# Deploy Spring Boot + MYSQL RDS Application to Docker and ECS

**Prerequisites:**

1. Python2
2. Maven
3. Java-1.8.0-openjdk & Java-1.8.0-openjdk-devel
4. Awscli
5. Docker
6. git

Create EC2 Instance 🡪 Here I used RHEL Amazon Machine Image

**Update yum command:**

yum update -y

**Install java,git and wget:**

yum install java-1.8.0-openjdk java-1.8.0-openjdk-devel wget git -y

**Install Python2:**

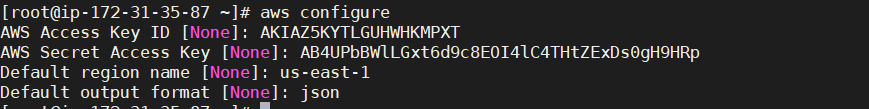
yum install python2-pip -y

alternatives --config python

**Install Awscli:**

pip2 install awscli

aws configure



**Install Docker on RHEL:**

Dnfconfig-manager --add repo=https://download.docker.com/linux/centos/docker-ce.repo

dnf install docker-ce-3:18.09.1-3.el7

systemctl enable docker

systemctl start docker

**Create Mysql RDS:**

* Please check below code to create MYSQL RDS instance
* Note: Here we should mention dbname to connect

aws rds create-db-instance \

--db-name mysqldb \

--allocated-storage 20 \

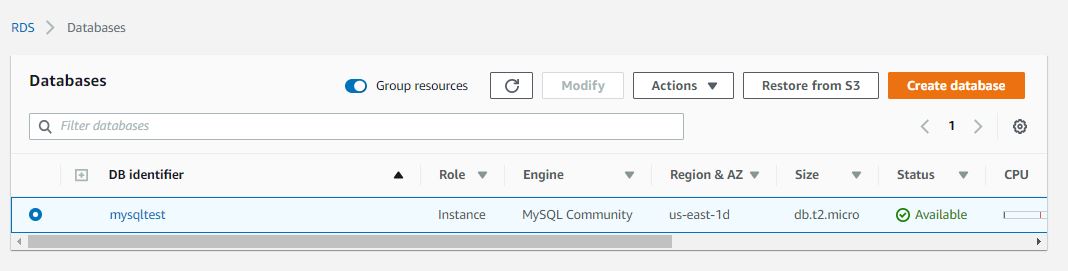
--db-instance-class db.t2.micro \

--db-instance-identifier mysqltest \

--engine mysql \

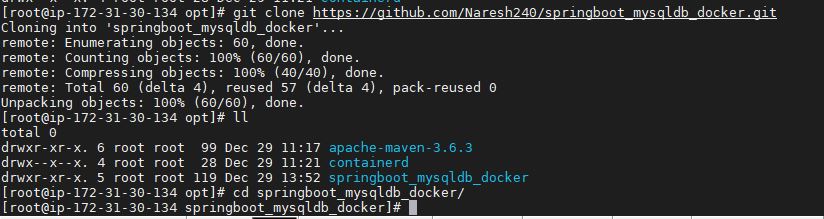
--master-username master \

--master-user-password Naresh#240



**Pull file from github:**

git clone <https://github.com/Naresh240/springboot_mysqldb_docker.git>

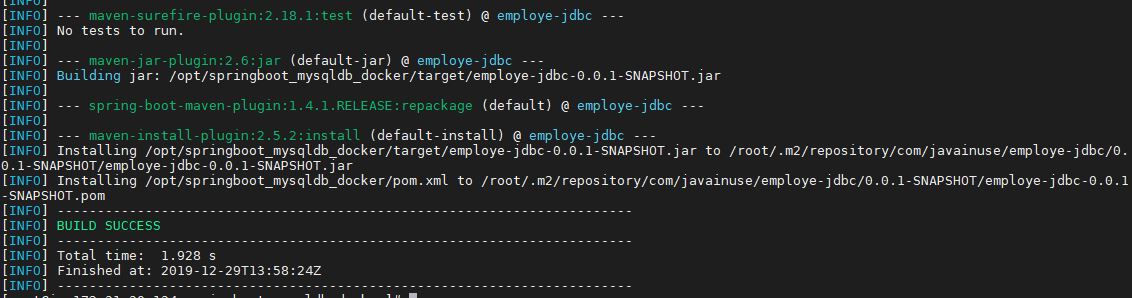


Move into file “springboot\_mysqldb\_docker”

cd springboot\_mysqldb\_docker

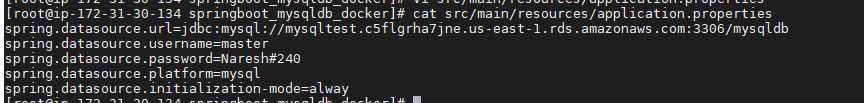
Then create jar file by using below command

mvn clean install



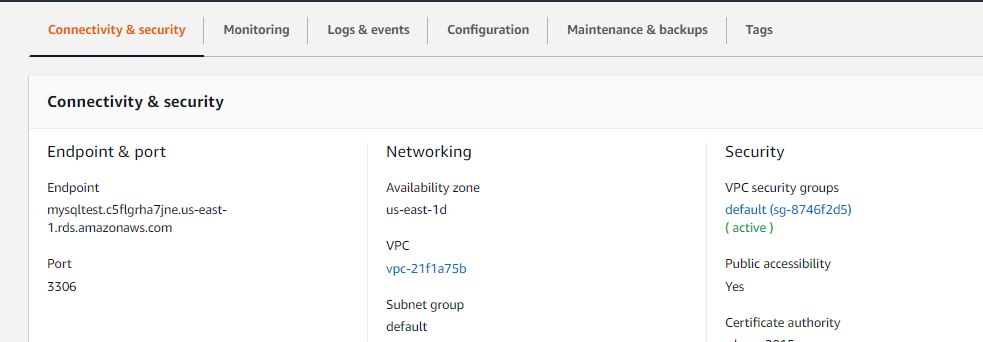
**Requirements to createDocker Image:**

* Here we need to check **application.properties** file [Path:src/main/resources/application.properties]



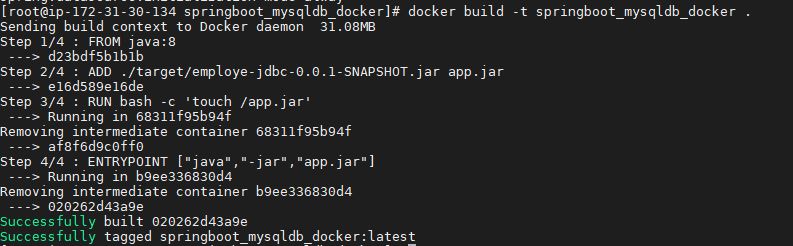
url=jdbc:mysql://<Endpoint of date base>:<port-number>/dbname

In below figure we can check Endpoint



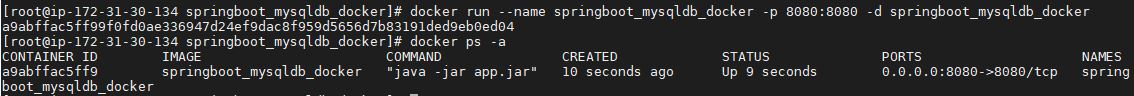
**Create docker image:**

docker build -t springboot\_mysqldb\_docker .



**Run Docker image using below command:**

docker run --name springboot\_mysqldb\_docker -p 8080:8080 -d springboot\_mysqldb\_docker

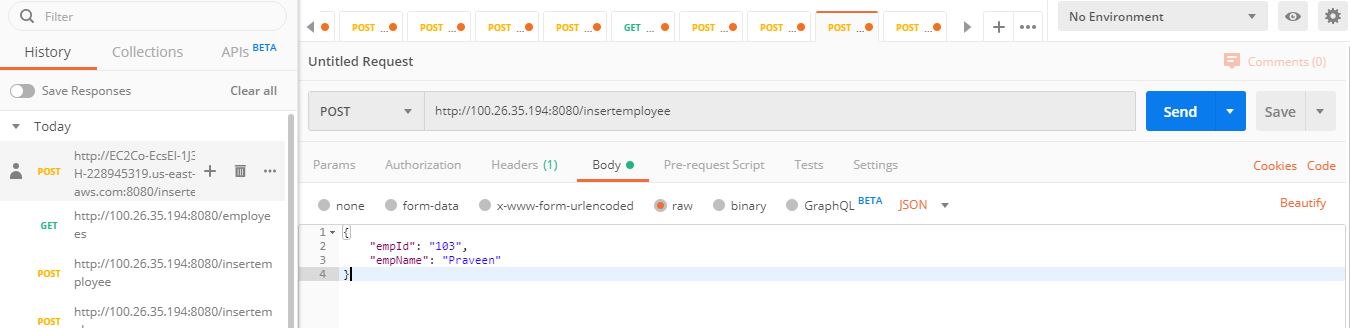


**To check output we need postman app:**

Goto postman:

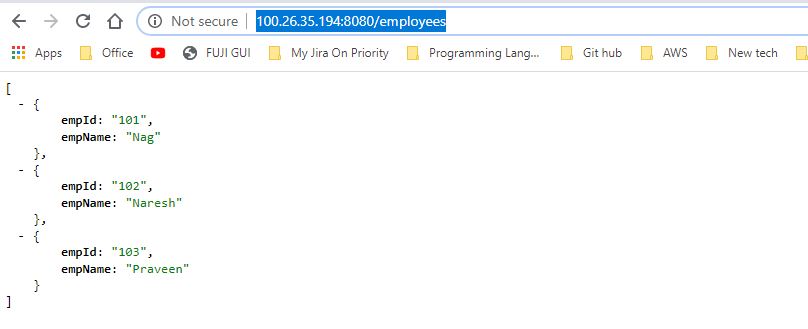
**To call insertemployeeapi:**

1. Fix to Post and give url as <http://100.26.35.194:8080/insertemployee>
2. Click on Body given data in Json format

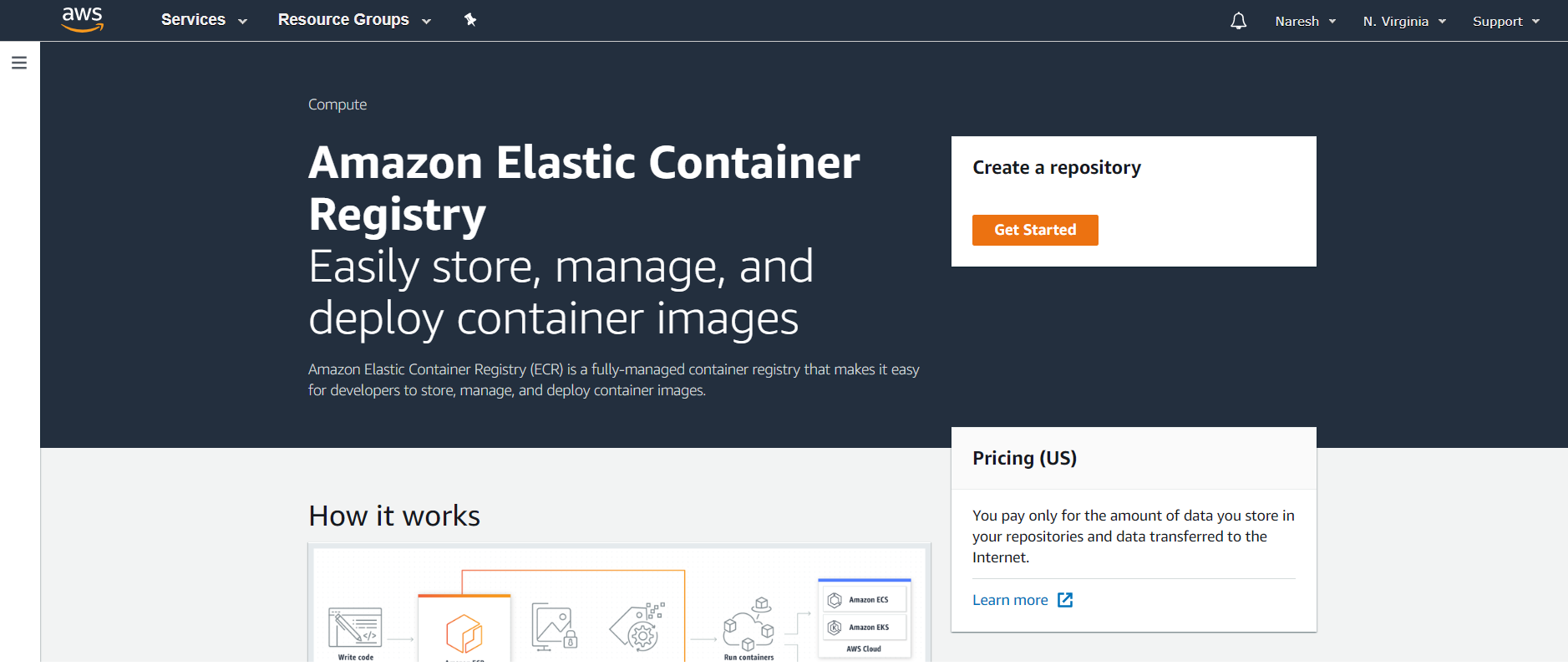


**To call employee api:**

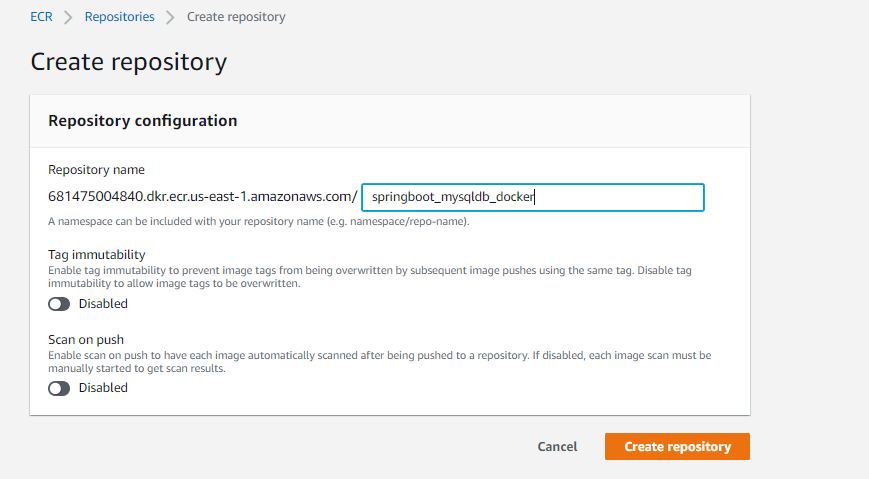
<http://100.26.35.194:8080/employees>



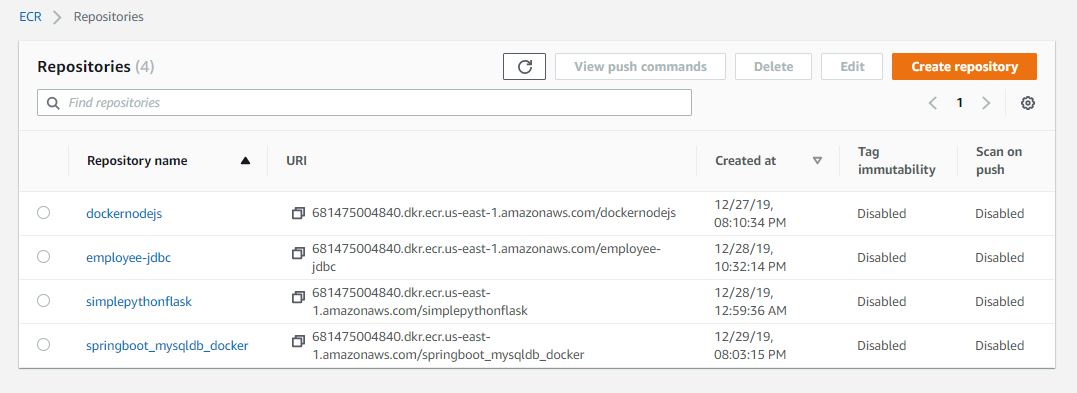
**Creating Repository on ECR using below steps:**



* Click on **Get Started**



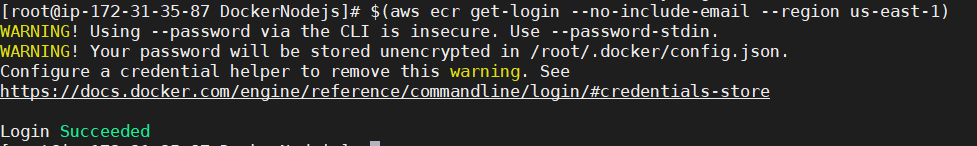
* Give Repository name as “**springboot\_mysqldb\_docker”** and Click on **Create repository**
* Below figure shows created Repository



**Steps to push docker image to ECR:**

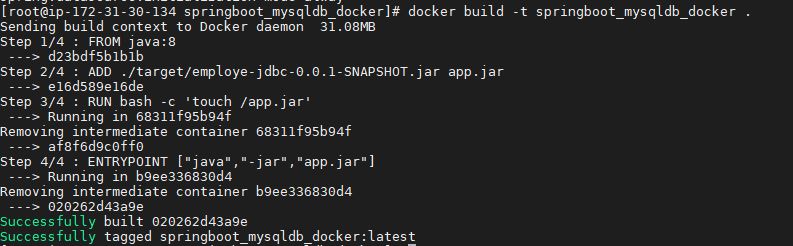
1. Login to aws ECR:

$(awsecr get-login --no-include-email --region us-east-1)



1. Build Docker images:

docker build -t springboot\_mysqldb\_docker.

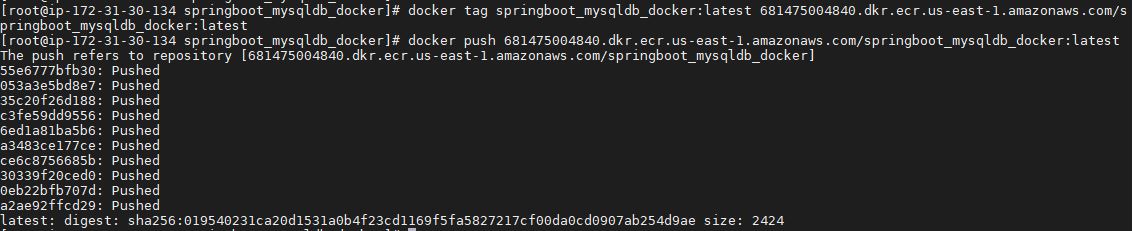


1. Tag Docker image:

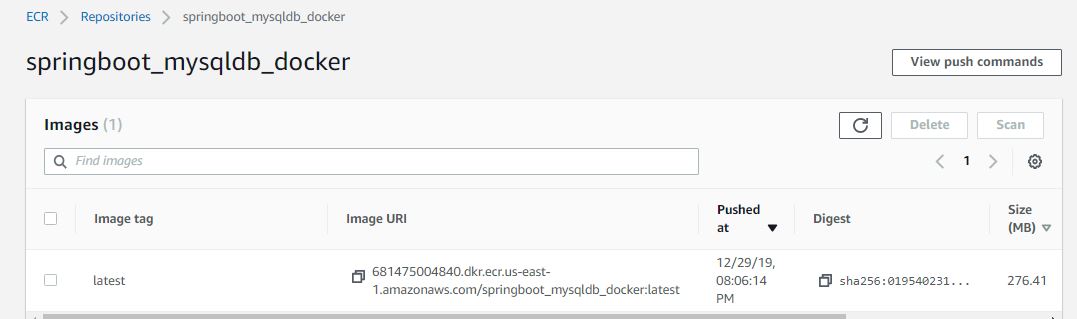
docker tag springboot\_mysqldb\_docker:latest 681475004840.dkr.ecr.us-east-1.amazonaws.com/springboot\_mysqldb\_docker:latest

1. Push Docker image to ECR:

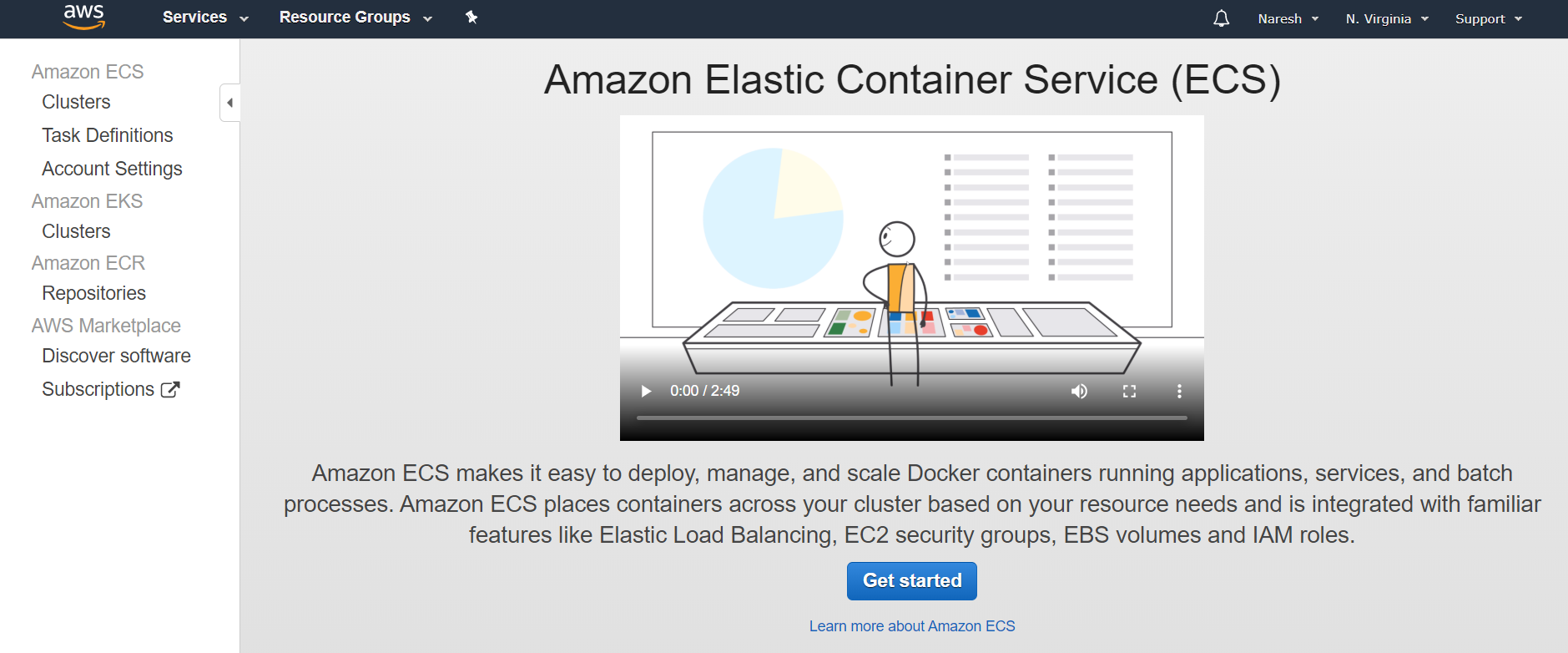
docker push 681475004840.dkr.ecr.us-east-1.amazonaws.com/dockernodejs:latest



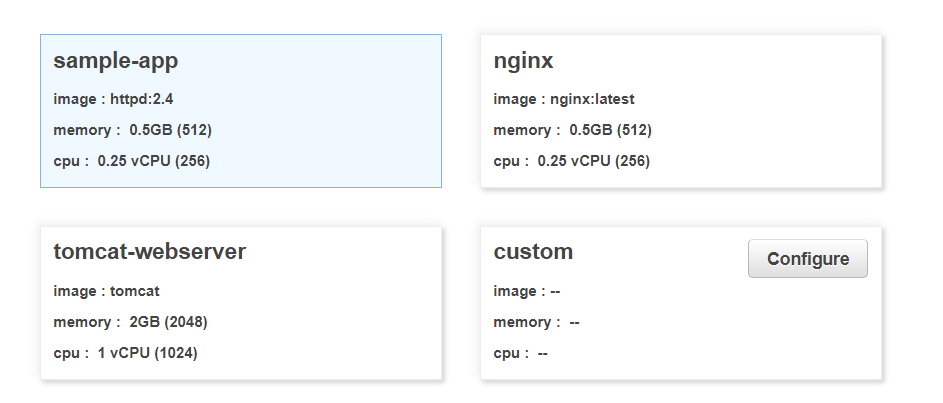
**Finally Repository stored in ECR:**



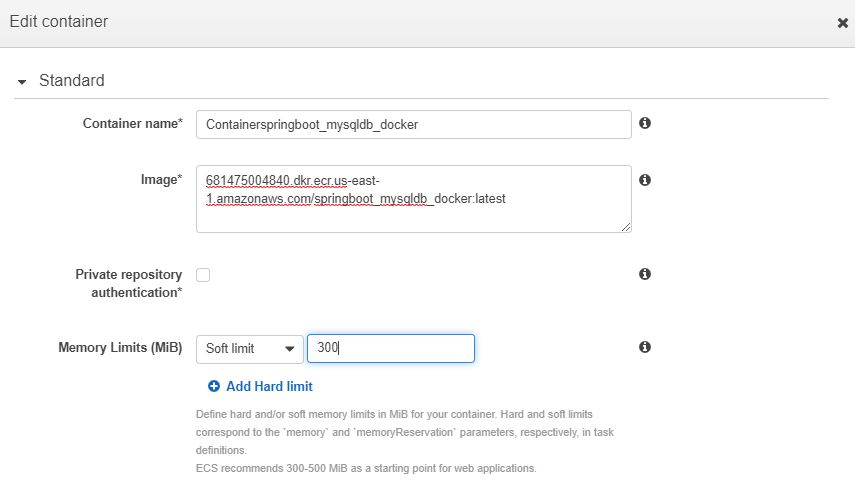
**Open ECS on AWS and Create Cluster using below steps and fill details:**



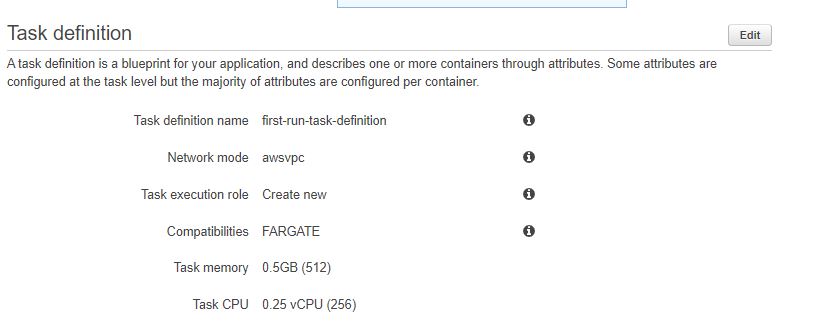
* Click on **Get started**



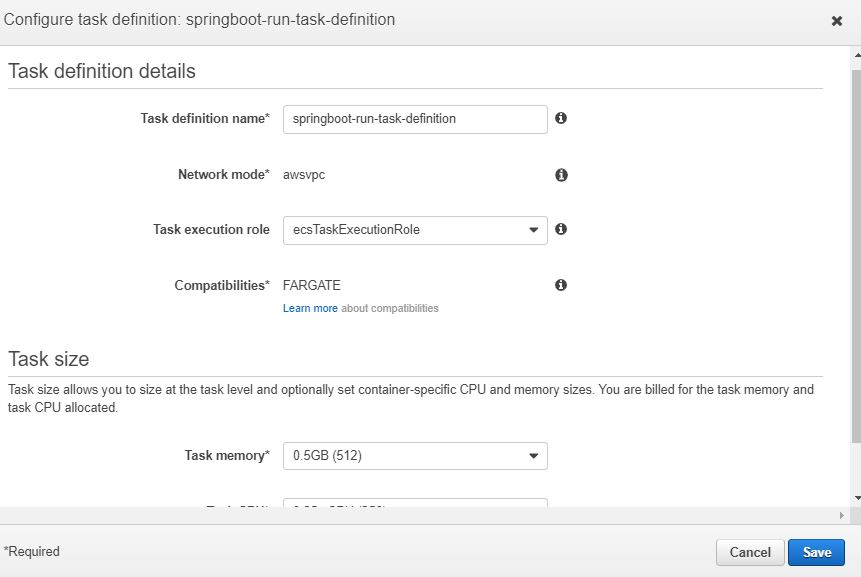
* Click on **Configure**



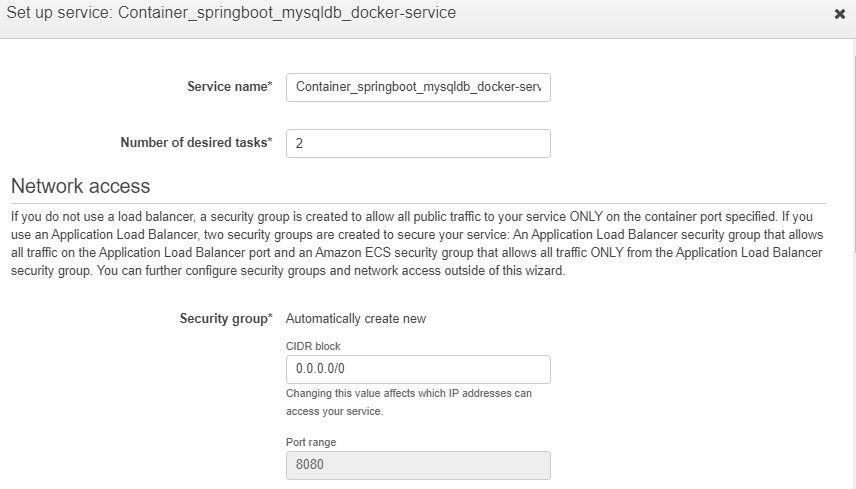
* Click on **update**

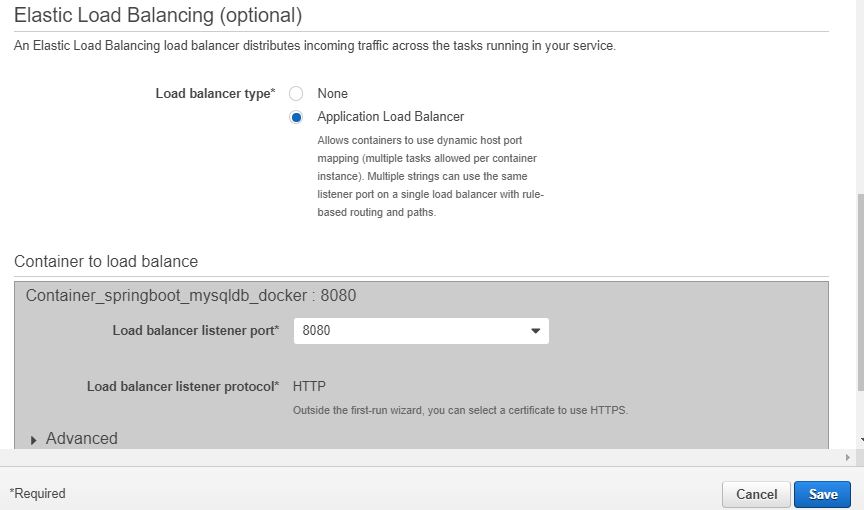


* Click on **Edit** to give Run Task defination details

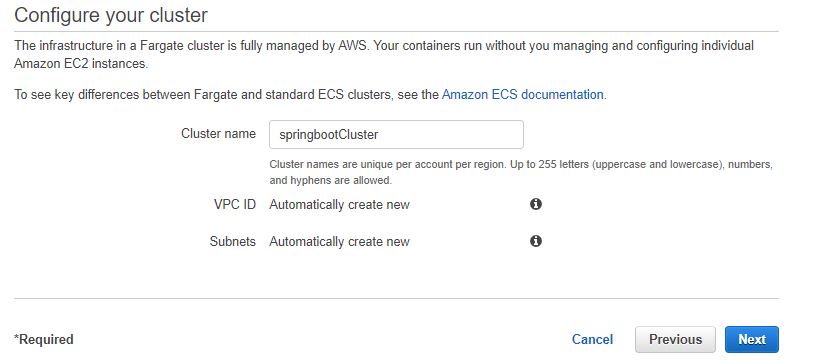


* Click on **save** and click **Next** and Edit details for Container Service



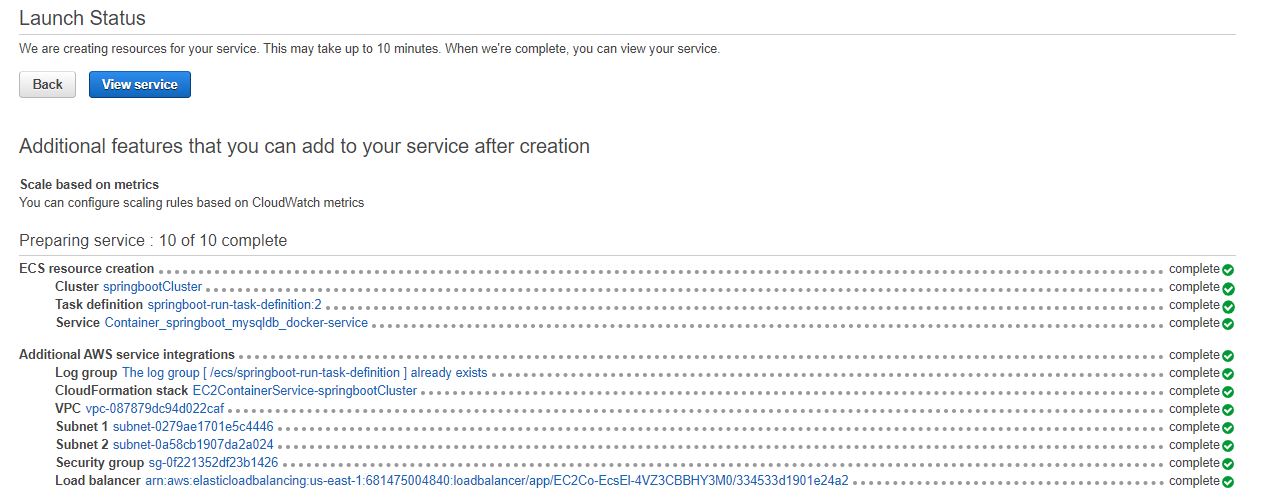


* Click on **save** and click on **Next**

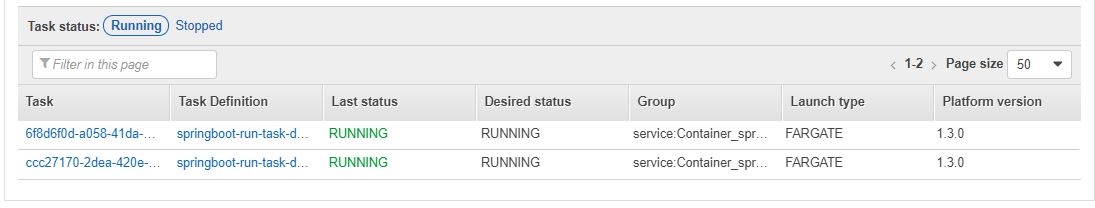


* Click on **Next**
* Click on **Create**

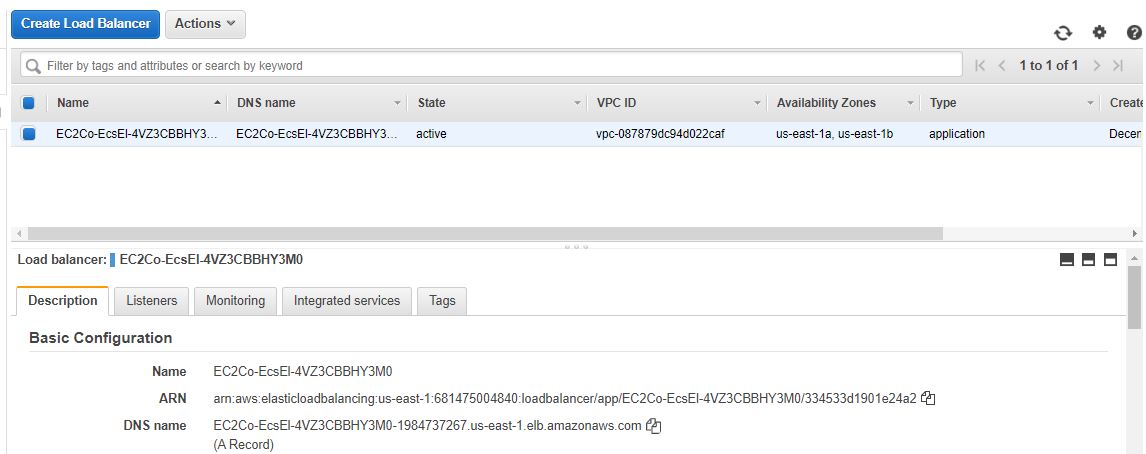
Check below figure shows Cluster succefully created



After Creation of Cluster need to wait untill task status shows in **RUNNING** mode



Check Load Balancer, you will get new ELB

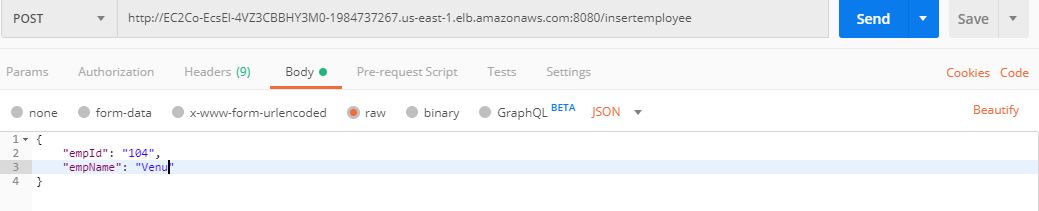


Copy DNS Name and give at GUI with port number and api

<http://EC2Co-EcsEl-4VZ3CBBHY3M0-1984737267.us-east-1.elb.amazonaws.com:8080/insertemployee>

**To call insertemployeeapi:**

1. Fix to Post and give url as above
2. Click on Body given data in Json format



**To call employee api:**

<http://ec2co-ecsel-4vz3cbbhy3m0-1984737267.us-east-1.elb.amazonaws.com:8080/employees>

