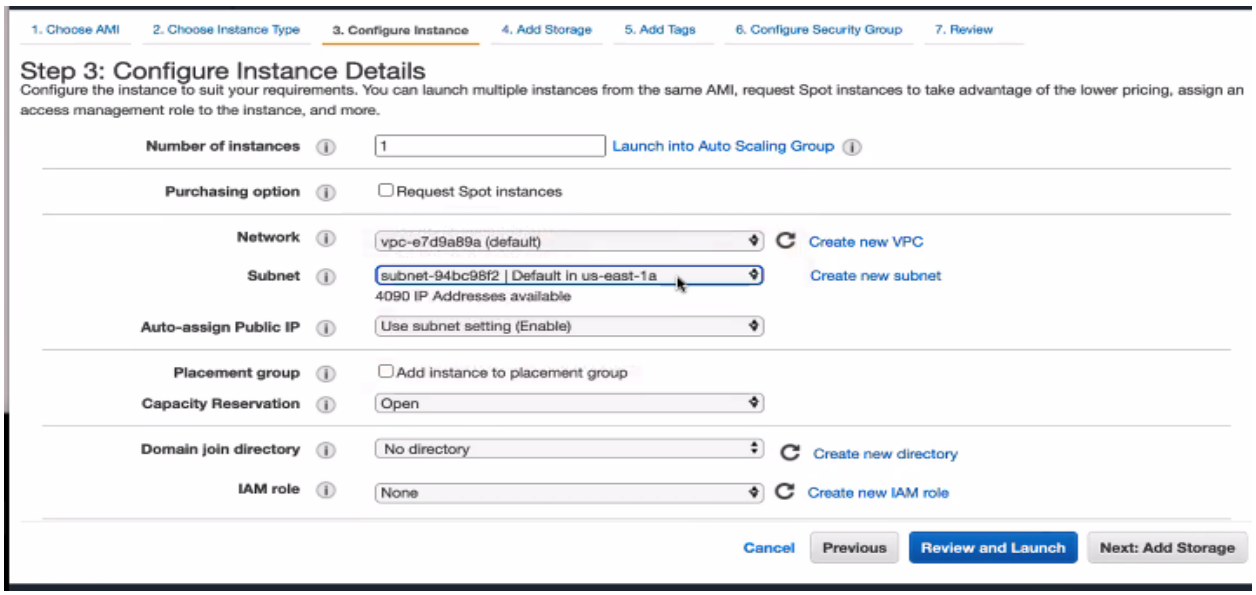
Deployment 7

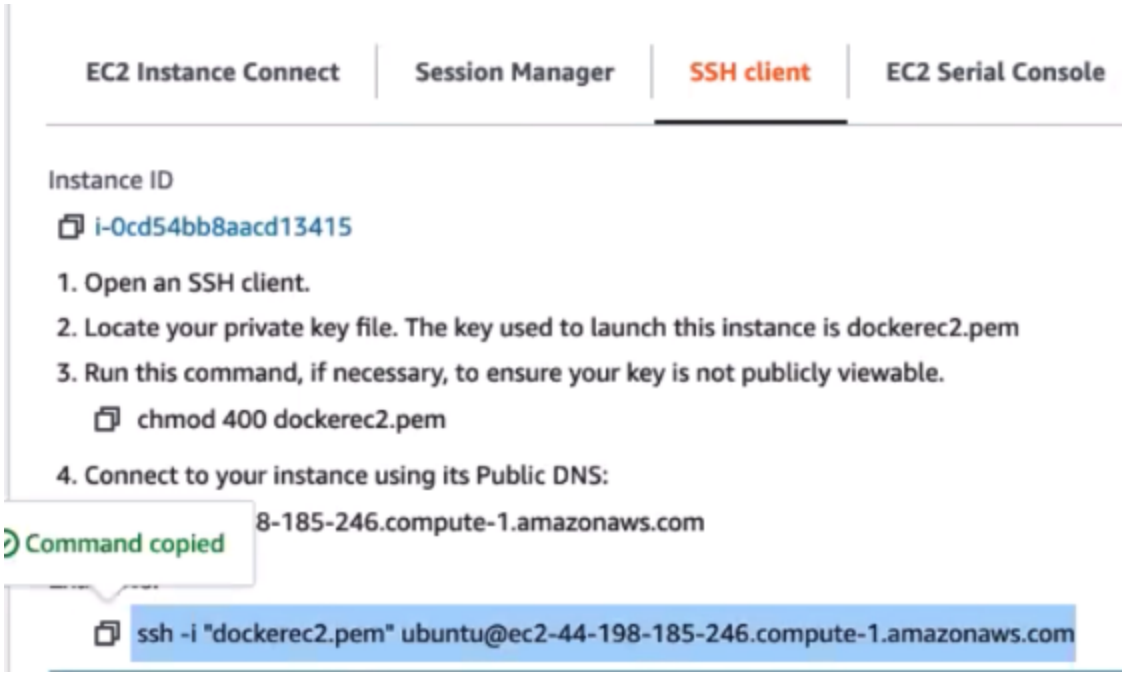
We need to create a container with Jenkins, to build a Java app onto a container hosted by an EC2.

Creating the EC2 using the Ubuntu AMI on a public subnet:

- EC2 instances and launch instance
- Select UBUNTU AMI, t2 micro, Default VPC, Public Subnet



- Configure Security group 22, 8080
- Go to your created instance, Instance ID, Connect, SSH client and execute this ssh commands



- ssh into public ec2
- sudo apt-get update
- Sudo apt-get upgrade

Now we need istall necessary packages Java, as well as Docker, installed on this EC2

https://docs.docker.com/engine/install/ubuntu/

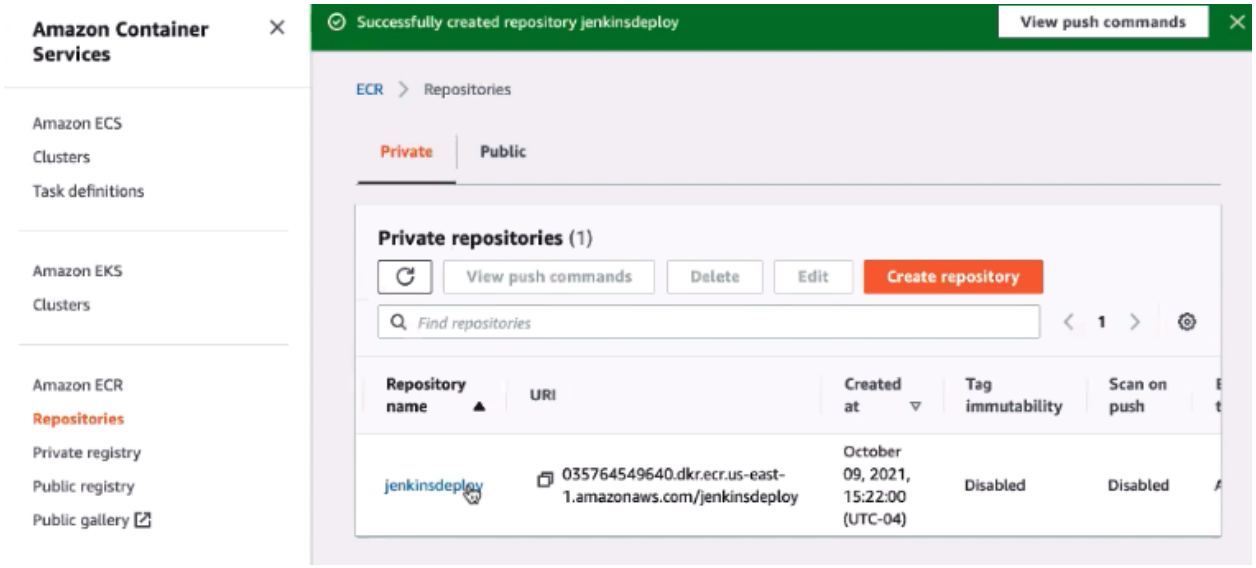
(or we can follow this link
https://dev.to/kamalhossain/how-to-run-docker-containers-in-aws-ec2-3bh0)

Sudo apt-get update
sudo apt install apt-transport-https ca-certificates curl software-properties-common
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu focal stable"
sudo apt update
sudo apt-cache policy docker-ce
sudo apt install docker-ce
sudo systemctl status docker
sudo usermod -aG docker ubuntu
sudo docker status

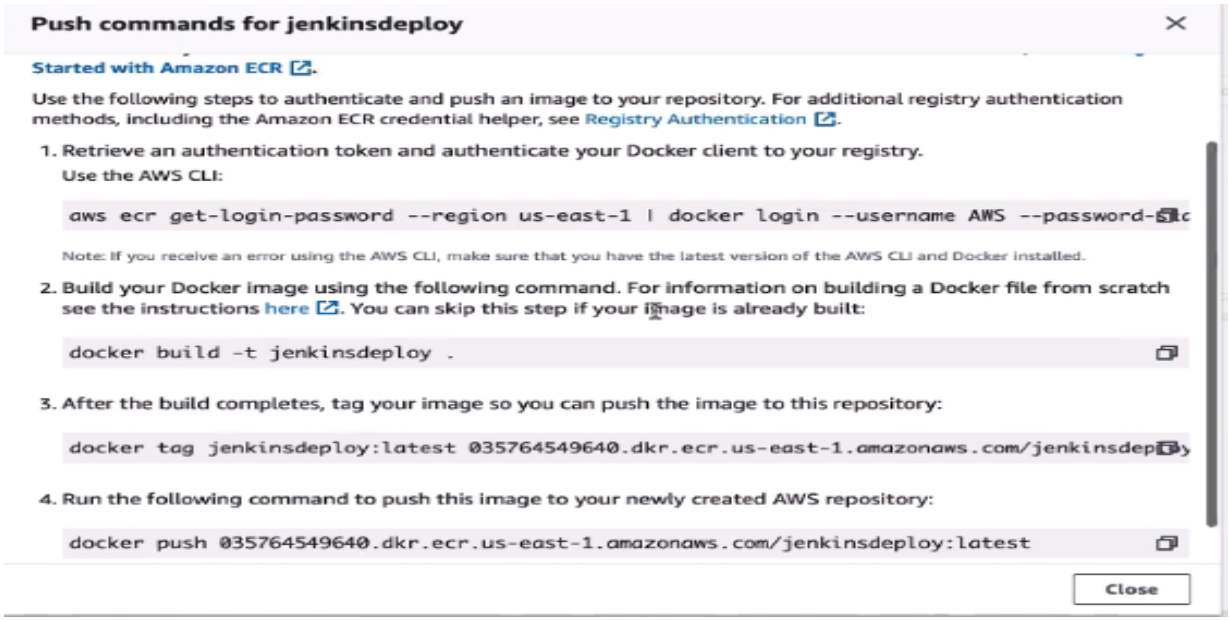
Create the ECS Cluster

On the ECS dashboard (Amazon Container Services), create a new repository under ECR.

- Select Private (we making private repository so we can utilize for Jenkin image for ECS) , enter your repository name, click Create Repository.



View push commands



We can run step 1, step 3 just change to jenkins/jenkins (docker tag jenkins/jenkins 03576549640.dkr.ecr..us-.....) step 4.

In your local terminal, get a Jenkins Docker image and push it to your ECR repository.

docker pull jenkins/jenkins
aws ecr-public get-login-password --region us-east-1 | docker login --username AWS
--password-stdin public.ecr.aws/repo-tag
docker tag image-name:tag-name public.ecr.aws/repo-tag/repo-name:tag-name
docker push public.ecr.aws/repo-tag/repo-name:tag-name

Github

Let's create GitHub repository fo Jenkins following instruction with three files:

https://github.com/DIrisova/DEPLOY07_ECS/blob/main/Dockerfile

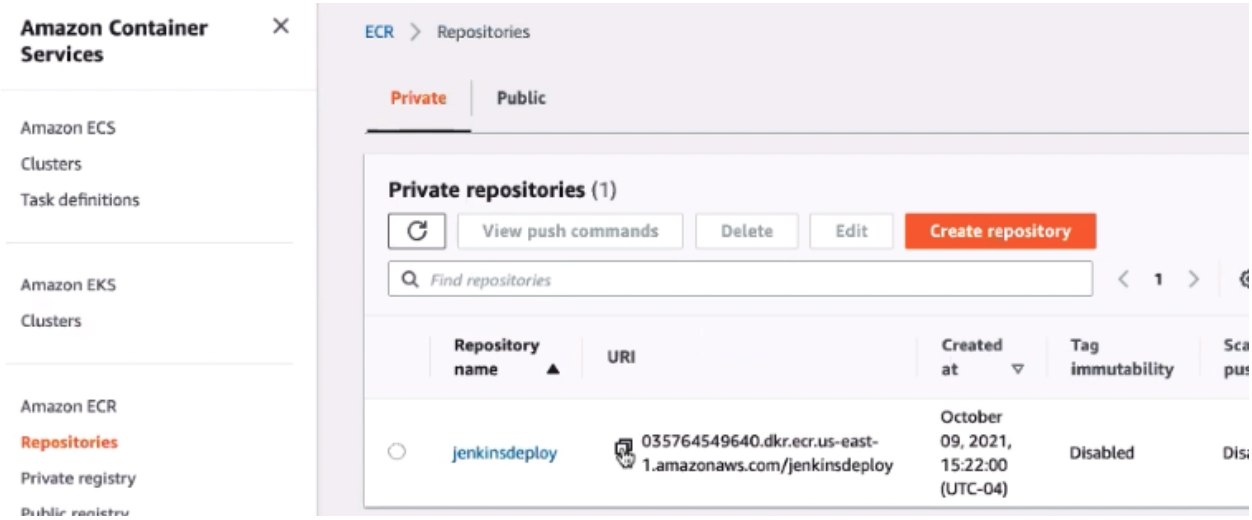
- The .jar file will holds the information to build our Java app.
- The Dockerfile will build an image of our Java app.
- The Jenkinsfile will run the Dockerfile and push our image to Dockerhub. Paste the code from the task pdf into the jenkinsfile

```
pipeline {
agent { label "your label"}
environment{
DOCKERHUB_CREDENTIALS = credentials('your username-dockerhub')
}
stages {
stage ('Build') {
steps {
sh ''' 'docker .......''' 
}
}
stage ('Login') {
steps {
sh ''' 'echo $DOCKERHUB_CREDENTIALS_PSW | docker login -u
$DOCKERHUB_CREDENTIALS_USR --password-stdin''' 
}}
stage ('Push') {
steps {
sh ''' 'docker .......''' 
}
}
}
}
```

- Drop the jar file into your repository
- Generate token from dockerhub
- Follow the steps provided by dockerhub when we generate your token..
- We should add the token as a password to our jenkins global credentials

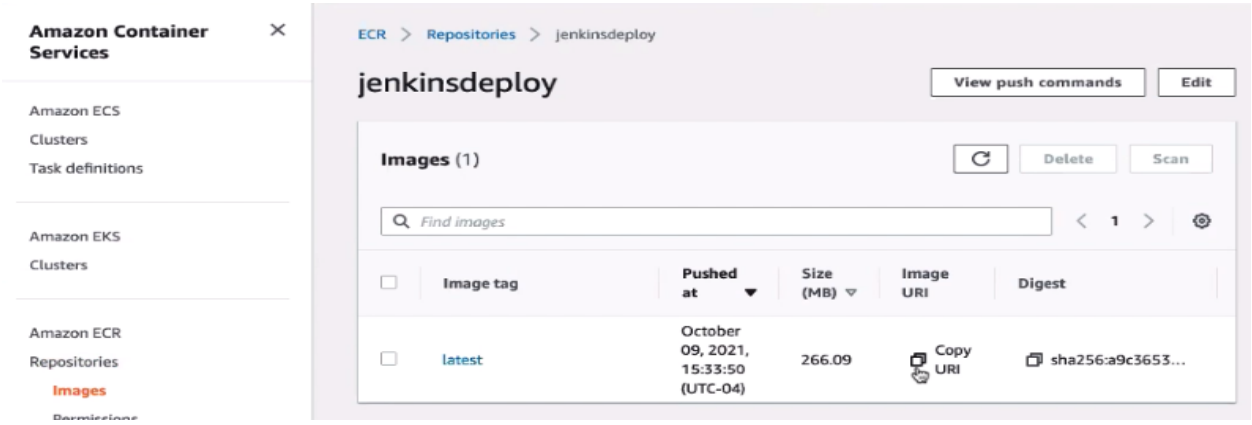
Create a cluster.

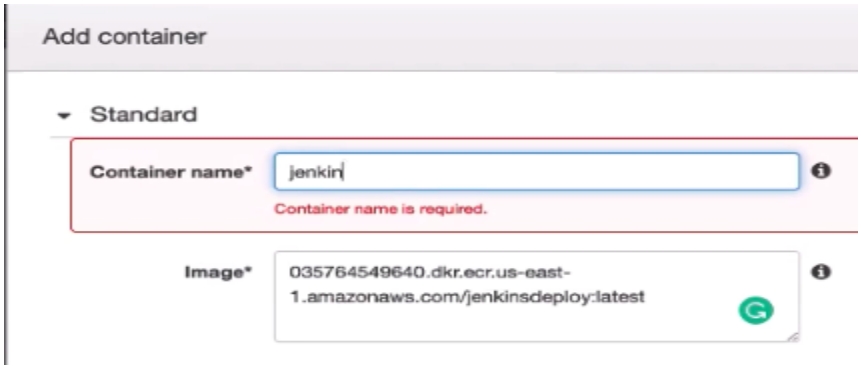
- Select Networking Only, name the cluster, create the cluster.



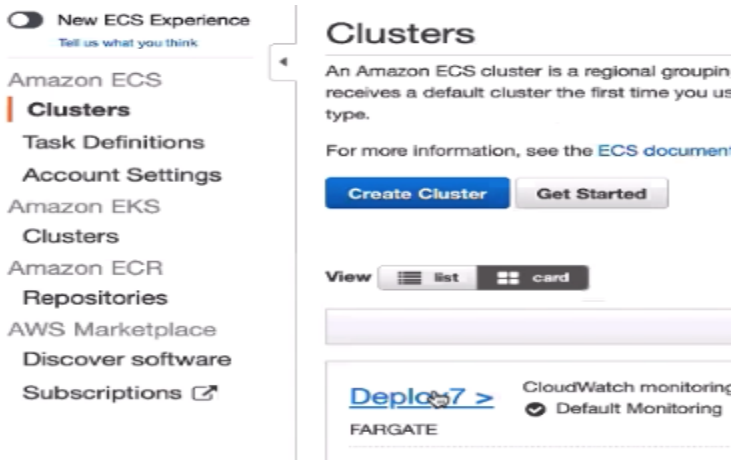
- Next, create a New Task Definition. When creating the container use the image URI from our Jenkins image we pushed earlier to ECR earlier.
- Launch status, View Cluster, Task Definitions, Create new Task Definitions, FARGATE, Task definition name, Task memory we can select minimume or 1GB Add container, for Image copy and past the image URI from our Jenkins image we pushed earlier to ECR earlier.

-

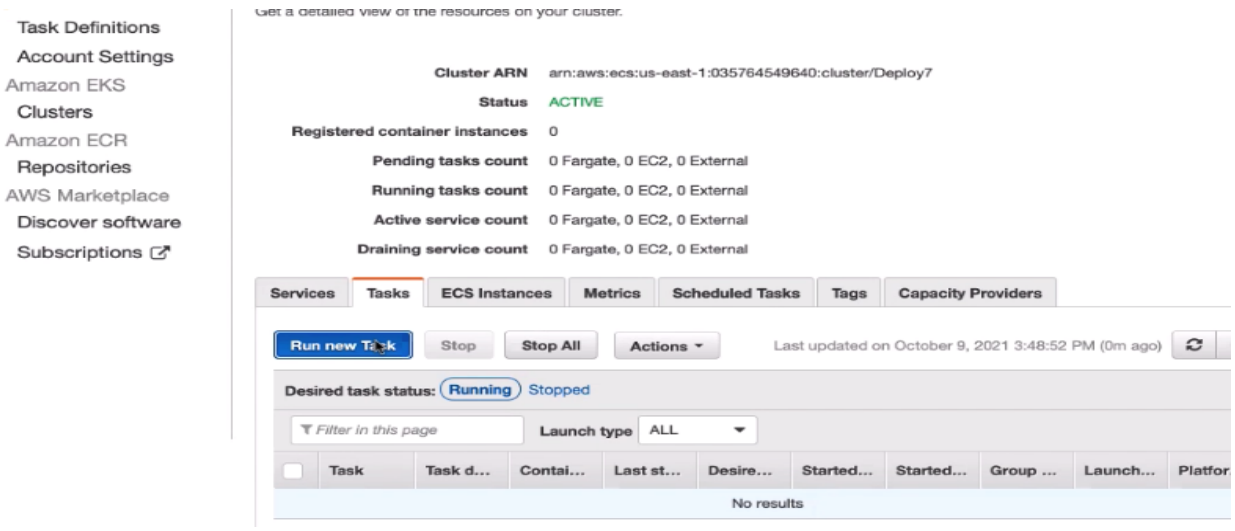




- Add
- View task definition
- Clusters, go to Deploy7

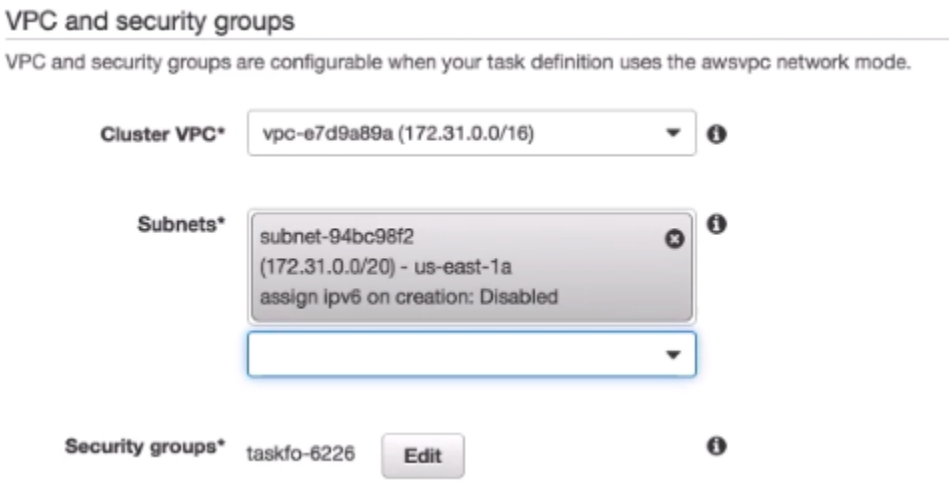


- Tasks
- Run new Task

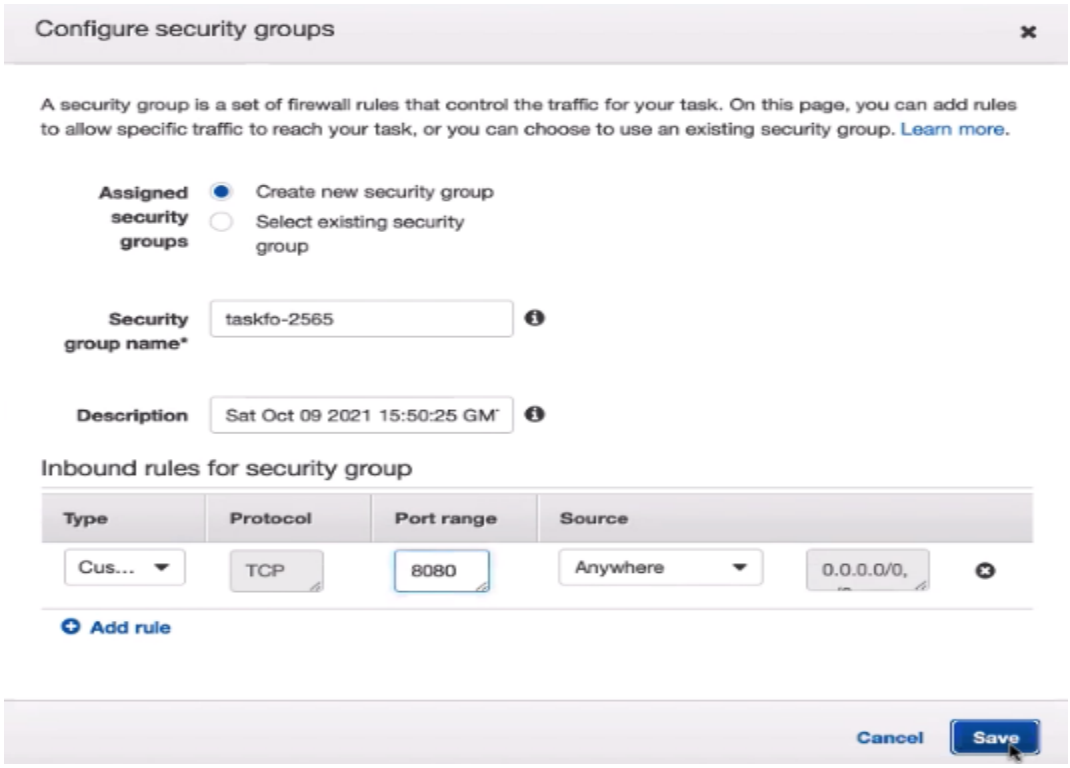


- Launch type, FARGATE

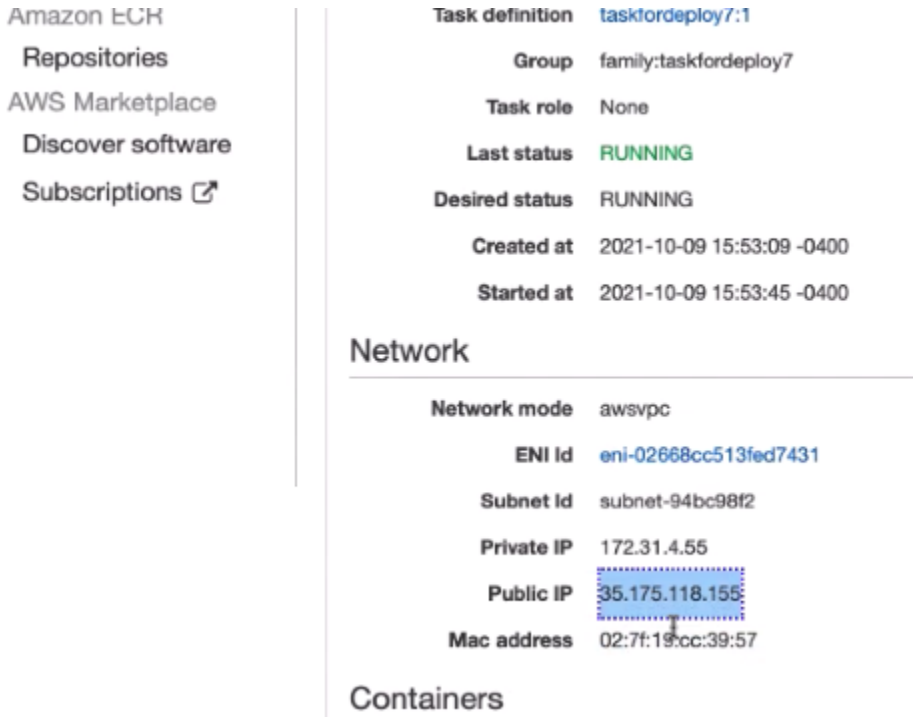
Select VPC and Subnets (should be same subnet when we created EC2)



- Security group, Edit,

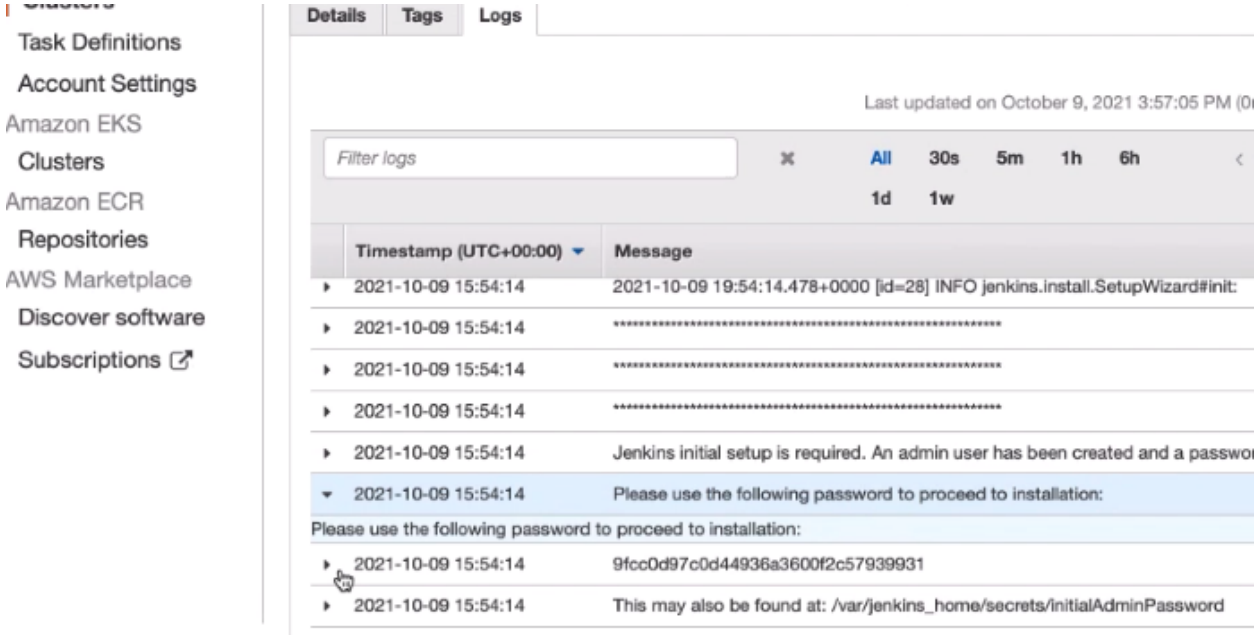


-
- Run Task
- So task running, click on Task, copy Public IP
-

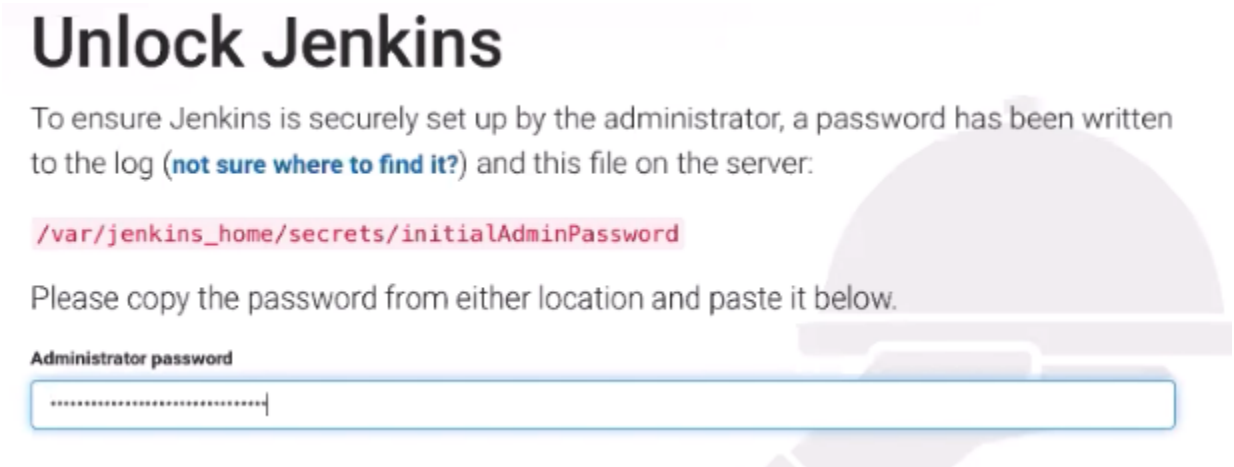


-

- Logs



- so 9fcc….. thats going to be our Jenkins password for the initial set up
- go to new tab, past Public IP and use port 8080 (Public IP:8080), past that Jenkins password.



- For Customize Jenkins select Install suggested plugins
- Create user name and password, Jenkins is ready!

Jenkins

Configure Jenkins. We need two plugins: Docker Pipeline and AWS EC2.

- Manage Jenkins, Manage Plugins

- New Item, Enter an Item name, Multibranch Pipeline

These are installed add an agent hosted by our EC2 (use private IP).

Add credentials:

- We need to add Github login using the Personal Access Token as the password.
- We need to add Docker login using the Personal Access Token as the password.
- We need to add SSH login using the IP and .pem file for the agent EC2.
- Create a multi-branch pipeline that pulls from the repo you set up.
- Run it and check your Dockerhub to see if the Java app image has been uploaded.