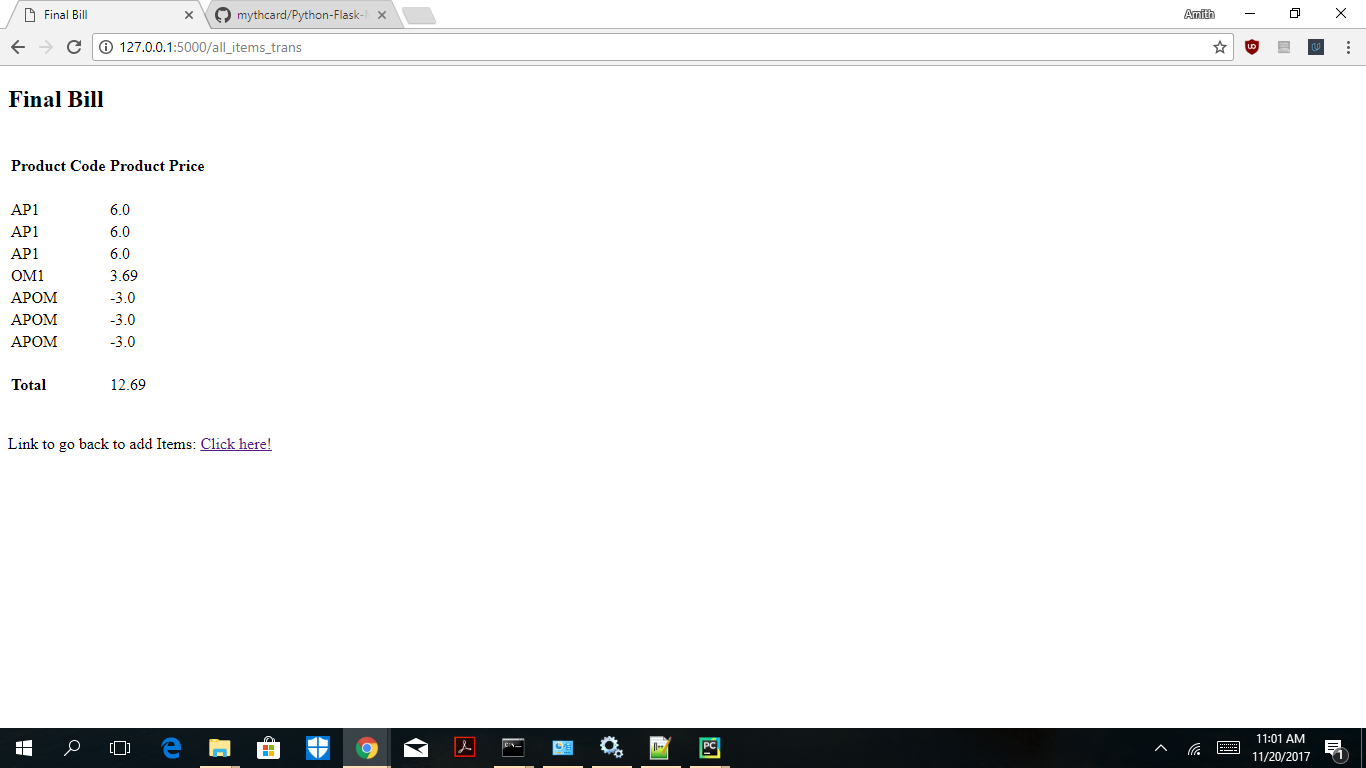
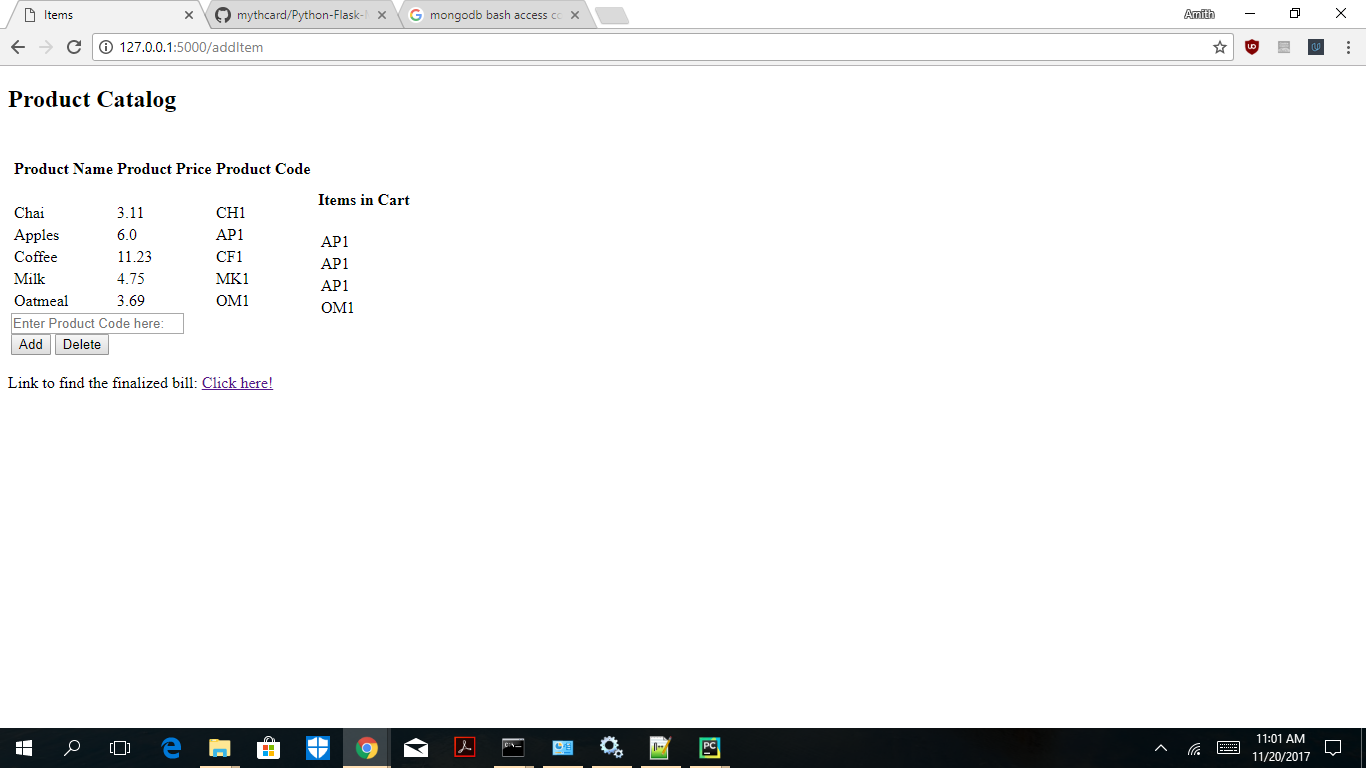
**Check Out System – The Farmers Market**

Below are the snapshots for one test case implementation for the system for Basket:

AP1, AP1, AP1, OM1



I considered 2 design choices

1. Use a SQL database to store product catalog with codes and prices and store a metadata of discounts and a transaction table to drive each transaction. I did not choose this as every interaction in a transaction would involve a client server interaction, and also a lot of overhead on the database
2. The second design consideration is to create an object Transaction in Python which would be triggered when finalizing the bill. This transaction object serves as an API to calculate the bill with discounts. Few properties like buyNgetNFree, checkLimit and addPriceDropFeatureItems are designed to handle future discounts or changes in parameters in current discounts. However a metadata to further automate and to reduce the code change to enhance operations can be performed.

I have used Python, Flask and MongoDB.

I have included a test integration file with test cases that I have used for testing. In addition to this the code contains unit test cases here and there in comments.

Below is the Github link to source codes and Docker Files:

<https://github.com/mythcard/Python-Flask-MongoDB>

It contains the instructions to create the docker containers. I have included a file ‘MongoDBSetUpContainer’ to perform the MongoDB set up. I understand the objects need to be handled manually in this case which can be done in a much better way in a docker environment.