product3_2.quantity = change_Quantity;
589
590         else if (take_product_location == 9)
591             Product3_3.quantity = change_Quantity;
592
593         cout << "    Successfully changed!\n";
594         Sleep(0700);
595         Sleep(0700);
596     }
597     break;
598
599 // FOR CHANGING THE NAME
600 case 51:
601 {
602     cout << "    Enter a new name: ";
603     string product_name = " ";
604     getline(cin, product_name);
605
606     if (take_product_location == 1)
607         Product1_1.name = product_name;
608
609     else if (take_product_location == 2)
610         Product1_2.name = product_name;
611
612     else if (take_product_location == 3)
613         Product1_3.name = product_name;
614
615     else if (take_product_location == 4)
616         Product2_1.name = product_name;
617 }
```

```
617     else if (take_product_location == 5)
618         Product2_2.name = product_name;
619
620     else if (take_product_location == 6)
621         Product2_3.name = product_name;
622
623     else if (take_product_location == 7)
624         Product3_1.name = product_name;
625
626     else if (take_product_location == 8)
627         Product3_2.name = product_name;
628
629     else if (take_product_location == 9)
630         Product3_3.name = product_name;
631
632     cout << "    Successfully changed!\n";
633     Sleep(0700);
634     Sleep(0700);
635 }
636 break;
637
638 case 52:
639 {
640     cout << "    Enter a new Class Name: ";
641     string class_name = " ";
642     getline(cin, class_name);
643
644     if (take_product_location == 1)
645         Product1_1.class_of_products = class_name;
646
647     else if (take_product_location == 2)
648         Product1_2.class_of_products = class_name;
649
650     else if (take_product_location == 3)
651         Product1_3.class_of_products = class_name;
652
653     else if (take_product_location == 4)
654         Product2_1.class_of_products = class_name;
655
656     else if (take_product_location == 5)
657         Product2_2.class_of_products = class_name;
658
659     else if (take_product_location == 6)
660         Product2_3.class_of_products = class_name;
661
662     else if (take_product_location == 7)
663         Product3_1.class_of_products = class_name;
664
665     else if (take_product_location == 8)
666         Product3_2.class_of_products = class_name;
667
668     else if (take_product_location == 9)
669         Product3_3.class_of_products = class_name;
670
671     cout << "    Successfully changed!\n";
672     Sleep(0700);
673     Sleep(0700);
674 }
675 break;
676
677 case 48:
678     j = 1000;
679     break;
680 }
681 }
682 }
683 }
```

```
685 void LegendarySanninsMart :: Owner_Customers_List()
686 {
687     system("cls");
688     cout << " _____\n";
689     cout << " L E G E N D R Y S A N N I N S M A R T      C U S T O M E R S   L I S T \n";
690     cout << " _____\n";
691
692     cout << "\n\t\t\t Customers list : " << endl;
693     cout << endl
694     |<< endl;
695
696     ifstream in;
697     int Num = 1;
698     string Info;
699
700     in.open("User_Info.txt");
701     while (in)
702     {
703         cout << "\t " << Num << ".\n";
704         cout << "\t-----\n";
705         getline(in, Info);
706         cout << "\t User Name : " << Info << endl;
707         getline(in, Info);
708         cout << "\t Phone      : " << Info << endl;
709         getline(in, Info);
710         cout << "\t Login      : " << Info << endl;
711         getline(in, Info);
712         cout << "\t Password   : " << Info << endl;
713         Num++;
714         cout << endl;
715     }
716     cout << "\t-----\n";
717     in.close();
718 }
719 // DONE
720
```

```
722 void LegendarySanninsMart :: User_Main_Menu()
723 {
724     for (int i = 0; i < 1000; i++)
725     {
726         system("cls");
727         cout << " _____ ";
728         cout << " L E G E N D R Y S A N N I N S M A R T ";
729         cout << " _____ ";
730         cout << "\n";
731         cout << "\t\t Categories\n\n";
732         cout << "\t\t 1. Vegetables & Fruits\n";
733         cout << "\t\t 2. Water & Beverages\n";
734         cout << "\t\t 3. Fast Food Products\n";
735         cout << "\t\t 4. Cart and Overall Sums\n";
736         cout << "\t\t 0. Go Back\n\n";
737         cout << "\t\t Your choice: ";
738
739         switch (_getch())
740         {
741             case 49:
742                 Class_Of_Products1_Menu();
743                 break;
744
745             case 50:
746                 Class_Of_Products2_Menu();
747                 break;
748
749             case 51:
750                 Class_Of_Products3_Menu();
751                 break;
752
753             case 52:
754                 Cart_Check();
755                 break;
756
757             case 48:
758                 // Back to Menu
759                 system("cls");
760                 i = 1000;
761                 Sign_in();
762                 break;
763
764             default:
765                 cout << endl
766                 | << endl;
767                 cout << "\t\t\t\t Your choice is not available in Menu." << endl;
768                 cout << "\t\t\t\t Please enter correct keys.\n"
769                 | << endl;
770                 system("pause");
771                 break;
772         }
773     }
774 }
```

```

776
777 void LegendarySanninsMart :: Table_For_Increasing_And_Decreasing()
778 {
779     cout << " _____ \n";
780     cout << " (+) Press 1' \n";
781     cout << " (-) Press 2' \n";
782     cout << " (0) Back' \n";
783     cout << " _____ \n\n";
784     cout << " Add to Cart: \n";
785 }
786 // DONE
787
788 void LegendarySanninsMart :: Class_Of_Products1_Menu()
789 {
790     for (int k = 0; k < 1000; k++)
791     {
792         system("cls");
793         cout << " _____ \n";
794         cout << " L E G E N D R Y S A N N I N S M A R T " PRODUCT 1 \n";
795         cout << " _____ \n\n";
796
797         cout << " Products List \t\t Category \t\t Price \t\t In Stock\n";
798         Product1_1.Display();
799         Product1_2.Display();
800         Product1_3.Display();
801
802         cout << " 0. Back\n\n";
803         cout << " Your choice: ";
804
805         switch (getch())
806         {
807             // for Product1_1
808             case 49:
809                 for (int j = 0; j < 1000; j++) {
810                     system("cls");
811                     cout << " Products List \t\t Category\t\t Price\t\t In Stock\n";
812                     Product1_1.Display();
813                     position1 = 1;
814
815                     Table_For_Increasing_And_Decreasing();
816
817                     switch (getch())
818                     {
819                         case 49:
820                             if (Product1_1.getQuantity() > 0) {
821                                 user.Product1_1_User++;
822                                 Product1_1.quantity--;
823                                 cout << " Quantity of " << Product1_1.getName() << " - " << user.Product1_1_User << endl;
824                                 cout << " Successfully added \n";
825                                 Sleep(0700);
826                                 Sleep(0700);
827                             }
828
829                         else {
830                             cout << " Product is over / finished. Sorry!\n";
831                             Sleep(0700);
832                             Sleep(0700);
833                         }
834                         break;
835
836                     case 50:
837                         if (user.Product1_1_User > 0) {
838                             user.Product1_1_User--;
839                             Product1_1.quantity++;
840                             cout << " Quantity of " << Product1_1.getName() << " - " << user.Product1_1_User << endl;
841                             cout << " Successfully decreased \n";
842                             Sleep(0700);
843                             Sleep(0700);

```

```

844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860 // for Product1_2
861 case 50:
862 {
863     for (int j = 0; j < 1000; j++)
864     {
865         system("cls");
866         cout << "    Products List \t\t Category\t\t Price\t\t In Stock\n";
867         Product1_2.Display();
868         position1 = 1;
869
870         Table_For_Increasing_And_Decreasing();
871
872         switch (getch())
873         {
874             case 49:
875                 if (Product1_2.getQuantity() > 0) {
876                     user.Product1_2_User++;
877                     Product1_2.quantity--;
878                     cout << " Quantity of " << Product1_2.getName() << " - " << user.Product1_2_User << endl;
879                     cout << "      Successfully added \n";
880                     Sleep(0700);
881                     Sleep(0700);
882                 }
883
884                 else {
885                     cout << " Product is over / finished. Sorry!\n";
886                     Sleep(0700);
887                     Sleep(0700);
888                 }
889                 break;
890
891             case 50:
892                 if (user.Product1_2_User > 0) {
893                     user.Product1_2_User--;
894                     Product1_2.quantity++;
895                     cout << " Quantity of " << Product1_2.getName() << " - " << user.Product1_2_User << endl;
896                     cout << "      Successfully decreased \n";
897                     Sleep(0700);
898                     Sleep(0700);
899                 }
900
901                 else {
902                     cout << " 0 (kg/pc) can not decrease \n";
903                     Sleep(0700);
904                     Sleep(0700);
905                 }
906                 break;
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987

```

```
906
907     case 48:
908         j = 1000;
909         break;
910     }
911 }
912 break;
913
914
915
916
917 // for Product1_3
918 case 51:
919 {
920     for (int j = 0; j < 1000; j++)
921     {
922         system("cls");
923         cout << "    Products List \t\t Category\t\t      Price\t\t In Stock\n";
924         Product1_3.Display();
925         position1 = 1;
926
927         Table_For_Increasing_And_Decreasing();
928
929         switch (_getch())
930         {
931             case 49:
932                 if (Product1_3.getQuantity() > 0) {
933                     user.Product1_3_User++;
934                     Product1_3.quantity--;
935                     cout << " Quantity of " << Product1_3.getName() << " - " << user.Product1_3_User << endl;
936                     cout << "   Successfully added \n";
937                     Sleep(0700);
938                     Sleep(0700);
939                 }
940
941             else {
942                 cout << " Product is over / finished. Sorry!\n";
943                 Sleep(0700);
944                 Sleep(0700);
945             }
946             break;
947
948         case 50:
949             if (user.Product1_3_User > 0) {
950                 user.Product1_3_User--;
951                 Product1_3.quantity++;
952                 cout << " Quantity of " << Product1_3.getName() << " - " << user.Product1_3_User << endl;
953                 cout << "   Successfully decreased \n";
954                 Sleep(0700);
955                 Sleep(0700);
956             }
957         }
958 }
```

```
957
958         else {
959             cout << " 0 (kg/pc) can not decrease \n";
960             Sleep(0700);
961             Sleep(0700);
962         }
963     break;
964
965     case 48:
966         j = 1000;
967         break;
968     }
969 }
970
971 break;
972
973 // Back to User menu
974 case 48:
975     k = 1000;
976     User_Main_Menu();
977     break;
978
979 default:
980     cout << "\n\n\t\t\tYour choice is not available in Menu." << endl;
981     cout << "\t\t\tPlease enter correct keys.\n\n";
982     system("pause");
983     break;
984
985 }
986 }
987
988 void LegendrySanninsMart :: Class_Of_Products2_Menu()
989 {
990     for (int k = 0; k < 1000; k++)
991     {
992         system("cls");
993         cout << "
994         L E G E N D R Y   S A N N I N S   M A R T
995         P R O D U C T   2
996         "
997
998         cout << "    Products List  \t\t Category\t\t      Price\t\t In Stock\n";
999         Product2_1.Display();
1000        Product2_2.Display();
1001        Product2_3.Display();
1002
1003        cout << " 0. Back\n\n";
1004        cout << " Your choice: ";
```

```
1005     switch (_getch())
1006     {
1007         // Product2_1
1008         case 49:
1009         {
1010             for (int j = 0; j < 1000; j++)
1011             {
1012                 system("cls");
1013                 cout << "      Products List    \t\t Category\t\t      Price\t\t In Stock\n";
1014                 Product2_1.Display();
1015                 position1 = 1;
1016
1017                 Table_For_Increasing_And_Decreasing();
1018                 switch (_getch())
1019                 {
1020                     case 49:
1021                         if (Product2_1.getQuantity() > 0) {
1022                             // checking for storage and user needs
1023                             Product2_1.quantity--;
1024                             user.Product2_1_User++;
1025                             cout << " Quantity of " << Product2_1.getName() << " - " << user.Product2_1_User << endl;
1026                             cout << "      Successfully added \n";
1027                             Sleep(0700);
1028                             Sleep(0700);
1029                         }
1030
1031                     else {
1032                         cout << " Product is over / finished. Sorry!\n";
1033                         Sleep(0700);
1034                         Sleep(0700);
1035                     }
1036                 break;
1037
1038             case 50:
1039                 if (user.Product2_1_User > 0) {
1040                     // Check for (-1 kg/pc)
1041                     Product2_1.quantity++;
1042                     user.Product2_1_User--;
1043                     cout << " Quantity of " << Product2_1.getName() << " - " << user.Product2_1_User << endl;
1044                     cout << "      Successfully decreased \n";
1045                     Sleep(0700);
1046                     Sleep(0700);
1047                 }
1048
1049                 else {
1050                     cout << " 0 (kg/pc) can not decrease \n";
1051                     Sleep(0700);
1052                     Sleep(0700);
1053                 }
1054             break;
1055
1056 }
```

```
1056         case 48:
1057             j = 1000;
1058             break;
1059         }
1060     }
1061 }
1062 break;
1063
// Product2_2
1064 case 50:
1065 {
1066     for (int j = 0; j < 1000; j++)
1067     {
1068         system("cls");
1069         cout << "    Products List \t\t Category\t\t      Price\t\t In Stock\n";
1070         Product2_2.Display();
1071         position1 = 1;
1072
1073         Table_For_Increasing_And_Decreasing();
1074         switch (_getch())
1075         {
1076             case 49:
1077                 if (Product2_2.getQuantity() > 0) {
1078                     // checking for storage and user needs
1079                     Product2_2.quantity--;
1080                     user.Product2_2_User++;
1081                     cout << " Quantity of " << Product2_2.getName() << " - " << user.Product2_2_User << endl;
1082                     cout << " Successfully added \n";
1083                     Sleep(0700);
1084                     Sleep(0700);
1085                 }
1086             }
1087         }
1088
1089         else {
1090             cout << " Product is over / finished. Sorry!\n";
1091             Sleep(0700);
1092             Sleep(0700);
1093         }
1094     break;
1095
1096     case 50:
1097         if (user.Product2_2_User > 0) {
1098             // Check for (-1 kg/pc)
1099             Product2_2.quantity++;
1100             user.Product2_2_User--;
1101             cout << " Quantity of " << Product2_2.getName() << " - " << user.Product2_2_User << endl;
1102             cout << " Successfully decreased \n";
1103             Sleep(0700);
1104             Sleep(0700);
1105         }
1106
1107         else {
1108             cout << " 0 (kg/pc) can not decrease \n";
1109             Sleep(0700);
1110             Sleep(0700);
1111         }
1112     break;
1113
1114     case 48:
1115         j = 1000;
1116         break;
1117     }
1118 }
1119 }
1120 break;
1121
```

```
1122 // Product2_3
1123 case 51:
1124 {
1125     for (int j = 0; j < 1000; j++)
1126     {
1127         system("cls");
1128         cout << "    Products List \t\t Category\t\t      Price\t\t In Stock\n";
1129         Product2_3.Display();
1130         position1 = 1;
1131
1132         Table_For_Increasing_And_Decreasing();
1133         switch (_getch())
1134         {
1135             case 49:
1136                 if (Product2_3.getQuantity() > 0) {
1137                     // checking for storage and user needs
1138                     Product2_3.quantity--;
1139                     user.Product2_3_User++;
1140                     cout << " Quantity of " << Product2_3.getName() << " - " << user.Product2_3_User << endl;
1141                     cout << "    Successfully added \n";
1142                     Sleep(0700);
1143                     Sleep(0700);
1144                 }
1145
1146                 else {
1147                     cout << " Product is over / finished. Sorry!\n";
1148                     Sleep(0700);
1149                     Sleep(0700);
1150                 }
1151             break;
1152
1153         case 50:
1154             if (user.Product2_3_User > 0) {
1155                 // Check for (-1 kg/pc)
1156                 Product2_3.quantity++;
1157                 user.Product2_3_User--;
1158                 cout << " Quantity of " << Product2_3.getName() << " - " << user.Product2_3_User << endl;
1159                 cout << "    Successfully decreased \n";
1160                 Sleep(0700);
1161                 Sleep(0700);
1162             }
1163
1164             else {
1165                 cout << " 0 (kg/pc) can not decrease \n";
1166                 Sleep(0700);
1167                 Sleep(0700);
1168             }
1169         break;
1170
1171         case 48:
1172             j = 1000;
1173             break;
1174         }
1175     }
1176 }
1177 break;
```

```

1178
1179
1180     case 48:
1181     {
1182         k = 1000;
1183         User_Main_Menu();
1184     }
1185     break;
1186
1187
1188     default:
1189         cout << "\n\n\t\t\tYour choice is not available in Menu." << endl;
1190         cout << "\t\t\t\tPlease enter correct keys.\n"
1191         << endl;
1192         system("pause");
1193         break;
1194
1195     } // switch ends
1196
1197 } // loop ends
1198
1199 } // function ends
1200
1201 void LegendrySanninsMart :: Class_Of_Products3_Menu()
1202 {
1203     for (int k = 0; k < 1000; k++)
1204     {
1205
1206         system("cls");
1207         cout << " _____ \n";
1208         cout << " L E G E N D R Y S A N N I N S M A R T   P R O D U C T   B   \n";
1209         cout << " _____ \n";
1210
1211         cout << "    Products List \t\t Category\t\t Price\t\t In Stock\n";
1212         Product3_1.Display();
1213         Product3_2.Display();
1214         Product3_3.Display();
1215
1216         cout << " 0. Back\n\n";
1217         cout << " Your choice: ";
1218         switch (_getch())
1219         {
1220             // Product3_1
1221             case 49:
1222             {
1223                 for (int j = 0; j < 1000; j++)
1224                 {
1225                     system("cls");
1226
1227                     cout << "    Products List \t\t Category\t\t Price\t\t In Stock\n";
1228                     Product3_1.Display();
1229                     position1 = 1;
1230
1231                     Table_For_Increasing_And_Decreasing();
1232                     switch (_getch())
1233                     {
1234                         case 49:
1235                             if (Product3_1.getQuantity() > 0) {
1236                                 Product3_1.quantity--;
1237                                 user.Product3_1_User++;
1238                                 cout << " Quantity of " << Product3_1.getName() << " - " << user.Product3_1_User << endl;
1239                                 cout << "      Successfully added \n";
1240                                 Sleep(8700);
1241                                 Sleep(8700);
1242                             }
1243                     }
1244                 }
1245             }
1246         }
1247     }
1248 }

```

```

1243
1244     else {
1245         cout << " Product is over / finished. Sorry!\n";
1246         Sleep(0700);
1247         Sleep(0700);
1248     }
1249     break;
1250
1251 case 50:
1252     if (user.Product3_1_User > 0) {
1253         Product3_1.quantity++;
1254         user.Product3_1_User--;
1255         cout << " Quantity of " << Product3_1.getName() << " - " << user.Product3_1_User << endl;
1256         cout << " Successfully decreased \n";
1257         Sleep(0700);
1258         Sleep(0700);
1259     }
1260
1261     else {
1262         cout << " 0 (kg/pc) can not decrease \n";
1263         Sleep(0700);
1264         Sleep(0700);
1265     }
1266     break;
1267
1268 case 48:
1269     j = 1000;
1270     break;
1271 }
1272 }
1273 }
1274 break;
1275
1276 // Product3_2
1277 case 50:
1278 {
1279     for (int j = 0; j < 1000; j++)
1280     {
1281         system("cls");
1282         cout << " Products List \t\t Category\t\t Price\t\t In Stock\n";
1283         Product3_2.Display();
1284         position1 = 1;
1285
1286         Table_For_Increasing_And_Decreasing();
1287         switch (_getch())
1288         {
1289             case 49:
1290                 if (Product3_2.getQuantity() > 0) {
1291                     Product3_2.quantity--;
1292                     user.Product3_2_User++;
1293                     cout << " Quantity of " << Product3_2.getName() << " - " << user.Product3_2_User << endl;
1294                     cout << " Successfully added \n";
1295                     Sleep(0700);
1296                     Sleep(0700);
1297                 }
1298
1299             else {
1300                 cout << " Product is over / finished. Sorry!\n";
1301                 Sleep(0700);
1302                 Sleep(0700);
1303             }
1304             break;
1305         case 50:
1306             if (user.Product3_2_User > 0) {
1307                 Product3_2.quantity++;
1308                 user.Product3_2_User--;
1309             }
1310         }
1311     }
1312 }
1313
1314
1315
1316
1317
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1388
1389
1390
1391
1392
1393
1394
1395
1396
1397
1398
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1498
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1598
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1798
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1898
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1998
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2010
2011
2012
2013
2014
2015
2016
2017
2018
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2098
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2119
2120
2121
2122
2123
2124
2125
2126
2127
2128
2129
2130
2131
2132
2133
2134
2135
2136
2137
2138
2139
2140
2141
2142
2143
2144
2145
2146
2147
2148
2149
2150
2151
2152
2153
2154
2155
2156
2157
2158
2159
2160
2161
2162
2163
2164
2165
2166
2167
2168
2169
2170
2171
2172
2173
2174
2175
2176
2177
2178
2179
2180
2181
2182
2183
2184
2185
2186
2187
2188
2189
2190
2191
2192
2193
2194
2195
2196
2197
2198
2199
2200
2201
2202
2203
2204
2205
2206
2207
2208
2209
2210
2211
2212
2213
2214
2215
2216
2217
2218
2219
2220
2221
2222
2223
2224
2225
2226
2227
2228
2229
2230
2231
2232
2233
2234
2235
2236
2237
2238
2239
2240
2241
2242
2243
2244
2245
2246
2247
2248
2249
2250
2251
2252
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281
2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325
2326
2327
2328
2329
2330
2331
2332
2333
2334
2335
2336
2337
2338
2339
2340
2341
2342
2343
2344
2345
2346
2347
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365
2366
2367
2368
2369
2370
2371
2372
2373
2374
2375
2376
2377
2378
2379
2380
2381
2382
2383
2384
2385
2386
2387
2388
2389
2390
2391
2392
2393
2394
2395
2396
2397
2398
2399
2400
2401
2402
2403
2404
2405
2406
2407
2408
2409
2410
2411
2412
2413
2414
2415
2416
2417
2418
2419
2420
2421
2422
2423
2424
2425
2426
2427
2428
2429
2430
2431
2432
2433
2434
2435
2436
2437
2438
2439
2440
2441
2442
2443
2444
2445
2446
2447
2448
2449
2450
2451
2452
2453
2454
2455
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467
2468
2469
2470
2471
2472
2473
2474
2475
2476
2477
2478
2479
2480
2481
2482
2483
2484
2485
2486
2487
2488
2489
2490
2491
2492
2493
2494
2495
2496
2497
2498
2499
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535
2536
2537
2538
2539
2540
2541
2542
2543
2544
2545
2546
2547
2548
2549
2550
2551
2552
2553
2554
2555
2556
2557
2558
2559
2560
2561
2562
2563
2564
2565
2566
2567
2568
2569
2570
2571
2572
2573
2574
2575
2576
2577
2578
2579
2580
2581
2582
2583
2584
2585
2586
2587
2588
2589
2590
2591
2592
2593
2594
2595
2596
2597
2598
2599
2600
2601
2602
2603
2604
2605
2606
2607
2608
2609
2610
2611
2612
2613
2614
2615
2616
2617
2618
2619
2620
2621
2622
2623
2624
2625
2626
2627
2628
2629
2630
2631
2632
2633
2634
2635
2636
2637
2638
2639
2640
2641
2642
2643
2644
2645
2646
2647
2648
2649
2650
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663
2664
2665
2666
2667
2668
2669
2670
2671
2672
2673
2674
2675
2676
2677
2678
2679
2680
2681
2682
2683
2684
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
269
```

```

1305     case 50:
1306         if (user.Product3_2_User > 0) {
1307             Product3_2.quantity++;
1308             user.Product3_2_User--;
1309             cout << " Quantity of " << Product3_2.getName() << " - " << user.Product3_2_User << endl;
1310             cout << " Successfully decreased \n";
1311             Sleep(0700);
1312             Sleep(0700);
1313         }
1314
1315     else {
1316         cout << " 0 (kg/pc) can not decrease \n";
1317         Sleep(0700);
1318         Sleep(0700);
1319     }
1320     break;
1321
1322     case 48:
1323         j = 1000;
1324         break;
1325     }
1326 }
1327 break;
1328
1329 // Product3_3
1330 case 51:
1331 {
1332     for (int j = 0; j < 1000; j++)
1333     {
1334         system("cls");
1335         cout << " Products List \t\t Category\t\t Price\t\t In Stock\n";
1336         Product3_3.Display();
1337         position1 = 1;
1338
1339         Table_For_Increasing_And_Decreasing();
1340         switch (_getch())
1341         {
1342             case 49:
1343                 if (Product3_3.getQuantity() > 0) {
1344                     Product3_3.quantity--;
1345                     user.Product3_3_User++;
1346                     cout << " Quantity of " << Product3_3.getName() << " - " << user.Product3_3_User << endl;
1347                     cout << " Successfully added \n";
1348                     Sleep(0700);
1349                     Sleep(0700);
1350                 }
1351
1352             else {
1353                 cout << " Product is over / finished. Sorry!\n";
1354                 Sleep(0700);
1355                 Sleep(0700);
1356             }
1357             break;
1358
1359         case 50:
1360             if (user.Product3_3_User > 0) {
1361                 Product3_3.quantity++;
1362                 user.Product3_3_User--;
1363                 cout << " Quantity of " << Product3_3.getName() << " - " << user.Product3_3_User << endl;
1364                 cout << " Successfully decreased \n";
1365                 Sleep(0700);
1366                 Sleep(0700);
1367             }
1368         }
1369     }
1370 }

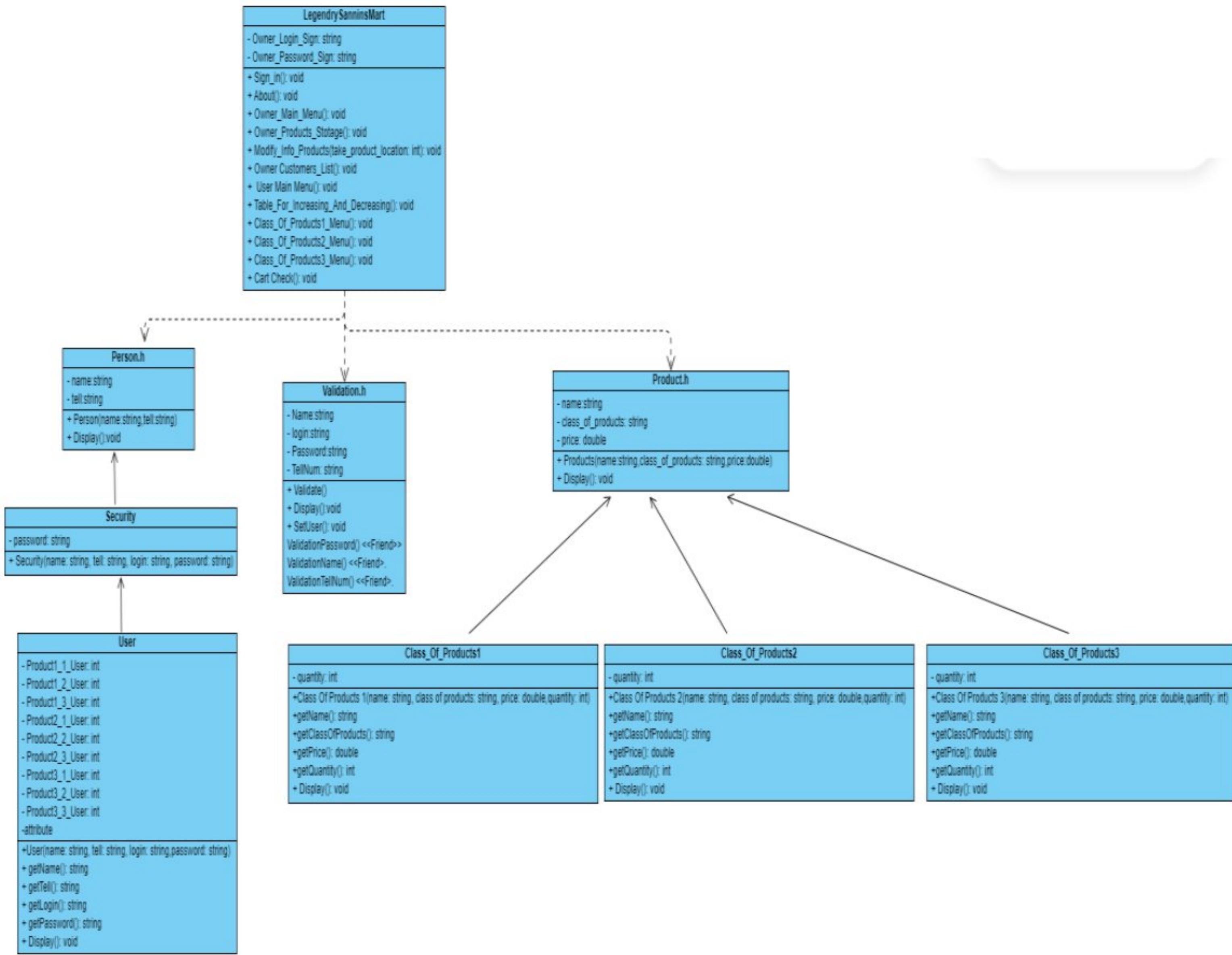
```

```
1369
1370     else {
1371         cout << " 0 (kg/pc) can not decrease \n";
1372         Sleep(0700);
1373         Sleep(0700);
1374     }
1375     break;
1376
1377     case 48:
1378         j = 1000;
1379         break;
1380     }
1381 }
1382 break;
1383
1384 // Back to F_User menu
1385 case 48:
1386     k = 1000;
1387     User_Main_Menu();
1388     break;
1389
1390 default:
1391     cout << "\n\n\t\t\t Your choice is not available in Menu." << endl;
1392     cout << "\t\t\t\t Please enter correct keys.\n" << endl;
1393     system("pause");
1394
1395 } // switch ends
1396 } // loop ends
1397 } // function ends
1398
1399 void LegendarySanninsMart :: Cart_Check() {
1400
1401     system("cls");
1402     cout << " _____ ";
1403     cout << " LEGENDRY SANNINS MART ";
1404     cout << " CART ";
1405     cout << " _____ ";
1406
1407     // Check
1408     double Overall_Sum = 0;
1409     for (int i = 1; i <= 1; i++)
1410     {
1411         if (user.Product1_i_User > 0) {
1412             cout << "\n\t\t\t " << i << ". " << Product1_i.getName() << " --- " << user.Product1_i_User << " --- Price: " << user.Product1_i_User * Product1_i.getPrice();
1413             i++;
1414             Overall_Sum += user.Product1_i_User * Product1_i.getPrice();
1415         }
1416
1417         if (user.Product1_2_User > 0) {
1418             cout << "\n\t\t\t " << i << ". " << Product1_2.getName() << " --- " << user.Product1_2_User << " --- price: " << user.Product1_2_User * Product1_2.getPrice();
1419             i++;
1420             Overall_Sum += user.Product1_2_User * Product1_2.getPrice();
1421         }
1422     }
1423 }
```

```
1422
1423     if (user.Product1_3_User > 0) {
1424         cout << "\n\t\t\t " << i << ". " << Product1_3.getName() << " --- " << user.Product1_3_User << " --- Price: " << user.Product1_3_User * Product1_3.getPrice();
1425         i++;
1426         Overall_Sum += user.Product1_3_User * Product1_3.getPrice();
1427     }
1428
1429     if (user.Product2_1_User > 0) {
1430         cout << "\n\t\t\t " << i << ". " << Product2_1.getName() << " --- " << user.Product2_1_User << " --- Price: " << user.Product2_1_User * Product2_1.getPrice();
1431         i++;
1432         Overall_Sum += user.Product2_1_User * Product2_1.getPrice();
1433     }
1434     if (user.Product2_2_User > 0) {
1435         cout << "\n\t\t\t " << i << ". " << Product2_2.getName() << " --- " << user.Product2_2_User << " --- Price: " << user.Product2_2_User * Product2_2.getPrice();
1436         i++;
1437         Overall_Sum += user.Product2_2_User * Product2_2.getPrice();
1438     }
1439     if (user.Product2_3_User > 0) {
1440         cout << "\n\t\t\t " << i << ". " << Product2_3.getName() << " --- " << user.Product2_3_User << " --- Price: " << user.Product2_3_User * Product2_3.getPrice();
1441         i++;
1442         Overall_Sum += user.Product2_3_User * Product2_3.getPrice();
1443     }
1444     if (user.Product3_1_User > 0) {
1445         cout << "\n\t\t\t " << i << ". " << Product3_1.getName() << " --- " << user.Product3_1_User << " --- Price: " << user.Product3_1_User * Product3_1.getPrice();
1446         i++;
1447         Overall_Sum += user.Product3_1_User * Product3_1.getPrice();
1448     }
1449     if (user.Product3_2_User > 0) {
1450         cout << "\n\t\t\t " << i << ". " << Product3_2.getName() << " --- " << user.Product3_2_User << " --- Price: " << user.Product3_2_User * Product3_2.getPrice();
1451         i++;
1452         Overall_Sum += user.Product3_2_User * Product3_2.getPrice();
1453     }
1454     if (user.Product3_3_User > 0) {
1455         cout << "\n\t\t\t " << i << ". " << Product3_3.getName() << " --- " << user.Product3_3_User << " --- Price: " << user.Product3_3_User * Product3_3.getPrice();
1456         i++;
1457         Overall_Sum += user.Product3_3_User * Product3_3.getPrice();
1458     }
1459
1460     if (i == 1) {
1461         // if nothing go to Menu
1462         cout << "\n\t You do not have any product in 'CART'.\n";
1463         cout << "\tPress any key to go to 'Products Menu'\n\n" << endl;
1464         system("pause");
1465         Overall_Sum = 0;
1466         User_Main_Menu();
1467     }
1468     if (i > 1) {
1469         // Menu for buying or back
1470         cout << "\n\t\t\t Overall Price: " << Overall_Sum << " Sums" << endl;
1471         cout << "\n\t\t 1. Buy now" << endl;
1472         cout << "\t\t 0. Products Menu" << endl;
```

```
1473
1474     switch (_getch()) {
1475     case 49: { //buy
1476         system("cls");
1477         /////
1478         cout << " _____ \n";
1479         cout << " C A R T \n";
1480         cout << " _____ \n\n";
1481
1482         cout << " Money will be taken from your 'Telephone Number': " << endl;
1483         cout << " 1, OK" << endl;
1484         cout << " Press any key to go back..." << endl;
1485         switch (_getch())
1486         {
1487             case 49:
1488             {
1489                 cout << "\n\t Transaction Successful!\n\t Congratulations ! :)" << endl;
1490                 system("pause");
1491                 Overall_Sum = 0;
1492                 // Storage
1493                 user.Product1_1_User = 0; user.Product1_3_User = 0; user.Product1_2_User = 0;
1494                 user.Product2_1_User = 0; user.Product2_2_User = 0; user.Product2_3_User = 0;
1495                 user.Product3_1_User = 0; user.Product3_2_User = 0; user.Product3_3_User = 0;
1496                 User_Main_Menu();
1497             }
1498             break;
1499
1500             default:
1501                 Cart_Check();
1502             }
1503             system("pause");
1504         }
1505         break;
1506
1507     case 48:
1508     { // Back
1509         system("cls");
1510         Overall_Sum = 0;
1511         User_Main_Menu();
1512     }
1513     break;
1514
1515     default:
1516         Cart_Check();
1517     }
1518     } // switch ends
1519 } //if ends
1520 } // loop end
1521 } // function ends
1522 }
```

UML DIAGRAM:



TECHNIQUES USED IN THIS PROGRAM:

- **INHERITANCE**
- **POLYMORPHISM (VIRTUAL FUNCTION)**
- **ENCAPSULATION**
- **PURE VIRTUAL FUNCTION.**

MOST CHALLENGING PART:

The most challenging part we faced while coding is firstly to decide that which approach should we go for, we have used multiple approaches so that anyone can easily understand our code, we have tried dictionary, different functions, used if-else statements in different ways and have tried many other things.

Additionally, managing the flow of data and interactions between different classes can be complex. Ensuring proper encapsulation, information hiding, and maintaining the integrity of the data poses challenges. Handling exceptions, error conditions, and ensuring data consistency across classes and methods are also crucial aspects to address.

MEMBERS CONTRIBUTION:

The work was done in the group but the individual contributions are:

MUHAIB SHAMSHER: [CR-22029]

- Syntax of code.
- Defining class interference.
- Header files.
- Including necessary libraries.
- Logics and calculations.
- Designing class structures.
- Use of tokens
- Implementation of whole project into code.

MUNEEZA BADAR: [CR-22022]

- Testing and quality insurance.
- Designing class structures.

- Use of conditional statements and loops.
- Debugs the bugs.
- UML Diagram.
- Logics and calculations.
- Features of project.
- Project Report.

MUHAMMAD HAMZA: [CR-22048]

- Main idea of the project.
- Project planning.
- Idea for the features.
- Debugs the bugs.
- Documentations and comments.
- Logics and calculations.
- Use of keywords.
- Error handling.

FUTURE EXPANSIONS/ LIMITATION OF APPROACH:

- Customer reviews and ratings: Add a feedback system that allows customers to rate and review products, helping other customers make informed purchasing decisions.
- Payment gateway integration: Integrate popular payment gateways to enable secure online payments, such as credit/debit cards, digital wallets, or online banking.
- Discounts, promotions, and coupons: Implement a system to apply discounts, promotions, or coupon codes to specific products or orders, enhancing customer engagement and loyalty.

CONCLUSION:

In conclusion, the C++ OOP Online Grocery Mart project has successfully developed a user-friendly and efficient online platform for customers to conveniently shop

for groceries. The project incorporated essential features like the loading page, registration page, login page, menu page, and validation mechanisms to provide a seamless and secure shopping experience. The loading page created a visually appealing start, while the registration and login pages allowed users to create accounts and authenticate themselves securely. The menu page provided a categorized display of grocery items for easy browsing. Additionally, robust validation techniques were implemented throughout the project to ensure data integrity and prevent errors. Overall, the project successfully met its objectives by offering an intuitive interface and incorporating necessary features to enhance the online grocery shopping experience.

CT-260 Object Oriented Programming**Complex Computing Problem (CCP)**

Choose a problem in any application domain of computing and **define** its solution through an effective use of object oriented programming principles. The solution must be in the form of a well-documented and well-formatted report. It must be explained by employing in-depth computing or domain knowledge, and an approach that is based on well-founded principles.

The following instructions must be taken into consideration while preparing the project report. The report must reflect conceptual thinking and must properly be modularized according to functionality. The report of the solution must be organized clearly and should:

- ✓ describe the steps of your solution,
- ✓ the main object oriented programming techniques used,
- ✓ the limitations of the adopted approach.

Complex Computing Problem Assessment Rubrics

Criteria and Scales		
Good (5-4)	Average (3-2)	Poor (1-0)
Criterion 1 Problem Understanding: To what extent has the student has understood the problem?		
<i>(CP3: Depth of Knowledge Required)</i>		
The student has clearly understood the Problem.	The student has some clarity of the Problem.	The student has misunderstood the Problem.
Criterion 2 Requirement Identification: To what extent has the student outlined the object oriented programming principles required for solving the problem.		
<i>(CP10: Requirement Identification)</i>		
The student has clearly outlined the required object oriented programming principles.	The student has outlined some of the required object oriented programming principles.	The students has not outlined the correct object oriented programming principles.
Criterion 3 Clarity of Solution: To what extent has the student defined the solution of the problem.		
<i>(CP3: Depth of Knowledge Required)</i>		
The student has clearly defined the solution which works for all possible test cases.	The student has defined a solution which works well for a few test cases.	The student has not defined a correct solution.

Total marks: _____ /5

Teacher's signature: _____