Milestone 4: Database

Chris Charlebois

RESTful API with MongoDB

*Briefly describe the artifact. What is it? When was it created?*

This is a MongoDB running on a Debian 10 Linux server with a Python script providing the RESTful interface using PyMongo and Bottle methods. This RESTful interface was originally created for a database course featuring MongoDB. That course utilized the Codio platform. This implementation is running on a platform that I built and maintained for this purpose.

*Justify the inclusion of the artifact in your ePortfolio. Why did you select this item? What specific components of the artifact showcase your skills and abilities in software development? How was the artifact improved?*

This project represents a functional system that is similar to systems that could be used in real-life applications. It provided a practical application and includes all the systems required to support the primary application. The improvement was the recreation of the systems that were supplied by the Codio platform in a specifically created environment.

*Did you meet the course objectives you planned to meet with this enhancement in Module One? Do you have any updates to your outcome-coverage plans?*

This project demonstrated some of the infrastructure aspects of database management. I still needed to preform general database management in order to verify the data was being recorded. I did increase the scope of the enhancement by using the BottleDeamon library to disassociate the script from the terminal that started it so that the script will continue to run when the terminal closes.

*Reflect on the process of enhancing and/or modifying the artifact. What did you learn as you were creating it and improving it? What challenges did you face?*

Install the Debian 10 system was straightforward. The Mongo install was a little more difficult. A bigger challenge came from configuring the security and the internal firewall to allow the traffic. However, the most difficult was presented by the need to convert the Python script into a daemon that will continue to run after the terminal closes. Ultimately, I was not able to get this working. When I used the bottledaemon library, the python script ran normally, but the pymongo method would not write to the data base. I found that the pymongo method is not fork-tolerant and will not connect to the database after bottledaemon forks the process. While testing this, I was able to confirm that the RaspberryPi Remote Sensor.py script could store readings for several minutes and not lose data, uploading the data after the connection is restored.