# 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

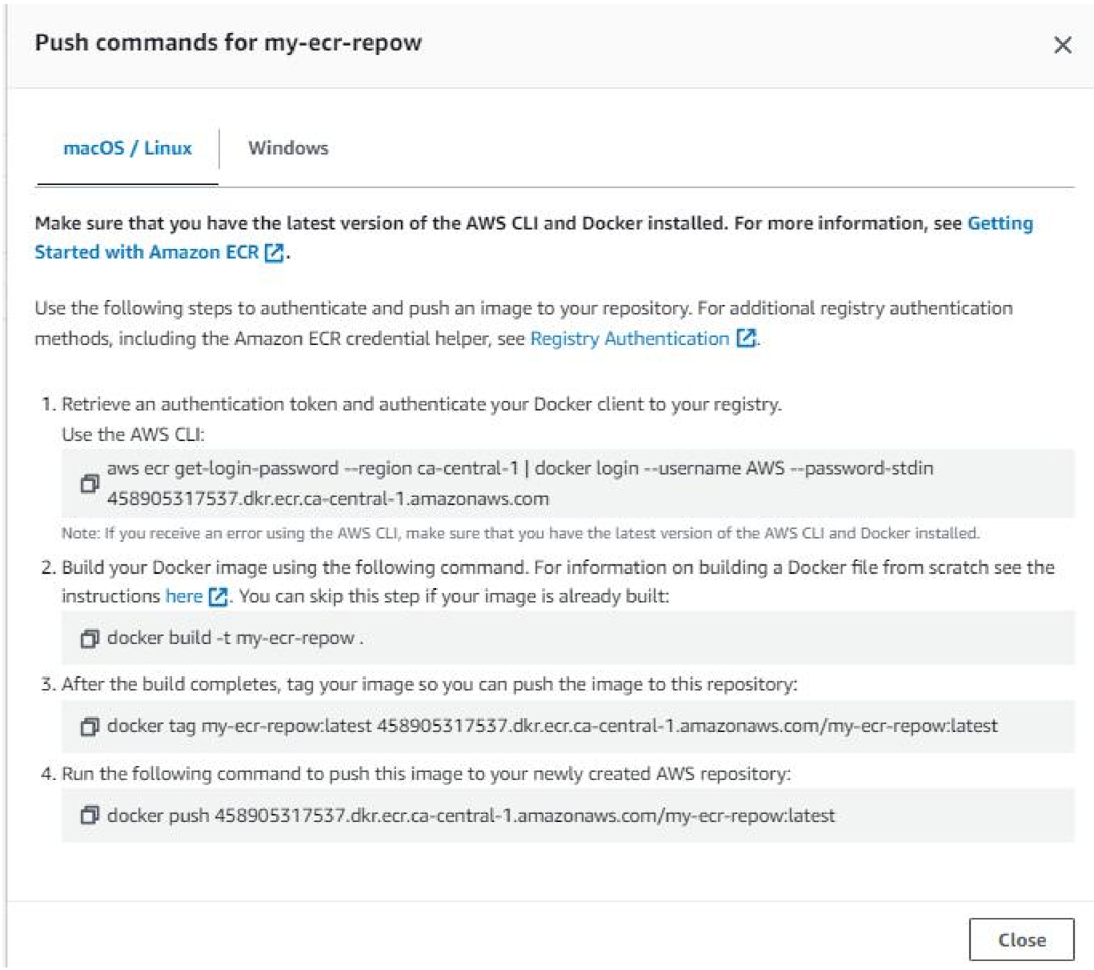
7: Create Service

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

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:

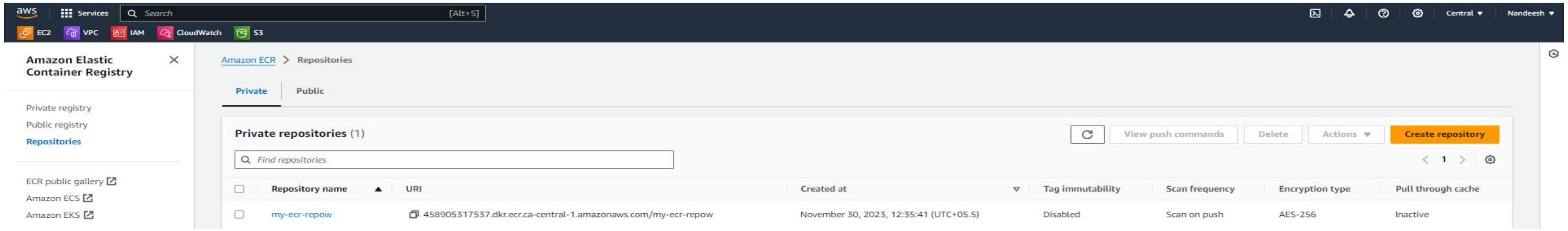
```
docker build -t my-ecr-repow .
```

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
```



Private repositories (1)

| | 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 |

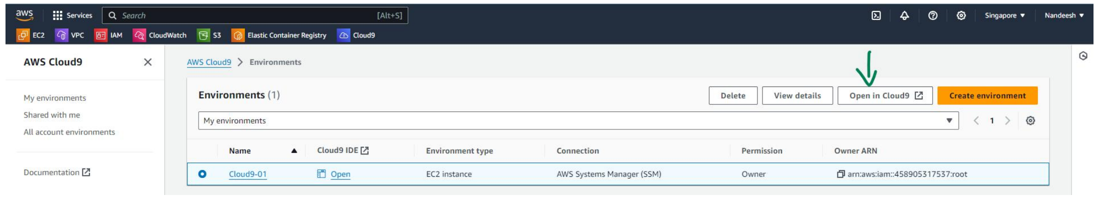
## **3.Create Cloud9 Environment :** Navigate to Cloud9 --> Create Environment

- -->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

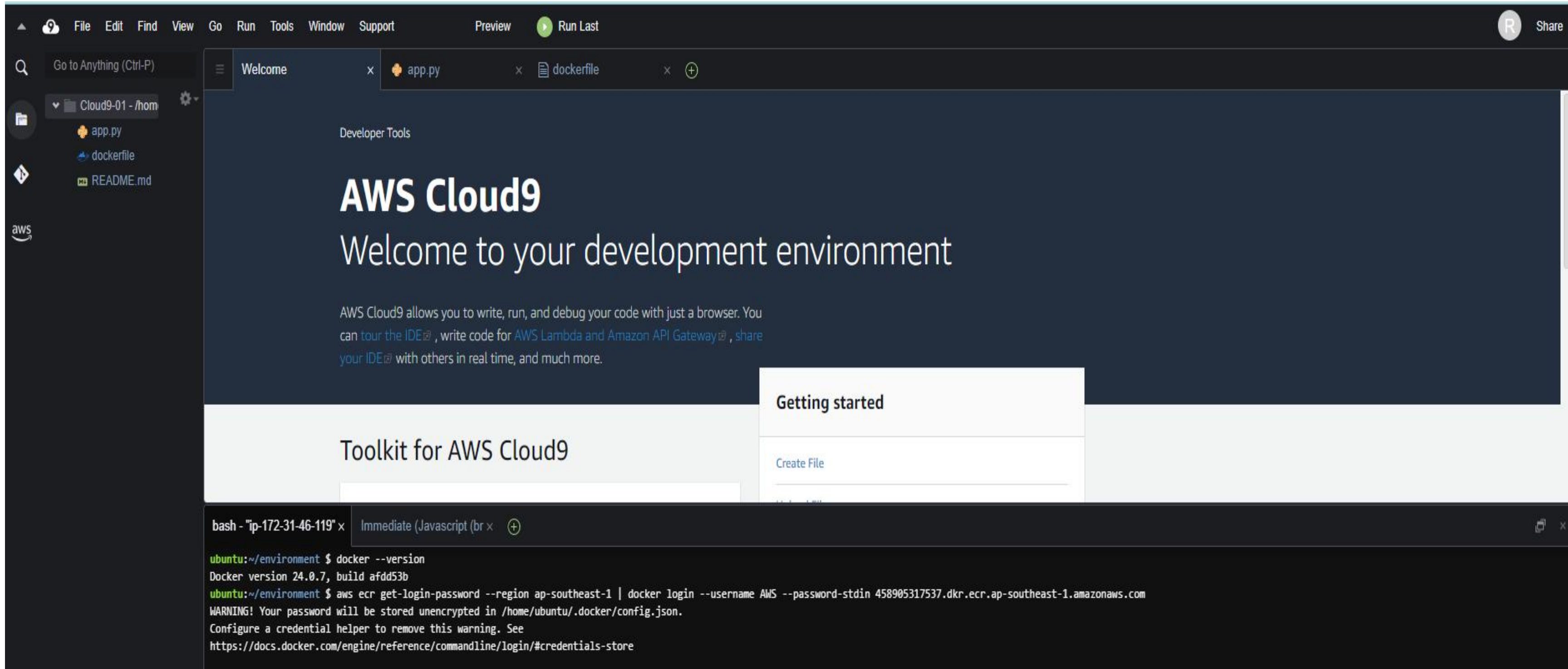
pending to Ready

- 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
```



| | Name | Cloud9 IDE | Environment type | Connection | Permission | Owner ARN |
|---|---|---|---|---|---|---|
| ● | Cloud9-01 | Open | EC2 instance | AWS Systems Manager (SSM) | Owner | arn:aws:iam::458905317537:root |

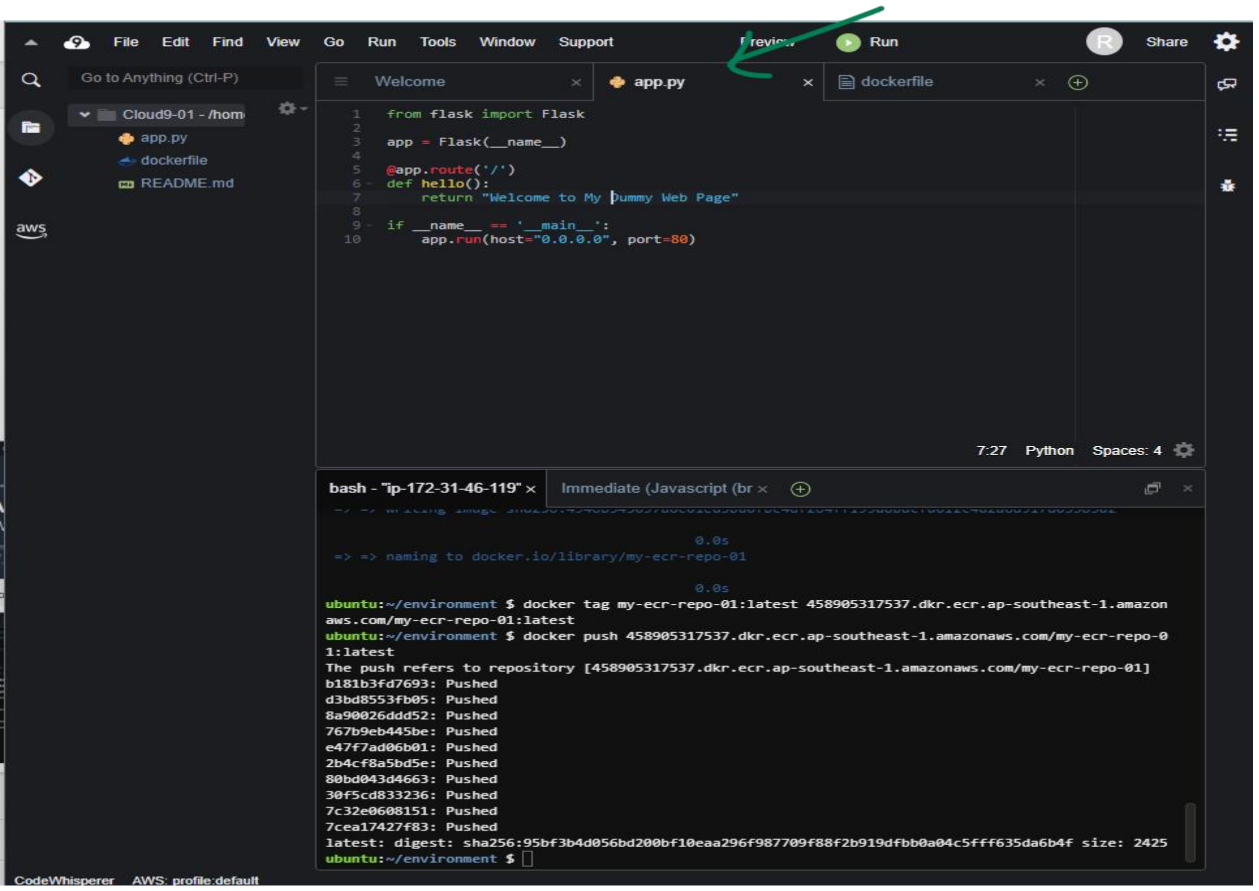


AWS Cloud9 allows you to write, run, and debug your code with just a browser. You can tour the IDE, write code for AWS Lambda and Amazon API Gateway, share your IDE with others in real time, and much more.

```
ubuntu:~/environment $ docker --version
Docker version 24.0.7, build afdd53b
ubuntu:~/environment $ aws ecr get-login-password --region ap-southeast-1 | docker login --username AWS --password-stdin 458905317537.dkr.ecr.ap-southeast-1.amazonaws.com
WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
```

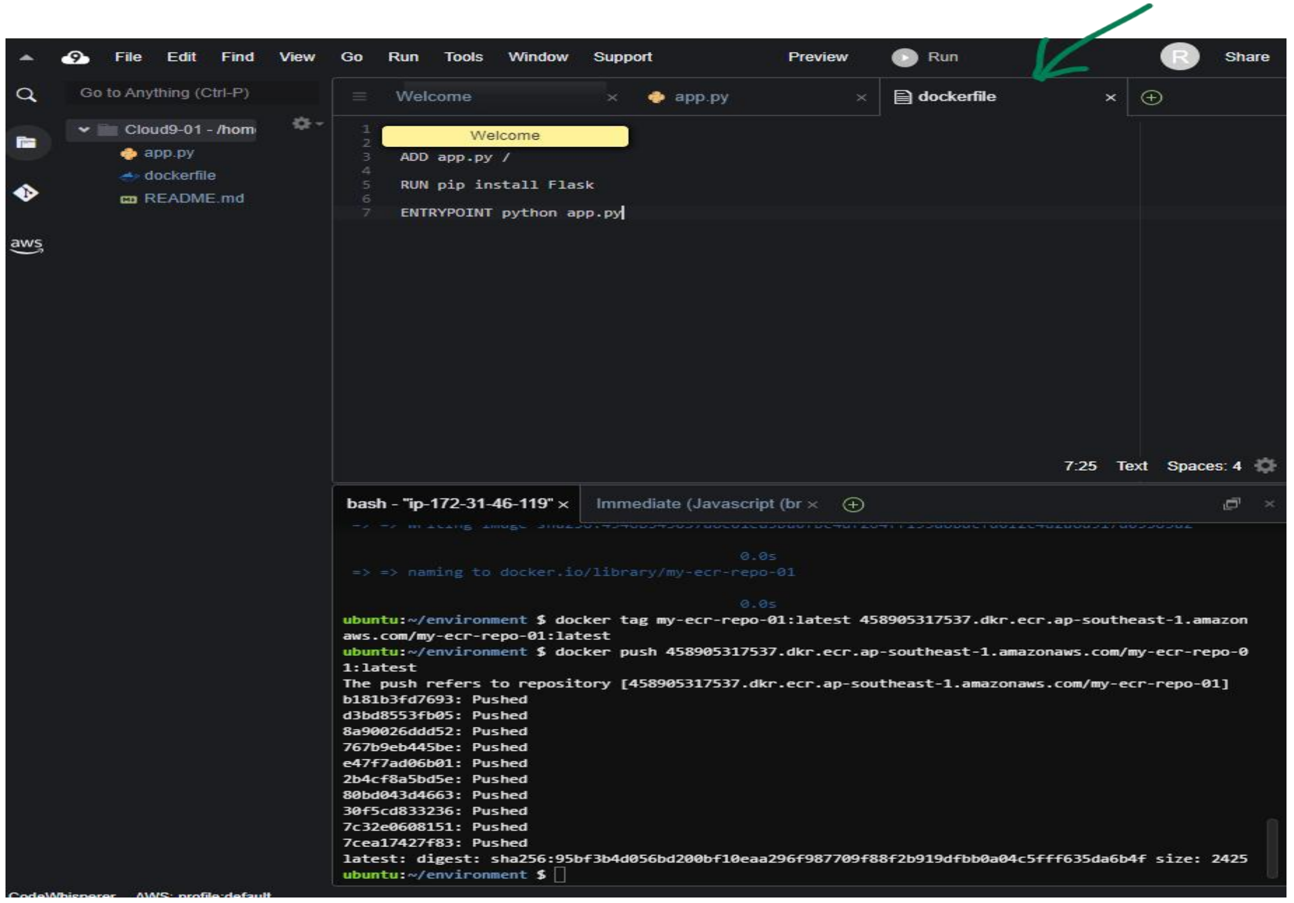
## 4.Create Docker Image

- 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



```
from flask import Flask

app = Flask(__name__)

@app.route('/')
def hello():
    return "Welcome to My Dummy Web Page"

if __name__ == '__main__':
    app.run(host="0.0.0.0", port=80)
```

```
                                                    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.amazon
aws.com/my-ecr-repo-01:latest
ubuntu:~/environment $ docker push 458905317537.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-0
1: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 $ 
```

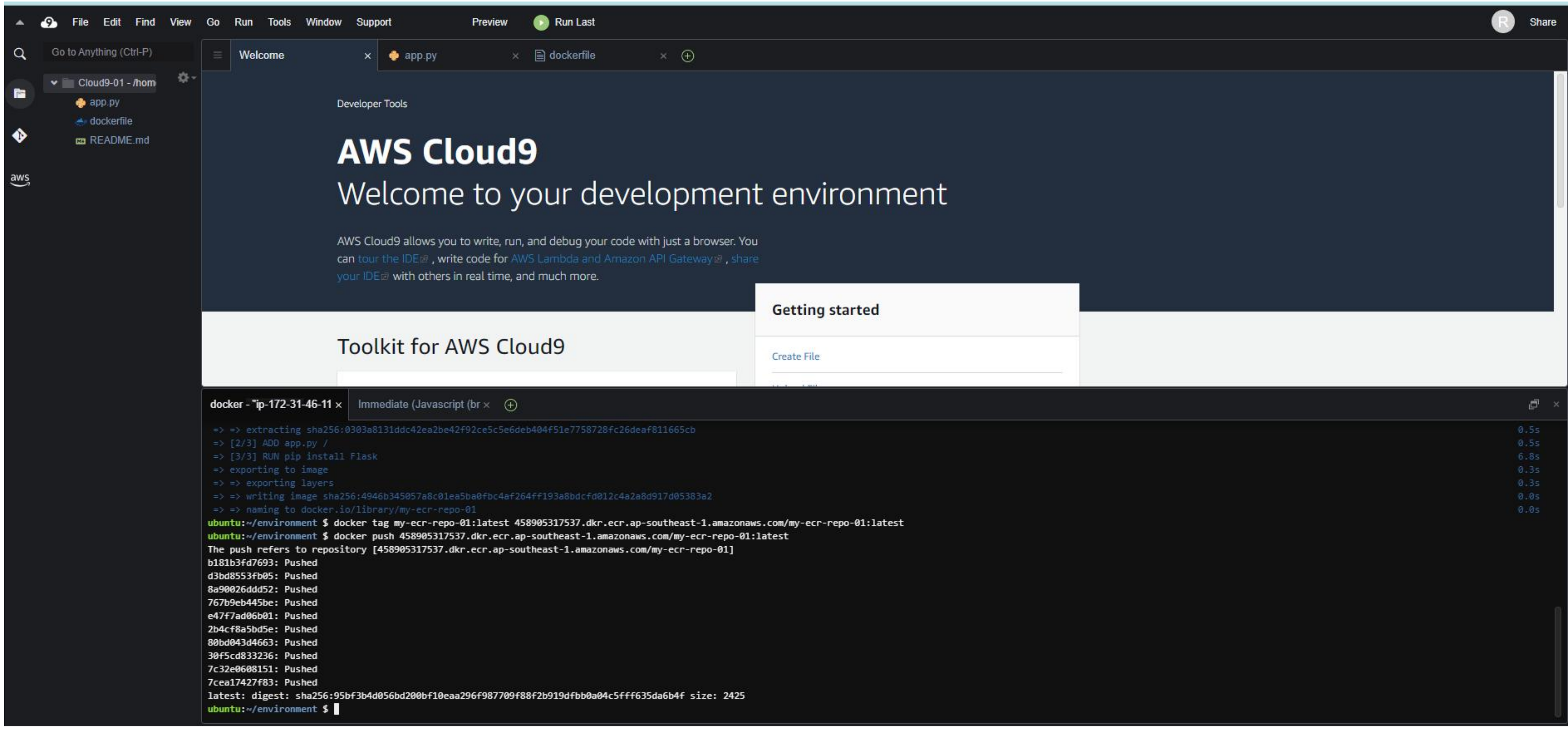


```

ADD app.py /

RUN pip install Flask

ENTRYPOINT python app.py
```

```
                                                    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.amazon
aws.com/my-ecr-repo-01:latest
ubuntu:~/environment $ docker push 458905317537.dkr.ecr.ap-southeast-1.amazonaws.com/my-ecr-repo-0
1: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 $ 
```

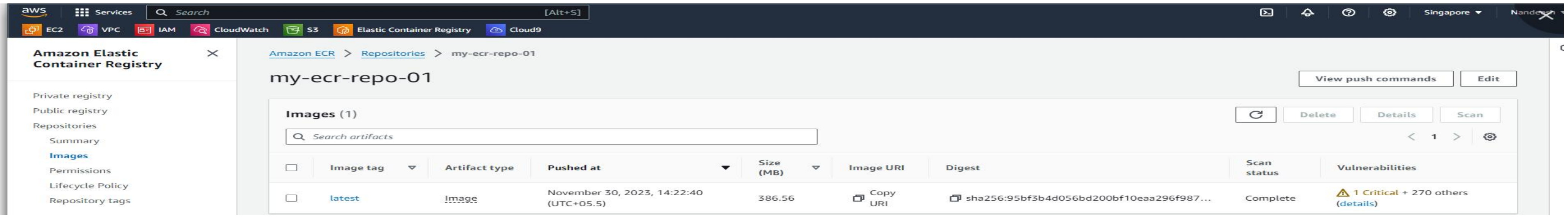
## 5.ECR Image

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



File Edit Find View Go Run Tools Window Support Preview Run Last Share
Go to Anything (Ctrl-P)
Welcome
app.py
dockerfile
Cloud9-01 - /hom
app.py
dockerfile
README.md
Developer Tools
AWS Cloud9
Welcome to your development environment
AWS Cloud9 allows you to write, run, and debug your code with just a browser. You can tour the IDE, write code for AWS Lambda and Amazon API Gateway, share your IDE with others in real time, and much more.
Getting started
Create File
Toolkit for AWS Cloud9
docker - "ip-172-31-46-11
Immediate (Javascript (br

```
 => => extracting sha256:0303a8131ddc42ea2be42f92ce5c5e6deb404f51e7758728fc26deaf811665cb        0.5s
 => [2/3] ADD app.py /                                                                           0.5s
 => [3/3] RUN pip install Flask                                                                  6.8s
 => exporting to image                                                                           0.3s
 => => exporting layers                                                                          0.3s
 => => writing image sha256:4946b345057a8c01ea5ba0fbc4af264ff193a8bdcfd012c4a2a8d917d05383a2     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 $
```



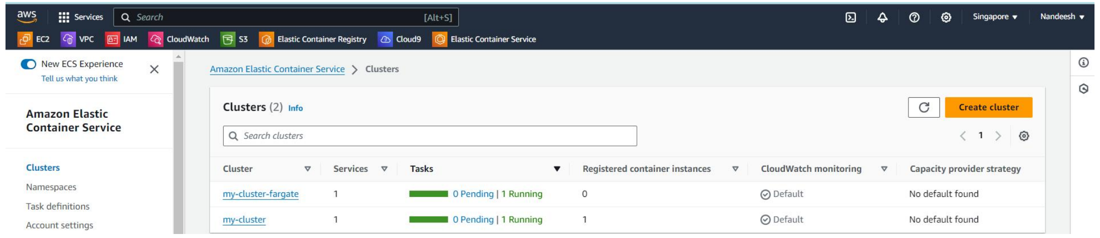
my-ecr-repo-01

Images (1)

| | 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 | sha256:95bf3b4d056bd200bf10eaa296f987... | Complete | ⚠ 1 Critical + 270 others (details) |

## 6.Launch ECS Cluster

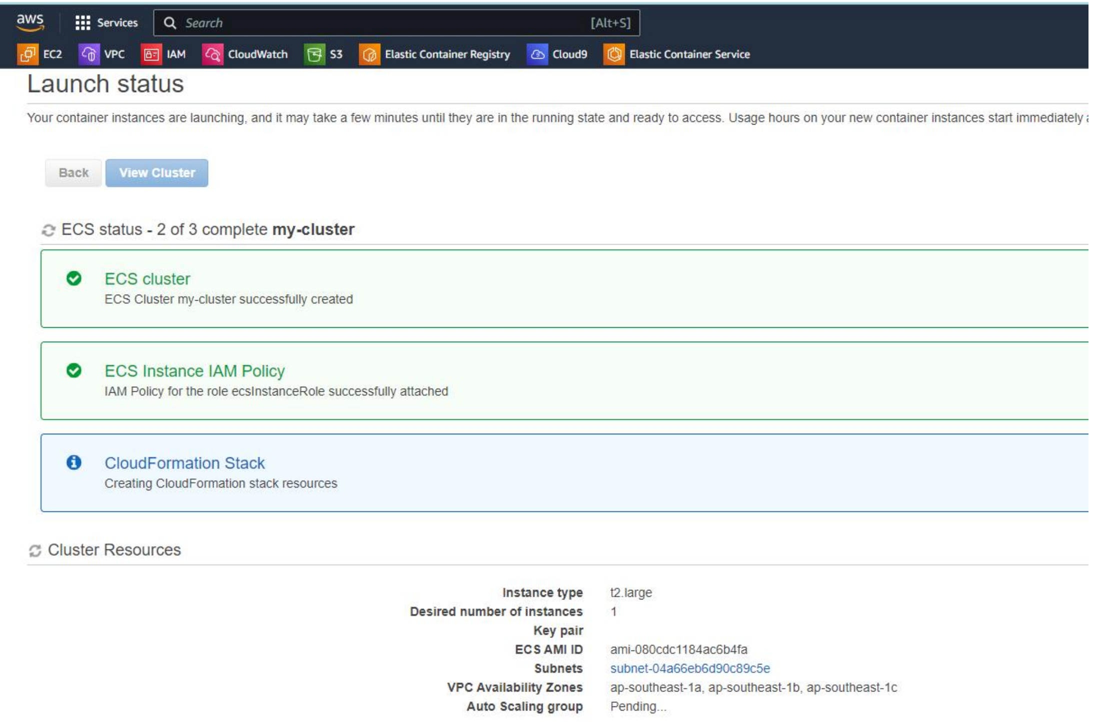
- Navigate to ECS--> Cluster --> Create Cluster
- Choose fargate --> Cluster name-->Instance Configuration-t2.large
- No of Instances-1 --> Type --> AmazonLinux-2
- Configure VPC-SUBNETS-SG and allow port 80 SSH.



Amazon Elastic Container Service > Clusters

Clusters (2) Info

| Cluster | Services | Tasks | Registered container instances | CloudWatch monitoring | Capacity provider strategy |
|---|---|---|---|---|---|
| my-cluster-fargate | 1 | 0 Pending \| 1 Running | 0 | Default | No default found |
| my-cluster | 1 | 0 Pending \| 1 Running | 1 | Default | No default found |



Launch status

Your container instances are launching, and it may take a few minutes until they are in the running state and ready to access. Usage hours on your new container instances start immediately

ECS status - 2 of 3 complete **my-cluster**

ECS cluster
ECS Cluster my-cluster successfully created

ECS Instance IAM Policy
IAM Policy for the role ecsInstanceRole successfully attached

CloudFormation Stack
Creating CloudFormation stack resources

Cluster Resources

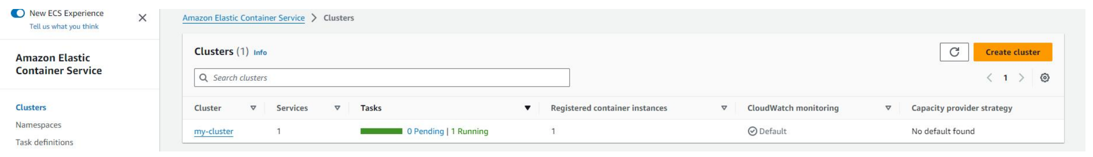
| | |
|---|---|
| Instance type | t2.large |
| Desired number of instances | 1 |
| Key pair | |
| ECS AMI ID | ami-080cdc1184ac6b4fa |
| Subnets | subnet-04a66eb6d90c89c5e |
| VPC Availability Zones | ap-southeast-1a, ap-southeast-1b, ap-southeast-1c |
| Auto Scaling group | Pending... |

## 7.Task Definitions

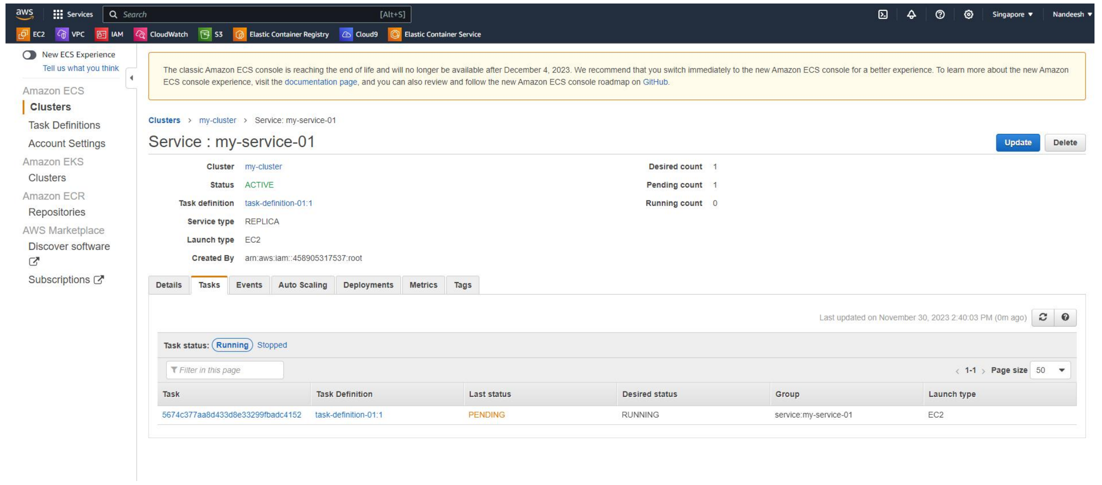
- Create a Task Definition --> choose fargate -->name-->Role=Create new role-->Network mode = Default --> task memory = 0.5gb --> Task CPU = 2vcpu
- add container -->name-->paste url of ecr image
- Hard limit = 128
- soft limit = 70
- port mapping 80 -->add container
- Create

# 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 = Fargate -->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
- chosse vpc for cluster from drop down
- 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



| Cluster | Services | Tasks | Registered container instances | CloudWatch monitoring | Capacity provider strategy |
|---|---|---|---|---|---|
| my-cluster | 1 | 0 Pending \| 1 Running | 1 | Default | No default found |



The classic Amazon ECS console is reaching the end of life and will no longer be available after December 4, 2023. We recommend that you switch immediately to the new Amazon ECS console for a better experience. To learn more about the new Amazon ECS console experience, visit the documentation page, and you can also review and follow the new Amazon ECS console roadmap on GitHub.

Clusters > my-cluster > Service: my-service-01

Service : my-service-01

| | | | |
|---|---|---|---|
| Cluster | my-cluster | Desired count | 1 |
| Status | ACTIVE | Pending count | 1 |
| Task definition | task-definition-01:1 | Running count | 0 |
| Service type | REPLICA | | |
| Launch type | EC2 | | |
| Created By | arn:aws:iam::458905317537:root | | |

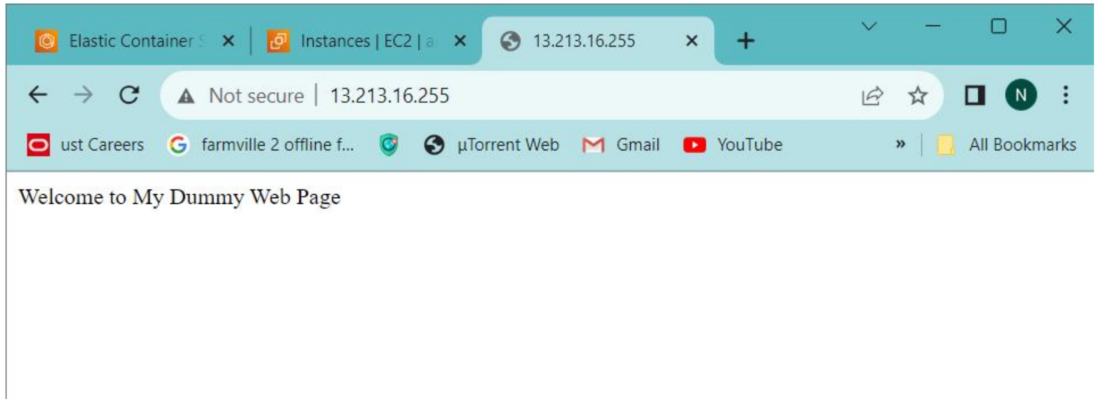
Last updated on November 30, 2023 2:40:03 PM (0m ago)

Task status: Running Stopped

| Task | Task Definition | Last status | Desired status | Group | Launch type |
|---|---|---|---|---|---|
| 5674c377aa8d433d8e33299fbadc4152 | task-definition-01:1 | PENDING | RUNNING | service:my-service-01 | EC2 |

## 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



Welcome to My Dummy Web Page