Programming and Data Structures I
CS3101

Course Project

On

Online E-Commerce System

(C-Kart)

by

Abhay Kshirsagar*(19MS172)

Mukesh Chaudhari*(19MS051)

Parth Bibekar*(19MS161)

Juee Dhar*(19MS102)[1]

(Group no. 19)

Course Instructor

Dr. Kripabandhu Ghosh

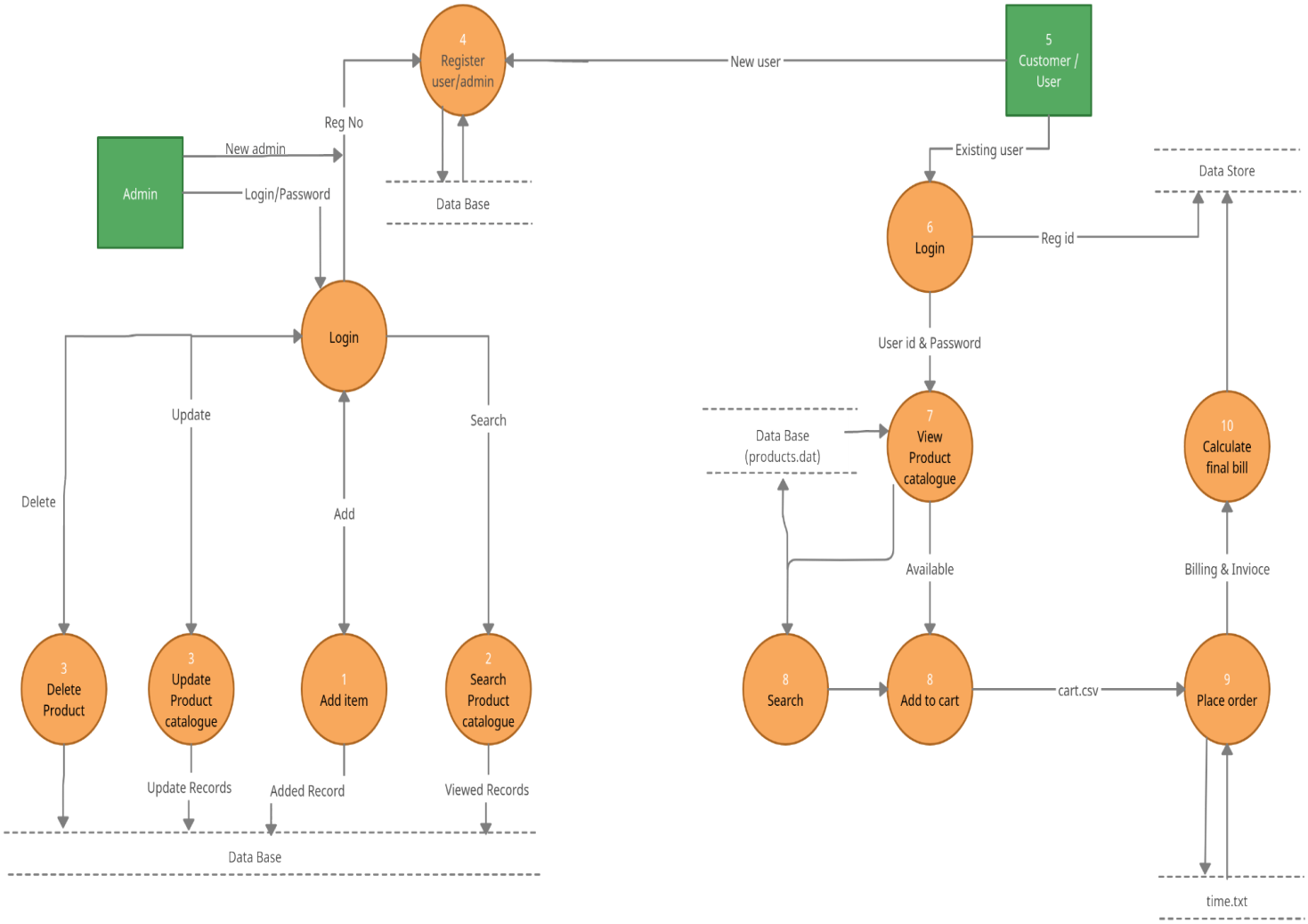
Department of Computer Science and Application

Indian Institute of Science Education and Research Kolkata

[1] * = equal contribution

1. Program workflow

The Complete Workflow of the Program is given in the flowchart below.



2. Compiling Instructions

User Manual (video) : https://youtu.be/Uuicn0ErnpE
Use GNU C Compiler for compiling the code. We Recommend that you use **gcc version 6.3.0 or 11.1.0 (These are the versions we tested on)** . The Program works better if you compile the code from the terminal using the following Command.

```
$ gcc main.c -o main #For Compiling
$ ./main #For Running the Code
```

The Code runs well on Windows as well as in Linux but there is an issue when Compiling it on Dev C++. Both the above commands can be executed on VS Code terminal which can be accessed by pressing Ctrl + `.

3. Login Functions

This module implements the login functionality in the online e-commerce system. Also it lets new users register into the website and login to the system once registered. The admin can also login to the system to make necessary changes in data. All the important functions used for logging into the system are listed below along with their details:

3.1. Structure Data

Username and Password are stored in two different files (one for admin and another for user) using this structure. The following structure variables are used to store the necessary data:

- char uname : stores username.
- char pass : stores password.
- int admin : exclusively for admin, it lets the program know that entered username and password is of admin.

```
typedef struct logins
{
    char uname[20];
    int admin;
    char pass[20];

}LOGIN;
```

3.2. login()

This function will display all the ways one can login into the online e-commerce system (admin or user). As per admin menu or user menu this function can also be called as “login menu”.
It lets a person login into the online e-commerce system after entering valid choices mentioned along with it :

1. Admin
2. User
0. Exit

```
****************************************
|        LOGIN MENU                    |
|                                      |
|   Give Option                        |
|   [1] Admin                          |
|   [2] User                           |
|   [0] Exit                           |
|                                      |
|   Option :                           |
|                                      |
****************************************
```

3.3. admin_login()

admin_login() is a login menu exclusively for admin (It lets new admins register into the website. It lets newly registered admin or existing admin to sign in into the system). It lets the admin perform following actions after entering valid choices mentioned along with it :

1. Sign In
2. Register
0. Exit

```
****************************************
|     WELCOME TO ADMIN LOGIN           |
|                                      |
|   Give Option                        |
|   [1] Sign In                        |
|   [2] Register                       |
|   [0] Exit                           |
|                                      |
|   Option :                           |
|                                      |
****************************************
```

3.4. user_login()

user_login() is a login menu exclusively for users (It lets new users register into the website. It lets newly registered users or existing users sign in into the system). It lets the user perform following actions after entering valid choices mentioned along with it :

1. Sign In
2. Register
0. Exit

```
*****************************************
|        WELCOME TO USER LOGIN          |
|                                       |
|   Give Option                         |
|   [1] Sign In                         |
|   [2] Register                        |
|   [0] Exit                            |
|                                       |
|   Option :                            |
|                                       |
*****************************************
```

3.5. register_admin()

Return type : void

This function will be called when the admin enters choice “2” in admin_login.
This function will let new admin register into the system.
Firstly, it asks the admin to enter a secret code (“Secret”).
If admin fails to enter correct secret code he will be redirected to admin_login().
If only the admin enters the secret code correctly he will be asked to enter a new username and new password to be registered. Once the admin is registered the admin will return to the admin_login() screen and after choosing “1”, the admin can sign in and will be redirected to admin_menu.

```
void register_admin() {
    system(CLEAR);
    splash_screen();
    gotoxy(7,2);
    printf("WELCOME TO ADMIN REGISTRATION\n");
    char secret[] = "Secret";
    char input_secret[20];
    gotoxy(3, 4);
    printf("Enter Secret Code : ");
    scanf("%[^\n]s", input_secret);
```

```c
    if(strcmp(secret, input_secret) == 0) {
        // printf("Entered Code is Valid %d",strcmp(secret, input_secret));
        gotoxy(3,5);
        printf("Secret Code is Valid");
        LOGIN l_1;
        scanf("%c", temp);
        fptr = fopen("login_a.dat", "a");
        gotoxy(3,7);
        printf("Enter a USERNAME :");
        scanf("%[^\n]s", l_1.uname);
        scanf("%c",temp);
        gotoxy(3,8);
        printf("Enter a PASSWORD :");
        scanf("%[^\n]s", l_1.pass);
        l_1.admin = 1;
        fwrite(&l_1, sizeof(LOGIN), 1, fptr);
        fclose(fptr);
    } else {
        gotoxy(3,5);
        printf("Secret Code is Invalid. Press Enter\n");
        scanf("%c",temp);
    }
}
```

```
*****************************************
|      WELCOME TO ADMIN REGISTRATION     |
|                                        |
| Enter Secret Code : Secret             |
| Secret Code is Valid                   |
|                                        |
| Enter a USERNAME :username             |
| Enter a PASSWORD :password             |
|                                        |
|                                        |
*****************************************
```

3.6. register_user()

Return type : void

This function will be called when the user enters choice “2” in user_login.
This function will let new users register into the system.
This is a simple registration function. It only asks users to enter a new username and new password to be registered. Once the user is registered the user will return to user_login() screen and after choosing “1”, the user can sign in and will be redirected to user_menu.

```
void register_user() {
    system(CLEAR);
    splash_screen();
    gotoxy(7,2);
    printf("WELCOME TO USER REGISTRATION\n");
    LOGIN l_1;
    // scanf("%c", temp);

    fptr = fopen("login_u.dat", "a");
    gotoxy(3,5);
    printf("Enter a USERNAME : ");
    scanf("%[^\n]s", l_1.uname);
    scanf("%c",temp);
    gotoxy(3,7);
    printf("Enter a PASSWORD : ");
    scanf("%[^\n]s", l_1.pass);
    l_1.admin = 0;
    fwrite(&l_1, sizeof(LOGIN), 1, fptr);
    fclose(fptr);
}
```

```
*****************************************
|      WELCOME TO USER REGISTRATION      |
|                                        |
|                                        |
| Enter a USERNAME : username            |
|                                        |
| Enter a PASSWORD : password            |
|                                        |
|                                        |
|                                        |
*****************************************
```

3.7. signin()

Return type : void Argument type : int

This function will be called when the admin enters choice “1” in admin_login.
And also it will be called when the user enters choice “1” in user_login.
It will ask the admin (user) to enter username and password and store them in temporary variables. The program will compare username with uname and password with pass of the login structure read from file “login_a.dat” (“login_u.dat”). If both of them are correct then admin (user) will be redirected to admin_menu (user_menu). Else it will print a message saying “username or password is incorrect”.
Existing credentials in “login_a.dat” (For admin) :
Username : admin
Password : pass

Existing credentials in “login_u.dat” (For user) :
Username : user
Password : pass

```
void signin(int n) {
    system(CLEAR);
    splash_screen();
    gotoxy(15,2);
    printf("Sign In Page\n");
    LOGIN l_1;
    char username[20], password[20];
    int flag = 0;
    int admin_val=0;
    if (n == 1) {
        fptr = fopen("login_a.dat", "r");
    } else {
        fptr = fopen("login_u.dat", "r");
    }

    gotoxy(3,4);
    printf("Enter USERNAME : ");
    scanf("%[^\n]s", username);
    scanf("%c",temp);
    gotoxy(3,5);
    printf("Enter PASSWORD : ");
    scanf("%[^\n]s", password);
    scanf("%c",temp);

    while(fread(&l_1, sizeof(LOGIN), 1, fptr)) {
```

```
        if ((strcmp(l_1.uname, username) == 0) && (strcmp(l_1.pass, password)
== 0)) {
            flag = 2;
        } else {
            flag=0;
        }
        if (flag==2) {
            admin_val = l_1.admin;
            break;
        }

    }
    gotoxy(3,8);
    if (flag == 0)
    {
        printf("USERNAME OR PASSWORD IS INCORRECT\n");
        printf("Press Enter to go back.");
        scanf("%c",temp);
    } else if (flag == 2) {
        printf("LOGIN Successful Press enter\n");
        if (admin_val == 1) {
            // Go To ADMIN MENU
            scanf("%c",temp);
            admin_menu();
        } else {
            //Go To USER MENU
            scanf("%c",temp);
            user_menu();
        }
    }
    fclose(fptr);
}
```

```
****************************************
|              Sign In Page            |
|                                      |
| Enter USERNAME : admin               |
| Enter PASSWORD : pass                |
|                                      |
|                                      |
| LOGIN Successful Press enter         |
|                                      |
|                                      |
****************************************
```

```
****************************************
|              Sign In Page            |
|                                      |
| Enter USERNAME : admin               |
| Enter PASSWORD : pass                |
|                                      |
|                                      |
| LOGIN Successful Press enter         |
|                                      |
|                                      |
****************************************
```

4. Admin Functions

This Module implements the admin side of the online e-commerce system. All the important functions used to implement the admin interface are listed below along with their details :

4.1. Structure Data

Information about Products is stored using structure variables. Product structure contains following variables:

- char product_name : stores the name of the product.
- char description : stores product details which helps you know more about the product.
- int id : stores unique product id.
- int quantity : stores the available quantity of product.
- int price : stores the final price of the product.

```
typedef struct product
{
    char product_name[20];
    int id;
    int quantity;
```

```
    float price;
    char description[100];
}Product;
```

4.2. admin_menu()

This function will display the admin menu after logging into the admin section of the online e-commerce system. It lets you perform the following actions after entering valid choices mentioned along with it :

1. display
2. append
3. search product
4. update product
5. delete product
6. Exit

```
            Welcome to the E-Commerce Application

  [1] Display Product Catalouge
  [2] APPEND A PRODUCT
  [3] SEARCH PRODUCT
  [4] UPDATE PRODUCT
  [5] DELETE PRODUCT
  [0] EXIT



            Enter choice :
```

4.3. display()

Return type : void

This function will be called when admin enters choice “1” in admin_menu.
The file “product.dat” is opened in mode “r” (Reading mode). The while loop will continue till it reads a line from the file, loop will stop when there will be no line left to read (fread will become NULL). At every iteration we read a line according to the structure of the product and display all the information about the product.

```
void display() {
    system(CLEAR);
```

```
    fptr = fopen("products.dat","r");

    printf("ID |Name  |  Des ");
    while (fread(&p_1,sizeof(Product), 1,fptr))
    {
        printf("\n%-5d|%-20s|%-50s|%8.2f|%4d", p_1.id, p_1.product_name,
p_1.description, p_1.price, p_1.quantity);
    }
    fclose(fptr);
    printf("\n");
    int flag = 0;
    printf("Go Back (1 = Yes, 0 = No) : ");
    scanf("%d",&flag);
    while (!flag)
    {
        printf("Go Back (1 = Yes, 0 = No) : ");
        scanf("%d",&flag);
    }}
```

```
    :::::::::::::::::::::::::::::::::::: Product catalogue ::::::::::::::::::::::::::::::::::::::::
    ==============================================================================================
    ID   |Name                |Description                                        |Price   |Quantity
    ==============================================================================================
    1    |Corn Flakes         |Nourishing and Tasty ready-2-eat breakfast         |   20.00|      40
    2    |Lays                |Tasty Snack                                        |   12.00|      55
    3    |Monster Energy      |A can of the meanest energy drink on the planet    |   60.45|      60
    4    |Heinz Can of Beans  |Its Everywhere                                     |   45.23|      12
    5    |Pepsi               |Soft Drink Better than Cola                        |   23.00|      12
    6    |mukesh              |wow                                                |213241.00|       1
Go Back (1 = Yes, 0 = No) :
```

4.4. append()

Return type : void

This function will be called when admin enters choice “2” in admin_menu.

This function will be used by the admin to add any new product which does not exist previously in the “product.dat” file. This file will be opened in mode “a” (Opens the file for writing in append mode and the program will start appending entered information in the file). Admin will be asked to enter product information according to the product structure(Name, Description, ID, Price, Quantity).New product information will be added to the end of the file.

```c
void append() {
    system(CLEAR);
    char temp[2];
    int n = 0,i, flag = 0, idflag = 1, app = 0;
    Product p;
    do{
        gotoxy(20, 2);
        printf("Do you want to append a new product \n(Yes = 1, No = 0 [GO
BACK])): ");
        scanf("%d",&app);
        if (app == 1) {
            fptr = fopen("products.dat","a");

            do {
                gotoxy(20,4);
                printf("Enter ID : ");
                scanf("%d", &p.id);
                if (check_id(p.id)) {
                    printf("This id is already present use other id\n");
                    idflag = 1;
                } else {
                    idflag = 0;
                }
            } while(idflag != 0);
            gotoxy(0,5);
            for (int k = 0; k < 41; k++)
            {
                printf(" ");
            }
            printf(" ");
            scanf("%c", temp);
            gotoxy(20,5);
            printf("Enter Name : ");
            scanf("%[^\n]s", p.product_name);
            scanf("%c", temp);
            gotoxy(20,6);
            printf("Enter the Product Description : ");
            scanf("%[^\n]s", p.description);
            gotoxy(20,7);
            printf("Enter the Product Price (float) : ");
            scanf("%f", &p.price);
            gotoxy(20,8);
            printf("Enter the Product Quantity (int) : ");
            scanf("%d", &p.quantity);
            fwrite(&p,sizeof(Product), 1, fptr);
```

```
            fclose(fptr);
            n++;
            gotoxy(10,10);
            printf("Appended %d value/s to the database\n", n);
            system(CLEAR);
        } else {
            flag = 1;
        }
    }while(flag != 1);
}
```

```
                    Do you want to append a new product
(Yes = 1, No = 0 [GO BACK])): 1
                    Enter ID : 7
                    Enter Name : Mobile
                    Enter the Product Description : Use anywhere
                    Enter the Product Price (float) : 24999.00
                    Enter the Product Quantity (int) : 10

          Appended 1 value/s to the database
```

4.5. search_product()

Return type : void

This function will be called when admin enters choice “3” in admin_menu.
Admin will be asked to enter the ID of the product which is to be searched. The “product.dat” file will be opened in mode “r” (Reading mode) and the variable found will be maintained at zero till matching ID is found (found=1). When found becomes one all the information of the product with given ID will be displayed on the admin screen. Even after going through the whole file if the variable found remains zero then a message saying “Record not found” will be displayed. In the end a loop asking to go back will be initiated for holding the screen till the admin wants. Till the admin enters 1 it will keep asking to go back.

```
void search_product() {

    int id, found = 0;
    fp = fopen("products.dat","r");
    printf("Enter the Product ID to Search : ");
    scanf("%d", &id);

    while (fread(&p_1,sizeof(Product), 1,fp))
    {
        if (p_1.id == id) {
```

```
            found = 1;
            printf("%-5d|%-20s|%-50s|%8.2f|%4d\n", p_1.id, p_1.product_name,
p_1.description, p_1.price, p_1.quantity);
        }
    }
    if(!found) {
        printf("\nRecord not found\n");
    }
    fclose(fp);

    int flag = 0;
    printf("Go Back (1 = Yes, 0 = No) : ");
    scanf("%d",&flag); while (!flag)
    {
        scanf("%d",&flag);
    }
}
```

```
Enter the Product ID to Search : 4
    :::::::::::::::::::::::::::::::::: Product catalogue ::::::::::::::::::::::::::::::::::::::
    ==========================================================================================
    ID   |Name                |Description                                        |Price   |Quantity
    ==========================================================================================
    4    |Heinz Can of Beans  |Its Everywhere                                     |   45.23|      12
Go Back (1 = Yes, 0 = No) :
```

4.6. update_product()

Return type : void

This function will be called when admin enters choice “4” in admin_menu.
Firstly the main file “product.dat” is opened in mode “r”(Reading mode) and a temporary file “temp.dat” is opened in mode “w” (writing mode). The admin will be asked to enter the ID of the product to be updated. Just like the search function matching ID will be searched in the main file.
If found = 1 or a matching ID is found, admin will be asked to enter a New name, description, price and quantity and it will be written to the temporary file, otherwise it keeps writing structures from main file to temporary file.
If found = 0 a message saying “Record not found” will be displayed. Else again the main file and temporary file will be opened but now in writing mode and reading mode respectively. And now the program will copy all data from temporary file to main file.
So, finally we will have updated main file.

```
void update_product() {
```

```c
    // ft is the temporary file for updating
    int id, found = 0;
    char temp[2];
    fp = fopen("products.dat","r");
    ft = fopen("temp.dat","w");
    printf("Enter the Product ID to Update : ");
    scanf("%d", &id);

    while (fread(&p_1,sizeof(Product), 1,fp))
    {
        if (p_1.id == id) {
            found = 1;
            scanf("%c",temp);
            printf("Enter Name : ");
            scanf("%[^\n]s", p_1.product_name);
            scanf("%c",temp);
            printf("Enter the Product Description : ");
            scanf("%[^\n]s", p_1.description);
            printf("Enter the Product Price (float) : ");
            scanf("%f", &p_1.price);
            printf("Enter the Product Quantity (int) : ");
            scanf("%d", &p_1.quantity);
        }
        fwrite(&p_1, sizeof(Product), 1, ft);
    }
    fclose(ft);
    fclose(fp);


    if(!found) {
        printf("\nRecord not found\n");
    } else {
        ft = fopen("temp.dat","r");
        fp = fopen("products.dat", "w");
        while (fread(&p_1, sizeof(Product), 1,ft)) {
            fwrite(&p_1, sizeof(Product), 1, fp);
        }

        fclose(fp);
        fclose(ft);

    }
}
```

```
    :::::::::::::::::::::::::::::::::: Product catalogue ::::::::::::::::::::::::::::::::::::::::
    ==========================================================================================
    ID   |Name                |Description                                        |Price   |Quantity
    ==========================================================================================
    1    |Corn Flakes         |Nourishing and Tasty ready-2-eat breakfast         |   20.00|      40
    2    |Lays                |Tasty Snack                                        |   12.00|      55
    3    |Monster Energy      |A can of the meanest energy drink on the planet    |   60.45|      60
    4    |Heinz Can of Beans  |Its Everywhere                                     |   45.23|      12
    5    |Pepsi               |Soft Drink Better than Cola                        |   23.00|      12
    6    |mobile              |use anywhere                                       |24999.00|      10
Enter the Product ID to Update : 6
Enter Name : mobile
Enter the Product Description : use wherever
Enter the Product Price (float) : 23000.00
Enter the Product Quantity (int) : 5
```

4.7. delete_product()

Return type : void

This function will be called when admin enters choice “5” in admin_menu.
This function works in the same way as the update_product() function works. It asks the admin to enter the ID of the product to be deleted. Again the main file “product.dat” will be opened in mode ”r” (Reading mode) and temporary file “temp.dat” will be opened in mode “w” (writing mode). A while loop will read structures from the main file and keep writing those to the temporary file until the record with entered ID is found. If a matching ID is found, a variable found will be set at one and that structure will not be written to the temporary file.
If found = 0, then a message saying “Record not found” will be displayed.
Else again the main file and temporary file will be opened in writing mode and reading mode respectively. And all data from the temporary file will be copied to the main file.
So, finally we will have updated main file without the product information corresponding to the entered ID.

```c
void delete_product() {

    int id, found = 0;
    char temp[2];
    fp = fopen("products.dat","r");
    ft = fopen("temp.dat","w");
    printf("Enter the Product ID to Delete : ");
    scanf("%d", &id);

    while (fread(&p_1,sizeof(Product), 1,fp))
    {
        if (p_1.id == id) // ignore that id
```

```c
            found = 1;
        else
            fwrite(&p_1, sizeof(Product), 1, ft);
    }
    fclose(ft);
    fclose(fp);


    if(!found) {
        printf("\nRecord not found\n");
    } else {
        ft = fopen("temp.dat","r");
        fp = fopen("products.dat", "w");
        while (fread(&p_1, sizeof(Product), 1,ft)) {
            fwrite(&p_1, sizeof(Product), 1, fp);
        }

        fclose(fp);
        fclose(ft);

    }
}
```

```
    :::::::::::::::::::::::::::::::::::: Product catalogue ::::::::::::::::::::::::::::::::::::::::
    ==============================================================================================
    ID   |Name               |Description                                        |Price   |Quantity
    ==============================================================================================
    1    |Corn Flakes        |Nourishing and Tasty ready-2-eat breakfast         |   20.00|      40
    2    |Lays               |Tasty Snack                                        |   12.00|      55
    3    |Monster Energy     |A can of the meanest energy drink on the planet    |   60.45|      60
    4    |Heinz Can of Beans |Its Everywhere                                     |   45.23|      12
    5    |Pepsi              |Soft Drink Better than Cola                        |   23.00|      12
    6    |mobile             |use wherever                                       |23000.00|       5
Enter the Product ID to Update : 6
```

5. User Functions

5. 1 void usermenu()

Return type: void

This is the first user function after logging in through user_login(). The interface is designed with gotoxy(int x, int y). It uses scanf() to get input from the user and then redirects the user to other functions using switch().

```
void usermenu(){
   int choice;
   system("clear");
   gotoxy(27,3);
   printf("MAIN MENU ");
   gotoxy(20,5);
   printf(" 1. View catalogue  ");
   gotoxy(20,7);
   printf(" 2. Search Products ");
   gotoxy(20,9);
   printf(" 3. Close Application");
   gotoxy(20,13);
   printf("Enter your choice:");
   scanf("%d",&choice);
   switch(choice)
   {
       case 1:{
           displayProduct();
           break;
       }
       case 2:{
           search();
           break;
       }
       case 3:{
           break;}
       default:{
           gotoxy(10,23);
           printf("\nInvalid entry!!Please re-enter a valid option. Redirecting
to main menu");
           sleep(2);
           usermenu();   }}}
```

```
MAIN MENU

1. View catalogue

2. Search Products

3. Close Application


Enter your choice:
```

5. 2 int displayProduct()

Return type: int

displayProduct() is a simple function that reads the “products.dat” binary file created by the admin and beautifully prints the product catalogue in the format of a table. It gives the user four choices to either buy the product by entering the product id and the quantity, or to search a product, or to view user’s cart or to go back to the main menu.

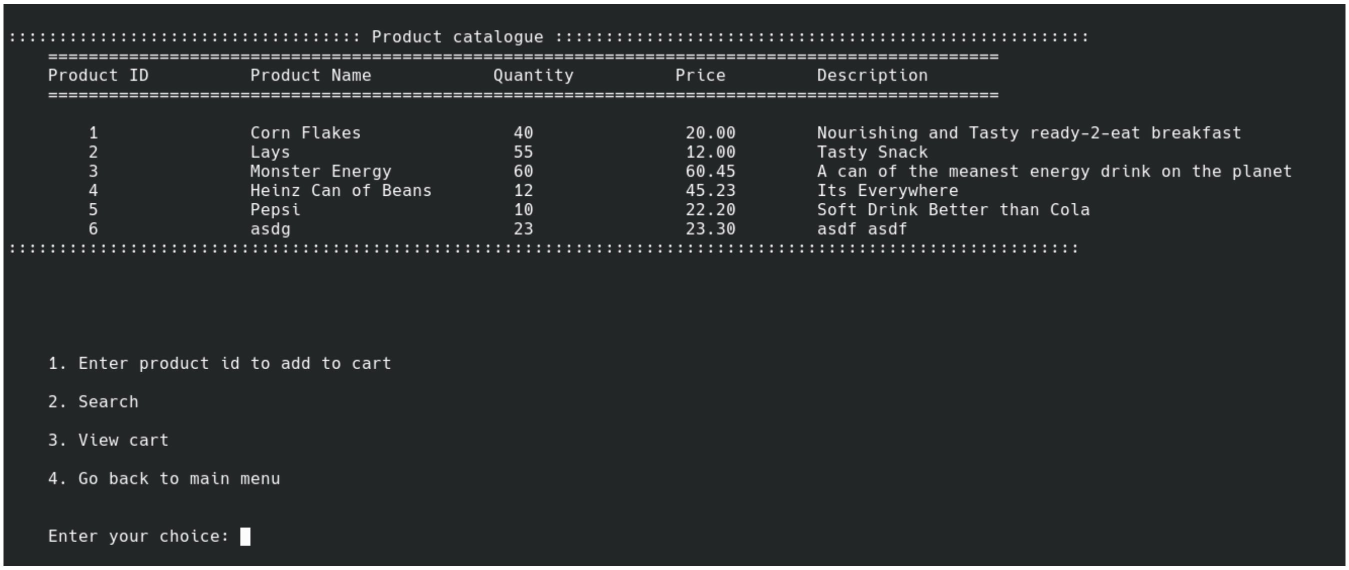
```
int displayProduct() {
   FILE *fp;
   struct product a;
   int choice;

   char true;
   system("clear");
   fp = fopen("products.dat", "rb");
   gotoxy(0, 5);
   printf("::::::::::::::::::::::::::::::::::: Product catalogue
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::");
   gotoxy(5, 6);
printf("=========================================================================
=======================");

   gotoxy(5, 7);
   printf("Product ID\t\tProduct Name\t\tQuantity\t  Price\t
Description"); //TABLE TITLES !
   gotoxy(5, 8);
```

```
printf("==========================================================================
=======================");
   gotoxy(5,9);
               while (fread(&a,sizeof(Product), 1,fp))
               {
               printf("\n\t%-2d\t\t%-20s\t%4d\t\t%8.2f\t%-50s", a.id,
a.product_name,a.quantity, a.price, a.description);
               }
               fclose(fp);
printf("\n:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
::::::::::::::::::::::::::::::::::::::");
   gotoxy(5,22);
   printf("1. Enter product id to add to cart");
   gotoxy(5,24);
   printf("2. Search");
   gotoxy(5,26);
   printf("3. View cart");
   gotoxy(5,28);
   printf("4. Go back to main menu");
   gotoxy(5,31);
   printf("Enter your choice: ");
   scanf("%d",&choice);
   switch(choice)
   {
   case 1:{
   goto Cleanup;
   Cleanup:;
   int id, quantity;
   gotoxy(10,33);
   printf("Product id: ");
   scanf("%d",&id);
   gotoxy(10,34);
   printf("Quantity: ");
   scanf("%d",&quantity);
   addtocart(id,quantity);
   break;
   }
   case 2:{
   search();
   break;
   }
   case 3:{
   printcart();
   break;
   }
```

```
    case 4:
    {
    system("clear");
    usermenu();
    }
    default:
    {
    gotoxy(10,32);
    printf("\aInvalid entry!!Please re-enter a valid option. Redirecting to
product catalogue.");
    sleep(3);
    displayProduct();
    }
    }return (0);}
```



```
::::::::::::::::::::::::::::::::::: Product catalogue :::::::::::::::::::::::::::::::::::::::::::::::::::::
    ===============================================================================================
    Product ID          Product Name            Quantity          Price         Description
    ===============================================================================================

        1               Corn Flakes               40               20.00        Nourishing and Tasty ready-2-eat breakfast
        2               Lays                      55               12.00        Tasty Snack
        3               Monster Energy            60               60.45        A can of the meanest energy drink on the planet
        4               Heinz Can of Beans        12               45.23        Its Everywhere
        5               Pepsi                     10               22.20        Soft Drink Better than Cola
        6               asdg                      23               23.30        asdf asdf
:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::




    1. Enter product id to add to cart

    2. Search

    3. View cart

    4. Go back to main menu

    Enter your choice: █
```

5. 3 int search()

Return type: int

search() function takes the product name as the input and matches the corresponding product from the "products.dat" binary file. If a match is found it gives the user the choice of adding the item to the cart or to view the product catalogue or to go back to the main menu.

```
                Enter the Product name to Search : Lays

::::::::::::::::::::::::::::::::::: Search result :::::::::::::::::::::::::::::::::::::::::::::::::::
==================================================================================================
Product ID          Product Name            Quantity          Price         Description
==================================================================================================

    2               Lays                     55               12.00        Tasty Snack


1. Add product to the cart

2. View product catalogue

3. Go back to main menu

Enter your choice:
```

```
int search(){
   system(CLEAR);
   Product p_1;
   FILE *fp;
   char n[30],temp[2];
   int id,value, found = 0;
   fp = fopen("products.dat","r");
   gotoxy(20,3);
   printf("Enter the Product name to Search : ");
   scanf("%c",temp);
   scanf("%[^\n]s",n);
   int choice;
   //scanf("%d", &id);

   while (fread(&p_1,sizeof(Product), 1,fp))
   {

 value=strcmp(n,p_1.product_name);
       if (value==0) {
           found = 1;
           //printf("%-5d|%-20s|%-50s|%8.2f|%4d\n", p_1.id, p_1.product_name,
p_1.description, p_1.price, p_1.quantity);
           gotoxy(0, 5);
           printf("    ::::::::::::::::::::::::::::::::::: Search result
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::");

           gotoxy(5, 6);

printf("==================================================================
==================================");

           gotoxy(5, 7);
```

```
            printf("Product ID\t\tProduct Name\t\tQuantity\t  Price\t
Description"); //TABLE TITLES !

            gotoxy(5, 8);

printf("=================================================================
=====================================");
            gotoxy(5,9);
            printf("\n\t%-2d\t\t%-20s\t%4d\t\t%8.2f\t%-50s", p_1.id,
p_1.product_name,p_1.quantity, p_1.price, p_1.description);

            gotoxy(5,13);
            printf("1. Add product to the cart");
            gotoxy(5,15);
            printf("2. View product catalogue");
            gotoxy(5,17);
            printf("3. Go back to main menu");
            gotoxy(5,19);
            printf("Enter your choice: ");
            scanf("%d",&choice);
            switch(choice)
            {
                case 1:{
                goto Cleanup;

                Cleanup:;
                int quantity;
                //printf("Product id: ");
                //scanf("%d",&id);
                gotoxy(10,21);
                printf("Quantity: ");
                scanf("%d",&quantity);
                addtocart(p_1.id,quantity);
                break;
                }
                case 2:{
                goto Cleanup1;
                Cleanup1:;
                displayProduct();
                break;
                }
                case 3:{
                goto Cleanup2;
                Cleanup2:;
                usermenu();
```

```
            break;
            }
```

5. 4 int addtocart(int id, int quantity)

Return type: int

addtocart(i,q) takes product id and the quantity as the input from either search() or displayProduct() and if sufficient quantity of the product is available then it adds the product to “cart.csv” file. Once the product is added to the cart subtract(int id, int quantity) subtracts the appropriate quantity from the “products.dat”.

```
int addtocart(int i, int q){
   FILE *fp, *fptr;
   int n,j;
   struct product a;
   Product *p;
   p = (Product*)calloc(n, sizeof(Product));
   fp = fopen("products.dat","r");
   fptr = fopen("cart.csv","a");

           while (fread(&a,sizeof(Product), 1,fp))
               {

                   if (a.id==i){
                       if (a.quantity<q){
                           printf("Only %d items available\nRedirecting to
catalogue...",a.quantity);

                           //sleep(1);
                           displayProduct();
                       }
                       else{

 fprintf(fptr,"%-1d,%-1s,%1d,%1f,%-1s\n", a.id, a.product_name, q, a.price,
a.description);
                           subtract(a.id,q);
                           //product_quantity(q);
                           //printf("\n%d",a.quantity);
                           //a.quantity=a.quantity-q;
                           //fwrite(&a,sizeof(Product), 1,fp);
                       }
                   }
               }
               printf("\nItem added to cart.");

   fclose(fp);
   fclose(fptr);
   displayProduct();

   return 0;
}
```

5. 5 void subtract(int i, int q)

Once the product is added to the cart subtract(i,q) subtracts the appropriate quantity from the "products.dat"

```
void subtract(int n,int q){
    FILE *fp, *ft;
    int found=0;
    fp = fopen("products.dat","r");
    ft = fopen("temp.dat","w");
    Product p_2;
    while (fread(&p_2,sizeof(Product), 1,fp))
    {
        if (p_2.id == n) {
            found = 1;
            strcpy(p_1.product_name,p_2.product_name);
            strcpy(p_1.description,p_2.description);
            p_1.id=p_2.id;
            p_1.price=p_2.price;

            p_1.quantity=p_2.quantity - q;
        }
        else{
            p_1=p_2;
        }
        fwrite(&p_1, sizeof(Product), 1, ft);
    }
    fclose(ft);
    fclose(fp);
    if(found==1){
        ft = fopen("temp.dat","r");
        fp = fopen("products.dat", "w");
        while (fread(&p_1, sizeof(Product), 1,ft)) {
            fwrite(&p_1, sizeof(Product), 1, fp);
        }

        fclose(fp);
        fclose(ft);
    }
}
```

5. 6 int placeorder()

Return type : void

This function systematically gives an option for take-away/collect-from-store(option 2) or home-delivery (option 1). For option 1, ‘avail home-delivery’ the system asks for the pincode of the region, and checks it against a list of pincodes. In case if the pin exists in the pin.txt (it is opened in the read mode “r”), we proceed to ask for the preferred timing of delivery (from a list) .We deliver at these pins, as per pin.txt : 123456, 234156, 567789, 425678, 567134, 890162. If the entered pin is not one of these, we aptly notify that we don’t deliver to that location. The preferred timing of delivery is asked for and checked against if a delivery van is available using a simple if … else statement, for that chosen slot from time.txt , else the customer is informed that delivery will be delayed. In option 2, ‘collect your order directly from the outlet’ we simply print the store timings for the customer’s reference. And option 3 redirects us back to the main menu.

```
int placeorder(){

      system("clear");
      int pin, time_slot, pin_found = 0, time_found = 0;
      char address[100];
      FILE *fp_pin = fopen("pin.txt", "r");
      int len;
      fscanf(fp_pin, "%i", &len);
      // pins[] contain the available pins
      int pins[len], index;
      for(index = 0; index < len; index += 1){
      fscanf(fp_pin, "%i", &pins[index]);
      }
      fclose(fp_pin);
      //accessing time.txt; checking availability of a delivery van
      FILE *fp_time = fopen("time.txt", "r");
      int availability_of_vans[5];

      for(index = 0; index < 5; index += 1){
      fscanf(fp_time, "%i", &availability_of_vans[index]);
      }
      fclose(fp_time);
      int opt1;
      gotoxy(0,4);
      printf("1. Avail Home delivery");
      gotoxy(0,6);
      printf("2. Collect your order directly from our outlet");
      gotoxy(0,8);
```

```
        printf("3. Back to main menu");
        gotoxy(5,11);
        printf("Please select your preferred option: ");
        scanf("%d", &opt1);

        switch(opt1)
        {
        case 1:
        gotoxy(5,13);
        printf("Enter the PINCODE of the delivery locality (pattern: ******): 
");
        scanf("%i", &pin);
        for(index = 0; index < len; index += 1){
                if(pins[index] == pin){
                        pin_found = 1;
                        break;}
        }
        if(pin_found)
        {

                fflush(stdin);
                scanf("%100[^\n]s", address);
                gotoxy(0,15);
                printf("Which of the time-slots is ideal to deliver to 
you?\n");
                printf("1. 5-7\n2. 7-9\n3. 9-13\n4. 13-17\n5. 17-22\n\n");
                printf("Enter your choice: ");
                scanf("%i", &time_slot);

                if(availability_of_vans[time_slot-1] == 1){
                        printf("Van available.\n");
                }
                else{
                        printf("No van available. Delivery might be delayed.\n");
                }
                printf("\nProceeding to the billing section.\n\n");
                sleep(3);
                generate_bill(address);
                break;
        }
        else
        {
        printf("Oops!\n");
```

```
        printf("Looks like we don't deliver to your locality.\n");
        sleep(3);
        placeorder();
        return 1;
        }

        case 2:
                printf("\nYour order is curated");
                printf("\nFeel free to drop by and receive your order!");
                printf("\nThe Outlet is open from 8am to 8pm\n");
                generate_bill(address);

        case 3:
                break;
        }
        return 0;
}
```

```
1. Avail Home delivery

2. Collect your order directly from our outlet

3. Back to main menu


    Please select your preferred option: 1

    Enter the PINCODE of the delivery locality (pattern: ******): 123456

Which of the time-slots is ideal to deliver to you?
1. 5-7
2. 7-9
3. 9-13
4. 13-17
5. 17-22

Enter your choice: 2
Van available.

Proceeding to the billing section.

====================== Aapki Dukaan ========================

1. Generate Invoice
2. Show all Invoices
3. Back to main menu

 Please select your preferred option:
```

5. 7 genBillheader

This function creates the bill header.

```
void genBillheader(FILE *STREAM, char name[30], char date[30], char
address[100])
{
      printf("\n\n\t\t   %s\n\t ---------------------------", "Aapki
Dukaan");
      printf("\n\tOrder placed on: %s\n\tInvoice To: %s\n \n", date, name);

printf("\n=============================================\n%-20s%-10s%-10s%-
10s",
      "Item", "ID", "Quantity", "Total");
      printf("\n---------------------------------------------\n\n");
```

```
	fprintf(STREAM, "\t\t   %s\n\t --------------------------", "Aapki
Dukaan");
	fprintf(STREAM, "\n\tOrder placed on: %s\n\tInvoice To: %s\n ", date,
name);
	fprintf(STREAM,
"\n==============================================\n%-20s%-10s%-10s%-10s",
	"Item", "ID", "Quantity", "Total");
	fprintf(STREAM,
"\n----------------------------------------------\n\n");
}
float genBillbody(FILE *STREAM, char item[20], int id, int qty, float
price)
{
	float TOTAL; // gets returned; total for one item
	TOTAL = qty * price;

	printf("%-20s%-10d%-10d%-8.2f\n", item, id, qty, TOTAL);
	fprintf(STREAM, "%-20s%-10d%-10d%-8.2f\n", item, id, qty, TOTAL);

	return(TOTAL);
}
```

5. 8 float genBillbody

This function is used to create the main bill body.

```
float genBillbody(FILE *STREAM, char item[20], int id, int qty, float
price)
{
	float TOTAL; // gets returned; total for one item
	TOTAL = qty * price;

	printf("%-20s%-10d%-10d%-8.2f\n", item, id, qty, TOTAL);
	fprintf(STREAM, "%-20s%-10d%-10d%-8.2f\n", item, id, qty, TOTAL);

	return(TOTAL);
}
```

5. 9 void genBillfooter

This function creates the bill footer. It accounts for the calculation of the grand total after accounting for the discounts and taxes.

```
void genBillfooter(FILE *STREAM, float total)
{
      float disc = 0.05*total; // 5% discount
      float netTotal = total - disc;
      float GST=0.15*netTotal, grandTotal = netTotal + GST;
      printf("\n----------------------------------------------\n");
      printf("%-40s%-18.2f\n", "Sub total", total);
      printf("%-40s%-18.2f\n", "Discount @5\%", disc);
      printf("-------------------------\n");
      printf("%-40s%-18.2f\n", "Net total", netTotal);
      printf("%-40s%-18.2f\n", "GST @15\%", GST);
      printf("-------------------------\n");
      printf("%-40s%-18.2f\n", "Grand total", grandTotal);
      printf("----------------------------------------------\n");
      printf("Thank you!! ----------------------------------\n");
      fprintf(STREAM,
"\n----------------------------------------------\n");
      fprintf(STREAM, "%-40s%-18.2f\n", "Sub total", total);
      fprintf(STREAM, "%-40s%-18.2f\n", "Discount @5\%", disc);
      fprintf(STREAM, "-------------------------\n");
      fprintf(STREAM, "%-40s%-18.2f\n", "Net total", netTotal);
      fprintf(STREAM, "%-40s%-18.2f\n", "GST @15\%", GST);
      fprintf(STREAM, "-------------------------\n");
      fprintf(STREAM, "%-40s%-18.2f\n", "Grand total", grandTotal);
      fprintf(STREAM, "----------------------------------------------\n");
      fprintf(STREAM, "Thank you!! ----------------------------------\n");
}
```

5. 10 int generate_bill

Here we generate or show all invoices, which furthermore directs the user using the switch() function. In option 1, i.e Generate invoice, we read the cart.csv, ask for the user's name and access invoices.txt file in the append mode as a final invoice as well as have the bill visible on the console itself. Option 2, shows all invoices generated where iit accesses the invoices.txt in the read mode.

```
            //header of the receipt
            genBillheader(fp_receipt, new_invoice.customer_name,
new_invoice.date, address);

            while(fscanf(fp_cart, "%d,%20[^,],%d,%f,%[^\n]s", &cart.id,
cart.item_name, &cart.qty,
                  &cart.price, cart.description)!=EOF){
                  if(index == 10)
                        break;
                  new_invoice.items[index] = cart;
                  //body of the receipt
                  net_total += genBillbody(fp_receipt, cart.item_name,
cart.id, cart.qty, cart.price);
                  index += 1;
            }
            fclose(fp_cart);
            //updating invoice
            new_invoice.numOfItems = index;
            new_invoice.total = net_total;
            //tail of the receipt
            genBillfooter(fp_receipt, net_total);
            fclose(fp_receipt);

            fprintf(fp_inv, "%d %s %10.2f\n", new_invoice.numOfItems,
temp, new_invoice.total);
            for(index = 0; index < new_invoice.numOfItems; index += 1){
                  fprintf(fp_inv, "%s,%d,%d\n",
new_invoice.items[index].item_name, new_invoice.items[index].id,
                  new_invoice.items[index].qty);
            }
            fclose(fp_inv);
            printf("\nProcess completed successfully.\n");
            exit(0);
            return 0;
      case 2:
            printf("Showing all purchases made...\n\n\n");
            char phrase[20];
            int items_per_purchase;
            index = 0;
            FILE *fp_iv = fopen("invoices.txt", "r");
            while(fscanf(fp_iv, "%s", phrase)!=EOF){
                  if(index == 0){
                  items_per_purchase = atoi(phrase);
```

```
                    index += 1;
                    }
                    else if(index == 1){
                    printf("Date of purchase: ");
                    char *p; p = phrase;
                    while(*p != '\0'){
                          if(*p == '_')
                                printf(" ");
                          else
                                printf("%c", *p);
                          p+=1;
                    }
                    printf("\t\t");
                    index += 1;
                    }
                    else{
                    printf("Total: %s\n", phrase);
                    index += 1;
                    printf("%-20s%-20s%-20s", "Item", "Product ID",
"Quantity");
                    for(index = 3; index<items_per_purchase+3; index +=1){
                          char t1[20], t2[10], t3[10];
                          fscanf(fp_iv, "%[^,],%[^,],%[^\n]", t1, t2, t3);
                          printf("%-20s%-20s%-20s", t1, t2, t3);
                    }
                    printf("\n_____________________________________");
                    printf("___________________________\n");
                    index = 0;
                    }
              }
              continue;
      case 3:
              printf("Press any key to go back\n");
              if(getch)
              usermenu();
              break;
      }
      }
      return 0;
}
```

```
====================== Aapki Dukaan ========================

1. Generate Invoice
2. Show all Invoices
3. Back to main menu

 Please select your preferred option:   1
Here is your receipt.
Also, please refer to Nov_15_2021.txt.


                    Aapki Dukaan
         ---------------------------
        Order placed on: Nov 15 2021
        Invoice To: ( }U
  Address:

================================================
Item                ID        Quantity  Total
------------------------------------------------

Corn Flakes         1         2         40.00

------------------------------------------------
Sub total                                 40.00
Discount @5%                              2.00
-------------------------
Net total                                 38.00
GST @15%                                  5.70
-------------------------
Grand total                               43.70
------------------------------------------------
Thank you!! ----------------------------------

Process completed successfully.
 Press [1] to continue shopping [0] to exit:
```

Thank You