Ryan Summers

Kallysa Wilson

CS-320

04-20-2024

# 7-2 Project Two

# The client requirements where pretty straight forward. There was nothing overly complex. The code I wrote met all of the client’s requirements. The contact class requirements for example could not be null and had maximum character lengths. I looked for null and the length in one line using || to save lines. I think this looked a lot better as well.

# A screen shot of a computer program Description automatically generated

# All of my Junit tests where at or above 90% coverage. I read that anything over 80% is considered good. I did get better coverage the more I wrote though. I think my coverage rate will improve the more I use Junit tests in my work. I made sure my test where efficient by trying to account for everything. For example, in the ContactServiceTest.java file I wanted to make sure the delete function worked. The code below deletes a contact, then tries to delete the same one again. It can’t assuming the original was deleted.

# A computer screen shot of text Description automatically generated

# I wrote my project class files, then tested them using unit tests. Unit testing basically checks the actual code itself. It is looking for errors early on. I did not run any system testing since the whole project was not ready for release. I did not have a way to test this as a user. I could look at basic functionality though. I also did not run any integration testing to ensure everything works together.

# My mindset going into this project was overly cautious. I am new to java and coding in general. I redid my first milestone 2 times due to me thinking I could do it better. I was trying to be overly complex in my initial coding. It was just sloppy and all over the place. I simplified it by testing for null and the length requirements in one line. Originally, I had that as two different lines.

# I tried to limit my bias when testing. I was expecting a lot of errors since I am new. I went into the project expecting to find errors, then find a way to address the errors. I could see a more seasoned developer going into testing thinking all of their work is perfect since the requirements were not that complex.

# It is important to remain disciplined when coding and testing. Cutting corners is not a good idea. Yes, it saves time, but what if something is overlooked? That could open up the software owner to all sorts of liability issues. I plan on taking things on slowly with an analytical approach. I will budget my time accordingly. Hopefully I can join a team with the same mindset.

Works Cited

Not applicable/None