Bi-Weekly Progress Report

Project Short Name: Genesis: Configuration Management

Your Name: Douglas Swanson

Number of hours worked week 1: 12 hours week 2: 8 hours

Report Due Date: 10/30/22

1. Tasks and Targets set for the report period
   1. Finish up Genesis Scenario Service.
   2. Start Working on Spacecraft Service.
   3. Start creating the service broker.
2. Adjustment and revisions made against the original tasks and targets. Explain the reasons if adjustments were made.
   1. The time allotted for generating the Spacecraft Service was taken up by researching the service broker.
   2. The service broker jumped in priority due to interest from the project sponsor at Boeing.
      1. Thus more time has been allocated to working on the broker, which is almost complete at this point.
3. Summary of Accomplishments and the contribution to the project.
   1. Completed the Genesis Scenario Service code and about half of the testing.
   2. Started working on the JSON version of the API documentation that will be sent from the service, when accessing the base route (‘/’).
   3. Started creating the Spacecraft Service, which is currently just the boiler plate code seen within the Genesis scenario service portion.
   4. Completed about half of the Service broker code, which allows the services to register and send keep alive messages to Redis.
   5. Also installed Redis within the project development environment.
4. Elaboration of the work done.
   1. Completed the Genesis Scenario Service code and about half of the testing.
      1. The service code has been written and has been tested for a few possible results using a tool called ThunderClient, which is an extension of Visual Studio code. This client allows me to build mock requests and send them to the service and displays the received response. This allows for some testing and allows me to actively debug the service as though a request was sent by a client application. Half the testing has been accomplished using the ThunderClient, the rest will need to be written using the JEST Library. The code still needs some internal commenting, which shouldn’t take too long to complete. But the service works as intended and can be considered complete at this point.
   2. Started working on the JSON version of the API documentation that will be sent from the service, when accessing the base route (‘/’).
      1. The JSON representation of the API documentation has been started, it is currently sitting at about 50% complete. This JSON representation will be sent to the user when they access the base route of the service with a GET request with a ContentType header of application/json. Should complete this within the makeup time period scheduled later this month.
   3. Started creating the Spacecraft Service, which is currently just the boiler plate code seen within the Genesis scenario service portion.
      1. Made a duplicate of the Genesis scenario service project within the same directory and started pulling out the scenario specific pieces to get ready to add in the Spacecraft Service database access code. This is about 10% complete and will be resumed after the Service Broker portion has been completed.
   4. Completed about half of the Service broker code, which allows the services to register and send keep alive messages to Redis.
      1. The service broker portion of this project is basically a library that allows the other components of this project to access the data that they need within the Redis database. In this case, the services that are currently running and their IP addresses. The code for allowing the services to register themselves within the data base has been created as well as the code that allows them to send “keep alive” messages to the database in order to keep themselves registered while they are running. The second half of this library will be the code that allows the other components to find a service they are looking for and get the ip address or be notified that the service they are looking for is not currently running and that they should go do something about it. This component of the project should be done later this week.
   5. Also installed Redis within the project development environment.
      1. In order to test the service broker library code, Redis needed to be added to the development environment. This was done by installing Redis Server on my computer within the Windows Subsystem for Linux (WSL). This was done because the later versions of Redis are available on Linux, but the last version that was ported to windows is at least 3 versions behind.
5. Conclusion Remarks
   1. All in all this was a very productive past 2 weeks and I am confident that I can keep the momentum up through the rest of the month and generate all of the deliverables for this project. The first service is up and running with the second on its way, which should take less time than the first due to being very similar to the first service. The service broker took less time than anticipated to generate the connections to Redis and create keys within the database, it looks like the second half will be about the same amount of work. This should allow me to make up some lost time. There was a minor change in the schedule due to a request from the sponsor and them wanting to see what the service broker would look like, I think the idea of a service broker is new to them and this is a comfort request than anything else.
6. References used
   1. <https://docs.redis.com/latest/rs/references/client_references/client_nodejs/> (accessing redis from NodeJs).
   2. <https://nordicapis.com/5-examples-of-excellent-api-documentation/> (examples of api documentation)
   3. <https://stoplight.io/api-documentation-guide> (API Documentation best practices)