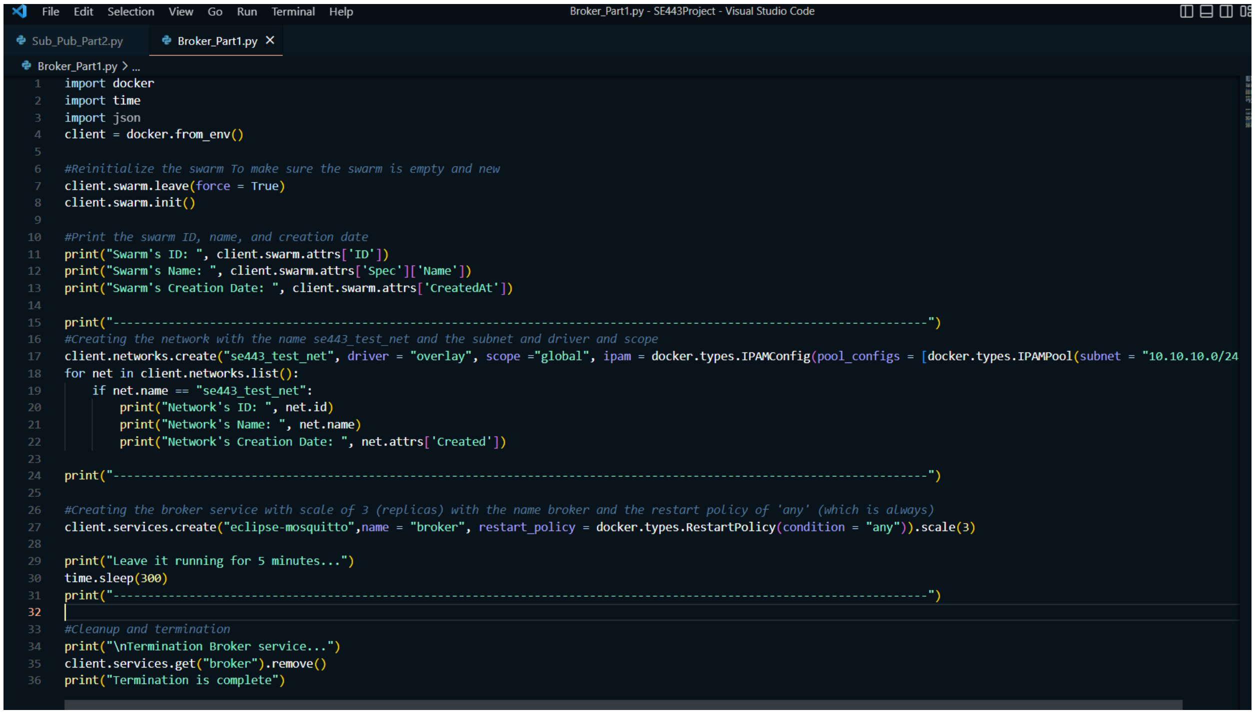
Meteb Almadi

200304

Github link: https://github.com/metebalmadi/SE443Project_200304.git

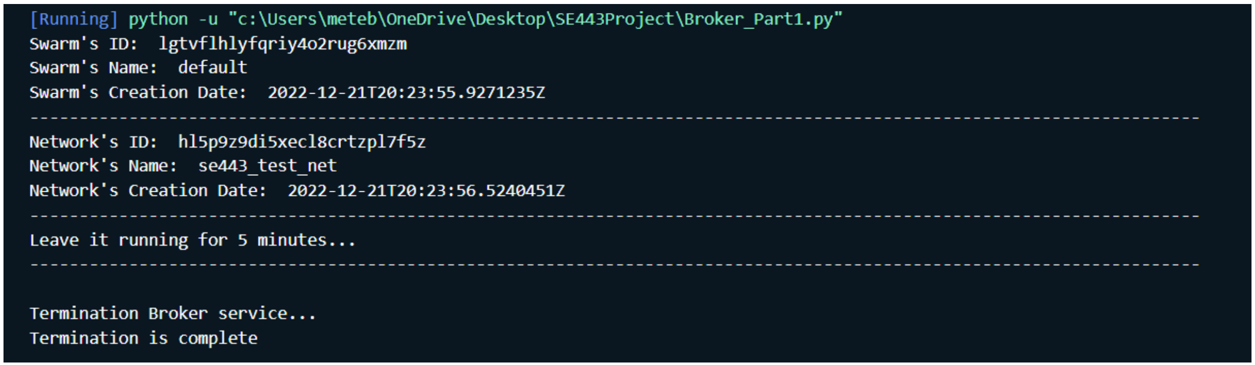
Screenshots:

Part 1 Code and output:

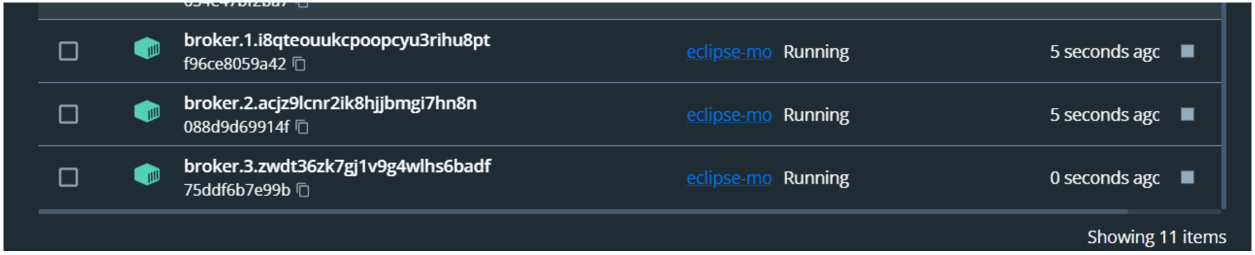


```python
import docker
import time
import json
client = docker.from_env()

#Reinitialize the swarm To make sure the swarm is empty and new
client.swarm.leave(force = True)
client.swarm.init()

#Print the swarm ID, name, and creation date
print("Swarm's ID: ", client.swarm.attrs['ID'])
print("Swarm's Name: ", client.swarm.attrs['Spec']['Name'])
print("Swarm's Creation Date: ", client.swarm.attrs['CreatedAt'])

print("-----------------------------------------------------------------------------------------------------------------------")
#Creating the network with the name se443_test_net and the subnet and driver and scope
client.networks.create("se443_test_net", driver = "overlay", scope ="global", ipam = docker.types.IPAMConfig(pool_configs = [docker.types.IPAMPool(subnet = "10.10.10.0/24
for net in client.networks.list():
    if net.name == "se443_test_net":
        print("Network's ID: ", net.id)
        print("Network's Name: ", net.name)
        print("Network's Creation Date: ", net.attrs['Created'])

print("-----------------------------------------------------------------------------------------------------------------------")

#Creating the broker service with scale of 3 (replicas) with the name broker and the restart policy of 'any' (which is always)
client.services.create("eclipse-mosquitto",name = "broker", restart_policy = docker.types.RestartPolicy(condition = "any")).scale(3)

print("Leave it running for 5 minutes...")
time.sleep(300)
print("-----------------------------------------------------------------------------------------------------------------------")

#Cleanup and termination
print("\nTermination Broker service...")
client.services.get("broker").remove()
print("Termination is complete")
```

Output:

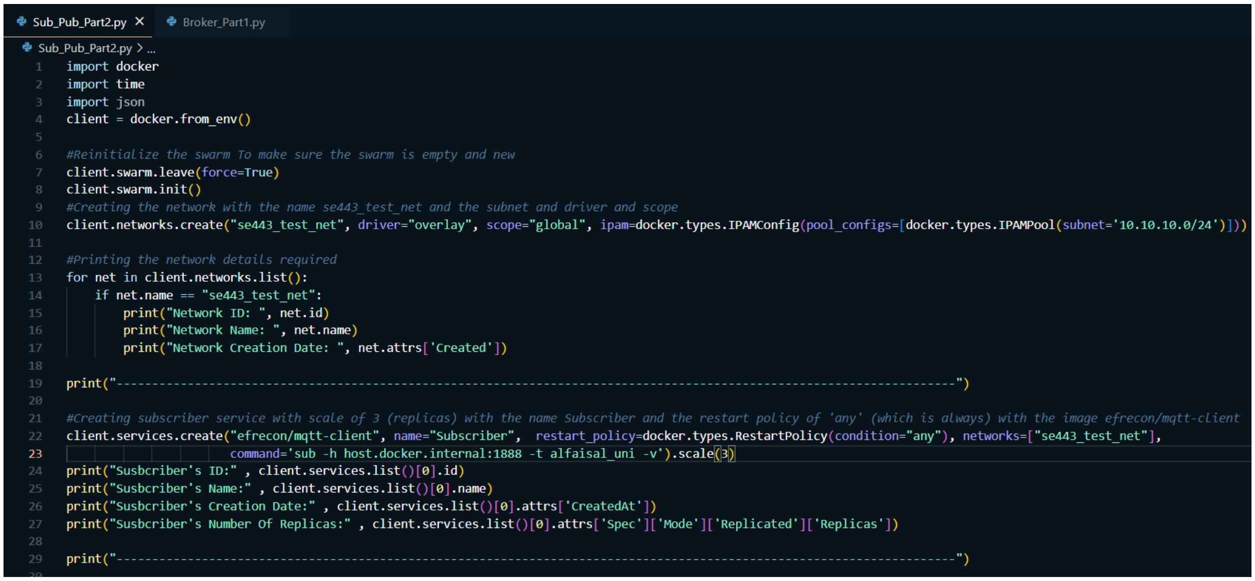


```
[Running] python -u "c:\Users\meteb\OneDrive\Desktop\SE443Project\Broker_Part1.py"
Swarm's ID:  lgtvflhlyfqriy4o2rug6xmzm
Swarm's Name:  default
Swarm's Creation Date:  2022-12-21T20:23:55.9271235Z
-----------------------------------------------------------------------------------------------------------------------
Network's ID:  hl5p9z9di5xecl8crtzpl7f5z
Network's Name:  se443_test_net
Network's Creation Date:  2022-12-21T20:23:56.5240451Z
-----------------------------------------------------------------------------------------------------------------------
Leave it running for 5 minutes...
-----------------------------------------------------------------------------------------------------------------------


Termination Broker service...
Termination is complete
```

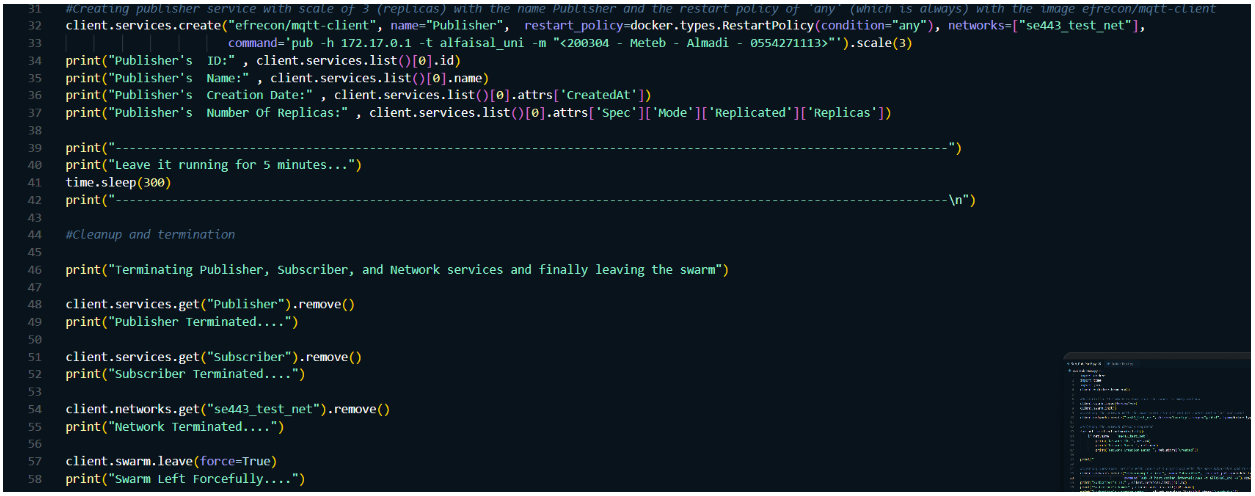


```
□  broker.1.i8qteouukcpoopcyu3rihu8pt    eclipse-mo  Running   5 seconds agc  ■
   f96ce8059a42
□  broker.2.acjz9lcnr2ik8hjjbmgi7hn8n    eclipse-mo  Running   5 seconds agc  ■
   088d9d69914f
□  broker.3.zwdt36zk7gj1v9g4wlhs6badf    eclipse-mo  Running   0 seconds agc  ■
   75ddf6b7e99b
                                                              Showing 11 items
```

Part 2 Code:

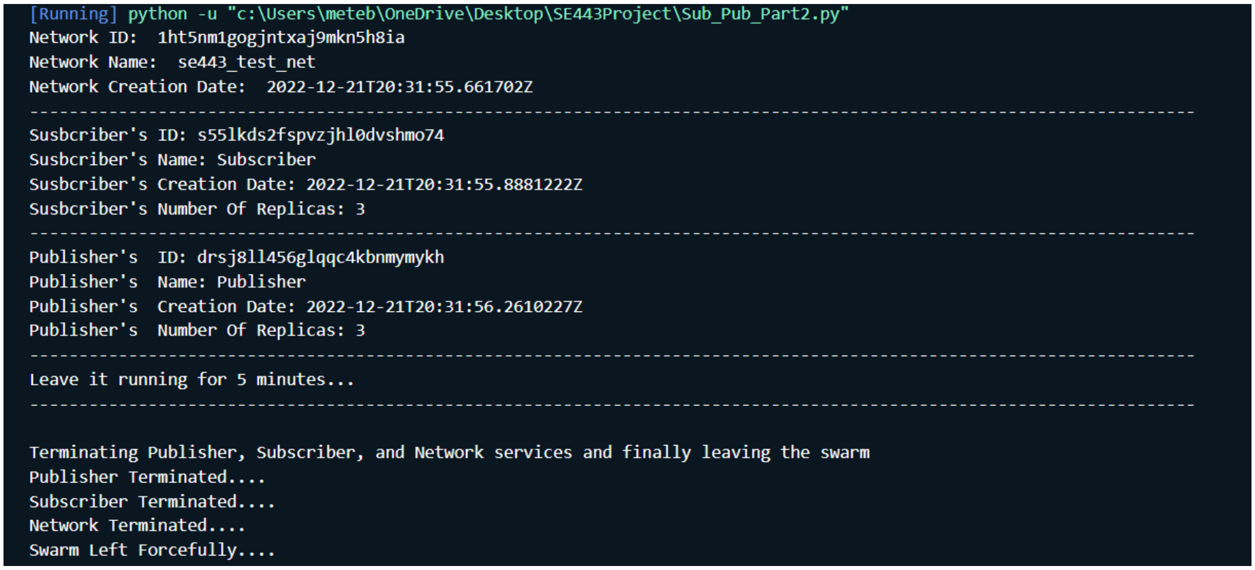


```python
import docker
import time
import json
client = docker.from_env()

#Reinitialize the swarm To make sure the swarm is empty and new
client.swarm.leave(force=True)
client.swarm.init()
#Creating the network with the name se443_test_net and the subnet and driver and scope
client.networks.create("se443_test_net", driver="overlay", scope="global", ipam=docker.types.IPAMConfig(pool_configs=[docker.types.IPAMPool(subnet='10.10.10.0/24')]))

#Printing the network details required
for net in client.networks.list():
    if net.name == "se443_test_net":
        print("Network ID: ", net.id)
        print("Network Name: ", net.name)
        print("Network Creation Date: ", net.attrs['Created'])

print("-----------------------------------------------------------------------------------------------------------------------")

#Creating subscriber service with scale of 3 (replicas) with the name Subscriber and the restart policy of 'any' (which is always) with the image efrecon/mqtt-client
client.services.create("efrecon/mqtt-client", name="Subscriber",  restart_policy=docker.types.RestartPolicy(condition="any"), networks=["se443_test_net"],
                       command='sub -h host.docker.internal:1888 -t alfaisal_uni -v').scale(3)
print("Susbcriber's ID:" , client.services.list()[0].id)
print("Susbcriber's Name:" , client.services.list()[0].name)
print("Susbcriber's Creation Date:" , client.services.list()[0].attrs['CreatedAt'])
print("Susbcriber's Number Of Replicas:" , client.services.list()[0].attrs['Spec']['Mode']['Replicated']['Replicas'])

print("-----------------------------------------------------------------------------------------------------------------------")
```

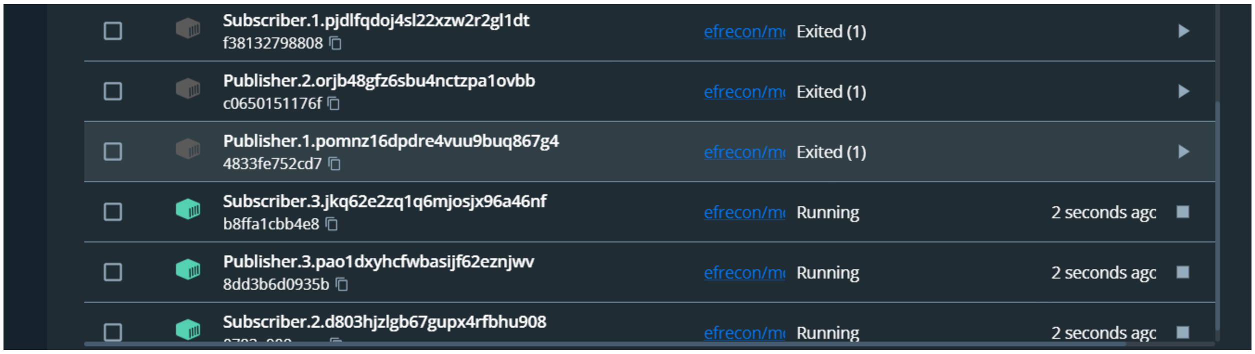


```python
#Creating publisher service with scale of 3 (replicas) with the name Publisher and the restart policy of 'any' (which is always) with the image efrecon/mqtt-client
client.services.create("efrecon/mqtt-client", name="Publisher",  restart_policy=docker.types.RestartPolicy(condition="any"), networks=["se443_test_net"],
                        command='pub -h 172.17.0.1 -t alfaisal_uni -m "<200304 - Meteb - Almadi - 0554271113>"').scale(3)
print("Publisher's  ID:" , client.services.list()[0].id)
print("Publisher's  Name:" , client.services.list()[0].name)
print("Publisher's  Creation Date:" , client.services.list()[0].attrs['CreatedAt'])
print("Publisher's  Number Of Replicas:" , client.services.list()[0].attrs['Spec']['Mode']['Replicated']['Replicas'])

print("----------------------------------------------------------------------------------------------------------")
print("Leave it running for 5 minutes...")
time.sleep(300)
print("----------------------------------------------------------------------------------------------------------\n")

#Cleanup and termination

print("Terminating Publisher, Subscriber, and Network services and finally leaving the swarm")

client.services.get("Publisher").remove()
print("Publisher Terminated....")

client.services.get("Subscriber").remove()
print("Subscriber Terminated....")

client.networks.get("se443_test_net").remove()
print("Network Terminated....")

client.swarm.leave(force=True)
print("Swarm Left Forcefully....")
```

Output:



```
[Running] python -u "c:\Users\meteb\OneDrive\Desktop\SE443Project\Sub_Pub_Part2.py"
Network ID:  1ht5nm1gogjntxaj9mkn5h8ia
Network Name:  se443_test_net
Network Creation Date:  2022-12-21T20:31:55.661702Z
-----------------------------------------------------------------------------------------------------------
Susbcriber's ID: s55lkds2fspvzjhl0dvshmo74
Susbcriber's Name: Subscriber
Susbcriber's Creation Date: 2022-12-21T20:31:55.8881222Z
Susbcriber's Number Of Replicas: 3
-----------------------------------------------------------------------------------------------------------
Publisher's  ID: drsj8ll456glqqc4kbnmymykh
Publisher's  Name: Publisher
Publisher's  Creation Date: 2022-12-21T20:31:56.2610227Z
Publisher's  Number Of Replicas: 3
-----------------------------------------------------------------------------------------------------------
Leave it running for 5 minutes...
-----------------------------------------------------------------------------------------------------------

Terminating Publisher, Subscriber, and Network services and finally leaving the swarm
Publisher Terminated....
Subscriber Terminated....
Network Terminated....
Swarm Left Forcefully....
```



□	Subscriber.1.pjdlfqdoj4sl22xzw2r2gl1dt f38132798808	efrecon/m	Exited (1)		▶
□	Publisher.2.orjb48gfz6sbu4nctzpa1ovbb c0650151176f	efrecon/m	Exited (1)		▶
□	Publisher.1.pomnz16dpdre4vuu9buq867g4 4833fe752cd7	efrecon/m	Exited (1)		▶
□	Subscriber.3.jkq62e2zq1q6mjosjx96a46nf b8ffa1cbb4e8	efrecon/m	Running	2 seconds agc	■
□	Publisher.3.pao1dxyhcfwbasijf62eznjwv 8dd3b6d0935b	efrecon/m	Running	2 seconds agc	■
□	Subscriber.2.d803hjzlgb67gupx4rfbhu908	efrecon/m	Running	2 seconds agc	■