

You are hired as a DevOps Engineer for Analytics Pvt Ltd. This company is a product-based organization which uses Docker for their containerization needs within the company. The final product received a lot of traction in the first few weeks of launch. Now with the increasing demand, the organization needs to have a platform for automating deployment, scaling and operations of application containers across clusters of hosts. As a DevOps Engineer, you need to implement a DevOps lifecycle such that all the requirements are implemented without any change in the Docker containers in the testing environment.

Up until now, this organization used to follow a monolithic architecture with just 2 developers. The product is present on: <https://github.com/hshar/website.git>

Following are the specifications of the lifecycle:

1. Git workflow should be implemented. Since the company follows a monolithic architecture of development, you need to take care of version control. The release should happen only on the 25th of every month.
2. CodeBuild should be triggered once the commits are made in the master branch.
3. The code should be containerized with the help of the Dockerfile. The Dockerfile should be built every time if there is a push to GitHub. Create a custom Docker image using a Dockerfile.
4. As per the requirement in the production server, you need to use the Kubernetes cluster and the containerized code from Docker Hub should be deployed with 2 replicas. Create a NodePort service and configure the same for port 30008.
5. Create a Jenkins Pipeline script to accomplish the above task.
6. For configuration management of the infrastructure, you need to deploy the configuration on the servers to install necessary software and configurations.
7. Using Terraform, accomplish the task of infrastructure creation in the AWS cloud provider.

Architectural Advice: Softwares to be installed on the respective machines using configuration management.

Worker1: Jenkins, Java

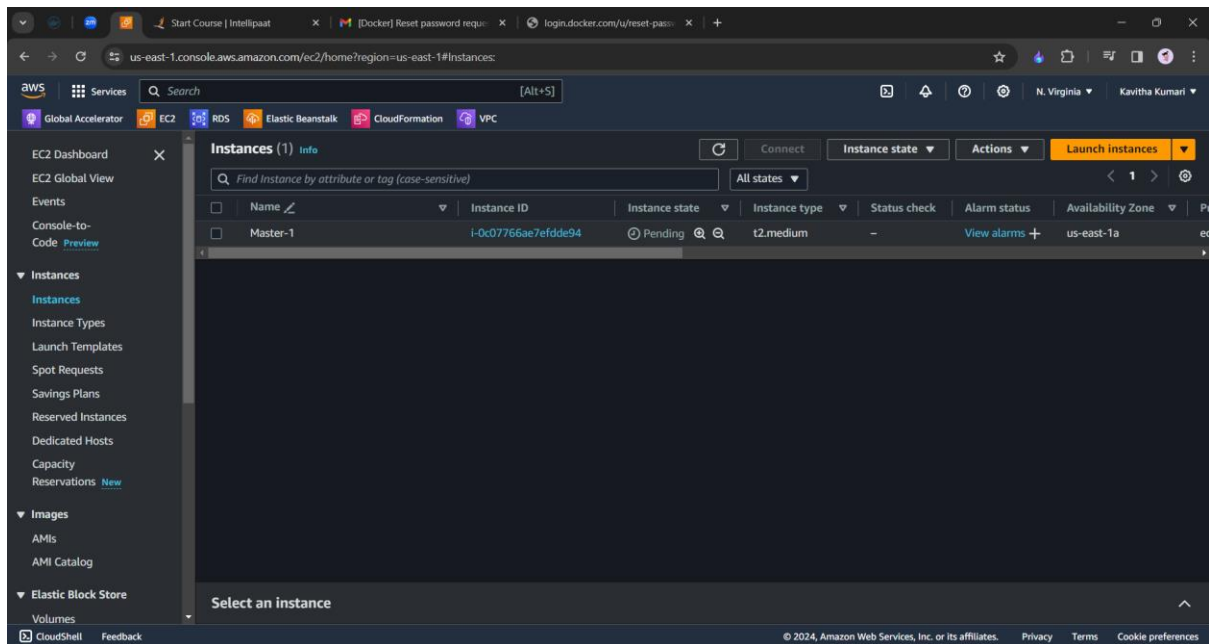
Worker2: Docker, Kubernetes

Worker3: Java, Docker, Kubernetes

Worker4: Docker, Kubernetes

Procedure: -

- **We will create one machine on EC2 (main-1) and we will create another 3 with the help of terraform**



- Launch the t2.medium type instance and install terraform “terraform init”, “terraform plan”, “terraform apply” (<https://developer.hashicorp.com/terraform/install>)
- The below is terraform.tf

```
provider "aws" {
    region="us-east-1"
    access_key=" "
    secret_key=" "
}

resource "aws_instance" "kubernetes-master" {
    ami="ami-0cd59ecaf368e5ccf"
    instance_type="t2.medium"
    subnet_id="subnet-0e8fec508d60f161d"
    vpc_security_group_ids=["sg-0af8f800022ebd9e5"]
    key_name="morningkey"
    associate_public_ip_address = true
    tags={
        Name="machine-3"
    }
}

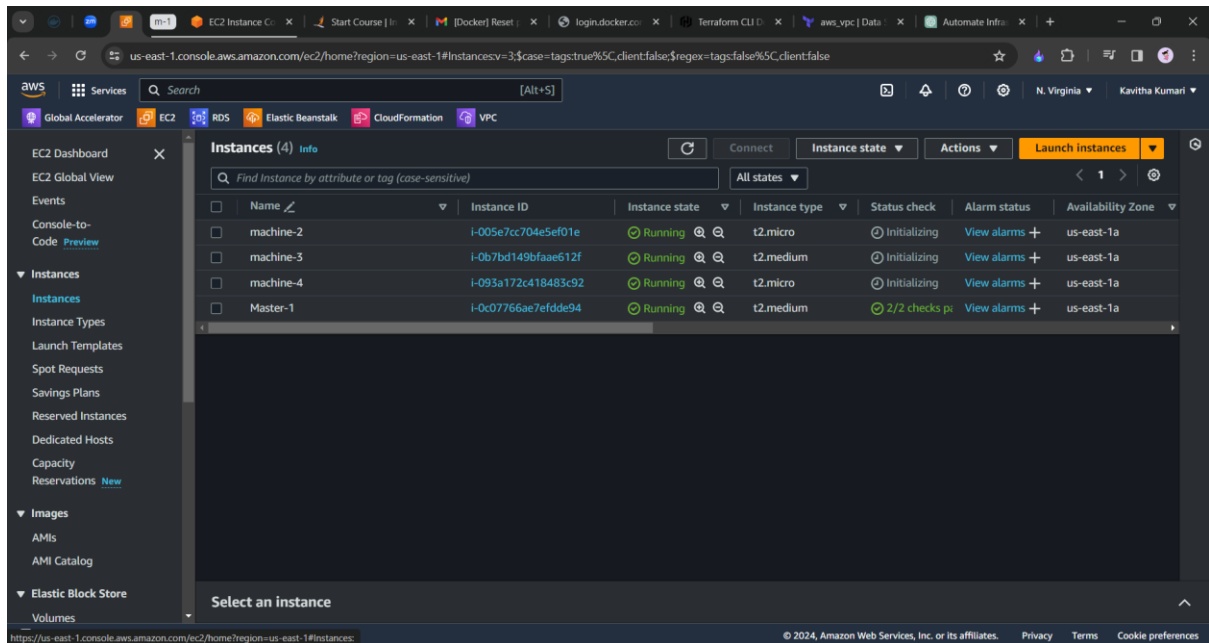
resource "aws_instance" "kubernetes-slave2" {
    ami="ami-0cd59ecaf368e5ccf"
    instance_type="t2.micro"
    subnet_id="subnet-0e8fec508d60f161d"
    vpc_security_group_ids=["sg-0af8f800022ebd9e5"]
    key_name="morningkey"
    associate_public_ip_address = true
    tags={
        Name="machine-4"
    }
}

resource "aws_instance" "kubernetes-slave1" {
    ami="ami-0cd59ecaf368e5ccf"
```

```

instance_type="t2.micro"
subnet_id="subnet-0e8fec508d60f161d"
vpc_security_group_ids=["sg-0af8f800022ebd9e5"]
key_name="morningkey"
associate_public_ip_address = true
tags={
    Name="machine-2"
}
}

```



- Now install ansible of the main machine (main-1) and also ssh-keygen and edit the hosts "sudo nano /etc/ansible/hosts"



- Now we will create the playbook play.yaml
- Also, the script files to install the resources.
- Script1.sh

```

sudo apt update
sudo apt install openjdk-11-jdk -y
//jenkins commands

```

- script2

```
sudo apt update
sudo apt install openjdk-11-jdk -y
sudo apt install docker.io -y
//commands show in the below link
```

<https://github.com/Origamini/K8s-Installation/blob/main/New%20K8s%20Installation.txt>

- script3.sh

```
sudo apt update
sudo apt install docker.io -y
//commands show in the above link
```

- play.yaml

```
---
- name: install jenkins and java on machine-1
  become: true
  hosts: localhost
  tasks:
    - name: running script
      script: script1.sh
- name: install k8s and java on machine-1
  become: true
  hosts: master
  tasks:
    - name: running script
      script: script2.sh
- name: install k8s and java on machine-1
  become: true
  hosts: slaves
  tasks:
    - name: running script
      script: script3.sh
```

- ansible-playbook play.yaml --check

```
sudo apt update
sudo apt upgrade -y
sudo apt install -y curl apt-transport-https ca-certificates
software-properties-common
curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo
apt-key add -
sudo add-apt-repository "deb http://apt.kubernetes.io/ kubernetes-
xenial main"
sudo swapoff -a
sudo apt update
sudo apt install -y kubelet kubeadm kubectl
```

- ansible-playbook play.yaml

```
changed: [localhost]
PLAY [install k8s and java on master] *****
TASK [Gathering Facts] *****
ok: [10.0.9.166]
TASK [Run_script2.sh] *****
changed: [10.0.9.166]
PLAY [install k8s and java on slaves] *****
TASK [Gathering Facts] *****
ok: [10.0.9.67]
ok: [10.0.14.155]
TASK [Run_script3.sh] *****
changed: [10.0.14.155]
changed: [10.0.9.67]
PLAY RECAP *****
10.0.14.155      : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.9.166      : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
10.0.9.67       : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
localhost       : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
ubuntu@ip-10-0-2-28:~$
```

i-04b4764842679705b (machine-1)
PublicIPs: 54.87.42.180 PrivateIPs: 10.0.2.28

- Now on the master-3 machine we need to run the command “kubeadm init” and then paste the following commands into the slaves in order to join the cluster.

IntelliPaat Online Session - Shared screen with speaker view

Chat Messages

Search chat

zamnuth sultana 06:43
yes sir

Gajanan Latti 06:52
yes

Dilshad Shaikh 14:26
hi

syed fahim 15:13
hi

shaillesh verma 15:25
hi

```
K8skubeadm Installation: 20.04.1
File Edit View
which Kubeadm && which Kubectl && which Kubelet
In Master Machine
sudo su
Kubeadm init
In Slave Machines Paste the Token being a root user
In master machine paste below command being a normal user
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config
kubectl apply -f https://github.com/weaveworks/weave/releases/download/v2.8.1/weave-daemonset-k8s.yaml
kubectl get nodes -w
That's all you have configured the K8s Cluster if you face
any issue take screenshot of the error and mail it to support@intellipa.com
```

01:16:10 / 02:51:59 Speed

IntelliPaat Online Session - Shared screen with speaker view

The screenshot shows a Zoom session titled "IntelliPaat Online Session - Shared screen with speaker view". The main window displays a terminal with the following commands and output:

```
Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:
https://kubernetes.io/docs/concepts/cluster-administration/addons/
Then you can join any number of worker nodes by running the following on each as root:

kubeadm join 172.31.19-181-6443 --token d9a1x7.0x8mmlv4c3z55q \
--discovery-token-ca-cert-hash sha256:b88f2b1aadaa24217711cb78ff5404acc5dfb1676842d9517683399e431e9
ubuntu@ip-172-31-19-181:~$ mkdir -p $HOME/.kube
ubuntu@ip-172-31-19-181:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
ubuntu@ip-172-31-19-181:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config
ubuntu@ip-172-31-19-181:~$ kubectl apply -f https://github.com/weaveworks/weave/releases/download/v2.8.1/weave-daemonset-k8s.yaml
serviceaccount/weave-net created
clusterrole.rbac.authorization.k8s.io/weave-net created
clusterrolebinding.rbac.authorization.k8s.io/weave-net created
role.rbac.authorization.k8s.io/weave-net created
rolebinding.rbac.authorization.k8s.io/weave-net created
daemonset.apps/weave-net created
ubuntu@ip-172-31-19-181:~$ kubectl get no
NAME                STATUS    ROLES    AGE   VERSION
ip-172-31-19-181    Ready     control-plane   75s   v1.27.3
ubuntu@ip-172-31-19-181:~$

i-064df4f739e507a1c (Machine-3)
PublicIPs: 3.16.156.153 PrivateIPs: 172.31.19.181
```

On the right, there is a "Chat Messages" window with the following messages:

- zamruth sultana 06:43
- yes sir
- Gajanan Latti 06:52
- yes
- Dilshad Shaikh 14:26
- hi
- syed fahim 15:13
- hi
- shailish verma 15:25
- hi

- Copy the generated token from the above command and paste it to the slave machine (machine-2, master-4)
- Configure the jenkins now.
- Add the nodes with the names k8s-master(master-3),

Jenkins

Dashboard > Manage Jenkins > Nodes > New node

New node

Node name

k8-master

Type

☒ Permanent Agent

Adds a plain, permanent agent to Jenkins. This is called "permanent" because Jenkins doesn't provide higher level of integration with these agents, such as dynamic provisioning. Select this type if no other agent types apply — for example such as when you are adding a physical computer, virtual machines managed outside Jenkins, etc.

Create

REST API Jenkins 2.453

Dashboard > Manage Jenkins > Nodes

Number of executors ?
1

Remote root directory ?
/home/ubuntu/jenkins

Labels ?

Usage ?
Use this node as much as possible

Launch method ?
Launch agents via SSH

Host ?

❗ The Host must be specified

Credentials ?

Save

Dashboard > Manage Jenkins > Nodes

Nodes + New Node Configure Monitors 🔄

S	Name	Architecture	Clock Difference	Free Disk Space	Free Swap Space	Free Temp Space	Response Time
	Built-In Node	Linux (amd64)	In sync	3.73 GiB	0 B	3.73 GiB	0ms
	k8-master	Linux (amd64)	In sync	2.98 GiB	0 B	2.98 GiB	38ms
Data obtained		0.27 sec	0.26 sec	0.25 sec	0.23 sec	0.25 sec	0.25 sec

Icon: S M L

Legend

REST API Jenkins 2.453

- **Note:** we can also use the key mentioned in the "id_rsa" to include the node
- We need to create credential in the jenkins.

Jenkins

Dashboard > Manage Jenkins > Credentials

Credentials

T	P	Store	Domain	ID	Name
		System	(global)	be37b198-7a22-4199-a134-185178ff23b	ubuntu

Stores scoped to Jenkins

P	Store	Domains
	System	(global)

Icons: S M L

REST API Jenkins 2.453

- Click on global and add credentials

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

username with password

Scope ?
Global (jenkins, nodes, items, all child items, etc)

Username ?
kavithakumar062001

☐ Treat username as secret ?

Password ?

ID ?

Description ?

Create

- Give the username and password of the dockerhub account. And create.

Global credentials (unrestricted) [+ Add Credentials](#)

Credentials that should be available irrespective of domain specification to requirements matching.

ID	Name	Kind	Description
be37b198-7a22-4199-a134-185178ff23b	ubuntu	SSH Username with private key	
bbc5b51c-edf6-4f2d-afb4-7403a22787e9	kavithakumari062001/r****	Username with password	

Icon: ☒ S ☐ M ☐ L

REST API Jenkins 2.453

- Goto the path on creating credentials click on global>>add credentials>>username(which is present in the **dockerhub**)>>password(which is used to login in the **dockerhub**)>>click on create
- Fork the repository give by the project. Create Dockerfile

```
FROM ubuntu/apache2
COPY . /var/www/html
```

- Commit changes
- Goto jenkins create a job with the pipeline selected and we need to write script for the pipeline.
- <https://github.com/hshar/website> open the repository and fork it
- We will create Dockerfile to containerize the code. And commit changes

github.com/Origimini/website10-04-2024/new/master

Origimini / website10-04-2024

Code Pull requests Actions Projects Wiki Security Insights Settings

website10-04-2024 / Dockerfile in master

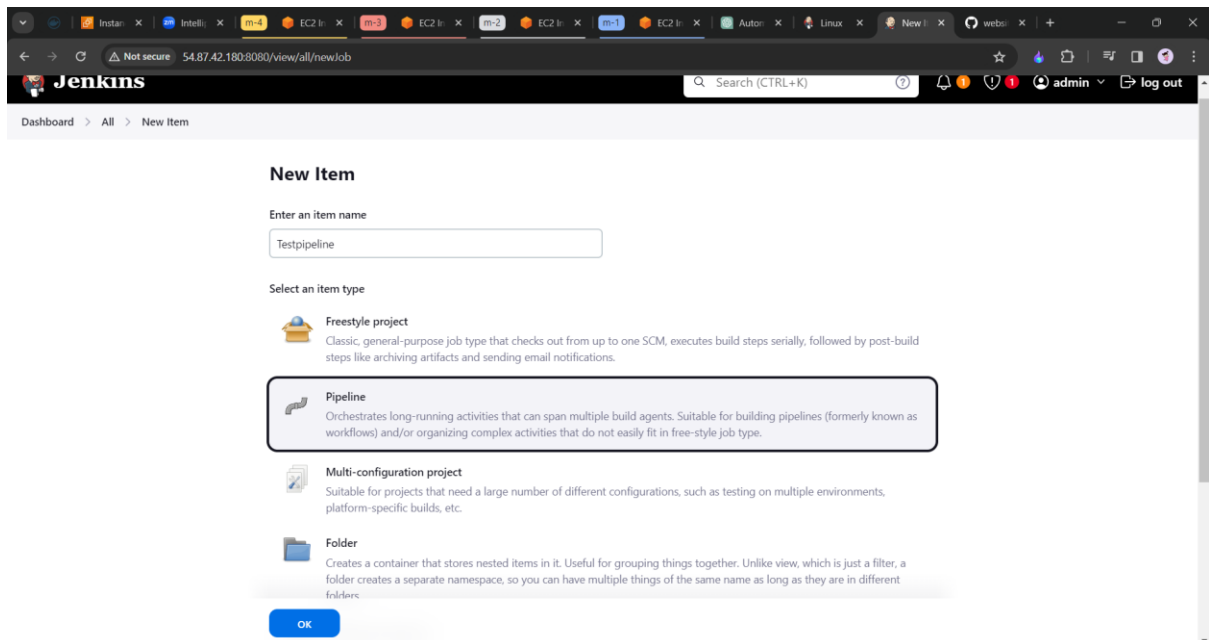
Cancel changes Commit changes...

Edit Preview Spaces 2 No wrap

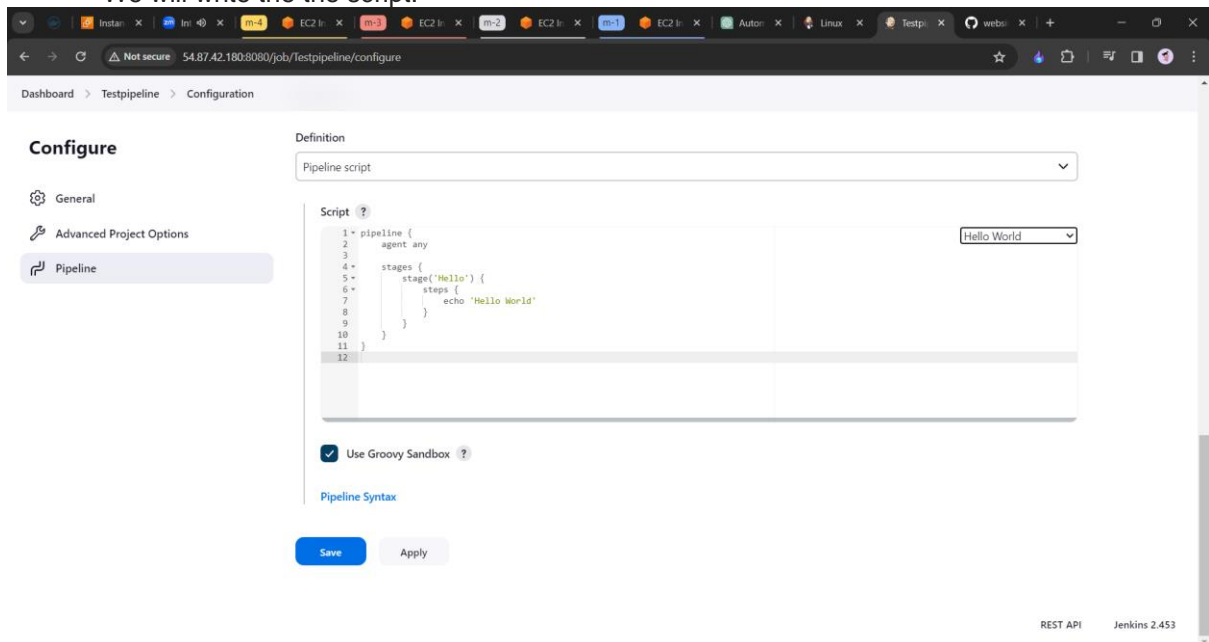
```
1 FROM ubuntu/apache2
2 COPY . /var/www/html
```

Use Control + Shift + M to toggle the tab key moving focus. Alternatively, use esc then tab to move to the next interactive element on the page.

- Come to jenkins and create a pipeline.



- We will write the the script.



Dashboard > Testpipeline > Configuration

Configure

- General
- Advanced Project Options
- Pipeline

Definition

Pipeline script

Script ?

```
1 pipeline {
2   agent none
3   environment {
4     DOCKERHUB_CREDENTIALS = credentials('bbc5b51c-edf6-4f2d-afb4-7403a22787e9')
5   }
6   stages {
7     stage('Hello') {
8       steps {
9         echo 'Hello World'
10      }
11    }
12  }
13 }
14
```

Use Groovy Sandbox ?

Pipeline Syntax

Save Apply

REST API Jenkins 2.453

• Click on save

Dashboard > Testpipeline > #1

- Status
- Changes
- Console Output
- View as plain text
- Edit Build Information
- Delete build '#1'
- Timings
- Pipeline Overview
- Pipeline Console
- Restart from Stage
- Replay
- Pipeline Steps
- Workspaces

Console Output

Started by user [admin](#)

```
[Pipeline] Start of Pipeline
[Pipeline] withCredentials
Masking supported pattern matches of $DOCKERHUB_CREDENTIALS or $DOCKERHUB_CREDENTIALS_PSW
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Hello)
[Pipeline] echo
Hello World
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] End of Pipeline
Finished: SUCCESS
```

REST API Jenkins 2.453

The screenshot shows the GitHub Actions configuration page for a test pipeline. The 'Advanced Project Options' tab is selected. The 'Pipeline' section shows a definition with steps: 'test task', 'clone repo', 'build and push image to Github', and 'create deployment to aws s3'. The 'Pipeline script' section shows the YAML code for the pipeline.

```

1 pipeline {
2   agent none
3   environment {
4     DOCKER_REGISTRY_CREDENTIALS='DockerRegistryCredentials'
5   }
6   stages {
7     stage('BUILD') {
8       steps {
9         echo 'hello world'
10      }
11    }
12  }
13 }

```



Dashboard > Testpipeline > #3

- Git Build Data
- Pipeline Overview
- Pipeline Console
- Restart from Stage
- Replay
- Pipeline Steps
- Workspaces
- Previous Build

```
[Pipeline] stage
[Pipeline] { (git)
[Pipeline] node
Running on k8-master in /home/ubuntu/jenkins/workspace/Testpipeline
[Pipeline] {
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Avoid second fetch
Checking out Revision c7898e09c9d96585997127139d23faf8d8190d9c (refs/remotes/origin/master)
Cloning repository https://github.com/Origamini/website10-04-2024.git
> git init /home/ubuntu/jenkins/workspace/Testpipeline # timeout=10
Fetching upstream changes from https://github.com/Origamini/website10-04-2024.git
> git --version # timeout=10
> git --version # 'git version 2.25.1'
> git fetch --tags --force --progress -- https://github.com/Origamini/website10-04-2024.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/Origamini/website10-04-2024.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/master^(commit) # timeout=10
> git config core.sparsecheckout # timeout=10
> git checkout -f c7898e09c9d96585997127139d23faf8d8190d9c # timeout=10
> git branch -a -v --no-abbrev # timeout=10
> git checkout -b master c7898e09c9d96585997127139d23faf8d8190d9c # timeout=10
Commit message: "Create Dockerfile"
First time build. Skipping changelog.
[Pipeline] }
[Pipeline] // node
[Pipeline] }
[Pipeline] } stage
[Pipeline] }
```

us-east-1.console.aws.amazon.com/ec2-instance-connect/ssh?region=us-east-1&connType=standard&instanceId=i-07d48019716fef42c&osUser=ubuntu&sshPort=22#

Global Accelerator EC2 EBS RDS Elastic Beanstalk CloudFormation VPC

```
clusterrolebinding.rbac.authorization.k8s.io/calico-node created
clusterrolebinding.rbac.authorization.k8s.io/calico-cni-plugin created
daemonset.apps/calico-node created
deployment.apps/calico-kube-controllers created
ubuntu@ip-10-0-9-166:~$
ubuntu@ip-10-0-9-166:~$ kubectl get nodes
NAME                                STATUS    ROLES    AGE   VERSION
ip-10-0-14-155                      Ready     <none>    104s  v1.28.8
ip-10-0-9-166                      NotReady control-plane 4m21s  v1.28.8
ip-10-0-9-67                      NotReady <none>    92s   v1.28.8
ubuntu@ip-10-0-9-166:~$ kubectl get nodes
NAME                                STATUS    ROLES    AGE   VERSION
ip-10-0-14-155                      Ready     <none>    3m2s  v1.28.8
ip-10-0-9-166                      Ready     control-plane 5m39s  v1.28.8
ip-10-0-9-67                      Ready     <none>    2m50s  v1.28.8
ubuntu@ip-10-0-9-166:~$ ls
calico.yaml  jenkins
ubuntu@ip-10-0-9-166:~$ cd jenkins
ubuntu@ip-10-0-9-166:~/jenkins$ ls
remoting  remoting.jar  workspace
ubuntu@ip-10-0-9-166:~/jenkins$ cd workspace
ubuntu@ip-10-0-9-166:~/jenkins/workspace$ ls
Testpipeline
ubuntu@ip-10-0-9-166:~/jenkins/workspace$ cd Testpipeline
ubuntu@ip-10-0-9-166:~/jenkins/workspace/Testpipeline$ ls
Dockerfile  images  index.html
ubuntu@ip-10-0-9-166:~/jenkins/workspace/Testpipeline$
```

i-07d48019716fef42c (machine-3)

Public IPs: 52.204.146.34 Private IPs: 10.0.9.166

CloudShell Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Dashboard > Testpipeline > Configuration

Configure

- General
- Advanced Project Options
- Pipeline

Pipeline

Definition

Pipeline script

```
Script ?
15      }
16 +    steps {
17      git 'https://github.com/Origami/website10-04-2024.git'
18    }
19  }
20 +  stage('Docker') {
21    agent {
22      label 'k8-master'
23    }
24 +    steps {
25      sh 'sudo docker build /home/ubuntu/jenkins/workspace/Testpipeline -t kavithakumari062001/project2'
26      sh 'sudo echo $DOCKERHUB_CREDENTIALS_PSW | sudo docker login -u $DOCKERHUB_CREDENTIALS_USR --password-stdin'
27      sh 'sudo docker push kavithakumari062001/project2'
28    }
29  }
30 }
31 }
```

☒ Use Groovy Sandbox ?

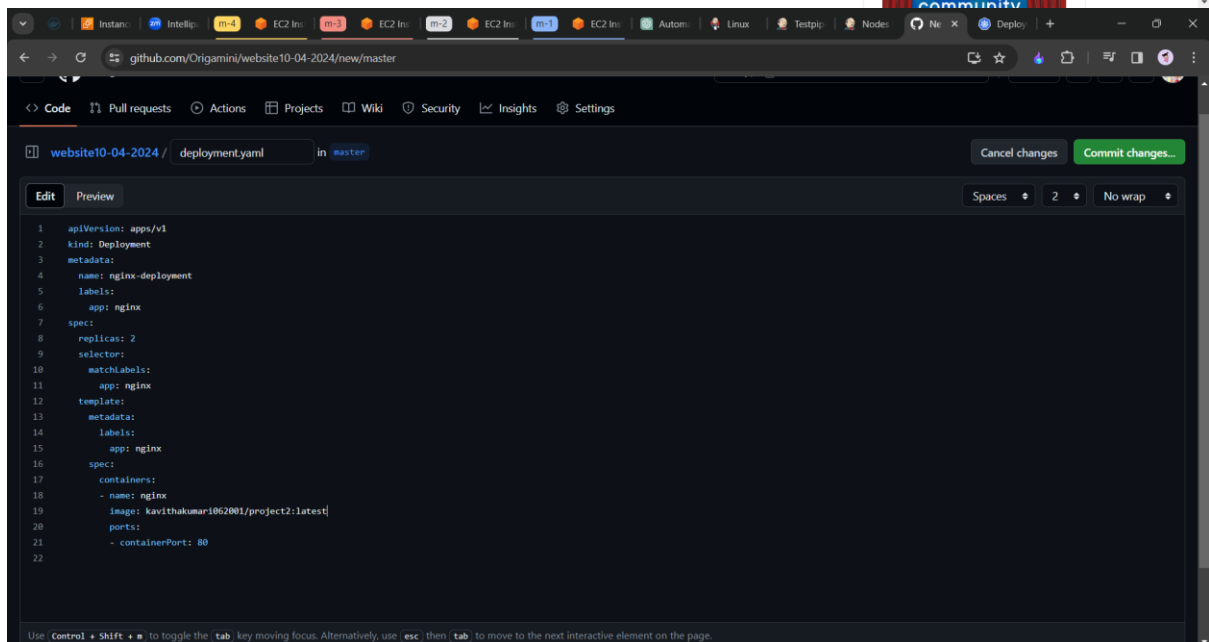
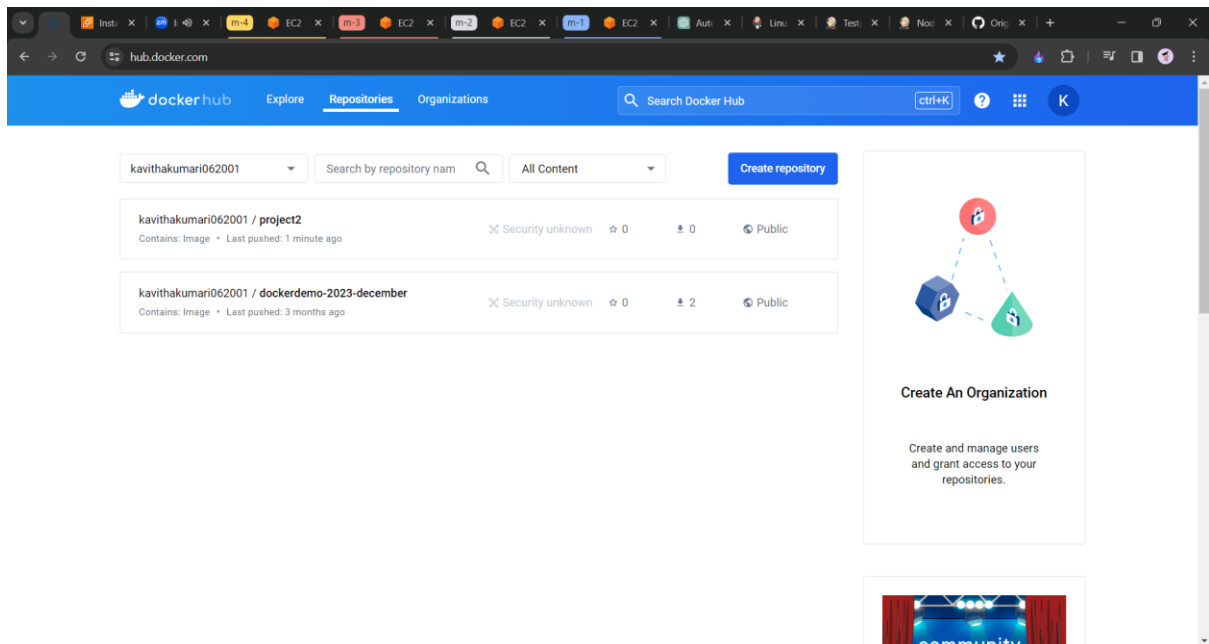
[Pipeline Syntax](#)

Save Apply

Dashboard > Testpipeline > #4

```
[Pipeline] #4
+ sudo echo ****
+ sudo docker login -u kavithakumari062001 --password-stdin
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
[Pipeline] sh
+ sudo docker push kavithakumari062001/project2
Using default tag: latest
The push refers to repository [docker.io/kavithakumari062001/project2]
5e807553f311: Preparing
5e43e67dce9f: Preparing
cdd2aaee2975: Preparing
a2d6ae7d60c7: Preparing
cdd2aaee2975: Mounted from ubuntu/apache2
5e43e67dce9f: Mounted from ubuntu/apache2
a2d6ae7d60c7: Mounted from ubuntu/apache2
5e807553f311: Pushed
latest: digest: sha256:35892c153b5eacecf4e3ce999d3d843cc350f591c7c1adc3e2c048d57a7740f size: 1158
[Pipeline] }
[Pipeline] // node
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] End of Pipeline
Finished: SUCCESS
```



The screenshot shows a web browser with two tabs. The top tab is a GitHub repository for 'website10-04-2024' on the 'master' branch, displaying a 'service.yaml' file. The bottom tab is a Jenkins 'Configure' page for a pipeline named 'Testpipeline', showing a 'Pipeline script' definition.

GitHub Repository: website10-04-2024 / service.yaml

```
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: my-service
5  spec:
6    type: NodePort
7    selector:
8      app: nginx
9    ports:
10     - port: 80
11       targetPort: 80
12       nodePort: 30008
```

Jenkins Configuration: Testpipeline > Configuration

Configure

- General
- Advanced Project Options
- Pipeline**

Definition

Pipeline script

Script

```
26 sh 'sudo echo $DOCKERHUB_CREDENTIALS_PSW | sudo docker login -u $DOCKERHUB_CREDENTIALS_USR --password-stdin'
27 sh 'sudo docker push kavithakumar1062001/project2'
28
29 }
30
31 stage('k8s') {
32   agent {
33     label 'k8s-master'
34   }
35   steps {
36     sh 'kubectl apply -f deployment.yaml'
37     sh 'kubectl apply -f service.yaml'
38   }
39 }
40
41 }
```

☒ Use Groovy Sandbox

[Pipeline Syntax](#)

Save **Apply**

REST API Jenkins 2.453

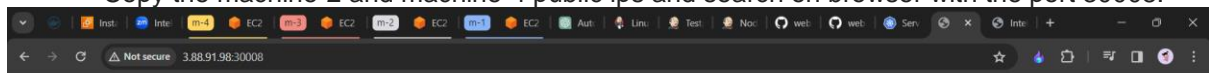
- Save and build the file.

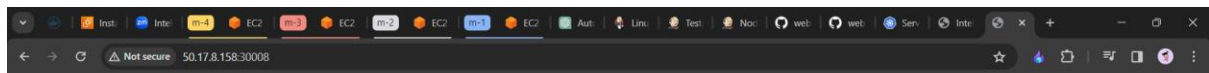

```
Dashboard > Testpipeline > #5

5740ee0490bf: Pushed
latest: digest: sha256:2115b96802619c9cc5331d067138542a6479fb68a4f76445fe8b0a4eb8f60175 size: 1158
[Pipeline] }
[Pipeline] // node
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (k8s)
[Pipeline] node
Running on k8-master in /home/ubuntu/jenkins/workspace/Testpipeline
[Pipeline] {
[Pipeline] sh
* kubectrl apply -f deployment.yaml
deployment.apps/nginx-deployment created
[Pipeline] sh
* kubectrl apply -f service.yaml
service/my-service created
[Pipeline] }
[Pipeline] // node
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withCredentials
[Pipeline] End of Pipeline
Finished: SUCCESS
```

REST API Jenkins 2.453

- Copy the machine-2 and machine-4 public ips and search on browser with the port 30008.



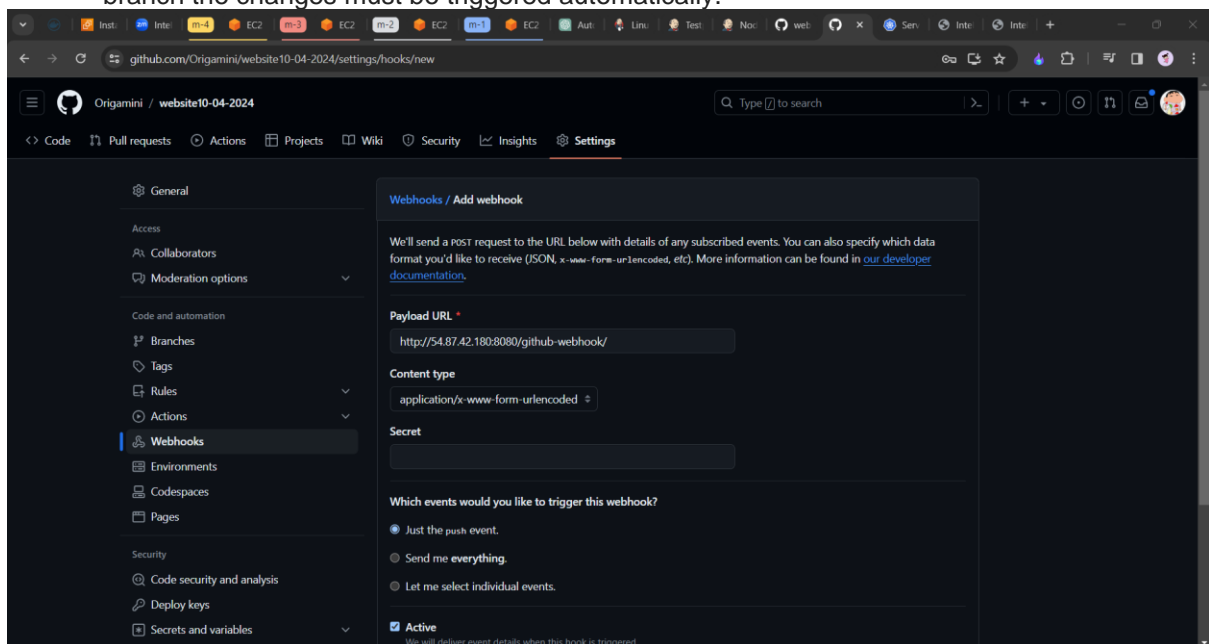


Hello world!



GitHub

- Now we will automate the project so that when ever the changes are made to the master branch the changes must be triggered automatically.



- Click on add webhooks. Now we will try to make changes in the index.html

Dashboard > Testpipeline > Configuration

Configure

- General
- Advanced Project Options
- Pipeline

Definition

Pipeline script

```
Script ?
26      sh 'sudo echo $DOCKERHUB_CREDENTIALS_PSW | sudo docker login -u $DOCKERHUB_CREDENTIALS_USR --password-stdin'
27      sh 'sudo docker push kavithakumar1962001/project2'
28    }
29  }
30
31  stage('k8s') {
32    agent {
33      label 'k8s-master'
34    }
35    steps {
36      sh 'kubectl delete deploy nginx-deployment'
37      sh 'kubectl apply -f deployment.yaml'
38      sh 'kubectl apply -f service.yaml'
39    }
40  }
41
42 }
```

☒ Use Groovy Sandbox ?

[Pipeline Syntax](#)

Save **Apply**

REST API Jenkins 2.453

- Save and build now.

54.87.42.180:8080/job/Testpipeline/configure

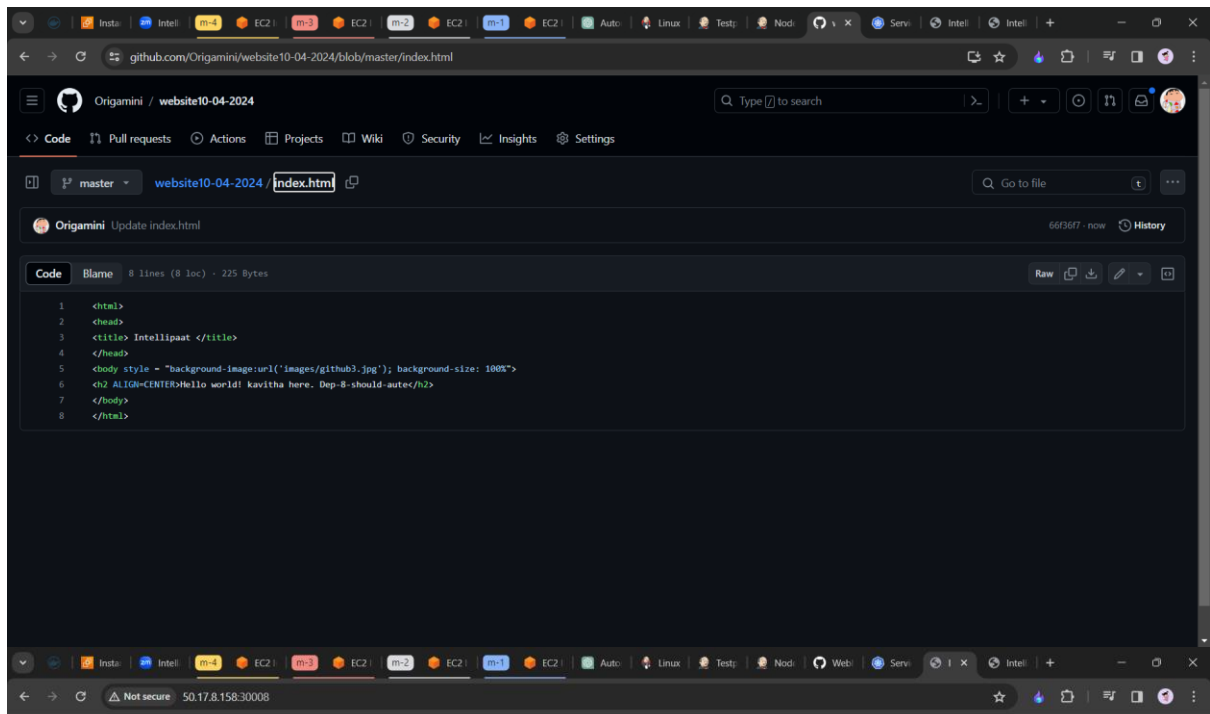
3.88.91.98:30008

Hello world! kavitha here. Dep-7



GitHub

- Now If I make changes in my repository. The pipeline will automatically be triggered



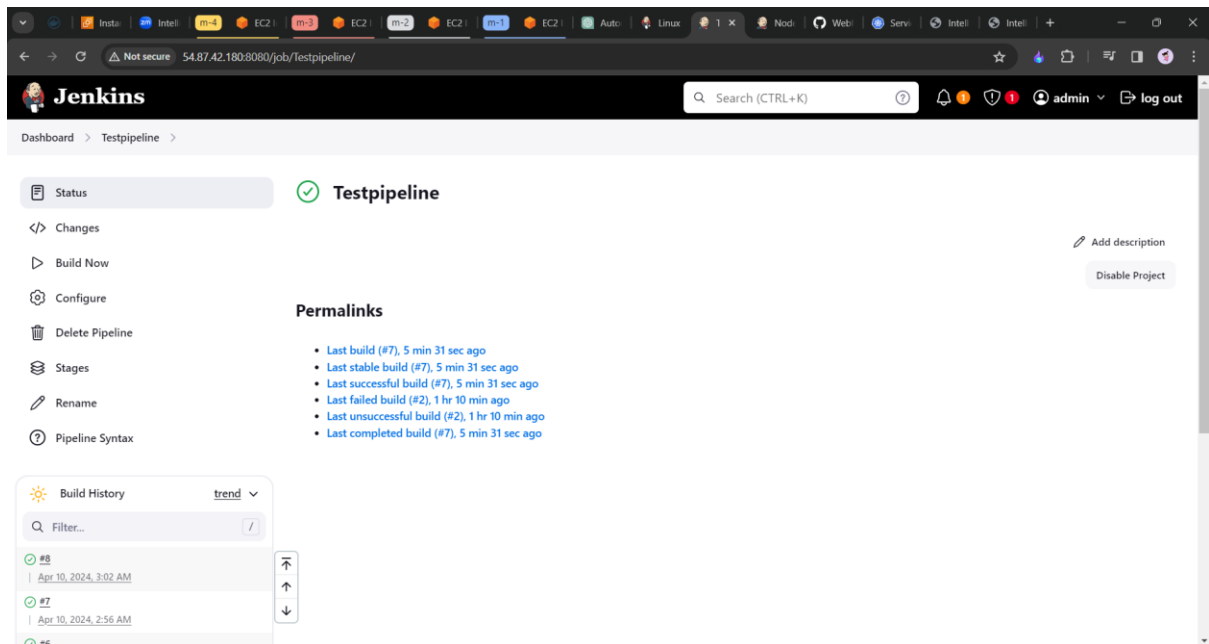
The screenshot shows a web browser displaying the GitHub repository page for 'Origimini' at the path 'website10-04-2024/blob/master/index.html'. The repository is named 'Origimini' and the file is 'index.html'. The code is as follows:

```
1 <html>
2 <head>
3 <title> Intellipaat </title>
4 </head>
5 <body style = "background-image:url('images/github3.jpg'); background-size: 100%">
6 <h2 ALIGN=CENTER>Hello world! kavitha here. Dep-8-should-aute</h2>
7 </body>
8 </html>
```

Hello world! kavitha here. Dep-8-should-aute



GitHub



Pipeline: -

```
pipeline {
  agent none
  environment {
    DOCKERHUB_CREDENTIALS = credentials('bbc5b51c-edf6-4f2d-afb4-7403a22787e9')
  }
  stages {
    stage('Hello') {
      steps {
        echo 'Hello World'
      }
    }
    stage('Git') {
      agent {
        label 'k8-master'
      }
      steps {
        git 'https://github.com/Origamini/website10-04-2024.git'
      }
    }
    stage('Docker') {
      agent {
        label 'k8-master'
      }
      steps {
        sh 'sudo docker build /home/ubuntu/jenkins/workspace/Testpipeline -t kavithakumari062001/project2'
        sh 'sudo echo $DOCKERHUB_CREDENTIALS_PSW | sudo docker login -u $DOCKERHUB_CREDENTIALS_USR --password-stdin'
        sh 'sudo docker push kavithakumari062001/project2'
```

```
    }  
  }  
  stage('k8s') {  
    agent {  
      label 'k8-master'  
    }  
    steps {  
      sh 'kubectl delete deploy nginx-deployment'  
      sh 'kubectl apply -f deployment.yaml'  
      sh 'kubectl apply -f service.yaml'  
    }  
  }  
}  
}
```

-----END-----