Andres Trujillo

9/29/2024

Southern New Hampshire University

CS320 Software Testing, Automation, and QA

Prof. Parul Hirpara

Module Four Journal

**Testing Approach, explain the extent to which the testing approach used was aligned to the software requirements.**

The testing approach used for the Task object class and the Task Service class aligned to the specified requirements. Aligning to the requirements involved creating a list of items that were directly related to the project’s requirements. So, for instance, the Task object is to have a field member variable for the Task name. The requirements were that the name cannot be null or be longer than 20 characters. This means that within my code I had to validate the way I created a Task object. The name couldn’t be null so that was a conditional line of code, and I had another conditional line of code checking the length of the name. Once these two requirements were satisfied, I could let my program continue running towards the creation of a Task object. A task object consists of a few other variables that had the same type of conditions implemented by understanding our requirements. The other variables or field members were that it needed to have a unique Id, and another variable was that it needed to have a description of our task. These variables had certain requirements given to us by our client of the mobile application. The Id had to be unique, so we needed to check for uniqueness amongst our database to identify any duplicates. The description of our Task object needed to be shorter than 50 characters long, and it couldn’t be null. The Id couldn’t be null as well, this allowed us to always be able to debug our mobile application in the future with meaningful data from a task source.

The Task Service class was bound to certain requirements that were the same as our Task class. Although, we needed the ability to add, delete, and update Tasks. But the creation of Tasks still had to meet our requirements, intertwining the two classes making it more complex for testing purposes. The Task Service required a database implementation, and that was achieved with a hash map pointing a unique Id to a Task object keeping track of all our Tasks scheduled within the Task Service. When adding tasks, we had to ensure that there were no duplicates, and the task wasn’t null. This required a valid Task object and a valid unique Id otherwise we would generate an error and adding the task object to our database wouldn’t succeed.