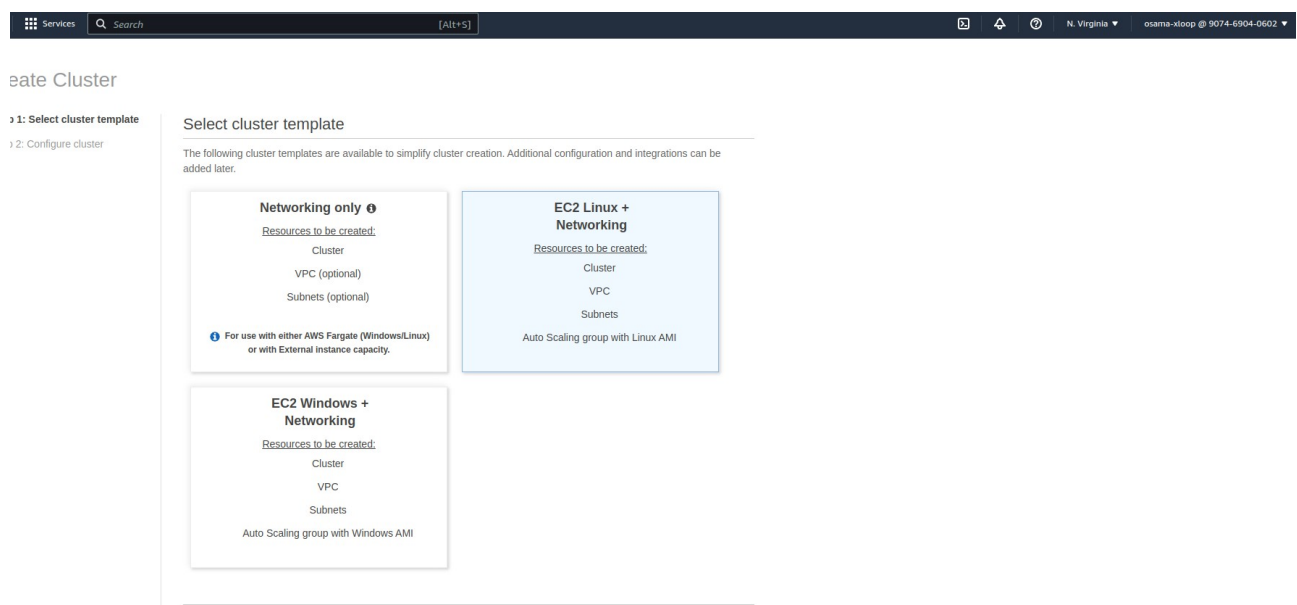


Name: Osama Abdul Razzak(2303.KHI.DEG.029)
Peer Name: Rahima Siddiqui(2303.KHI.DEG.030)
Peer Name: M Humza Moeen(2303.KHI.DEG.019)

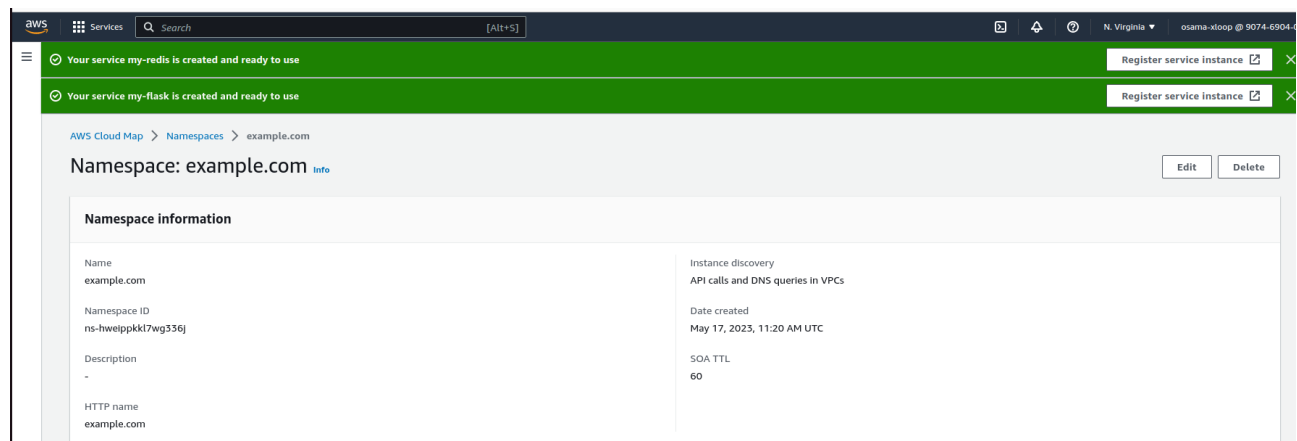
Assignment 4.5

Based on the solution from day 1 (/tasks/5_microservices_development/day_1_microservices/integrating_flask_redis/) add Redis as another ECS service and connect it with existing application. Incorporate results from function `get_and_increase_hit_count()` into the application and show the results on the main page.

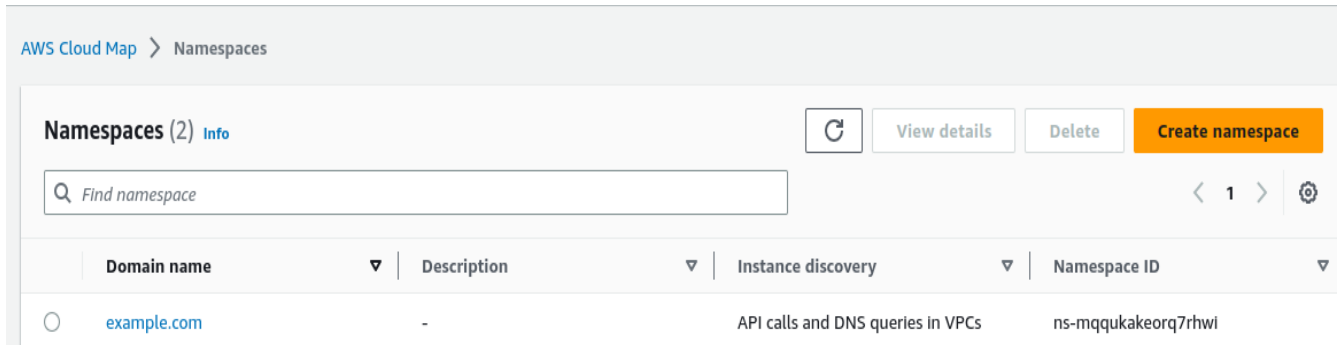
First, we create ECS cluster name 'flaskapp', and select following cluster template



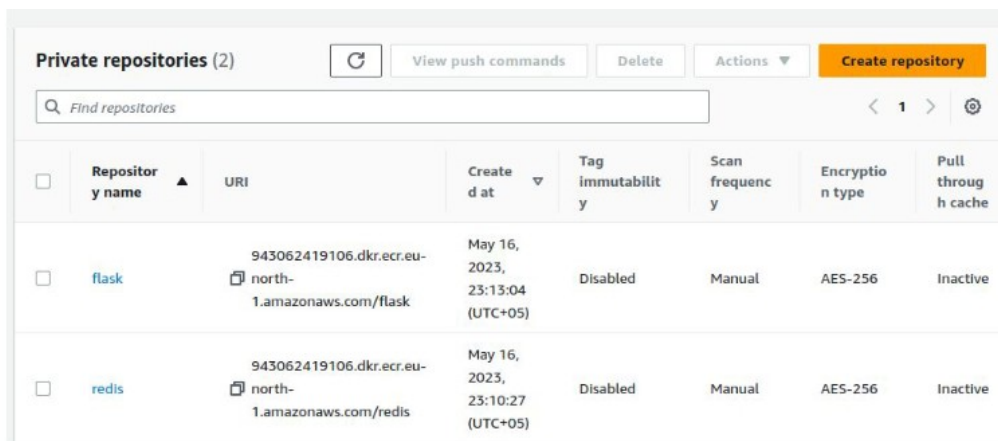
Then, In AWS cloud map we create Namespace: example.com



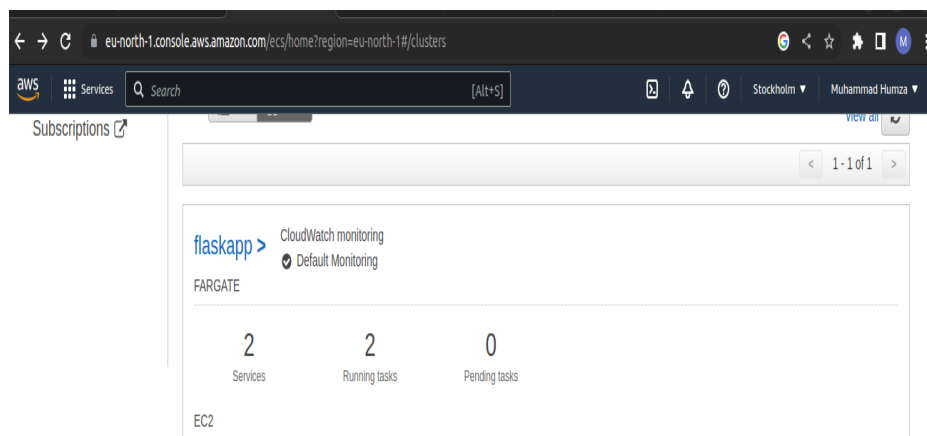
Name: Osama Abdul Razzak(2303.KHI.DEG.029)
Peer Name: Rahima Siddiqui(2303.KHI.DEG.030)
Peer Name: M Humza Moeen(2303.KHI.DEG.019)



In ECR we create two repositories flask and redis



And here is our cluster with two services



And in that repo, we build the both and upload it in our repo

Name: Osama Abdul Razzak(2303.KHI.DEG.029)
Peer Name: Rahima Siddiqui(2303.KHI.DEG.030)
Peer Name: M Humza Moeen(2303.KHI.DEG.019)

Amazon ECR > Repositories > flask

flask

[View push commands](#) [Edit](#)

Images (1) [Refresh](#) [Delete](#) [Details](#) [Scan](#)

< 1 > [Settings](#)

<input type="checkbox"/>	Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest	Scan status	Vulneral
<input type="checkbox"/>	latest	Image	May 16, 2023, 23:26:20 (UTC+05)	79.94	Copy URI	sha256:8947f1e...	-	-

Amazon ECR > Repositories > redis

redis

[View push commands](#) [Edit](#)

Images (1) [Refresh](#) [Delete](#) [Details](#) [Scan](#)

< 1 > [Settings](#)

<input type="checkbox"/>	Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest	Scan status	Vulneral
<input type="checkbox"/>	latest	Image	May 16, 2023, 23:19:21 (UTC+05)	42.47	Copy URI	sha256:9ca9747...	-	-

For both flask app and frontend, do the following:

From the ECS Console, navigate to Task Definitions and create new Fargate task definition

Name: Osama Abdul Razzak(2303.KHI.DEG.029)
Peer Name: Rahima Siddiqui(2303.KHI.DEG.030)
Peer Name: M Humza Moeen(2303.KHI.DEG.019)

Task Definitions

Task definitions specify the container information for your application, such as how many containers are part of your task, what resources they will use, how they are linked together, and which host ports they will use. [Learn more](#)

[Create new Task Definition](#) [Create new revision](#) [Actions](#) Last updated on May 17, 2023 1:11:13 AM (0m ago) [Refresh](#) [Help](#)

Status: [ACTIVE](#) [INACTIVE](#)

< 1-2 > Page size 50

<input type="checkbox"/>	Task Definition	Latest revision status
<input type="checkbox"/>	flask_task	ACTIVE
<input type="checkbox"/>	redis_task	ACTIVE

Then open cluster 'flaskapp' and from Services choose your flask service

Navigate to Tasks and choose existing task

Find the Elastic Network (ENI ID) attached to the task, and click it

From the Network Interfaces list, choose the correct one, and open it

Find Public IPv4 DNS for the ENI, copy it and paste it in the browser address bar, adding the port number 5000 in the end

e.g. ec2-13-53-84-145.eu-north-1.compute.amazonaws.com:5000



Here is aap.py and Docker file SS,

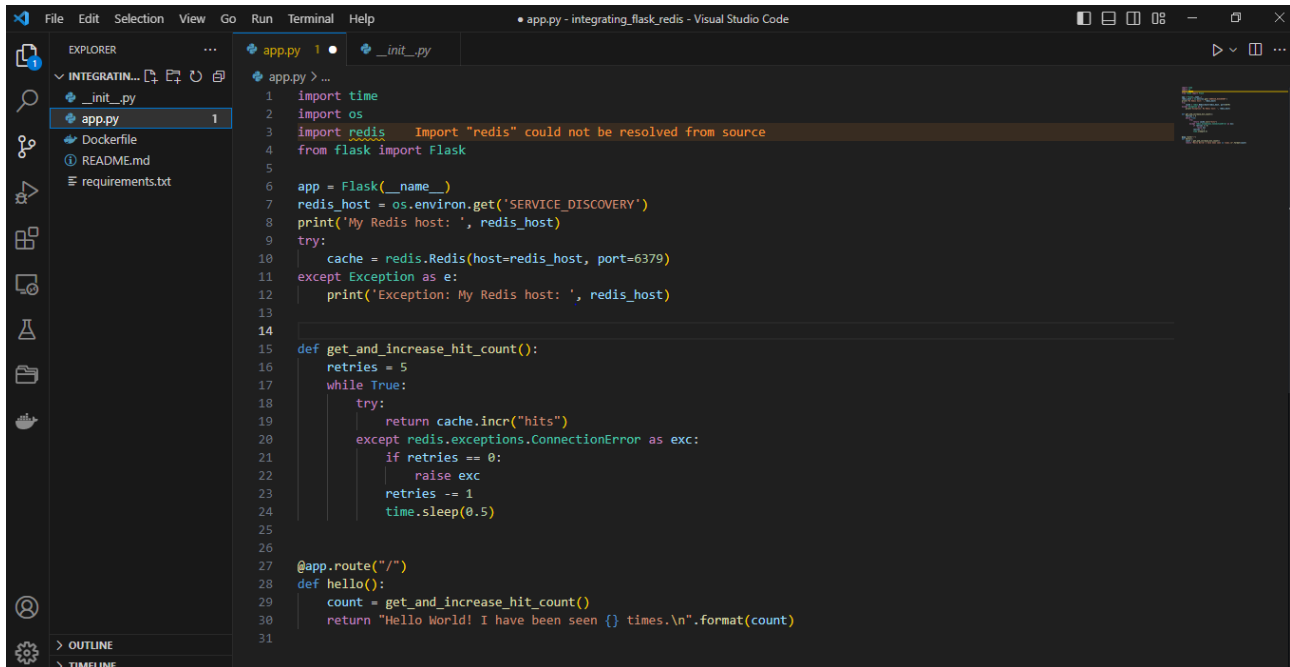
For flask_app service:

Edit newly created security group: add inbound rule of type Custom TCP with port range 5000

For redis service:

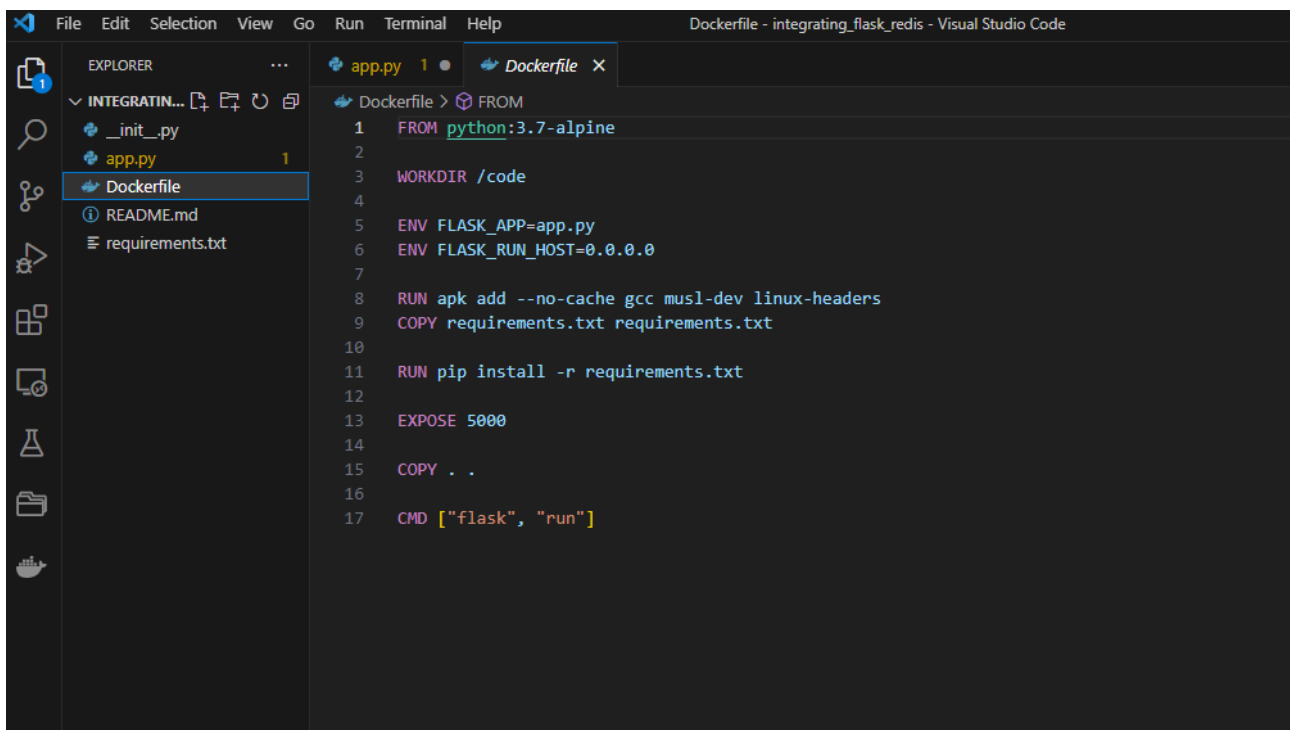
Edit newly created security group: add inbound rule of type Custom TCP with port range 6379

Name: Osama Abdul Razzak(2303.KHI.DEG.029)
Peer Name: Rahima Siddiqui(2303.KHI.DEG.030)
Peer Name: M Humza Moeen(2303.KHI.DEG.019)



The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left containing files: `_init_.py`, `app.py`, `Dockerfile`, `README.md`, and `requirements.txt`. The main editor displays the `app.py` file. A red error message is visible on line 3: `Import "redis" could not be resolved from source`. The code in `app.py` is as follows:

```
1 import time
2 import os
3 import redis
4 from flask import Flask
5
6 app = Flask(__name__)
7 redis_host = os.environ.get('SERVICE_DISCOVERY')
8 print('My Redis host: ', redis_host)
9 try:
10     cache = redis.Redis(host=redis_host, port=6379)
11 except Exception as e:
12     print('Exception: My Redis host: ', redis_host)
13
14
15 def get_and_increas_hit_count():
16     retries = 5
17     while True:
18         try:
19             return cache.incr("hits")
20         except redis.exceptions.ConnectionError as exc:
21             if retries == 0:
22                 raise exc
23             retries -= 1
24             time.sleep(0.5)
25
26
27 @app.route("/")
28 def hello():
29     count = get_and_increas_hit_count()
30     return "Hello World! I have been seen {} times.\n".format(count)
31
```



The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left containing files: `_init_.py`, `app.py`, `Dockerfile`, `README.md`, and `requirements.txt`. The main editor displays the `Dockerfile` file. The content of the `Dockerfile` is as follows:

```
1 FROM python:3.7-alpine
2
3 WORKDIR /code
4
5 ENV FLASK_APP=app.py
6 ENV FLASK_RUN_HOST=0.0.0
7
8 RUN apk add --no-cache gcc musl-dev linux-headers
9 COPY requirements.txt requirements.txt
10
11 RUN pip install -r requirements.txt
12
13 EXPOSE 5000
14
15 COPY . .
16
17 CMD ["flask", "run"]
```