**CS-499 Computer Science Capstone: Final Project**

Raul Ramirez



Author’s Note

This paper was prepared for the Professor Stefanelli who teaches CS-499 Computer Science Capstone 20EW1.

**Outline**

I. Introduction

II. Informal Code Review

A. Existing functionality included

B. Code Analysis included

C. Look for enhancement include

III. Artifacts & Narratives

1. Software Design and Engineering

* Authentication System

1. Description the artifacts
2. Inclusion justification of the artifact in the ePortfolio
3. Reflection
4. Algorithm and Data Structures

* Search Algorithm

1. Description the artifacts
2. Inclusion justification of the artifact in the ePortfolio
3. Reflection
4. Database

* RESTful API for a MongoDB database

1. Description the artifacts
2. Inclusion justification of the artifact in the ePortfolio
3. Reflection

IV. Professional Self-Assessment:

A. Discussion

**I. Introduction**

This Final Project is a Computer Science Capstone that includes a professional electronic portfolio (ePortfolio) that demonstrates the knowledge and skills acquired in the Computer Science with concentration in Information Security program taken at Southern New Hampshire University.

The project was performed in four Milestones. The evidence is an incorporation of three artifacts collected in three different classes taken at Southern New Hampshire University. In this capstone, the following artifacts were polished:

|  |  |
| --- | --- |
| **Category** | **Artifact** |
| **Software Design/Engineering** | Authentication System |
| **Algorithms and Data Structures** | Search Algorithm |
| **Databases** | RESTful API for a MongoDB database |

The professional electronic portfolio (ePortfolio) will serve as digital evidence of the work accomplished in the program. It intends to provide the skills, attributes, strengths, and value to potential employees.

**Abstract**

The first artifact is an authentication system that is not entirely functional. It is supposed to allow three attempts before letting a user know and therefore exiting the session supposed to use message digest five hash (MD5) to check the credentials. However, it is not working, therefore, it was improved in this capstone. The authentication system also had poor comments.

The second artifact is a Search algorithm that very poor structure and weak comments. Therefore, the structure was improved and made more effective, efficient, compact and organized. The comments were also improved.

The third artifact is a RESTful API for a MongoDB database that supposed to be developed using a Python web service framework. However, was not functional and capable of being reusable. Therefore, it was improved and capable of being used again.

**II. Informal Code Review**

In section, one can see the link to the GitHub pages ePortfolio. The existing functionality, code analysis, and the explained enhancement are addressed:

[**http://rrulo-11.github.io/**](http://rrulo-11.github.io/)

**III. Artifacts & Narratives**

This section exhibits the improved artifacts and the narratives:

**A. Software Design and Engineering**

1. This artifact is an Authentication System created August, 2018. This artifact was retrieved from the course, “IT-145 Fundamentals of Application Development.”

This authentication system queries the users to input their usernames and passwords. Then, it converts the password using message digest five hash (MD5). This is because the MD5 checks the users’ credentials found in credentials file. The authentication system only permits three attempts before allowing users that they will be allowed to access the system and therefore exiting the whole system. After that, the credentials check again the input and if the credentials are correct, then the user gets access. Then, the authentication system displays all the content. The authentication system has a user log out and it is able stay on the login screen until someone runs out their three attempts, signs in or until someone exits the system.

2. I selected this item because I find cybersecurity really interesting and I thought this artifact was the best way to demonstrate my skills and knowledge in software development. Also, since we are supposed to choose three artifacts, I decided to use some three most common programming languages. (C++, Java, Python). This artifact is coded in Java.

This artifact shows my skills in software development by demonstrating I learned to write reusable, constant, effective, and organized code. Also, by showing that I was able to follow the Software Development Life Cycle (Planning, Analyzing, Designing, Implementing, Testing, and Maintaining) and that I was able to use the best coding practices such as having direct comments, using a constant structure and indentation, and using small chunks of code to be as organized as possible. I also demonstrated how an authentication system authorizes or denies the access to a user which is a great way to show maintained code.

In my first draft I was not able to make everything work. I was still missing some processes. For instance, I was not able to make that the login screen stayed until one ran out the three attempts. I did not even include it in my pseudocode. So, this is something I improved. I also made the comments clearer and improved the output messages.

**Authentication System Source Code (Artifact)**

“package authenticationsystem;

//Importing the libraries

import java.util.Scanner;

import java.io.File;

import java.security.MessageDigest;

//Creating a class named AuthenticationSystem and inserting the method main along with the IOException

public class AuthenticationSystem {

public static void main(String[] args) throws Exception {

//Creating the scanner object to get data from the user

Scanner readInput = new Scanner(System.in);

//Declaring the variables

int numAttempts = 0;

/\* Creating the while loop that makes the system stays on the login screen until three unsuccessful attempts are made, a successful attempt is made, or user exits intentionally \*/

while(true) {

// Prompting user to input the username and password

System.out.println("Enter username: ");

String userName = readInput.nextLine();

System.out.println("Enter password: ");

String original = readInput.nextLine();

/\* Converting the password using a message digest five (MD5) hash so the

credentials are checked \*/

MessageDigest md = MessageDigest.getInstance("MD5");

md.update(original.getBytes());

byte[] digest = md.digest();

StringBuilder sb;

sb = new StringBuilder();

for (byte b : digest) {

sb.append(String.format("%02x", b & 0xff));

}

//Declaring the boolean to check if the authentication system is true or false

boolean authenticationBoolean = false;

//Creating scanner object to open identifications file

//Checking credentials against credentials in the credentials file

//Passing the path to the file as a parameter

File file = new File("/Users/raulramirez/Documents/credentials.txt");

Scanner userCredentials = new Scanner(file);

//Creating a while loop to retrieve the credentials in the file

while(userCredentials.hasNextLine()) {

String fileRecords = userCredentials.nextLine();

String columns[] = fileRecords.split("\t");

if(columns[0].trim().equals(userName)) {

//Using the hashed passwords in the file

//Checking if the password is valid

if(columns[1].trim().equals(sb.toString())) {

authenticationBoolean = true;

//Displaying the role in the file

//Passing the path to the files as parameters

Scanner roleFile = new Scanner(new File("/Users/raulramirez/Documents/"+ columns[3].trim()+ ".txt"));

while(roleFile.hasNextLine()) {

System.out.println(roleFile.nextLine());

}

break;

}

}

}

//Allowing user to log out

if(authenticationBoolean) {

System.out.println("Are you sure you want to log out? Yes (y) No (n): ");

String confirmLogout = readInput.nextLine();

//If user types (y), then exit

if(confirmLogout.toLowerCase().charAt(0) == 'y') {

System.out.println("Logged out successfully. ");

break;

}

else {

authenticationBoolean = false;

}

}

//Else if password is not valid, increment the attempts

else {

numAttempts++;

//If the password is not valid after three attempt, then print the following message

if(numAttempts == 3) {

System.out.println("Unable to login. The system is shutting down");

break;

}

//Else stay on the login screen until successful attempt

else {

System.out.println("Please sign in again entering valid credentials!");

}

}

}

}

}”

3. I think I did meet the course objectives that I planned in module one because I demonstrated that I can use solid software engineering and design skills and I was also able to show that I can implement best computer science practices in order to develop an authentication system. I also reviewed the structure of the source code, found some mistakes, and enhanced the code. As previously, made the comments more direct. So far, I do not have any updates to my outcome-coverage plans.

4. Somethings I learned as I was creating the artifact was how make message digest five hash (MD5) work. To be honest, I had no idea how to convert the so the credentials located in the file could be checked. As previously mentioned, as I was improving it, I learned how to make that the login screen stayed until it ran out of three tries. Both of these processes were the main challenges I faced. I had to asked for help from my previous instructor. Of course I also had many issues making the algorithm work. I has many complex functions, so of course I spent plenty of time working on it, testing it, and fixing it.

**B. Algorithm and Data Structures**

1. The artifact is a search algorithm that uses single-linked and doubly linked list. It was retrieved from the class, “CS-260 Data Structures & Algorithms.” And it was created back in August, 2019.

The scenario was that we were supposed to help the managers at SNHU Software to allow the users to access the items sold on a list in website called eBid Nashville.

In this artifact we were supposed to fix certain parts of the program instead of coding all of the program. The best data structures and algorithms were applied in order to have the most effective functions.

2. I selected this item because it was the item that that showed the best practices, algorithms, and tools of data structures and algorithms. The artifact was improved by making the program and comments more professional and effective.

**Search Algorithm Source Code (Artifact)**

List and Searching algorithm. Copyright: Copyright © 2017 SNHU COCE:

“// Name : LinkedList.cpp

// Author: Raul Ramirez

// Copyright: Copyright © 2017 SNHU COCE

// Description: Lists and Searching

#include <algorithm>

#include <iostream>

#include <time.h>

#include "CSVparser.hpp"

using namespace std;

// The definitions, methods, and classes can be seen in here

// The declarations are here

double strToDouble(string str, char ch);

// In here, a structure that holds the bid data was created

struct Bid {

string bidId;

string title;

string fund;

double amount;

Bid() {

amount = 0.0;

}

};

// Here is the linked-list class

// Defining the class that has the methods to perform the linked-list

class LinkedList {

private:

//Here is the internal structure for the list entries

struct node {

Bid bid;

node\* nextNode;

};

node\* head;

node\* tail;

int size = 0;

public:

LinkedList();

virtual ~LinkedList();

void Append(Bid bid);

void Prepend(Bid bid);

void PrintList();

void Remove(string bidId);

Bid Search(string bidId);

int Size();

};

// Here starts the default constructor

LinkedList::LinkedList() {

// Initializes housekeeping variables.

head = nulptr;

tail = nullptr;

}

// Here starts the default destructor

LinkedList::~LinkedList() {

}

//Appending a new bid in the list

void LinkedList::Append(Bid bid) {

// Implementing the appended logic

node\* nodeNew = new node(bid);

tail = node;

if (head == nullptr)

head = nextNode;

}

else {

if (tail != nullptr) {

tail->nextNode = nodeNew;

}

}

size++;

}

// In here a new new bid was prepended

void LinkedList::Prepend(Bid bid) {

// Implementing the prepended logic

node\* nodeNew = new node(bid);

nodeNew->nextNode = nullptr;

if (head == nullptr) {

head = nextNode;

tail = nodeNew;

size++;

}

else {

nodeNew->nextNode = head;

head = nodeNew;

size++;

}

}

// Implementing the print logic

void LinkedList::PrintList() {

node\* entry = head;

while(entry != nullptr){

cout << enrty->bid.bidId << " ! " ;

cout << entry->bid.title << " | " ;

cout << entry->bid.amount << " | " ;

cout << entry->bid.fund << endl;

entry = entry->nextNode;

}

}

//Removing the bid bidId

void LinkedList::Remove(string bidId) {

// Implementing the removing logic

node\* entry = head;

if (entry == nullptr) {

return;

}

else if (head->bid.bidId == bidId && head->nextNode == nullptr) {

head = nullptr;

tail = nullptr;

delete entry;

}

else {

while (entry->bid.bidID != bidId) {

prevNode = entry

entry = entry-> nextNode;

}

prevNode->nextNode = entry->nextNode;

delete entry;

}

}

//Searching the bid bidId

Bid LinkedList::Search(string bidId) {

// Here is the Implemented search logic

node\* entry = head;

node\* current = new node;

current->bid.bidId = "";

while(entry != nullptr) {

cout << entry->bid.bidId << endl;

if (entry->bid.bidId == bidId) {

return entry->data;

}

entry = entry->nextNode;

}

return current->data;

}

// Returning the number of elements

int LinkedList::Size() {

return size;

}

// Here is the static methods used to do the testing

//Displaying the bid data bid that has the bid info

void displayBid(Bid bid) {

cout << bid.bidId << ": " << bid.title << " | " << bid.amount

<< " | " << bid.fund << endl;

return;

}

// Prompting the users for the bid data @return Bid struct

Bid getBid() {

Bid bid;

cout << "Enter Id: ";

cin.ignore();

getline(cin, bid.bidId);

cout << "Enter title: ";

getline(cin, bid.title);

cout << "Enter fund: ";

cin >> bid.fund;

cout << "Enter amount: ";

cin.ignore();

string strAmount;

getline(cin, strAmount);

bid.amount = strToDouble(strAmount, '$');

return bid;

}

//Loading the CSV file

void loadBids(string csvPath, LinkedList \*list) {

cout << "Loading CSV file " << csvPath << endl;

// Here initializes the CSV Parser

csv::Parser file = csv::Parser(csvPath);

try {

// Here on can see the loop thar reads the rows of the CSV file

for (int i = 0; i < file.rowCount(); i++) {

// Here initializes the bid using the information from current row

Bid bid;

bid.bidId = file[i][1];

bid.title = file[i][0];

bid.fund = file[i][8];

bid.amount = strToDouble(file[i][4], '$');

cout << bid.bidId << ": " << bid.title << " | " << bid.fund << " | " << bid.amount << endl;

// Add the bid at the end

list->Append(bid);

}

} catch (csv::Error &e) {

std::cerr << e.what() << std::endl;

}

}

//C function that converts a string to a double

//credit: http://stackoverflow.com/a/24875936

double strToDouble(string str, char ch) {

str.erase(remove(str.begin(), str.end(), ch), str.end());

return atof(str.c\_str());

}

int main(int argc, char\* argv[]) {

//Command line arguments

string csvPath, bidKey;

switch (argc) {

case 2:

csvPath = argv[1];

bidKey = "98109";

break;

case 3:

csvPath = argv[1];

bidKey = argv[2];

break;

default:

csvPath = "eBid\_Monthly\_Sales\_Dec\_2016.csv";

bidKey = "98109";

}

clock\_t ticks;

LinkedList bidList;

Bid bid;

int choice = 0;

while (choice != 9) {

cout << "Menu:" << endl;

cout << " 1. Enter a Bid" << endl;

cout << " 2. Load Bids" << endl;

cout << " 3. Display All Bids" << endl;

cout << " 4. Find Bid" << endl;

cout << " 5. Remove Bid" << endl;

cout << " 9. Exit" << endl;

cout << "Enter choice: ";

cin >> choice;

switch (choice) {

case 1:

bid = getBid();

bidList.Append(bid);

displayBid(bid);

break;

case 2:

ticks = clock();

loadBids(csvPath, &bidList);

cout << bidList.Size() << " bids read" << endl;

ticks = clock() - ticks; // current clock ticks minus starting clock ticks

cout << "time: " << ticks << " milliseconds" << endl;

cout << "time: " << ticks \* 1.0 / CLOCKS\_PER\_SEC << " seconds" << endl;

break;

case 3:

bidList.PrintList();

break;

case 4:

ticks = clock();

bid = bidList.Search(bidKey);

ticks = clock() - ticks;

if (!bid.bidId.empty()) {

displayBid(bid);

} else {

cout << "Bid Id " << bidKey << " not found." << endl;

}

cout << "time: " << ticks << " clock ticks" << endl;

cout << "time: " << ticks \* 1.0 / CLOCKS\_PER\_SEC << " seconds" << endl;

break;

case 5:

bidList.Remove(bidKey);

break;

}

}

cout << "Good bye." << endl;

return 0;

}”

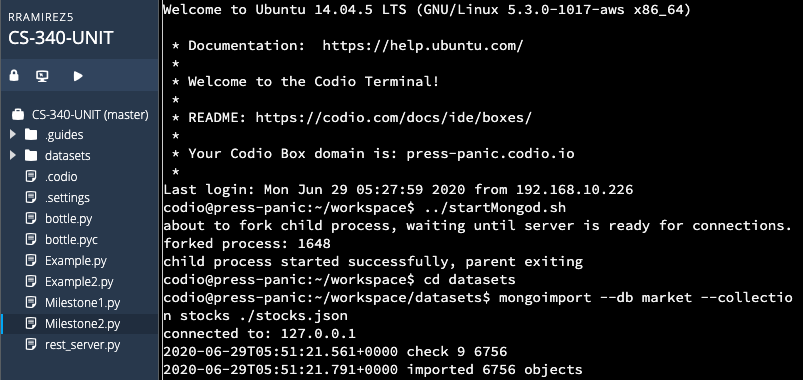
3. I did meet the course objectives that I planned in the first module. I did use the best computing solutions using the requested data structures and algorithm principles. Some of the updates I made were making the structure more efficient and making the comments more understandable and straightforward.

4. Some of the things I learned as I was creating the artifact were the different linked-lists data structures. The two of the best ways to organize data are the singly-linked and doubly-linked lists. Both of them are utilized to implement abstract data types that include pointers and arrays.

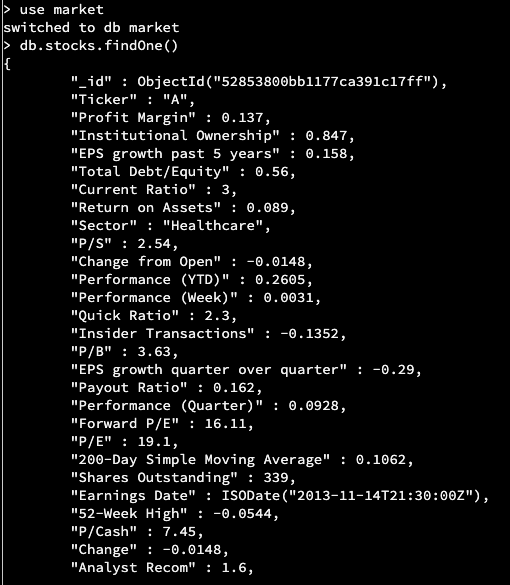
Singly-linked lists are linked lists that do not have a direction. So, from first node to the last node or the head to the tail. They hold the data and a pointer to the succeeding node. Singly-linked list do not require a lot of memory plus they are very efficient, flexible, and easy to expand because they have circular direction. Doubly-linked lists have two directions. They can be iterated in both directions, so, forward and reverse. So, the nodes have data and a pointer to the succeeding node, but, there is a pointer to the preceding node. Next, the list has to point to the first node and then to the last. They are great for iterating in sequence. Also, it is easier to delete because they do not search in all lists. They divide the search in chunks. This program uses both to be as effective and efficient as possible.

**C. Database**

1. The artifact is a RESTful API created with MongoDB shell. The first thing that was done was to create the database, “market” and the collection, “stocks” which were the database and the collection that were used in the RESTful API.



After that, the single and compound indexes were added and the documents in the, “Json file” were added as well.

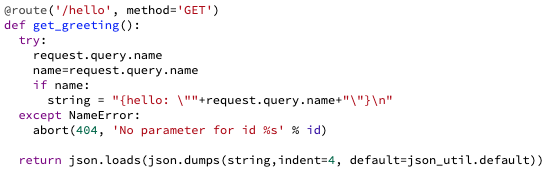


2. So, the reason why this artifact was included is because it demonstrates advanced database skills using, “python.” Also, it is innovative and creative even though it took me long time to get the interface to work.

The parts of the artifact that show my skills are the following:

In this part, the RESTful API was created using the database, “market” and the collection, “stocks.”



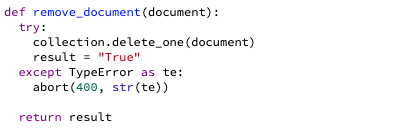


Here one can see how the command, “curl” was used to test that the protocols in the HTTP request were working in the RESTful API. One can see it returned the message “hello: “world” in Json.

../Desktop/Screen%20Shot%202020-06-22%20at%207.20.34%20AM.png

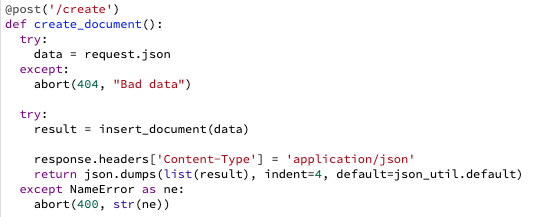
Then, in here, all functionalities were added in the framework.

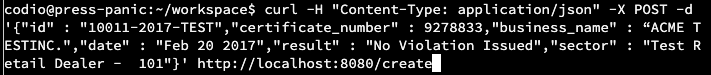


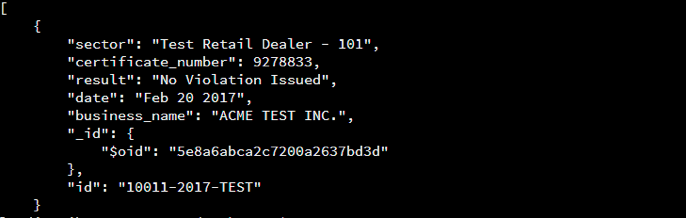


Then, using the python framework, the URIs paths were created to handle the requests and CRUD operations were implemented.

URI path for the RESTful service #1:

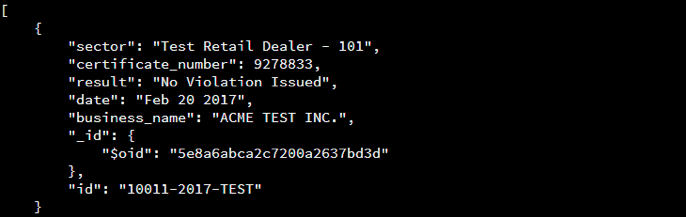






URI path for the RESTful service #2:

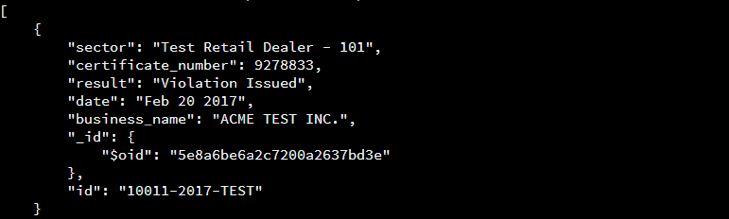


../Desktop/Screen%20Shot%202020-06-29%20at%202.24.11%20AM.png

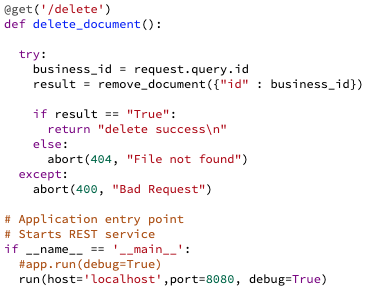
URI path for the RESTful service #3:

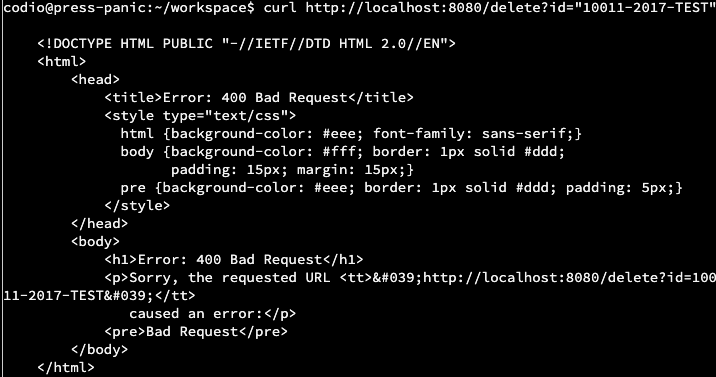


../Desktop/Screen%20Shot%202020-06-29%20at%202.24.39%20AM.png



URI path for the RESTful service #4:





3. I think I met the course objectives of enhancement of module one. This artifact did show my advanced skills and practices in database development and creating and using RESTful API framework for a MongoDB database. It is finally functional as one can see, professional, organized, and reusable.

4. Some of the things I learned were how implement and query CRUD operations in a RESTful API for a MongoDB database. I was actually quite complex. I first had to enable MongoDB, then create the database and the collection, then I had to populate them with all the data in the Json file. Creating the RESTful API was practically impossible at the beginning. I could not get it to handle all the interactions. The things that I had to fix were the logic in order to interact with the data and return all the queries. Luckily, I was able to make it work. To be honest, I really enjoy working with MongoDB since it is great for structured and unstructured data. In SQL, it is extremely hard to work with unstructured data. So, MongoDB is great because a schema is not really required since it uses documents, collections, and subcollections that are naturally fitted and stored in Json files. They are basic units of data, not really rows. They are actually lists with documents instead of tables. And they are designed like this, “key/value”. So, this is why MongoDB can handle multiple databases at the same time. The collections are groups of documents. The documents use keys which are associated with the values. That is why they are more dynamic and why the collections can be easier to manipulate. This why MongoDB is so fast, flexible, scalable when it comes to storage, and great for sharding which is when the data gets distributed between several servers allowing horizontal scaling and partitioning when the data is too large. This adds more machines and therefore this allows more traffic when creating an API.

**IV. Professional Self-Assessment**

1. Completing the coursework throughout the program and developing the ePortfolio has helped me demonstrate my abilities, shaped my professional aspirations, and prepared me to become employable in the computer science field by helping me become a more experienced, productive, skillful, and motivated individual.

This ePortfolio will help me self-promote myself for a potential job in the computer science field that I like. It also helped me become acquire more knowledge about courses I previously took. It improved my skills working in a collaborative environment through GitHub and the code review. It showed me that working constructively with other people can speed the work, and eventually be more productive, professional, proactive, methodical, informed, and embraced in a community.

The project also taught me how to communicate professionally with stakeholders. It taught me how to be more white-collar. For instance, it taught how to introduce myself properly. It also taught me how to look for opportunities in work field. It also taught how make an ePortfolio in order to show a stakeholder that I have the necessary skills and interest in the job. Everyone will be able to see my work, employees, coworkers, customer, instructors, etc. The also shaped my interest in my career goals. I made me realized that wanted to seek a career in cybersecurity.

As previously mentioned, when it comes to Software design/engineering, I how make the message digest five (MD5) work. I got to know how to make the login screen stay until it ran out of three attempts. In general, got improve the algorithm since it was not working. Some of the things I learned as I was creating the artifact were how the different linked-lists data structures work. I learned that the two best ways to order data are by singly-linked and doubly-linked lists. When it comes to working with the MongoDB database, I got to improve my skills using, MongoDB and Python. Working with the MongoDB shell in order to create a RESTful API using advanced python skills was extremely hard, but at the end, I got to make it work. To be honest, I really had a good time working with it. I got to use documents and collections which are what distinguish MongoDB and MySQL. I got to learn how documents, collections and subcollecitons are the reason why this database does not need any schema at all, and why is very convenient for structured and unstructured data, why it is more dynamic, why it is faster and scalable, why it handles multiple databases simultaneously and why it is very convenient for sharding.

The only thing that is not very convenient when working with ePortfolios when it comes to security, is that since this information shown online for everyone, the work can be exposed to plagiarism, theft, and hacking. In order to mitigate users must use the best APIs, the best version control repository hosting services, best security practices and tools, best protocols, the users should cite their work with their name, date, and sources. And the users should use plagiarizer-catching tools, and they can register to the U.S Copyright Office to legally secure their work.

If users want to be more technical, they should use the best security essentials such as Windows security essentials, install an antimalware, use a Firewall, close all the unnecessary ports, use a network protocol analyzer, patch all their software and applications, make their passwords more secure, use password managers, use a VPN, they should harden all their network devices such as the routers, modems, hubs, bridges, switches, WAPs, repeaters, servers, etc. And they should use a network monitoring software.

**References**

zyBooks. (2018) IT 145: Introduction to Software Development. Retrieved from https://learn.zybooks.com/zybook/SNHUIT145AY16-1

zyBooks. (2019). Data Structures Essentials with C++. CS-260 Data Structures. Retrieved from, https://learn.zybooks.com/

K. Chodorow (2013). The Definitive Guide. *MongoDB*. Retrieved from, https://mbsdirect.vitalsource.com/#/books/

SNHU. (2020). CS 499 Final Project Guidelines and Rubric. Retrieved from, https://learn.snhu.edu/content/enforced/623492-CS-499-Q1527-OL-TRAD-UG.20EW1/

SNHU. (2020). CS 499 Milestone Two. Retrieved from, https://learn.snhu.edu/content/enforced/598-CS-499-Q1527-OL-TRAD-UG.20EW1/Course%20Documents/CS%20499%20Milestone%20Two%20Guidelines%20and%20Rubric.pdf

SNHU. (2020). CS 499 Milestone Three. Retrieved from, https://learn.snhu.edu/content/enforced/525-CS-499-Q1527-OL-TRAD-UG.20EW1/Course%20Documents/CS%20499%20Milestone%20Three%20Guidelines%20and%20Rubric.pdf

SNHU. (2020). CS 499 Milestone Four. Retrieved from, https://learn.snhu.edu/content/enforced/5298-CS-499-Q1527-OL-TRAD-UG.20EW1/Course%20Documents/CS%20499%20Milestone%20Four%20Guidelines%20and%20Rubric.pdf

R. Browning. (2020). HOW TO BUILD AN IMPRESSIVE PORTFOLIO SITE. *Skillcrush*. Retrieved from. https://skillcrush.com/blog/impressive-tech-portfolio/