Using Frameworks for

Master of Science in Information Technology

Software Design and Programming

James McKenna

University of Denver College of Professional Studies

11/07/2025

Faculty: Nirav Shah, M.S.

Director: Cathie Wilson, M.S.

Dean: Michael J. McGuire, MLS

Abstract

This portion of our class assignment is to continue making use of our Parking System class that will enhance the functionality and design that we've built upon in previous assignments. This time the focus was to use different methods such as equals or hashCode that utilized any List from all our classes. This design decision not only allowed decoupling in our classes it also forced more getters and redeveloping of our classes. This allowed safer practices such as encapsulation, coupling, and flexibility. This allows our code to have more flexibility, enhanced design, and ease of use when adding even further capabilities to our code.

CONTENTS

What did you find difficult or easy?

With this particular assignment I found myself going back and redeveloping a lot of code. There were a lot of classes where I had to not only add the new equal and hashCode methods, but I also had to include more getters so that there was less coupling between classes. By using more getters I was able to get rid of tighter coupling with classes so that only by calling one part of a getter one class would be able to pull all it needed and was less dependent on the classes themselves. I had to also incorporate the new frameworks for collections such as our equals and hashCode. Also by forcing encapsulation now I was able to make it so that whoever is trying to utilize or incorporate our code has to do it through my getters and can't manipulate our data directly.

What Helped You?

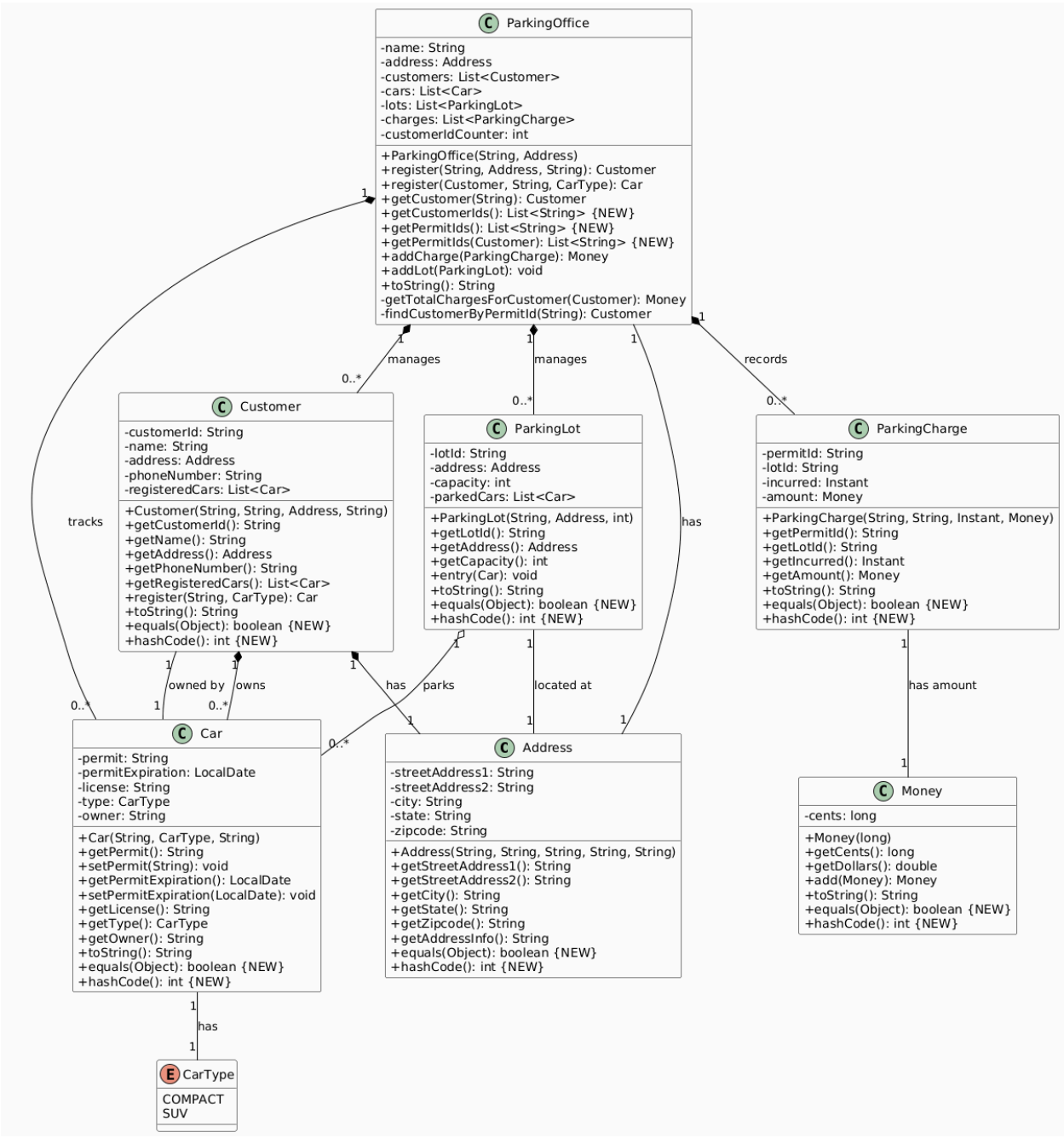
A lot of this assignment was reinforcing previous classes and concepts that we've learned in previous assignments or forcing myself to look it up. I incorporated more encapsulation, more getters, and I required more loose coupling in my code such as the ParkingOffice.java with the getPermitIds method. This method now is only dependent on the getPermitIds, I have tried going through the code with the principle of using the least amount of information as possible from other classes. Also now using the equals method this was a little tricky. I had to look up the importance of using equals in terms of the instantiation for our objects that we're passing in, making sure they're not null, and the we're comparing various fields in between our objects. This took a little time to incorporate as well as test. I also was grateful for previous unit tests as I was able to test further functionality and only really needed

to tweak my unit tests so that we're testing the equals and hashCode methods I added in because of this assignment.

What did you wish you knew before?

Personally I wish I knew more about proper software designs and patterns before this assignment. I had to look a lot of information up specifically about coupling, using equals and hashCode and why this specifically matters. I was aware of dictionaries in Python which are pretty much similar to a Hash Map where we're using this in our hashCode methods. At one point I wasn't quite sure why this was being incorporated but now I understand its in order to reduce look up time and make our code more efficient by explicitly using a hash in order to do look ups. Another key component I was wondering about was why we needed an equals method. I had to look up more information from our class overview lessons and further information online. I realized that we needed an equals method because we needed to prove that the object we were creating and using was the same, and not a different one in a different memory address; even if they were instantiated using the same attributes. Thus the equals method was for saying if I had the same customer with the same ID it's the same customer. My equals method now allows for comparison with customer IDs, it can detect duplicates, can be used with our collections we defined and can remove by value as well.

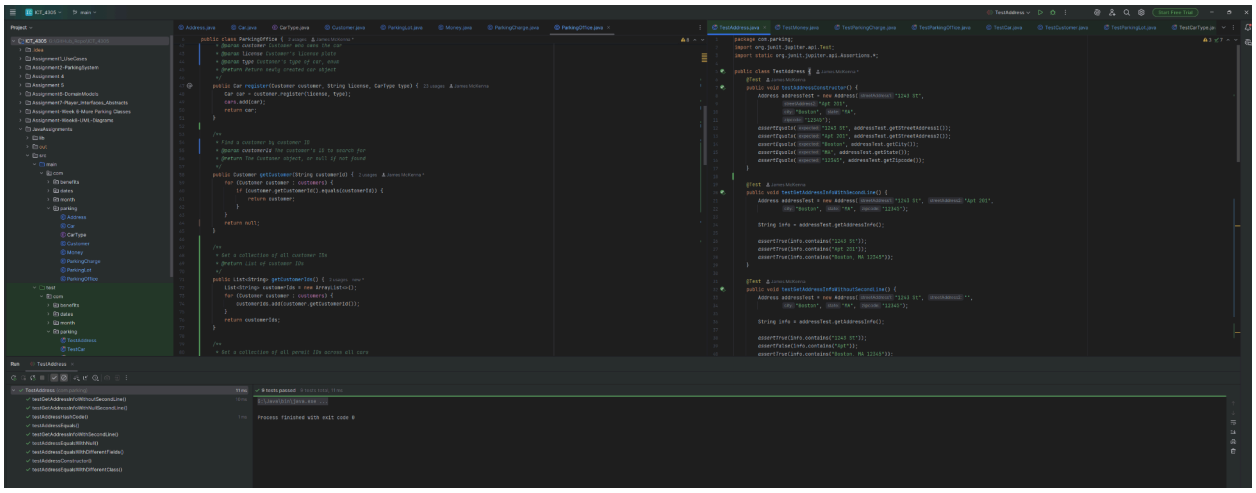
Implementations and Design Decisions



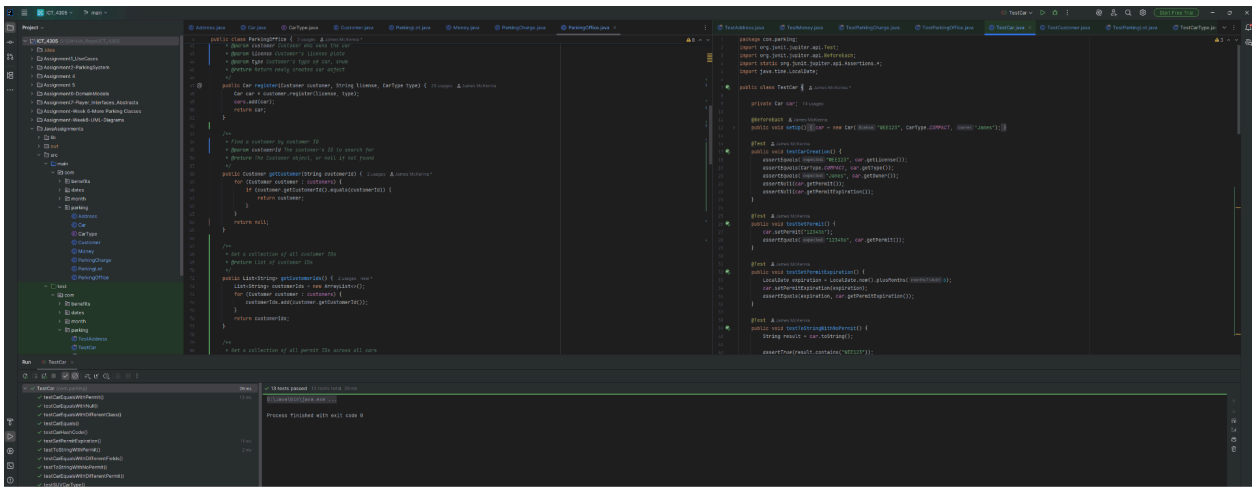
I've briefly talked about it before, but overall the major design and architecture didn't really change between all the classes I've created. I primarily added new getters with our ParkingOffice.java such as the getCusdtomerIds, the getPermitIds, and getPermitIds(Customer). Across all the other classes I primarily added hashCode and equals into all the classes that utilized lists in order to make sure that our classes that are being used in lists can be found and looked up appropriately. I also stopped using a diagram I was making by hand for our UML Class diagram and made my own through PlantUML. It was getting harder and harder to keep all the methods and attributes for all our classes as we kept growing our system for our Parking Office and class.

Successful Unit Test Screenshots

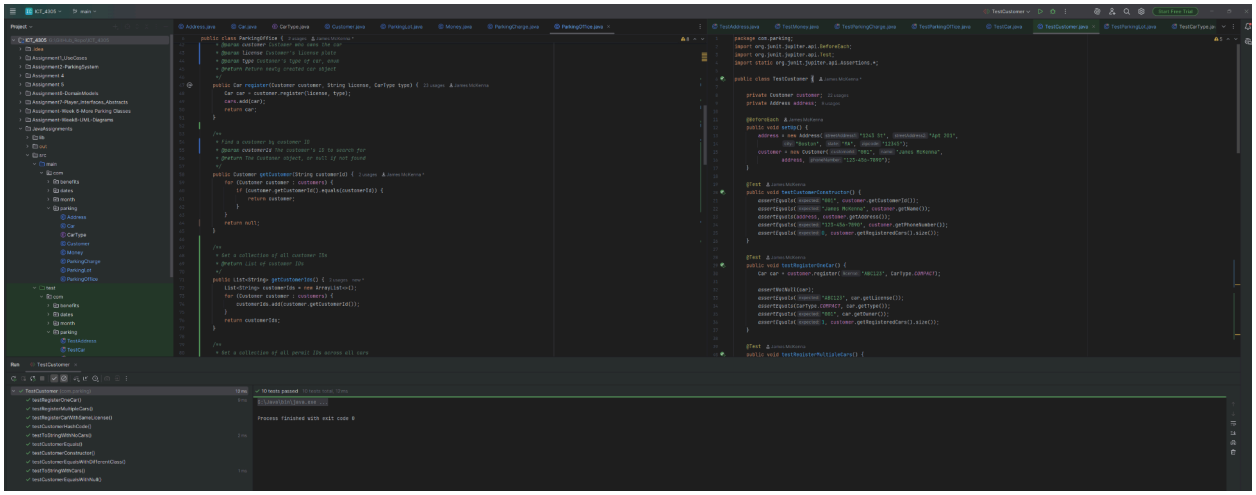
TestAddress.java



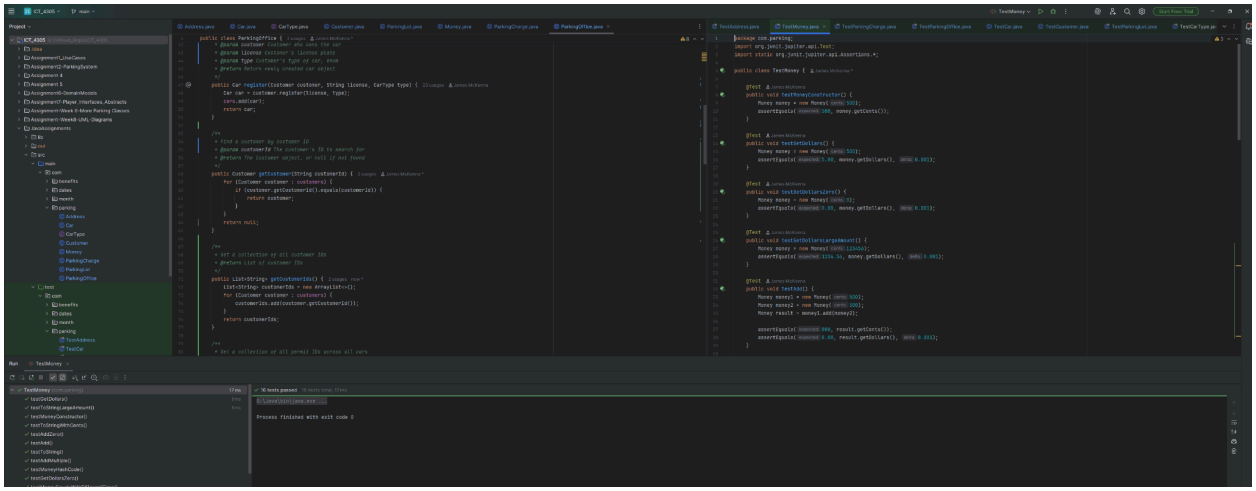
TestCar.java



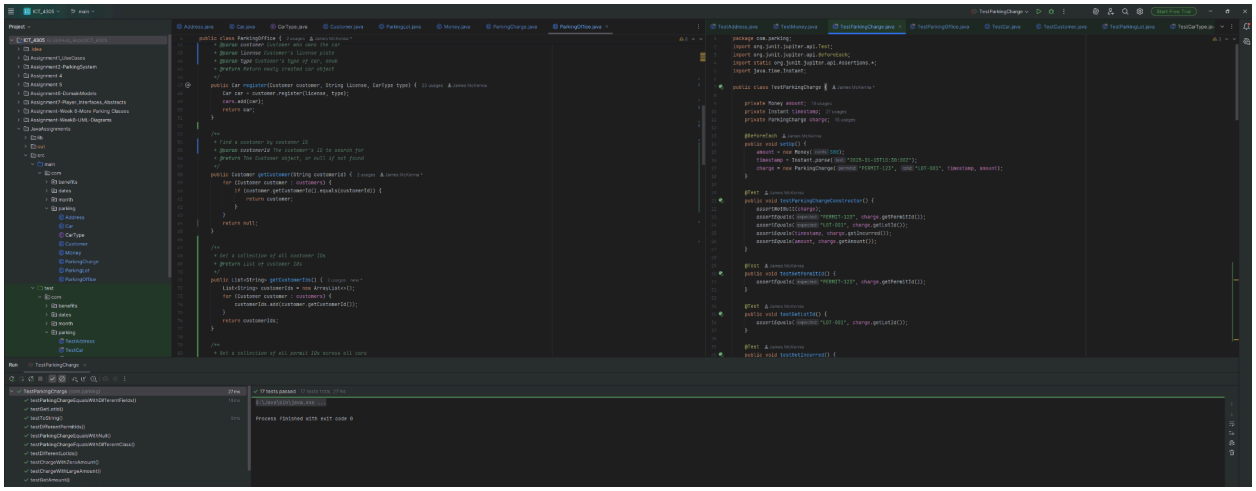
TestCustomer.java



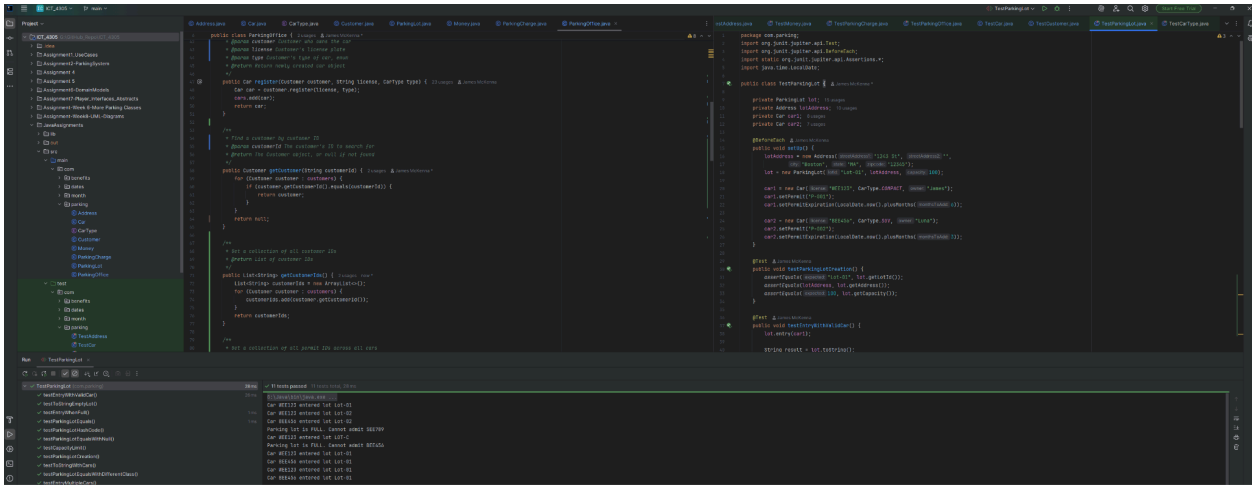
TestMoney.java



TestParkingCharge.java



TestParkingLot.java



TestParkingOffice.java