**Style For You !!!**

A logo of a university of engineering and technology

Description automatically generated

# Session 2023 – 2027

# Submitted by:

Saad Qaiser 2023-CS-37

# Supervised by:

Dr. Muhammad Awais Hassan

# Course:

CSC-102 Programming Fundamentals

Department of Computer Science

# University of Engineering and Technology

# Lahore Pakistan

# Style For You !!!

This application serves as an online shoe store, offering customers a diverse range of footwear options. The primary aim is to develop a user-friendly application that seamlessly guides users through the selection and purchase of shoes. Online retail is an embodiment of computer science, contributing to the field by creating a platform that facilitates the acquisition of shoes through a digital interface. Upon completion, the application will feature an extensive assortment of shoe products for customers to choose from.

# Users Of Application

This application will have two users:

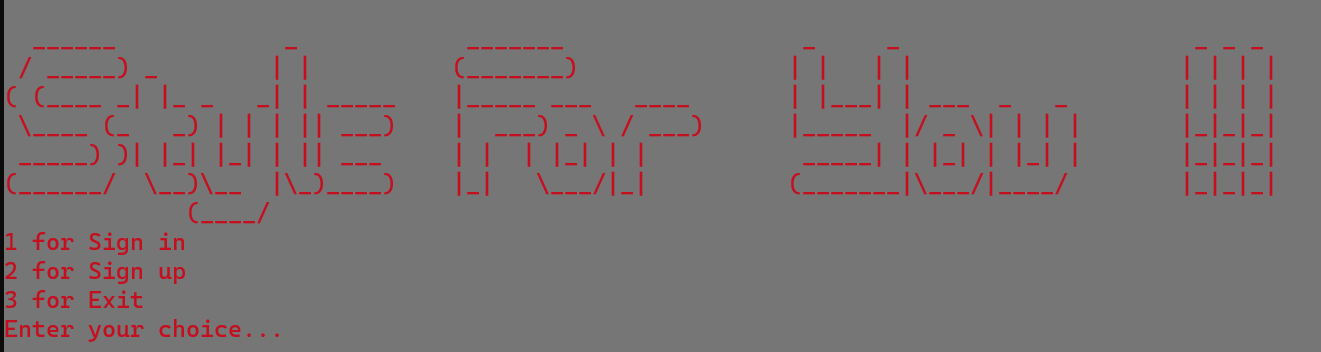
* Customer: This user will have access to all the items that are available.
* Admin: The admin can make changes in the function displayed to the customers.

# Functional Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| User Story ID | User Type | Required Function | Result of Action Performed |
| 1 | Admin | 1. View List Shoes | List of Shoes appears |
| 1. Change prices | Change prices of items |
| 1. Change Available Stock | Increment or decrement in the quantity available of items |
| 1. Check Reviews | Review from User |
| 1. Add item | Addition in items |
| 1. Remove item | Remove any items |
| 1. Update Tax | It will update tax |
| 1. Special Days | Discounts on Special Days |
| 1. Status of Delivery | Check status of delivery |
| 1. Exit | It will exit |
|  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| User Story ID | User Type | Required Function | Result of Action Performed |
| 2 | Customer | Categories | Shows available stock |
| Add to Cart | Enlist item |
| View cart | List of purchased items |
| Total Amount | Total amount appears |
| Remove from Cart | Remove item from cart |
| View Tax | Tax menu appears |
| View payment options | Payment Option Menu appears |
| Details | Delivery details |
| Feedback | Opinion |
| Exit | Exit from Shop |
|  |  |

# Wireframes



**Figure 1: Sub Menu Before Main Menu**



**Figure 2: Customer Menu**



**Figure 3: Employee Menu**

# Data Structures (Parallel Arrays)

* + string username[] ( index ).
  + string password[] ( index ).
  + string role[] ( index ).
  + String name[](productCount)
  + Float price[](productCount)
  + Int quantity[](productCount)
  + Int stock[](productCount)
  + Float tax[](productCount)
  + Float total[](productCount)

# Function Prototypes

void sign\_up(string username[100],string password[100],string role[100],int &index);

void sign\_in(string name[100],float price[100],int quantity[100],int stock[100],float tax[100],float total[100],string &option,int &productCount,int &del,string username[100],string password[100],string role[100],int &index,string reviews[100]);

bool flag(string name,string username[100],string password[100],string role[100],int &index);

int check(string name,string username[100],string password[100],string role[100],int &index);

void header();

void about();

string menu();

string menuAdmin();

void product(string &name,float &price,int &quantity,float &tax,float &total);

float totalCost(float &price,int &quantity,float &tax);

void message();

void categories(string name[],int stock[],int &productCount);

void taxProduct(string name[],float tax[],int &productCount);

string payment();

int details();

void cart(string[],int[],int &productCount,int stock[]);

void feedback(string reviews[],string username[],int &index);

void checkFeedBacks(string username[],int &index,string reviews[]);

int cartOption(int &option,string name[],int &productCount);

void cust(string name[100],float price[100],int quantity[100],int stock[100],float tax[100],float total[100],string &option,int &productCount,int &del,string username[100],string password[100],string role[100],int &index,string reviews[100]);

void admin(string name[100],float price[100],int quantity[100],int stock[100],float tax[100],float total[100],string &option,int &productCount,int &del,string username[100],string password[100],string role[100],int &index,string reviews[100]);

int changePrices(string name[],int &productCount);

int changeStock(string name[],int &productCount);

int changeTax(string name[],int &productCount);

void viewProducts(string name[],float price[],int stock[],int &productCount,int &del);

void addNewItem(string name[],float price[],int stock[],float tax[],int &productCount);

void delItem(string name[],int &del,float price[],int quantity[],int &productCount,int stock[],float tax[]);

void removeUser(string username[],string password[],string role[],int &index);

void showUser(string username[],string role[], int &index);

void bill(int &productCount,int quantity[]);

void storeDataLocally(string name[],float price[],int stock[],float tax[],int &productCount);

void loadData(string name[],float price[],int stock[],float tax[],int &productCount);

void storeDataLocally2(string username[],string password[],string role[],int &index);

void loadData2(string username[],string password[],string role[],int &index);

string getFieldData(string data, int count);

void clearScreen();

bool isValid\_int(string num);

int convertStoInt(string num);

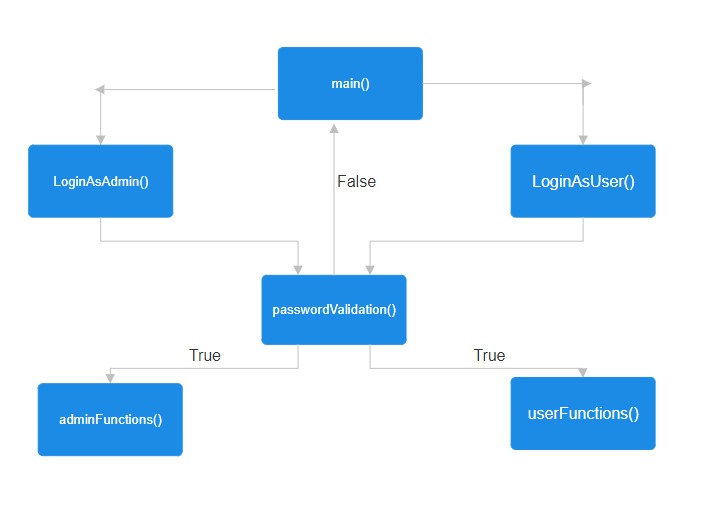
bool password\_vald(string password);

# Function Working Flow

This is a diagram of the working flow of the Business Application Fashion Is U.

(Continued on Next Page)

**Fashion Is U Flow diagram:**



# Complete Code of Style For You !!!

#include <iostream>

#include <conio.h>

#include<windows.h>

#include<limits>

#include <fstream>

using namespace std;

void sign\_up(string username[100],string password[100],string role[100],int &index);

void sign\_in(string name[100],float price[100],int quantity[100],int stock[100],float tax[100],float total[100],string &option,int &productCount,int &del,string username[100],string password[100],string role[100],int &index,string reviews[100]);

bool flag(string name,string username[100],string password[100],string role[100],int &index);

int check(string name,string username[100],string password[100],string role[100],int &index);

void header();

void about();

string menu();

string menuAdmin();

void product(string &name,float &price,int &quantity,float &tax,float &total);

float totalCost(float &price,int &quantity,float &tax);

void message();

void categories(string name[],int stock[],int &productCount);

void taxProduct(string name[],float tax[],int &productCount);

string payment();

int details();

void cart(string[],int[],int &productCount,int stock[]);

void feedback(string reviews[],string username[],int &index);

void checkFeedBacks(string username[],int &index,string reviews[]);

int cartOption(int &option,string name[],int &productCount);

void cust(string name[100],float price[100],int quantity[100],int stock[100],float tax[100],float total[100],string &option,int &productCount,int &del,string username[100],string password[100],string role[100],int &index,string reviews[100]);

void admin(string name[100],float price[100],int quantity[100],int stock[100],float tax[100],float total[100],string &option,int &productCount,int &del,string username[100],string password[100],string role[100],int &index,string reviews[100]);

int changePrices(string name[],int &productCount);

int changeStock(string name[],int &productCount);

int changeTax(string name[],int &productCount);

void viewProducts(string name[],float price[],int stock[],int &productCount,int &del);

void addNewItem(string name[],float price[],int stock[],float tax[],int &productCount);

void delItem(string name[],int &del,float price[],int quantity[],int &productCount,int stock[],float tax[]);

void removeUser(string username[],string password[],string role[],int &index);

void showUser(string username[],string role[], int &index);

void bill(int &productCount,int quantity[]);

void storeDataLocally(string name[],float price[],int stock[],float tax[],int &productCount);

void loadData(string name[],float price[],int stock[],float tax[],int &productCount);

void storeDataLocally2(string username[],string password[],string role[],int &index);

void loadData2(string username[],string password[],string role[],int &index);

string getFieldData(string data, int count);

void clearScreen();

bool isValid\_int(string num);

int convertStoInt(string num);

bool password\_vald(string password);

main()

{

string name[100];

float price[100];

int quantity[100];

int stock[100];

float tax[100];

float total[100];

string option;

int productCount=0;

int del=0;

string username[100];

string password[100];

string role[100];

int index = 0;

string reviews[100];

string op;

loadData(name,price,stock,tax,productCount);

loadData2(username,password,role,index);

while(true)

{

system("cls");

header();

cout << "1 for Sign in" << endl;

cout << "2 for Sign up" << endl;

cout << "3 for Exit" << endl;

cout << "Enter your choice...";

getline(cin, op);

if(isValid\_int(op))

{

if(op=="1")

{

system("cls");

sign\_in(name,price,quantity,stock,tax,total,option,productCount,del,username,password,role,index,reviews);

}

else if(op=="2")

{

system("cls");

sign\_up(username,password,role,index);

}

else if(op=="3")

{

storeDataLocally(name,price,stock,tax,productCount);

storeDataLocally2(username,password,role,index);

return 0;

}

}

else

{

cout << "Invalid Input.....\n";

clearScreen();

}

}

}

// This function is like another main but for customer

void cust(string name[100],float price[100],int quantity[100],int stock[100],float tax[100],float total[100],string &option,int &productCount,int &del,string username[100],string password[100],string role[100],int &index,string reviews[100])

{

while (true)

{

system("cls");

header();

option = menu();

if (option == "1")

{

system("cls");

header();

categories(name,stock,productCount);

}

if (option == "2")

{

system("cls");

header();

cart(name,quantity,productCount,stock);

}

if (option == "3")

{

system("cls");

header();

for(int x=0;x<productCount;x++)

{

total[x]=totalCost(price[x],quantity[x],tax[x]);

}

float totalAmount=0;

for(int x=0;x<productCount;x++)

{

totalAmount+=total[x];

}

cout << "All Products Data" << endl;

cout << "Name" << " \t \t \t" << "Price " << "\t" << "No. of Pairs" << "\t" << "Tax(%)" << "\t" <<

"TotalPerProduct" << endl;

for(int x=0;x<productCount;x++)

{

product(name[x], price[x], quantity[x], tax[x], total[x]);

}

cout << "Total Amount is: "<<totalAmount<<endl;

}

if (option == "4")

{

system("cls");

header();

for(int x=0;x<productCount;x++)

{

total[x]=totalCost(price[x],quantity[x],tax[x]);

}

float totalAmount=0;

for(int x=0;x<productCount;x++)

{

totalAmount+=total[x];

}

cout << "Total Amount So Far (including tax): $" << totalAmount << endl;

}

if (option == "5")

{

system("cls");

header();

about();

}

if (option == "6")

{

system("cls");

header();

taxProduct(name,tax,productCount);

}

if (option == "7")

{

system("cls");

header();

bill(productCount,quantity);

message();

}

if (option == "8")

{

system("cls");

header();

int day = details();

if(day<=3)

{

cout<<"We are prepering the Package...";

}

else if (day>3 && day<7)

{

cout<<"Your Package is on his way...";

}

else if (day>=7)

{

cout<<"Package should be recieved...";

}

}

if (option == "9")

{

system("cls");

header();

feedback(reviews,username,index);

}

if (option == "10")

{

break;

}

cout << "Press any Key to Continue: ";

getch();

system("cls");

}

}

// This function is like another main but for admin

void admin(string name[100],float price[100],int quantity[100],int stock[100],float tax[100],float total[100],string &option,int &productCount,int &del,string username[100],string password[100],string role[100],int &index,string reviews[100])

{

while (true)

{

system("cls");

header();

option = menuAdmin();

if (option == "1")

{

system("cls");

header();

viewProducts(name,price,stock,productCount,del);

}

if (option == "2")

{

system("cls");

header();

int prices=changePrices(name,productCount);

int x=0;

while(x<productCount+1)

{

if(prices==x)

{

cout<<"Enter new Price of "<<name[x-1]<<": ";

cin>>price[x-1];

break;

}

x++;

}

}

if (option == "3")

{

system("cls");

header();

int stocks=changeStock(name,productCount);

int x=0;

while(x<productCount+1)

{

if(stocks==x)

{

cout<<"Enter new Stock of "<<name[x-1]<<": ";

cin>>stock[x-1];

break;

}

x++;

}

}

if (option == "4")

{

system("cls");

header();

checkFeedBacks(username,index,reviews);

}

if (option == "5")

{

system("cls");

header();

addNewItem(name,price,stock,tax,productCount);

}

if (option == "6")

{

system("cls");

header();

delItem(name,del,price,quantity,productCount,stock,tax);

}

if (option == "7")

{

system("cls");

header();

int stocks=changeTax(name,productCount);

int x=0;

while(x<productCount+1)

{

if(stocks==x)

{

cout<<"Enter new tax of "<<name[x-1]<<": ";

cin>>tax[x-1];

break;

}

x++;

}

}

if (option == "8")

{

system("cls");

header();

showUser(username,role,index);

}

if (option == "9")

{

system("cls");

header();

removeUser(username,password,role,index);

}

if (option == "10")

{

break;

}

cout << "Press any Key to Continue: ";

getch();

system("cls");

}

}

//This function is the Header

void header()

{

system ("Color 84");

cout << R"(

\_\_\_\_\_\_ \_ \_\_\_\_\_\_\_ \_ \_ \_ \_ \_

/ \_\_\_\_\_) \_ | | (\_\_\_\_\_\_\_) | | | | | | | |

( (\_\_\_\_ \_| |\_ \_ \_| | \_\_\_\_\_ |\_\_\_\_\_ \_\_\_ \_\_\_\_ | |\_\_\_| | \_\_\_ \_ \_ | | | |

\\_\_\_\_ (\_ \_) | | | || \_\_\_) | \_\_\_) \_ \ / \_\_\_) |\_\_\_\_\_ |/ \_ \| | | | |\_|\_|\_|

\_\_\_\_\_) )| |\_| |\_| | || \_\_\_ | | | |\_| | | \_\_\_\_\_| | |\_| | |\_| | |\_|\_|\_|

(\_\_\_\_\_\_/ \\_\_)\\_\_ |\\_)\_\_\_\_) |\_| \\_\_\_/|\_| (\_\_\_\_\_\_\_|\\_\_\_/|\_\_\_\_/ |\_|\_|\_|

(\_\_\_\_/ )"<<endl;

}

//This funtion returns a option for Customer Choice

string menu()

{

string option;

cout << "Select one of the following options number..." << endl;

cout << "1. Categories" << endl;

cout << "2. Add to Cart" << endl;

cout << "3. View Cart" << endl;

cout << "4. TotalAmount" << endl;

cout << "5. About" << endl;

cout << "6. View tax" << endl;

cout << "7. View Payment Options" << endl;

cout << "8. Delivery Details" << endl;

cout << "9. Feedback" << endl;

cout << "10. LogOut" << endl;

cout << "Your Option..";

cin.clear();

cin.sync();

getline(cin,option);

if(isValid\_int(option))

{

return option;

}

else if(option.length() == 2 && option[1] > '0' && option[1] <= '9')

{

cout << "Invalid please try again";

Sleep(1000);

system("cls");

header();

menu();

}

else

{

cout << "Invalid Please try again...\n";

Sleep(1000);

system("cls");

header();

menu();

}

}

//This funtion returns a option for Admin Choice

string menuAdmin()

{

string option;

cout << "Select one of the following options number..." << endl;

cout << "1. View List Shoes" << endl;

cout << "2. Change Prices" << endl;

cout << "3. Change Available Stock" << endl;

cout << "4. Check Reviews" << endl;

cout << "5. Add item" << endl;

cout << "6. Remove item" << endl;

cout << "7. Update Tax" << endl;

cout << "8. Show Users" << endl;

cout << "9. Remove Users" << endl;

cout << "10. LogOut" << endl;

cout << "Your Option..";

cin.clear();

cin.sync();

getline(cin,option);

if(isValid\_int(option))

{

return option;

}

else if(option.length() == 2 && option[1] > '0' && option[1] <= '9')

{

cout << "Invalid please try again";

Sleep(1000);

system("cls");

header();

menu();

}

else

{

cout << "Invalid Please try again...\n";

Sleep(1000);

system("cls");

header();

menu();

}

}

// This function Calculates Price

float totalCost(float &price, int &quantity, float &tax)

{

float total,totalPrice;

total = price \* quantity;

totalPrice = total + (total \* (tax / 100));

return totalPrice;

}

//This function is used to show product data

void product(string &name, float &price, int &quantity, float &tax, float &total)

{

cout << name << " \t \t" << price << "\t" << quantity << "\t\t" << tax << "\t" << total << endl;

}

\

//This function shows a simple msg

void message()

{

cout << R"(

\_\_\_\_\_ \_ \_ \_\_\_\_\_

|\_ \_|| |\_\_ \_\_ \_ \_ \_\_ | | \_\_ \_\_\_ | \_\_\_|\_\_\_ \_ \_\_

| | | '\_ \ / \_` || '\_ \ | |/ // \_\_| | |\_ / \_ \ | '\_\_|

| | | | | || (\_| || | | || < \\_\_ \ | \_|| (\_) || |

|\_| |\_| |\_| \\_\_,\_||\_| |\_||\_|\\_\|\_\_\_/ |\_| \\_\_\_/ |\_|

\_\_\_\_ \_ \_ \_ \_ \_

/ \_\_\_| | |\_\_ \_\_\_ \_ \_\_ \_ \_\_ (\_) \_ \_\_ \_\_ \_ | || || |

\\_\_\_ \ | '\_ \ / \_ \ | '\_ \ | '\_ \ | || '\_ \ / \_` | | || || |

\_\_\_) || | | || (\_) || |\_) || |\_) || || | | || (\_| | |\_||\_||\_|

|\_\_\_\_/ |\_| |\_| \\_\_\_/ | .\_\_/ | .\_\_/ |\_||\_| |\_| \\_\_, | (\_)(\_)(\_)

|\_| |\_| |\_\_\_/

)"<<endl;

}

//This function shows product names and their stock availbility

void categories(string name[],int stock[],int &productCount)

{

cout << "Name \t \t Availability "<<endl;

for (int x=0;x<productCount;x++)

{

cout<<name[x]<<"\t \t \t"<<stock[x]<<endl;

}

}

//This function is used to show taxes

void taxProduct(string name[],float tax[],int &productCount)

{

cout << "Name \t \t Tax "<<endl;

for (int x=0;x<productCount;x++)

{

cout<<name[x]<<"\t \t \t"<<tax[x]<<endl;

}

}

//this function is for payment options

string payment()

{

string option;

cout << "Enter Payment Option (Cash/Card): ";

cin.clear();

cin.sync();

getline(cin,option);

return option;

}

//This function shows delivery details

int details()

{

int days;

cout << "Enter the No. of Days since you placed the order: "<<endl;

while(true)

{

if( cin >> days)

{

break;

}

else

{

cout << "Enter a valid index integer value..";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(),'\n');

}

}

return days;

}

//this function returns a option choose by customer

int cartOption(int &option,string name[],int &productCount)

{

for (int x=0;x<productCount;x++)

{

cout<<x+1<<" for "<<name[x]<<endl;

}

cout<<"Enter your option...";

while(true)

{

if( cin >> option)

{

break;

}

else

{

cout << "Please Enter a valid integer value..";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(),'\n');

}

}

return option;

}

//this function is used to show user how many pairs does he/she want

void cart(string name[],int quantity[],int &productCount,int stock[])

{

int option;

int op = cartOption(option,name,productCount);

for (int x=0;x<productCount+1;x++)

{

if(op==x)

{

cout<<"How many pairs of "<<name[x-1]<<" do you want: ";

while(true)

{

if( cin >> quantity[x-1])

{

break;

}

else

{

cout << "Please Enter a valid integer value..";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(),'\n');

}

}

if(quantity[x-1]>stock[x-1])

{

cout<<"Invalid"<<endl;

quantity[x-1]=0;

break;

}

break;

}

}

}

//this funtion is used to sign in whether as admin or customer

void sign\_in(string name[100],float price[100],int quantity[100],int stock[100],float tax[100],float total[100],string &option,int &productCount,int &del,string username[100],string password[100],string role[100],int &index,string reviews[100])

{

string names;

string pass;

int idx;

cout << "Enter Username: ";

cin.clear();

cin.sync();

getline(cin,names);

cout << "Enter Password: ";

cin.clear();

cin.sync();

getline(cin,pass);

if(password\_vald(pass))

{

if(flag(names,username,password,role,index))

{

idx = check(names,username,password,role,index);

if(pass==password[idx])

{

string role1= role[idx];

if(role1=="A" )

{

admin(name,price,quantity,stock,tax,total,option,productCount,del,username,password,role,index,reviews);

}

else if(role1=="C")

{

cust(name,price,quantity,stock,tax,total,option,productCount,del,username,password,role,index,reviews);

}

}

}

else{

cout<<"Incorrect id";

}

}

else

{

cout << "Invalid password.. Try Again..\n";

Sleep(1000);

sign\_in(name,price,quantity,stock,tax,total,option,productCount,del,username,password,role,index,reviews);

}

}

//this funtion is used to sign up as admin or customer with valid username and password

void sign\_up(string username[100],string password[100],string role[100],int &index)

{

string name;

string role1;

string pass;

cout << "Enter Username: ";

cin.clear();

cin.sync();

getline(cin,name);

if(flag(name,username,password,role,index))

{

cout << "Username Already Taken.."<<endl;

sign\_up(username,password,role,index);

}

else

{

username[index]=name;

cin.clear();

cin.sync();

cout << "Enter Password: ";

getline(cin, pass);

if(password\_vald(pass)){

password[index] = pass;

cout << "Enter Role (A for Admin or C for Customer): ";

cin >> role1;

if(role1=="A" || role1=="C")

{

role[index]=role1;

index++;

}

else

{

cout << "Incorrect role.."<<endl;

sign\_up(username,password,role,index);

}

}

else

{

cout << "Password should contain at one alphabet, no comma and space"<<endl;

Sleep(1000);

system("cls");

header();

sign\_up(username,password,role,index);

}

}

}

//this funtion is used to check whether username is valid or not

int check(string name,string username[100],string password[100],string role[100],int &index)

{

for(int x=0; x<index; x++)

{

if(name==username[x])

{

return x;

break;

}

}

}

//this funtion is used to check whether username is valid or not

bool flag(string name,string username[100],string password[100],string role[100],int &index)

{

for(int x=0; x<100; x++)

{

if(name==username[x])

{

return true;

}

}

return false;

}

//this funtion is used to change price of products

int changePrices(string name[],int &productCount)

{

int price;

cout<<"Enter the type of shoes you want to Change price of: "<<endl;

for (int x=0;x<productCount;x++)

{

cout<<x+1<<" for "<<name[x]<<endl;

}

cout<<"Your Option...";

while(true)

{

if( cin >> price)

{

break;

}

else

{

cout << "Enter a valid price integer value..";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(),'\n');

}

}

return price;

}

//this funtion is used to change stock of products

int changeStock(string name[],int &productCount)

{

int stock;

cout<<"Enter the type of shoes you want to Change Stock of: "<<endl;

for (int x=0;x<productCount;x++)

{

cout<<x+1<<" for "<<name[x]<<endl;

}

cout<<"Your Option...";

while(true){

if( cin >> stock)

{

break;

}

else

{

cout << "Enter a valid stock integer value..";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(),'\n');

}

}

return stock;

}

//this funtion is used to change tax of products

int changeTax(string name[],int &productCount)

{

int tax;

cout<<"Enter the type of shoes you want to Change Tax of: "<<endl;

for (int x=0;x<productCount;x++)

{

cout<<x+1<<" for "<<name[x]<<endl;

}

cout<<"Your Option...";

while(true)

{

if( cin >> tax)

{

break;

}

else

{

cout << "Enter a valid tax integer value..";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(),'\n');

}

}

return tax;

}

//this funtion is used to get feedback/reviews by user

void feedback(string reviews[],string username[],int &index)

{

string review="";

string names;

cout << "Enter Your Name: ";

cin >> names;

cout << "Enter your Honest Review: ";

cin >> review;

reviews[index]=review;

username[index]=names;

index++;

}

//this funtion is used to check feedbacks

void checkFeedBacks(string username[],int &index,string reviews[])

{

//review=feedback();

for(int x=0;x<index;x++)

{

if(reviews[x]!="")

cout <<"UserNO." <<x+1 <<"\t\t\t"<<username[x]<<"\t\t\t"<< reviews[x]<<endl;

}

}

//this funtion is used to add new products

void addNewItem(string name[],float price[],int stock[],float tax[],int &productCount)

{

string names;

int prices,quantitys,taxs;

cout <<"Enter name of the item: ";

cin >> names;

cout <<"Enter price of the item: ";

cin >> prices;

cout <<"Enter the quantity: ";

cin >> quantitys;

cout << "Enter the tax: ";

cin >> taxs;

name[productCount] = names;

price[productCount] = prices;

stock[productCount] = quantitys;

tax[productCount] = taxs;

productCount ++;

}

//this funtion is used to del existing products

void delItem(string name[],int &del,float price[],int quantity[],int &productCount,int stock[],float tax[])

{

del = 1;

viewProducts(name,price,stock,productCount,del);

int num;

cout << "\nEnter the item number you want to remove: ";

while(true){

if( cin >> num)

{

break;

}

else

{

cout << "Enter a valid index integer value..";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(),'\n');

}

}

for(int i=num;i<productCount-1;i++)

{

name[i] = name[i+1];

quantity[i] = quantity[i+1];

price[i] = price[i+1];

tax[i] = tax[i+1];

}

productCount--;

viewProducts(name,price,stock,productCount,del);

}

//this funtion is used to viewProducts table

void viewProducts(string name[],float price[],int stock[],int &productCount,int &del)

{

system("cls");

cout << "No. Products Name\t\tPrice\t\tQuantity "<< endl;

for (int i=0 ; i<productCount ; i++)

{

if (name[i] != "")

cout << i+1 << " " << name[i] << "\t\t\t" << price[i] << "\t\t" << stock[i]<< endl;

}

if (del == 0)

{

//admin();

}

}

//this funtion is used to clear screen

void clearScreen()

{

cout << "Press Any Key to Continue.." << endl;

getch();

system("cls");

}

//this funtion is used to show all existing users to admin

void showUser(string username[],string role[], int &index)

{

for(int x=0;x<index;x++)

{

cout <<"User No."<< x+1 <<"\t \t \t"<< username[x] <<"\t \t \t "<< role[x] << endl;

}

}

//this funtion is used to remove user from exiting users

void removeUser(string username[],string password[],string role[],int &index)

{

string user;

int idx;

cout << "Users are: "<<endl;

for(int x=0;x<index;x++)

{

cout <<"UserNo."<<x+1 <<"\t\t\t"<< username[x]<< endl;

}

cout << "Enter an index to delete: ";

while(true){

if( cin >> idx)

{

break;

}

else

{

cout << "Enter a valid index integer value..";

cin.clear();

cin.ignore(numeric\_limits<streamsize>::max(),'\n');

}

}

user = username[idx-1];

for(int i=idx-1;i<index-1;i++)

{

username[i] = username[i+1];

password[i] = password[i+1];

role[i] = role[i+1];

}

username[index-1] = "";

password[index-1] = "";

role[index-1] = "";

index--;

cout << user << " removed"<<endl;

}

//this funtion is a description about shop

void about()

{

cout << R"(Welcome to "Style For You," the ultimate destination where fashion effortlessly blends with functionality,

and your journey through footwear becomes a seamless fusion of style and comfort.

Our cutting-edge Shoes Selling Application, designed to enhance your shopping experience,

invites you into a virtual realm where curated collections cater to every occasion - from formal sophistication to casual chic.

With the intuitive "Style For You" interface, discovering the perfect pair is a breeze, ensuring that your every step is a statement in itself.

More than just a shopping app, "Style For You" is a vibrant community of fashion enthusiasts.

Stay ahead of trends with curated guides, exclusive offers, and share your favorite finds to let your

unique style shine. With secure and user-friendly features, your dream pair is just a click away.

Step into the world of "Style For You" where fashion is personalized, and your footwear journey is a celebration of individuality.

)";

}

//this funtion is used to get bill by customer

void bill(int &productCount,int quantity[])

{

cout << "Press any key to pay bill: ";

getch();

for(int x=0;x<productCount;x++)

{

quantity[x]=0;

}

}

//File Handling

//this funtion is used to load data products

void loadData(string name[],float price[],int stock[],float tax[],int &productCount)

{

try

{

fstream loadFile;

string data = "";

loadFile.open("data.txt", ios::in);

while (!loadFile.eof())

{

getline(loadFile, data);

name[productCount] = getFieldData(data, 0);

price[productCount] = stof(getFieldData(data, 1));

stock[productCount] = stof(getFieldData(data, 2));

tax[productCount] = stof(getFieldData(data, 3));

productCount++;

}

loadFile.close();

}

catch(std::invalid\_argument)

{

system("cls");

cout << "\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n";

cout << "\033[1m";

cout << "\033[31m";

remove("data.txt"); /// removing the file as it is corrupted

cout << "\n\t\t\t\t\t\t\t\t\t\tData file is corrupted.";

cout << "\n\t\t\t\t\t\t\t\t\t\tCreating new record file";

cout << "\n\t\t\t\t\t\t\t\t\t\tDo you want to proceed.";

//userPressAnyKey();

cout << "\033[0m";

cout << "\033[0m";

}

}

//this funtion is used to seperate data by commas in file

string getFieldData(string data, int count)

{

string result = "";

int comma = 0;

for (int i =0; i<data.length(); i++)

{

if (data[i] == ',')

comma++;

else if (comma == count)

result += data[i];

else if (comma > count)

break;

}

return result;

}

//this funtion is used to store data of products

void storeDataLocally(string name[],float price[],int stock[],float tax[],int &productCount)

{

fstream storeFile;

string data = "";

storeFile.open("data.txt", ios::out);

for (int i = 0; i < productCount; i++)

{

if (i == productCount - 1) // last line not adding "\n"

data += name[i] + "," + to\_string(price[i]) + "," + to\_string(stock[i]) + "," + to\_string(tax[i]);

else

data += name[i] + "," + to\_string(price[i]) + "," + to\_string(stock[i]) + "," + to\_string(tax[i]) + "\n";

storeFile << data;

data = "";

}

storeFile.close();

}

//this funtion is used to load data into file of all users

void loadData2(string username[],string password[],string role[],int &index)

{

try

{

fstream loadFile;

string id = "";

loadFile.open("id.txt", ios::in);

bool firstIteration = true;

while (!loadFile.eof())

{

getline(loadFile, id);

if (firstIteration)

{

firstIteration = false;

continue;

}

username[index] = getFieldData(id, 0);

password[index] = getFieldData(id, 1);

role[index] = getFieldData(id, 2);

index++;

}

loadFile.close();

}

catch(std::invalid\_argument)

{

system("cls");

cout << "\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n\n";

cout << "\033[1m";

cout << "\033[31m";

remove("data.txt"); /// removing the file as it is corrupted

cout << "\n\t\t\t\t\t\t\t\t\t\tData file is corrupted.";

cout << "\n\t\t\t\t\t\t\t\t\t\tCreating new record file";

cout << "\n\t\t\t\t\t\t\t\t\t\tDo you want to proceed.";

cout << "\033[0m";

cout << "\033[0m";

}

}

//this funtion is used to store data into file of all users

void storeDataLocally2(string username[],string password[],string role[],int &index)

{

fstream storeFile;

string id = "";

storeFile.open("id.txt", ios::out);

storeFile << "Name,Password,Role"<<endl;

for (int i = 0; i < index; i++)

{

if (i == index - 1) //last line not adding "\n"

id += username[i] + "," + password[i] + "," + role[i];

else

id += username[i] + "," + password[i] + "," + role[i] + "\n";

storeFile << id;

id = "";

}

storeFile.close();

}

// Validations

//this funtion is used to get valid integer input

bool isValid\_int(string num)

{

bool result = true;

for(int i = 0; i < num.length(); i++)

{

if(!(num[i] >= '0' && num[i] <= '9'))

{

result = false;

}

}

return result;

}

//this funtion is used to convert string to int

int convertStoInt(string num)

{

int number = 0;

int crNum;

int difference = 1;

for(int i=num.length()-1; i>=0;i--){

crNum = num[i] - '0';

number += (crNum \* difference);

difference \*= 10;

}

return number;

}

//this funtion is used to check password validation

bool password\_vald(string password)

{

if (password[0] == '\0')

{

return false;

}

int hasAlphabet = 0;

int hasNoSpace = 0;

int hasNoComma = 1;

for (int i = 0; password[i] != '\0'; i++)

{

if (('a' <= password[i] && password[i] <= 'z') || ('A' <= password[i] && password[i] <= 'Z'))

{

hasAlphabet = 1;

}

if (password[i] != ' ')

{

hasNoSpace = 1;

}

if (password[i] == ',')

{

hasNoComma = 0;

}

}

return hasAlphabet && hasNoSpace && hasNoComma;

}

# Weakness in the Business Application

* + The perimeters are way too many for most of the functions.
  + Single functionality is not properly executed in functions.
  + The carts are the same for different customers.
  + There’s no such difference between admin and customer. Anyone can sign up as admin or customer.

# Future Directions

* + Reduce the number of perimeters.
  + Replace void functions.
  + Properly execute single functionality.
  + Make a proper admin which will have the power to create other admins
  + User only sign up as customer and admin can check different customers data.