**Task Practical Implementation**

Task:

1. Please create an automation using Terraform or Cloudformation to provision one Linux VM in AWS capable to run docker containers and with the port 80 exposed to the Internet.
2. Then create a Dockerfile of a nginx container.
3. Then create a CI/CD pipeline using Travis/Github Actions/Jenkins to build and deploy the container in the Linux VM.
4. Then create a README with all necessary steps in order to run your automation. We would like to see your considerations for credential handling in this scenario

----------------------------------------------------------------------------------------------------

Submission of Task includes:

* Terraform Code
* Nginx Dockerfile
* Pipeline Jenkinsfile
* Test Web code
* README containing step by step implementation.

----------------------------------------------------------------------------------------------------

Step By Step Practical Implementation of Task:

Solution Consists these steps:

* Set Terraform Variables and apply Terraform code
* Set Jenkins Agent
* Upload Dockerfile and web code on github and set trigger to run Jenkins seedjob automatically
* Create Jenkins pipeline seedjob using Jenkinsfile

NOTE: I am considering Aws CLI is configured ( for Terraform provider authentication), and company have already configured public Jenkins server.

Step 1: Terraform Code

1. As Terraform code is already provided, so, you need to set varable in Terraform.tfvars file in Terraform code directory.
2. Available Varaibles are :
3. **aws\_region**  [ default region = ap-south-1 ] : AWS Region where instance would be launch.
4. **vpcid** ( required) : Aws VPC Id. i.e., vpc-8e743434
5. **subnetid** ( required) : Aws VPC Subnet Id i.e., subnet-8e743434
6. **i\_type** [ default value = t2.micro ] : Instance Type . i.e., t2.micro.
7. **key\_name** [ default value = mytfkey] : Name of key which is attached to the instance. i.e., mykey
8. **image** ( required) : Amazon Machine Image (AMI) id. i.e.,
9. **profile** [ default value = default ] : Authentication Profile for Terraform. (can be set using Aws CLI).

**NOTE:**

# for Profile based credential  ( By Default Profile Based Authentication is used )

#profile="myprofile"

#shared\_credentials\_file = "/Users/tf\_user/.aws/creds"     # IF credential is not at default path, set path using this option

# for Manual based credential  EXPORT Credential As Environment Variable

# export AWS\_ACCESS\_KEY\_ID="anaccesskey"

# export AWS\_SECRET\_ACCESS\_KEY="asecretkey"

# Note: Remove profile keyword in provider.tf, if you are using manual way.

1. As variables are set, now initialize Terraform by command : **terraform init**
2. Now check Terraform plan using command: **terraform plan.**
3. Now apply Terraform plan using commad: **Terraform apply.**

Now EC2 Instance launched with docker and post 80 is exposed.

NOTE: JAVA and git are also installed for Jenkins pipeline and Jenkins agent purpose.

STEP 2: Set Jenkins Agent

1. Go to Jenkins and login.
2. At Dasboard. Goto Manage Jenkins >> Manage Node and Clouds >> New Node
3. Configure Node (agent0
4. Set **Node Name** and **select permanent node**
5. Set **description, number of executors** and **remote root directory**
6. Select Usage : **Only Build jobs with label matching the node**
7. Select launch Method : **Launch agents via SSH**
8. Set instance ip as Host and add ssh credentials with private key. ( Note: Copy private key from terraform workspace from key “\*.pem”. )
9. Select **Nonverifying Verification strategy** and save configuration.
10. Jenkins Agent is successfully configured.

Step3: Add Web code and Dockerfile on github

1. As code, Jenkinsfile and Dockerfile is already there, so create a git repo and commit the code.
2. Create repository over github and add it to local git as remote repository.
3. Push the commited code.

Step4: Set Jenkins Pipeline

1. Goto Jenkins Dashboard and add new item.
2. Type pipeline **name** and **select pipeline**. ( **Required Jenkins Plugins are: Docker plugin, Github plugin, Pipeline, SSH server and SSH agent plugin** )
3. Configure Pipeline
4. Add **description, Github project url**.
5. Select **GitHub hook trigger for GITScm polling**.
6. At pipeline option, select **Pipeline scipt from SCM** and set github url and branch there.
7. Save the pipeline.
8. Set the github webhook.
9. Change Agent Name, github\_url and environment variables as per need in Jenkinsfile at your local workspace and push that code on github.
10. **Manual Build first build of Pipeline. IMPORTANT**
11. Now again change web code and push again, this time pipeline trigger itself.

Everything is configured successfully.