# 

# ######DEPLOYMENT PROCESS##########

docker build . ---> Image is created

docker tag <image id > hostname/project-id/image-name [Ex:docker tag 842b4da9dd2a gcr.io/devops-project-376507/microstock-image]generated from above comand>

docker push hostname/project-id/image[ex:docker push gcr.io/citric-celerity-374512/storefront-image] name--> image is pushed into container registry

Now in vm

docker pull hostname/project-id/image-name

docker run -d -p 8082:8082 <image id>

curl http://localhost:8082/

##############################################

docker container ls/docker ps -a

docker rm -f <container id>

docker logs <container id>--->to check logs

gcloud compute instances set-service-account demo-vm --scopes=storage-rw--->IN order to give permission to vm to access container registry.

# ####Mysql INSTALLATION IN VM########

https://hevodata.com/learn/docker-mysql/ ====>>>Link follwed for installation

mysql --host=13.233.35.88 --port=13306 -u root -p ====>it will open mysql

GRANT ALL PRIVILEGES ON \*.\* to root@'%' IDENTIFIED BY 'root';

FLUSH PRIVILEGES;

SELECT host FROM mysql.user WHERE User = 'root';

CREATE USER 'root'@'%' IDENTIFIED BY 'root';

we have to first set firewall rules for port 13306

sudo systemctl stop mysql

sudo apt purge mysql\*

In Mysql we created DBs.

sudo docker pull mysql/mysql-server:latest

sudo docker run --name=mysqlContainer -d mysql/mysql-server:latest

mysql> ALTER USER 'root'@'localhost' IDENTIFIED BY 'root';

sudo mkdir -p /root/docker/mysqlContainer/conf.d

sudo nano /root/docker/mysqlContainer/conf.d/my-custom.cnf

sudo apt install mysql-client

apt-cache search mysql-server

2JG@.I8874U3CB9M,Q&.u7rYy.,uH@aU

# #####POSTGRES SQL SETUP IN VM########

https://cloud.google.com/community/tutorials/setting-up-postgres ====>link followed

Frst we have to create firewall rule 5432

# IPv4 remote connections for the tutorial:

host    all             all          0.0.0.0/0         md5 ===>>we have to set this environment variables in file.

Postgres passwrd:postgres123

username:postgres

# #### INSTALL DOCKER IN VM####

https://runnable.com/docker/install-docker-on-linux ===> LInk followed

sudo apt-get install docker.io

sudo groupadd dockernm

sudo usermod -aG docker ${USER}

docker run hello-world



# ##### ####Git #######

we created folders named repo and main repo.

In main repo :

git init

git clone <repo url>==> codes extracted from main branch

In order to push code into gcp source repo[composable\_E-commerce]

In cloud sdk we will open repo folder:

gcloud init

git pull https://source.developers.google.com/p/citric-celerity-374512/r/composable\_E-commerce

cd composable\_E-commerce

we have to copy all the contents of main repo folder in composable\_E-commerce folder in repo.

git add .

git commit -m "msg"

git push -u origin

# ###### Jenkins ########

sudo usermod -a -G docker jenkins -🡪 to add jenkins in docker

docker pull jenkins/jenkins

docker run --name jenkins --rm -u root -p 8081:8080 -d -p 50000:50000 -v $(which docker):/usr/bin/docker -v $HOME/.jenkins/:/var/jenkins\_home -v /var/run/docker.sock:/var/run/docker.sock docker.io/jenkins/jenkins:latest

docker run -d -p 8080:8080 -p 50000:50000 -v jenkins\_home:/var/jenkins\_home docker.io/jenkins/jenkins:latest

docker run -d --rm --name**=**agent1 -p 22:22

-e "JENKINS\_AGENT\_SSH\_PUBKEY= ssh-rsa  shalu\_a\_gupta@agent-vm "

jenkins/ssh-agent:alpine

sudo mkdir jenkins

sudo chmod 777 jenkins

sudo chown -R jenkins:jenkins /var/lib/jenkins

sudo service jenkins restart

sudo su -s /bin/bash Jenkins

cat /etc/passwd | grep jenkins

docker run -d --rm --name**=**agent1 -p 22:22 \

-e "JENKINS\_AGENT\_SSH\_PUBKEY=ssh-rsa  shalu\_a\_gupta@demo-vm" \

jenkins/ssh-agent:alpine

<https://stackoverflow.com/questions/73110198/jenkins-error-buildind-docker-lib-x86-64-linux-gnu-libc-so-6-version-glibc/73312606#73312606>

docker run -p 8000:8080 -p 50000:50000 -d -v /var/run/docker.sock:/var/run/docker.sock -v jenkins\_home:/var/jenkins\_home jenkins/jenkins:lts

docker logs <id>

http://13.233.35.88:8080/

26d02fb0a88b4e0e8fc8a075f3f2e44b

docker exec -it -u0 de4fdce3073b bash

curl https://get.docker.com > dockerinstall && chmod 777 dockerinstall && ./dockerinstall

Ecommerce

Ecommerce@123

jXHa\_tCHA-m\*2yGiyg5a

AWS Hostname: [admin@ec2-13-233-35-88.ap-south-1.compute.amazonaws.com](mailto:admin@ec2-13-233-35-88.ap-south-1.compute.amazonaws.com)

<https://asf.alaska.edu/how-to/data-recipes/moving-files-into-and-out-of-an-aws-ec2-instance-windows/> --🡪 to upload files in AWS vm from local

gcloud auth login

gcloud auth activate-service-account devops-project-376507@appspot.gserviceaccount.com --key-file=devops-project-376507-44d84d6873d3.json

gcloud auth configure-docker

//Jenkins

<http://13.233.35.88:8000/>

//storefront-nextjs

<http://13.233.35.88:3001/>

//microreview

<http://13.233.35.88:8080/>

//microprice

<http://13.233.35.88:8082/>

//microstock

<http://13.233.35.88:8081/>

//backoffice

<http://13.233.35.88/>ok

//b2borg

<http://13.233.35.88:9090/>

git push origin feature/storefront

git pull origin feature/storefront

git add storefront-nextjs/Jenkinsfile

http://13.233.35.88:8080/github-webhook/

1152822d9ded49c9316defcedf0c4bbed1 🡪 jenkin api token new vm

114167fab0b9858d1428d7f0dde8ca437e

11cd7457113f31f259a249e9adf7ffc803 🡪jenkin api token old vm

ghp\_lGZ6v67SVAHSRqtbNg4m1vLkOQkqmh4DXA86 🡪git hub token for name gitserver

1. Add SonarQuabe in CI/CD pipeline.
2. SonarQuabe integration with GitHub.
3. Deploy serverless functions on VM using Serverless Framework to avoid cloud dependency.

**docker run** -d --name sonarqube -p 9090:9000 -p 9092:9092 **sonarqube**

<http://13.233.35.88:9090/>

<https://docs.sonarqube.org/latest/devops-platform-integration/github-integration/>

-----BEGIN RSA PRIVATE KEY-----

MIIEowIBAAKCAQEApfD+ivRbuoRTFC8/WuoZ3iNAwJHWrOl3yj+TNlZ9+fO73Xd+

Efuv887Nhwq9RcWCEF4XQMFWY0SyIO7styKj9TyFS7fVFOWhDkbfJjiO2mWIaD9j

wDKAODP70+4sHtx1aSfoRDzOKUpRjS5GpQftVuOwNjgEA73mjylqO1JxeTU6Lwup

XzIooi9C68PCF+hniEx3YNAlVGT9dei0jnEkh8jEkBJtcS+qNAxN8uN0FyJAnrY/

k0e8AKBgKazZ7LhOUs/hNeL71AEfp3sHYIvycR4CNbQGkMi1OZN7xT0u3+/kGrhe

75aLwTWn8Z6B7w3+hfO/EioDP3bUgvGYaBlbFQIDAQABAoIBAEOl7E7/cQmneirn

W/+XIfaCDpqymX6iKtiDdGMZfX/RUgvl87onDcLp0rWzthdkfjJn+TjH2Izda3Sr

kkrST17hTkHg7nf3aOHmGIB/XBVCgQpmlSa8L7M51fa+yLPIEgFNaI2w8Ca7CmXw

sJIEmkfqyGJCEbjoyL4iFxGZ+XJ8jT3pKehWXFexqF4mRy5Qp0QB9M1BJOZ9eKN0

JjM6D6JCjVZTqWpB0AiLufpEfnWxouLTY3L91InAyJPrP5sBlVAsu1Noww0q33cE

Dj3StOcycqtqCnZnMkEuI2t1J//6kLDCsVtC+eM8qF6bhSU92cDIayt+vt8hz1DO

fZc2nkECgYEA2FKZnVfWIqScarVqJ7VhIVEhRAjk0kMkRT9J92sM/KzvbFD8/FWN

96CHsP+gwIP6Ld2n5LP7coNCJLyu7j1Y5X7UOQ23OWX7k5GtvYg57ZZyRTXNDV1l

GPGQCPSByVmZE8wQH/++pvrouVI2cAXoPxOBU5aRYRzCRrdfNFUjYNECgYEAxGC/

qYgNNKTzUwQWMO4jdDPNqeroBGm4FIufjn1pKxmeS5SN79bP6InEFCPE7o8t2uzr

YNt43dwtlOKmXZyPxC8xXkdtIeoKhhQQBiZifi5p1cyG28LGsNS15XUOa2M5vs68

J+vr0CIUmB/Lr5fOfrcUn3mfh1/cMKJJBLVGxwUCgYAtxZSBaXOC0wfRwF8c1dEe

hpceWS9pmhFToaeZ9YnC8Sib56sQDV5DtjDf78bo1UQbblo3FyouOb3GHjJ5cHud

2681/0P0njKUL+/uOaPgNMQrBs5DdK6Oj5Zlfq7zlsTAHp6O1sNVe6PAirfADk/t

AMiNkj9ip/4njmdToGwi8QKBgQCxS+wewn+xtO8eV0U1vfryccbQDw6U1WHl4r0W

Z3GpAQMil6eiDx48gw0tZbHp9AjhbB+CngO8wonArvSm9a/y3uzCgtgr02LW9IwX

EVbBN2REK9jh1jOO+SyUUp/HIpnDBGFp5Z1Ml0hio+pAyV0cmGydIrB+B3UDz7D+

+C9DhQKBgE7jQjdzHnRFJFicE2NH0hTXOeWlh3Hm3WNpXUv3aFVQZ4CU0usVtar9

zamZjbn6VCBfR07qxUmMFgenWlqSiWqEVlW4OSeytcdCEvAVPIKRHAEEBjjKzNvw

OewP0gIaTMEEAMKeJAAee3JVuhdJlWJg9EnjPlxcX4faFviK2kS9

-----END RSA PRIVATE KEY-----

Above one is githubapp private key

admin

admin123

<http://13.233.35.88:9090/admin/settings?category=housekeeping>

Delete closed issues after

Capgemini@890

28dc99da758ec71c04dd059239d40c5514b488d2 -> client secret from githubapp

sqa\_aafbe8ff8b753bcef76104ece3ac13aec3fce557 – sonar token

Create file called [sonar-project.properties](https://github.com/vks414/composable-commerce/blob/main/storefront-nextjs/sonar-project.properties) and add below details in it

sonar.projectKey=vks414\_composable-commerce\_AYbaCYVSuyElDI5Qgxh

sonar.projectKey=vks414\_composable-commerce\_AYbaCYVSsuyElDI5Qgxh – old one

sonar.projectKey=vks414\_composable-commerce\_AYf\_rsd9fPVLSJ\_oYD4L – new one

and in Jenkinsfile add below stage for sonar analysis

|  |
| --- |
| stage('SonarQubeAnalysis') { |
|  | def scannerHome = tool 'SonarScanner'; |
|  | withSonarQubeEnv('sonar-server') { |
|  | sh "${scannerHome}/bin/sonar-scanner" |
|  | } |
|  | }  sqp\_677d7c5dc1e00cd247910058c5a944277f09fd02 -> SONAR\_TOKEN generated for composable-cmmerce project in sonar and then add in git repo setting as secret  also add SONAR\_HOST\_URL and it’s url <http://13.233.35.88:9090/> in git as repo secret  ghp\_NuJv9XNzj6nxIWBumVb89Wwn4oPv1c4HTZhR ->access token for vks414 git acc  Dependencies/libraries were not provided for analysis of SOURCE files. The 'sonar.java.libraries' property is empty. Verify your configuration, as you might end up with less precise results. |

You should replace **/path/to/library1.jar** and **/path/to/library2.jar** with the actual paths to the external libraries used in your Java project.

Alternatively, you can set the 'sonar.java.libraries' property in the SonarQube project settings by navigating to the project dashboard, selecting "Administration" -> "General Settings" -> "Java" and adding the paths to the external libraries in the "Analysis Scope" section.

Once you have specified the external libraries used in your Java project, you should re-run the SonarQube analysis to ensure that the analysis is using these libraries and to avoid getting the warning message.

node {

stage('SCM Checkout'){

git branch: 'main', credentialsId: 'access-token-2', url: 'https://github.com/vks414/composable-commerce.git'

}

stage('SonarQube Analysis') {

//def scannerHome = tool 'SonarScanner';

nodejs(nodeJSInstallationName: 'nodejs'){

sh 'npm install'

withSonarQubeEnv('sonar-server') {

sh 'npm install sonar-scanner'

sh 'npm run sonar'

}

}

}

}

node {

stage('node'){

nodejs 'nodejs'

// This is required if you want to clean before build

skipDefaultCheckout(true)

}

stage('SCM Checkout'){

cleanWs()

git branch: 'main', credentialsId: 'access-token-2', url: 'https://github.com/vks414/composable-commerce.git'

}

stage('SonarQube Analysis') {

def scannerHome = tool 'SonarScanner';

withSonarQubeEnv('sonar-server') {

sh "${scannerHome}/bin/sonar-scanner"

}

}

}

sonar.java.binaries=.

stage("Quality gate") {

steps {

script {

def qualitygate = waitForQualityGate()

sleep(10)

if (qualitygate.status != "OK") {

waitForQualityGate abortPipeline: true

}

}

}

}

Commerce@123

rm -r storefront-nextjs

**##### Terraform setup #######**

wget <https://releases.hashicorp.com/terraform/1.4.6/terraform_1.4.6_linux_amd64.zip>

sudo apt-get install unzip

unzip terraform\_1.4.6\_linux\_amd64.zip

sudo cp terraform /usr/bin/

terraform -v

apt-get remove terraform -y

echo $PATH

terraform -help

terraform init

terraform apply

terraform destroy

**##### Commercetools #######**

manage\_api\_clients:composable-commerce123

**project\_key -** composable-commerce123

**client\_id -** UI1rcrBeffnNjMPhiqUCEB8X

**secret -** 9PCgC2gxs6z\_NazojICW9pqBL7IOmuAw

**scope -** manage\_project:composable-commerce123

**API URL -** <https://api.us-central1.gcp.commercetools.com>

**Auth URL -** <https://auth.us-central1.gcp.commercetools.com>

provider "commercetools" {

client\_id = "UI1rcrBeffnNjMPhiqUCEB8X"

client\_secret = "9PCgC2gxs6z\_NazojICW9pqBL7IOmuAw"

project\_key = "composable-commerce123"

scopes = "manage\_project:composable-commerce123"

api\_url = "<https://api.us-central1.gcp.commercetools.com>"

token\_url = "<https://auth.us-central1.gcp.commercetools.com>"

}

export TF\_VAR\_project\_key="your\_project\_key"

export TF\_VAR\_client\_id="your\_client\_id"

export TF\_VAR\_client\_secret="your\_client\_secret"

export TF\_VAR\_api\_url="https://api.europe-west1.gcp.commercetools.com"

curl https://auth.us-central1.gcp.commercetools.com/oauth/token \

--basic --user "UI1rcrBeffnNjMPhiqUCEB8X:9PCgC2gxs6z\_NazojICW9pqBL7IOmuAw" \

-X POST \

-d "grant\_type=client\_credentials&scope=manage\_project:composable-commerce123"

export CTP\_DEBUG=1

echo $TF\_VAR\_api\_url

<https://github.com/labd/terraform-provider-commercetools>

Commerce@123

<https://mc.us-central1.gcp.commercetools.com/composable-commerce123/welcome>

<https://mc.europe-west1.gcp.commercetools.com/composable-asset/welcome>

<https://impex.europe-west1.gcp.commercetools.com/>

terraform state list

terraform destroy -target RESOURCE\_TYPE.NAME

eg, terraform destroy -target commercetools\_subscription.my-subscription

terraform state show <resource\_name>

eg , terraform state show amplience\_content\_type.sample

**##### Algolia #######**

<https://github.com/philippe-vandermoere/terraform-provider-algolia/blob/main/examples/main.tf>

Algolia@123

09d92dfd1e9669b1d8e5ee86653e7a3f-> Algolia api key (use admin api key)

VHXT4I8OD8 -> Algolia appln id

**##### GCP #######**

<https://github.com/hashicorp/terraform-provider-google>

https://source.developers.google.com/p/jenkins-ms/r/cloudFun

<https://source.developers.google.com/projects/my-project/repos/my-repo/moveable-aliases/main/paths/index.js>

https://source.developers.google.com/projects/jenkins-ms/repos/demo/moveable-aliases/master/paths/[my\_functions](https://github.com/sgupt187/terraform-demo/tree/main/my_functions)/**product**

**##### Amplience#######**

<https://github.com/labd/terraform-provider-amplience>

Amplience Client Secret and Client ID

Client ID  
3062a4c3-a68d-4f4e-8fbc-c9be2e836aed

Secret  
ff866eeff4f4df2db8d03b76e04aca3585c032d7141b21adb9ee03c113e2039a

export AMPLIENCE\_CLIENT\_ID=3062a4c3-a68d-4f4e-8fbc-c9be2e836aed

export AMPLIENCE\_CLIENT\_SECRET=ff866eeff4f4df2db8d03b76e04aca3585c032d7141b21adb9ee03c113e2039a

export AMPLIENCE\_HUB\_ID=capgsandbox

[vikas.mahajan@capgemini.com](mailto:vikas.mahajan@capgemini.com)

[Test@890](mailto:Test@890)

hub id = capgsandbox

export TF\_LOG=”DEBUG”

* **amplience\_dynamic\_content\_item**: used to create and manage Dynamic Content items
* **amplience\_dynamic\_content\_type**: used to create and manage Dynamic Content types
* **amplience\_image\_set**: used to create and manage Image Sets
* **amplience\_schema**: used to create and manage content schemas
* **amplience\_taxonomy**: used to create and manage taxonomies
* **amplience\_webhook**: used to create and manage webhooks

4ten17V\_mrrvbTRzMKRzDsqaomjFKIrP

endpoint = https://<your-app>.appspot.com/commercetools/projects/<your-project-key>/subscriptions/example-subscription

push\_config {

push\_endpoint = "https://<your-app>.appspot.com/receive"

}

New Amplience Client Id and Secret-

client

d60acc03-42cd-4a0b-8fe9-8d6f9b6004be

secret

0da2fa971b2bb42b8c6b66e4eee201017a2de91a8568b33efeef93938d6f5758

Terraform Cloud :

<https://developer.hashicorp.com/terraform/tutorials/certification/associate-study?in=terraform%2Fcertification>

<https://app.terraform.io/app/getting-started>

shalu321

Terraform@123

**##### Kubernetes#######**

gcloud container clusters get-credentials $(terraform output -raw kubernetes\_cluster\_name) --region $(terraform output -raw region)

gcloud container clusters get-credentials learnk8s-cluster-prod --region= europe-west1

gcloud container clusters get-credentials [capgemini-ga-bigquery-gke](https://console.cloud.google.com/kubernetes/clusters/details/us-central1-c/capgemini-ga-bigquery-gke?project=capgemini-ga-bigquery)

--region=us-central1-c

Step1:

gcloud container clusters get-credentials jenkins-ms -gke --region=us-central1-c

Step2:

export KUBECONFIG=~/.kube/config

<https://support.count.ly/hc/en-us/articles/4411604040217-Docker-and-Kubernetes-Connecting-to-Private-GCR-and-Pulling-Images>

Step3:

cat serviceaccount.json | docker login -u \_json\_key --password-stdin https://gcr.io

cat jenkins-ms-5aa532b8865f.json | docker login -u \_json\_key --password-stdin <https://gcr.io>

Step4:

gcloud auth activate-service-account `Service\_account\_email` --key-file=serviceaccount.json

gcloud auth activate-service-account terraformsetup@jenkins-ms.iam.gserviceaccount.com --key-file=jenkins-ms-5aa532b8865f.json

Step5:

kubectl create secret docker-registry countly-registry \

--docker-server=[https://gcr.io](https://gcr.io/) \

--docker-username=\_json\_key \

--docker-password="$(cat serviceaccount.json)"

Replace serviceaccount with ur account

Step6:

Include countly-registry in .yml file

apiVersion: apps/v1

kind: Deployment

metadata:

name: myapp-deployment

namespace: default

labels:

app: myapp

spec:

replicas: 3

selector:

matchLabels:

app: myapp

template:

metadata:

labels:

app: myapp

spec:

containers:

- name: microreview-images

image: gcr.io/jenkins-ms/microreview-images:latest

ports:

- containerPort: 8080

protocol: TCP

imagePullSecrets:

- name: countly-registry

kubectl apply -f

gcloud config set project <projectid>

gcloud config set project capgemini-ga-bigquery

<https://raw.githubusercontent.com/kubernetes/dashboard/v2.0.0-beta8/aio/deploy/recommended.yaml>

kubectl get deployments

kubectl config get-contexts

kubectl config use-context <context-name>

gke\_capgemini-ga-bigquery\_us-central1-c\_capgemini-ga-bigquery-gke gke\_capgemini-ga-bigquery\_us-central1-c\_capgemini-ga-bigquery-gke gke\_capgemini-ga-bigquery\_us-central1-c\_capgemini-ga-bigquery-gke

kubectl config get-contexts -o=name

kubectl config delete-context context1

kubectl get pods -n <namespace>

kubectl cluster-info

kubectl get nodes

kubectl get componentstatuses

kubectl get namespaces

kubectl get secrets --namespace <namespace> --output json

kubectl create secret docker-registry gcr-secret \

--docker-server=https://gcr.io \

--docker-username=\_json\_key \

--docker-password="$(cat gcr-config.json)" \

--docker-email=shalu.a.gupta@capgemini.com

**##### Elasticsearch#######**

<https://medium.com/@apaar11/getting-started-with-elasticsearch-and-kibana-on-windows-a1536608d01b>

elasticsearch-service.bat start

for password reset

jXHa\_tCHA-m\*2yGiyg5a

<https://www.elastic.co/guide/en/elasticsearch/reference/current/zip-windows.html#windows-service>

<https://localhost:9200/>

enter username: elastic

password: jXHa\_tCHA-m\*2yGiyg5a

{

"name" : "INLEN8520017902",

"cluster\_name" : "elasticsearch",

"cluster\_uuid" : "6A-cm6yBSpezHg5NLdGgkA",

"version" : {

"number" : "8.7.1",

"build\_flavor" : "default",

"build\_type" : "zip",

"build\_hash" : "f229ed3f893a515d590d0f39b05f68913e2d9b53",

"build\_date" : "2023-04-27T04:33:42.127815583Z",

"build\_snapshot" : false,

"lucene\_version" : "9.5.0",

"minimum\_wire\_compatibility\_version" : "7.17.0",

"minimum\_index\_compatibility\_version" : "7.0.0"

},

"tagline" : "You Know, for Search"

}

**##### Kivana#######**

**Go inside bin and then open cmd -> run** kibana.bat

[**http://localhost:5601/?code=453809**](http://localhost:5601/?code=453809)

**Token generated by elasticsearch to configure kivana:**

**elasticsearch-create-enrollment-token.bat --scope kibana -> run cmd from elasticsearch bin**

eyJ2ZXIiOiI4LjcuMSIsImFkciI6WyIxOTIuMTY4LjEwMS43OTo5MjAwIl0sImZnciI6ImE3MjVkYzcxZTU1ZDM5ZDEwN2JjNzE1MDY1MGJiNDIyYTY3YzdlNTEwMjI5MzAxNzk4OTQ0ZDg3MzBhMGY2ZjYiLCJrZXkiOiI0ZEIyQllnQnNvN1VnTVE2OFBpbjpPTWpERjd2SFRTU0s4M2hoZ0lxSUN3In0=

then enter username : elastic

password : jXHa\_tCHA-m\*2yGiyg5a

<http://localhost:5601/app/home#/>

**##### SAP Commerce#######**

<http://13.233.35.88:3000/>

Jenkins

Jenkins@123

ghp\_na7ox0eACXBAqnPOEAhnGWbssc3SzU3BTan4

ghp\_wjfFc3yRMiWIgfzNbZWgiKUz2r3FCV0Ukg5G

>**githubCodeRepoCredentials(vks414)**

**sqa\_77ecd7c8c1cd17757c641cb46f5d42f98b755432 > SONAR\_TOKEN\_SAP**

**generated in sonar**

Password [Capgemini@890](mailto:Capgemini@890)

For changing default port go to below 2 files and make the changes and perform further steps

cd /etc/default/Jenkins

cd /usr/lib/systemd/system/jenkins.service

sudo systemctl daemon-reload

**sudo systemctl restart jenkins.service**

**sudo systemctl status jenkins.service**

**Jenkins sercets 6bad562ba8984614aef7c16b7c7058a0**

[Thursday 14:00] Sharma, Manoj Kumar

subscription id- a0c6431af84c48b9b5a964af9adee14b

Grocery CCV2 API Token- c8ad3ce67fa6577edde4679e192390

1. If the server has a firewall or any security groups, ensure that it allows incoming SSH connections on port 2222.

**sudo systemctl stop Jenkins**

**sudo systemctl start Jenkins**

**sudo rm -rf <dir name>**

**sudo chmod +w /var/lib/jenkins/workspace/BuildEveryDay@tmp/**

**in order to transfer file from local to gcp vm , need to add pub key in authorized\_keys in .ssh dir of vm**

ps -ef | grep tomcat -> to check java version is running or not

**http://34.28.242.179:9002/hac**

**stage('Which Java?') {**

**steps {**

**sh 'java --version'**

**}**

**}**

tools {

jdk 'Java17'

}

Jenkins-SAP

JenkinsSAP123