

Project - Medicure

Submitted by - **Maulik DEVANI**

Date of submission - **13-07-2024**

Submitted to - Mr.Vikul

● **Problem Statement :**

- The Medicure would centrally like to manage all the doctor's and patient's data across the Medicure hospitals in various cities. They have developed an microservice, which offers these services. In order to reduce unnecessary maintenance cost and manual labor, they would like to automate their application build and deployment process using DevOps. They are fine to use any one of the (AWS, Azure, GCP) cloud platform as their primary cloud service provider.
- The company's primary goal is to deliver the product updates frequently to production with High quality & Reliability.
- Kubernetes cluster must contain at least 2 servers and must be monitored continuously using Prometheus and dashboard must be visualized using Grafana.

Following are the problems:

- ✓ Building Complex builds is difficult
 - ✓ Manual efforts to test various components/modules of the project
 - ✓ Incremental builds are difficult to manage, test and deploy
 - ✓ Creation of infrastructure and configure it manually is very time consuming
 - ✓ Continuous manual monitoring the application is quite challenging.
-

● **Aim:**

As soon as the developer pushes the updated code on the GIT master branch, the Jenkins pipeline should be triggered and code should be checkout, compiled, tested, packaged and containerized.

Tools:

- ✓ Git - For version control for tracking changes in the code files
 - ✓ Jenkins - For continuous integration and continuous deployment
 - ✓ Docker - For containerizing applications
 - ✓ Ansible - Configuration management tools
 - ✓ Terraform - For creation of infrastructure.
 - ✓ Kubernetes – for running containerized application in managed cluster.
-

● Infrastructure:

Jenkins Master: #Create and Schedule the job

Jenkins Slave: #To perform build app. Docker & Build

K8s Master: #To deploy the pods

K8s Worker Node-1:

K8s Worker Node-2:

● Tools:

Jenkins Master: #JDK, Jenkins, Git, Ansible

Jenkins Slave(Build-Server): #JDK, Git, Maven, Docker

K8s Master: #cri, kubelet, kubectl, kubeadm,....

K8s Worker Node-1:

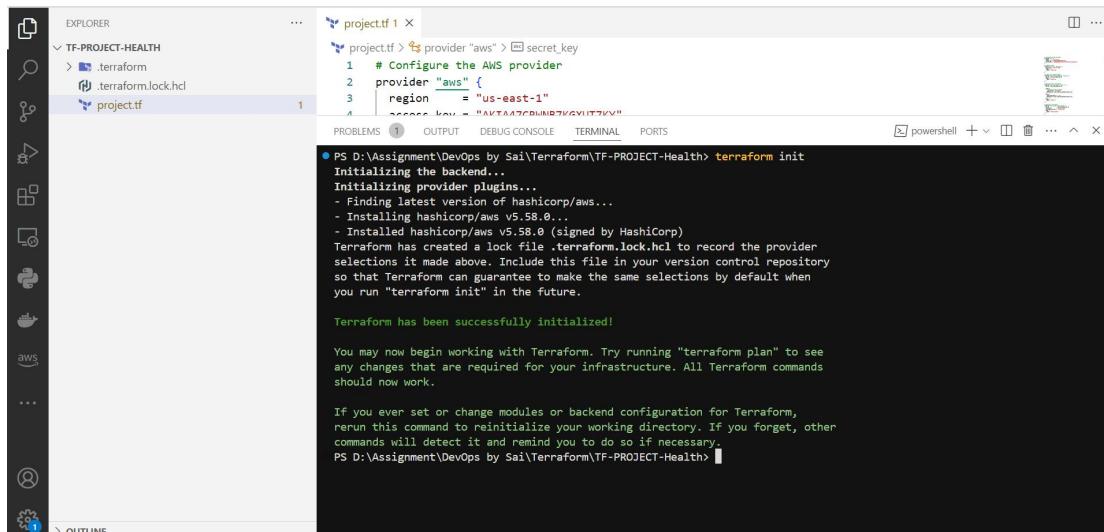
K8s Worker Node-2:

● Installation and Integration of server:

With terraform:

Providing build server:

➤ Initialize terraform



The screenshot shows the Visual Studio Code interface with the Terraform extension installed. The Explorer sidebar shows a project structure with files like .terraform, .terraform.lock.hcl, and project.tf. The code editor displays the contents of project.tf, which includes provider configurations for AWS. The Terminal tab shows the command "terraform init" being run in a PowerShell window, outputting logs about initializing the backend, finding provider plugins, and creating a lock file. The status bar indicates the command was run from "D:\Assignment\DevOps by Sai\Terraform\TF-PROJECT-Health".

```
project.tf 1
provider "aws" {
  region = "us-east-1"
  access_key = "XXXXXXXXXXXXXX"
  secret_key = "XXXXXXXXXXXXXX"
}

PS D:\Assignment\DevOps by Sai\Terraform\TF-PROJECT-Health> terraform init
Initializing the backend...
Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.58.0...
- Installed hashicorp/aws v5.58.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
PS D:\Assignment\DevOps by Sai\Terraform\TF-PROJECT-Health>
```

➤ Terraform Plan:

The screenshot shows the VS Code interface with the terminal tab active. The terminal window displays the Terraform plan output for a project named 'TF-PROJECT-HEALTH'. The output shows 9 resources to be added, 0 to be changed, and 0 to be destroyed. It includes resource definitions for cidr_blocks, default_route_table_ids, and security_group_ids, along with their respective configurations like region, access_key, and secret_key.

```

+ cidr_block          = "10.0.0.0/16"
+ default_network_acl_id = "(known after apply)"
+ default_route_table_id = "(known after apply)"
+ default_security_group_id = "(known after apply)"
+ dhcp_options_id     = "(known after apply)"
+ enable_dns_hostnames = "(known after apply)"
+ enable_dns_support   = true
+ enable_network_address_usage_metrics = "(known after apply)"
+ id                  = "(known after apply)"
+ instance_tenancy    = "default"
+ ipv6_association_id = "(known after apply)"
+ ipv6_cidr_block      = "(known after apply)"
+ main_route_table_id  = "(known after apply)"
+ owner_id             = "(known after apply)"
+ tags                = {
    + "Name" = "JBuild-Server"
  }
+ tags_all            = {
    + "Name" = "JBuild-Server"
  }
}

Plan: 9 to add, 0 to change, 0 to destroy.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
PS D:\Assignment\DevOps\Sal\Terraform\TF-PROJECT-Health>

```

Terraform apply:

The screenshot shows the VS Code interface with the terminal tab active. The terminal window displays the Terraform apply output for the same project. It shows the execution of various AWS provider configurations and the creation of a VPC. The output includes log messages for subnet, route table, and security group creation, along with instance and EIP allocation. The final message indicates that 9 resources were added successfully.

```

1 # Configure the AWS provider
2 provider "aws" {
3   region        = "us-east-1"
4   access_key    = "AKIA47CRWNB7KGXUT7KY"
5   secret_key    = "ASha[REDACTED]sinme0fdjlmz"
6 }
7
8 # Creating a VPC
9 resource "aws_vpc" "demo_vpc" {
10   cidr_block = "10.0.0.0/16"
11   tags = {
}

aws_subnet.demo_subnet: Creation complete after 1s [id=subnet-0d6395185d6dbdb1d]
aws_route_table.demo_rt: Creation complete after 2s [id=rtb-0a079ef336826db60]
aws_route_table_association.proj_rt_sub_assoc: Creating...
aws_security_group.demo_sg: Creation complete after 3s [id=sg-0d2e0ec4a29cab01f]
aws_network_interface.demo_ni: Creating...
aws_network_interface.demo_ni: Creation complete after 1s [id=eni-08e9a582166ca9025]
aws_instance.test_server: Creating...
aws_instance.test_server: Still creating... [10s elapsed]
aws_instance.test_server: Still creating... [20s elapsed]
aws_instance.test_server: Still creating... [30s elapsed]
aws_instance.test_server: Creation complete after 34s [id=i-0276373b8df0490e3]
aws_eip.demo_eip: Creating...
aws_eip.demo_eip: Creation complete after 2s [id=ipalloc-0d4b0eebfdeccf65c]

Apply complete! Resources: 9 added, 0 changed, 0 destroyed.
PS D:\Assignment\DevOps\Sal\Terraform\TF-PROJECT-Health>

```

The screenshot shows the AWS EC2 Instances page for the instance 'i-0276373b8df0490e3' (JBuild-Server). The instance summary table provides detailed information about the instance, including its public and private IP addresses, instance state, and VPC configuration. The VPC section is highlighted with a red box, showing the VPC ID ('vpc-0a9d099e8ebbac097') and Subnet ID ('subnet-0d6395185d6dbdb1d'). Other visible details include the instance type (t2.micro), IAM role, and Auto Scaling Group name.

Instance summary for i-0276373b8df0490e3 (JBuild-Server)		
Updated 1 minute ago	Actions	
Instance ID	Public IPv4 address	Private IPv4 addresses
i-0276373b8df0490e3 (JBuild-Server)	44.194.200.199 open address	10.0.1.10
IPv6 address	Instance state	Public IPv4 DNS
-	Running	-
Hostname type	Private IP DNS name (IPv4 only)	Elastic IP addresses
IP name: ip-10-0-1-10.ec2.internal	ip-10-0-1-10.ec2.internal	44.194.200.199 (JBuild-Server) [Public IP]
Answer private resource DNS name	Instance type	AWS Compute Optimizer finding
-	t2.micro	Opt-in to AWS Compute Optimizer for recommendations.
Auto-assigned IP address	VPC ID	Learn more
-	vpc-0a9d099e8ebbac097 (JBuild-Server)	
IAM Role	Subnet ID	Auto Scaling Group name
-	subnet-0d6395185d6dbdb1d (JBuild-Server)	-
IMDSv2	Instance ARN	
Optional	arn:aws:ec2:us-east-1:891377051774:instance/i-0276373b8df0490e3	
⚠ EC2 recommends setting IMDSv2 to required Learn more		

● Installation and configuration of Jenkins Master:

❖ Jenkins Master: #JDK, Jenkins, Git

The screenshot shows the AWS EC2 Instances page. There are two instances listed: 'JMaster-Health' (Instance ID: i-067b081b46c9fcf78) and 'JBuild-Server' (Instance ID: i-0276373b8df0490e3). Both instances are running, t2.micro type, and located in us-east-1d and us-east-1b respectively. The 'JMaster-Health' instance has 2/2 checks passed. The 'Details' tab is selected for the 'JMaster-Health' instance, showing its public IPv4 address (34.238.135.227), private IP (172.31.86.64), public DNS (ec2-34-238-135-227.compute-1.amazonaws.com), and instance type (t2.micro).

➤ Installation jdk:

```
root@ip-172-31-86-64:~# apt-get install openjdk-17-jdk
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
```

```
root@ip-172-31-86-64:~# java --version
openjdk 17.0.11 2024-04-16
OpenJDK Runtime Environment (build 17.0.11+9-Ubuntu-122.04.1)
OpenJDK 64-Bit Server VM (build 17.0.11+9-Ubuntu-122.04.1, mixed mode, sharing)
root@ip-172-31-86-64:~# git --version
git version 2.34.1
```

➤ Installation jenkins:

<https://www.jenkins.io/doc/book/installing/linux/#debianubuntu>

```
root@ip-172-31-86-64:~# cat jenkins.sh
sudo wget -O /usr/share/keyrings/jenkins-keyring.asc \
  https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key
echo "deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
  https://pkg.jenkins.io/debian-stable binary/" \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update
sudo apt-get install jenkins
root@ip-172-31-86-64:~# sh jenkins.sh
```

```
root@ip-172-31-86-64:~# systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
  Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)
  Active: active (running) since Sat 2024-07-13 11:15:57 UTC; 5s ago
    Main PID: 5410 (java)
       Tasks: 44 (limit: 1120)
      Memory: 329.5M
        CPU: 43.064s
       CGroup: /system.slice/jenkins.service
           └─5410 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080
```

➤ Installing Ansible on JMaster server for configure JSlave:

```
root@ip-172-31-86-64:~# cat ansible.sh
sudo apt update
sudo apt install software-properties-common
sudo add-apt-repository --yes --update ppa:ansible/ansible
sudo apt install ansible -y
```

```
root@ip-172-31-86-64:~# sh ansible.sh
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Ign:5 https://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:6 https://pkg.jenkins.io/debian-stable binary/ Release
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
11 packages can be upgraded. Run 'apt list --upgradable' to see them.
Reading package lists... Done
```

➤ Provide host(Target server):

```
root@ip-172-31-86-64:/etc/ansible# cat hosts
[testnodes]
44.194.200.199
```

➤ Configure ansible tool on Jenkins:

The screenshot shows the Jenkins Plugins page. A search bar at the top contains the text 'ansible'. Below it, a table lists two plugins:

Install	Name	Released
<input checked="" type="checkbox"/>	Ansible 403.v8d0ca_dcb_b_502 pipeline External Site/Tool Integrations DevOps Build Tools Deployment Invoke Ansible Ad-Hoc commands and playbooks.	25 days ago
<input type="checkbox"/>	Ansible Tower 0.16.0 This plugin connects Jenkins with Ansible Tower	4 yr 0 mo ago

Tool

The screenshot shows the Jenkins Tools configuration page under the 'Ansible installations' section. A new instance is being added with the following details:

- Name: ansible
- Install automatically:

At the bottom, there are 'Save' and 'Apply' buttons.

Pipeline script:

Dashboard > ansible-configuration > Configuration

Configure

Pipeline

General

Advanced Project Options

Pipeline

Definition

Pipeline script

Script

```
1 pipeline {
2     agent any
3
4     stages {
5         stage('SCM-Checkout') {
6             steps {
7                 git 'https://github.com/maulik23ii/capstone-health.git'
8             }
9         }
10        stage('jSLAVE-ansIBLE') {
11            steps {
12                ansiblePlaybook become: true, credentialsId: 'ansible-jslave', disableHostKeyChecking: true
13            }
14        }
15    }
16 }
17 }
```

Save

Apply

Pipeline: syntax

Dashboard > ansible-configuration > Pipeline Syntax

Online Documentation

Examples Reference

IntelliJ IDEA GDSDL

ansiblePlaybook: Invoke an ansible playbook

ansiblePlaybook

Ansible tool

ansible

Playbook file path in workspace

ansible-playbook.yml

Inventory file path in workspace

/etc/ansible/hosts

SSH connection credentials

- none -

+ Add ▾

Jenkins Credentials Provider: Jenkins

Add Credentials

Domain

Global credentials (unrestricted)

Kind

SSH Username with private key

Scope ?

Global (Jenkins, nodes, items, all child items, etc)

ID ?

ansible-jslave

Description ?

none

Jenkins Credentials Provider: Jenkins

ubuntu

Treat username as secret ?

Private Key

Enter directly

Key

Enter New Secret Below

```
MABXA0GAZ08A81181F8RXZUKSChFI05MFeVmGtzUirqULV/vnhrpvf/Zqv0K2Cs0R
1+v9hdzreRwUcv7R2GmAz+vBfw2rGbmoVXvoruCri+nobpqviAxKUiLMZWp/B2ZK
IE9dqx+mflmcCz1kJtVO+d4/89RqXPuG4YoNif2WxpuSuNpdVo+k=
-----END RSA PRIVATE KEY-----
```



Passphrase

Dashboard > ansible-configuration > Pipeline Syntax

Disable the host SSH key check

Colorized output

Extra parameters

Generate Pipeline Script

```
ansiblePlaybook become: true, credentialsId: 'ansible-jslave', disableHostKeyChecking: true, installation: 'ansible', inventory: '/etc/ansible/hosts', playbook: 'ansible-playbook.yml', vaultTmpPath: ''
```

On Jslave server:

```
/var/lib/jenkins/workspace/ansible-configuration/ssh2743017449552478781.key -u ubuntu

PLAY [Configure Docker on EC2 Instances] ****
TASK [Gathering Facts] ****
ok: [44.194.200.199]

TASK [updating apt] ****
changed: [44.194.200.199]

TASK [Install Java] ****
changed: [44.194.200.199]

TASK [Install Git] ****
changed: [44.194.200.199]

TASK [Install Docker] ****
changed: [44.194.200.199]

TASK [Start Docker Service] ****
changed: [44.194.200.199]

PLAY RECAP ****
44.194.200.199 : ok=7    changed=6    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[Pipeline] }
[Pipeline] // stage
[Pipeline] }
```

```
root@ip-10-0-1-10:~# docker --version
Command 'docker' not found, but can be installed with:
snap install docker          # version 24.0.5, or
apt install docker.io         # version 24.0.7-0ubuntu2~22.04.1
apt install podman-docker     # version 3.4.4+ds1-1ubuntu1.22.04.2
See 'snap info docker' for additional versions.
root@ip-10-0-1-10:~# docker --version
Docker version 24.0.7, build 24.0.7-0ubuntu2~22.04.1
root@ip-10-0-1-10:~#
```

Integration Jenkins master Jenkins slave:

On Jenkins Server

Dashboard > Manage Jenkins > Nodes >

Remote root directory ?
/home/devopsadmin

Labels ?
slave1

Usage ?
Only build jobs with label expressions matching this node

Launch method ?
Launch agents via SSH

Host ?
44.194.200.199

Credentials ?
devopsadmin

+ Add ▾

Ssh credentials:

Jenkins Credentials Provider: Jenkins

Add Credentials

Domain
Global credentials (unrestricted)

Kind
SSH Username with private key

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

ID ?
jslave

Description ?

Username

devopsadmin

Treat username as secret [?](#)

Private Key

Enter directly

Key

```
-----BEGIN OPENSSH PRIVATE KEY-----
b3B1bnNzaC1rZXktdjEAAAAABG5vbmUAAAAEbm9uZQAAAAAAAAABAAAArAAAABn1Y2RzYS
1zaGEvIw5nc3RwNTIxAAAAACG5pc3RwNTIxAAAAh0QA6g63FwlU0XX1fb/mkeMSUzaN1Bgb
-----END OPENSSH PRIVATE KEY-----
```

Enter New Secret Below

Passphrase

Save

Dashboard > Manage Jenkins > Nodes >

Host [?](#)
44.194.200.199

Credentials [?](#)
devopsadmin
[+ Add](#)

Host Key Verification Strategy [?](#)
Manually trusted key Verification Strategy
 Require manual verification of initial connection [?](#)

Advanced [?](#)

Availability [?](#)
Keep this agent online as much as possible

Node Properties

Disable deferred wipeout on this node [?](#)

Agent Connected

```
[07/13/24 12:23:21] [SSH] Checking java version of java
[07/13/24 12:23:21] [SSH] java -version returned 17.0.11.
[07/13/24 12:23:21] [SSH] Starting sftp client.
[07/13/24 12:23:21] [SSH] Copying latest remoting.jar...
[07/13/24 12:23:21] [SSH] Copied 1,369,595 bytes.
Expanded the channel window size to 4MB
[07/13/24 12:23:21] [SSH] Starting agent process: cd "/home/devopsadmin" && java -jar remoting.jar -workDir /home/devopsadmin -jar-cache /home/devopsadmin/remoting/jarCache
Jul 13, 2024 12:23:21 PM org.jenkinsci.remoting.engine.WorkDirManager initializeWorkDir
INFO: Using /home/devopsadmin/remoting as a remoting work directory
Jul 13, 2024 12:23:21 PM org.jenkinsci.remoting.engine.WorkDirManager setupLogging
INFO: Both error and output logs will be printed to /home/devopsadmin/remoting
<===[JENKINS REMOTING CAPACITY]==>channel started
Remoting version: 3206.vb_15dcf73f6a_9
Launcher: SSHLauncher
Communication Protocol: Standard in/out
This is a Unix agent
Agent successfully connected and online
```

Writing pipeline script for app build, containerized and deployment :

CICD :: git/jenkins/docker/kubernetes

==> CICD Pipeline - to automate the build and deployment.

1. Developers Create the Source Code

2. Commit the Source_Code to Source_Code Repository(Github)

3. Jenkins_pipeline ---> for Monolith Application Architecture

- SCM_Checkout - Download the source_code to build server

- Application_Build - Process of compiling the source code and create artifacts(Binaries - *.war/*.jar)

- Deploy the artifacts to Target Server(QA/UAT/PROD)

3.1 Jenkins Pipeline ---> for Containerized - Micro-Service Based Application Architecture

- SCM_Checkout

- Application_Build using Maven

- Application Image Build using Docker

- Published to Container Registry(Dockerhub)

- Deploy the Container Image in the Target Server and run the Application using Container.

Stage 1: "SCM checkout"

Script ?

```
1 * pipeline {
2     agent {label 'slave1'}
3
4     stages {
5         stage('SCM_Checkout') {
6             steps {
7                 git 'https://github.com/maulik2311/capstone-health.git'
8             }
9         }
10    }
11 }
12 }
```

Console Output

```
Started by user Maulik DEVANI
[Pipeline] Start of Pipeline
[Pipeline] node
Running on slave1 in /home/devopsadmin/workspace/Health_care
[Pipeline] {
[Pipeline] stage
[Pipeline] { (SCM_Checkout)
[Pipeline] git
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/maulik2311/capstone-health.git
> git init /home/devopsadmin/workspace/Health_care # timeout=10
```

```

devopsadmin@ip-10-0-1-10:~/workspace/Health_care$ pwd
/home/devopsadmin/workspace/Health_care
devopsadmin@ip-10-0-1-10:~/workspace/Health_care$ ls
Dockerfile ansible-playbook.yml k8sdeploy.yaml mvnw mvnw.cmd pom.xml src
devopsadmin@ip-10-0-1-10:~/workspace/Health_care$
```

Stage-2; "Application Build"

Script ?

```

1 ▼ pipeline {
2     agent {label 'slave1'}
3
4     stages {
5         stage('SCM_Checkout') {
6             steps {
7                 git 'https://github.com/maulik2311/capstone-health.git'
8             }
9         }
10        stage('App_Build_mvn') {
11            steps {
12                sh 'mvn clean package'
13            }
14        }
15    }
16}
17
```

```

completed.
[1;34mINFO[m]
[1;34mINFO[m] Results:
[1;34mINFO[m]
[1;34mINFO[m] [1;32mTests run: 3, Failures: 0, Errors: 0, Skipped: 0[m
[1;34mINFO[m]
[1;34mINFO[m]
[1;34mINFO[m] [1m-- [0;32mmaven-jar-plugin:3.2.2:jar@m [1m(default-jar)@m @ [36mmedicure@[0;1m --@[m
[1;34mINFO[m] Building jar: /home/devopsadmin/workspace/Health_care/target/medicure-0.0.1-SNAPSHOT.jar
[1;34mINFO[m]
[1;34mINFO[m] [1m-- [0;32mspring-boot-maven-plugin:2.7.4:repackage@m [1m(repackage)@m @ [36mmedicure@[0;1m --@[m
[1;34mINFO[m] Replacing main artifact with repackaged archive
[1;34mINFO[m] [1m-----[m
[1;34mINFO[m] [1;32mBUILD SUCCESS[m
[1;34mINFO[m] [1m-----[m
[1;34mINFO[m] Total time: 21.863 s
[1;34mINFO[m] Finished at: 2024-07-13T12:38:44Z
[1;34mINFO[m] [1m-----[m
[Pipeline]
[Pipeline] // stage
[Pipeline]
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

```

devopsadmin@ip-10-0-1-10:~/workspace/Health_care/target$ ls
classes generated-test-sources maven-status medicure-0.0.1-SNAPSHOT.jar.original test-classes
generated-sources maven-archiver medicure-0.0.1-SNAPSHOT.jar surefire-reports
```

Stage-3: "Application image build-Docker"

Dockerfile:

```
FROM openjdk:11
ARG JAR_FILE=target/*.jar
COPY ${JAR_FILE} app.jar
ENTRYPOINT ["java","-jar","/app.jar"]
```

Script ?

```
1 ▼ pipeline {
2     agent {label 'slave1'}
3
4 ▼     stages {
5         stage('SCM_Checkout') {
6             steps {
7                 git 'https://github.com/maulik2311/capstone-health.git'
8             }
9         }
10        stage('App_Build_mvn') {
11            steps {
12                sh 'mvn clean package'
13            }
14        }
15        stage('Docker_Image_build') {
16            steps {
17                sh "docker build -t maulikd2397/health:${BUILD_NUMBER} ."
18                sh "docker tag maulikd2397/health:${BUILD_NUMBER} maulikd2397/health:latest "
19                sh 'docker images'
20            }
21        }
22    }
23}
```

Image build failed:

```
+ docker build -t maulikd2397/health:7 .
DEPRECATED: The legacy builder is deprecated and will be removed in a future release.
Install the buildx component to build images with BuildKit:
https://docs.docker.com/go/buildx/

permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock: Post
"http://<2Fvar%2Frun%2Fdocker.sock/v1.24/build?
buildargs=%7B%7D&cachefrom=%5B%5D&cgroupparent=&cuperiod=0&cpquota=0&cpusetcps=&cpusetmems=&cpushares=0&dockerfile=Dockerfile&labels=%7B%7D&memory=0&memswap=0&networkmode=default&rm=1&shmsize=0&t=maulikd2397%2Fhealth%3A7&target=&ulimits=null&version=1": dial unix /var/run/docker.sock: connect: permission denied
[Pipeline]
[Pipeline] // stage
[Pipeline]
[Pipeline] // node
[Pipeline] End of Pipeline
ERROR: script returned exit code 1
Finished: FAILURE
```

Reason:

```
docker:x:122:
devopsadmin:x:1001:
devopsadmin@ip-10-0-1-10:~/workspace/Health_care/target$
```

Add user to docker group

```
root@ip-10-0-1-10:~# usermod -aG docker devopsadmin
```

```
docker:x:122:devopsadmin
devopsadmin:x:1001:
root@ip-10-0-1-10:~#
```

Build Success:

```
Successfully tagged maulikd2397/health:11
[Pipeline] sh
+ docker tag maulikd2397/health:11 maulikd2397/health:latest
[Pipeline] sh
+ docker images
REPOSITORY          TAG        IMAGE ID      CREATED       SIZE
maulikd2397/health  11         9f85a618b030  1 second ago  695MB
maulikd2397/health  latest     9f85a618b030  1 second ago  695MB
openjdk             11         47a932d998b7  23 months ago  654MB
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Stage-4: Login2Dockerhub

Pipeline- Syntax

The screenshot shows the Jenkins Pipeline Syntax Snippet Generator interface. On the left, there's a sidebar with links like Snippet Generator, Declarative Directive Generator, Declarative Online Documentation, Steps Reference, Global Variables Reference, Online Documentation, Examples Reference, and IntelliJ IDEA GSLS. The main area has a title 'Overview' and a description explaining the Snippet Generator's purpose: to help learn Pipeline Script code by generating statements for specific steps. Below this is a 'Sample Step' section containing the 'withCredentials' step. A dropdown menu next to it shows 'Bind credentials to variables'. The 'Bindings' section below contains two entries: 'Secret text' and 'Variable', both with their respective help icons.

Pipeline-syntax

This screenshot is identical to the one above, showing the Jenkins Pipeline Syntax Snippet Generator. The 'withCredentials' step is selected, and the 'Variable' dropdown now shows the value 'dockerhublogin'. The rest of the interface, including the sidebar and the 'Bindings' section, remains the same.

Credentials: Secret=Dockerhub password

Jenkins Credentials Provider: Jenkins

Domain

Global credentials (unrestricted)

Kind

Secret text

Scope ?

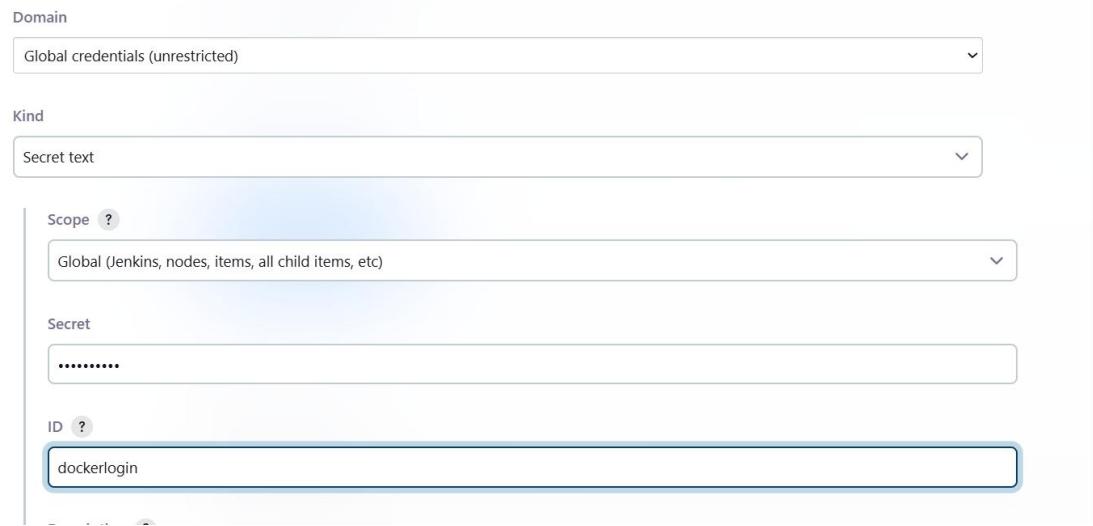
Global (Jenkins, nodes, items, all child items, etc)

Secret

.....

ID ?

dockerlogin



Generate pipeline syntax:

Dashboard > Health_care > Pipeline Syntax

Bindings

Secret text ?

Variable ?

dockerhublogin

Credentials ?

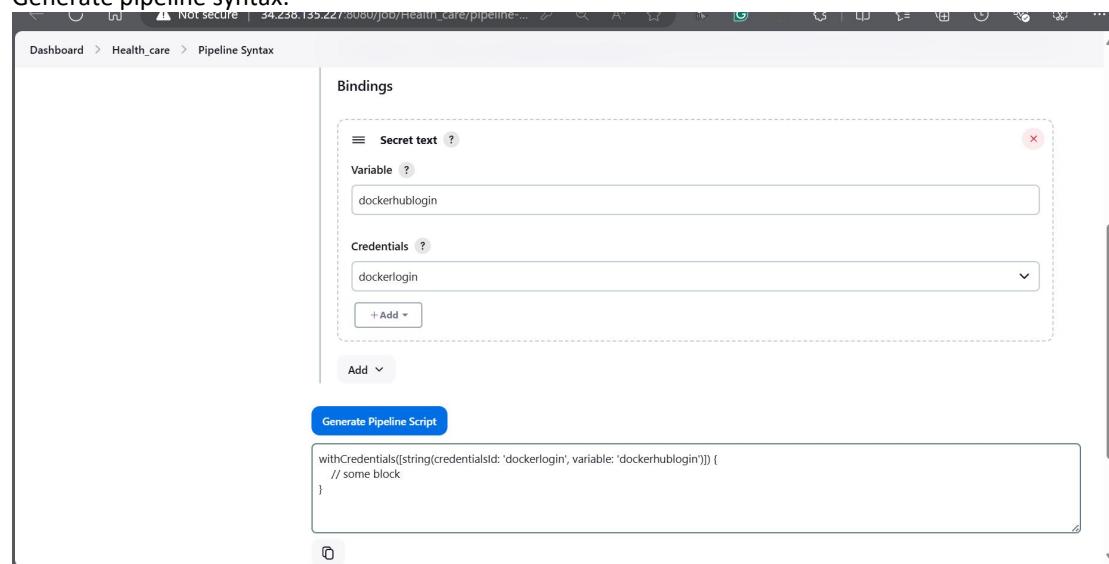
dockerlogin

+ Add ▾

Add ▾

Generate Pipeline Script

```
withCredentials([string(credentialsId: 'dockerlogin', variable: 'dockerhublogin')) {  
    // some block  
}
```



Copy to pipeline script:

```
Script ?  
1~ pipeline {  
2~   agent {label 'slave1'}  
3~  
4~   stages {  
5~     stage('SCM_Checkout') {  
6~       steps {  
7~         git 'https://github.com/maulik2311/capstone-health.git'  
8~       }  
9~     }  
10~    stage('App_Build_mvn') {  
11~      steps {  
12~        sh 'mvn clean package'  
13~      }  
14~    }  
15~    stage('Docker_Image_build') {  
16~      steps {  
17~        sh "docker build -t maulikd2397/health:${BUILD_NUMBER} ."  
18~        sh "docker tag maulikd2397/health:${BUILD_NUMBER} maulikd2397/health:latest "  
19~        sh 'docker images'  
20~      }  
21~    }  
22~    stage('Login2Dockerhub') {  
23~      steps {  
24~        withCredentials([string(credentialsId: 'dockerlogin', variable: 'dockerhublogin')]) {  
25~          sh "docker login -u maulikd2397 -p ${dockerhublogin}"  
26~        }  
27~      }  
28~    }  
29~  }  
30~ }
```

<https://docs.docker.com/engine/reference/commandline/login/#credentials-store>

```
Login Succeeded  
[Pipeline] }  
[Pipeline] // withCredentials  
[Pipeline] }  
[Pipeline] // stage  
[Pipeline] }  
[Pipeline] // node  
[Pipeline] End of Pipeline  
Finished: SUCCESS
```

Stage-5: “Push2Dockerhub”

```
Script ?  
19~   sh 'docker images'  
20~ }  
21~ }  
22~ stage('Login2Dockerhub') {  
23~   steps {  
24~     withCredentials([string(credentialsId: 'dockerlogin', variable: 'dockerhublogin')]) {  
25~       sh "docker login -u maulikd2397 -p ${dockerhublogin}"  
26~     }  
27~   }  
28~ }  
29~ stage('Push2Dockerhub') {  
30~   steps {  
31~     sh 'docker push maulikd2397/health:latest '  
32~   }  
33~ }  
34~ }  
35~ }  
36~ }
```

Pushed

```
9db07f618750: Pushed  
latest: digest: sha256:5b4bd101880fe39d062272d927e0c9afbd495b1ab1a91799b9ad9020a2c7e631 size: 2007  
[Pipeline] }  
[Pipeline] // stage  
[Pipeline] }  
[Pipeline] // node  
[Pipeline] End of Pipeline  
Finished: SUCCESS
```

Dockerhub:

The screenshot shows a Dockerhub repository page for 'maulikd2397/health'. The 'General' tab is selected. The repository has one tag, 'latest', which was pushed a few seconds ago. The 'Automated Builds' section is visible on the right.

Stage-6: "k8sdeployment"

For Automated Deployments on k8s cluster, k8s master should be integrate with Jenkins master:

Steps:

ssh connection.

Kubernetes cluster already configured with user and user .kube configuration

```
devopsadmin@kmaster-node:~/.ssh$ kubectl get node
NAME      STATUS   ROLES      AGE      VERSION
kmaster-node  Ready    control-plane  24d     v1.29.6
worker-node1  Ready    <none>    24d     v1.29.6
worker-node2  Ready    <none>    24d     v1.29.6
devopsadmin@kmaster-node:~/.ssh$ ls
authorized_keys  id_ecdsa  id_ecdsa.pub
devopsadmin@kmaster-node:~/.ssh$ ls
authorized_keys  id_ecdsa  id_ecdsa.pub
devopsadmin@kmaster-node:~/.ssh$
```

➤ Install plugin and restart Jenkins

The screenshot shows the Jenkins 'Manage Jenkins > Plugins' page. The 'Available plugins' tab is selected. The 'Publish Over SSH' plugin by Maulik DEVANI is listed, with the 'Install' button highlighted.

➤ Add ssh servers from System.

Dashboard > Manage Jenkins > System >

Path to key ?

Key ?

Disable exec ?

SSH Servers

Add

Advanced ▾

Save Apply

REST API Jenkins 2.452.3

Dashboard > Manage Jenkins > System >

≡ SSH Server

Name ?
kuscluster

The name of this configuration. This will appear in the drop down list in the job configuration.
(from: Publish Over SSH)

Hostname ?
172.31.18.18

Username ?
devopsadmin

Remote Directory ?
/home/devopsadmin

Avoid sending files that have not changed ?

Advanced ^ Edited

Use password authentication, or use a different key ?
Passphrase / Password ?
Path to key ?
Key ?

```
f9jUSeRtZeNsZfjn8FW+AWxtx9HG5o0nADDl93+T8efyQRHcDL49adJPobBZ  
wakNY0VgBEqKhZQTYLkO2MKVOc0lQz7YikNCB3cAAAQgGaCbr4K/qDtaV+YA3QV  
p2exS9IMimio/qemjAzCwtVHp5nOA910kbsVORSV0QLRBMejoKILFFz05NLeYU6lQAAAB  
hkZXzcInhZG1pbkBrbWFzdGVyLW5vZGUBAg==  
-----END OPENSSH PRIVATE KEY-----
```

Proxy user ?
Proxy password

Success Test Configuration

Add Advanced ▾

Save Apply

Test= Success

Pipeline:

Dashboard > Health_care > Configuration

Configure

- General
- Advanced Project Options
- Pipeline

Pipeline script

```

1/      sh 'docker build -t maulikd2397/health:${BUILD_NUMBER} '
2/      sh 'docker tag maulikd2397/health:${BUILD_NUMBER} maulikd2397/health:latest '
3/      }
4/  }
5/  }
6/  }
7/  }
8/  }
9/  }
10/ }
11/ }
12/ }
13/ }
14/ }
15/ }
16/ }
17/ }
18/ }
19/ }
20/ }
21/ }
22/ }
23/ }
24/ }
25/ }
26/ }
27/ }
28/ }
29/ }
30/ }
31/ }
32/ }
33/ }
34/ }
35/ }
36/ }
37/ }
38/ }
39/ }
40/ }
41/ }

```

Use Groovy Sandbox

Pipeline Syntax ←

Save **Apply**

Pipeline syntax:

Dashboard > Health_care > Pipeline Syntax

Sample Step

Online Documentation

Examples Reference

IntelliJ IDEA GDSL

sshPublisher: Send build artifacts over SSH

SSH Publishers

SSH Server

Name: kuscluster

Advanced

Transfers

Transfer Set

Source files: k8sdeploy.yaml

Remove prefix:

Transfer Set

Source files: k8sdeploy.yaml

Remove prefix:

Remote directory: .

Exec command: `kubectl apply -f k8sdeploy.yaml`

All of the transfer fields (except for Exec timeout) support substitution of Jenkins environment variables

Advanced

Add Transfer Set

Generate Pipeline Script

```

sshPublisher(publishers: [sshPublisherDescriptor(configName: 'kuscluster', transfers: [sshTransfer(cleanRemote: false, excludes: '', execCommand: 'kubectl apply -f k8sdeploy.yaml', execTimeout: 120000, flatten: false, makeEmptyDirs: false, noDefaultExcludes: false, patternSeparator: '[.]', remoteDirectory: '.', remoteDirectoryName: 'k8sdeploy', removePrefix: '', sourceFiles: 'k8sdeploy.yaml'), usePromotionTimestamp: false, useWorkspaceInPromotion: false, verbose: false])])

```

Pipeline:

Copy pipeline syntax and tun the job.

```
Script ?  
13 }  
14 }  
15 } stage('Docker_Image_build') {  
16 steps {  
17 sh "docker build -t maulikd2397/health:${BUILD_NUMBER} ."  
18 sh "docker tag maulikd2397/health:${BUILD_NUMBER} maulikd2397/health:latest "  
19 sh 'docker images'  
20 }  
21 }  
22 } stage('Login2Dockerhub') {  
23 steps {  
24 withCredentials([string(credentialsId: 'dockerlogin', variable: 'dockerhublogin')) {  
25 sh "docker login -u maulikd2397 -p ${dockerhublogin}"  
26 }  
27 }  
28 }  
29 } stage('Push2Dockerhub') {  
30 steps {  
31 sh 'docker push maulikd2397/health:latest '  
32 }  
33 }  
34 } stage('k8sdeployment') {  
35 steps {  
36 sshPublisher(publishers: [sshPublisherDesc(configName: 'kuscluster', transfers: [sshTransfer(cleanRemote: fa  
37 ])  
38 }  
39 }  
40 }  
41 }
```

Deployment file:

```
1 apiVersion: apps/v1  
2 kind: Deployment  
3 metadata:  
4 name: health-deployment  
5 labels:  
6 app: app  
7 spec:  
8 replicas: 3  
9 selector:  
10 matchLabels:  
11 app: app  
12 template:  
13 metadata:  
14 labels:  
15 app: app  
16 spec:  
17 containers:  
18 - name: c001  
19 image: maulikd2397/health  
20 ports:  
21 - containerPort: 8082  
22  
23  
24 ...  
25  
26 apiVersion: v1  
27 kind: Service  
28 metadata:  
29 name: health-service  
30 labels:  
31 app: app  
32 spec:  
33 selector:  
34 app: app  
35 type: NodePort  
36 ports:  
37 - nodePort: 31231  
38 port: 80  
39 targetPort: 8082
```

Deployment success

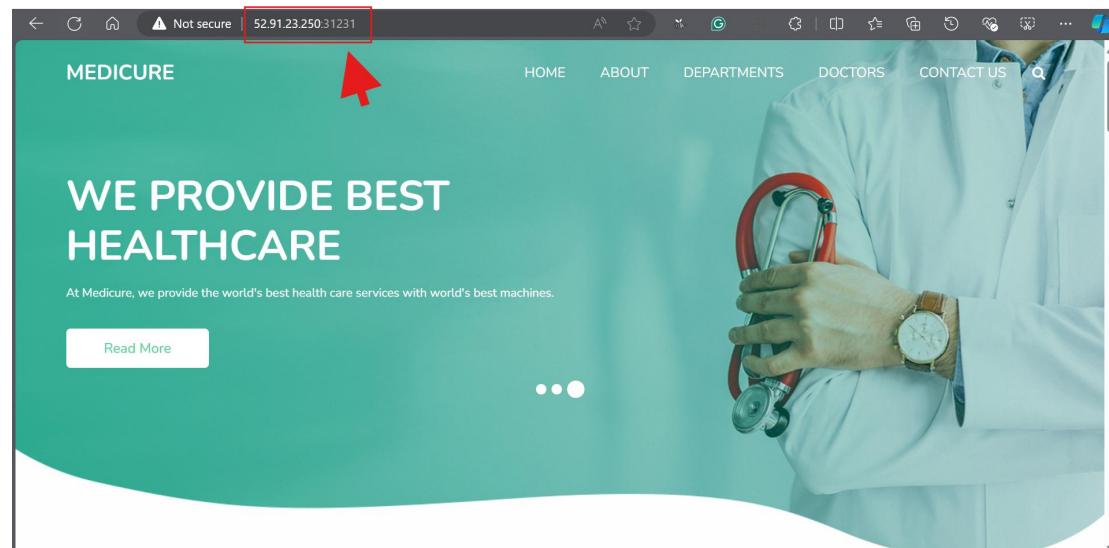
```
devopsadmin@kmaster-node:~/ssh$ kubectl get all  
NAME READY STATUS RESTARTS AGE  
pod/health-deployment-677f4db968-hdv2g 1/1 Running 0 41s  
pod/health-deployment-677f4db968-rfh45 1/1 Running 0 41s  
pod/health-deployment-677f4db968-t85v9 1/1 Running 0 41s  
  
NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE  
service/finance-svc-nodeport NodePort 10.111.112.91 <none> 8081:31434/TCP 5d1h  
service/health-service NodePort 10.97.6.71 <none> 80:31231/TCP 41s  
service/kubernetes ClusterIP 10.96.0.1 <none> 443/TCP 24d  
  
NAME READY UP-TO-DATE AVAILABLE AGE  
deployment.apps/health-deployment 3/3 3 3 41s  
  
NAME DESIRED CURRENT READY AGE  
replicaset.apps/health-deployment-677f4db968 3 3 3 41s
```

Access the application.

```
devopsadmin@kmaster-node:~/ssh$ kubectl get pods -o wide
NAME           READY   STATUS    RESTARTS   AGE     IP          NODE      NOMINATED NODE   READINESS GATES
health-deployment-677f4db968-hdv2g  1/1    Running   0          2m16s  10.244.1.87  worker-node1  <none>        <none>
health-deployment-677f4db968-rfh45  1/1    Running   0          2m16s  10.244.2.89  worker-node2  <none>        <none>
health-deployment-677f4db968-t85v9  1/1    Running   0          2m16s  10.244.2.88  worker-node2  <none>        <none>
devopsadmin@kmaster-node:~/ssh$
```

Worker-Node1

Instance summary for i-0c0bd31ead82803c4 (KNode1) Info	
Updated less than a minute ago	
Instance ID i-0c0bd31ead82803c4 (KNode1)	Public IPv4 address 52.91.23.250 open address
IPv6 address -	Instance state Running
Hostname type IP name: ip-172-31-18-255.ec2.internal	Private IP DNS name (IPv4 only) ip-172-31-18-255.ec2.internal
Answer private resource DNS name IPv4 (A)	Instance type t2.micro
Auto-assigned IP address 52.91.23.250 [Public IP]	VPC ID vpc-09d289602571b4bf6 (default)
IAM Role -	Subnet ID subnet-0290daab69c5138cb
IMDSv2 Required	Instance ARN arn:aws:ec2:us-east-1:891377051774:instance/i-0c0bd31ead82803c4
Private IPv4 addresses 172.31.18.255	
Public IPv4 DNS ec2-52-91-23-250.compute-1.amazonaws.com open address	
Elastic IP addresses -	
AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. Learn more	
Auto Scaling Group name -	



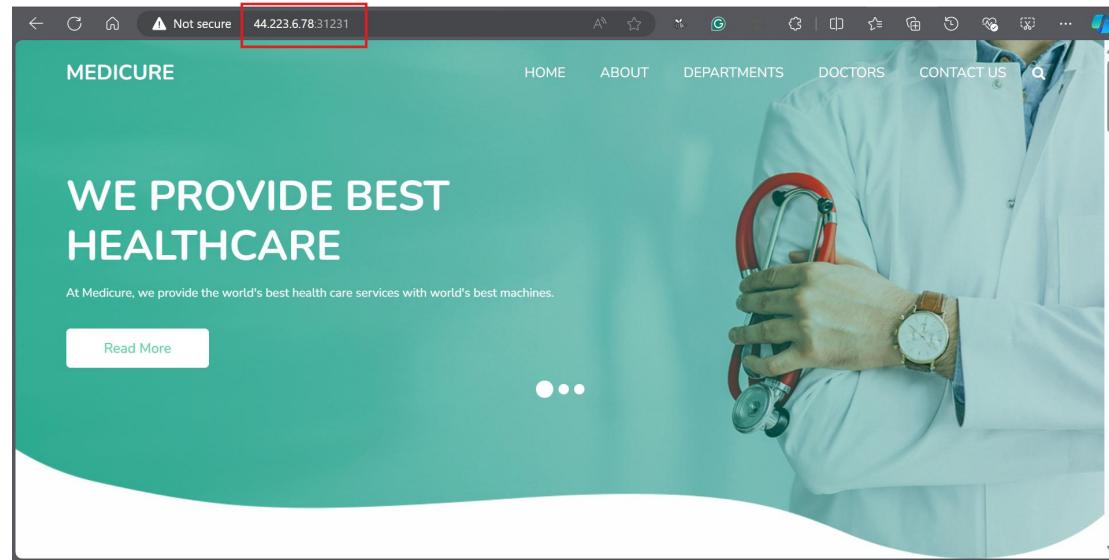
Worker Node-2

EC2 > Instances > i-01589c20994d7ea65

Instance summary for i-01589c20994d7ea65 (KNode2) [Info](#)

Updated less than a minute ago

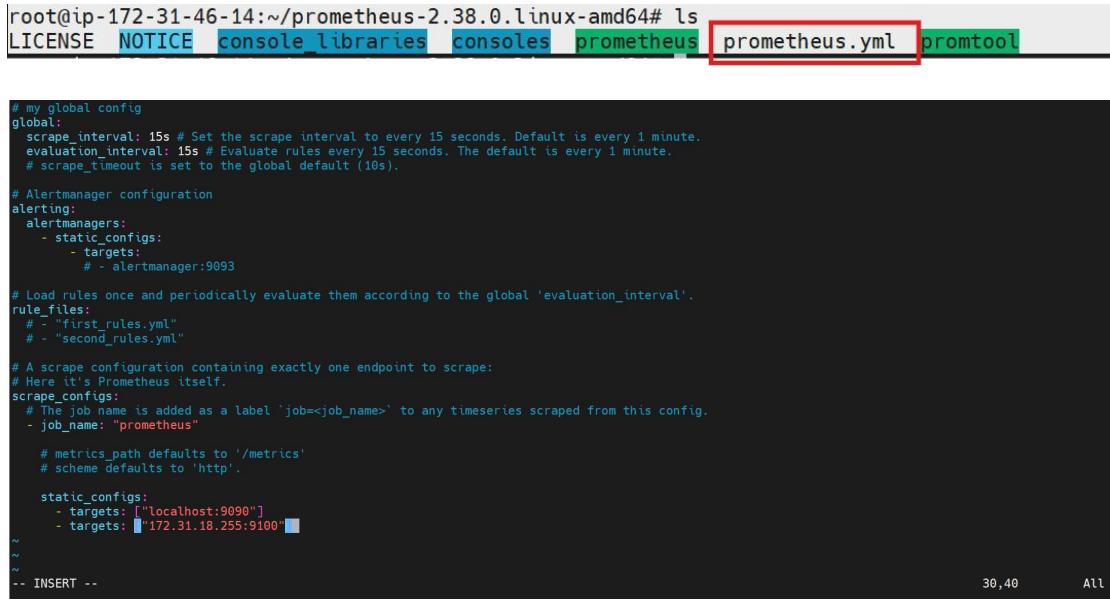
Instance ID i-01589c20994d7ea65 (KNode2)	Public IPv4 address 44.223.6.78 open address	Private IPv4 addresses 172.31.27.146
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-44-223-6-78.compute-1.amazonaws.com open address
Hostname type IP name: ip-172-31-27-146.ec2.internal	Private IP DNS name (IPv4 only) ip-172-31-27-146.ec2.internal	Elastic IP addresses -
Answer private resource DNS name IPv4 (A)	Instance type t2.micro	AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. Learn more
Auto-assigned IP address 44.223.6.78 [Public IP]	VPC ID vpc-09d289602571b4bf6 (default)	Auto Scaling Group name -
IAM Role -	Subnet ID subnet-0290daab69c5138cb	
IMDSv2 Required	Instance ARN arn:aws:ec2:us-east-1:891377051774:instance/i-01589c20994d7ea65	



High availability

Adding server to the target of prometheus: Node Exporter

```
root@ip-172-31-46-14:~/prometheus-2.38.0.linux-amd64# ls
LICENSE NOTICE console_libraries consoles prometheus prometheus.yml promtool
```



```
# my global config
global:
  scrape_interval: 15s # Set the scrape interval to every 15 seconds. Default is every 1 minute.
  evaluation_interval: 15s # Evaluate rules every 15 seconds. The default is every 1 minute.
  # scrape_timeout is set to the global default (10s).

  # Alertmanager configuration
  alerting:
    alertmanagers:
      - static_configs:
          - targets:
              # - alertmanager:9093

  # Load rules once and periodically evaluate them according to the global 'evaluation_interval'.
  rule_files:
    # - "first_rules.yml"
    # - "second_rules.yml"

  # A scrape configuration containing exactly one endpoint to scrape:
  # Here it's Prometheus itself.
  scrape_configs:
    # The job name is added as a label 'job=<job_name>' to any timeseries scraped from this config.
    - job_name: "prometheus"

      # metrics_path defaults to '/metrics'
      # scheme defaults to 'http'.

      static_configs:
        - targets: ["localhost:9090"]
        - targets: [172.31.18.255:9100]
```

30,40 All

Installing NodeExporter on target server:

```
root@worker-node1:~# wget https://github.com/prometheus/node_exporter/releases/download/v1.4.0
rc.0/"Cde_exporter-1.4.0-rc.0.linux-amd64.tar.gz
root@worker-node1:~# wget https://github.com/prometheus/node_exporter/releases/download/v1.8.1/node_exporter-1.8.1.linux-amd64.tar.gz
--2024-07-13 16:06:28-- https://github.com/prometheus/node_exporter/releases/download/v1.8.1/node_exporter-1.8.1.linux-amd64.tar.gz
Resolving github.com (github.com)... 140.82.113.4:443... connected.
Connecting to github.com (github.com)|140.82.113.4|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://objects.githubusercontent.com/github-production-release-asset-2e65be/9524057/e07e4ee4-e4b0-48dc-9c04-eaad890c81b3?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetprod032f20240713%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20240713T160628Z&X-Amz-Expires=300&X-Amz-Signature=e78a708085751f10b16760b02393201ed39cd426f9960360e8a9576b71badab8&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=9524057&response-content-disposition=attachment%3Bfilename%3Dnode_exporter-1.8.1.linux-amd64.tar.gz&response-content-type=application%2Foctet-stream [following]
--2024-07-13 16:06:28-- https://objects.githubusercontent.com/github-production-release-asset-2e65be/9524057/e07e4ee4-e4b0-48dc-9c04-eaad890c81b3?X-Amz-Algorithm=AWS4-HMAC-SHA256&X-Amz-Credential=releaseassetprod032f20240713%2Fus-east-1%2Fs3%2Faws4_request&X-Amz-Date=20240713T160628Z&X-Amz-Expires=300&X-Amz-Signature=e78a708085751f10b16760b02393201ed39cd426f9960360e8a9576b71badab8&X-Amz-SignedHeaders=host&actor_id=0&key_id=0&repo_id=9524057&response-content-disposition=attachment%3Bfilename%3Dnode_exporter-1.8.1.linux-amd64.tar.gz&response-content-type=application%2Foctet-stream
Resolving objects.githubusercontent.com (objects.githubusercontent.com)... 185.199.111.133, 185.199.108.133, 185.199.109.133, ...
Connecting to objects.githubusercontent.com (objects.githubusercontent.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 10672684 (10M) [application/octet-stream]
Saving to: 'node_exporter-1.8.1.linux-amd64.tar.gz'

node_exporter-1.8.1.linux-amd64.tar.gz 100%[=====] 10.18M --.-KB/s   in 0.07s

2024-07-13 16:06:28 (152 MB/s) - 'node_exporter-1.8.1.linux-amd64.tar.gz' saved [10672684/10672684]

root@worker-node1:~# ls
node_exporter-1.8.1.linux-amd64.tar.gz  snap
root@worker-node1:~# tar -xvf node_exporter-1.8.1.linux-amd64.tar.gz
node_exporter-1.8.1.linux-amd64/
node_exporter-1.8.1.linux-amd64/NOTICE
node_exporter-1.8.1.linux-amd64/LICENSE
node_exporter-1.8.1.linux-amd64/node_exporter
```

Adding service file and start node exporter:

```
vi /etc/systemd/system/node_exporter.service
```

[Unit]

Description=Prometheus Server

Documentation=https://prometheus.io/docs/introduction/overview/

After=network-online.target

[Service]

User=root

Restart=on-failure

```
ExecStart=/root/node_exporter-1.8.1.linux-amd64/node_exporter
```

[Install]

WantedBy=multi-user.target

Restart:

```
root@worker-node1:~/node_exporter-1.8.1.linux-amd64# vi /etc/systemd/system/node_exporter.service
root@worker-node1:~/node_exporter-1.8.1.linux-amd64# systemctl daemon-reload
root@worker-node1:~/node_exporter-1.8.1.linux-amd64# systemctl start node_exporter
root@worker-node1:~/node_exporter-1.8.1.linux-amd64# systemctl status node_exporter
● node_exporter.service - Prometheus Server
   Loaded: loaded (/etc/systemd/system/node_exporter.service; disabled; vendor preset: enabled)
     Active: active (running) since Sat, 2024-07-13 16:10:25 UTC; 5s ago
       Docs: https://prometheus.io/docs/introduction/overview/
   Main PID: 36403 (node_exporter)
     Tasks: 3 (limit: 1120)
    Memory: 2.0M
      CPU: 7ms
     CGroup: /system.slice/node_exporter.service
             └─36403 /root/node_exporter-1.8.1.linux-amd64/node_exporter

Jul 13 16:10:25 worker-node1 node_exporter[36403]: ts=2024-07-13T16:10:25.129Z caller=node_exporter.go:118 level=info collector=time
Jul 13 16:10:25 worker-node1 node_exporter[36403]: ts=2024-07-13T16:10:25.129Z caller=node_exporter.go:118 level=info collector=timex
Jul 13 16:10:25 worker-node1 node_exporter[36403]: ts=2024-07-13T16:10:25.129Z caller=node_exporter.go:118 level=info collector=udp_queues
Jul 13 16:10:25 worker-node1 node_exporter[36403]: ts=2024-07-13T16:10:25.129Z caller=node_exporter.go:118 level=info collector=uname
Jul 13 16:10:25 worker-node1 node_exporter[36403]: ts=2024-07-13T16:10:25.129Z caller=node_exporter.go:118 level=info collector=vmstat
Jul 13 16:10:25 worker-node1 node_exporter[36403]: ts=2024-07-13T16:10:25.129Z caller=node_exporter.go:118 level=info collector=watchdog
Jul 13 16:10:25 worker-node1 node_exporter[36403]: ts=2024-07-13T16:10:25.129Z caller=node_exporter.go:118 level=info collector=xfs
Jul 13 16:10:25 worker-node1 node_exporter[36403]: ts=2024-07-13T16:10:25.129Z caller=node_exporter.go:118 level=info collector=zfs
Jul 13 16:10:25 worker-node1 node_exporter[36403]: ts=2024-07-13T16:10:25.130Z caller=tls_config.go:313 level=info msg="Listening on " address=:9100
Jul 13 16:10:25 worker-node1 node_exporter[36403]: ts=2024-07-13T16:10:25.130Z caller=tls_config.go:316 level=info msg="TLS is disabled." http2=false ad
lines 1-21 (END)
```

Node Exporter

Prometheus Node Exporter

Version: (version=1.8.1, branch=HEAD, revision=400c3979931613db930ea035f39ce7b377cd8b5b)

- Metrics

Series	Value
up{instance="172.31.18.255:9100", job="prometheus"}	1
up{instance="localhost:9090", job="prometheus"}	1

Target server added

Prometheus

Alerts Graph Status Help

Use local time Enable query history Enable autocomplete Enable highlighting Enable linter

up

Evaluation time

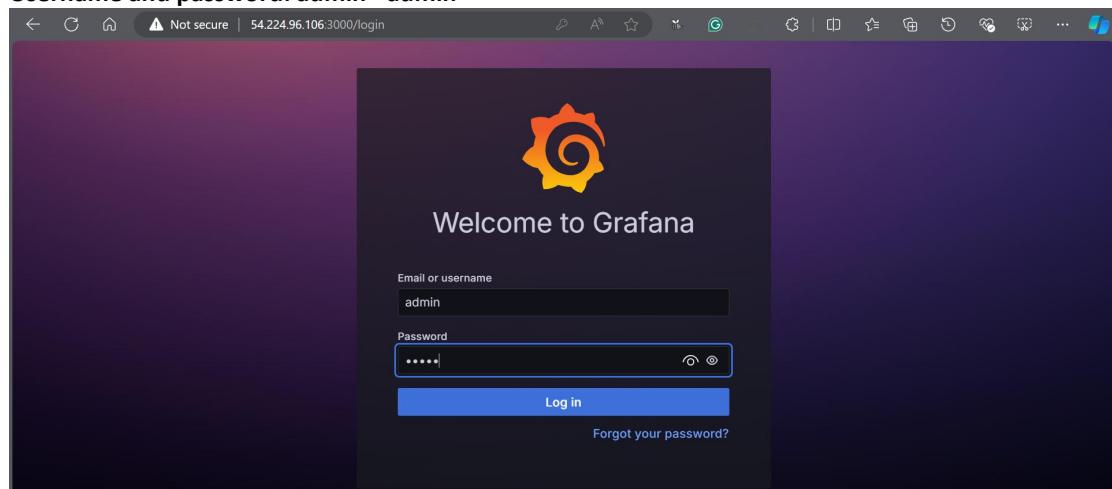
Evaluation time	Value	Series
	1	up{instance="172.31.18.255:9100", job="prometheus"}
	1	up{instance="localhost:9090", job="prometheus"}

Load time: 132ms Resolution: 14s Result series: 2

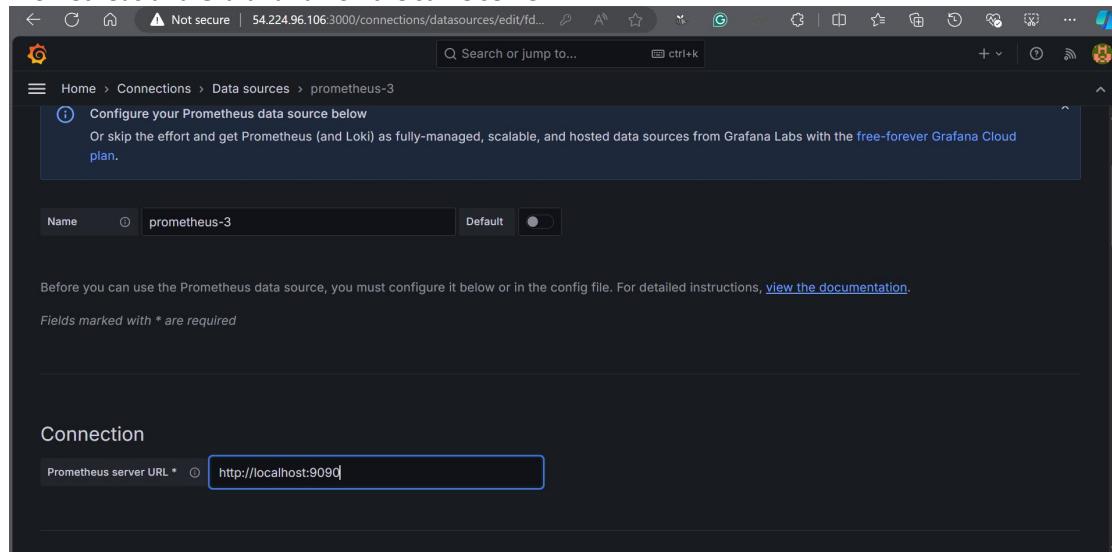
Add Panel Remove Panel

Login to Grafana:

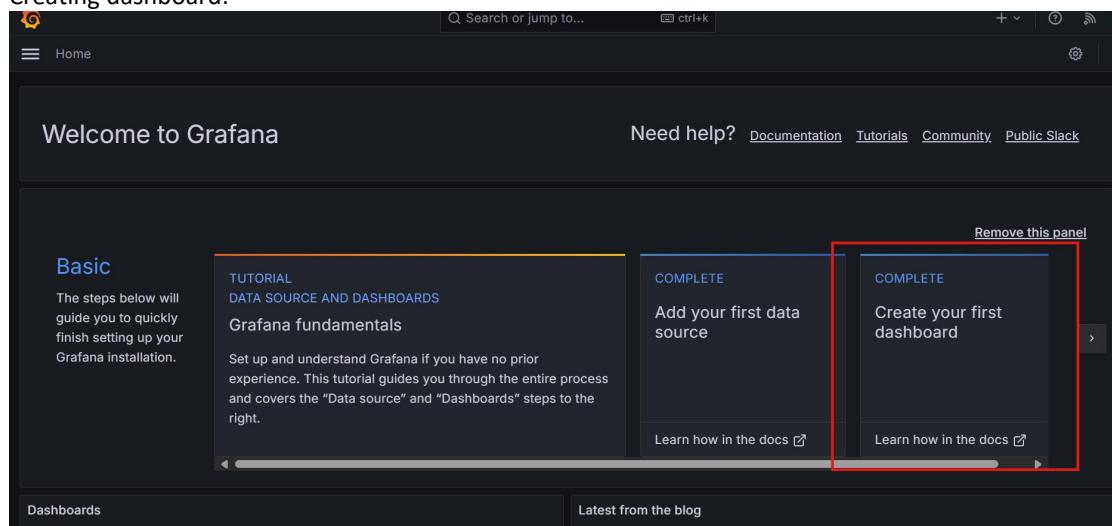
Username and password: admin - admin



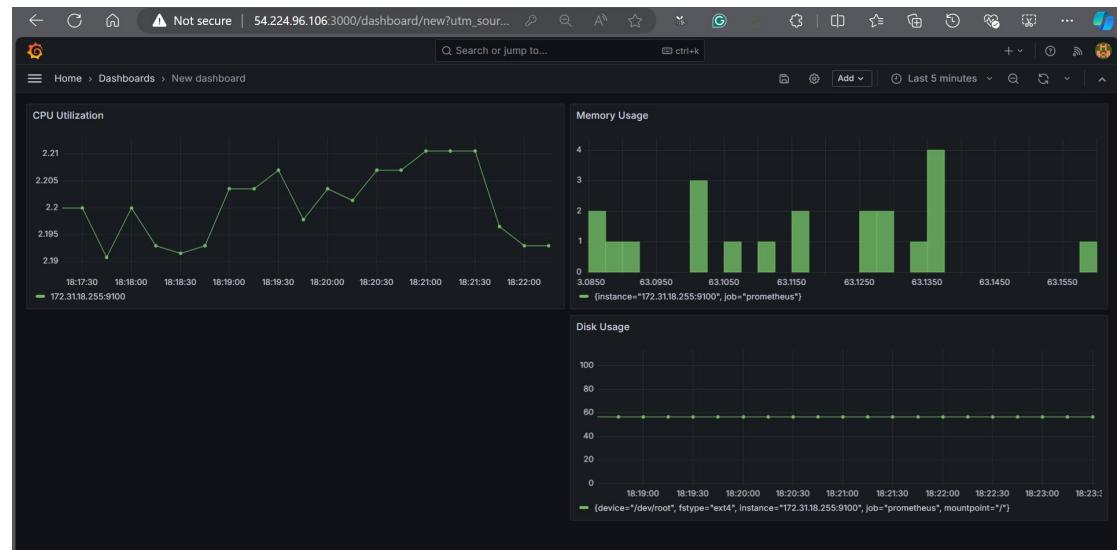
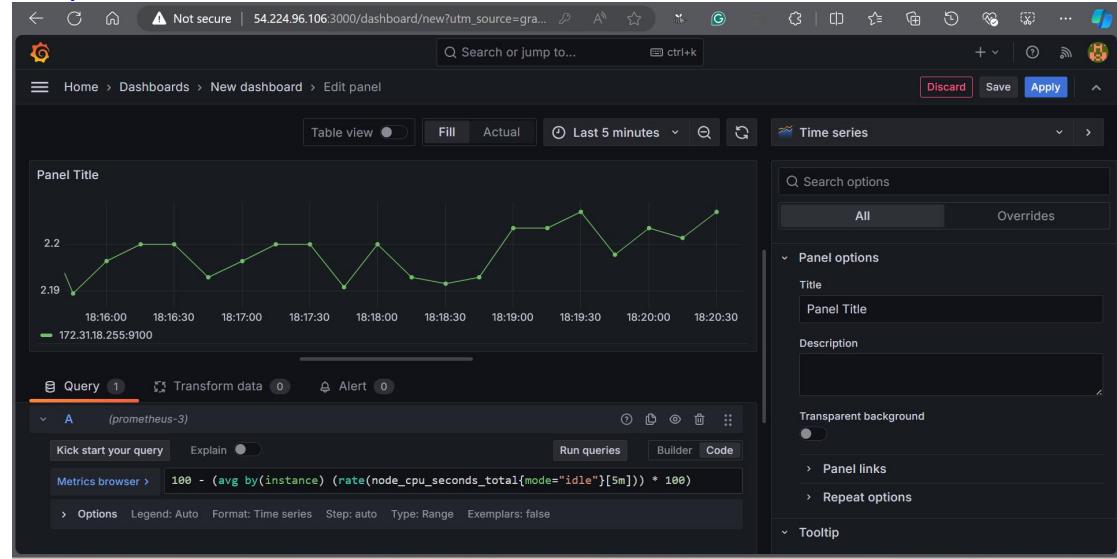
Prometheus and Grafana run on the same server:



Creating dashboard:



Run queries:



Jenkins pipeline script:

```
pipeline {
    agent {label 'slave1'}

    stages {
        stage('SCM_Checkout') {
            steps {
                git 'https://github.com/maulik2311/capstone-health.git'
            }
        }
        stage('App_Build_mvn') {
            steps {
                sh 'mvn clean package'
            }
        }
        stage('Docker_Image_build') {
            steps {
                sh "docker build -t maulikd2397/health:${BUILD_NUMBER} ."
                sh "docker tag maulikd2397/health:${BUILD_NUMBER} maulikd2397/health:latest "
                sh 'docker images'
            }
        }
        stage('Login2Dockerhub') {
            steps {
                withCredentials([string(credentialsId: 'dockerlogin', variable: 'dockerhublogin')]) {
                    sh "docker login -u maulikd2397 -p ${dockerhublogin}"
                }
            }
        }
        stage('Push2Dockerhub') {
            steps {
                sh 'docker push maulikd2397/health:latest '
            }
        }
        stage('k8sdeployment') {
            steps {
                sshPublisher(publishers: [sshPublisherDesc(configName: 'kuscluster', transfers: [
                    sshTransfer(cleanRemote: false, excludes: "", execCommand: 'kubectl apply -f k8sdeploy.yaml',
                    execTimeout: 120000, flatten: false, makeEmptyDirs: false, noDefaultExcludes: false,
                    patternSeparator: '[, ]+', remoteDirectory: '.', remoteDirectorySDF: false, removePrefix: '',
                    sourceFiles: 'k8sdeploy.yaml')], usePromotionTimestamp: false, useWorkspaceInPromotion: false, verbose: false)])
            }
        }
    }
}
```

% % % % % % *Thank you* % % % % % %