**CS 499 Final Narrative**

The artifact I chose is software written in a python script that will query and manipulate data within a Mongo database using algorithms and data structures. I created this program in one of my last classes just a couple months ago, so the material is freshest in my mind. In addition, there were features that I wanted to add but could not be due to time constraints. This gives me the opportunity to work on adding those extra features. The artifact shows my experience with the python language, as well as my organization and modular approach to writing code. My improvements will continue to showcase these skills, as well as my use of data structures and algorithms to manipulate a vast Mongo database. My enhancements will also show how I can build upon existing algorithms and data structures to expand their functionality. The database used with this artifact contains a wide range of data relating to stocks, to include the company’s identifying information, as well as the health and trends of their stocks.

My initial enhancements covered both the design of my code and the algorithms and data structures used. The database related enhancements were planned for the second iteration of improvements, but I did not get to them until the third round of enhancements. Since my goal is to expand the functionality and design of the software to include option selections, I created a WHILE loop in the main function to show the user a menu of options until they decide to quit or not perform any further actions. After the user completes or exits from a task, the WHILE loop asks the user if they wish to complete an additional task. This will continue until the user chooses not to continue or selects the quit option. Within the WHILE loop, I added IF statements that take user input and pull up the related function. In addition, there is a prompt asking the user if the selected option is the step they want to complete. Before the WHILE loop initiates, the user is asked if they wish to start the RESTful API service. If they choose no, it will continue to the WHILE loop. Each time I attempted to add the RESTful API option within the WHILE loop, the RESTful API service would not initiate, and the loop would just move to the menu prompt. I wanted to get everything under the same loop, but I do not anticipate having time to troubleshoot this issue.

My next steps were to continue adding options within each function to confirm they wish to fully execute their changes, such as update, delete, and create. This proved to be difficult, as my efforts resulted in the function throwing the exception. The confirmation option is the only enhancement I wanted to get done in the first iteration for categories one and two. Unfortunately, due to the complexity of incorporating data mining for category three, I thought that I might have to remove the confirmation option from my enhancement plan. If I was able to quickly add the data mining, and time permitted, I still wanted try to add the extra options, but it became a much lower priority.

In my second iteration, I struggled with the logic of where to put the loops and how to call the necessary functions. At first, I tried to put everything in the main function, but the data structures were not working together as expected. Instead, my IF statements would get stuck on the last action and not call the appropriate function aligned with the user selection. Instead, it would call the last function under an IF statement. Another hurdle I ran into was giving the user the option to run through the menu again. I was able to get it to bring up the menu again, but when the user selected a new option, the program would exit out completely. To overcome both obstacles, I defined a new function for just showing the menu and taking in the user’s input. From there, I assigned a variable to the menu function and created the WHILE loop to call and compare the user’s input fed to the menu function. I learned that assigning the menu function to a variable allowed the IF statements to match the user input successfully. I then learned that the WHILE loop was how I could have the user loop through the menu continuously until they decided to quit. However, I had to do some research on Booleans to make sure the loop worked as intended. I still need to figure out how to have the option to start the RESTful API service within the loop logic.

Moving to my next round of improvements, I continued to build on my initial enhancements regarding the design of my software, as well as the data structure and algorithms used. I was able to successfully find a way to add the confirmation option for any function that modified the database. I learned that I did not have to have all actions within the TRY/EXCEPT logic. In fact, I discovered that I could have the TRY within an IF loop, which is how I enabled the “Are you sure” option. This ended up being easier to implement than I expected, and I was glad to find a logical way to include this functionality. I have a line of code that states if the selection is less than 1 or greater than 8, the selection is invalid. This is where the program would exit, which did not make sense, as it didn’t give the user the option to continue. I spent several hours trying to rethink the logic of my algorithms to ensure the menu loop flowed as expected. It was a point of struggle to figure out how to keep the menu options looping but still informing the user that their selection was invalid. Sometimes the simplest solution is the best. In this case, I removed any arguments that told the program to exit and left only a simple print statement telling the user the selection was invalid. The program then moves to the next line of code that asks the user if they wish to continue. If they answer yes, the loop starts over, and if they say no the loop exits the program. At this point, I still have not figured out how to include the option to start the RESTful API service within the menu loop, so it will only prompt the user at the beginning of the program start. I believed I had an idea of how to get it to flow with the loop, but I needed to take some time to test it out.

After this enhancement, I will have met all my goals for categories one and two, which just left category three related enhancements. My software design is cleaner and organized logically with the added algorithms for added functionality. Now, implementing data mining capabilities is all that remains. I was ahead of my anticipated schedule for my enhancements, which allowed me to really focus on the data mining enhancement planned. I learned that aggregating is a common way to data mine MongoDB databases. Therefore, I will focus on improving the current aggregation function to allow broader user inputs. Initially, my code requested specific input and outputs a limited amount of data. I want to allow the user to use any search criteria and decide what output they wish to see. I’m unsure if the output can be manipulated much, but I am hopeful that I can widen the type of data the user can input as the search criteria. In addition, I will explore expanding the output of the RESTful API service aggregations to provide more information to the user.

My goal for this enhancement iteration was to expand the user’s ability to query the database, to include broader aggregation. Initially, the script took specific field data from the user and output predetermined data. I was able to successfully change the input criteria allowing the user to choose what they wish to search and output. This ended up being much easier than expected, as I used what I learned on previous enhancements. When updating the primary functions of the code, I did not run into any challenges. I had a clear idea of how I wanted the program to perform, and I understood what code I needed to change to make it happen. In addition, I was able to duplicate a few lines of code across multiple functions, making the enhancement even easier. Since I made the input more open, I needed to change the menu selections to better align with what the user could accomplish.

Since I was able to accomplish my database related improvements so quickly, I explored enhancing the RESTful API service functions. However, when I attempted to broaden the search abilities of the RESTful API service, I was not successful. Currently, my program takes input for specific field entries, which limits the user on what they can query. If the user attempts to READ any field other than the company name, it will not return any successful results. I will do some research and continue to explore this as I review and refine everything, but as of now I have met all my goals for my enhancements. The RESTful API service enhancements would be an extra improvement not originally planned.

My code is designed in away that will allow potentially adding authentication features, such as verifying login credentials and access privileges. This would add an extra layer of security to ensure only those authorized can access or modify selected databases. This could include a user having the rights to read a database, but not have the ability to delete any entries. Further, I would be interested in exploring if the ports used in the RESTful API service can be changed to secure ports such as SSH or TLS protocols.

ePortfolio: <https://github.com/kellybedard316/ePortfolioKB>