**Infy-bank application deployment in GCP Kubernetes**

Contents

[Set-Up in GCP instance 3](#_Toc529869842)

[Gcloud 3](#_Toc529869843)

[Jenkins 4](#_Toc529869844)

[Docker 4](#_Toc529869845)

[Kubernetes 4](#_Toc529869846)

[Database 5](#_Toc529869847)

[Create Jenkins Job to Deploy application 6](#_Toc529869848)

# Set-Up in GCP instance

To Deploy the application on GCP need to complete the required set up on GCP instance.

Prerequisite:

* Gcloud
* Jenkins
* Docker
* Kubernetes
* Database

## Gcloud

* Create environment variable for correct distribution

export CLOUD\_SDK\_REPO="cloud-sdk-$(lsb\_release -c -s)"

* Add the Cloud SDK distribution URI as a package source

echo "deb http://packages.cloud.google.com/apt $CLOUD\_SDK\_REPO main" | sudo tee -a /etc/apt/sources.list.d/google-cloud-sdk.list

* Import the Google Cloud Platform public key

curl https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add –

* Update the package list and install the Cloud SDK

sudo apt-get update && sudo apt-get install google-cloud-sdk

* RUN –

export CLOUD\_SDK\_REPO="cloud-sdk-$(lsb\_release -c -s)" && \

echo "deb http://packages.cloud.google.com/apt $CLOUD\_SDK\_REPO main" | tee -a /etc/apt/sources.list.d/google-cloud-sdk.list && \

curl https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add - && \

apt-get update -y && apt-get install google-cloud-sdk -y

* To initialize the SDK:

Run the following at a command prompt:

gcloud init

## Jenkins

Install Jenkins with following commands:

* wget -q -O - https://pkg.jenkins.io/debian/jenkins-ci.org.key | sudo apt-key add -
* sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'
* sudo apt-get update
* sudo apt-get install jenkins

## Docker

* Update the apt package index:

$ sudo apt-get update

* Install packages to allow apt to use a repository over HTTPS:

$ sudo apt-get install \

apt-transport-https \

ca-certificates \

curl \

software-properties-common

* Add Docker’s official GPG key:

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

* Verify that you now have the key with the fingerprint 9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88, by searching for the last 8 characters of the fingerprint.

$ sudo apt-key fingerprint

$ sudo add-apt-repository \

"deb [arch=amd64] [https://download.docker.com/linux/ubuntu \](https://download.docker.com/linux/ubuntu%20\) $(lsb\_release -cs) \ stable"

$ sudo apt-get update

$ sudo apt-get install docker-ce

## Kubernetes

Run following commands to setup Kubernetes in GCP instance (VM):

* sudo apt-get update && sudo apt-get install -y apt-transport-https
* curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -
* echo "deb http://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee -a /etc/apt/sources.list.d/kubernetes.list
* sudo apt-get update
* sudo apt-get install -y kubectl

## Database

In order to connect application to database it’s necessary to setup database in GCP instance.

Two ways to carry out the set up is –

1. install the Oracle 11g database in VM and setup the database
2. Use the dmp file of already created database and set up the database

Dockerfile to install and copy dmp file in oracle.

dokcerfile

FROM wnameless/oracle-xe-11g:latest

ADD devops.dmp /home/oracle/

EXPOSE 1521

EXPOSE 8080

Command to Create the image with this docker file.

docker build -t oracle-db-image .

Docker compose file:

docker-compose

version: "3.6"

services:

    oracle-db:

        image: oracle-db-image

        container\_name: oracle-db

        volumes:

          - ./data:/u01/app/oracle

        ports:

          - 1521:1521

          - 8082:8080

        network\_mode: "bridge"

shell script to import the database:

docker exec -i oracle-db bash <<'EOF'

cp /home/oracle/devops.dmp /u01/app/oracle/admin/XE/dpdump/

impdp system/oracle dumpfile=devops.dmp logfile=devops.log schemas=devops

exit

EOF

Above shell script will load the dmp data to the database

To connect the database to application need to provide following lines to hibernate.cnf.xml file.

<property name="hibernate.connection.url">jdbc:oracle:thin:@<ip>:<port>:xe</property>

<property name="hibernate.connection.username"><username></property>

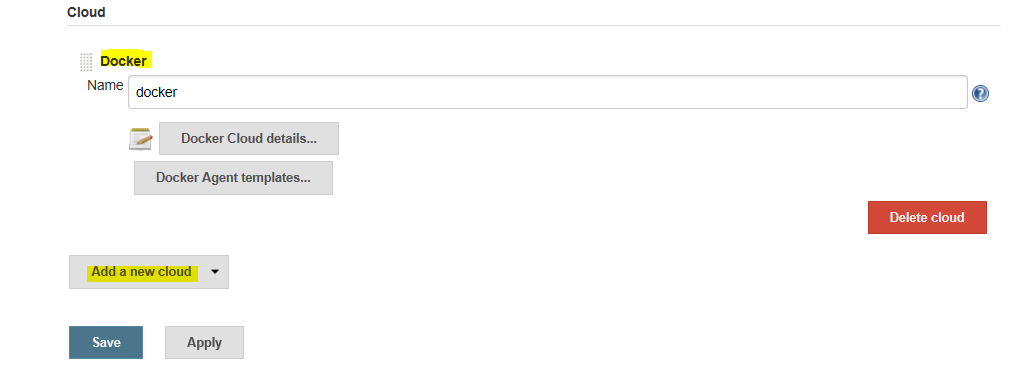
<property name="hibernate.connection.password"><password></property>

# Create Jenkins Job to Deploy application

1. Plugin need to configure with Jenkins:
2. Docker plugin
3. Build publish docker image plugin

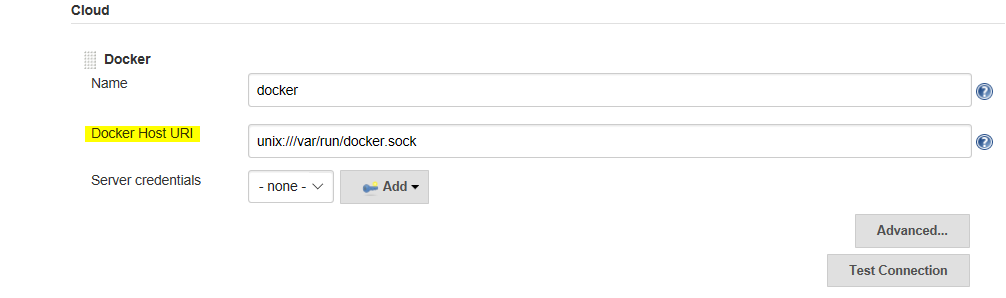
* To configure plugins in Jenkins:

Manage Jenkins 🡪 configure system 🡪cloud 🡪 add new cloud 🡪

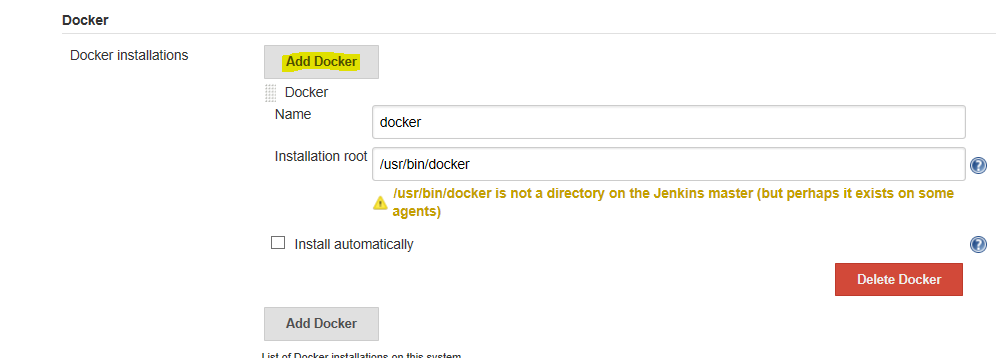


* Click on Docker cloud details

Need to Provide Docker URI and test the connection

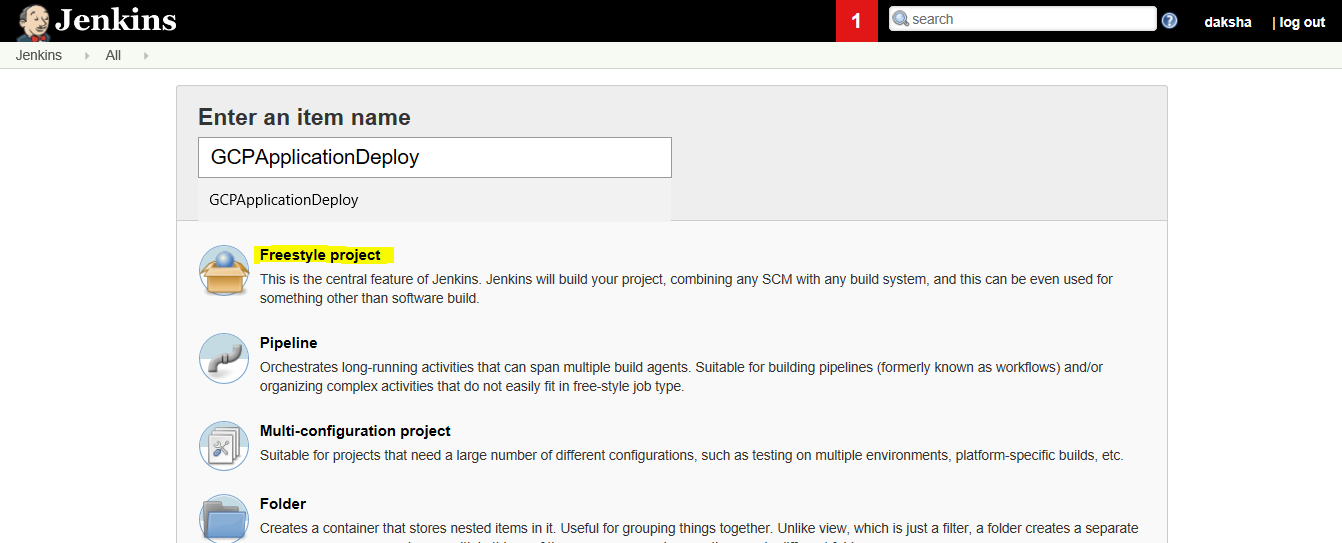


* Manage Jenkins🡪 global tool configuration🡪Docker 🡪 add Docker



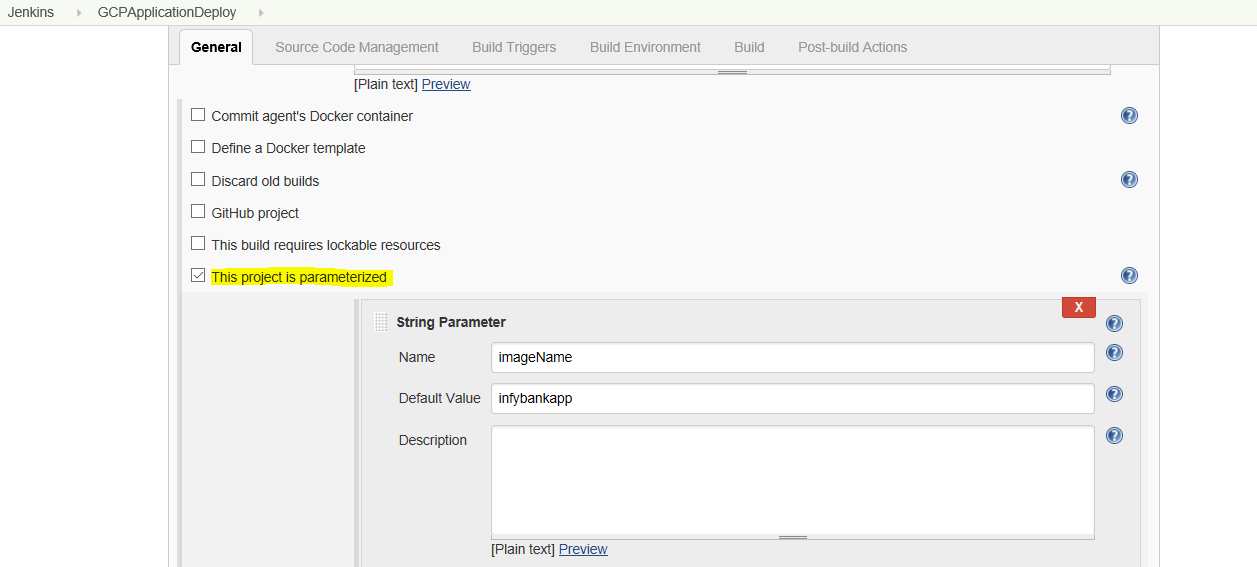
* Create Jenkins Job:

1. Select freestyle job



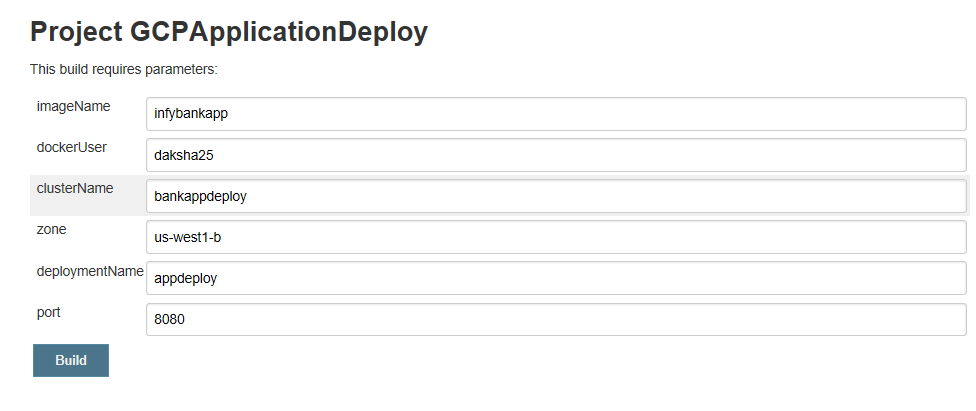
1. Configure the Jenkins job:

To pass the parameters runtime -

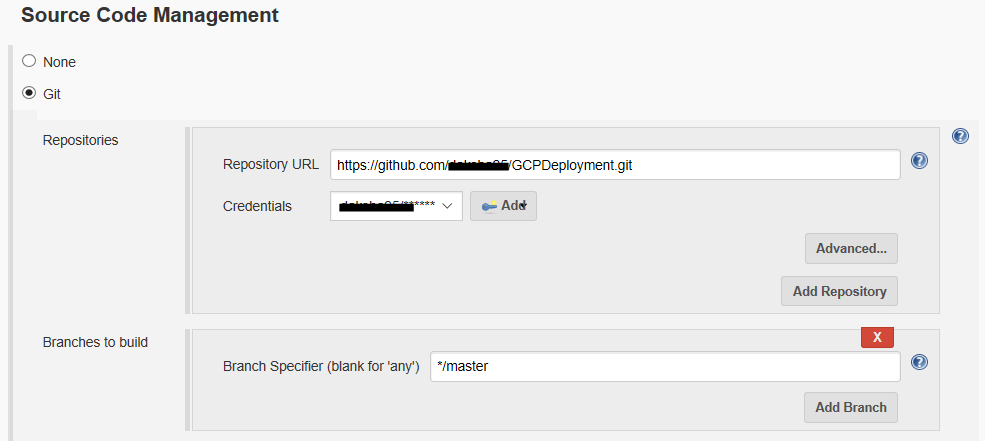


Provide following parameters -

* Image Name
* Port
* Deployment Name
* Cluster Name
* Zone
* Project id
* Docker Username



1. Configure SCM to check out the source code



1. Execute the shell script

To build the image of application

Dockerfile:

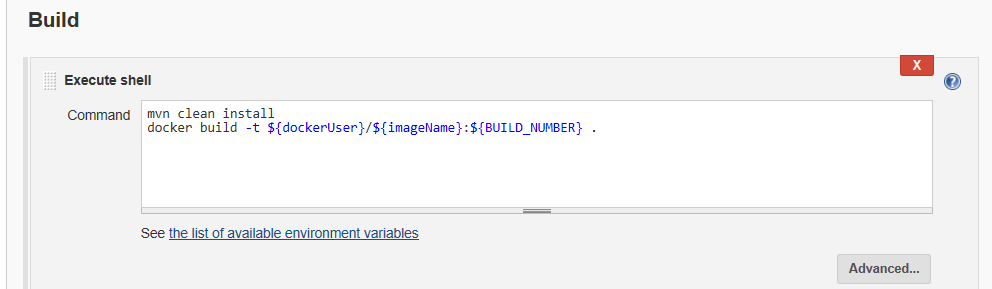
FROM tomcat:8.5

COPY /target/infybank.war /usr/local/tomcat/webapps

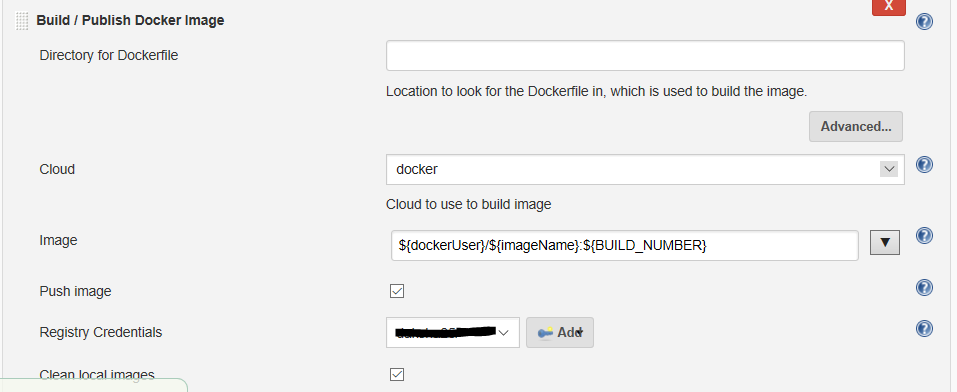
WORKDIR /usr/local/tomcat

RUN chmod +x bin/catalina.sh

CMD ["bin/catalina.sh", "run"]



1. Push the image to docker hub



1. Execute the Shell script with following commands –

* Create a cluster first time:

gcloud container clusters create ${clusterName} --zone ${zone}

* Connect with the cluster:

gcloud container clusters get-credentials ${clusterName} --zone ${zone} --project ${Project\_ID}

* Deploy the application on cluster:

kubectl run ${deploymentName} --image=${dockerUser}/${imageName}:${BUILD\_NUMBER} 11 --env=”CATALINA\_OPTS=-Doracle.jdbc.timezoneAsRegion=false -Duser.timezone=CET” --port=${port}

* Expose the application:

deployment ${deploymentName} --type="LoadBalancer"

* To check the IP and port of cluster on which the application is exposed:

kubectl get services