7-1: Final Project Submission- Professional Self-Assessment

Maria Herring

CS 499 Computer Science Capstone

Instructor: Brooke Goggin

February 23, 2024

Professional Self-Assessment

During my years in the Computer Science program at Southern New Hampshire University (SNHU), I have had the opportunity to learn a great deal about many computer science disciplines, such as the software development process, data structures and algorithms, software testing and automation, secure coding practices, cybersecurity fundamentals, and emerging technologies. In addition, I have obtained computer certifications in networking and security to further my understanding of secure computing structures. Throughout my time in the Computer Science program, I have developed a willingness and commitment to learning which will continue throughout my life. This ePortfolio is a consolidated representation of the skills and knowledge I have developed throughout my studies.

Collaboration in a team environment is an important part of any work environment. Through the discussions at SNHU, I was able to collaborate with instructors and peers to further my understanding of topics and projects. Through these collaborations, I have also developed a better understanding of problem-solving and how different points of view can help improve a project. In CS-250 (Software Development Lifecycle), I learned about the different stages of agile development, and the different team roles and responsibilities of a team. I learned the value of implementing an agile framework to develop software in conjunction with assessing the impact on the entire team throughout the lifecycle.

Collaborating in a team environment to understand a client’s needs and applying project management strategies to meet those needs taught me the importance of communication, flexibility, and time management. A well-functioning team can produce more than the individual team members could on their own. However, this productivity can only be achieved through effective communication and collaboration. Developing user stories and PowerPoint presentations to communicate with team members and stakeholders emphasized the importance of quality communication and organization. In CS-255 (System Analysis and Design) I learned the principles, methods, and techniques used in system development to create a system model. System models are used in design because they are detailed diagrams and documentation that accurately define system requirements. Understanding the client’s needs and how to translate those needs into pseudocode or a flowchart that results in a program is an important part of the development process. I learned skills in how to effectively communicate technical concepts and design decisions that are coherent and sound to specific audiences, such as in a team environment.

**In software engineering and design, I used software and design solutions to develop projects. In CS-330 (Computational Graphics and Visualization), I learned to use application programming interface (API) libraries and adhere to coding best practices to develop a fully formed 3D graphical application.** The use of software libraries provided advanced functionality for data structures. Learning how to implement them was critical when choosing existing components to use in a project. **In CS-360 (Mobile Architecture and Programming), I used mobile development principles and best practices to develop a functional, user centric, and secure mobile application. These courses helped me** use well-founded and innovative techniques, skills, and tools in computing practices for the purpose of implementing computer solutions that deliver value and accomplish industry-specific goals.

**Data structures** are an integral part of computers and are used to arrange data in memory. They are essential and responsible for organizing, processing, accessing, and storing data efficiently. In CS-300 (Data Structures and Algorithms: Analysis and Design), I learned the necessity of data structures and algorithms in programming. I learned how to evaluate different programming development methodologies in algorithmic design and problem solving. I learned how to evaluate data structures and algorithms in terms of space and time complexity to determine the most efficient function for a program. These skills I learned align with computing solutions that solve a given problem using algorithmic principles and computer science practices and standards appropriate to its solution.

In DAD-220 (Introduction to Structured Database Environments), I learned how to develop a structured database environment incorporating the basic programming functionality for data management and manipulation. In CS-340 (Client/Server Development), I expanded my database knowledge by learning how to develop an authenticated web-based application that used MongoDB to store the data. These courses helped me develop a security mindset that anticipates adversarial exploits in software architecture and designs to expose potential vulnerabilities, mitigate design flaws, and ensure privacy and enhanced security of data and resources.

I have improved my secure coding mindset, organization, and attention to detail as a software developer. Applying best practices and standards helped me produce programs that are properly structured, easily understood, and secure. These skills are critical to developing usable programs that are free of vulnerabilities that could be exploited. In addition, CS-320 (Software Testing, Automation, and Quality Assurance) taught me how to integrate software engineering testing protocols and strategies that are fundamental to developing quality programs. The practical application of testing ensures the development of secure quality code by diagnosing problems early in the development process. In CS-305 (Software Security), I learned how to develop secure code that complies with security testing protocols and how to apply encryption technologies necessary for secure communication. I learned skills essential to implementing hash functions and encryption to reduce vulnerabilities and increase the security of web applications.

Course Outcomes

Course Outcome #1: Employ strategies for building collaborative environments that enable diverse audiences to support organizational decision making in the field of computer science. I reached this outcome by creating a code review that detailed the existing functionality of my original artifacts, targeted areas for improvement, and detailed a plan to enhance the artifacts. Quality assurance of a code base is critical in projects. I demonstrated the ability to perform a complete code review using a checklist that covered all the different facets of the artifact.

My insights and description of the code base included both technical and non-technical vocabulary that could be easily understood by a diverse audience. In a team environment, a diverse audience would have been able to follow the review of the project and provide valuable input to support organizational decision making. I reached this outcome throughout the artifacts, by including detailed comments describing the code functionality that another developer could easily understand. This is important to increase and strengthen collaboration efforts by creating code bases that are easy to read.

Course Outcome #2: Design, develop, and deliver professional-quality oral, written, and visual communications that are coherent, technically sound, and appropriately adapted to specific audiences and contexts. I reached this outcome by recording and delivering a code review as well. I clearly articulated the original purpose and functionality of the artifacts, areas where I could improve the artifacts, and a plan to improve them. I demonstrated the ability to communicate appropriately for specific audiences and contexts by providing a visual review that explained the artifacts in detail. I also reached this outcome by developing technically sound written narratives for each artifact, describing why the artifact was chosen, and providing specific details on how each artifact was enhanced.

Course Outcome #3: Design and evaluate computing solutions that solve a given problem using algorithmic principles and computer science practices and standards appropriate to its solution, while managing the trade-offs involved in design choices (data structures and algorithms). I reached this outcome by completing the following enhancements: I expanded the complexity of the use of data structures and algorithms for my OpenGL 3D scene. I implemented algorithms and data structures in conjunction with OpenGL functions, improving efficiency of the code. In enhancement two (Algorithms and Data Structure), the code was refactored to include code for a surface normal calculation algorithm function.

The use of an algorithm improved the efficiency and accuracy of the surface normal calculations for the points on the surface. The implementation of this function provided a computing solution that solved a given problem using algorithmic principles. The implementation of this function involved constant time operations that improved the time complexity and optimization of the code. An interleaved structure format was used to process the vertex data. The vertices, normals, and texture coordinate attributes were processed in a single buffer object instead of using multiple buffers. The tradeoff using an interleaved structure to process vertex data is that the graphics processing unit (GPU) processes the data from one location, which improves performance, but also potentially uses more memory.

Course Outcome #4: Demonstrate an ability to use well-founded and innovative techniques, skills, and tools in computing practices for the purpose of implementing computer solutions that deliver value and accomplish industry-specific goals (software engineering/design/database). I reached this outcome by completing the following enhancements: I refactored artifact code bases to use coding best practices and I implemented structure and logic for program efficiency and functionality. In enhancement one (Software Engineering and Design), I refactored the code to include the appropriate libraries for the program. I modified the logic of the main application using best practices to create separate reusable functions for the program. A switch statement was added to ensure that the program complied efficiently. Using standard coding principles, the structure and design of the application was improved to make the program more functional and more readable.

In enhancement two (Algorithms and Data Structure) I refactored the code base by creating a separate class for the shaders and separate header files. This provided more segmentation, reusability, and readability in the program. The structure and design of the application was improved by implementing additional classes and header files and by implementing standard coding principles.

In enhancement three (Databases), I developed a fleet management application that allows users to interact and manipulate data. The entire software design and engineering process was involved in the development of this artifact. The mechanisms to initiate CRUD functionality are complete and fully usable to manipulate and interact with the data. The use of descriptive variable names makes the program code clear, readable, and easy to understand. The naming conventions for methods and variables are consistent and descriptive. The implementation of the structure and logic of the code base has been designed using industry best practices and standards.

Course Outcome #5: Develop a security mindset that anticipates adversarial exploits in software architecture and designs to expose potential vulnerabilities, mitigate design flaws, and ensure privacy and enhanced security of data and resources. I reached this outcome by completing the following enhancements: including input validation, error checking, authentication, and query parameterization.

In enhancement one (Software Engineering and Design) the program was refactored to include error checking of user input. By ensuring that only valid data is entered into the application it allowed a user to operate the program properly. Incorrect input prompts the user to re-enter the appropriate data and allows the application to continue with its process. By implementing input validation this enhances the software design of the application and mitigates design flaws while also increasing security.

In enhancement three (Databases), the program was developed using error checking engineering practices that allow the application to fail gracefully, exit without crashing, and help to make the program more secure. To help prevent SQL injection, the application was developed using practices such as input filtering and query parameterization. Adherence to secure coding best practices was used to develop all of my enhancement projects, which will help ensure that the applications function properly, are secure, and are maintainable.