**Task 1: Git/GitHub**

* git clone <https://github.com/sagarkulkarni1989/Devops_task1.git>
* git branch
* mkdir Tasks1
* cd Tasks1/
* touch README.md
* git add .
* git commit -m "added file"
* git push origin main
* git branch dev
* git checkout dev
* touch testfile.md
* git add .
* git commit -m "added file"
* git push origin dev
* git checkout main
* git branch %USERNAME-new\_feature
* git checkout %USERNAME-new\_feature
* git status
* git log
* touch ./Task1/README.md
* git status
* touch .gitignore
* vim .gitignore add .\*
* cat .gitignore
* git add .
* git commit -m "added gitignore file"
* git push origin %USERNAME-new\_feature
* git checkout sagar-new\_feature
* echo "Devops certification" > README.md
* cat README.md
* git restore README.md
* git checkout main
* git merge dev
* git branch delete sagar-new\_feature
* git push origin --delete sagar-new\_feature

**Task 2: AWS Cloud**

**Task**

1. Done
2. Done
3. Done
4. check SSH connection from your host to the created EC2:

ssh -i mydevopskey.pem [ec2-user@15.207.71.250](mailto:ec2-user@15.207.71.250)

1. ping and SSH passed from one instance to another and vice versa. Configure SSH connectivity:

* login to both instances using default username and create user in both instances
* create a user : Create ansadmin
* passwd ansadmin
* sudo su – ansadmin
* ssh-keygen
* ssh-copy-id ansadnin@targetinstance
* ssh ansadmin@target machine public IP

1. Install web server- nginx

* sudo amazon-linux-extras list | grep epel
* sudo amazon-linux-extras enable epel
* sudo yum install epel-release
* sudo yum install nginx
* systemctl start nginx
* systemctl enable nginx
* systemctl status nginx
* nginx –v
* cd /usr/share/nginx/html
* cat /etc/os-release OS information
* vim index.html - Add information
* hit public ip to browser

EXTRA

2. Write BASH script for installing web server -nginx

**installweb.sh**

#!/bin/bash

echo "installing nginx on server"

yum install epel-release -y

yum install nginx -y

systemctl enable nginx

systemctl start nginx

echo -e "Hello World" > /usr/share/nginx/html/index.html

echo " os version is: $(cat /etc/\*-release)" >> /usr/share/nginx/html/index.html

**Task 3: Terraform/IaC introduction**

Not done

**Task 4: Docker**

1. bash script for installing Docker

**dockerinstallation.sh**

#!/bin/bash

echo "installing docker on server"

yum install docker -y

service docker start

versioninfo=$(docker --version)

echo "docker version $versioninfo"

1. docker container "hello world".

* docker pull hello-world
* docker images
* docker run --name firsthello hello-world
* docker container prune

**EXTRA 2.1 , 3.1 , 3.1.1, 3.2, 3.2.1**

**Dockerfile**

FROM ubuntu

ARG DEBIAN\_FRONTEND=noninteractive

env DEVOPS="test"

RUN apt-get update

RUN apt-get -y install apache2

RUN apt-get update

RUN apt-get -y install apache2-utils

#RUN apt-get clean

EXPOSE 80

COPY index.html /var/www/html

ENTRYPOINT ["./webpage.sh"]

ENTRYPOINT ["apache2ctl"]

CMD ["-DFOREGROUND"]

\*Keep webpage.sh in same location \*

#!/bin/bash

echo "<html><head><title>Exadel -Task4</title></head><body><h1>DEVOPS: $DEVOPS <h1></body></html>" > /var/www/html/index.html

apachectl –DFOREGROUND

**build the image**

* docker build -t webserver .
* docker images

**Container creation**

* docker run -itd --name firstweb -p 80:80 webserver
* docker exec -it 5776e409283c /bin/bash

**TASK 4 Push your docker image to docker hub**

* docker build -t sagarkulkarni1989/webserver:v1 . <new image build>
* docker push sagarkulkarni1989/webserver:v1
* docker tag webserver2 sagarkulkarni1989/webserver:v1 <existing images>

**Task 5 create a docker-compose file**

vi docker-compose.yml

version: '3.9'

services:

apache2:

image: sagarkulkarni1989/webserver:v1

container\_name: my-apache-app

ports:

- '8081:80'

#volumes:

#- ./website:/usr/local/apache2/htdocs

java:

image: bitnami/java:latest

command: "java -jar package.jar"

volumes:

- .:/app

db:

image: mysql

command: --default-authentication-plugin=mysql\_native\_password

restart: always

environment:

MYSQL\_ROOT\_PASSWORD: example

docker compose up

**Task 5: Ansible for beginners**

TASK: Ansible setup

* AWS EC2 instance : 3

Setup Steps

* sudo amazon-linux-extras install ansible2 - Control node
* Create a user in 3 instances
* useradd ansadmin (applicable for all instances )
* passwd ansadmin (applicable for all instances )
* sudo su - ansadmin
* ssh-keygen
* ssh-copy-id [ansadmin@172.31.36.192](mailto:ansadmin@172.31.36.192) Copy key on both nodes
* update hosts file with ip details of node machines
* validation : ansible all -m ping
* ansible all -m ping -i hosts

EXTRA 1: Write a playbook for installing Docker and WordPress

---

- hosts: all

become: yes

tasks:

- name: Install docker.

shell: "yum install docker -y"

- name: Start docker service

service:

name: docker

state: started

enabled: yes

- name: Test Docker with hello world example

shell: "docker run hello-world"

register: hello\_world\_output

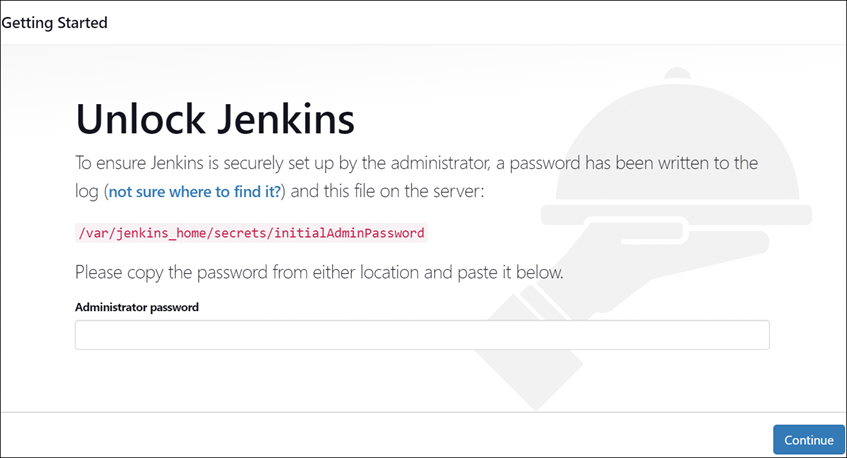
- name: wordpress installation

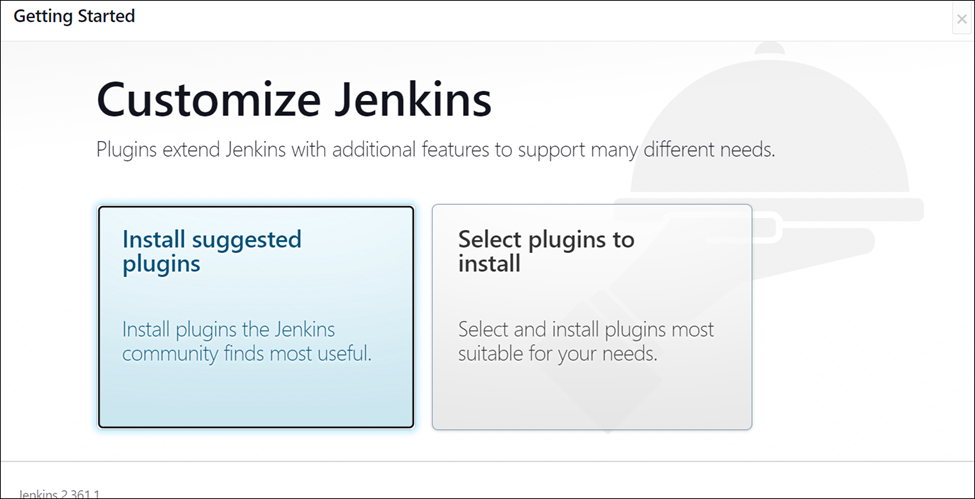
shell: "docker run --name some\_wordpress -p 8080:80 -d wordpress"

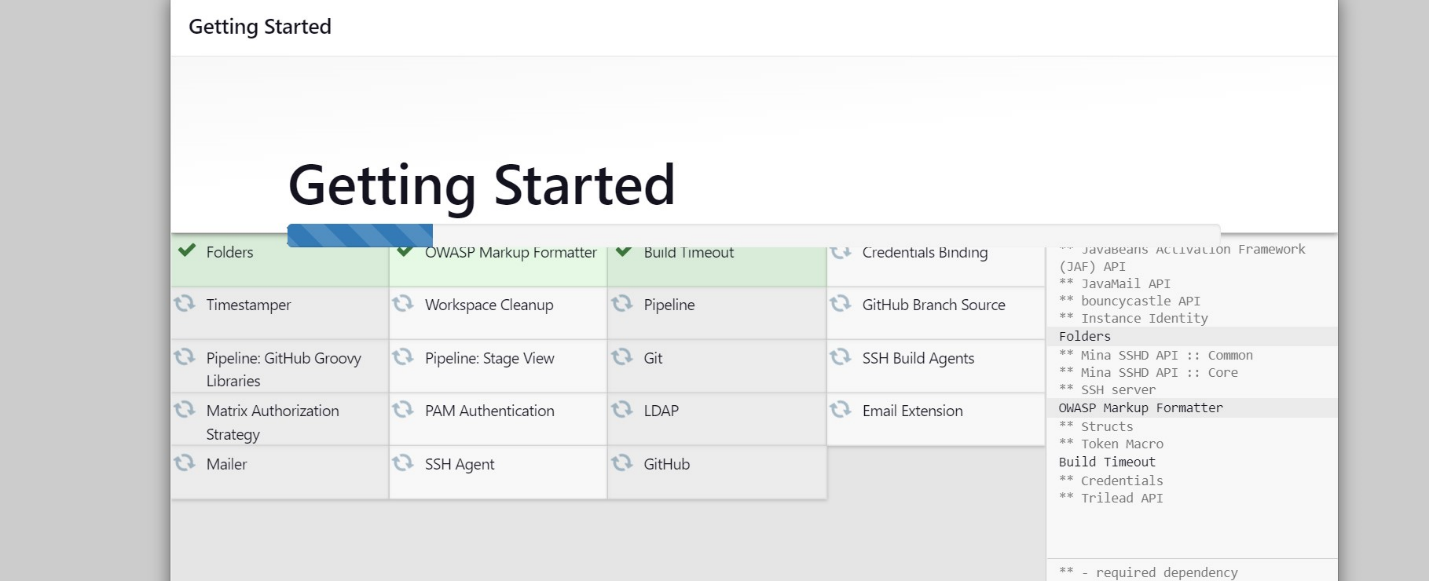
**Task 6: Jenkins. Automate, Manage and Control**

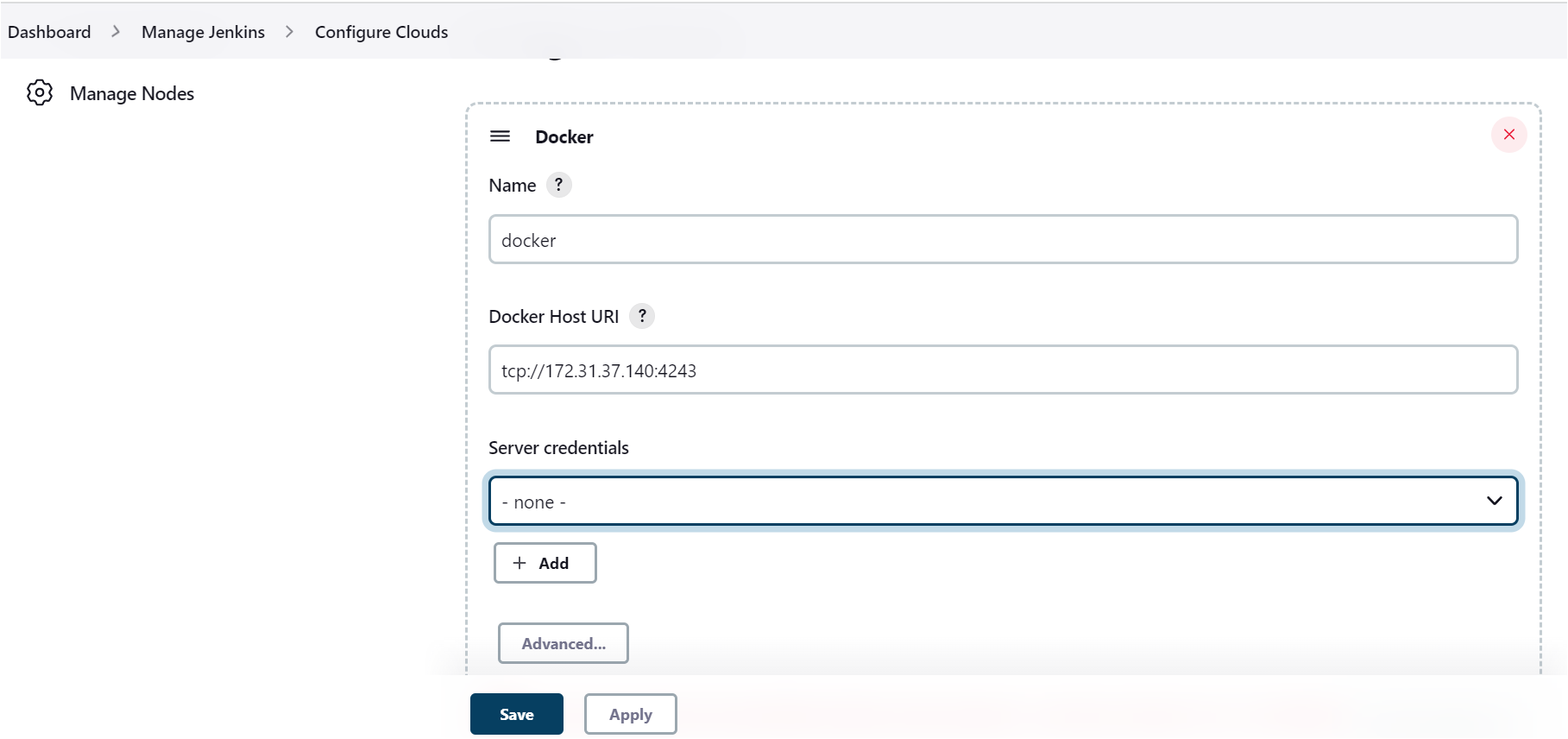
TASK 1,2,3

* AWS account
* get one amazon linux machine
* sudo su -
* yum install docker -y
* docker --version
* service docker start
* service docker status
* docker pull jenkins/jenkins:lts
* docker images
* docker run -u 0 -d -p 8080:8080 -p 50000:50000 -v /data/jenkins:/var/jenkins\_home jenkins/jenkins:lts (The easy fix it to use the -u parameter. Keep in mind this will run as a root user (uid=0))









1. Create a Freestyle project. Which will show the current date as a result of execution.

* New item - <name of the job>
* freestype project –
* Build steps - Execute shell
* date -u
* Execute job :

1. Create Pipeline which will execute docker ps -a in docker agent, running on Jenkins master’s Host.

New item - <name of the job>

Pipeline

Jenkins file

Top of Form

Bottom of Form

Top of Form

Bottom of Form

|  |  |
| --- | --- |
|  | pipeline { |
|  | agent any |
|  | stages { |
|  | stage("build") { |
|  | steps { |
|  | sshagent(credentials: ['Jenkins']) { |
|  | sh ''' |
|  | ssh -o StrictHostKeyChecking=no -l root 172.31.37.140 docker ps -a |
|  | ''' |
|  | } |
|  | } |
|  | } |
|  | } |
|  | } |

EXTRA 2 create an Ansible playbook, which will deploy Jenkins.

---

- hosts: kubernetes

become: yes

tasks:

- name: Install jenkins docker image

shell: "docker run -u 0 -d -p 8080:8080 -p 50000:50000 -v /data/jenkins:/var/jenkins\_home jenkins/jenkins:lts"

**Task 7: Logging & Monitoring**

* wget <https://repo.zabbix.com/zabbix/6.0/debian/pool/main/z/zabbix-release/zabbix-release_6.0-3+debian10_all.deb>
* dpkg -i zabbix-release\_6.0-3+debian10\_all.deb
* apt update
* apt-get install -y zabbix-apache-conf

EXTRA 1.2.1: Complete this task using ansible

---

- hosts: all

become: true

gather\_facts: true

tasks:

- name: install pcre2

yum:

name: "pcre2-devel.x86\_64"

- name: install zabbix centOS 7 rpm file

yum:

name: "https://repo.zabbix.com/zabbix/6.0/rhel/7/x86\_64/zabbix-agent-6.0.0-1.el7.x86\_64.rpm"

validate\_certs: no

- name: upgrade all packages

yum: name=\* state=latest

- name: install zabbix-agent 6.x for centOS 7

yum:

name: zabbix-agent

state: latest

- name: Start service Zabbix Agent, if not started

ansible.builtin.service:

name: zabbix-agent

state: started

ELK:

Docker images and run container

sudo docker run -d -p 5601:5601 -p 9200:9200 -p 5044:5044 -itd --name elk sebp/elk