Robert Robinson

CS 499-Professional Self-Assessment

Collaborating in a team environment

As I took courses in the program, I learned how to collaborate in a team environment with others. The first experience I had with this was in CS310, Collaboration, and Team projects. I learned about scrum and the agile process. I was able to work with others in the class on building a jukebox project. I also assumed roles such as scrum master, developer as well as the project owner. I gained a firm understanding of the importance of each role and how they play a part in the overall project success. I learned to look at all the ideas of the team and pick the best one even if it wasn't my own.

Communicating to stakeholders

Also, in CS310, I learned the proper way to communicate with stakeholders. Some of the items I learned were being to identify a stakeholder, defining what the objective is, being able to plan and deliver a product as well as to measure results. The more you understand a project and the customer's need, the easier it will be to relay it to the team in terms of objectives. I need to be able to balance what the customer is asking for with what the team can realistically provide giving the time limitation.

Data structures and Algorithms

While I was going through the program, I learned about data structures and algorithms. Some of the data structures I practicing using were a list, queues, stacks, hash tables, vectors, and

binary trees. I also gained experiences in different sorting algorithms such as bubble sort, insertion sort, quick and selection sort. I feel that I'm pretty comfortable using algorithms and plan on continuing different algorithms in the future. Designing and developing algorithms gives me a chance to logically think out problems and find solutions that will maximize efficiency.

Software engineering

One of the takeaways I have from going through the Computer Science program was all the experience I was able to obtain using different languages. In my program, I learned how to used mostly Java and Python. I also used C++, C#, and JavaScript. When learning the languages, I felt it was important to learn the fundamentals of a single language first. Once I learned the fundamentals of declarations, if/then/for/while loops, functions, methods and data structures of one language, it was easy to translate it into a different language. For example, the first language I learned was Python via ZyBooks. When I learned about Java, I went through the same material but this time formatted around the Java language. The fundamentals were the same. As the new languages are created, the fundamentals of programming will stay the same.

Database

While in the Computer Science program, I learned about two types of databases. I first about relational databases in DAD220 which was an introduction to the SQL language. I learned how to create, read, update and delete data using SQL commands. My final project had me create a few databases, add data to each, use a relational table to link the data together and then output data based on a request from the user. In CS 340, server/client development, I learned about non-relational databases and created a project using MongoDB. While in the class, I learned how to

use Python to create routes that would create, read, update and delete data from MongoDB. For a final project, I created a Representational state transfer (REST) API.

Security

From this course and some research, I understand the importance of developing a security mindset. It's this mindset that will help me develop code that will take into account the security of the program as well as security for the data that is collected and used in the program. The first I need to take into account is the ability to anticipate adversarial exploits in software architecture. An article I read makes the point, that to satisfy security requirements, software architects often adopt security tactics that provide mechanisms for resisting, detecting, reacting to and recovering from attacks. Consequently, flaws in the implementation of security tactics or their deterioration during software evolution and maintenance can introduce severe vulnerabilities that could be exploited by attackers. (https://design.se.rit.edu/papers/TacticalVulnerabilities.pdf). With this in mind, it's import that I develop a mindset that will build secure coding techniques into every project I'm involved in. Also, in developing a security mindset, I need to create a test that could expose potential vulnerabilities. As part of a member of a quality assurance team or just a tester, I need to write my test that will test the code against known vulnerabilities to ensure that a secure product is being sent out. Some of the areas covered by security testing include authentication, authorization, confidentiality as well as integrity. There are a few open-source tools that can assist with security testing. One such tool is "Nogotofail" from Google. It's a network traffic tool that can exploit vulnerabilities such (Secure Socket Layers) SSL certificate verification issues, SSL injections as well as (Transport Layer Security) TLS injections. The more exploits that can be located, the sooner they can be fixed and incorporated as a best practice technique for future

iterations of the software. These practices will also assist in mitigating design flaws. One issue that I've also learned techniques that ensure the privacy of the user and their information as well as enhanced security of data and resources. A few ways to enhance security are to limit data access especially sensitive data to those who need to know. Of course, before you do that, you should identify what is sensitive data. Personally identifiable information (PII) such as social security number or a home address is considered PII and must be protected. Another method to protect data is to ensure passwords are strong and not easily guessable. Some business requires passwords that are 9-13 digits and be a combination of numbers, letters, and sometimes special characters such as @,#,$,% or &. Regular data backups are also great practice to ensure data is protected. If the data is ever breached or erased, a daily backup will assist in restoring the data.

Artifact 1:

The first artifact is a continuation of a project I started in CS 350, Emerging System Architect and Technology. The original program was a weather sensing station that was written using the Python language. The program was written to measure temperature, humidity and light intensity using a Raspberry Pi and then based on the results, would light an LED telling the user that the weather conditions met a certain requirement. As soon as the program was executed, it would start taking readings at a preset interval based on what the program dictated. The enhancement I made on this program was the addition of a user menu. The menu allowed the user to run a self-test, take a user-set number of readings based on a user set time interval as well as exit the program. This artifact shows that I can take an existing product and improve it by making it more user-friendly and add enhance it by letting the user select some of the parameters for taking readings.

Artifact 2:

For my second artifact, I chose the travel destinations slide show from CS165. The original program was constructed in Java using Swing to display five travel destinations. The user would be allowed to click a previous or next button which would allow them to progress through the different sites. Each site would display a photo of the destination, the location name, and a brief description. In creating an enhancement for this project, I utilized what I learned in the data structures/algorithm class to store all of the locations in a vector data structure. Additionally, I created two extra buttons on the button panel to allow the user to view the locations either by number or alphabetically. This was accomplished but taking the data in the vector and performing a simple bubble sort based on the names or rank of each location. If I had more destinations, I would have used another sorting technique. I also created an exit button on the button panel as well. The artifact demonstrates my skills in using algorithms and data structures and in the future, I'll be able to develop more complex algorithms that deal with larger sets of data.

Artifact 3:

The third artifact represents my work involving databases. This project virtually started from nothing. My original concept was to take a working Mongo database from one of my projects and build a front-end in JavaScript/HTML and call it good. One of the issues I ran into was that original databases were housed on a server I didn't have access to. I ended up installing a local version of the MongoDB server on my home system and created the database from there. For database operations, I installed Node.JS to handle the server-side commands and then added

a simple HTML/JavaScript front-end to allow the user to enter documents into the database. The users can also, view, update and delete data as well. I'm not currently an expert with JavaScript/Node.JS but in time with more experience, I can create more functional programs. From this artifact, I have shown, that I can learn and assimilate a new technology (JavaScript) to create a project from scratch. These skills will serve me well as new technology continues to be created in the field of Computer Science.

The link to the GitHub EPortfolio is https://github.com/RobertR1971/RobertR1971.github.io