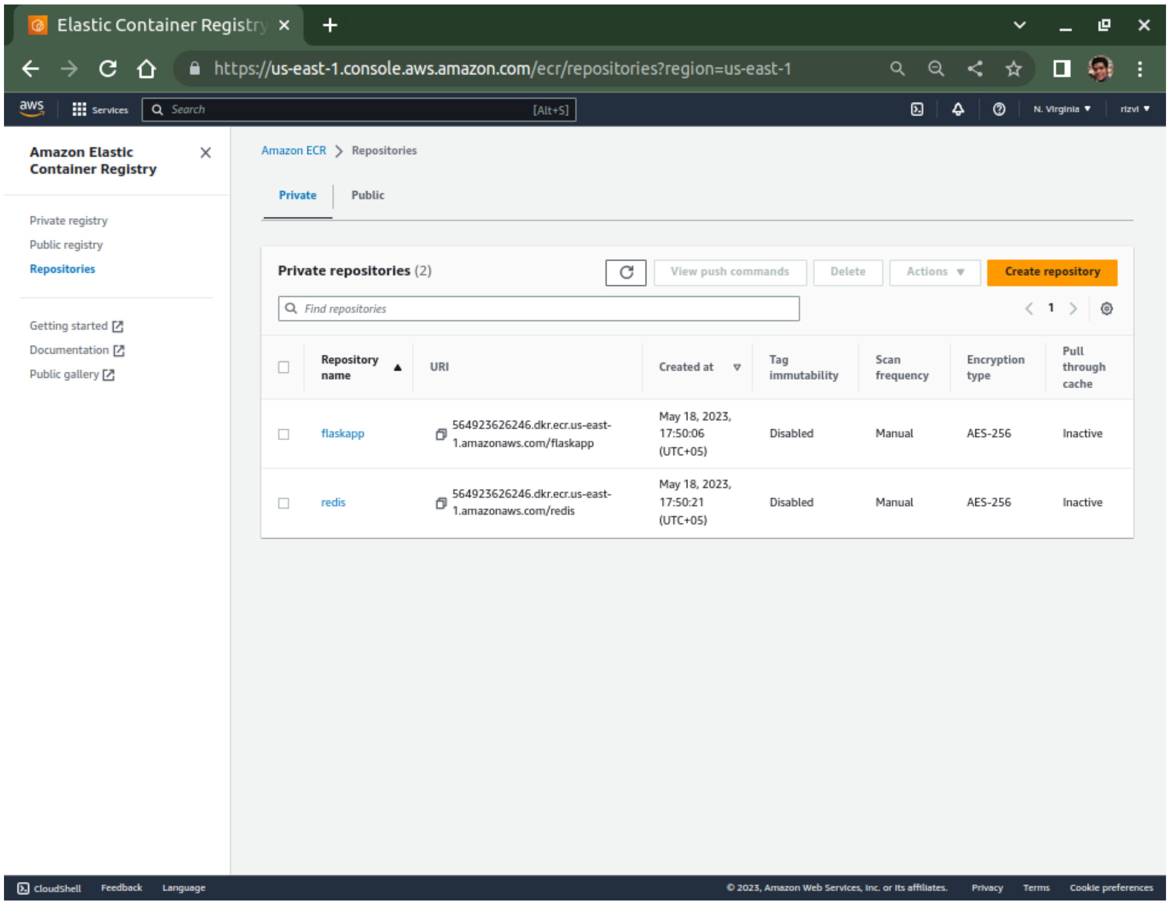
Members:

1. Syed Mohammad Anjil Hussain Rizvi (2303.KHI.DEG.031)
2. Huzefa Anver (2303.KHI.DEG.002)

STEP 1: Uploading docker images

We start off by creating two repositories in an Elastic Container Registry.

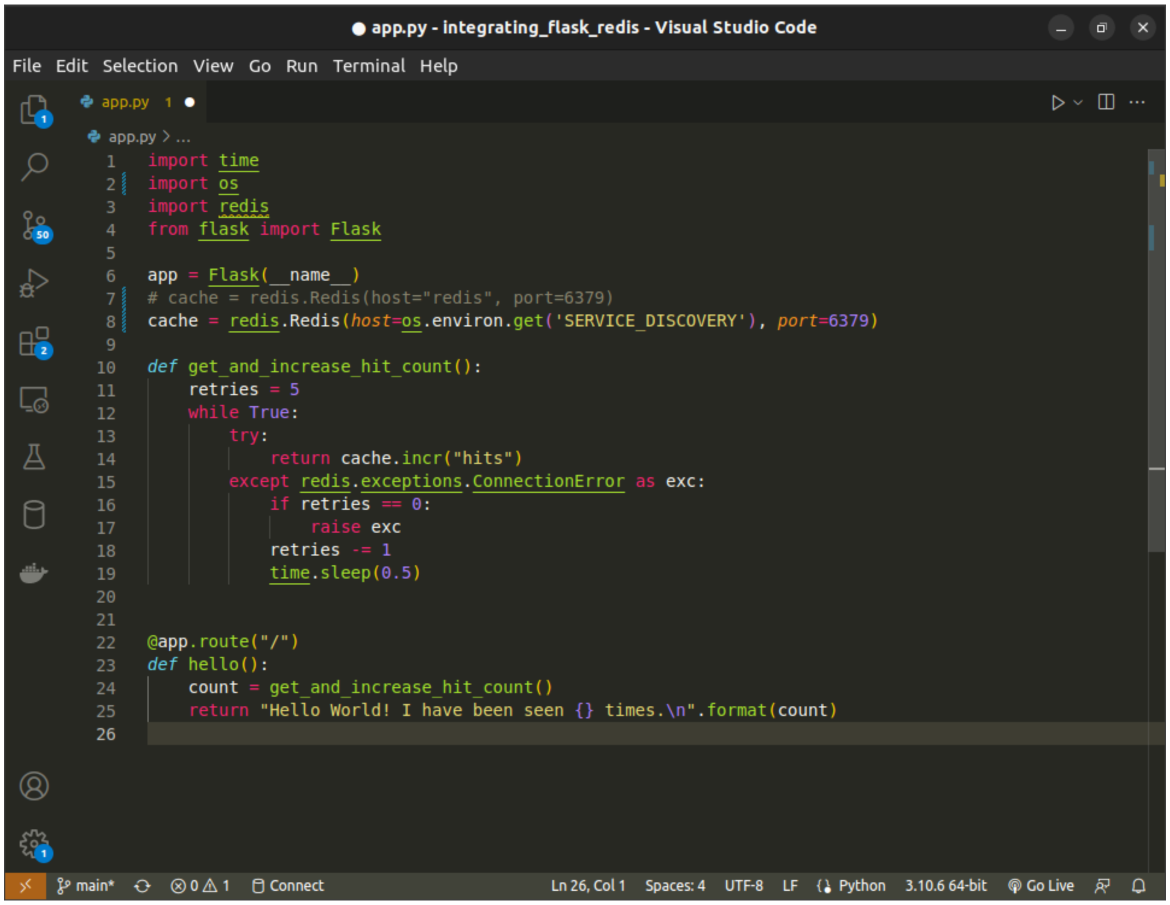


Private repositories (2)

Repository name	URI	Created at	Tag immutability	Scan frequency	Encryption type	Pull through cache
flaskapp	564923626246.dkr.ecr.us-east-1.amazonaws.com/flaskapp	May 18, 2023, 17:50:06 (UTC+05)	Disabled	Manual	AES-256	Inactive
redis	564923626246.dkr.ecr.us-east-1.amazonaws.com/redis	May 18, 2023, 17:50:21 (UTC+05)	Disabled	Manual	AES-256	Inactive

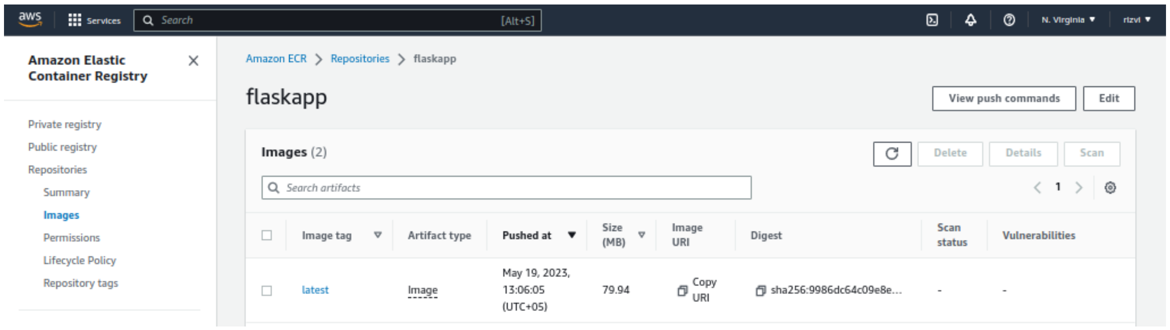
Inside the repository named "flaskapp" is the image of our flask app. Before building and pushing to the repository however, a small change in the app.py file had to made; which is shown below:

1. Line 2 was added to import os
2. Line 8 was added to allow reading redis host through an environment variable named "SERVICE_DISCOVERY".


● app.py - integrating_flask_redis - Visual Studio Code
File Edit Selection View Go Run Terminal Help
app.py 1 ●
app.py > ...
main*
0 ⚠ 1
Connect
Ln 26, Col 1 Spaces: 4 UTF-8 LF Python 3.10.6 64-bit Go Live

```python
import time
import os
import redis
from flask import Flask

app = Flask(__name__)
# cache = redis.Redis(host="redis", port=6379)
cache = redis.Redis(host=os.environ.get('SERVICE_DISCOVERY'), port=6379)

def get_and_increase_hit_count():
    retries = 5
    while True:
        try:
            return cache.incr("hits")
        except redis.exceptions.ConnectionError as exc:
            if retries == 0:
                raise exc
            retries -= 1
            time.sleep(0.5)


@app.route("/")
def hello():
    count = get_and_increase_hit_count()
    return "Hello World! I have been seen {} times.\n".format(count)

```

Flask image:

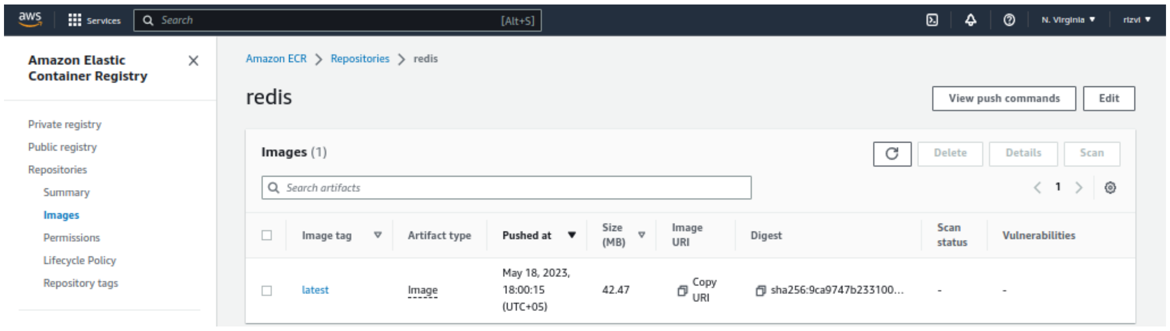


flaskapp

Images (2)

	Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest	Scan status	Vulnerabilities
☐	latest	Image	May 19, 2023, 13:06:05 (UTC+05)	79.94	Copy URI	sha256:9986dc64c09e8e...	-	-

Redis image:



redis

Images (1)

Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest	Scan status	Vulnerabilities
latest	Image	May 18, 2023, 18:00:15 (UTC+05)	42.47	Copy URI	sha256:9ca9747b233100...	-	-

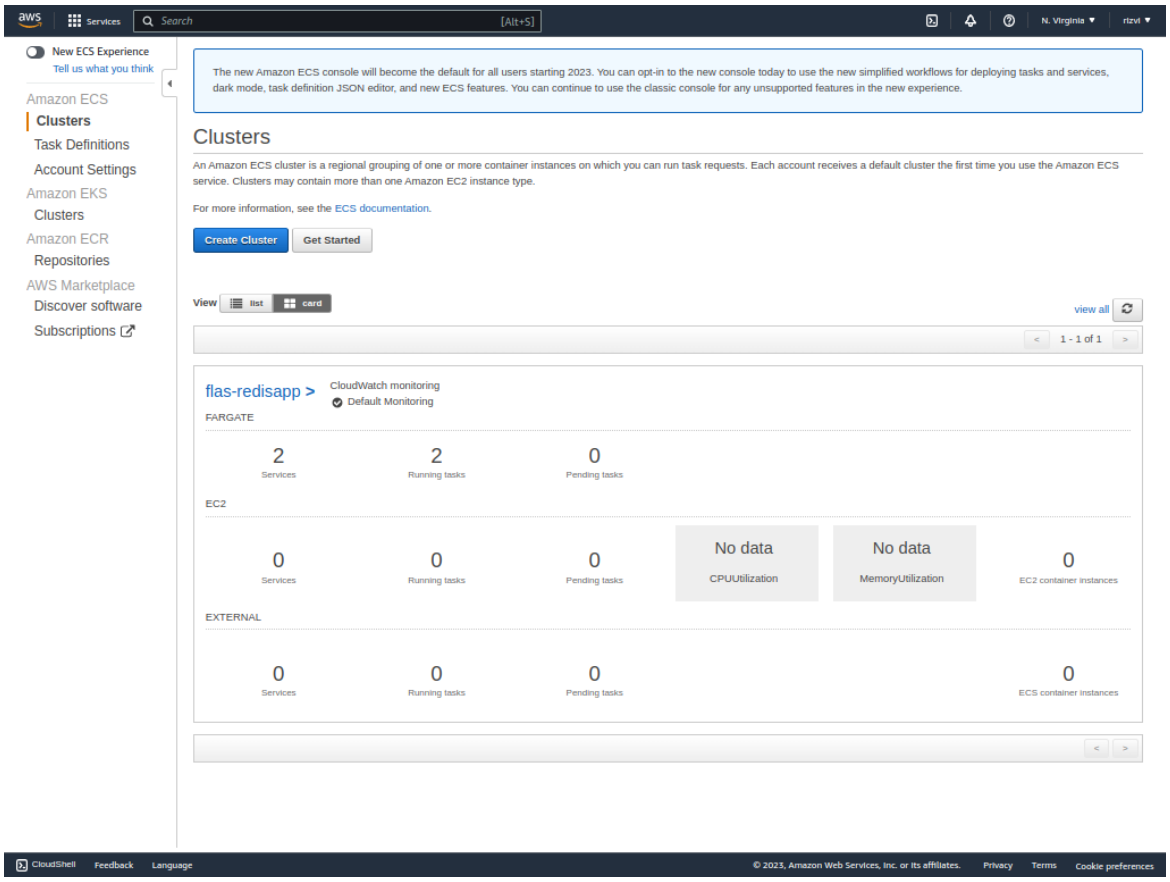
STEP 2: creating an ECS cluster

1. Cluster Template: EC2 Linux + Networking
2. CLuster name: flas-redisapp
3. Number of instances: 1
4. Key value pair: used the same one from previous assignments where ec2 was explored (not needed in this nonetheless I have chosen the keypair; would not have mattered in this case if left as None)
5. New VPC created
    a. Originally the CIDR was set to 10.0.0.0/16. We changed the netmask from 16 to 23 as provisions for so many subnets felt unnecessary. Setting the CIDR to 10.0.0.0/23 allowed two subnets to exist:
        i. 10.0.0.0/24
        ii. 10.0.1.0/24

    (Perhaps only one subnet was needed but provisions are made for two)

    b. The newly created vpc was named to “ECSvpc” for easier referencing in later stages.

The cluster that was created:



The new Amazon ECS console will become the default for all users starting 2023. You can opt-in to the new console today to use the new simplified workflows for deploying tasks and services, dark mode, task definition JSON editor, and new ECS features. You can continue to use the classic console for any unsupported features in the new experience.

Clusters

An Amazon ECS cluster is a regional grouping of one or more container instances on which you can run task requests. Each account receives a default cluster the first time you use the Amazon ECS service. Clusters may contain more than one Amazon EC2 instance type.

For more information, see the ECS documentation.

flas-redisapp >
CloudWatch monitoring
Default Monitoring

FARGATE

2	2	0
Services	Running tasks	Pending tasks

EC2

0	0	0	No data	No data	0
Services	Running tasks	Pending tasks	CPUUtilization	MemoryUtilization	EC2 container instances

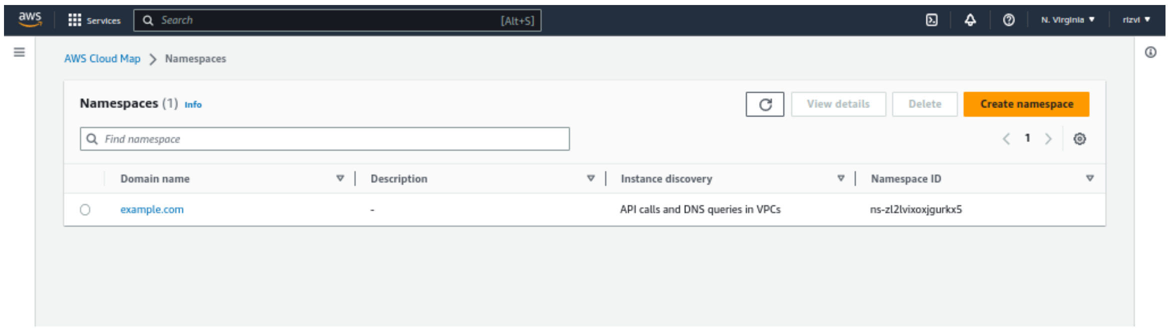
EXTERNAL

0	0	0	0
Services	Running tasks	Pending tasks	ECS container instances

© 2023, Amazon Web Services, Inc. or its affiliates.

(screenshots were taken after deployment, which is why 2 services can be seen running, otherwise at this stage there shouldn't be anything running yet.)

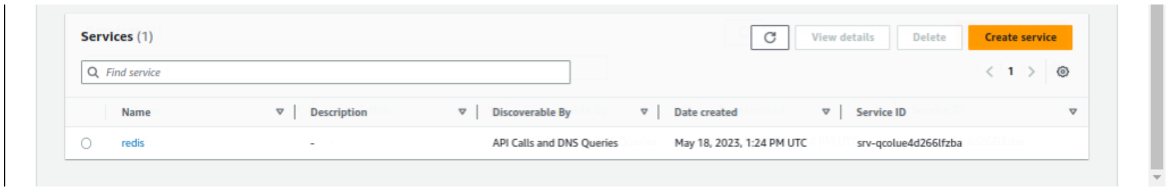
STEP 3: Create a new Namespace and a Service Discovery service inside of it



Namespaces (1) Info

Domain name	Description	Instance discovery	Namespace ID
example.com	-	API calls and DNS queries in VPCs	ns-zl2lvixoxjgurkx5

Inside this namespace is a Service Discovery service named “redis”:


Services (1)

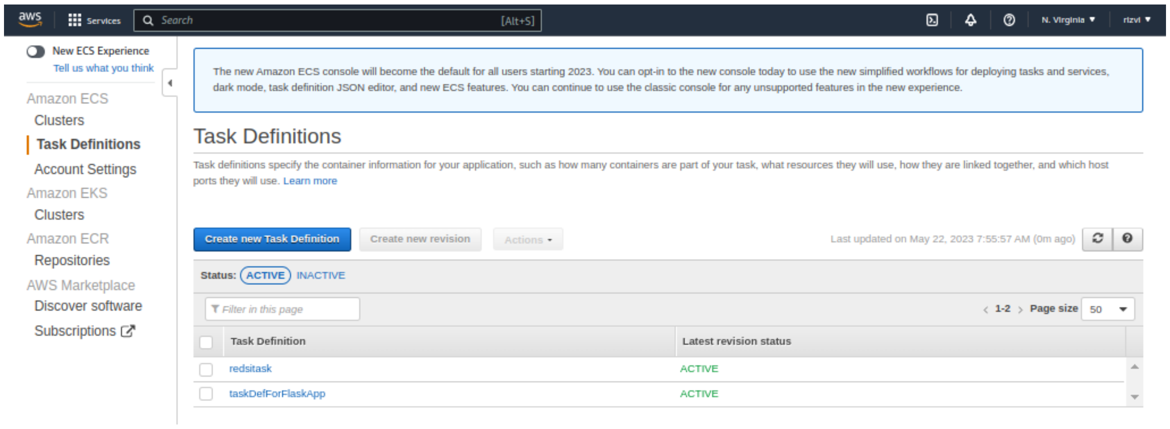
Find service

Name	Description	Discoverable By	Date created	Service ID
redis	-	API Calls and DNS Queries	May 18, 2023, 1:24 PM UTC	srv-qcolue4d266lfzba

STEP 4 : Defining task definitions

1. Launch type: Fargate
2. Task memory: 2gb
3. Task CPU: 1vCPU

4. Provided container images for both flask and redis in separate task definitions for each.
    a. Port mapping for redis: 6379
    b. Port mapping for flask: 5000
5. While defining the container for flask, we also provided a key-value pair for environment variable we talked about in step 1.
    a. Key: SERVICE_DISCOVERY
    b. Value: redis (name of the Service Discovery service shown in previous step)


The new Amazon ECS console will become the default for all users starting 2023. You can opt-in to the new console today to use the new simplified workflows for deploying tasks and services, dark mode, task definition JSON editor, and new ECS features. You can continue to use the classic console for any unsupported features in the new experience.

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

Last updated on May 22, 2023 7:55:57 AM (0m ago)

Status: ACTIVE INACTIVE

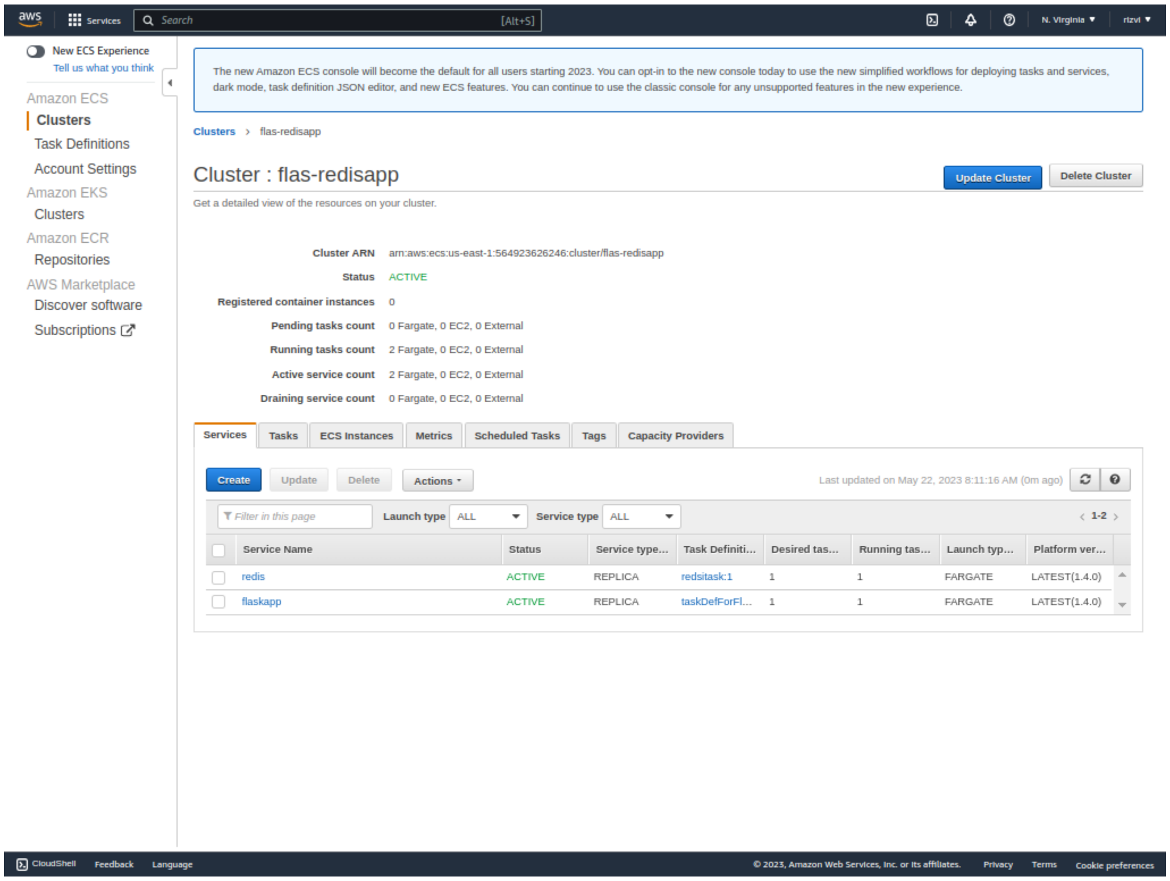
Task Definition	Latest revision status
redsitask	ACTIVE
taskDefForFlaskApp	ACTIVE

STEP 5: Create services inside the cluster

1. Configure Service:
    a. Launch type: FARGATE
    b. Task definition: taskDefForFlaskApp or redistask (former for flask service and latter for redis service)

c. Cluster: flas-redisapp
d. Number of tasks: 1

2. Configure Network:
   a. Selected the VPC named ECSvpc
      i. Selected both subnets that were created
   b. For Flask service only, we also entered the service discovery section and chose the namespace “example.com” and Service Discovery service “redis”. (for redis service this step was not required)



The new Amazon ECS console will become the default for all users starting 2023. You can opt-in to the new console today to use the new simplified workflows for deploying tasks and services, dark mode, task definition JSON editor, and new ECS features. You can continue to use the classic console for any unsupported features in the new experience.

Clusters › flas-redisapp

Cluster : flas-redisapp

Get a detailed view of the resources on your cluster.

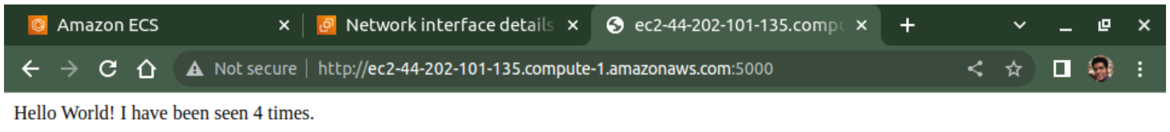
Cluster ARN	arn:aws:ecs:us-east-1:564923626246:cluster/flas-redisapp
Status	ACTIVE
Registered container instances	0
Pending tasks count	0 Fargate, 0 EC2, 0 External
Running tasks count	2 Fargate, 0 EC2, 0 External
Active service count	2 Fargate, 0 EC2, 0 External
Draining service count	0 Fargate, 0 EC2, 0 External

Last updated on May 22, 2023 8:11:16 AM (0m ago)

Launch type ALL Service type ALL

Service Name	Status	Service type...	Task Definiti...	Desired tas...	Running tas...	Launch typ...	Platform ver...
redis	ACTIVE	REPLICA	redsitask:1	1	1	FARGATE	LATEST(1.4.0)
flaskapp	ACTIVE	REPLICA	taskDefForFl...	1	1	FARGATE	LATEST(1.4.0)

OUTPUT:


Amazon ECS
Network interface details
ec2-44-202-101-135.compu
Not secure
http://ec2-44-202-101-135.compute-1.amazonaws.com:5000

```
Hello World! I have been seen 4 times.
```