

# Amazon Elastic Container Service

- 1: Pre-requisite
- 2: Create ECR Repository
- 3: Create Cloud9 Environment
- 4: Create Docker Image
- 5: Launch ECS Cluster
- 6: Create Task Definition
- 7: Create Service
- 8: Clean up

# 1. Pre-requisite: Login to AWS console or Create an Account

## 2.Create ECR Repository

- Navigate to services in AWS and search for Elastic container service (ECR).
- Create a repository with Identifiable name
- Visibility Settings --> Private
- Scan on push --> Turn on (latest changes will be used)

Push commands for my-ecr-repow

macOS / Linux

Windows

Make sure that you have the latest version of the AWS CLI and Docker installed. For more information, see [Getting Started with Amazon ECR](#).

Use the following steps to authenticate and push an image to your repository. For additional registry authentication methods, including the Amazon ECR credential helper, see [Registry Authentication](#).

1. Retrieve an authentication token and authenticate your Docker client to your registry.  
Use the AWS CLI:

```
aws ecr get-login-password --region ca-central-1 | docker login --username AWS --password-stdin 458905317537.dkr.ecr.ca-central-1.amazonaws.com
```

Note: If you receive an error using the AWS CLI, make sure that you have the latest version of the AWS CLI and Docker installed.
2. Build your Docker image using the following command. For information on building a Docker file from scratch see the instructions [here](#). You can skip this step if your image is already built:

```
docker build -t my-ecr-repow .
```
3. After the build completes, tag your image so you can push the image to this repository:

```
docker tag my-ecr-repow:latest 458905317537.dkr.ecr.ca-central-1.amazonaws.com/my-ecr-repow:latest
```
4. Run the following command to push this image to your newly created AWS repository:

```
docker push 458905317537.dkr.ecr.ca-central-1.amazonaws.com/my-ecr-repow:latest
```

Close

aws

Services

Search

[Alt+S]

EC2

VPC

IAM

CloudWatch

S3

Amazon Elastic Container Registry

Private registry

Public registry

Repositories

ECR public gallery

Amazon ECS

Amazon EKS

Amazon ECR

Repositories

Private

Public

Private repositories (1)

Find repositories

Create repository

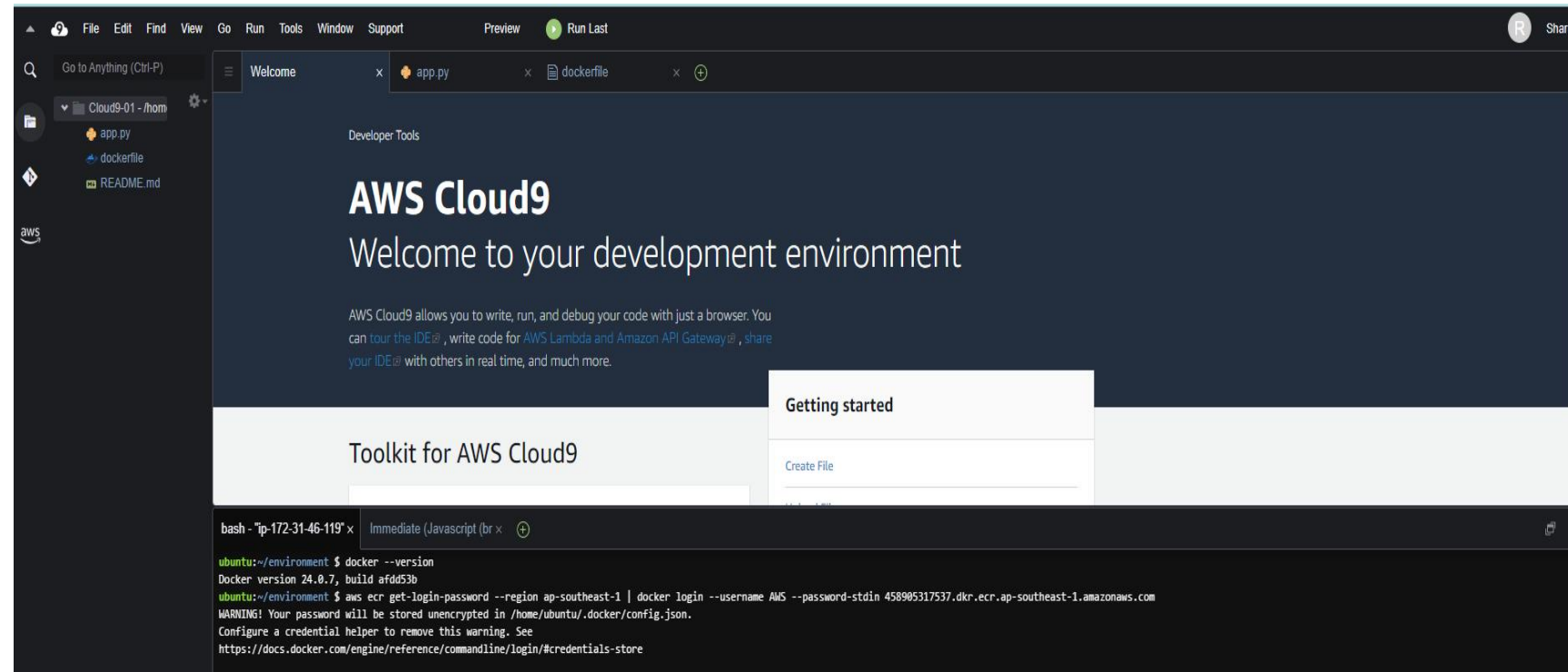
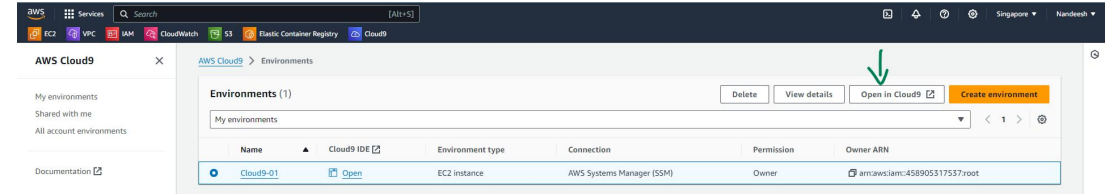
Repository name	URI	Created at	Tag immutability	Scan frequency	Encryption type	Pull through cache
my-ecr-repow	458905317537.dkr.ecr.ca-central-1.amazonaws.com/my-ecr-repow	November 30, 2023, 12:35:41 (UTC+05.5)	Disabled	Scan on push	AES-256	Inactive

### 3.Create Cloud9 Environment : Navigate to Cloud9 --> Create Environment

- -->Choose new instance --> Instance type = t3.small(2gb ram)
- --> Platform --> ubuntu Server --> cost saving -->30 min
- --> We can select the vpc configs in case of own vpc deployments
- Create and wait for few minutes untill status turn from pending to Ready
- Navigate to Ec2 to check instance created by cloud9
- bottom screen shows the bash screen
- Check for docker version

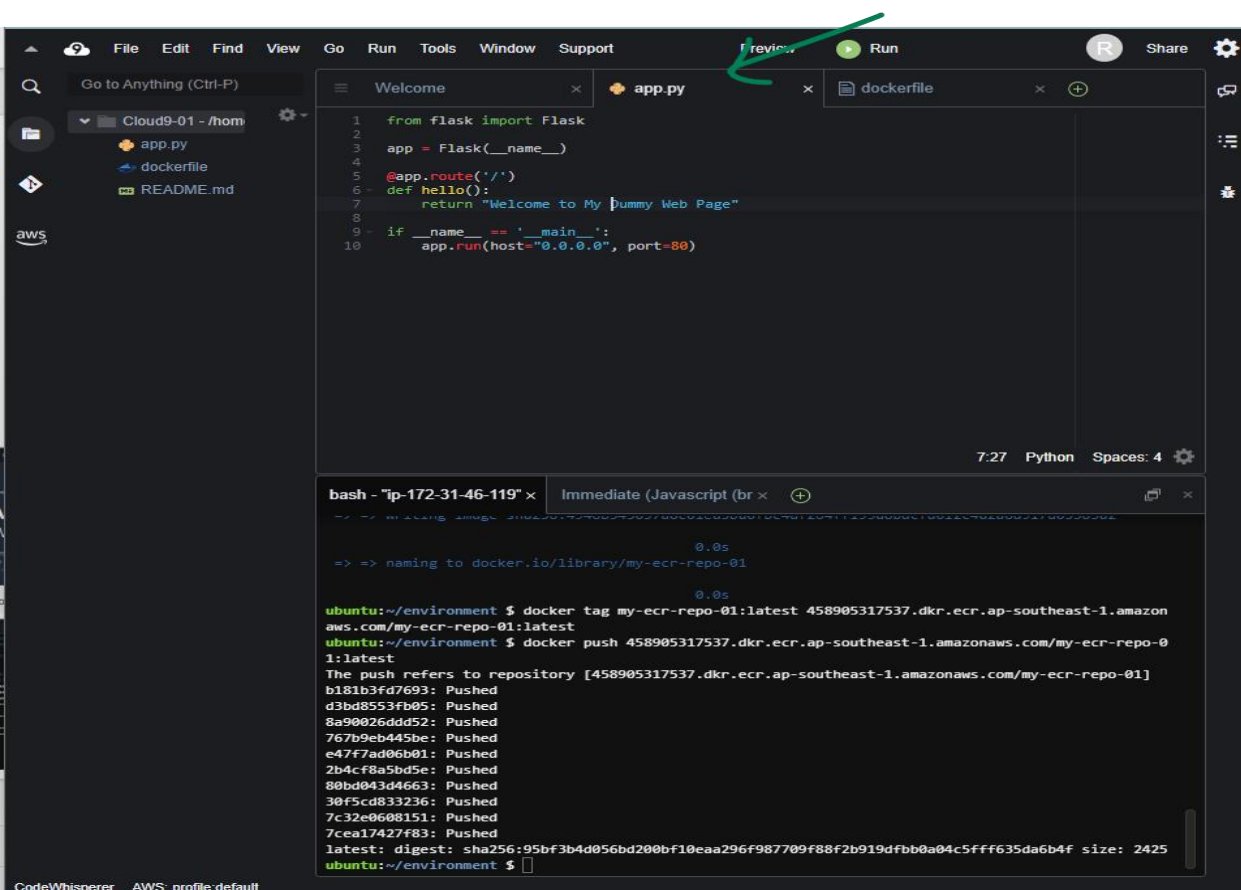
sudo apt-get update

docker --version



## 4. Create Docker Image

- Create a .py file using cloud9 IDE paste the following commands in image or [click on link](#).
- Create a Dockerfile in the new tab enter the following commands in image or [click on link](#)
- Files are ready use push commands from aws ECR Repository
- --> navigate to ECR click on your repo --> View Commands

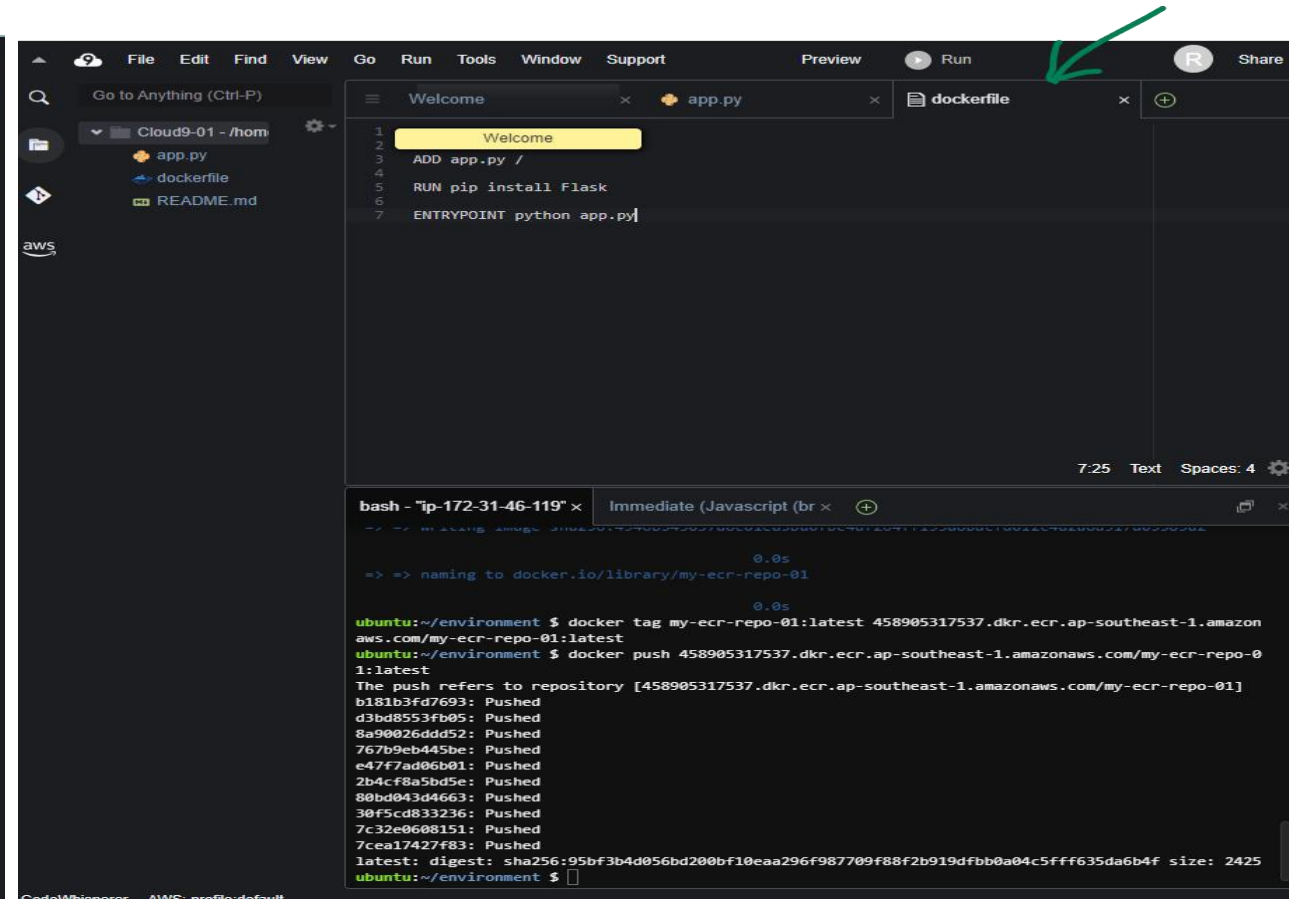


```
File Edit Find View Go Run Tools Window Support Review Run Share
Go to Anything (Ctrl-P)
Cloud9-01 - /home
app.py
dockerfile
README.md
aws

1 from flask import Flask
2 app = Flask(__name__)
3
4 @app.route('/')
5 def hello():
6     return "Welcome to My Jummy Web Page"
7
8
9 if __name__ == '__main__':
10     app.run(host="0.0.0.0", port=80)

7:27 Python Spaces: 4

bash - "ip-172-31-46-119" x Immediate (Javascript (br x +)
0.0s
=> => naming to docker.io/library/my-ecr-repo-01
0.0s
ubuntu:~/environment $ docker tag my-ecr-repo-01:latest 458905317537.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-01:latest
ubuntu:~/environment $ docker push 458905317537.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-01:latest
The push refers to repository [458905317537.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-01]
b181b3fd7693: Pushed
d3bd8553fb05: Pushed
8a90026ddd52: Pushed
767b9eb445be: Pushed
e47f7ad06b01: Pushed
2b4cf8a5bd5e: Pushed
80bd043d4663: Pushed
30f5cd833236: Pushed
7c32e0608151: Pushed
7cea17427f83: Pushed
latest: digest: sha256:95bf3b4d056bd200bf10eaa296f987709f88f2b919dfbb0a04c5fff635da6b4f size: 2425
ubuntu:~/environment $
```



```
File Edit Find View Go Run Tools Window Support Preview Run Share
Go to Anything (Ctrl-P)
Cloud9-01 - /home
app.py
dockerfile
README.md
aws

1 Welcome
2 ADD app.py /
3
4 RUN pip install Flask
5
6
7 ENTRYPOINT python app.py

7:25 Text Spaces: 4

bash - "ip-172-31-46-119" x Immediate (Javascript (br x +)
0.0s
=> => naming to docker.io/library/my-ecr-repo-01
0.0s
ubuntu:~/environment $ docker tag my-ecr-repo-01:latest 458905317537.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-01:latest
ubuntu:~/environment $ docker push 458905317537.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-01:latest
The push refers to repository [458905317537.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-01]
b181b3fd7693: Pushed
d3bd8553fb05: Pushed
8a90026ddd52: Pushed
767b9eb445be: Pushed
e47f7ad06b01: Pushed
2b4cf8a5bd5e: Pushed
80bd043d4663: Pushed
30f5cd833236: Pushed
7c32e0608151: Pushed
7cea17427f83: Pushed
latest: digest: sha256:95bf3b4d056bd200bf10eaa296f987709f88f2b919dfbb0a04c5fff635da6b4f size: 2425
ubuntu:~/environment $
```

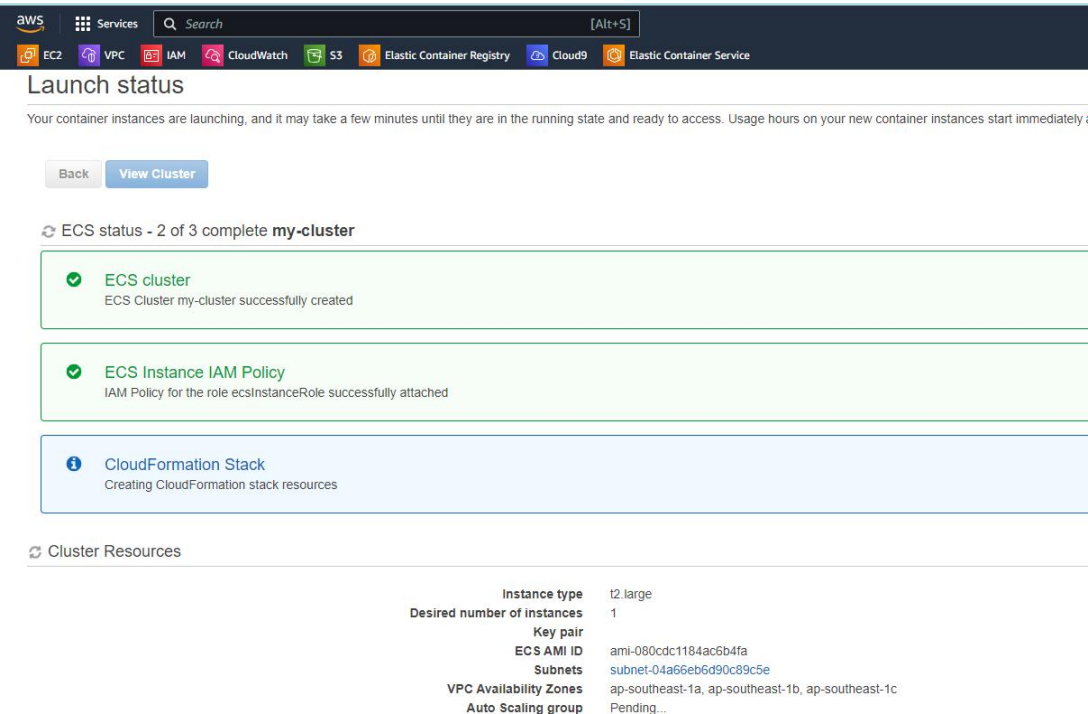
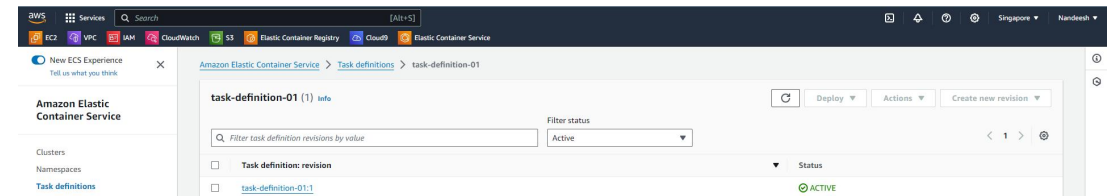
## 5.ECR Image

- After entering the command from ECR repository the image will be pushed to Registry and can view the Image url.

Image tag	Artifact type	Pushed at	Size (MB)	Image URI	Digest	Scan status	Vulnerabilities
latest	Image	November 30, 2023, 14:22:40 (UTC+05.5)	386.56	Copy URI	sha256:95bf3b4d056bd200bf10eaa296f987709f88f2b919dfbb0a04c5fff635da6b4f	Complete	1 Critical + 270 others (details)

# 6. Launch ECS Cluster

- Navigate to ECS--> Cluster --> Create Cluster
- Choose EC2 --> Cluster name-->Instance Configuration-t2.large
- No of Instances-1 --> Type --> AmazonLinux-2
- Configure VPC-SUBNETS-SG and allow port 80 SSH.



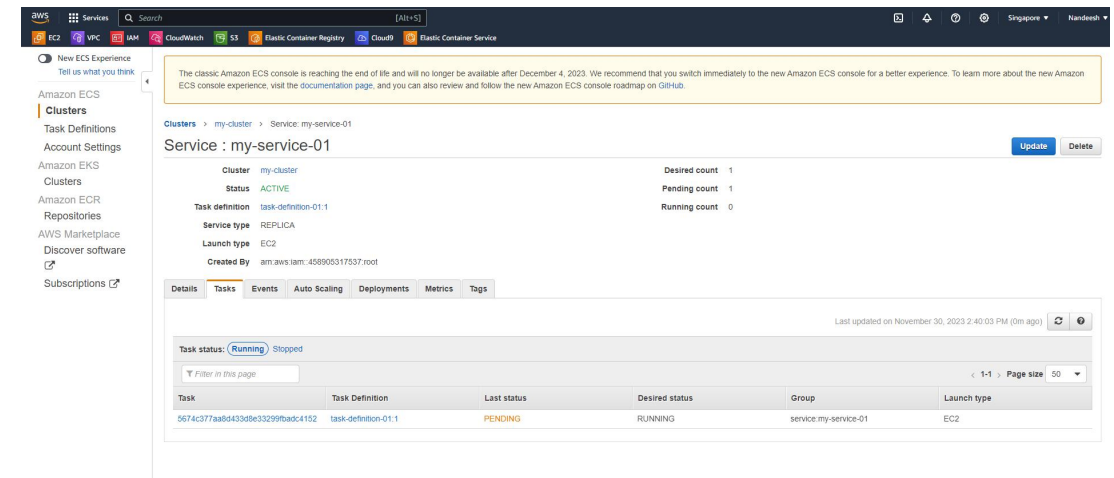
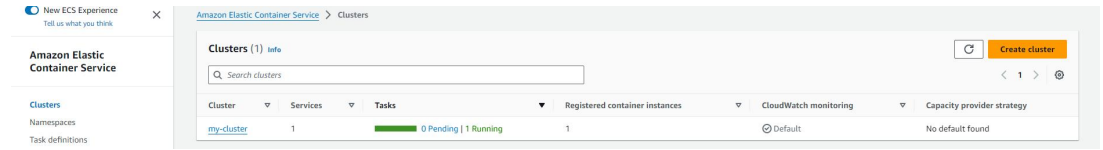
## 7.Task Definitions

- Create a Task Definition --> choose EC2 -->name-->Role= none
- -->Network mode = Default --> task memory = 128 --> Task CPU = 2vcpu
- add container -->name-->paste url of ecr image
- Hard limit = 128
- soft limit = 70
- port mapping Host = 80 conatiner = 80
- Create



# config ECS cluster -Create service

- Navigate to ECS Cluster --> Click on the cluster and the view the details
- Below we can see the Service icon click and create a Service
- launch type = EC2 --> Task definition =select the created task --> Revision = latest
- cluster = my-cluster-01 --> Service name.
- Choose Replica
- no of task = 1 , min health = 100, max health = 200
- Deploy circuit breaker = Disabled
- Load balancing = none
- Auto scalling = none
- Desired count = do not adjust
- Create and you can see the task run by service



# Validation

- Choose the Public IP of Instance Created by Cloud9 and Hit it on the internet we can see our web page as we specified in the .py file

