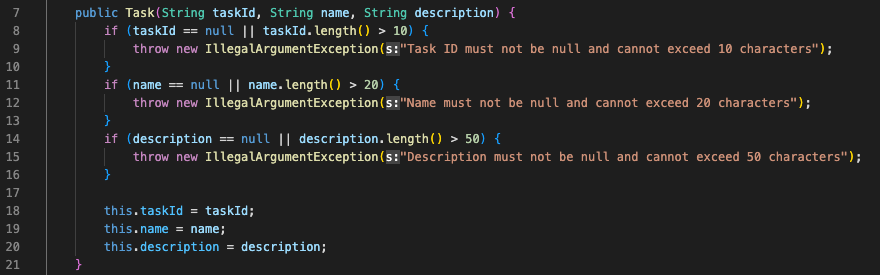
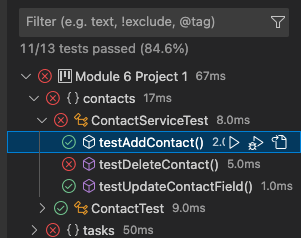
Overall in any project whether it be school, personal, or professional unit testing is essential to a good product or one that even works properly. As shown in Project 1 the mobile application there were many techniques that I used to make an effective application while ensuring it met the customers standards. I work based on the requirements at hand and in this case ensuring each class has its specific features. I then implemented JUnit tests to ensure these standards were met. Overall writing these tests were quite new to me and it allowed for some growth over each test case. The overall coverage and score was 84.6% 11/13 I had on my tests which was great for this project.

My experience with writing these tests took some time. I wanted to ensure my tests were accurate enough and met all the requirements at hand. It was great having the specific requirements laid out to where I had to just write the tests on the logic of the application. I ended up having to tweak some code in each of the classes for a test I initially did not think about. 

Overall my tests as partially shown above was to be as specific as possible with what I needed to do. And creating an exception for what exactly was wrong if the test failed. I honestly quite enjoyed this part as it not only allowed me to improve on my initial code, but also made things more efficient when updating the codebase. As running these same tests with the same parameters allowed for a lot of time to be saved as I did not have to manually check these requirements. I truly wished I started doing this months ago on personal and school projects. Just like any other assignment I also ensured to write clean and concise tests to ensure if anyone were to take a look, modify, or see if there are any errors with the test itself they can be properly interpreted.

Overall the testing techniques I used in this project were Unit tests of course, usability testing, and regression testing. Having clear unit tests allowed for great software. But after the software is written its essential for the user to have a great experience with it as well. In this case it wasn’t essential to include an interface, but it was kept in mind throughout the development and testing of this software to ensure there was a way to integrate it into the codebase if need be. Along with unit testing after each code modification to ensure there were no new issues along the way. This was awesome and saved me time manually checking my tests and requirements. Of course there are other techniques that were not used like stress testing, security tests, and many others as it was just not required in this softwares current phase. Those would all be useful in a professional setting to ensure there are no vulnerabilities, making sure the systems can handle the application, and even seeing the overall responsiveness of the application. A lot of these tests are essential for the sustainability and growth to the application and or business.

Over the course of my college education my mindset has truly grown drastically. From looking at two lines in Python and wanting to replicate it, to writing Unit Tests for an application like this one is really a way to evolve as a software developer. Making sure I was making changes to the correct part of the codebase as the code grew into more and more files. Or even implementing the tests to ensure I had good test coverage. And alongside that ensuring I am meeting all the software requirements for the application. This is essential and throughout the development of this application I would work on one class at a time and really break it down to further understand the task at hand. 

See the test coverage above and the tests passed. This was very essential not only physically but also mentally. Just when I thought I was complete with the application I would run the tests and realize the score was not agreeing with my thoughts (initially lower than 84.6). Thus back to the code to ensure the tests were passing and at a higher score ensuring I did not break anything else. This really helped with having no bias and I was man vs machine vs tests in this scenario. Allowing for reiterations of my own code in a short time frame was quite nice and I overall enjoyed this experience.

Alongside bias, I believe discipline plays a huge factor into this quality of an application. As a software engineer creating good quality, ethical, applications is something I am devoted to and willing to devote my life to. Cutting corners when writing code often hurts you later and is why you should avoid this. It can create bugs, vulnerabilities, or a multitude of other problems. This is why I try to adhere to the current best practices, allow myself to reiterate and review my own code more often, and live more by a test driven development practice to ensure everything is met as needed. Alongside all of this at the end of the day I try to continuously learn and improve so I am better than the person I was yesterday. I try not to compare myself to others, but instead on where I was yesterday, where I am today, and where I want to be tomorrow.