**Matthew Hocking: CS 499 Professional Self-Assessment**

CS 499 has provided me with an opportunity to showcase my strengths via its wide variety of focused subjects that allowed me to focus on and show proficiency in many different components of computer science. These focus areas included collaborating in a team environment, communicating with stakeholders, data structures and algorithms, software engineering, and databases, and working with a security mindset. Some of these focuses were required as the central focus of an artifact while others were part of course goals, which were required in at least one artifact. Alongside this specific coursework, my entire journey through my CS bachelor’s has led me to these specific topics many times.

I have worked in a team environment on many occasions throughout my academic career. In specifically CS-related settings, I worked on a major game project that included five other team members. This project required handling different requirements and delegating tasks related to graphics, engine work, and gameplay systems. This provided me with the opportunity to hone my skills related to teamwork in a CS setting and display leadership skills. Although this project occurred early in my academic career, I still carry the skills I gained from it both technically and socially.

Communication with stakeholders (instructors in this case) has been an important part of computer science throughout all of my courses. It became particularly important as I moved into more advanced classes that had fewer rigid guidelines for assignments. For example, my open-GL course required me to introduce lighting that properly illuminated a student-selected scene. This led to an inquiry into what exactly that entailed. How much lighting, how much shadow, how many lights? These were all topics that I discussed with my instructor while going over several lighting setups I had created to determine which was best suited for the scene. As computer science becomes more complex, it also seems to become more open-ended. Due to this, it’s important to always get as many details from a stakeholder as possible and try to communicate with them about the state of the project and its direction frequently.

Data structures and algorithms are consistently a part of computer science, but both subjects open up immensely as you begin to enter more advanced topics. You go from storing everything in arrays to working with structures such as linked lists and binary search lists based on the needs of any given situation. An example of having to work around these unique requirements was the final project of CS 300. This required students to analyze a given problem (courses with varying information) and come up with a proper solution for sorting them. My selected solution was a binary search tree due to the random nature of how the courses were added to the structure. This led to a binary search tree having a vastly faster lookup time compared to alternatives.

Software engineering is a vast and very diverse topic. My academic career has led me to work on a massive variety of projects, each with its own unique set of challenges. These projects include C++ and C-based games, MatLab math applications, Android and iPhone mobile applications, and ARM Assembly language programs for Raspberry PI platforms. This only covers a fraction of the projects I have worked on, and they have all strengthened my flexibility when it comes to working with diverse and challenging projects.

Databases are an integral part of computer science that allows for efficient storage and access of massive quantities of data. I have been introduced to and worked with both SQL and NoSQL platforms and studied the different strengths and weaknesses of each respective platform throughout several classes. Projects related to this include a crash data analysis project centered around SQL and an animal rescue dashboard developed with NoSQL (MongoDB). These two projects did a wonderful job of allowing me to learn how to work with each platform and leverage their features.

Security is an extremely important aspect to work with that can easily be overlooked. Security extends from data storage all the way to user interfaces. It’s important to keep it in mind throughout any project regardless of its purpose. My courses have presented me with opportunities to work with many different security tools in courses related specifically to security and others. This has led to me gaining experience working with various forms of encryption, creating secure connections to backends, secure coding practices, and working with various tools to confirm code and dependency security.

CS 499 led to me working directly with two specific artifacts to show proficiency in these categories and to create a final portfolio to show off these skills. These two artifacts include a jukebox application designed to hold playlists and print them out on demand and a chess implementation that allowed for two players. Both of these artifacts received heavy enhancements that evolved their functionality and showed proficiency in each of these categories.

The jukebox project initially had very little functionality and held its data in an extremely inefficient way. The application only supported viewing playlists, and all data was held in individual Java classes. This meant that a new artist would require a new .java file to be generated holding said artist's information. Alongside this, there were no features that allowed the user to explore artists or songs within the application. The primary aim of this artifact was to amend these issues.

The storage component of the project targeted showing proficiency in databases by migrating all of the data held within Java classes to a MySQL database. MySQL was chosen specifically due to it being based on SQL principles, hence requiring some additional work to make it function with a concept that has flexible variable quantities (such as a playlist). Although it requires problem-solving, the end product is a responsive and efficient method of storing data.

The database also introduced a new set of data not included in the initial application focused on improving the security of the program. This table contained student/user data focused on only allowing users to use the application if they are registered and limiting the playlists they can see to their own. This included the implementation of secure passwords via SHA hashing. This adds an extra layer of security that protects user information in the case of a data breach.

The jukebox application also received a massive upgrade to its user functionality to show proficiency in software engineering. This update included the ability to add new data entries for artists/playlists/songs, edit existing data, and view the data in a variety of ways. This led to a variety of changes in the UI and API that led to many challenging situations. This resulted in a complete rework of the initial implementation that uses classes related to playlists, songs, and students. Overall, this final application provides a much more robust user experience that allows for the manipulation of nearly every part of the database, aside from secured components (students).

Alongside the jukebox application, the chess implementation gave me the opportunity to display my skills in the other categories. Specifically, the chess implementation required that two separate players be present to play a game, which may not always be viable. Hence, a chess AI serves as the perfect solution to solve this problem and display competence in algorithms and data structures.

The chess AI aimed to be much more complex than a simple AI with just a few rules to dictate its individual moves. I pursued a Minmax-based AI that weighed each possible outcome for a given number of plys and made its move based upon what it calculated to be the most optimal while considering the opponent's (player) possible moves as well. This led to me implementing a new algorithm into a practical environment and altering it to result in the desired output. Overall, it gave me a wonderful opportunity to display competence in working with an unknown algorithm.

Alongside the addition of an AI, I took steps to alter the user interface to include additional data about the game. This includes the current turn and the turn history. Due to the interface never having access to the inner components of the chess model, this led me to generate a new class to hold the turns in a way optimal to display. This new class was composed of a rolling linked list that removed the oldest value once a certain maximum number of values was reached. This allowed for a specific maximum number of turns to display to be defined during instantiation, and the class would take care of all of the management from there. This is a great example of working with different and specialized data structures to fulfill a specific purpose.

Both these projects provided me with very specific and targeted examples of the main computer science topics aside from collaborating in a team environment and communicating with stakeholders. Fortunately, CS 499 itself is structured in a way that shows expertise in both of these categories.

Although none of the artifacts were particularly collaborative, we were encouraged to focus on documenting them both in and out of code in such a way that another programmer would be able to open the project for the first time and easily gain a basic understanding of what everything does. This focus on documentation led to me gaining a much greater understanding of commenting with purpose instead of function (commenting WHY something is there as opposed to only commenting on WHAT it does). An example of this focus is individually focusing on the individual parts of each SQL statement in the jukebox API instead of simply leaving a comment denoting that the statement does *X*. This gives any programmer foreign to the project a much better platform to begin learning about the project.

Alongside this, communication with stakeholders proved to be more important in CS 499 than in any other course throughout my academic career. As each student’s path through the course is completely unique and determined by them, keeping up to date with instructor feedback and reaching out to solve discrepancies was essential to success in such a dynamic course.

Overall, CS 499 has allowed me to showcase additional proficiency in each of the major components of computer science in an accelerated and dynamic environment. It has also allowed me to refine multiple pieces of software to the point of being portfolio pieces and introduced me to a variety of new skills and tools.