Kiryana Connect

Your One-Stop Shop for Seamless Store Management

Critical Overview

PROJECT REPORT

Kiryana Connect Making Retail Easier

Thursday, January 11, 2024

1st Year Programming Fundamentals

DATABASE MENU SYSTEM

Mr. M. OBAID MAJEED

PROJECT REPORT

Course: Programming Fundamentals (CT-175)

Group Members:

Muhammad Obaid (CT-025)

Akhyar Ahmed Turk (CT-034)

Problem Statement:

The Problem: -

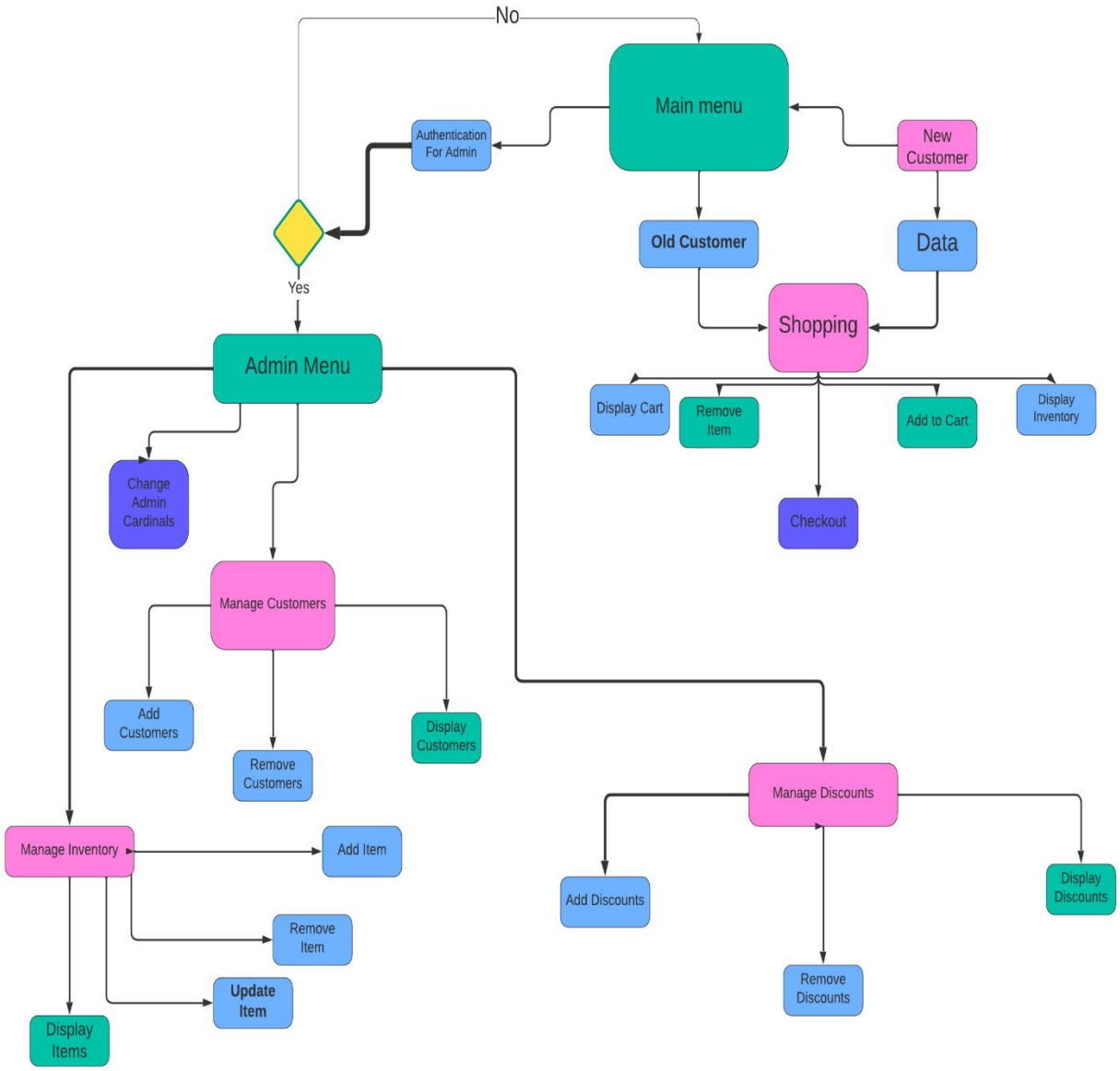
Running a superstore and managing its data, which includes inventory, customer information, and checkout records is necessary for shopkeepers. However, doing all of this manually can be a significant challenge for them. Firstly, managing this large amount of data manually is not user-friendly. Secondly, it can lead to errors while managing inventory, registering customers, and administering discounts. Moreover, this old register method is time-consuming and inefficient.

The Proposed Solution: -

Keeping in mind, the problems faced by shopkeepers regarding data handling, we designed a Kiryana OR General Store Data Base Management

System. This program includes an interface with admin access where the shopkeeper can handle store inventory, and manage customers and discounts. The financial management of the shop can be done through this program. Other than that, the program provides a digital ordering system for customers at the local level. The program aims to provide an efficient way to manage data for shopkeepers with no chance of error.

Functionalities Map:



Functionalities:

Main Menu:

Main Function:

Main function Firstly, load customers from Customer file, Inventory from inventory file and then give the user options, that whether he want to continue with admin account, new customer (If new than registered him/her first), old customer, etc.

Then redirect them to their respective functions.

```
1151  int main() {
1152      struct Customer newCustomer;
1153      char customerPhoneNum[20] ;
1154      int pin;
1155      int choice;
1156      loadCustomersWithDiscountsFromFile( ) ;
1157      loadInventoryFromFile("inventory_data.txt");
1158      gotoxy(84, 0) ;
1159      printf("Admin Interface Formattion Credits: ") ;
1160      gotoxy(96, 1) ;
1161      printf("Akhyar Ahmed") ;
1162
1163      do {
1164          gotoxy(20, 3);
1165          printf("========= GENERAL STORE MANAGEMENT SYSTEM =========\n");
1166          gotoxy(20, 4);
1167          printf("\t      ******************************");
1168          gotoxy(35, 7) ;
1169          printf("DEVELOPED BY M.OBAID");
1170          gotoxy(25, 10);
1171          printf("1. New Customer Shopping\n");
1172          gotoxy(25, 12);
1173          printf("2. Existing Customer Shopping\n");
1174          gotoxy(25, 14);
1175          printf("3. Admin Menu\n");
1176          gotoxy(25, 16);
1177          printf("0. Exit\n");
1178          gotoxy(25, 18);
1179          printf("\nEnter your choice: ");
1180          scanf("%d", &choice);
1181          fflush(stdin) ; // Used flush buffer where necessary to remove white spaces or new lines wrongly given in integer to avoid code breakage
1182
```

Gotoxy: -

This function positions the console cursor at the specified coordinates `(x, y)` using Windows API. It is used for formatting output on the console.

```
void gotoxy(int x, int y) {
    HANDLE hConsoleOutput;
    COORD dwCursorPosition;

    hConsoleOutput = GetStdHandle(STD_OUTPUT_HANDLE);
    dwCursorPosition.X = x;
    dwCursorPosition.Y = y;

    SetConsoleCursorPosition(hConsoleOutput, dwCursorPosition);
}
```

Admin Menu:

Admin menu first authenticate whether, the user put right info(Username, admin) then, it gave options to user to manage inventory, manage customers, manage discounts, etc.

Then redirect them to their respective functions.

```
714  void adminMenu(void) {
715      char username[50];
716      char password[50];
717      int choice;
718  
719      system("cls");
720      gotoxy(20,5);
721      printf("========Authentication For Admin========");
722      gotoxy(18,6);
723      printf("*******************************************\n");
724      gotoxy(24,8);
725      printf("Enter Admin Username: ");
726      fgets(username, sizeof(username), stdin) ;
727      username[ strcspn( username, "\n" ) ] = '\0';
728      fflush(stdin);
729      gotoxy(24,10);
730      printf("Enter Admin Password: ");
731  
732      for (int i = 0; i < 50; i++) {
733          char ch = getch();
734          if (ch != 13) { // If character is not ENTER KEY (\r For Conio) then
735              password[i] = ch;
736              ch = '*';
737              printf("%c", ch); // Visually print start but first store in password
738          }
739          else {
740              password[i] = '\0' ;
741              break;
742          }
```

```
748     if (authenticateAdmin(username, password)) {
749         system("cls");
750         gotoxy(10,3);
751         printf("\033[1;32m========Authentication successful! Welcome, Admin========\033[0m\n");
752         do {
753            gotoxy(20,5);
754             printf("========Admin Menu========");
755             gotoxy(20,7);
756             printf("1. Manage Inventory");
757             gotoxy(20,9);
758             printf("2. Manage Customers\n");
759             gotoxy(20,11);
760             printf("3. Manage Discounts\n");
761             gotoxy(20,13);
762             printf("4. Change Admin Credentials\n");
763             gotoxy(20,15);
764             printf("5. Exit Admin Menu\n");
765             gotoxy(20,17);
766             printf("Enter your choice: ");
767             scanf("%d", &choice);
768             fflush( stdin ) ;
769
```

Manage Inventory:

In this section, Admin can manage inventory, he/she can add items, remove items, update items, also can see whole inventory, etc.

```
void manageInventory(void) {
    int choice;
    do {
        gotoxy(15,3);
        printf("========Inventory Management========\n");
        gotoxy(20,5);
        printf("1. Add Item\n");
        gotoxy(20,7);
        printf("2. Remove Item\n");
        gotoxy(20,9);
        printf("3. Update Item Quantity\n");
        gotoxy(20,11);
        printf("4. Display Inventory\n");
        gotoxy(20,13);
        printf("0. Back to Admin Menu\n");
        gotoxy(20,15);
        printf("Enter your choice: ");
        scanf("%d", &choice);
        fflush( stdin ) ;
```

Manage Customers:

In this section, Admin can manage Customers, he/she can add customers, remove customers, also can Display customers, etc.

```c
void manageCustomers(void) {
    int choice;
    do {
        gotoxy(15,3);
        printf("========Customer Management========");
        gotoxy(20,6);
        printf("1. Add Customer\n");
        gotoxy(20,8);
        printf("2. Remove Customer\n");
        gotoxy(20,10);
        printf("3. Display Customers\n");
        gotoxy(20,12);
        printf("0. Back to Admin Menu\n");
        gotoxy(20,14);
        printf("Enter your choice: ") ;
        scanf("%d", &choice);
        fflush(stdin) ;
```

Manage Discounts:

In this section, Admin can manage Discounts, he/she can add discounts, remove discounts, also can see whole discount list, etc.

```c
void manageDiscounts(void) {
    int choice;
    do {
        gotoxy(12,2);
        printf("========Discount Management========");
        gotoxy(20,4);
        printf("1. Add Discount\n");
        gotoxy(20,6);
        printf("2. Remove Discount\n");
        gotoxy(20,8);
        printf("3. Display Customer Discount List\n");
        gotoxy(20, 10);
        printf("0. Back to Admin Menu\n");
        gotoxy(20,12);
        printf("Enter your choice: ");
        scanf("%d", &choice);
        fflush( stdin ) ;
```

Change Admin Cardinals:

It change the admin Details like, username, password, etc.

```c
void changeAdminKeys( void ) {
    char existingPass[50] ;
    char newUsername[50] ;
    char newPassword[50] ;
    gotoxy( 20, 4 );
    printf("Enter the previous password for admin : ") ;

    for (int i = 0; i < 49; i++) { // Limiting input to 49 characters to avoid buffer overflow
        char ch = getch() ;
        if (ch != 13) {
            existingPass[i] = ch;
            printf("*"); // Masking the input
        }
        else {
            existingPass[i] = '\0'; // Null-terminate the password string (To avoid Buffer overflow)
            break;
        }
    }

    gotoxy( 0 , 7 ) ;
    printf("PLEASE WAIT!!\n\n\t\t\033[1;33;5mVALIDATING CREDENTIALS...\033[0m\n") ;
    Sleep(2250) ;
    if ( strcmp (existingPass, adminUser.password) == 0 ) {
        gotoxy(15, 9) ;
        printf("\033[1;31mNote:-\033[0mChange Password At your own risk.\n") ;
        gotoxy(15, 11) ;
        printf("Enter new admin username: ");
        fgets(newUsername, sizeof(newUsername), stdin ) ;
        newUsername[ strcspn(newUsername, "\n") ] = '\0' ;
```

New Customers:

In old Customer Section, It first collect the information from user than store his data in the customer list then redirect it to Shopping function.

```
1184            case 1:
1185                system("cls") ;
1186                printf("\n\n!!CREATING NEW CUSTOMER ACCOUNT!!") ;
1187                newCustomer.id = customerCount + 1 ;
1188                printf("\n\nEnter customer name: ");
1189                fgets(newCustomer.name, sizeof(newCustomer.name), stdin ) ;
1190                newCustomer.name[ strcspn( newCustomer.name, "\n" ) ] = '\0' ;
1191                printf("\n\nEnter customer phone number: ");
1192                fgets(newCustomer.phoneNumber, sizeof(newCustomer.phoneNumber), stdin ) ;
1193                newCustomer.phoneNumber[ strcspn( newCustomer.phoneNumber, "\n" ) ] = '\0' ;
1194                fflush( stdin );
1195                printf("\n\nSet PIN for customer authentication: ");
1196                scanf("%d", &newCustomer.pin);
1197                fflush( stdin );
1198
1199                for(int i = 0; i < customerCount; ++i) {
1200                    if( strcmp ( newCustomer.phoneNumber , customers[i].phoneNumber ) == 0 ) {
1201                        gotoxy(15, 18) ;
1202                        // ANSI escape code for red text
1203                        printf("\033[1;31m%s Phone_Number is already registered.\n\t\t\t\tPlease Try Again!\033[0m\n", newCustomer.phoneNumber ) ;
1204                        return 1 ;
1205                    }
1206                }
1207
1208                system("cls");
1209                printf("PLEASE WAIT....\n\nYOUR DATA IS BEING PROCESSED....");
1210                Sleep( 1500 ) ;
1211
1212                gotoxy(30, 10);
1213
1214                printf("ACCOUNT CREATED SUCCESSFULLY....");
1215                gotoxy(0, 20);
1216                addCustomer(newCustomer);
1217                printf("Press enter to login");
1218
1219                getch();
1220                // Perform shopping for the new customer
1221                system("cls") ;
1222                shopping(newCustomer);
1223                break;
```

Old Customer:

In old Customer Section, It first collect the information from user than authenticate the data whether user is an old customer or not, if user's data found in the customer list then redirect it to Shopping function.

```c
case 2:
    system("cls") ;
    printf("\n\n!!LOGGING IN CURRENT CUSTOMER!!") ;
    printf("\n\nEnter customer phone number: ");
    fgets(customerPhoneNum, sizeof(customerPhoneNum), stdin) ;
    customerPhoneNum[ strcspn( customerPhoneNum, "\n" ) ] = '\0' ;
    fflush( stdin ) ;
    printf("\n\nEnter PIN for authentication: ");
    scanf("%d", &pin);
    fflush( stdin );

    // Authenticate existing customer
    int customerIndex = -1;
    for (int i = 0; i < customerCount; ++i) {
        if ( ( strcmp ( customers[i].phoneNumber , customerPhoneNum ) == 0 ) && customers[i].pin == pin) {
            customerIndex = i;
            break;
        }
    }

    system("cls");
    printf("PLEASE WAIT....\n\nFETCHING ACCOUNT DETAILS....");
    Sleep( 1500 ) ;
    if (customerIndex != -1) {
        gotoxy(30, 10);
        printf("\033[1;32mAuthentication successful! Welcome back, %s.\033[0m\n", customers[customerIndex].name);
        gotoxy(0 , 20) ;
        printf("Enter any key to login ...\n") ;
        getch() ;
        system("cls") ;
        // Perform shopping for the existing customer
        shopping(customers[customerIndex]);
    }
    else {
        system("cls") ;
        gotoxy(0, 21) ;
        printf("\033[1;31mAuthentication failed! Access denied.\033[0m\n");
    }
    break;
```

Shopping Function:

The shopping function display the user options like, display whole inventory, display section wise inventory, Add items to Cart, Remove Items from Cart, Display Cart, Checkout and exit shopping.

When user select any option then it perform that task.

```
854   // Function for shopping
855   void shopping(struct Customer customer) {
856       int choice;
857       int customerId = customer.id;
858       double discount = getDiscount(customerId); // Get customer's discount
859       struct Item cart[MAX_ITEMS]; // Array to hold items in the cart
860       int cartItemCount = 0; // Variable to track the number of items in the cart
861       char *NewCustomer = customer.name ;
862
863       do {
864           gotoxy(25, 3);
865           printf("Shopping-Menu:\n");
866           gotoxy(25, 4);
867           printf("_____________") ;
868           gotoxy( 20, 7 ) ;
869           printf("1. Display Whole-Inventory\n");
870           gotoxy( 20, 9 ) ;
871           printf("2. Display Section Wise Items\n") ;
872           gotoxy( 20, 11 ) ;
873           printf("3. Add Item to Cart\n");
874           gotoxy( 20, 13 ) ;
875           printf("4. Remove Item from Cart\n");
876           gotoxy( 20, 15 ) ;
877           printf("5. Display Cart\n");
878           gotoxy( 20, 17 ) ;
879           printf("6. Checkout\n");
880           gotoxy( 20, 19 ) ;
881           printf("0. Exit Shopping\n");
882           gotoxy( 20, 21 ) ;
883           printf("\nEnter your choice: ");
884           scanf("%d", &choice);
885           fflush(stdin);
886
```

Display Inventory:

It display all the items currently available in the inventory.

```
switch (choice) {
    case 1:
        system("cls") ;
        displayInventory();
        // 33 for yellow color and 5 for blinking effect
        printf("\n\033[1;33;5mPress any key to exit viewing.\033[0m\n") ;
        getch( ) ;
        system("cls") ;
        break;
```

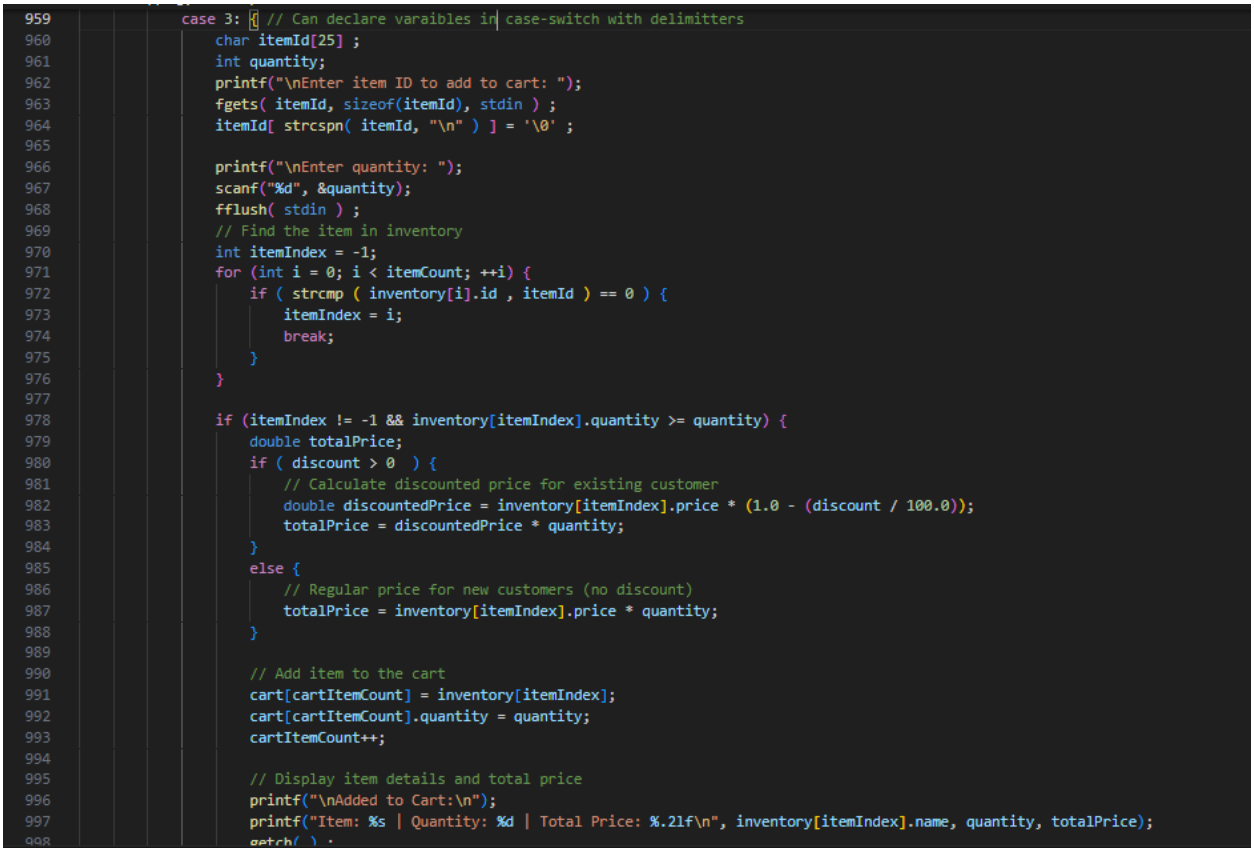
Display section wise inventory:

It gives the user options of different sections in the inventory (like, Dairy, grains, beverages, snacks, etc.) If user select any category it shows the items available in that Category.

```
case 2: {
    system("cls") ;
    int category_choice ;
    gotoxy( 10, 1 ) ;
    printf("Section-Wise Inventory\n") ;
    gotoxy( 10, 2 ) ;
    printf("_____\n\n") ;
    printf("1. Dairy\n\n") ;
    printf("2. Grains_and_cooking_products\n\n") ;
    printf("3. Snacks\n\n") ;
    printf("4. Beverages\n\n") ;
    printf("5. Others Section\n\n") ;
    printf("Enter the option_number: ") ;
    scanf("%d", &category_choice ) ;
    fflush( stdin ) ;
```

Add items to Cart:

It ask user to enter the Id number of item and quantity and if, the Id number to any item in the product list, and user's desired quantity available in the inventory than, it add that item in the Cart and display the details.



```
959         case 3: { // Can declare varaibles in case-switch with delimitters
960             char itemId[25] ;
961             int quantity;
962             printf("\nEnter item ID to add to cart: ");
963             fgets( itemId, sizeof(itemId), stdin ) ;
964             itemId[ strcspn( itemId, "\n" ) ] = '\0' ;
965 
966             printf("\nEnter quantity: ");
967             scanf("%d", &quantity);
968             fflush( stdin ) ;
969             // Find the item in inventory
970             int itemIndex = -1;
971             for (int i = 0; i < itemCount; ++i) {
972                 if ( strcmp ( inventory[i].id , itemId ) == 0 ) {
973                     itemIndex = i;
974                     break;
975                 }
976             }
977 
978             if (itemIndex != -1 && inventory[itemIndex].quantity >= quantity) {
979                 double totalPrice;
980                 if ( discount > 0  ) {
981                     // Calculate discounted price for existing customer
982                     double discountedPrice = inventory[itemIndex].price * (1.0 - (discount / 100.0));
983                     totalPrice = discountedPrice * quantity;
984                 }
985                 else {
986                     // Regular price for new customers (no discount)
987                     totalPrice = inventory[itemIndex].price * quantity;
988                 }
989 
990                 // Add item to the cart
991                 cart[cartItemCount] = inventory[itemIndex];
992                 cart[cartItemCount].quantity = quantity;
993                 cartItemCount++;
994 
995                 // Display item details and total price
996                 printf("\nAdded to Cart:\n");
997                 printf("Item: %s | Quantity: %d | Total Price: %.2lf\n", inventory[itemIndex].name, quantity, totalPrice);
998                 getch( ) ;
```

Remove Items from Cart:

It ask user to enter the Id number of item and if, the Id number match to any item in the cart, it remove that item.

```
1012            case 4: {
1013                if (cartItemCount == 0) {
1014                    system("cls") ;
1015                    printf("\n\033[1;31mCart is empty. Unable to remove.\033[0m\n");
1016                }
1017                else {
1018                    char itemIdToRemove[25];
1019                    printf("\nEnter item ID to remove from cart: ");
1020                    fgets(itemIdToRemove, sizeof(itemIdToRemove), stdin);
1021                    itemIdToRemove[strcspn(itemIdToRemove, "\n")] = '\0';
1022
1023                    int removeIndex = -1;
1024                    for (int i = 0; i < cartItemCount; ++i) {
1025                        if (strcmp(cart[i].id, itemIdToRemove) == 0) {
1026                            removeIndex = i;
1027                            break;
1028                        }
1029                    }
1030
1031                    system("cls") ;
1032                    if (removeIndex != -1) {
1033                        // Shift elements to remove the specified item from the cart
1034                        for (int i = removeIndex; i < cartItemCount - 1; ++i) {
1035                            cart[i] = cart[i + 1];
1036                        }
1037                        cartItemCount--;
1038
1039                        printf("\n\033[1;32m%s-Item removed from cart successfully.\033[0m\n", itemIdToRemove );
1040                    }
1041                    else {
1042                        printf("\n\033[1;31m%s Item not found in the cart.\033[0m\n", itemIdToRemove );
1043                    }
1044                }
1045                break;
1046            }
```

Display Cart:

It Display the items in the Cart with all Details.

```
case 5: {
    if (cartItemCount == 0) {
        printf("\n\n\033[1;31mCart is empty.\033[0m\n");
    }
    else {
        int row = 29 ;
        gotoxy( 55, 25 ) ;
        printf("Cart Contents:\n");
        gotoxy( 55, 26 ) ;
        printf("_______________\n") ;
        gotoxy( 45, row ) ;
        printf("ID     |     Name     |  Price  | Quantity\n") ;
        double totalCartPrice = 0.0;
        for (int i = 0; i < cartItemCount; ++i) {
            gotoxy( 45, ++row ) ;
            double itemPrice ;
            if ( discount > 0 ) {
                double discountedPrice = cart[i].price * (1.0 - (discount / 100.0));
                itemPrice = discountedPrice * cart[i].quantity ;
                printf("%-6s | %-12s | %-7.21f | %-3d\n", cart[i].id, cart[i].name, itemPrice, cart[i].quantity);
                totalCartPrice += itemPrice ;
            }
            else {
                itemPrice = cart[i].price * cart[i].quantity;
                printf("%-6s | %-12s | %-7.21f | %-3d\n", cart[i].id, cart[i].name, cart[i].price, cart[i].quantity);
                totalCartPrice += cart[i].price ;
            }
        }
        printf("\033[1;33;5mTotal Cart Price: %.21f\nPress any key to move back.\033[0m\n", totalCartPrice) ;
        getch( ) ;
        system("cls") ;
    }
    break;
}
```

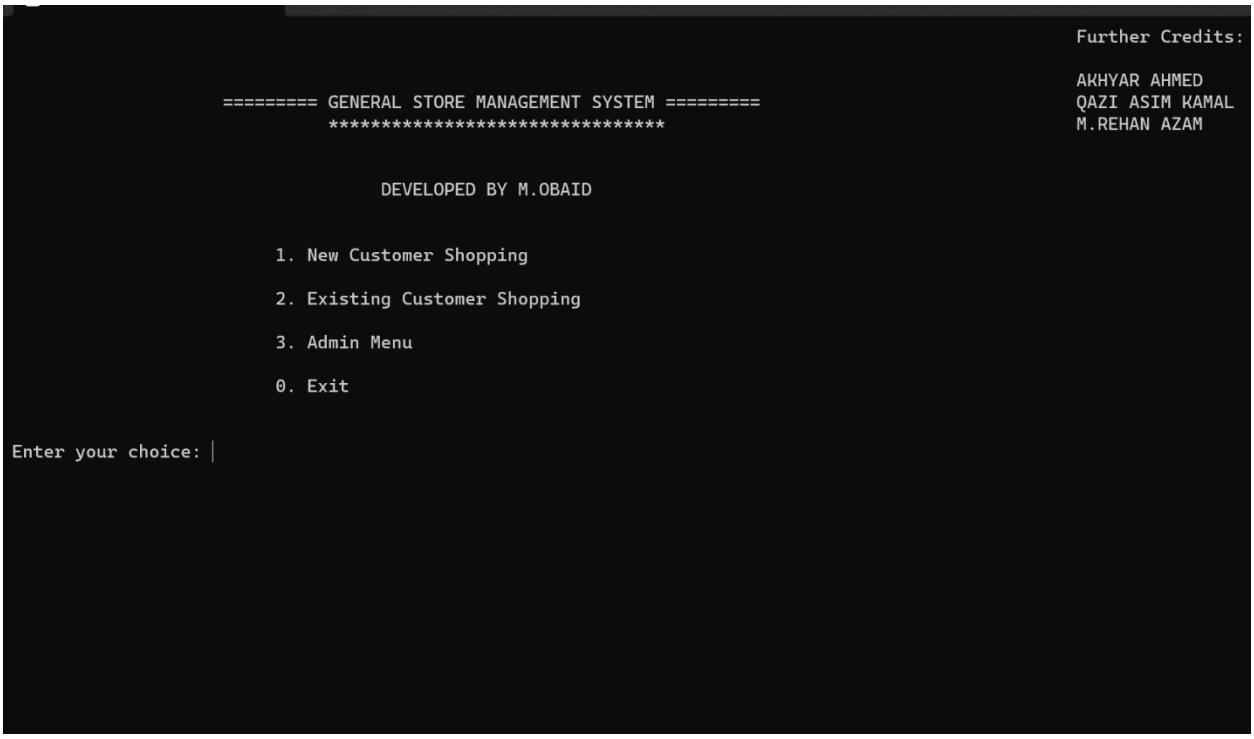
Checkout:

It Display the items in the Cart with all Details and display the total payable amount then complete checkout process.

```
case 6: {
    system("cls") ;
    if (cartItemCount == 0) {
        printf("\033[1;31mCart is empty. Unable to checkout.\033[0m\n");
    }
    else {
        int row = 5 ;
        printf("\nCheckout Process:\n");
        printf("\nItems Purchased are:\n");
        gotoxy( 15, row ) ;
        printf("ID     |     Name     |  Price  | Quantity\n") ;
        double totalCheckoutPrice = 0.0;
        double checkoutPrices[cartItemCount] ;
        for (int i = 0; i < cartItemCount; ++i) {
            gotoxy( 15 , ++row ) ;
            if ( discount > 0 ) {
                double discountedPrice = cart[i].price * (1.0 - (discount / 100.0));
                checkoutPrices[i] = discountedPrice * cart[i].quantity ;
                printf("%-6s | %-12s | %-7.21f | %-3d\n", cart[i].id, cart[i].name, checkoutPrices[i], cart[i].quantity);
            }
            else {
                checkoutPrices[i] = cart[i].price * cart[i].quantity;
                printf("%-6s | %-12s | %-7.21f | %-3d\n", cart[i].id, cart[i].name, cart[i].price, cart[i].quantity);
            }
            totalCheckoutPrice += checkoutPrices[i] ;
        }
        gotoxy(0 , ++row ) ;
        if( discount > 0 )
            printf("\n\033[1;31mTotal Checkout Price(With %1f%%-Discount): %.21f\033[0m\n",discount, totalCheckoutPrice) ;
        else
            printf("\n\033[1;31mTotal Checkout Price: %.21f\033[0m\n", totalCheckoutPrice) ;

        // Perform checkout - Reset the cart
        gotoxy(0 , ++row ) ;
        printf("\n\033[1;33;5mCheckout completed. Thanks you for shopping!\nWe hope to see you again soon!\033[0m(Regards Admin Obaid)\n") ;
        getch( ) ;
        system("cls") ;
```

First Time User Interface: -



Tools used:

Visual Studio Code + C