George Velonis

2020-03-28

SNHU CS Capstone

Software Design/Engineering

The artifact I chose to enhance for the Software Design/Engineering aspect of the capstone requirement was my final from CS-340 Client Server Development, completed February 2020. It was a monolithic file that contained all the project requirements but not in an organized way. It was a ReST API, command line interface, and multiple levels of functions to interact with a mongo database. I selected it because it represents the most complete work I still had but also represented a work that had a lot of functionality but was not well designed.

To improve it I split the single large file in to four separate files, two main applications and two classes. First is the ReST API which can now be run on its own and only contains the API calls, second the command line interface which is just a menu. They both are used to access the functions of the MarketDbIfc class which does the heavy lifting of the project. It contains all the functions needed for the main applications, but they are specific to the documents found in the market database. That class uses two instances of the MongoIfc class in the fourth file, which contains basic functions for accessing a mongodb database with generic documents. The majority of the work done was refactoring an adding comments, the logic itself is unchanged. I also created a shell script to initialize the market database, create indices, and import sample data.

I’ve definitely met the course objective I intended to with this artifact – CS-499-04. By using an object oriented design and widely used tools in the form of MongoDB to give the user access to data useful to the finance industry, I demonstrated the 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.

The process of implementing the enhancements was straightforward enough, but it was also new to me. It was the first time I created and used my own objects in python, so I had to teach myself how to do that. I also struggled at first trying to create overloaded definitions, but a little research showed that in python you can’t overload definitions but can use default values and conditional logic instead. In the \_\_init\_\_ method for the MongoIfc class I use default values for the host/port, and in the read\_doc method for the MarketDbIfc class I use conditional logic to see if a second argument was passed. The other challenge I faced was in getting a development environment going, not because of technical issues but due to indecisiveness and too many options. I originally spun up a debian VM and got PyCharm, mongod, PyMongo, and bottle installed and setup. I did my code review on that but due to resolution and performance issues I realized it would slow me down too much when it came time to make the changes. I then had to decide between Windows on my gaming PC or Windows on my laptop but wanted to stick to developing on Linux so I had to take the time to install a Ubuntu derivative on my laptop and reconfigure my development environment.