CS 340 Reflection

This course was focused on creating a very simple data driven application from front to back. We built a database using the NoSQL database program MongoDB, and linked it Jupyter Notebook using the PyMongo module. We then created our own CRUD module using PyMongo and Python, and implemented it in Jupyter Notebook. Lastly, we used Dash framework and JupyterDash to create a simple UI that displayed information retrieved from the database. We also practiced writing effective README files for more complex programs.

This course was a big exercise in learning and implementing programming tools. Before this course started, I had no experience with Mongo, PyMongo, Jupyter or Dash, and very little experience using Linux. So I spent a lot of time this term digging through API documentation and manuals. I really enjoyed working with Linux, Mongo and Python, but I had some troubles with Dash, especially trying to implement the JupyterDash library. While I still feel like I have a lot to learn about Dash framework and MongoDB, I got a really good foundation in the principles of creating a program over the full stack.

One really interesting part of the project for this course was creating my CRUD module. I tried to keep the module as adaptable as possible, although depending on how you plan on using it some of the hard coded bits like the database, collection, and connection could be swapped to parameters. All in all, the module could be used on pretty much any MongoDB with very little modification. The fact that the module is so simple means it can be used in a variety of different ways. You can predefine queries, like was done for this project, or you could collect query information from users and pass them into a python dictionary, then pass them to my module.

When I’m considering how I am going to design a project, I usually try to imagine how each object I’m designing is going to be used, or could be used in future. If we take the example of our CRUD module again, I designed it to be as simple as possible, so it could be used by the project I was currently working on, and could be easily adapted and used for future Mongo projects. On the other hand, if we look at the notebook we made for the UI for this course, I made it very specialized, because there’s no real reason it will be reused. Essentially, I try to match the modularity of a piece of a project to the specialization it requires. If a part of a project has requirements that are very specific to the project, or very unique, then I will design it as such.

I think this kind of decision making is very central to computer science as a whole. The field is very cumulative, with better and more efficient solutions almost always built on the back of the simple and modular solutions currently in use. Having this understanding, and the skill to implement it makes a programmer more effective at designing new systems and programs, and therefor more valuable to their employers and coworkers.