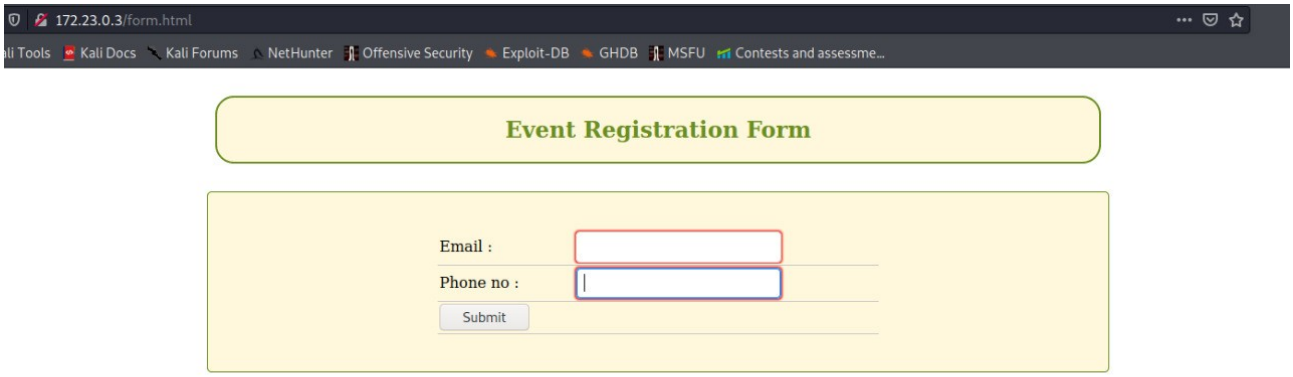
Mid Semester Evaluation Project

Name: Kush Sharma
SAP-ID: 500060182
Subject : SPCM

Steps to deploy dockerimage to dockerhub:-

1. Create Small Web Application with data base attached to it



2. Create a job in Jenkins to make a build of the application

Workflow:-

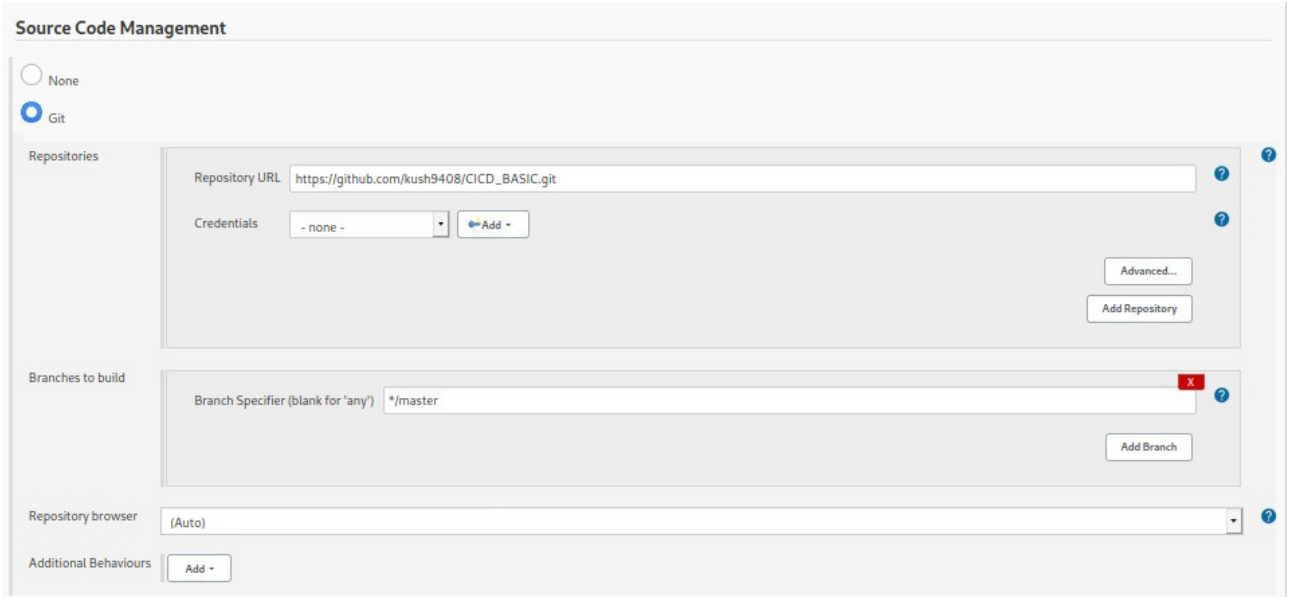
Two jobs have been created
Job 1: This job deals with fetching image from git repository, then building the image and at last deploying the built image to dockerhub repository.

Job 2: This job is triggered if the job 1 is successful.
This job deploys the docker instances on the local machine

JOB 1 (Image-Build)
Docker plugin Used:

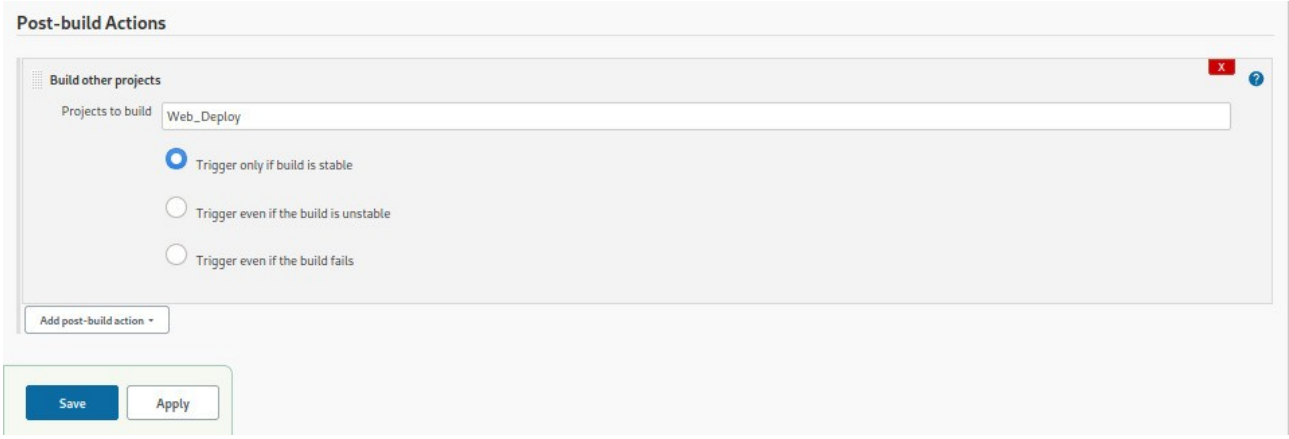
CloudBees Docker Build & Publish Plugin used

1. Git repo to Jenkins to fetch code from:-



2. Using plugin to configure jenkins to dockerhub repository

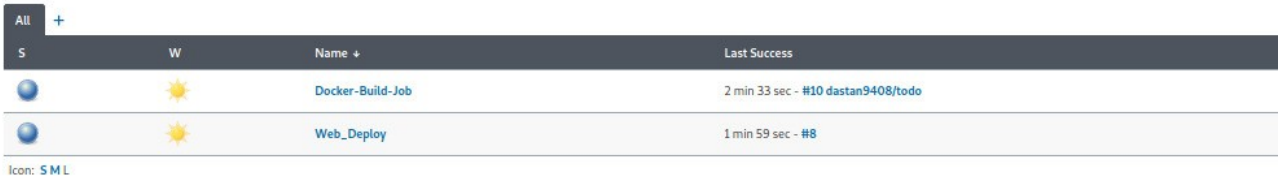
3. Set the post build actions to trigger the deployment build



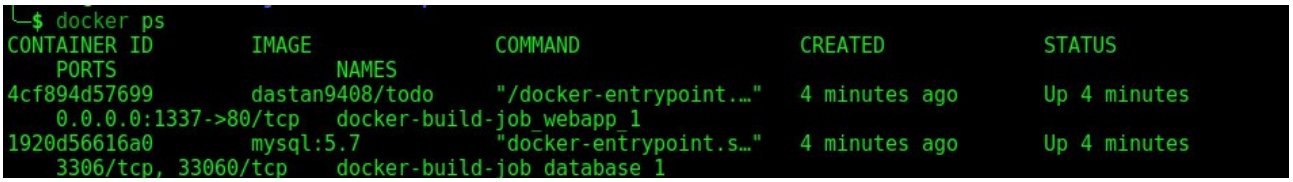
JOB 2 (Image-Deploy)

Using docker-compose to build

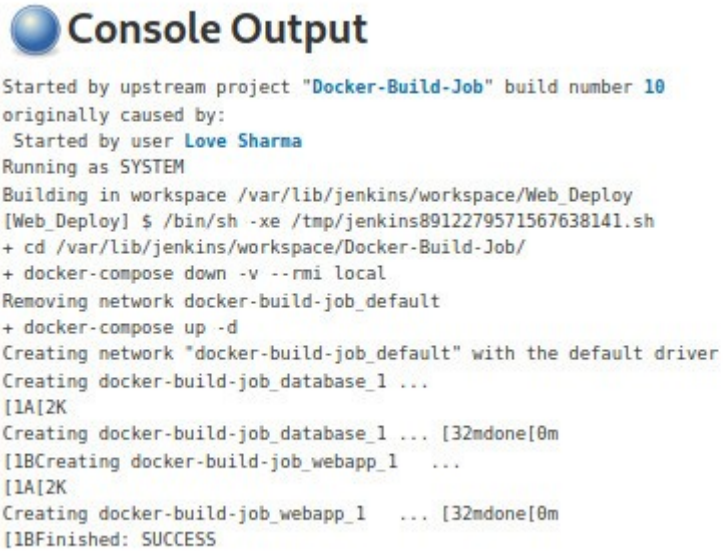
JOBS in Action



Running Locally



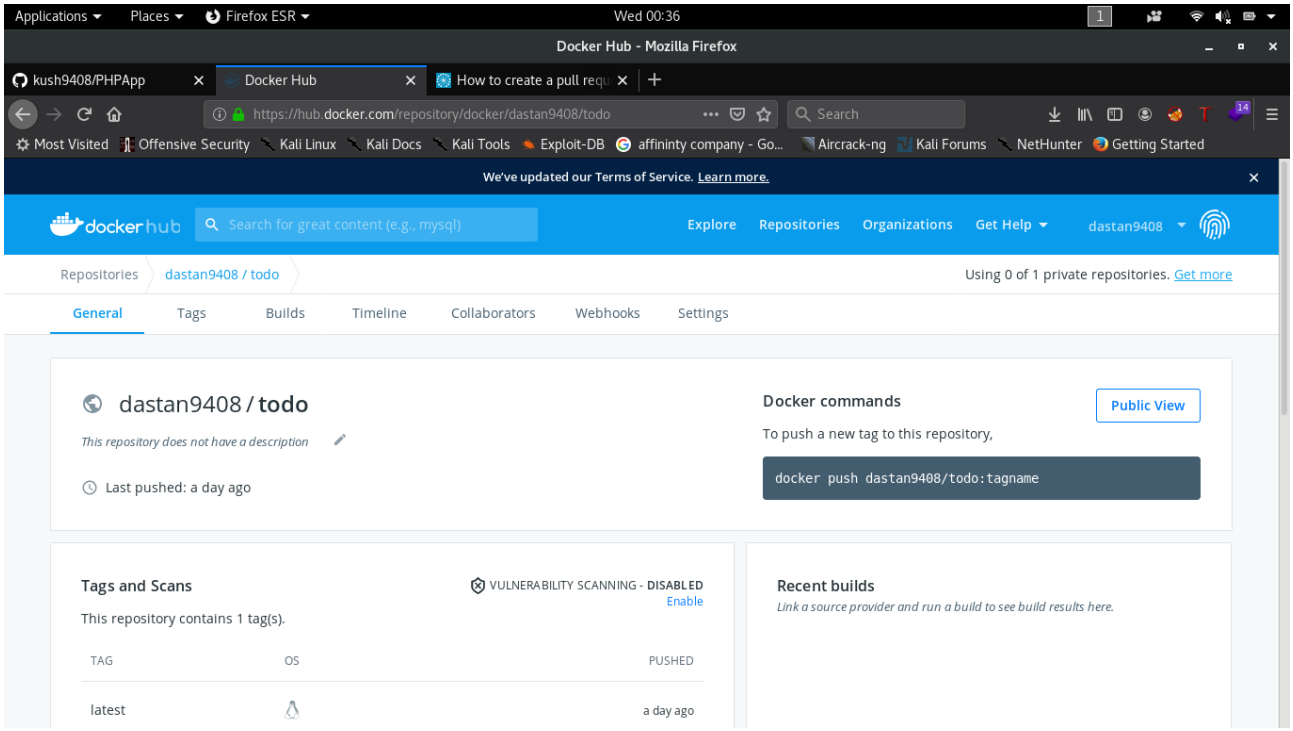
Console Output:-



Console Output

```
Started by user Love Sharma
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/Docker-Build-Job
The recommended git tool is: NONE
No credentials specified
 > git rev-parse --is-inside-work-tree # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/kush9408/CICD_BASIC.git # timeout=10
Fetching upstream changes from https://github.com/kush9408/CICD_BASIC.git
 > git --version # timeout=10
 > git --version # 'git version 2.28.0'
 > git fetch --tags --force --progress -- https://github.com/kush9408/CICD_BASIC.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 97cfd4a9f5fbfe4e3d64b486b0ea16ca053d7430 (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 97cfd4a9f5fbfe4e3d64b486b0ea16ca053d7430 # timeout=10
Commit message: "CICD_BASIC"
First time build. Skipping changelog.
[Docker-Build-Job] $ docker build -t dastan9408/todo --pull=true /var/lib/jenkins/workspace/Docker-Build-Job
Sending build context to Docker daemon  88.06kB

Step 1/4 : FROM phpmyadmin:latest
latest: Pulling from library/phpmyadmin
Digest: sha256:740c0b462c41f47590d004829becbb636de75bc9046c8fce55da7514dd48c6c2
Status: Image is up to date for phpmyadmin:latest
 ---> e837bfe05419
Step 2/4 : RUN apt-get update -y
 ---> Using cache
 ---> d22c2b6af656
Step 3/4 : COPY ./todolist/apache2.conf /etc/apache2/apache2.conf
 ---> Using cache
 ---> c656ee8ceb61
Step 4/4 : COPY ./todolist /var/www/html/
 ---> Using cache
 ---> 2a4cf2feae2b
Successfully built 2a4cf2feae2b
Successfully tagged dastan9408/todo:latest
[Docker-Build-Job] $ docker inspect 2a4cf2feae2b
[Docker-Build-Job] $ docker push dastan9408/todo
The push refers to repository [docker.io/dastan9408/todo]
029557229d76: Preparing
d15e2099db5f: Preparing
f872381531b8: Preparing
7a9b4025edae: Preparing
cfb98416ab94: Preparing
07a08d45e22f: Preparing
f3ece369facc: Preparing
4229aa54e6c2: Preparing
98ef391dc574: Preparing
f9a17ae2f2e7: Preparing
f1b0431d3523: Preparing
c2f378b62ca2: Preparing
47f3deef0d2b: Preparing
918cd028f24b: Preparing
e4a515aa9d7b: Preparing
c1e4b1575eba: Preparing
e037cf30d364: Preparing
710c0bd46e1d: Preparing
b62571eecc86: Preparing
```

Build Finally Pushed to dockerhub



Terraform Script to deploy the image on AWS docker platform (ECS)

```
root@kali:~/Documents/myspace/college/system_provisioning/Event# tree
.
├── docker-compose.yml
├── dockerfile
├── MidSem.odt
├── README.md
├── service.json
├── Terraform
│   ├── main.tf
│   ├── service.json
│   └── variables.tf
├── terraform.tfstate
├── terraform.tfstate.backup
└── todolist
    ├── apache2.conf
    ├── form.html
    ├── insert.php
    └── style.css

2 directories, 14 files
```

Steps in creating the main.tf terraform script:-

1. Initialize the provider to set the API's to be used. After setting "aws" as a provider you would be able to use the AWS resources.

2. Next step is to set up the VPC. This is the place where we would be deploying our ECS cluster.

3. After creating VPC we would configuration.

4. Next step involves configuration of our vpc and subnets.

5. The next step is to implement AWS security groups for incoming traffic and to configure permissions ECS by IAM policies.

6. Configuring the ECS.

Final Script as per the above steps:-

```
provider "aws" {
access_key = ""
secret_key = ""
region = "us-east-1"
}


resource "aws_vpc" "main"{
    cidr_block = "132.0.0.0/16"
    tags = {
        Name=var.vpc_name
    }
}

resource "aws_subnet" "main" {
  count = 2
  vpc_id     = aws_vpc.main.id
  cidr_block = cidrsubnet(aws_vpc.main.cidr_block, 8, count.index)
  map_public_ip_on_launch=true
  tags = {
    Name = var.subnet_name
  }
}

resource "aws_internet_gateway" "internetgateway" {
  vpc_id = aws_vpc.main.id
}
```

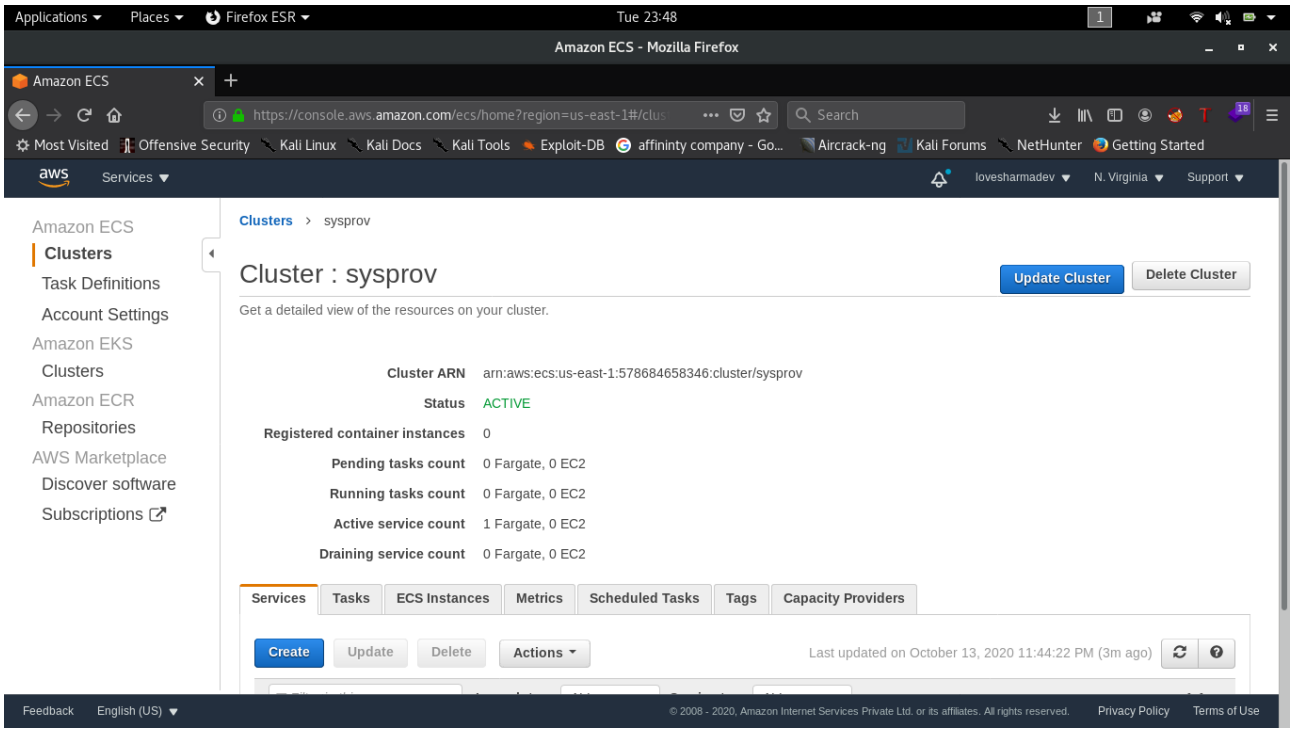
```
resource "aws_route" "internet_access" {
  route_table_id = aws_vpc.main.main_route_table_id
  destination_cidr_block = "0.0.0.0/0"
  gateway_id = aws_internet_gateway.internetgateway.id
}

resource "aws_security_group" "accessgroups" {
  name = "allowinbound"
  vpc_id = aws_vpc.main.id
  ingress {
    cidr_blocks=["0.0.0.0/0"]
    from_port=0
    to_port=65535
    protocol="tcp"
  }
  ingress {
    cidr_blocks=["0.0.0.0/0"]
    from_port=0
    to_port=0
    protocol=-1
  }
  tags = {
    Name = "ECS-Access"
  }

}

data "aws_iam_policy_document" "ecs_task_execution_role" {
  version = "2012-10-17"
  statement {
    sid = ""
    effect = "Allow"
    actions = ["sts:AssumeRole"]

    principals {
      type        = "Service"
      identifiers = ["ecs-tasks.amazonaws.com"]
    }
  }
}

# ECS task execution role
resource "aws_iam_role" "ecs_task_execution_role" {
  name               = "MyEcsTaskExecutionRole"
  assume_role_policy = data.aws_iam_policy_document.ecs_task_execution_role.json
}

# ECS task execution role policy attachment
resource "aws_iam_role_policy_attachment" "ecs_task_execution_role" {
  role       = aws_iam_role.ecs_task_execution_role.name
  policy_arn = "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
}

resource "aws_ecs_cluster" "nodecluster" {
  name = "sysprov"
}

resource "aws_ecs_task_definition" "PHPApp" {
  family                = "service"
  container_definitions = file("service.json")
  execution_role_arn=aws_iam_role.ecs_task_execution_role.arn
  network_mode="awsvpc"
  requires_compatibilities=["FARGATE"]
```

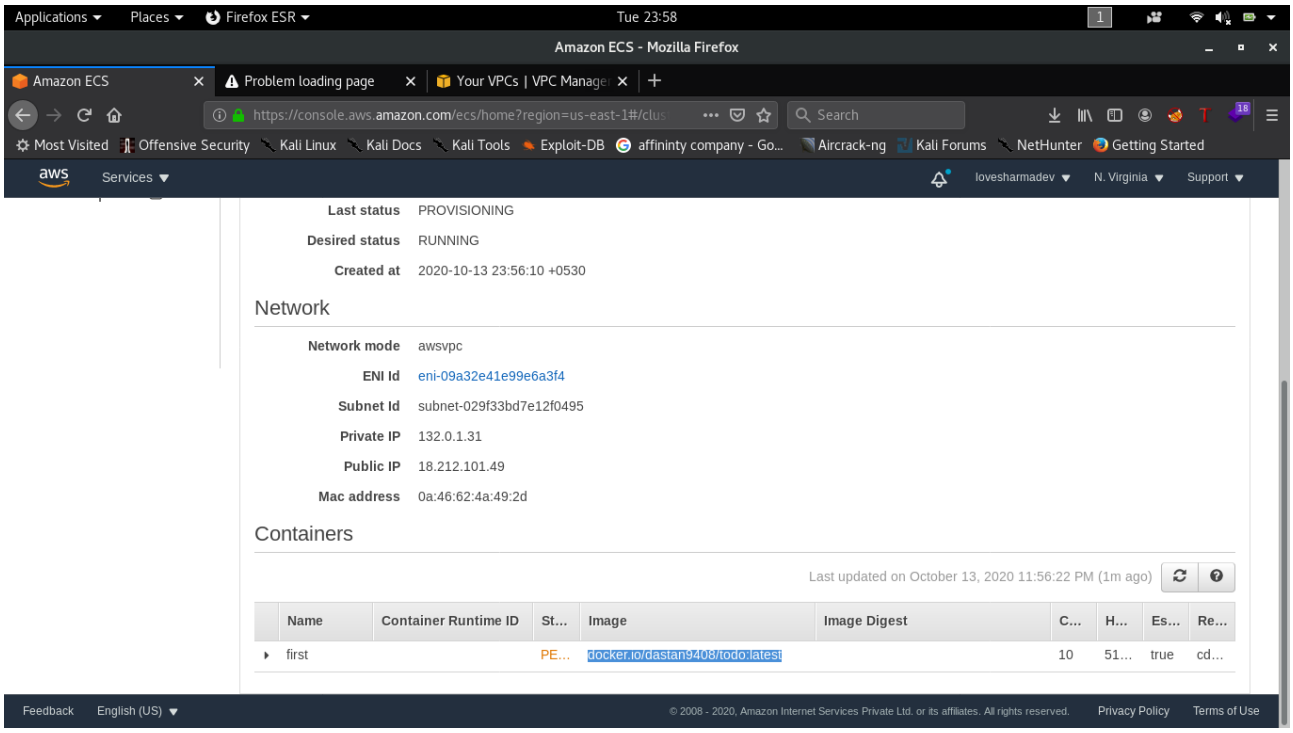
```
  memory="1024"
  cpu="512"
}

resource "aws_ecs_service" "main" {
  name = "service-ecs"
  cluster = aws_ecs_cluster.nodecluster.name
  task_definition = aws_ecs_task_definition.PHPApp.arn
  launch_type = "FARGATE"
  network_configuration {
    security_groups = [aws_security_group.accessgroups.id]
    subnets = aws_subnet.main.*.id
    assign_public_ip = true
  }
  depends_on=[aws_iam_role_policy_attachment.ecs_task_execution_role]
}
```

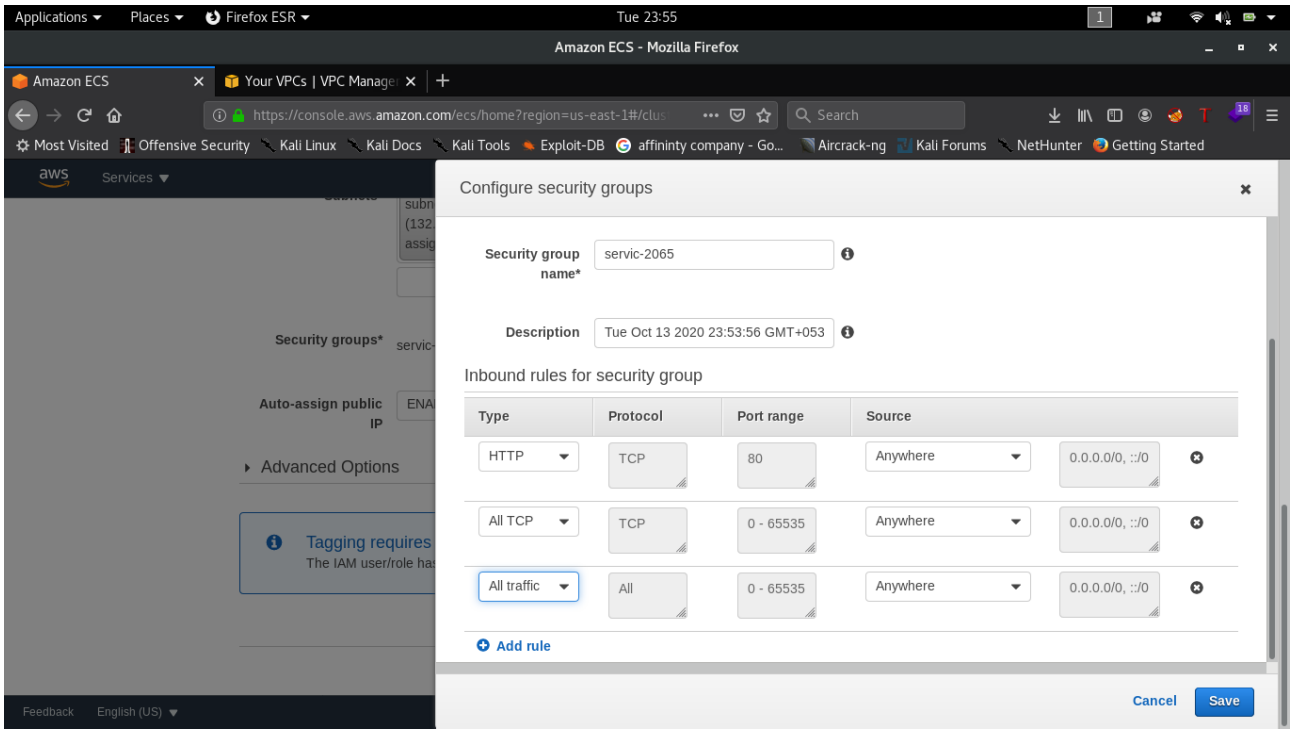
After Creating the above, run the following commands:-

1. terraform init
2. terraform plan
3. terraform apply

Now create task and run it on ECS




Applications
Places
Firefox ESR
Tue 23:58
Amazon ECS - Mozilla Firefox
Amazon ECS
Problem loading page
Your VPCs | VPC Manage
https://console.aws.amazon.com/ecs/home?region=us-east-1#/clus
Search
Most Visited
Offensive Security
Kali Linux
Kali Docs
Kali Tools
Exploit-DB
affininty company - Go...
Aircrack-ng
Kali Forums
NetHunter
Getting Started
aws
Services
lovesharmadev
N. Virginia
Support
Last status PROVISIONING
Desired status RUNNING
Created at 2020-10-13 23:56:10 +0530
Network
Network mode awsvpc
ENI Id eni-09a32e41e99e6a3f4
Subnet Id subnet-029f33bd7e12f0495
Private IP 132.0.1.31
Public IP 18.212.101.49
Mac address 0a:46:62:4a:49:2d
Containers
Last updated on October 13, 2020 11:56:22 PM (1m ago)
Name
Container Runtime ID
St...
Image
Image Digest
C...
H...
Es...
Re...
first
PE...
docker.io/dastan94d8/todo:latest
10
51...
true
cd...
Feedback
English (US)
© 2008 - 2020, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use


Applications
Places
Firefox ESR
Tue 23:55
Amazon ECS - Mozilla Firefox
Amazon ECS
Your VPCs | VPC Manage
https://console.aws.amazon.com/ecs/home?region=us-east-1#/clus
Search
Most Visited
Offensive Security
Kali Linux
Kali Docs
Kali Tools
Exploit-DB
affininty company - Go...
Aircrack-ng
Kali Forums
NetHunter
Getting Started
aws
Services
Security groups*
Auto-assign public IP
Advanced Options
Tagging requires
The IAM user/role has
Configure security groups
Security group name*
servic-2065
Description
Tue Oct 13 2020 23:53:56 GMT+053
Inbound rules for security group
Type
Protocol
Port range
Source
HTTP
TCP
80
Anywhere
0.0.0.0/0, ::/0
All TCP
TCP
0 - 65535
Anywhere
0.0.0.0/0, ::/0
All traffic
All
0 - 65535
Anywhere
0.0.0.0/0, ::/0
Add rule
Cancel
Save
Feedback
English (US)

After creating tasks and making configurations, you would be provided with a Public IP address.

On accessing it, we would get our final output