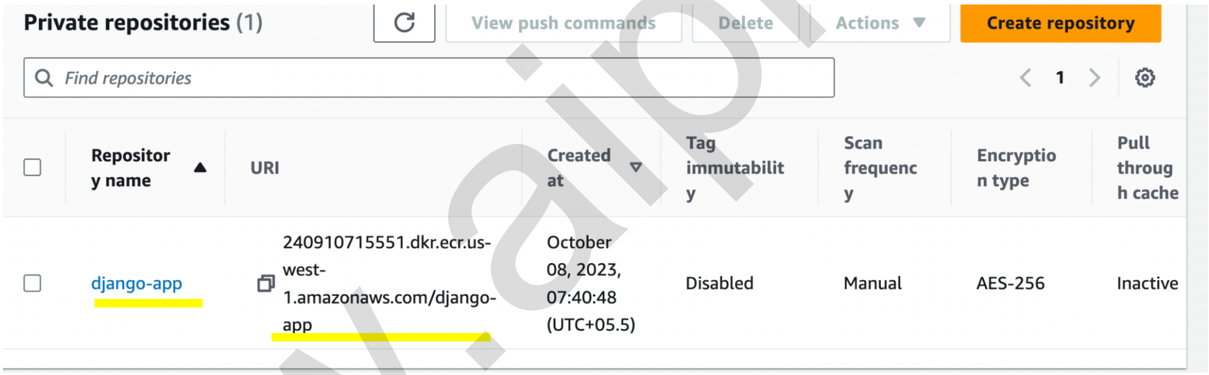
AWS INFRA CREATION WITH PYTHON DJANGO APP DEPLOYMENT [LOCAL]

Step1 - Clone the repository and set the aws configure in local terminal

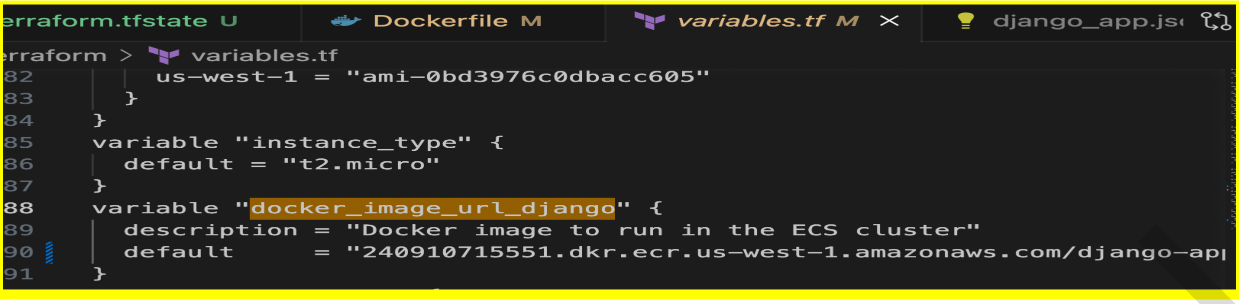
https://github.com/praveen1994dec/Terraform_withcontainers.git

Step2 - Create a repo in aws ECR with name - > django-app



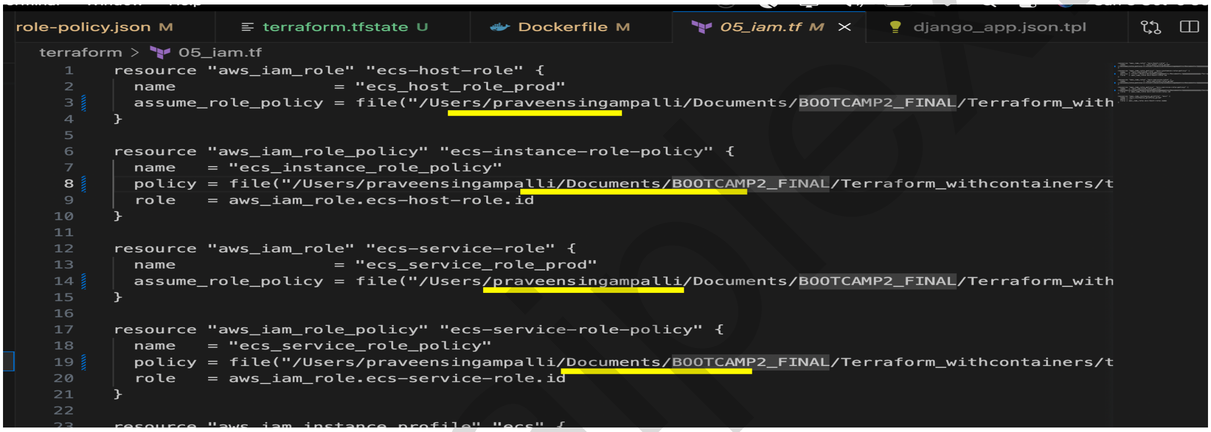
□	Repository name ▲	URI	Created at ▽	Tag immutability	Scan frequency	Encryption type	Pull through cache
□	django-app	240910715551.dkr.ecr.us-west-1.amazonaws.com/django-app	October 08, 2023, 07:40:48 (UTC+05.5)	Disabled	Manual	AES-256	Inactive

Step3 - Once the repo is created copy the repo url and add it in the variables.tf file in place of variable docker_image_url_django



```
    us-west-1 = "ami-0bd3976c0dbacc605"
  }
}
variable "instance_type" {
  default = "t2.micro"
}
variable "docker_image_url_django" {
  description = "Docker image to run in the ECS cluster"
  default     = "240910715551.dkr.ecr.us-west-1.amazonaws.com/django-ap
}
```

Step4 - Change the policy file paths in iam.tf and variables.tf file



```
resource "aws_iam_role" "ecs-host-role" {
  name               = "ecs_host_role_prod"
  assume_role_policy = file("/Users/praveensingampalli/Documents/BOOTCAMP2_FINAL/Terraform_with
}

resource "aws_iam_role_policy" "ecs-instance-role-policy" {
  name   = "ecs_instance_role_policy"
  policy = file("/Users/praveensingampalli/Documents/BOOTCAMP2_FINAL/Terraform_withcontainers/t
  role   = aws_iam_role.ecs-host-role.id
}

resource "aws_iam_role" "ecs-service-role" {
  name               = "ecs_service_role_prod"
  assume_role_policy = file("/Users/praveensingampalli/Documents/BOOTCAMP2_FINAL/Terraform_with
}

resource "aws_iam_role_policy" "ecs-service-role-policy" {
  name   = "ecs_service_role_policy"
  policy = file("/Users/praveensingampalli/Documents/BOOTCAMP2_FINAL/Terraform_withcontainers/t
  role   = aws_iam_role.ecs-service-role.id
}

resource "aws_iam_instance_profile" "ecs" {
```

Step5 - Login into ECR by hitting the below command in terminal local

aws ecr get-login-password --region us-west-1 | docker login --username AWS --password-stdin <YOUR ECR REPO URL>

Step6 - Go to app folder

cd app/

Step7 - Build the docker image

docker build --platform=linux/amd64 -t <URL ECR REPO URL>:latest .

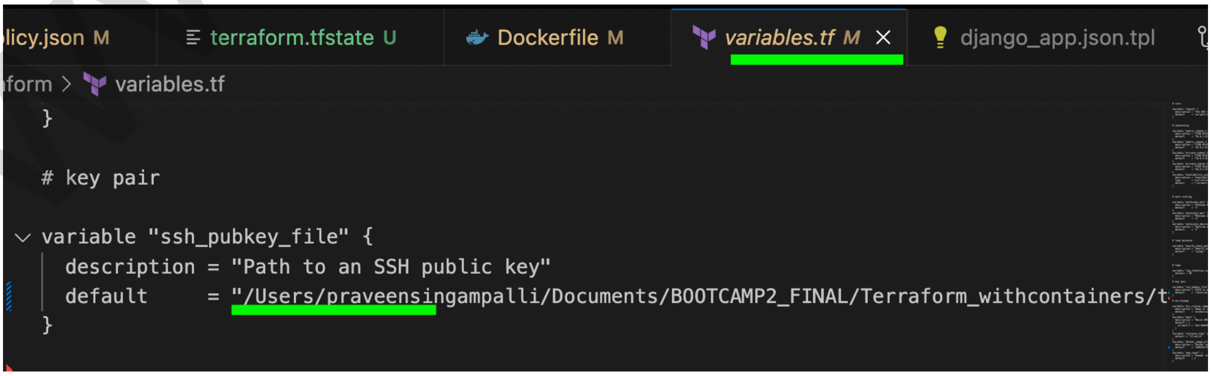
Step8 - Push the image to ECR

docker push <URL ECR REPO URL>:latest

Step9 - Go to terraform folder and hit this below command to create the key pair

ssh-keygen -f california-region-key-pair

Step10 - Add the above file key file path in variable.tf



```
}

# key pair

variable "ssh_pubkey_file" {
  description = "Path to an SSH public key"
  default     = "/Users/praveensingampalli/Documents/BOOTCAMP2_FINAL/Terraform_withcontainers/t
}
```

Step 11 - Hit the below commands in terraform folder

```
terraform init

terraform plan -out terraform.out

terraform apply "terraform.out"
```

Step12 - Install Python BOTO3

```
pip install boto3 click
```

Step13 - Export the AWS access/Secret key and region

```
export AWS_ACCESS_KEY_ID=""

export AWS_SECRET_ACCESS_KEY=""

export AWS_DEFAULT_REGION="us-west-1"
```

Step14 - Go to deploy folder

```
python3 update-ecs.py --cluster=production-cluster
--service=production-service
```

Step15 - Validate the ECS service and there should be 0 Tasks pending in the dashboard

Go to cloudwatch/VPC and check the data and logs stream

Step16 - > Go to EC2-> Loadbanacer -> Copy the DNS name -> Hit the below url in the browser

<DNS_NAME>/ping/

Step17 - Hit the command and delete the architecture

terraform destroy