# **Project 2**

Title

**Interactive Shopping Cart**

Course

**CIS-17B**

Section

**43671**

Due Date

**June 9th, 2024**

Author

**Janaye Jackson**

# **Table of Contents**

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

# **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 Login:**

**Username: JJAdmin**

**Password: mypassword**

**Admin:** The index file loads the register screen. If an account is already made the admin can select login in the selection bar. Upon successful login. Admin has access to all tabs displayed on the navigation bar. There is a specific tab named Admin that allows the admin to view admins, view users, and view the catalog. In each of these tabs there is an edit an delete option next to the list of admins, user, or items. The admin can edit and update information on the update tabs and delete information on the delete tabs. Admins also can register new users, change passwords, and view the catalog that users would see. They cannot add items to a cart or checkout.

**User:** The index file loads the register screen. If an account is already made the user may login to their account else they have to register. If they do not have an account or they do not make one they cannot access anything on the site besides the registration page. Once they login user have the option to register another account, change their password, view the catalog and view their cart. Registering an account will pop up fields that are required and if they are not filled account cannot be made. Changing password will bring up users information and they must confirm their old password and their new password else they cannot change their password. Viewing the catalog will show all items available in the catalog and an add to cart button. Items can be sorted by price, size, and name. The view cart option will show all items added to the cart. It will also display quatitites if multiple of the same thing were added. The total price of all items will be displayed at the bottom and users will have the option to checkout. Checking out will add the users information to a database for admins to look over.

# 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

ShoppingCartJSV1 – 8 hours

ShoppingCartJSV1 – 10 hours

ShoppingCartJSV1 – 13 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

ShoppingCartJSV1:

Implemented User and Admin login and Admin Functions

PHP Files 16

ShoppingCartJSV2:

Implemented Catalog for Admin to edit. Fixed admin functions to edit users and admins

PHP Files 18

ShoppingCartJSV1:

Implemented Cart and Catalog for users.

PHP Files 22

# **Developmental Summary – Current Version**

## **Files CPP:**

**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

## **Files PHP:**

## **Classes CPP:**

**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 CPP**

Total Lines – 1,138

Lines with code - 891

White space/Comments (without code) - 247

## **Functions CPP**

**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

# **Output of Program PHP**

A screenshot of a computer

Description automatically generatedA screenshot of a computer

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 | Work on JS and php |
| Week 6 – May 12th | Work on JS and php |
| Week 7 – May 19th | Work on JS and php |
| Week 8 – May 26th | Work on JS and php |
| Week 8 – June 2nd | Work on JS and php |
| Week 9 – June 9th | Finish JS and php |

# **Github**

<https://github.com/JanayeJackson/JacksonJanaye_CIS17B/tree/main/CIS-17B/Project>

Found in file name FinalShoppingCart

\

# **Description of Concepts Used**

1. MVC - How you delineated your objects
   1. Model is displayed in css files
   2. View is displayed in my html files
   3. Controller is displayed in my PHP diles

2) Objects - Javascript/PHP - Serialization

Passing info with Javascript Objects to PHP Objects with Cookies

I passed information into PHP Objects with cookies using SESSION variables

All variables that needed to be reference multiple times were kept in a SESSION variable. Another way I passed objects was with \_GET and \_POST variables and including the variables in the reference link

3) Reading/Writing Files/Local Storage

JSON – Javascript

All information is written to a database

4) Databases SQL - Identify Entities, Xref, and Enum Tables

I have a catalog, user, admin, order, and order\_contents identity table

I also have an enum\_size table to reference the size of an item in the catalog.

Sizes are based on a number 1 being the biggest and 3 being the smallesr

5) Form Validation - Reqular Expressions

All required inputs have a required tab in their respective form and all image, float, and integer values are verified as their respective fields before being added to database tables.

6) User-Admin-Login

Admins and user use same login pade. Function is used to determine if user is admin. If not they are logged in as user. If neither are found user if redirected to registration page.

7) Cookies - Sessions - Securing Pages

Admin id and User id are stored in cookies, if either is present user can access site but if neither is present user cannot do anything. Admin id can be used to access pages user\_id cannot. Admin\_id and User\_id are validated through login function.