During implementation of the services and various tests that were needed for the application, many different techniques were utilized to ensure that the application would meet the needs of the customer. Additionally, various techniques were weighed and discussed to find what the best approach would be to develop an application that would be tested for the use cases that an application would be used in.

During development of the application, testing was needed to ensure that implementations were following the guidelines given for the application. For example, testing needed to ensure that the application would react correctly to the various create, updated, and delete requests made for each object where each object had specific requirements on how the calls would react to the parameters give. To align the application to the software requirements, methods of testing were employed to ensure that the application would react as expected. Testing was implemented in a couple different ways. One implementation of testing was utilizing unit testing. Unit testing ensures that the code does what it is supposed to do (Thanh, 2020). This can be done through testing functions and methods that are going to be used in the implementation and testing their outputs and how they react. An additional method that was implemented in the testing approach was through integration testing. Integration is done through running unit tests to test individual units, and then testing units together to make sure that the connections are all working correctly (Thanh, 2020). Integration testing implementations can be seen through the testing that was done on the services, where after testing the class itself, the functions are called in the service which relies on the object’s code being correct and the service’s code also being correct. This ensured that the integration between the class and the service would work as expected and yield the correct results.

The unit tests were implemented such that the tests would cover all the needed interactions with the services. For example, the different operations that were implemented in the services, such as creating, updating, and deleting were each tested using the different cases. Creating was tested with using correct values and incorrect values to ensure that the service and class would handle the bad cases as well as the correct cases. Additionally, these tests were implemented in a way that would define clearly what the problem would be such that the test wouldn’t pass just from not throwing an error, the application had to throw the correct error at the right time. This is how I can know that the test coverage shows how well the application in the end is tested, not purely from the amount of coverage but from the mentality that was used during development.

To ensure that my code was technically sound, I utilized tests that would test the edge cases and cases in which that would be close to not being considered correct. For example, I would use labels that would be close to the max length and over the max length to ensure that the case would react exactly as expected and that the test being a success would not be an outlier that didn’t properly convey the capabilities of the application. Additionally, code was ensured to be efficient by analyzing the time and space complexities of the application. For example, instead of implementing list functionalities, the data was stored in a HashMap to make operations faster on the data. Additionally, functions were kept very simple with the looping and other iterative operations that were needed to be done. To analyze improvements and the application status in performance, the test durations could be observed and analyzed to ensure that the application was not overly complex.

Software testing methods that were not included in the approach were testing methodologies such as load testing and Penetration testing. Load testing is a form of testing that tests how an application will respond to rising demand (Thanh, 2020). Load testing helps to ensure that if demand were to increase in the application, that it would be able to handle the requests. This form of testing was not done since scale was not a consideration in the requirements, however, since the space and time complexities of the application were kept low, and the overall logic is simple, the application should be able to scale well. The application may approach a point where it would have too many items in memory to hold, but otherwise it should scale well. Penetration testing is a form of security testing that tests how well an application is secured (Thanh, 2020). Penetration testing could be useful to applications that are sensitive and require being secure. Since the implementation so far has only been on the services, the security would be implemented in the API implementation so penetration testing would be done on that layer of the application.

Using caution when testing an application is crucial. If caution is not taken to methodically test an application, you could potentially miss issues or implement tests that are not correct. During development, I was methodical to ensure that I covered various cases that may occur and was careful to not miss any cases in which the application would need to operate correctly by providing a range of different values in the objects.

When reviewing the code, a developer must be careful to remove bias. If the developer responsible for the development is reviewing the code, special exemptions could be made during their development that would help to hide certain considerations, this could be done either deliberately or without being aware of the vulnerability they were introducing. During development, I was careful to eliminate bias by implementing many different test cases that would ensure the application would follow the requirements. By sticking to testing the requirements, I ensured that the application was tested to work, and not reduce workload for me.

To be a good developer in the field, you must have discipline. Often, not too many others will see the benefits of adding comments or integrating unit testing. It is your job as the writer of the code to ensure that you are leaving it in good condition such that another could come and edit it easily. Cutting corners in code may work in the short term but I have found that the more corners that get cut, the harder it is to expand the functionalities later. In applications I have developed, I have implemented quick solutions that later have required being rewritten again due to additional features being requested on top of what I had already built, this both wasted time that I spent when I could have developed it correctly in the first place, and made the application have the potential to not operate correctly after the new implementation. In my work, I will work to avoid technical debt by ensuring that I am delivering functionalities that are well developed and properly tested and are well documented. This will allow for applications to be easy for others to later add to or fix if need be.

References

Thanh, V. (2020, May 22). *15 testing methods all developers should know | CircleCI*. CircleCI;

CircleCI. <https://circleci.com/blog/testing-methods-all-developers-should>

know/?utm\_source=google&utm\_medium=sem&utm\_campaign=sem-google-dg—uscan

en-dsa-maxConv-auth-nb&utm\_term=g\_

\_c\_\_dsa\_&utm\_content=&gclid=Cj0KCQiA6LyfBhC3ARIsAG4gkF8ythuJv6WHcJtN

uW0ECTPlui4g4TpWoh7-IBP2\_tZUFXw90TB3FUaAubZEALw\_wcB