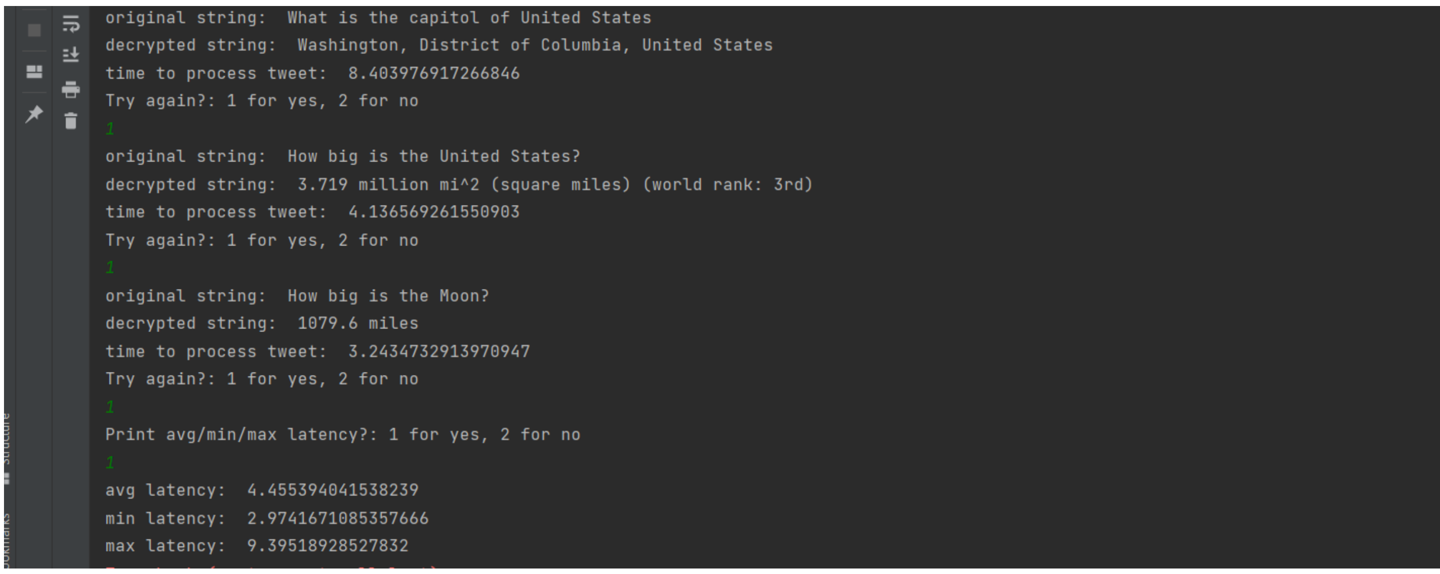
ECE:4564, Project 3 Report

Team 7 members:
Azam Shoaib, shazam23@vt.edu
Parker Chesney, parkerchesney@vt.edu
Jihoon Park, jpark918@vt.edu

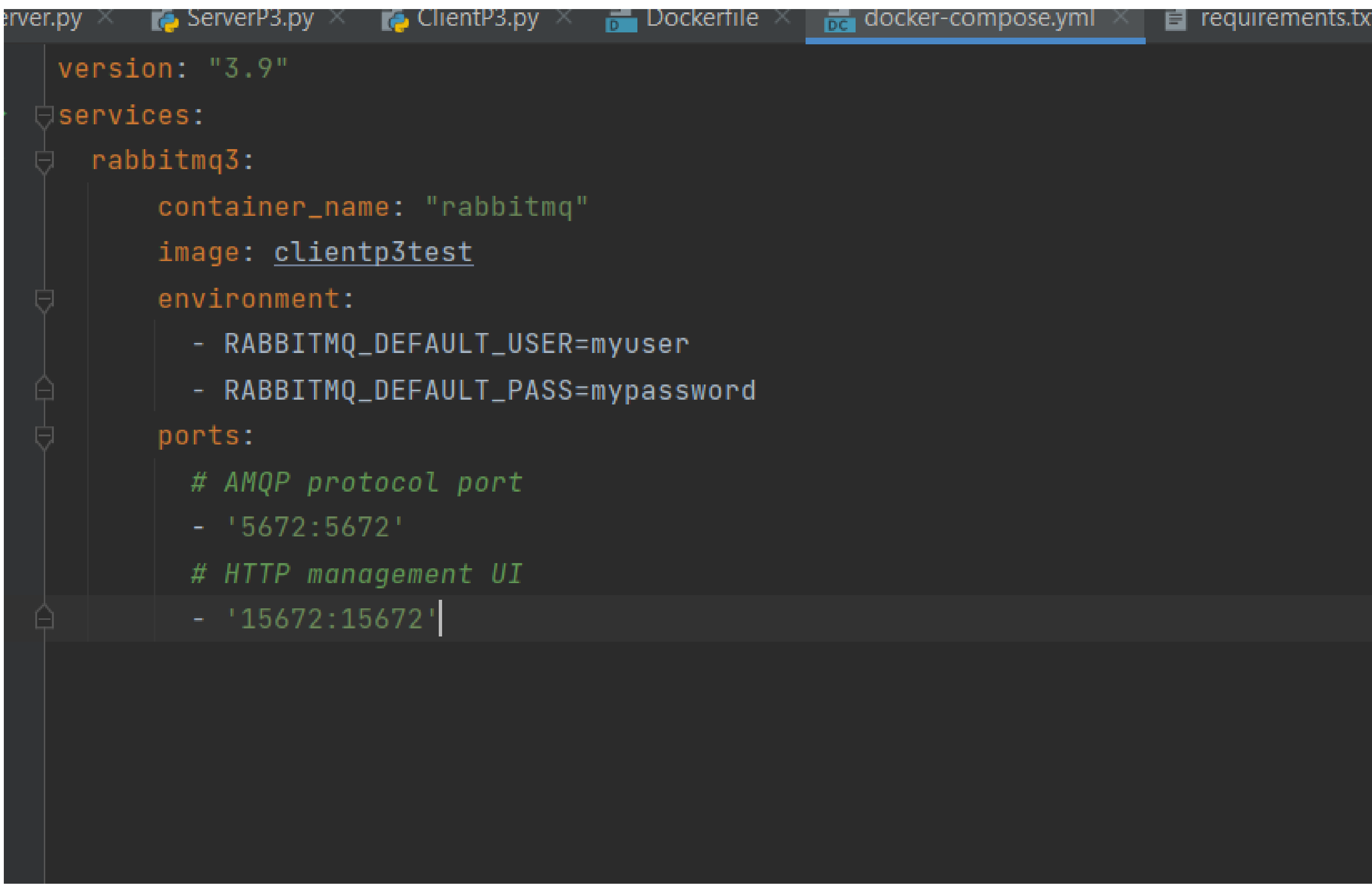
Description:

For this project we had a four file design: Client.py, Server.py, ClientKey.py, and ServerKey.py. The importance of ClientKey.py and ServerKey.py is that it gives the main code robustness. For project 2, Client.py and Server.py came from a TCP client and server assignment we all did in Intro to Computer Networking. This time for project 3, we had to make a non-stopping client program without the use of socket programming. This was achieved through the use of RabbitMQ, which is a message broker based on the AMQP protocol. This allows us to exchange/receive messages from producers and pushes them to queues.Knowing there could be multiple different formats a question can be given to us, we made a for loop that extracts the question portion of the tweet. Once the question is extracted, we send an encrypted version to the server to be answered via RabbitMQ. The server will decrypt the message. The answer will be encrypted and sent over to the client via RabbitMQ. The client will decrypt the message, display the latency process it took, and display it in the console. The user is then prompted if another tweet would like to get answered.
After 20+ messages, the user is prompted if they would like to see the avg/min/max latency. Below is an example of the client process:



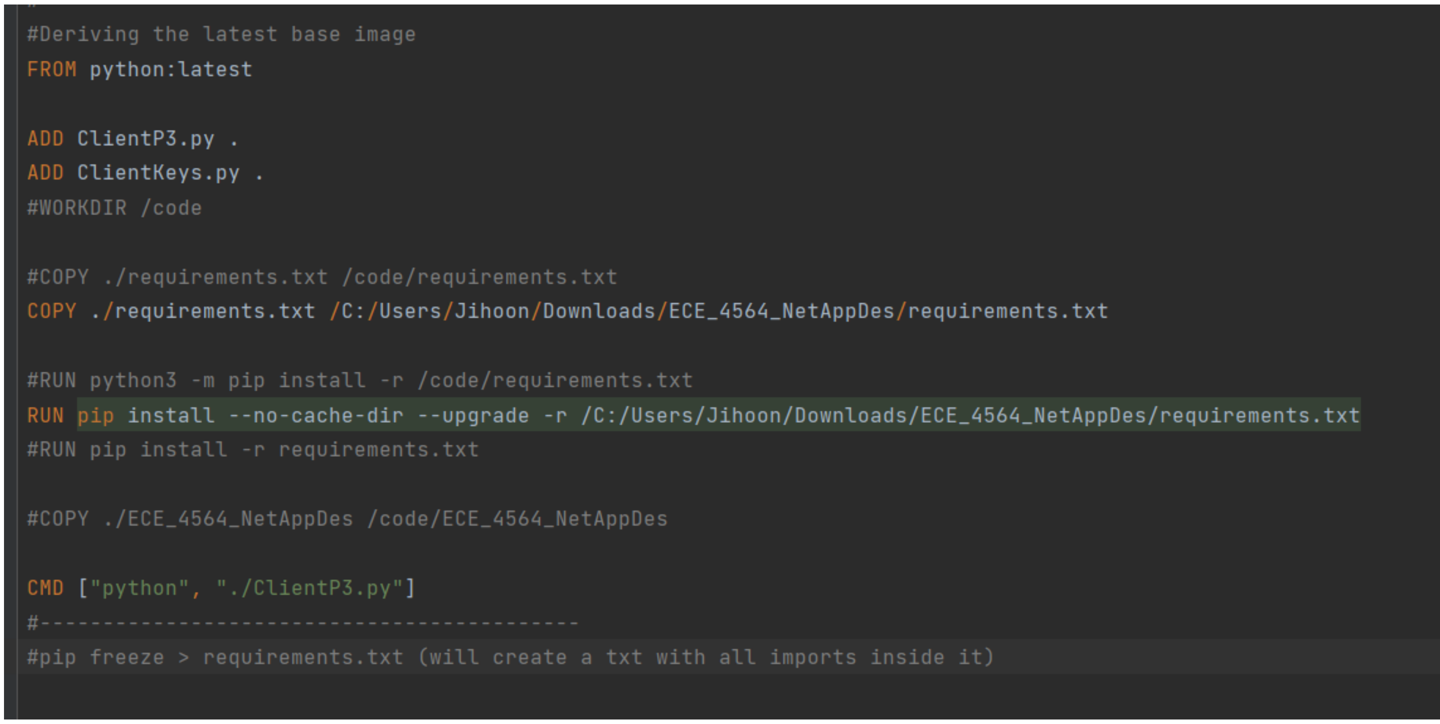
```
original string:  What is the capitol of United States
decrypted string:  Washington, District of Columbia, United States
time to process tweet:  8.403976917266846
Try again?: 1 for yes, 2 for no
1
original string:  How big is the United States?
decrypted string:  3.719 million mi^2 (square miles) (world rank: 3rd)
time to process tweet:  4.136569261550903
Try again?: 1 for yes, 2 for no
1
original string:  How big is the Moon?
decrypted string:  1079.6 miles
time to process tweet:  3.2434732913970947
Try again?: 1 for yes, 2 for no
1
Print avg/min/max latency?: 1 for yes, 2 for no
1
avg latency:  4.455394041538239
min latency:  2.9741671085357666
max latency:  9.39518928527832
```

In addition, the dockerfile allows us to add in python script files as well as copy a requirements.txt file which includes all the libraries/dependencies that the python script file uses. This dockerfile can help us create an image in the docker application. Which can then be run by creating a container in a docker-compose.yml file.



```
version: "3.9"
services:
  rabbitmq3:
      container_name: "rabbitmq"
      image: clientp3test
      environment:
        - RABBITMQ_DEFAULT_USER=myuser
        - RABBITMQ_DEFAULT_PASS=mypassword
      ports:
        # AMQP protocol port
        - '5672:5672'
        # HTTP management UI
        - '15672:15672'
```

Composer file



```
#Deriving the latest base image
FROM python:latest

ADD ClientP3.py .
ADD ClientKeys.py .
#WORKDIR /code

#COPY ./requirements.txt /code/requirements.txt
COPY ./requirements.txt /C:/Users/Jihoon/Downloads/ECE_4564_NetAppDes/requirements.txt

#RUN python3 -m pip install -r /code/requirements.txt
RUN pip install --no-cache-dir --upgrade -r /C:/Users/Jihoon/Downloads/ECE_4564_NetAppDes/requirements.txt
#RUN pip install -r requirements.txt

#COPY ./ECE_4564_NetAppDes /code/ECE_4564_NetAppDes

CMD ["python", "./ClientP3.py"]
#-----------------------------------------
#pip freeze > requirements.txt (will create a txt with all imports inside it)
```

dockerfile

Contributions:
Jihoon Contribution: allow client/server continuously via user input, allow latency checking, create dockerfile/composer file, write report.
Azam Contribution: RabbitMQ implementation in server/client, tweepy library extension, report.
Parker Contribution: RabbitMQ implementation in server/client, tweepy library extension, report.

https://docs.docker.com/config/containers/container-networking/
https://www.rabbitmq.com/download.html
https://forums.docker.com/t/python-error-reported-by-docker-using-pip-to-install-requirements-files/86778
https://developer.twitter.com/en/docs/twitter-api/tweets/timelines/api-reference/get-users-id-tweets