# **Project 1**

Title

**Interactive Shopping Cart**

Course

**CIS-17B**

Section

**43671**

Due Date

**May 12th, 2024**

Author

**Janaye Jackson**

# **Table of Contents**

1. [Introduction……………………………………………………………………………………3](#_Introduction)
   1. [Shopping Cart…………………………………………………………………………….3](#_Shopping_Cart)
   2. [How to use the shopping cart…………………………………………………………….3](#_How_to_Use)
2. [Versions………………………………………………………………………………………..4](#_Versions)
   1. [Time Spent………………………………………………………………………………..4](#_Time_Spent:)
   2. [Accomplished Task……………………………………………………………………….4](#_Accomplished_Task:)
3. [Developmental Summary – Current Version………………………………………………….6](#_Developmental_Summary_–)
   1. [Files……………………………………………………………………………………….6](#_Files:)
   2. [Classes……………………………………………………………………………………7](#_Classes:)
   3. [Lines……………………………………………………………………………………...7](#_Lines)
   4. [Functions………………………………………………………………………………….7](#_Functions)
4. [Flowchart – Current Version…………………………………………………………………10](#_Flowchart)
   1. [Login – matchP()………………………………………………………………………..10](#_matchP())
5. [Pseudocode – Current Version……………………………………………………………….11](#_Pseudocode)
6. [UML – Current Version……………………………………………………………………...14](#_UML)
7. [Output of Program – Current Version………………………………………………………..15](#_Output_of_Program)
8. [GANTT Chart………………………………………………………………………………..17](#_GANTT_CHART)

# **Introduction**

## **Shopping Cart**

For this project I chose to do a shopping cart. With this shopping cart there is a user/admin system in place. Each is verified upon login. Admins control what the users see in terms of the shopping cart. This project is written in C++ currently but will be translated to JavaScript and PHP. It implements a binary system to hold each user’s cart and the logins of users/admins. The catalog available to see is a text file. The user can add an or delete item from their cart and also view their cart. Their cart includes total amounts of each item, the item, and total price of all items. The admin can add items to or delete them from a catalog for the user to see, add and delete more admins, and view the shopping cart themselves. The catalog is read in from a text file when the program starts, including a lost of users/admins. After verification a menu is displayed based on who is logging in.

## **How to Use**

**Admin:** Upon running the program, the user is prompted to enter their username and password. The username is compared to a list of admins and if there is a match the user is prompted for their password. If a username cannot be found in admins it is looked for in users and if nothing is found a new account is created. Though this account cannot be an admin account. Only admins can add admins. Once the admin is validated, they are prompted with a menu to add/delete items, add/delete admins, view catalog, and to exit. If an admin chooses to add an item they are prompted to input the information regarding the item, including name, price, and image file. If an admin chooses to delete an item, they are shown the catalog of items and prompted to input the number that corresponds with the item they wish to delete. This same method is used to delete an Admin, however there must always be an Admin so when the list gets to one this function will no longer work. If an admin chooses to add an Admin, they are prompted to enter a username and password and that Admin is then added to the admins binary file. The view catalog option shows all items currently in the catalog and the exit option quits the program.

**User:** Upon running the program, the user is prompted to enter their username, if a username is found the user is then prompted for their password. The user has three tries to enter the correct password or the program exits. If the username and password is validated the program reads in that user’s cart, if it exists. If no username exist the user is asked whether they would like to create a new account, else the program exits. After validating the user, the user is then shown a menu with the options to add an item to their cart, delete items from their cart, view their cart, view the catalog, and exit the program. If the user chooses to add an item, they are prompted whether they want to add a new item or increase quantity of another item. If they choose to add an item, they are shown the catalog, and they must input a number which corresponds to the items they want. If they choose to increase, they are shown their cart and prompted to pick the item they want to increase then they are prompted with the amount they want to increase. Then the menu is shown. The delete method uses the same process it just removes or decreases an item. The view cart method shows the users current items and the total price. The view catalog methods shows the catalog and the exit method exits the program.

# Versions

## **Time Spent:**

Shopping Cart V1 – 30 minutes

Shopping Cart V2 – 1 hour

Shopping Cart V3 – 1 hour 15 minutes

Shopping Cart V4 – 45 minutes

Shopping Cart V5 – 12 hours

Shopping Cart V6 – 7 hours

## **Accomplished Task:**

Lines do not include comments

Shopping Cart V1:

Start Amount of Lines – 0, End Amount of Line – 37

CPP Files 2

Header Files

Started Outline for project

Shopping Cart V2:

Start Amount of Lines – 37, End Amount of Line – 172

CPP Files 2

Header Files 4

Started Implementing methods in Admin class, implemented constructor, and catalog

Shopping Cart V3:

Start Amount of Lines – 132, End Amount of Line – 168

Implented Add and Delete items from catalog, read in catalog, and menu functions

CPP Files 2

Header Files 4

Shopping Cart V4:

Start Amount of Lines – 168, End Amount of Line – 212

Finished User Class with exception of binary

CPP Files 2

Header Files 5

Shopping Cart V5:

Start Amount of Lines – 212, End Amount of Line – 349

Implented Binary write to file and readin to file, got stuck on this version, could not figure out at first

CPP Files 2

Header Files 5

Shopping Cart V6:

Start Amount of Lines – 349, End Amount of Line – 883

Implented User class and Login class, Finished shopping cart c++ version

CPP Files 4

Header Files 6

# **Developmental Summary – Current Version**

## **Files:**

**Admin.h**

Total Lines – 45

Lines with code - 33

White space/Comments (without code) - 12

**Admin.cpp**

Total Lines – 395

Lines with code - 319

White space/Comments (without code) - 76

**CatStruct.h**

Total Lines – 19

Lines with code - 8

White space /Comments (without code) - 11

**itemStructure.h**

Total Lines – 24

Lines with code - 12

White space/Comments (without code) - 12

**Login.h**

Total Lines – 29

Lines with code - 15

White space/Comments (without code) - 14

**Login.cpp**

Total Lines – 212

Lines with code - 184

White space/Comments (without code) - 28

**profile.h**

Total Lines – 22

Lines with code – 10

White space/Comments (without code) - 12

**User.h**

Total Lines – 37

Lines with code - 26

White space/Comments (without code) - 11

**User.cpp**

Total Lines – 322

Lines with code - 284

White space/Comments (without code) - 38

**Main.cpp**

Total Lines – 49

Lines with code - 16

White space/Comments (without code) - 33

## **Classes:**

**Admin:** Contains all methods related to the admin’s functions in this program

**CatStruct:** Contains structure of items to represent the catalog for this program

**Login:** Contains login validation for users

**User:** Contains all methods related to the users functions in this program

**itemStructure:** Contains all elements related to the items in the catalog.

**Profile:** Contains all elements related to a profile for users/admins

## **Lines**

Total Lines – 1,138

Lines with code - 891

White space/Comments (without code) - 247

## **Functions**

**Admin:**

Constructor(); initialize catalog and list of admins

Deconstructor(): Deletes catalog elements

addAdmin(): adds an admin to admins binary file

addItem(): adds an item to catalog text file

deleteAdmin(): rewrite binary file without admin that is set to be deleted

deleteItem(): rewrite text file without item meant to be deleted

getSize(): count the number of items in the catalog text file and return that number

menu(): List options for admin to select

printAdmin(): print list of admins

readIn(): read in list of catalog items from text file

readInAdmin(int): read in a line filled with data for profile structure(admin) return data

showCatalog(): List all items in the catalog structure

wrtBin(profile \*a): write profile to binary

wrtNBin(fstream &, profile \*a): Create new file inorder to remove an admin and add the new list of admins to file.

**Login**

Constructor(char \*): validate username and password, then create object of appropriate class

addUser(char\*, char\*): add new user if input does not match any of the usernames in admin/user files

readInAdmin(int): read in list of admins

readInUser(int): read in list of users

match(char\*, char\*): Check to see if username input matches – case doesn’t matter

matchP(char\*, char\*): check to see if password input matches – case matters

**User**

Constructor(char\*): Open and/or create binary file to hold user cart information, initialize user and catalog

Deconstructor(): Delete catalog structure

addToCart(): Add item to cart structure

deleteFromCart(): Delete item from cart structure

done(): Add all items currently in cart to binary file and exit program

getSize(): Get number of items in catalog

menu(): List options for user to select

ReadIn(): read in catalog to catalog structure

ReadInCart(int): read in cart to cart structure

showCart(); list items in cart

showCatalog(): list items in catalog

**Main:**

Main(int, char\*\*): run the program

# **Flowchart**

## **matchP()**

A diagram of a data flow

Description automatically generated

# **Pseudocode**

START main program

Username input

Create login object

Validate login for user/admin

If no user or admin

Create new user

Create admin object if username matches an admin

Read in catalog and admins

Show menu

If add Admin

User input username and password

Add username and password to admins structure

Add structure to binary file

Show menu

If add item

User input item name, image, and price

Add item to text file

Show menu

If delete admin

User input for which admin

Rewrite binary file without selected admin

Show menu

If delete item

User input for which item

Rewrite text file without selected item

Show menu

If show admins

Output list of admins

If show catalog

Output list of items

If exit

Exit program

Create user object if username matches user or new user created

Read in cart and catalog

Show menu

If add/increase item

User input for add item or increase item

If Add item

User input for item

Add item to cart structure

If increase

User input for item increase and how much

Increase that item

If delete/decrease item

User input for delete or decrease item

If delete

Delete item from cart structure

If decrease

User input for item to decrease and how much

Decrease item

If show cart

List cart

If show catalog

List catalog

If done

Save cart to binary file

Exit program

# **UML**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

# **Output of Program**

A screenshot of a computer

Description automatically generatedA computer screen shot of a white screen

Description automatically generatedA screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generated

# **GANTT CHART**

|  |  |
| --- | --- |
| Week | Task |
| Week 1 – April 7th | Start Outlines |
| Week 2 – April 14th | Implement Admins |
| Week 3 – April 21st | Implement User |
| Week 4 – April 28th | Start JS and php |
| Week 5 – May 5th | Continue JS and php |
| Week 6 – May 12th | Finish JS and php |