Joshua Ray CS 499 – T5477

Enhancement One: Software Design and Engineering

The artifact used for this enhancement was a bot I made to run in Discord. The bot uses text commands and direct messages to play Texas Hold'em with other Discord users. I call this bot Poker Bot and I created it about 2 years ago as a side project. I selected this artifact because it was a project that I never finished but always wanted to. I also chose this because it was something I created before I had many of the skills I have now and would be able to easily find enhancements that would display my growth. Specifically, in the category of software design and engineering, I noticed I had a logging service that was not utilized and was designed poorly. The "service" was a static class that used methods to print to console with a timestamp. After implementing this enhancement, the service was redesigned to be an object that is fed to the necessary components using dependency injection. In my experience, dependency injection is an important concept to understand when looking for a C# or .NET development position, and this enhancement proves that I can use dependency injection efficiently.

I believe with this enhancement that I did meet the course objectives I planned to meet. I had originally planned on creating templates that would be used to create specific logs. I realized while working on the enhancement that creating log objects that extend a log interface or abstract class would allow me to create these templates while ensuring they follow a common structure. I also originally planned on forcing the user to use an environment variable to specify which logging level they would like. I decided to take an extra step and prompt the user for a level when one is not given. This creates a better user experience while displaying my console application design skills.

While creating this enhancement, the most useful learning experience was choosing between an abstract class and an interface. I have been faced with the question of the differences between these in an interview and realized the importance of knowing when to use which. In the case of my artifact, I decided to go with an abstract class to avoid the extra declaration of the shared properties. Other than this minor detail, and interface would have served the same purpose. I also had the challenge of deciding where something will be logged and how they would be logged. I decided the template objects would override the ToString() method to get the actual string template. From there, I had to decide where to implement the logging. I realized the best route would be to include it in the modules, but I decided to include it mostly in the poker service until I am able to reorganize the logic. I feel confident I will be able to find less key points that useful logs could be created to help with the overall readability of the project as well as implementing future changes.