Name: Aubierge Bikoi

Class: CS 499

Date: 8/21/2022

Assignment: Final Project E Portfolio

**YouTube Link to Code Review:**

<https://youtu.be/zBhASsxaNVU>

**Link to GitHub :**

[aubierge1986/**aubierge**1986.github.io](https://github.com/aubierge1986/aubierge1986.github.io)

[GitHub - aubierge1986/aubierge1986.github.io](https://github.com/aubierge1986/aubierge1986.github.io)

**Self-Assessment**

Hello, my name is Aubierge, and I am in my senior year in software engineering at SNHU. Taking this class has been both rewarding and challenging. While developing this e-portfolio, I was able to know where my strengths are and what I really want to do after graduation. This portfolio also enabled me to apply my knowledge of the program, while reviewing my work a lot more critically. I have also learned to communicate with various stakeholders, while solving real life problems. For this class, I have work with Java, JavaScript, python, C++ and SQL.

Previously, I studied Software development Life Cycle, which later prepared me for the next course: CS 260(Data Structures and Algorithm). I have gained a better understanding of collaboration with team members to solve problems effectively. While working on this project, I have gained more skills and discovered some hidden passions. I have learned the various roles within a team such as a Product Owner, a Scrum Master, a Tester and a Developer. All these components are very important for the success of a team and for Quality assurance. While taking courses at SNHU, I also learned about Hardware, Software, and I found myself a hidden interest with reversed Engineering. For Database to be properly automated, there needs to be a good Data Structure and Algorithm in Place. In CS 260, we have dived into Algorithms, and often had to build them from Scratch, using a huge database. In CS 340, we used Mongo DB Software in order to treat databases. We often had to upload data, create databases and files, in order to execute specific functions. As we all know, with the Internet of Things, hackers get creative. Therefore, we should not leave security for the end. While working on Databases, we often had to create users’ authentications, in order to only grant access to those who are allowed access to data. These principles have been applied throughout the whole program, across various platforms such as SQL, Mongo and C++. Coding requires a lot of practice even after the class.

For this e-portfolio, I have included 2 artifacts: For my Design and Databases enhancements, I have included a Pizza app’s code which was developed in class while taking CS360. The code was very basic, but not well commented. I have commented the Java code and translated it into a python file. I also updated the design by making it accessible via internet, since the initial code was only compatible with Android Studio. I selected this artifact because I wanted to create a reusable code, but also make it possible for people to access data via internet.

Design:

This assignment will aim at enhance a class project from CS 360: the pizza party app. The initial artifact was built with Android studio, from a code provided by ZYBOOKS. This application was created to help party goers, with determining the amount of pizza needed for their party based on two factors: the number of guests, as well as the level of hunger. Using this assignment, I wanted to allow users to log into the app using their Facebook account, for Facebook users. In order to make it possible, I have used my Facebook for developers’ account to follow the instructions provided for Android app developers.

**Initial Artifact:**

<?xml version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"

xmlns:tools="http://schemas.android.com/tools"

android:id="@+id/activity\_main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:paddingBottom="16dp"

android:paddingLeft="16dp"

android:paddingRight="16dp"

android:paddingTop="16dp"

tools:context="com.zybooks.pizzaparty.MainActivity">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Number of people?"

android:textSize="24sp" />

<EditText

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:inputType="number"

android:ems="5"

android:id="@+id/attendEditText" />

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="20dp"

android:text="How hungry?"

android:textSize="24sp" />

<RadioGroup

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:orientation="horizontal"

android:id="@+id/hungryRadioGroup">

<RadioButton

android:text="Light"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/lightRadioButton" />

<RadioButton

android:text="Medium"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:checked="true"

android:id="@+id/mediumRadioButton" />

<RadioButton

android:text="Ravenous"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/ravenousRadioButton" />

</RadioGroup>

<TextView

android:id="@+id/answerTextView"

android:text="Total pizzas: ?"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="20dp"

android:textSize="24sp"/>

<Button

android:text="Calculate"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/calcButton"

android:layout\_marginTop="20dp"

android:onClick="calculateClick" />

</LinearLayout>

**Output:**

Graphical user interface, text, application

Description automatically generated

**Updated artifact: into python**

from tkinter import\*

root = Tk()

root.geometry('500x500')

root.title("Pizza Party")

# determine how many guests are invited

label\_0 = Label(root, text="Pizza Party",width=20,font=("bold", 20))

label\_0.place(x=90,y=53)

label\_1 = Label(root, text="Number of people",width=20,font=("bold", 10))

label\_1.place(x=80,y=130)

entry\_1 = Entry(root)

entry\_1.place(x=240,y=130)

# find out how much pizza is needed

label\_2 = Label(root, text="Total pizzas",width=20,font=("bold", 10))

label\_2.place(x=68,y=180)

entry\_2 = Entry(root)

entry\_2.place(x=240,y=180)

#How hungry is the crowd

label\_3 = Label(root, text="How hungry?",width=20,font=("bold", 10))

label\_3.place(x=70,y=230)

var = IntVar()

Radiobutton(root, text="Light",padx = 5, variable=var, value=1).place(x=235,y=230)

Radiobutton(root, text="Medium",padx = 20, variable=var, value=2).place(x=290,y=230)

Radiobutton(root, text="Giant",padx = 40, variable=var, value=2).place(x=300,y=230)

# compute the subtotal

Button(root, text='Calculate',width=20,bg='yellow',fg='black').place(x=180,y=380)

root.mainloop()

exit()# the end

I believe I have achieve my goals as this platform can now be launched on the internet.

**Data Structure and Algorithm**

In order to address the data structure and algorithm part of my project, I have selected the following artifact: Project one from CS 410. This artifact created in June 2022, aimed at using a binary code, which was later converted into a C++ code. The code generated provided the names of the clients in the database, with their usernames and passwords. Users’ authentication is very important in coding since we only need to grant access to those who have permission to access data. However, the C++ code auto generated came with some coding errors. That is why, in this section, I have decided to enhance the data structure related to this code, in order to have a fully functional code. Please note, this code can only convert into Assembly, if the initial code is working properly. I will also convert the Updated C+ + code into an assembly code, and then I will go over how assembly translate into C++.

**Initial Artifact C++:**

#include<iostream>

using namespace std;

// Declare and initialize global arrays for usernames, passwords, and clients

string usernames [] = {"j.smith", "s.brown", "r.garcia"};

string passwords [] = {"banana", "kitty", "amor"};

string clients [] = {"Bob Jones", "Sarah Davis", "Amy Friendly", "Johnny Smith",

"Carol Spears"};

/\*

\* Allows the user to change a client's service selection

\*

\*/

void ChangeCustomerChoice() {

// Initialize local variables

int client\_number = 0;

int service\_choice = 0;

int k = 0;

// Prompt for and read the number of the client to be changed

cout << "Enter the number of the client that you wish to change." << endl;

cin >> client\_number;

// Prompt for new choice selection and read input from user

cout << "Please enter the client's new service choice (1 = Brokerage, 2 =

Retirement)" << endl;

cin >> service\_choice;

// Iterates through the clients array to find a matching client number (array

index)

for (k = 0; k < sizeof(clients) ; ++k) {

if (client\_number == k) {

cout << "Selected option: " << k << endl;

}

}

}

/\*

\* Prompts user for a username and password.

\* Checks for the existence of each and then compares them

\* to determine if they match

\*

\* @return boolean: True if username and password are valid and match. False

otherwise.

\*

\*/

bool CheckUserPermissionAccess() {

// Declare local variables

string password;

string username;

int i = 0;

int j = 0;

bool match = false;

// Prompt user for username and read their input

cout << "Enter your username: ";

cin >> username;

// Prompt user for password and read input

cout << "Enter your password: ";

cin >> password;

// Search the usernames array for a matching string.

for (i = 0; i < sizeof(usernames) + 1; ++i) {

//If found break from the loop

if (username == usernames[i]) {

break;

}

// If not found, set i to a value greater than the length of the array

else if (i == sizeof(usernames)) {

// i = username array length + 2

i = (sizeof(usernames) + 2);

}

}

// Search the passwords array for a matching string

for (j = 0; j < sizeof(passwords) + 1; ++j) {

// If found, break from loop

if (password == passwords[j]) {

break;

}

// If not found, set j to value > array length

else if (j == sizeof(passwords)) {

// j = passwords array length + 1

j = (sizeof(passwords) + 1);

}

}

// Compare i and j

if (i == j) {

// If equal, return True

match = true;

}

return match;

}

/\*

\* Displays the list of clients

\*

\*/

void DisplayInfo() {

int l = 0;

// Iterates through each item in the clients array, printing each one

preceded by a number starting with 1

for (l = 0; l < sizeof(clients); ++l) {

cout << l << ". " << clients[l] << endl;

}

}

int main() {

// Declare local variables

bool authentication = false;

bool exit = false;

int menu\_choice = 0;

// Print welcome message

cout << "Hello! Welcome to our Investment Company." << endl;

cout << "Program written by Aubierge Bikoi" << endl;

// Call the ChechUserPermissionAccess function until authentication passed

while (!authentication) {

authentication = CheckUserPermissionAccess();

if (authentication) {

break;

}

else {

cout << "Invalid Password. Please try again.";

break;

}

}

// The selection menu will display after each switch statement until the user

selects 3

while (!exit) {

cout << "What would you like to do?" << endl;

cout << "DISPLAY the client list (enter 1)" << endl;

cout << "CHANGE a client's choice (enter 2)" << endl;

cout << "EXIT the program (enter 3)" << endl;

cin >> menu\_choice;

cout << "You chose " << menu\_choice << endl;

switch (menu\_choice) {

case 1 :

DisplayInfo();

break;

case 2 :

ChangeCustomerChoice();

break;

case 3 :

exit = true;

break;

}

}

return 0;

}

Output:

Text

Description automatically generated

**Updated Artifact:**

/\*

\* Description: Assignment 5.2, Project 1, SC 410.

\*

\* Author: Aubierge Bikoi

\*

\* Date: 06/12/2022

\*/

#include<iostream>

using namespace std;

// Declare and initialize global arrays for usernames, passwords, and clients

string usernames[] = { "j.smith", "s.brown", "r.garcia" };

string passwords[] = { "banana", "kitty", "amor" };

string clients[] = { "Bob Jones", "Sarah Davis", "Amy Friendly", "Johnny Smith", "Carol Spears" };

/\*

\* Allows the user to change a client's service selection

\*

\*/

void ChangeCustomerChoice()

{ // <-- this marks the beginning of ChangeCustomerChoice. Where should it end?

// Initialize local variables

int client\_number = 0;

int service\_choice = 0;

int k = 0;

// Prompt for and read the number of the client to be changed

cout << "Enter the number of the client that you wish to change." << endl;

cin >> client\_number;

// Prompt for new choice selection and read input from user

cout << "Please enter the client's new service choice (1 = Brokerage, 2 = Retirement)" << endl;

cin >> service\_choice;

string clients[5] = { "Bob Jones", "Sarah Davis", "Amy Friendly", "Johnny Smith", "Carol Spears" };

for (int k = 0; k < 5; k++)

{

if (client\_number == k)

{

cout << "Selected option: " << k << endl;

}

}

}

/\*

\* Prompts user for a username and password.

\* Checks for the existence of each and then compares them

\* to determine if they match

\*

\* @return boolean: True if username and password are valid and match. False

otherwise.

\*

\*/

bool CheckUserPermissionAccess() // <-- Notice that we're still inside of the ChangeCustomerChoice() function!

{ // We cannot write a function inside of another function. (not usually at least).

// Declare local variables

string password;

string username;

int i = 0;

int j = 0;

bool match = false;

// Prompt user for username and read their input

cout << "Enter your username:";

cin >> username;

// Prompt user for password and read input

cout << "Enter your password:";

cin >> password;

// Search the usernames array for a matching string.

for (i = 0; i < sizeof(usernames) + 1; ++i) {

//If found break from the loop

if (username == usernames[i]) {

break;

}

// If not found, set i to a value greater than the length of the array

else if (i == sizeof(usernames)) {

// i = username array length + 2

i = (sizeof(usernames) + 2);

}

}

// Search the passwords array for a matching string

for (j = 0; j < sizeof(passwords) + 1; ++j) {

// If found, break from loop

if (password == passwords[j]) {

break;

}

// If not found, set j to value > array length

else if (j == sizeof(passwords)) {

// j = passwords array length + 1

j = (sizeof(passwords) + 1);

}

}

// Compare i and j

if (i == j) {

// If equal, return True

match = true;

}

return match;

}

/\*

\* Displays the list of clients

\*

\*/

void DisplayInfo()

{

int i = 0;

/\*

\*Iterates through each item in the clients array, printing each one

preceded by a number starting with 1

\*/

// Any sort of "flow control", like for loops, while loops, if statements,

// those things are \*not\* supposed to end with a semi-colon.

// v

for (i = 0; i < sizeof(clients); i++)

{

// = is for assigning a value to a variable.

// == is for comparing whether two values are equal (common when writing if statements and while loops)

// Printing text to the screen, however, doesn't require any of those operations!

//

// TODO:

// We want to print out the value that's at the ith index of our clients array.

cout << i << clients[i] << endl;

}

}

int main() {

// Declare local variables

bool authentication = false;

bool exit = false;

int menu\_choice = 0;

// Print welcome message with my name on it

cout << "Hello! Welcome to our Investment Company." << endl;

cout << "Program written by Aubierge Bikoi" << endl;

// Call the ChechUserPermissionAccess function until authentication passed

while (!authentication) {

authentication = CheckUserPermissionAccess();

if (authentication) {

break;

}

else {

cout << "Invalid Password. Please try again.";

break;

}

}

/\*

\*The selection menu will display after each switch statement until the user

selects 3

\*/

while (!exit) {

cout << "What would you like to do?" << endl;

cout << "DISPLAY the client list (enter 1)" << endl;

cout << "CHANGE a client's choice (enter 2)" << endl;

cout << "EXIT the program (enter 3)" << endl;

cin >> menu\_choice;

cout << "You chose" << menu\_choice << endl;

switch (menu\_choice) {

case 1:

DisplayInfo();

break;

case 2:

ChangeCustomerChoice();

break;

case 3:

exit = true;

break;

}

}

return 0;

}

**Output:**

Text

Description automatically generated

**What did I Enhance?**

First, I updated the comments as I was fixing the code’s errors. In line 22 of the code, a new function started, but the curly bracket was missing to mark the end of the function. As a result, I have added one curly bracket in line 44. In line 49 of the initial code, I removed the curly bracket next to bool CheckUserPermissionAccess(), since we were still inside of the ChangeCustomerChoice() function. Hence, we couldn’t add a function inside of another function. I have also updated the comment to reflect the changes I made. Given that I had the tendency to mistake the l for 1, I changed all the l from the initial artifact into i in the enhanced artifact. Line 100 to 101 was a long comment. However, the following symbols were use “//” rather than “/\*”. Once the comment went to line 101, C++ compiler compiled it as another command and thus there was an error message. I have updated the comment from “**// Iterates through each item in the client’s array, printing each one**

**preceded by a number starting with 1**” to “**/\***

**\*Iterates through each item in the client’s array, printing each one**

**preceded by a number starting with 1**

**\*/**”. Given that any sort of "flow control", like for loops, while loops, if statements, are not supposed to end with a semi-colon, I removed the extra semi colons, and I also updated the comment to reflect the changes I made. Finally, line 125 was a long comment which extended to 126. As a result, the compiler mistook it for a command and provided an error message. This has also been fixed.

**Assembly vs C++:**

In order to get the assembly code, I entered the following command into Codio’s terminal, once I corrected the cpp file for project 1.

g++ -S Project1Files/Project1.cpp

|  |  |
| --- | --- |
| Assembly vs C++ | |
| Assembly | C++ |
| .LFE1996: .size \_\_tcf\_2, .-\_\_tcf\_2 .section .rodata .LC18: .string "j.smith" .LC19: .string "s.brown" .LC20: .string "r.garcia" .LC21: .string "banana" .LC22: .string "kitty" .LC23: .string "amor" | string passwords [] = {"banana", "kitty", "amor"}; |
| .LC10: .string "Hello! Welcome to our Investment Company." .align 8 .LC11: .string "Program written by Aubierge Bikoi" .align 8 .LC12: .string "Invalid Password. Please try again." .LC13: .string "What would you like to do?" .align 8 .LC14: .string "DISPLAY the client list (enter 1)" .align 8 .LC15: .string "CHANGE a client's choice (enter 2)" .LC16: .string "EXIT the program (enter 3)" .LC17: .string "You chose" | cout << "Hello! Welcome to our Investment Company." << endl;  cout << "Program written by Aubierge Bikoi" << endl; // Call the ChechUserPermissionAccess function until authentication passed while (!authentication) { authentication = CheckUserPermissionAccess(); if (authentication) { break; } else { cout << "Invalid Password. Please try again."; break; } } // The selection menu will display after each switch statement until the user selects 3 while (!exit) { cout << "What would you like to do?" << endl; cout << "DISPLAY the client list (enter 1)" << endl; cout << "CHANGE a client's choice (enter 2)" << endl; cout << "EXIT the program (enter 3)" << endl; |

**What did I learn? What skills did I display?**

This assignment allowed me to showcase my understanding of various languages such as Assembly, and C++. The C++ code provided recursive functions with a lot of if-else statements. Looking at the C++ code, we can also notice that this code contains a partition algorithm using two indexes such i and j as shown below:

// Compare i and j

if (i == j) {

// If equal, return True

match = true.

}.

Doing this assignment allowed me to analyze the data structure and algorithm in a project. I was able to evaluate computing solutions in order to solve a problem using algorithm and data structure by making sure the errors in the data structure are addressed. It also allowed me to enhance my security mindset, and anticipate on potential attacks, by only granting access to those that have permission to access data. While reviewing the initial code generated by Codio, I was able to implement some of the core concepts of secure coding such as documentation (by commenting), debugging, but also reviewing and addressing compiling errors within the data structure of the code. This project allowed me to convert a binary file into an Assembly file and was able to explain the functionalities of the blocks of assembly codes regarding C++, with no inaccuracies, as shown by the output.

**Databases:**

Assignment: Milestone 4 Databases

In order to address the databases component of this project, I have decided to use an artifact from CS 360, section 1.4. It is an excerpt from the ZYBOOKs and, the assignment, which was very basic, consisted in creating a pizza party app. The goal of this app was to allow pizza party goers to determine how hungry they were and how many guests were attending the party, in order to know how much pizza was needed. For this enhancement, I have decided to create a set of data tables, within the same database, using SQL. This will enable me to implement the CRUD operations, and to show how comfortable I am with managing data with SQL.

**Initial artifact:**

<?**xml** version="1.0" encoding="utf-8"?>

<LinearLayout xmlns:android="http:**//**schemas.android.com/apk/res/android"

xmlns:tools="http:**//**schemas.android.com/tools"

android:id="@+id/activity\_main"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:orientation="vertical"

android:paddingBottom="16dp"

android:paddingLeft="16dp"

android:paddingRight="16dp"

android:paddingTop="16dp"

tools:context="com.zybooks.pizzaparty.MainActivity">

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:text="Number of people?"

android:textSize="24sp" />

<EditText

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:inputType="number"

android:ems="5"

android:id="@+id/attendEditText" />

<TextView

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="20dp"

android:text="How hungry?"

android:textSize="24sp" />

<RadioGroup

android:layout\_width="fill\_parent"

android:layout\_height="wrap\_content"

android:orientation="horizontal"

android:id="@+id/hungryRadioGroup">

<RadioButton

android:text="Light"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/lightRadioButton" />

<RadioButton

android:text="Medium"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:checked="true"

android:id="@+id/mediumRadioButton" />

<RadioButton

android:text="Ravenous"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:id="@+id/ravenousRadioButton" />

</RadioGroup>

<TextView

android:id="@+id/answerTextView"

android:text="Total pizzas: ?"

android:layout\_width="wrap\_content"

android:layout\_height="wrap\_content"

android:layout\_marginTop="20dp"

android:textSize="24sp"/>

<Button

android:text="Calculate"

android:layout\_width="match\_parent"

android:layout\_height="wrap\_content"

android:id="@+id/calcButton"

android:layout\_marginTop="20dp"

android:onClick="calculateClick" />

</LinearLayout>

**Enhancement:** SQL Table

CREATE TABLE Customers2(

PersonID int,

LastName varchar(40),

FirstNamer varchar(40),

Age int,

City varchar(40),

State varchar(40),

Phone int,

email varchar(40)

);

CREATE TABLE Pizza(

Number int,

Type varchar(40),

Dough varchar(40),

Size varchar(40)

);

**Output:**

A picture containing table

Description automatically generated

In the next section, I will be inputting data into the table using SQL :

**Inserting data into Customers2 Table**

Graphical user interface

Description automatically generated with medium confidence

Graphical user interface, text, application

Description automatically generated

Insert more data into the Customers2 Table

A picture containing text

Description automatically generated

Output:

Graphical user interface, text, application, chat or text message

Description automatically generated

Inserted data into the table Pizza:

Text

Description automatically generated

In order to get the output from the table, I used the function SELECT \* FROM [Pizza]

**Output:**

A picture containing table

Description automatically generated

Now we can see that there is a duplicated data into the Pizza table. That is why, in order to delete that record, we can use the following command:

DELETE FROM Pizza

WHERE Number = 1;

This duplicated record has been deleted for both rows, as shown below:

A picture containing graphical user interface

Description automatically generated

After deleting the duplicated record, I have added the first row in the Pizza Table.

Table

Description automatically generated with medium confidence

Most people would like to have their database organized a certain way. For instance, if I want to sort data by ID number in the pizza table, I will be using the following code in SQL:

SELECT \* FROM Pizza

ORDER BY Number;

SQL will then provide me with data from the smallest id to the biggest id number as shown below.

Graphical user interface, application

Description automatically generated

Likewise, if I need to review the Customer2 table by NumberId, I will enter the following command into SQL:

SELECT \* FROM Customers2

ORDER BY PersonID;

Graphical user interface, application

Description automatically generated

I believe I have been able to use innovative skills to make the enhancement I needed for this assignment. Initially, I wanted to use Mongo DB to address this portion of my assignment. However, since Mongo, Access, and MySQL did not work, I ended up using SQL through w3schools online platform. I was able to create a database, access data, create tables, and manipulate the data. Therefore, I also believe I demonstrated some problem-solving skills in this assignment. Using Microsoft Visual Studio provided by SNHU’s VDI also allowed me to test all the code I wrote for this assignment. After all, it is very important to be security conscious, as many open-source platforms do not provide any user authentication.

Code review:

<https://studio.youtube.com/video/zBhASsxaNVU/edit>