Department of Information Technology

T.E / I.T / Sem V/

ITL503: DevOps Lab Journal

Submitted By

Ms. Sanchita Warade

Roll No: 59

Don Bosco Institute of Technology,
Mumbai 400070.

(Affiliated to the University of Mumbai)

Table Of Contents

Name : Sanchita Warade
Roll No.59
Batch : C

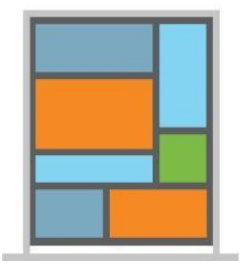
Experiment A - Self Study

Aim : is to understand the need and requirement of DevOps in the Industries.

Question :

What do you mean by Microservice?

Microservices describes the architectural process of building a distributed application from separatelydeployable services that perform specific business functions and communicate overweb interfaces.



What's meant by a lightweight server?

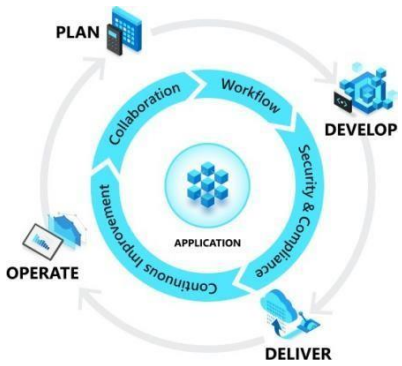
Lightweight servers are designed to be lean and efficient, consuming fewer resources than their traditionalcounterparts. The philosophy behind these servers is to provide only the essential functionality necessary for specific tasks, eliminating any superfluous elements thatmay consume unnecessary resources.

What is meant by CI/CD?

CI And CD is the practice of automating the integration of code changes from multiple developers into a single codebase. It is a software development practice where the developerscommit their work frequentlyto the central code repository (Github or Stash).

What is DevOps?

DevOps is a software development process that combines development (Dev) and operations (Ops) to unite people, process, and technology in application planning, development, delivery, and operations. DevOps enables coordination and collaboration.



What are the stages of Devops ?

The key phases of the DevOps lifecycle:

Continuous feedback
Discover Plan Build Test Monitor Operate

What are the various tools that are used at every stage ?Discover

Jira Product Discovery BETA MURAL miro

Plan

Jira

Build

kubernetes docker

Test

Monitor

Operate

Continuous feedback

What do you mean by versioning?

The sole purpose of version control in DevOps is**to keep a tab on the history of all the code changes made over the course of the project life cycle**. This helps developers get a clear ideaof who made whatchanges and when as well as restore any previous version of a codebase ifneeded.

Rubrics: For Evaluation

With the hour & excellent	Within the next week	Within the semester
A++ to A	A maximum	B+
A++ = 10 A+=9 A=8	A=8 B++ = 7 B+= 6	B= 5

Conclusion : All the given questions were learned and successfully documented .

References:

1. **https://learn.microsoft.com/en-us/devops/develop/git/what-is-version-control**

2. **https://www.geeksforgeeks.org/what-is-ci-cd/**

3. **https://www.google.com/search?client=ubuntu&channel=fs&q=various+tool s+that+are+used+at+every+stage+od+devops**

Name : Sanchita Warade
Roll no.59
Batch : C

Experiment 1 - Git
Installation & Versioning

Aim: To install git (local repository) and synchronize with github (remote repository) and perform version controlling.

Steps for installation and version control:

git config
Usage: git config –global user.name “[name]”
Usage: git config –global user.email “[email address]”
This command sets the author name and email address respectively to be used with your commits.

```
sanch@LAPTOP-O06HP269 MINGW64 ~
$ git config --global user.name "sanchitavarade"

sanch@LAPTOP-O06HP269 MINGW64 ~
$ git config --global user.email "varadesanchita@gmail.com"

sanch@LAPTOP-O06HP269 MINGW64 ~
$ git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=D:/Git/mingw64/etc/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=false
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=main
user.email=varadesanchita@gmail.com
user.name=sanchitavarade
```

git init

```
sanch@LAPTOP-O06HP269 MINGW64 ~
$ mkdir git-devops

sanch@LAPTOP-O06HP269 MINGW64 ~
$ git init
Initialized empty Git repository in C:/Users/sanch/.git/

sanch@LAPTOP-O06HP269 MINGW64 ~ (main)
```

Usage: git init [repository name]To make git clone

Usage: git clone [url]
This command is used to obtain a repository from an existing URL.

```
sanch@LAPTOP-O06HP269 MINGW64 ~/git
$ git clone "https://github.com/ShaikhMasud/Academix.git"
Cloning into 'Academix'...
remote: Enumerating objects: 1359, done.
remote: Counting objects: 100% (230/230), done.
remote: Compressing objects: 100% (146/146), done.
remote: Total 1359 (delta 116), reused 184 (delta 84), pack-reused 1129 (from 1)
Receiving objects: 100% (1359/1359), 29.06 MiB | 1.87 MiB/s, done.
Resolving deltas: 100% (723/723), done.
```

git add
Usage: git add [file]
This command adds a file to the staging area.

Usage: git add *
This command adds one or more to the staging area.

```
sanch@LAPTOP-O06HP269 MINGW64 ~/git-devops (main)
$ touch file.txt

sanch@LAPTOP-O06HP269 MINGW64 ~/git-devops (main)
$ git add *
```

git commit
Usage: git commit -m "[Type in the commit message]"
This command records or snapshots the file permanently in the version history.

```
sanch@LAPTOP-O06HP269 MINGW64 ~/git-devops (main)
$ git commit -m "New msg"
[main (root-commit) 7ad1beb] New msg
 1 file changed, 0 insertions(+), 0 deletions(-)
 create mode 100644 file.txt
```

Usage: git commit -a
This command commits any files you've added with the git add command and also commits any files you've changed since then.

```
sanch@LAPTOP-O06HP269 MINGW64 ~/git-devops (main)
$ git commit -a
On branch main
nothing to commit, working tree clean
```

This command shows the file differences which are not yet staged.

Usage: git diff–staged
This command shows the differences between the files in the staging area and the latest version is present.

Usage: git diff[first branch] [second branch]
This command shows the differences between the two branches mentioned.

git reset
Usage: git reset [file]

This command unstages the file, but it preserves the file contents.Usage:

git reset[commit]
This command undoes all the commits after the specified commit and preserves the changes locally.

Usage: git reset –hard [commit] This command discards all history andgoesback to the specified commit.

Git status: usage
This command lists all the files that have to be committed.

```
sanch@LAPTOP-O06HP269 MINGW64 ~/git-devops (main)
$ git status
On branch main
nothing to commit, working tree clean
```

git rm
Usage: git rm [file]
This command deletes the file from your working directory and stages thedeletion.

git log
Usage: git log
This command is used to list the version history for the current branch.

```
sanch@LAPTOP-O06HP269 MINGW64 ~/git-devops (main)
$ git log
commit 7ad1bebba0f11dde365ecbd4a448fef7161245b6 (HEAD -> main)
Author: sanchitavarade <varadesanchita@gmail.com>
Date:   Sun Oct 27 19:48:24 2024 +0530

    New msg
```

Usage: git log –follow[file]
This command lists version history for a file, including the renaming of files also.git show
Usage: git show [commit]
This command shows the metadata and content changes of the specified commit.

```
sanch@LAPTOP-O06HP269 MINGW64 ~/git-devops (main)
$ git show
commit 7ad1bebba0f11dde365ecbd4a448fef7161245b6 (HEAD -> main)
Author: sanchitavarade <varadesanchita@gmail.com>
Date:   Sun Oct 27 19:48:24 2024 +0530

    New msg

diff --git a/file.txt b/file.txt
new file mode 100644
index 0000000..e69de29
```

git tag
Usage: git tag [commitID]
This command is used to give tags to the specific commit.

git branch Usage:
git branch
This command lists all the local branches in the current

repository.Usage: git branch [branch name]

```
sanch@LAPTOP-O06HP269 MINGW64 ~/git-devops (main)
$ git branch
* main
```

This command creates a new branch.

Usage: git branch -d [branch name]
This command deletes the feature branch.

```
sanch@LAPTOP-O06HP269 MINGW64 ~/git-devops (main)
$ git branch master

sanch@LAPTOP-O06HP269 MINGW64 ~/git-devops (main)
$ git branch -d master
Deleted branch master (was 7ad1beb).
```

git checkout

Usage: git checkout [branch name]

This command is used to switch from one branch to another.

```
sanch@LAPTOP-O06HP269 MINGW64 ~/git-devops (main)
$ git checkout main
Already on 'main'
```

Usage: git checkout -b [branch name]

This command creates a new branch and also switches to it.

```
sanch@LAPTOP-O06HP269 MINGW64 ~/git-devops (main)
$ git checkout -b master
Switched to a new branch 'master'
```

git merge

Usage: git merge [branch name]

This command merges the specified branch's history into the current branch.

```
sanch@LAPTOP-O06HP269 MINGW64 ~/git-devops (main)
$ git merge master
Already up to date.
```

git remote

Usage: git remote add [variable name] [Remote Server Link]

This command is used to connect your local repository to the remoteserver.

git push

Usage: git push [variable name] master

This command sends the committed changes of master branch to your remote repository.

Usage: git push [variable name] [branch]
This command sends the branch commits to your remote repository.

Usage: git push –all [variable name]
This command pushes all branches to your remote repository.

```
sanch@LAPTOP-OO6HP269 MINGW64 ~/git-devops (main)
$ git push --all
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 212 bytes | 212.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'master' on GitHub by visiting:
remote:      https://github.com/sanchitavarade/Devops_2024/pull/new/master
remote:
To https://github.com/sanchitavarade/Devops_2024.git
 * [new branch]      master -> master
```

Usage: git push [variable name] :[branch name]
This command deletes a branch on your remote repository.

git pull
Usage: git pull [Repository Link]
This command fetches and merges changes on the remote server to yourworking directory.

```
sanch@LAPTOP-OO6HP269 MINGW64 ~/git-devops (main)
$ git pull
remote: Enumerating objects: 12, done.
remote: Counting objects: 100% (12/12), done.
remote: Compressing objects: 100% (10/10), done.
remote: Total 12 (delta 2), reused 5 (delta 0), pack-reused 0 (from 0)
Unpacking objects: 100% (12/12), 2.34 KiB | 53.00 KiB/s, done.
From https://github.com/sanchitavarade/Devops_2024
 * [new branch]      main       -> origin/main
There is no tracking information for the current branch.
Please specify which branch you want to merge with.
See git-pull(1) for details.

    git pull <remote> <branch>

If you wish to set tracking information for this branch you can do so with:

    git branch --set-upstream-to=origin/<branch> main
```

git stash
Usage: git stash save
This command temporarily stores all the modified tracked files.

Usage: git stash pop
This command restores the most recently stashed files.

Usage: git stash list
This command lists all stashed changesets.

Usage: git stash drop
This command discards the most recently stash009564ed changeset.

Get Token -

1. Log into GitHub.
2. Click on your name / Avatar in the upper right corner and select Settings.
3. On the left, click Developer settings.
4. Select Personal access tokens and click Generate new token.
5. Give the token a description/name and select the scope of the token. ...
6. Click Generate token.
7. This configures the computer to remember the complex token by enable caching of the credentials.
   git config --global credential.helper cache
8. If needed, you can later clear the token from the local computer by running .

Conclusion -

In conclusion, installing Git and GitHub is crucial for effective version control in software development.They facilitate seamless collaboration, allowing developers to track changes and manage project versions efficiently. The installation process is straightforward, making it accessible for beginners.
Mastering Git commands enhances your workflow and organization in development projects.

References -

Git Installation: https://git-scm.com/downloads
GitHub Desktop Installation: https://desktop.github.com
Understanding Version Control: https://git-scm.com/book/en/v2/Getting-Started-About-Version-Control Git and GitHub for Beginners Tutorial: https://www.freecodecamp.org/news/git-and-github-for- beginners/

Name : Sanchita Warade
Roll no. 59
Batch : C

Experiment 2 - Jenkins Installation & Setup for CI/CD
Uninstall any version of java

$java_version=`java -version 2>&1 | head -n 1 | awk -F"\"" '{print $2}'`

-Remove all the Java related packages (Sun, Oracle, OpenJDK, IcedTea plugins, GIJ):

$ apt-cache search java | awk '{print($1)}' | grep -E -e '^(ia32-)?(sun|oracle)-
java' -e '^openjdk-' -e'^icedtea' -e '^(default|gcj)-j(re|dk)' -e '^gcj-(.*)-j(re|dk)' -
e 'java-common' | xargs sudo apt- get -y remove
$ sudo apt-get -y autoremove

-Purge config files:
$ dpkg -l | grep ^rc | awk '{print($2)}' | xargs sudo apt-get -y purge

-Remove Java config and cache directory:
$ sudo bash -c 'ls -d /home/*/.java' | xargs sudo rm -rf

-Remove manually installed JVMs:
$ sudo rm -rf /usr/lib/jvm/*

Intall java : Jenkins requires Java to run. Install the OpenJDK package by running:

sudo apt-get install openjdk-11-jdk
or
sudo apt install

openjdk-11-jdk -

yjava -version

```
root@LAPTOP-O06HP269:~# java --version
openjdk 17.0.12 2024-07-16
OpenJDK Runtime Environment (build 17.0.12+7-Ubuntu-1ubuntu224.04)
OpenJDK 64-Bit Server VM (build 17.0.12+7-Ubuntu-1ubuntu224.04, mixed mode, sharing)
root@LAPTOP-O06HP269:~# |
```

Install Jenkins

1. Before installing Jenkins, ensure your system

package list is updated: sudo apt update

```
root@LAPTOP-O06HP269:~# sudo apt update
Hit:1 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:3 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:4 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:5 https://ppa.launchpadcontent.net/ansible/ansible/ubuntu noble InRelease
Hit:6 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Ign:2 https://packages.cloud.google.com/apt kubernetes-xenial InRelease
```

sudo apt upgrade -y

```
root@LAPTOP-O06HP269:~# sudo apt update -y
Hit:1 http://archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://security.ubuntu.com/ubuntu noble-security InRelease
Hit:4 http://archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:5 http://archive.ubuntu.com/ubuntu noble-backports InRelease
Ign:3 https://packages.cloud.google.com/apt kubernetes-xenial InRelease
Hit:6 https://ppa.launchpadcontent.net/ansible/ansible/ubuntu noble InRelease
```

2. As a prerequisite add the Jenkins repository to your system with:

wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-keyadd –

3. Then, append the Jenkins repository to your system's sources list: sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'

4. After adding the repository, install Jenkins:

sudo apt update
sudo apt install jenkins -y

```
root@LAPTOP-O06HP269:~# sudo apt install jenkins -y
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Package jenkins is not available, but is referred to by another package.
This may mean that the package is missing, has been obsoleted, or
is only available from another source
```

5. To start Jenkins and enable it to run at boot, use: sudo systemctl start jenkins sudo systemctl enable Jenkins

```
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ sudo systemctl start jenkins
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ sudo systemctl enable jenkins
Synchronizing state of jenkins.service with SysV service script with /lib/systemd/systemd-sysv-install.
Executing: /lib/systemd/systemd-sysv-install enable jenkins
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$
```

6. Adjust Firewall settings : If you have a firewall enabled, allow

traffic onport 8080:sudo ufw allow 8080

```
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ sudo ufw allow 8080
Skipping adding existing rule
Skipping adding existing rule (v6)
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$
```

7. Check UFW status to confirm the change:

sudo ufw status

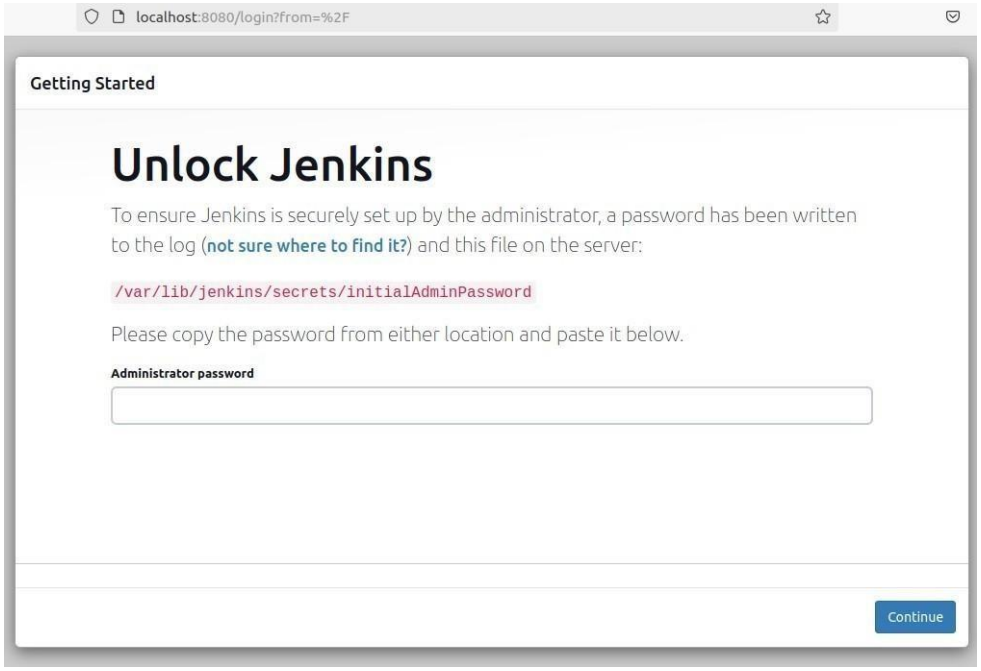
```
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ sudo ufw status
Status: active

To                         Action      From
--                         ------      ----
8080                       ALLOW       Anywhere
8080 (v6)                  ALLOW       Anywhere (v6)

onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$
```

8. Configure Jenkins

To access Jenkins, navigate to http://localhost:8080 or http://localhost:8080 in your web browser. You'll be prompted to enter the Administrator password, which can be retrievedfrom:

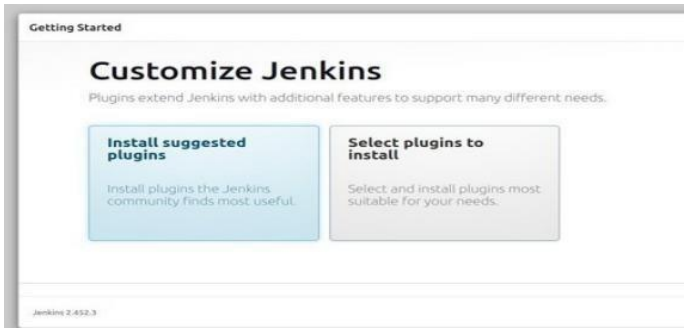


Get the password or the key to acess Jenkins using the path suggested along with sudo cat command

$sudocat/var/lib/jenkins/secrets/initialAdminPassword

9. Initial Setup Wizard -

Upon entering the Administrator password, you'll be greeted by the Initial Setup Wizard. Here, youcan install the suggested plugins or select specific ones according to your needs.

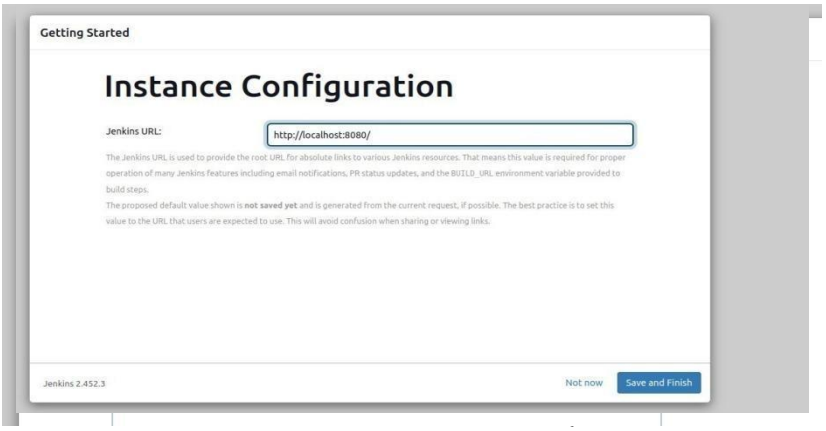


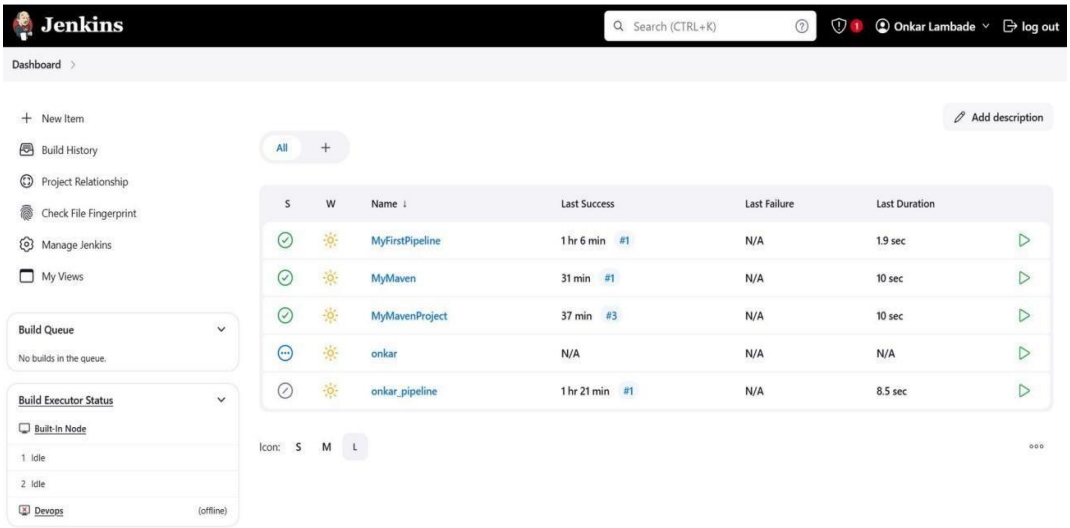
10. Create Admin User

After plugin installation, create an admin user with a username, password, and relevant details.(always give user name as dbit, and password dbit and email as dbit@one.com)

11. Instance configuration

Finally, confirm the Jenkins URL and complete the setup. You're now ready to start creatingyourCI/CD pipelines!





Conlusion :

In conclusion, with Jenkins installed on your Ubuntu 22 system, you've significantly advanced your ability to automate development processes. Jenkins provides a powerful platform for continuous integration and continuous delivery (CI/CD), streamlining your workflow and improving efficiency. By automating repetitive tasks, such as testing and deployment, you can enhance collaboration among team members and reduce the time to market for your applications. Embracing Jenkins will ultimately lead to more reliable software releases and a more agile development environment.

References :

https://reintech.io/blog/installing-configuring-jenkins-ubuntu-22

https://www.jenkins.io/doc/

https://www.digitalocean.com/community/tutorials/how-to-install-jenkins-on-ubuntu-22-04

https://plugins.jenkins.io/

https://www.jenkins.io/doc/book/pipeline/

Name: Sanchita Warade
Roll No. 59
Batch : C

Experiment 3 - Jenkins Pipeline & Maven

Aim: is to create pipeline adn maven project using jenkins

Procedure :

Steps to write here are

Part A - Snapshots of your project creation and execution with outputgenerated for pipeline

- Jenkins installed and running on your local machine or server. If not, you can follow theofficial Jenkins installation guide.

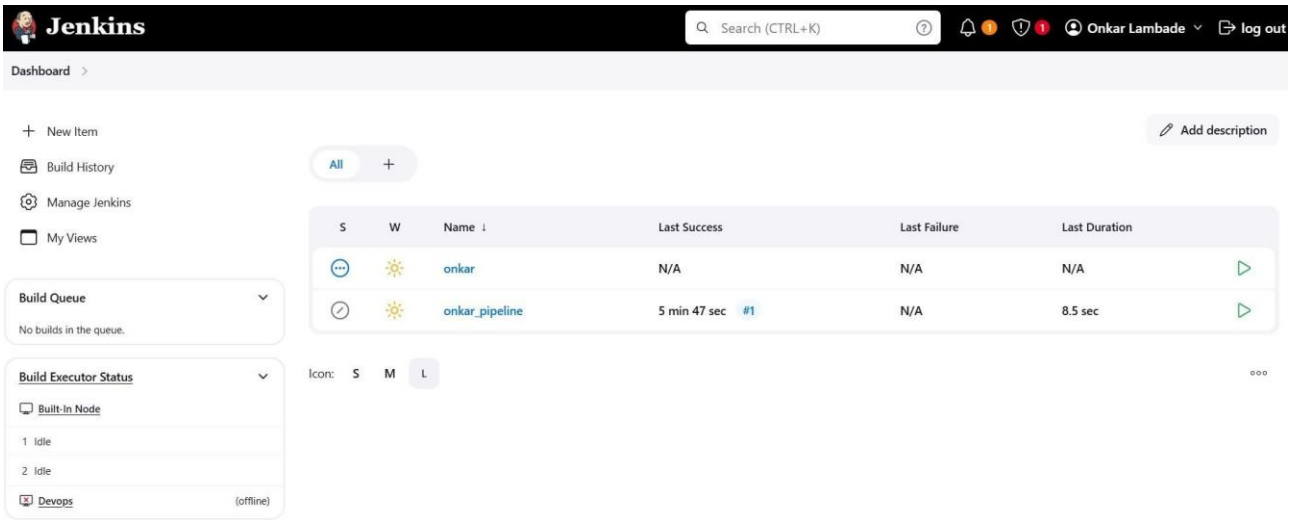
- Java installed on your machine.

```
root@LAPTOP-O06HP269:~# java --version
openjdk 17.0.12 2024-07-16
OpenJDK Runtime Environment (build 17.0.12+7-Ubuntu-1ubuntu224.04)
OpenJDK 64-Bit Server VM (build 17.0.12+7-Ubuntu-1ubuntu224.04, mixed mode, sharing)
root@LAPTOP-O06HP269:~#
```

Step-by-Step Guide for Creating and Executing a Pipeline in JenkinsStep 1:Log in to Jenkins

Open a web browser and go to your Jenkins instance URL (typically http://localhost:8080).

Log in using your Jenkins credentials.



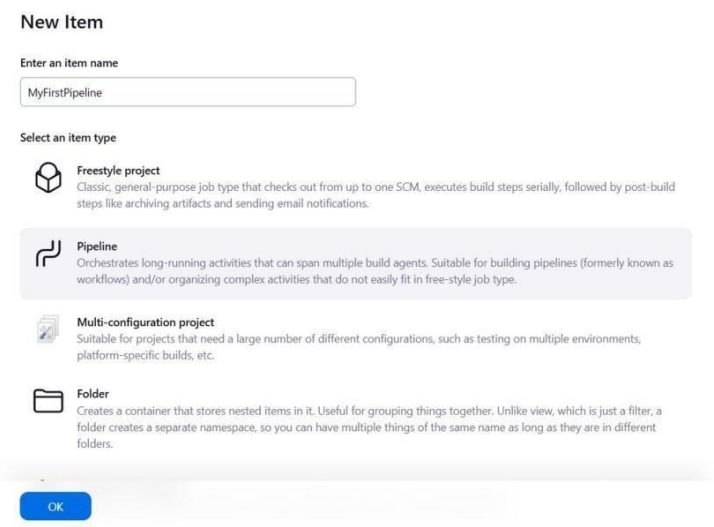
Step 2: Create a New Pipeline Project

1. From the Jenkins dashboard, click **"New Item"** on the left-hand menu.

2. Give your project a name, for example, MyFirstPipeline.

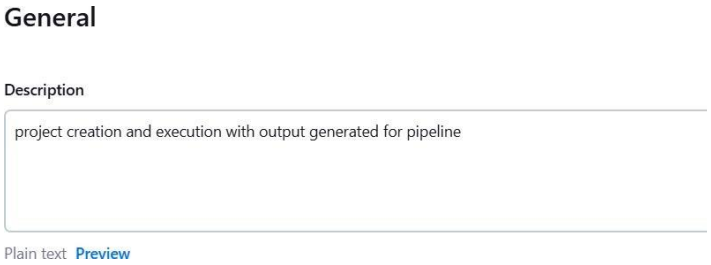
3. Select **"Pipeline"** as the project type.

4. Click **"OK"** to proceed.

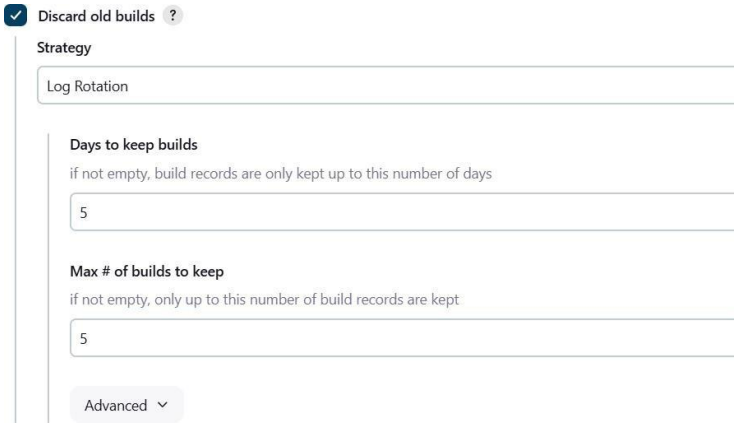


Step 3: Configure the Pipeline

1. On the project configuration page, you can fill in the following fields:
Description: Describe what the pipeline does (optional).



Discard old builds: You can check this option to limit the number of builds to keep.



Scroll down to the "Pipeline" section:

Definition: Choose **"Pipeline script"** from the dropdown menu.

Script: In this field, you will define your pipeline

```
pipeline {
agent any
stages {
stage('Checkout')
{steps {
echo 'Checking out code from SCM...'
}
}
stage('Build')
{steps {
echo 'Building the project...'
}
}
stage('Test')
{steps {
echo 'Running tests...'
}
}
stage('Package') {
steps {
echo 'Packaging the application...'
}
}
stage('Deploy')
{steps {
echo 'Deploying the application...'
}
}
}
}
```

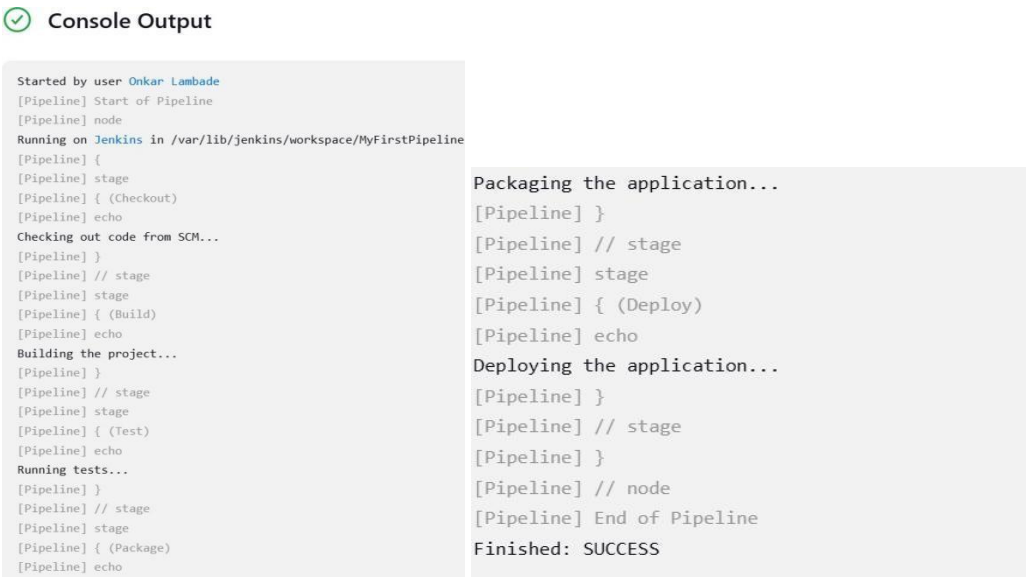
Step 5: Save and Build the Pipeline

1. **Click on "Save"** at the bottom of the configuration page.
2. On the project's main page, you should see a **"Build Now"** option on the left sidebar. Clickonit to trigger the pipeline.

Step 6: Check the Output

1. Once the build is triggered, you can click on the build number under the **"Build History"** section to see the build details.

Click **"Console Output"** to see the real-time logs of your pipeline execution. You shouldsee theoutput messages:



If you see this output, congratulations! Your pipeline is working.

Part B -Snapshots of your project creation and execution with outputgenerated for marven

1. Jenkins installed and running on your local machine or server. If not, you can follow theofficial Jenkins installation guide.

2. Java installed on your machine.

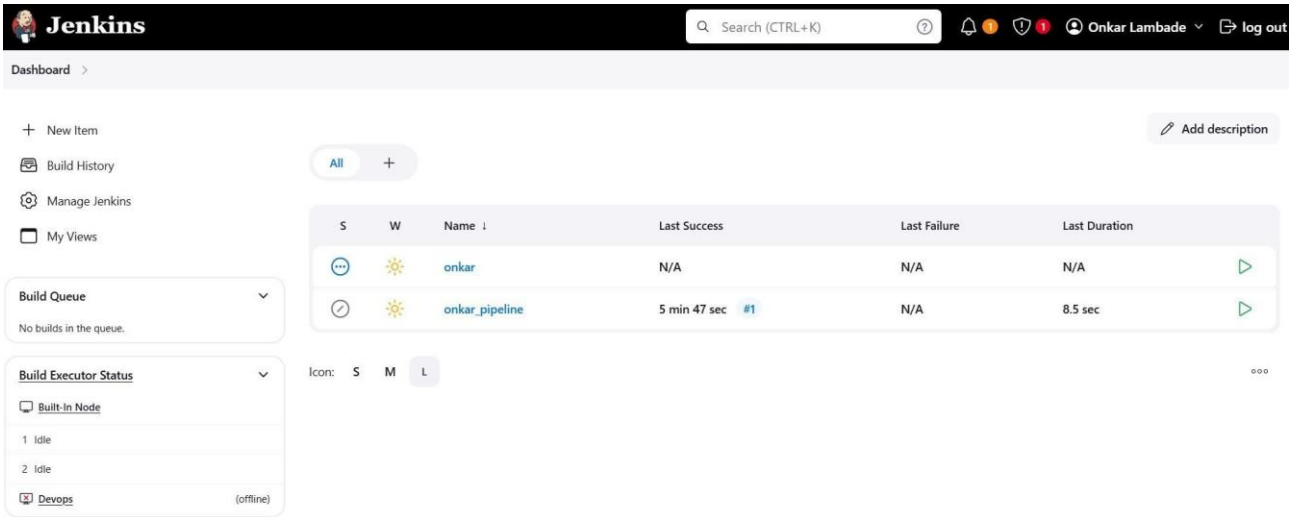
```
root@LAPTOP-O06HP269:~# java --version
openjdk 17.0.12 2024-07-16
OpenJDK Runtime Environment (build 17.0.12+7-Ubuntu-1ubuntu224.04)
OpenJDK 64-Bit Server VM (build 17.0.12+7-Ubuntu-1ubuntu224.04, mixed mode, sharing)
root@LAPTOP-O06HP269:~#
```

Step-by-Step Guide for Creating and Executing a

Pipeline in Jenkins:-Step 1:Log in to Jenkins

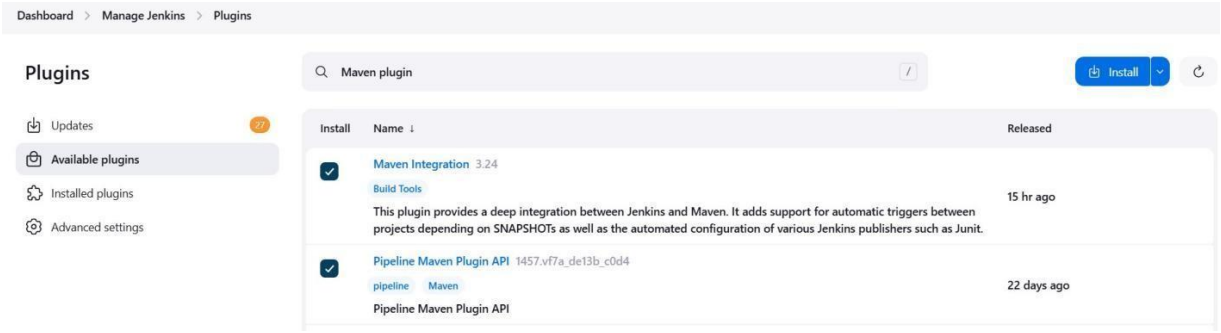
- Open a web browser and go to your Jenkins instance URL (typically http://localhost:8080).

Log in using your Jenkins credentials.



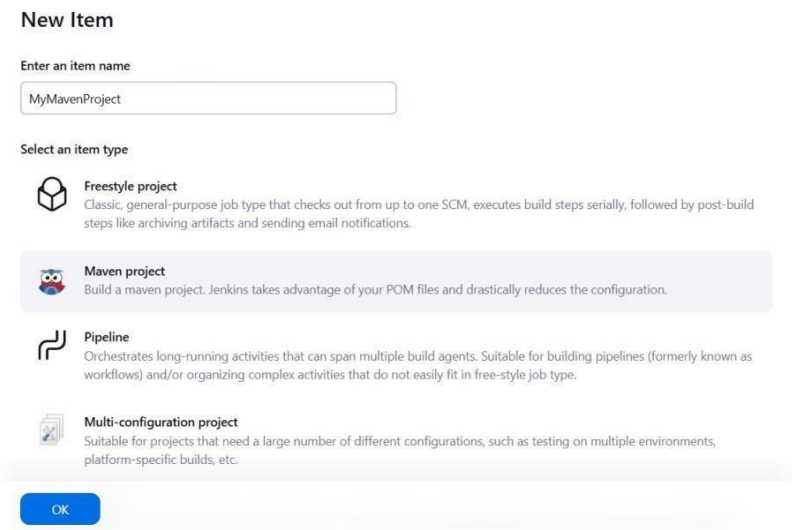
Step 2: Install the Maven Plugin (if not already installed)

1. Go to **"Manage Jenkins"** from the dashboard.
2. Click on **"Manage Plugins"**.
3. In the **"Available"** tab, search for **"Maven Integration"** or **"Maven Plugin"**.
4. Install the plugin if it's not already installed. Restart Jenkins if prompted.



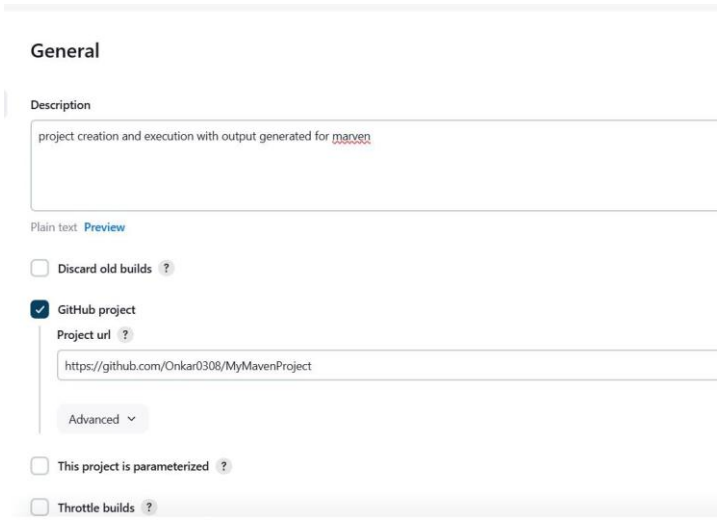
Step 3: Create a New Maven Project

1. From the Jenkins dashboard, click on “New Item”.
2. Enter a name for your project, e.g., MyMavenProject.
3. Select “Maven Project” and click “OK”.



Step 4: Configure the Maven Project

Description: Optionally provide a description of your project.



GitHub project: If your Maven project is hosted on GitHub, you can provide the URLhere.IF you don't Have Github repository follow this steps:-

Step 1:Set up maven

- Ensure that you have Maven installed on your machine. You can check byrunning mvn -v in your command line or terminal.

Create a New Maven Project:

- Open your command line or terminal.
- Navigate to the directory where you want to create your project:
- Use the following Maven command to create a new project:

```
sanchita@LAPTOP-O06HP269:~$ mkdir mavenprojects
sanchita@LAPTOP-O06HP269:~$ cd mavenprojects
sanchita@LAPTOP-O06HP269:~/mavenprojects$ mvn archetype:generate -DgroupId=com.example -DartifactId=MyMavenProject -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

mvn archetype:generate -DgroupId=com.example -DartifactId=MyMavenProject - DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false

- Replace com.example with your desired group ID.
- Replace MyMavenProject with your desired artifact ID.

```
sanchita@LAPTOP-O06HP269:~/mavenproject$ mvn archetype:generate -DgroupId=com.example -DartifactId=MyMavenProject - D
archetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
```

Navigate to Your Project Directory:bash

```
[INFO] ----------------------------------------------------------------------------
[INFO] Using following parameters for creating project from Old (1.x) Archetype: maven-archetype-quickstart:1.0
[INFO] ----------------------------------------------------------------------------
[INFO] Parameter: basedir, Value: /home/sanchita/mavenproject
[INFO] Parameter: package, Value: com.example
[INFO] Parameter: groupId, Value: com.example
[INFO] Parameter: artifactId, Value: MyMavenProject
[INFO] Parameter: packageName, Value: com.example
[INFO] Parameter: version, Value: 1.0-SNAPSHOT
[INFO] project created from Old (1.x) Archetype in dir: /home/sanchita/mavenproject/MyMavenProject
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  35.158 s
[INFO] Finished at: 2024-10-27T20:20:40+05:30
[INFO] ------------------------------------------------------------------------
```

Build Your Project (optional):

- You can build your Maven project to ensure it's set up correctly:bash

mvn clean install

```
onkar@DESKTOP-D1SJIU7:~/mavenprojects$ ls
FirstMavenProject
onkar@DESKTOP-D1SJIU7:~/mavenprojects$ cd FirstMavenProject/
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ mvn clean install
[INFO] Scanning for projects...
[INFO]
[INFO] ---------------------< com.onkar:FirstMavenProject >----------------------
[INFO] Building FirstMavenProject 1.0-SNAPSHOT
[INFO] --------------------------------[ jar ]---------------------------------
```

Step 2: Create a GitHub Repository

1. Log in to GitHub:

- Open a web browser and go to GitHub.
- Log in to your account (or create one if you don't have an account).

2. Create a New Repository:

- Click on the "+" icon in the top right corner and select "New repository".
- Fill in the details:
- Repository name: Enter a name for your repository, e.g.,MyMavenProject.
- Description: Add an optional description.
- Public/Private: Choose whether you want your repository to be public orprivate.
- Do not initialize with a README: Since you will be pushing an existing project.
- Click "Create repository".

Step 3: Initialize Git in Your Local Project

1. Initialize Git:

- In your command line or terminal, still within

```
sanchita@LAPTOP-006HP269:~/mavenproject/firstmavenproject$ git init
hint: Using 'master' as the name for the initial branch. This default branch name
hint: is subject to change. To configure the initial branch name to use in all
hint: of your new repositories, which will suppress this warning, call:
hint:
hint:   git config --global init.defaultBranch <name>
hint:
hint: Names commonly chosen instead of 'master' are 'main', 'trunk' and
hint: 'development'. The just-created branch can be renamed via this command:
hint:
hint:   git branch -m <name>
Initialized empty Git repository in /home/sanchita/mavenproject/firstmavenproject/.git/
```

the project directory(MyMavenProject), run:

2. Add Remote Repository:

- Add the GitHub repository as a remote:

```
sanchita@LAPTOP-006HP269:~/mavenproject/firstmavenproject$ git remote add origin https://github.com/sanchitavarade/Ma
venproject.git
```

- Replace yourusername with your GitHub username and adjust the URLaccording to therepository name.

3. Stage Your Files:

- Stage all files in your project for commit:

```
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ git add .
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ git commit -m "Initial commit"
[master (root-commit) d31d562] Initial commit
 5 files changed, 70 insertions(+)
 create mode 100644 pom.xml
 create mode 100644 src/main/java/com/onkar/App.java
 create mode 100644 src/test/java/com/onkar/AppTest.java
 create mode 100644 target/maven-status/maven-compiler-plugin/compile/default-compile/createdFiles.lst
 create mode 100644 target/maven-status/maven-compiler-plugin/compile/default-compile/inputFiles.lst
```

- Commit Your Changes:
- Commit the staged files:


```
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ git add .
onkar@DESKTOP-D1SJIU7:~/mavenprojects/FirstMavenProject$ git commit -m "Initial commit"
[master (root-commit) d31d562] Initial commit
 5 files changed, 70 insertions(+)
 create mode 100644 pom.xml
 create mode 100644 src/main/java/com/onkar/App.java
 create mode 100644 src/test/java/com/onkar/AppTest.java
 create mode 100644 target/maven-status/maven-compiler-plugin/compile/default-compile/createdFiles.lst
 create mode 100644 target/maven-status/maven-compiler-plugin/compile/default-compile/inputFiles.lst
```


Step 4: Push Your Project to GitHub

1. Push to GitHub:

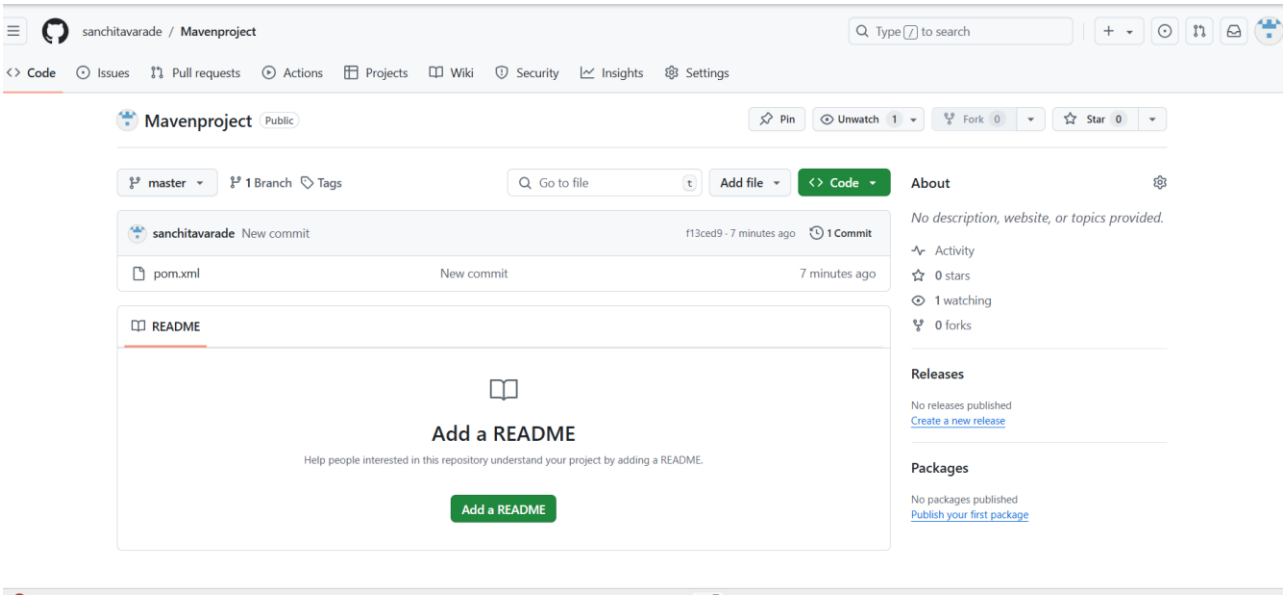
- Push your local commits to the GitHub repository:

```
sanchita@LAPTOP-O06HP269:~/mavenproject/firstmavenproject$ git push origin master
Username for 'https://github.com': sanchitavarade
Password for 'https://sanchitavarade@github.com':
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Delta compression using up to 16 threads
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 815 bytes | 815.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To https://github.com/sanchitavarade/Mavenproject.git
 * [new branch]      master -> master
```

Step 5: Verify Your Project on GitHub

1. Go back to GitHub:

- Refresh your repository page. You should see your Maven project files uploadedto GitHub.

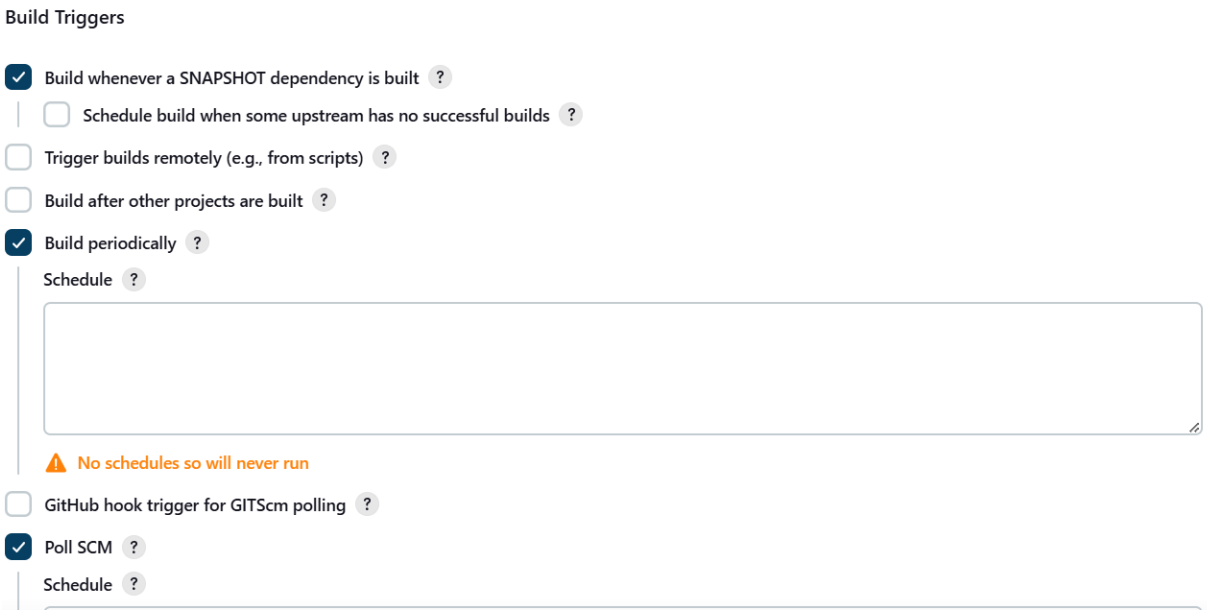


2. Source Code Management:

- Choose "Git" if your project is in a Git repository.
- Enter the repository URL and credentials if needed.

4. Build Triggers:

- You can check options like "Poll SCM" or "Build periodically" depending onyour needs.

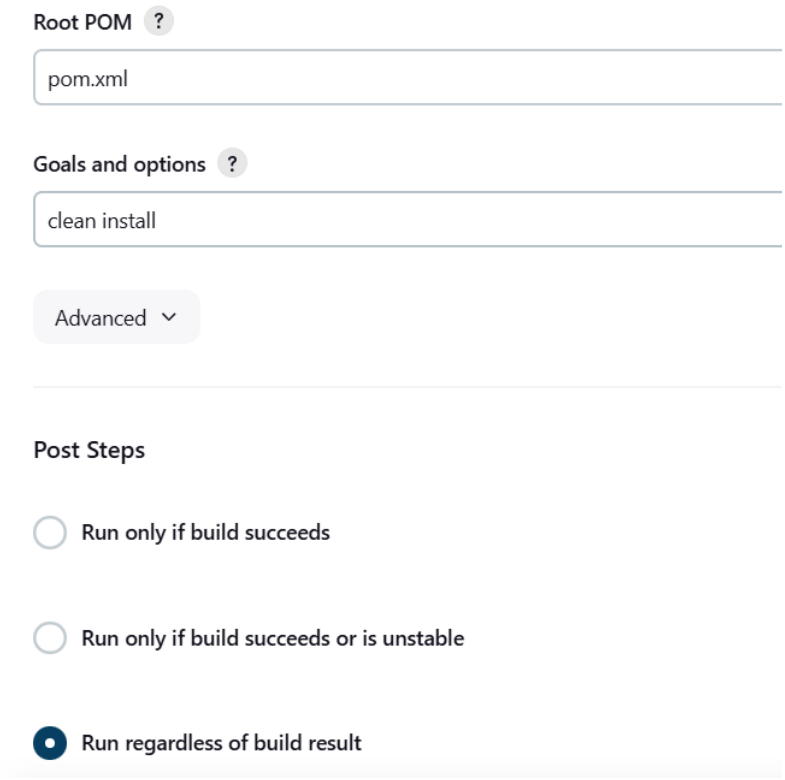


5. Build:

- In the “Goals and options” section, enter the Maven goals, for example, cleaninstall to cleanand build the project.

- Optionally specify other parameters, such as -DskipTests to skip tests.

- Advanced Settings: Click on “Advanced” if you need to set up specific Maven settingsorprofiles.

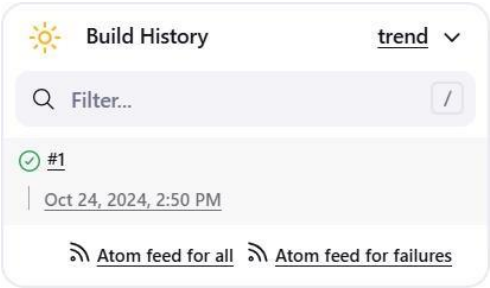


Step 5: Save the Configuration

- Click “Save” at the bottom of the page to store your project settings.

Step 6: Build the Maven Project

- On the project’s main page, you will see a “Build Now” option on the left sidebar. Clickit totrigger the Maven build.
- You will see a build history entry with a timestamp.



Step 7: Check the Output

- Click on the build number (usually labeled as #1 for the first build) under the “BuildHistory”section to view build details.

- Click on “Console Output” to see the logs of your Maven build process.Sample Output: You should see output logs

similar to the following:

Console Output

Download | Copy | View as plain text

```
Started by user Onkar Lambade
Running as SYSTEM
Building on the built-in node in workspace /var/lib/jenkins/workspace/MyMavenProject
Unpacking https://repo.maven.apache.org/maven2/org/apache/maven/apache-maven/3.9.9/apache-maven-3.9.9-bin.zip to
/var/lib/jenkins/tools/hudson.tasks.Maven_MavenInstallation/maven on Jenkins
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/Onkar0308/MyMavenProject
 > git init /var/lib/jenkins/workspace/MyMavenProject # timeout=10
Fetching upstream changes from https://github.com/Onkar0308/MyMavenProject
 > git --version # timeout=10
 > git --version # 'git version 2.34.1'
 > git fetch --tags --force --progress -- https://github.com/Onkar0308/MyMavenProject +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/Onkar0308/MyMavenProject # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
 > git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision d31d56247394d7ceb95e65ac9d1363c8eae2b25d (refs/remotes/origin/master)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f d31d56247394d7ceb95e65ac9d1363c8eae2b25d # timeout=10
Commit message: "Initial commit"
First time build. Skipping changelog.
Parsing POMs
Discovered a new module com.onkar:FirstMavenProject FirstMavenProject
Modules changed, recalculating dependency graph
```

```
commons-1.14.jar 37207
<===[JENKINS REMOTING CAPACITY]===>channel started
Executing Maven:  -B -f /var/lib/jenkins/workspace/MyMavenProject/pom.xml clean install
[INFO] Scanning for projects...
[INFO]
[INFO] --------------------< com.onkar:FirstMavenProject >---------------------
[INFO] Building FirstMavenProject 1.0-SNAPSHOT
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-clean-plugin/3.2.0/maven-clean-plugin-3.2.0.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-clean-plugin/3.2.0/maven-clean-plugin-3.2.0.pom
(5.3 kB at 17 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/35/maven-plugins-35.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/35/maven-plugins-35.pom (9.9 kB at 367
kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/35/maven-parent-35.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-parent/35/maven-parent-35.pom (45 kB at 1.0 MB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/apache/25/apache-25.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/apache/25/apache-25.pom (21 kB at 642 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-clean-plugin/3.2.0/maven-clean-plugin-3.2.0.jar
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-clean-plugin/3.2.0/maven-clean-plugin-3.2.0.jar
(36 kB at 776 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/3.3.1/maven-resources-plugin-
3.3.1.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-resources-plugin/3.3.1/maven-resources-plugin-
3.3.1.pom (8.2 kB at 272 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/39/maven-plugins-39.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/plugins/maven-plugins/39/maven-plugins-39.pom (8.1 kB at 476
```

```
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/resolver/maven-resolver-api/1.9.18/maven-resolver-api-
1.9.18.pom
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/resolver/maven-resolver-api/1.9.18/maven-resolver-api-
1.9.18.pom (2.7 kB at 167 kB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/resolver/maven-resolver-util/1.9.18/maven-resolver-util-
1.9.18.jar
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/resolver/maven-resolver-util/1.9.18/maven-resolver-util-
1.9.18.jar (196 kB at 5.2 MB/s)
[INFO] Downloading from central: https://repo.maven.apache.org/maven2/org/apache/maven/resolver/maven-resolver-api/1.9.18/maven-resolver-api-
1.9.18.jar
[INFO] Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/resolver/maven-resolver-api/1.9.18/maven-resolver-api-
1.9.18.jar (157 kB at 4.2 MB/s)
[INFO] Installing /var/lib/jenkins/workspace/MyMavenProject/pom.xml to /var/lib/jenkins/.m2/repository/com/onkar/FirstMavenProject/1.0-
SNAPSHOT/FirstMavenProject-1.0-SNAPSHOT.pom
[INFO] Installing /var/lib/jenkins/workspace/MyMavenProject/target/FirstMavenProject-1.0-SNAPSHOT.jar to
/var/lib/jenkins/.m2/repository/com/onkar/FirstMavenProject/1.0-SNAPSHOT/FirstMavenProject-1.0-SNAPSHOT.jar
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  12.695 s
[INFO] Finished at: 2024-10-24T14:43:22+05:30
[INFO] ------------------------------------------------------------------------
Waiting for Jenkins to finish collecting data
[JENKINS] Archiving /var/lib/jenkins/workspace/MyMavenProject/pom.xml to com.onkar/FirstMavenProject/1.0-SNAPSHOT/FirstMavenProject-1.0-SNAPSHOT.pom
[JENKINS] Archiving /var/lib/jenkins/workspace/MyMavenProject/target/FirstMavenProject-1.0-SNAPSHOT.jar to com.onkar/FirstMavenProject/1.0-
SNAPSHOT/FirstMavenProject-1.0-SNAPSHOT.jar
channel stopped
Finished: SUCCESS
```

Conclusion : Frame your conclusion here

References: Include your references here

Rubrics: 5 marks for 1st part and 5 mark for second part - 100- 80 % - 5, 80- 60%- 4 and so on.

Name: Sanchita Warade
Roll no. 59
Batch : C

Experiment 4. Docker Installation & Basic Commands for docker

Part A:

<u>Steps for Installing Docker:</u>

1. Open the terminal on Ubuntu.

2. Remove any Docker files that are running in the system, using thefollowingcommand:

$ sudo apt-get remove docker docker-engine docker.io

After entering the above command, you will need to enter the password of the rootandpress enter.

3. Check if the system is up-to-date using the following command:
$ sudo apt-get update

```
sanchita@LAPTOP-O06HP269:~$ sudo apt-get remove docker docker-engine docker.io
[sudo] password for sanchita:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
E: Unable to locate package docker-engine
sanchita@LAPTOP-O06HP269:~$ sudo apt-get update
Get:1 https://apt.releases.hashicorp.com jammy InRelease [12.9 kB]
Get:2 http://security.ubuntu.com/ubuntu jammy-security InRelease [129 kB]
Hit:3 http://archive.ubuntu.com/ubuntu jammy InRelease
Get:4 http://archive.ubuntu.com/ubuntu jammy-updates InRelease [128 kB]
Get:5 https://apt.releases.hashicorp.com jammy/main amd64 Packages [154 kB]
Get:6 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [1896 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy-backports InRelease [127 kB]
Get:8 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 Packages [2113 kB]
Get:9 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [305 kB]
Get:10 http://archive.ubuntu.com/ubuntu jammy-updates/main Translation-en [363 kB]
Get:11 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [13.3 kB]
Get:12 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [2517 kB]
Get:13 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [17.9 kB]
Get:14 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [435 kB]
Get:15 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [911 kB]
Get:16 http://security.ubuntu.com/ubuntu jammy-security/universe Translation-en [180 kB]
Get:17 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 c-n-f Metadata [19.5 kB]
Ign:13 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata
Get:18 http://archive.ubuntu.com/ubuntu jammy-updates/restricted amd64 Packages [2577 kB]
Get:19 http://archive.ubuntu.com/ubuntu jammy-updates/restricted Translation-en [445 kB]
Get:20 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 Packages [1133 kB]
Get:21 http://archive.ubuntu.com/ubuntu jammy-updates/universe Translation-en [265 kB]
Get:22 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 c-n-f Metadata [26.4 kB]
Get:13 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 c-n-f Metadata [17.9 kB]
Fetched 13.8 MB in 60s (230 kB/s)
Reading package lists... Done
```

4. Install Docker using the following command:
$ sudo apt install docker.io

You'll then get a prompt asking you to choose between y/n - choose y

5. Install all the dependency packages using the following command:
$ sudo snap install docker

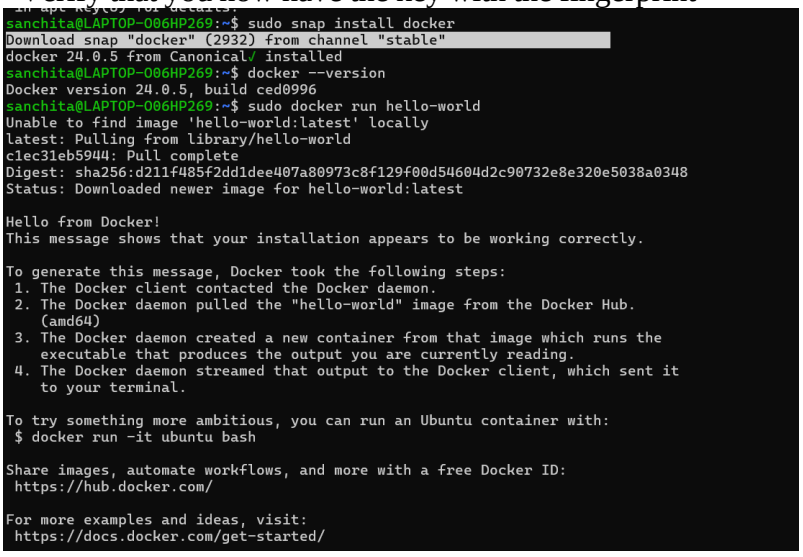
alternate commands to install docker are

$ sudo apt-get install \apt-transport- https \
ca-certificates \
curl \
software-properties-common

To nstall packages to allow apt to use a repository over HTTPS

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key addcommadto add Docker's official GPG key

$ sudo apt-key fingerprint 0EBFCD88
Verify that you now have the key with the fingerprint



```
sanchita@LAPTOP-O06HP269:~$ sudo apt install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools btrfs-progs cgroupfs-mount | cgroup-lite debootstrap docker-doc rinse zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz runc ubuntu-fan
0 upgraded, 8 newly installed, 0 to remove and 8 not upgraded.
Need to get 75.5 MB of archives.
After this operation, 284 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://archive.ubuntu.com/ubuntu jammy/universe amd64 pigz amd64 2.6-1 [63.6 kB]
Get:2 http://archive.ubuntu.com/ubuntu jammy/main amd64 bridge-utils amd64 1.7-1ubuntu3 [34.4 kB]
Get:3 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 runc amd64 1.1.12-0ubuntu2~22.04.1 [8405 kB]
Get:4 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 containerd amd64 1.7.12-0ubuntu2~22.04.1 [37.8 MB]
Get:5 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 dns-root-data all 2023112702~ubuntu0.22.04.1 [5136 B]
Get:6 http://archive.ubuntu.com/ubuntu jammy-updates/main amd64 dnsmasq-base amd64 2.90-0ubuntu0.22.04.1 [374 kB]
Get:7 http://archive.ubuntu.com/ubuntu jammy-updates/universe amd64 docker.io amd64 24.0.7-0ubuntu2~22.04.1 [28.8 MB]
Get:8 http://archive.ubuntu.com/ubuntu jammy/universe amd64 ubuntu-fan all 0.12.16 [35.2 kB]
Fetched 75.5 MB in 55s (1363 kB/s)
Preconfiguring packages ...
Selecting previously unselected package pigz.
(Reading database ... 44762 files and directories currently installed.)
Preparing to unpack .../0-pigz_2.6-1_amd64.deb ...
Unpacking pigz (2.6-1) ...
Selecting previously unselected package bridge-utils.
Preparing to unpack .../1-bridge-utils_1.7-1ubuntu3_amd64.deb ...
Unpacking bridge-utils (1.7-1ubuntu3) ...
Selecting previously unselected package runc.
Preparing to unpack .../2-runc_1.1.12-0ubuntu2~22.04.1_amd64.deb ...
Unpacking runc (1.1.12-0ubuntu2~22.04.1) ...
Selecting previously unselected package containerd.
Preparing to unpack .../3-containerd_1.7.12-0ubuntu2~22.04.1_amd64.deb ...
Unpacking containerd (1.7.12-0ubuntu2~22.04.1) ...
Selecting previously unselected package dns-root-data.
Preparing to unpack .../4-dns-root-data_2023112702~ubuntu0.22.04.1_all.deb ...
Unpacking dns-root-data (2023112702~ubuntu0.22.04.1) ...
Selecting previously unselected package dnsmasq-base.
Preparing to unpack .../5-dnsmasq-base_2.90-0ubuntu0.22.04.1_amd64.deb ...
Unpacking dnsmasq-base (2.90-0ubuntu0.22.04.1) ...
Selecting previously unselected package docker.io.
Preparing to unpack .../6-docker.io_24.0.7-0ubuntu2~22.04.1_amd64.deb ...
Unpacking docker.io (24.0.7-0ubuntu2~22.04.1) ...
```

```
Setting up dnsmasq-base (2.90-0ubuntu0.22.04.1) ...
Setting up runc (1.1.12-0ubuntu2~22.04.1) ...
Setting up dns-root-data (2023112702~ubuntu0.22.04.1) ...
Setting up bridge-utils (1.7-1ubuntu3) ...
Setting up pigz (2.6-1) ...
Setting up containerd (1.7.12-0ubuntu2~22.04.1) ...
Created symlink /etc/systemd/system/multi-user.target.wants/containerd.service → /lib/systemd/system/containerd.service.
Setting up ubuntu-fan (0.12.16) ...
Created symlink /etc/systemd/system/multi-user.target.wants/ubuntu-fan.service → /lib/systemd/system/ubuntu-fan.service.
Setting up docker.io (24.0.7-0ubuntu2~22.04.1) ...
Adding group `docker' (GID 118) ...
Done.
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd/system/docker.service.
Created symlink /etc/systemd/system/sockets.target.wants/docker.socket → /lib/systemd/system/docker.socket.
Processing triggers for dbus (1.12.20-2ubuntu4.1) ...
```

6. Before testing Docker, check the version installed using the following command:
$ docker –version

```
sanchita@LAPTOP-O06HP269:~$ docker --version
Docker version 24.0.5, build ced0996
```

7. Pull an image from the Docker hub using the following command:
$ sudo docker run hello-worldHere, hello-world is the docker image present on the Docker hub. Output will be like this asinfigure.

```
sanchita@LAPTOP-O06HP269:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
c1ec31eb5944: Pull complete
Digest: sha256:d211f485f2dd1dee407a80973c8f129f00d54604d2c90732e8e320e5038a0348
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

8. The actual Hellow World command of docker is
$ docker run docker/whalesay cowsay boo
The default image of docker appears with the message boo.

```
sanchita@LAPTOP-O06HP269:~$ sudo docker run docker/whalesay cowsay boo
Unable to find image 'docker/whalesay:latest' locally
latest: Pulling from docker/whalesay
[DEPRECATION NOTICE] Docker Image Format v1, and Docker Image manifest version 2, schema 1 support will be removed in an upcoming release. Suggest the author of docker.io/docker/whalesay:l
atest to upgrade the image to the OCI Format, or Docker Image manifest v2, schema 2. More information at https://docs.docker.com/go/deprecated-image-specs/
e190868d63f8: Pull complete
909cd34c6fd7: Pull complete
0b9bfabab7c1: Pull complete
a3ed95caeb02: Pull complete
00bf65475aba: Pull complete
c57b6bcc83e3: Pull complete
8978f6879e2f: Pull complete
8eed3712d2cf: Pull complete
Digest: sha256:178598e51a26abbc958b8a2e48825c90bc22e641de3d31e18aaf55f3258ba93b
Status: Downloaded newer image for docker/whalesay:latest
 _____
< boo >
 -----
    \
     \
      \
                    ##        .
              ## ## ##       ==
           ## ## ## ##      ===
       /"""""""""""""""""___/ ===
  ~~~ {~~ ~~~~ ~~~ ~~~~ ~~ ~ /  ===- ~~~
       \______ o          __/
        \    \        __/
          \____\______/
```

following command:
$ sudo docker images

To display all the containers pulled, use the following command:
$ sudo docker ps -a

To check for containers in a running state, use the following command:

```
sanchita@LAPTOP-O06HP269:~$ sudo docker images
REPOSITORY        TAG       IMAGE ID       CREATED         SIZE
hello-world       latest    d2c94e258dcb   18 months ago   13.3kB
docker/whalesay   latest    6b362a9f73eb   9 years ago     247MB
sanchita@LAPTOP-O06HP269:~$ sudo docker ps -a
CONTAINER ID   IMAGE             COMMAND        CREATED              STATUS                          PORTS     NAMES
d1d4c5a3059f   docker/whalesay   "cowsay boo"   About a minute ago   Exited (0) About a minute ago             trusting_cohen
6865b7cc6acd   hello-world       "/hello"       3 minutes ago        Exited (0) 3 minutes ago                  zen_napier
sanchita@LAPTOP-O06HP269:~$ sudo docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
sanchita@LAPTOP-O06HP269:~$
```

$ sudo docker ps

Part B: Docker search, docker Pull and docker run

Use the command docker search to search for public images on the Docker hub. It willreturn information about the image name, description, stars, official and automated.

Now that we know the name of the image, we can pull that from the Docker hubusing thecommand docker pull. Here, we are setting the platform option as well.

$ sudo docker search mysql

or alternate

$ sudo docker pull -platform linux/x86_64 mysql

or alternate

$sudo docker pull mysql/mysql-erver:tag

```
sanchita@LAPTOP-O06HP269:~$ sudo docker run --name mysql -e MYSQL_ROOT_PASSWORD=rootpassword -d mysql
Unable to find image 'mysql:latest' locally
latest: Pulling from library/mysql
eba3c26198b7: Pull complete
97f7c8c33abe: Pull complete
aa23d877fa04: Pull complete
a143609ddd2d: Pull complete
78308a3437c4: Pull complete
c0880e4b3737: Pull complete
4bab267f9ce1: Pull complete
e575f6d9b17a: Pull complete
607f86c00053: Pull complete
cd68caa5febe: Pull complete
Digest: sha256:fd8d1b4e287c49e1e35eb5a103f337111947662130eb8a3e6c3e823813f47f7d
Status: Downloaded newer image for mysql:latest
e4b6c5f9563cf493a5d93549eafd93cc17d4d4bc22cc7f4b259d184d2e2344c1
```

Log into MySQL within the docker container using the docker execcommand:
$ sudo docker exec -it mysql bash

Now run this command

$ sudo docker run --name mysql -p 3406:3306 -e MYSQL_ROOT_PASSWORD=anypassword -d mysql/mysql-server:5.7

You can check it by running the following command…The first image as you can see in the snippetis themysql-server image in anew terminal

$ sudo docker ps -a

Remember, when we created and ran the MySQL container, we provided MYSQL_ROOT_PASSWORD=anypassword. Create a database and user, and grant privileges in MySQL (from

within the container). og into MySQL within the docker container using the dockerexec command, Log intoMySQL if you haven't already. After login, the mysql> prompt shows up:

$ mysql -uroot -panypassword

$ SHOW DATABASES ;

```
sanchita@LAPTOP-O06HP269:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 9
Server version: 8.0.39-0ubuntu0.22.04.1 (Ubuntu)

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> show databases
    -> ;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.04 sec)
```

$ use database;

$ exit

$ exit

$docker restart

Let's restart our stopped contained by using the following command. We may wantto use this after wereboot our machine.

docker restart f8c52bedeecc

$docker rename
Now, let's change the container name from compassionate_fermi to test_db. We maywant to change thename to keep track of our containers more easily.
docker rename compassionate_fermi test_db

$docker exec
Access the running container test_db by running the following command. It's helpful, if we want to access the MySQL command line and execute MySQL queries.

```
docker exec -it
test_db
bashmysql -uroot
- pmy-secret-pw
SHOW
DATABASES;
```

The -i and -t options are used to access the container in an interactive mode. Then we provide the name of the container we want to access, which in this casetest_db. Finally, the bash command is used to get abash shell inside the container.

$docker logs
This command is helpful to debug our Docker containers. It will fetch logs froma specified container.

```
$docker logs test_db
```

If we want to continue to stream new output,use the option -follow.docker logs -follow
test_db
$docker rm

To remove a container, we can use the following command.docker rmtest_db

You may encounter an error like
Error response from daemon: You cannot remove a running container
……Stop thecontainer before attempting removal or force remove
As it recommends, we can stop the container first and then remove it or useoption -f to remove arunning container forcefully.

```
$docker stop test_db
$docker rm test_db# ordocker rm -f test_db
```

$docker rmi

To free some disk space, we can use the docker rmi command with the image id toremove an image.

```
$docker rmi eb0e825dc3cf
```

These commands come with plenty of helpful options. If you want to know aboutother available options, run the docker command_name --help

command. For example:

$docker logs-help

Conclusion -

Installing Docker on your system empowers you to create, deploy, and manage applications in isolated containers,enhancing development efficiency and consistency. Docker simplifies the development process by allowing you topackage applications with their dependencies, ensuring they run seamlessly across different environments. By mastering basic Docker commands, you can streamline your workflows, facilitate collaboration among teams, and improve resource utilization. Embracing Docker will ultimately lead to faster deployment cycles and a more agile approach to application development.

References -

https://docs.docker.com/
https://docs.docker.com/get-docker/
https://docs.docker.com/docker-for-windows/install/
https://docs.docker.com/engine/install/ubuntu/
https://docs.docker.com/engine/reference/commandline/cli/

Name: Sanchita Warade
Roll No.59
Batch : C

Experiment 5: Miniproject(Ansible)

Aim: Installation and demonstration of Ansible

To follow this tutorial, you will need:

- One Ansible Control Node: The Ansible control node is the machine we'll use to connect to and control the Ansible hosts over SSH. Your Ansible control node can either be your local machine or a server dedicated to running Ansible, though this guide assumes your control node is an Ubuntu 20.04 system. Make sure the control node hasa non-root user with sudo privileges.

- To set this up, you can follow Steps 2 and 3of our Initial Server Setup Guide for Ubuntu 20.04. However, please note that if you're using a remote server as your Ansible Control node, you should follow everystep of this guide. Doing so will configure a firewall on the server with ufw and enable external access toyour non-root user profile, both of which will help keepthe remote server secure.

- An SSH keypair associated with this user. To set this up, you can follow Step 1 ofour guide on How to SetUp SSH Keys on Ubuntu 20.04. One or more Ansible Hosts: An Ansible host is any machine that your Ansible control node is configured to automate. This guide assumes your Ansible hosts are remote Ubuntu 20.04servers. Make sure each Ansible host has:

- The Ansible control node's SSH public key added to the authorized_keys of a system user. This user can be either root or a regular user with sudo privileges. To set this up, you can follow Step 2 of How to Set Up SSHKeys on Ubuntu 20.04.

Ansible can be installed on various Linux distributions. We'll cover the installation process forUbuntu/Debian and CentOS/RHEL systems.

Ubuntu/Debian

To install Ansible on Ubuntu or Debian-based systems, follow these steps:

1. Update your system's package index:

sudo apt update

2. Install software-properties-common (if not already installed): sudo apt install software-properties-common

3. Add the Ansible PPA (Personal Package Archive):

sudo apt-add-repository --yes --update ppa:ansible/ansible

4. Install Ansible:

sudo apt install ansible

5. Verify the installation:

ansible --version

CentOS/RHEL

For CentOS or Red Hat Enterprise Linux (RHEL) systems, follow these steps:

1. Enable the EPEL (Extra Packages for Enterprise Linux) repository:sudo yum install epel-release

2. Install Ansible:

sudo yum install ansible

3. Verify the installation:

ansible --version

2. Installing Ansible on AWS EC2

Installing Ansible on an AWS EC2 instance is similar to installing it on a regular Linux system. However, thereare a few additional considerations:

1. Launch an EC2 instance:

1. Choose an Amazon Machine Image (AMI) based on your preferred Linux distribution.
2. Select an instance type that meets your requirements.
3. Configure security groups to allow SSH access (port 22).
4. Launch the instance and connect to it using SSH.

2. Once connected to your EC2 instance, follow the installation steps for your chosen Linux distribution asoutlined in the previous section.
3. After installation, it's recommended to create an IAM role with the necessary permissions for Ansible tointeract with other AWS services. Attach this role to your EC2 instance.
4. Configuring Ansible

After installing Ansible, you'll need to configure it for your environment. Here are some essentialconfiguration steps:

1. Create an inventory file:
The inventory file lists the hosts that Ansible will manage. Create a file named `hosts` in the `/etc/ansible/`directory:

sudo nano /etc/ansible/hosts

Add your hosts

to this file. For

example:

```
[webservers]
web1 ansible_host=192.168.1.10 web2 ansible_host=192.168.1.11
```

```
[databases]
db1 ansible_host=192.168.1.20
```

2. Configure SSH key-based authentication:
For seamless communication between your Ansible control node and managed nodes, set up SSH key-based authentication:

a. Generate an SSH key pair (if you haven't already):
```
ssh-keygen -t rsa -b 4096
```

b. Copy the public key to your managed nodes:
```
ssh-copy-id user@host
```

3. Test the connection:
Verify that Ansible can communicate with your managed nodes:

```
ansible all -m ping
```

If everything is set up correctly, you should see a success message for each host.

4. Basic Ansible Usage

Now that Ansible is installed and configured, let's explore some basic usage:

1. Ad-hoc commands:
Run simple tasks on your managed nodes using ad-hoc commands:

```
ansible webservers -a "uptime"
```

This command will display the uptime of all hosts in the `webservers` group.

2. Playbooks:
Playbooks are Ansible's configuration, deployment, and orchestration language. They allow you to describea set of steps in YAML format. Here's a simple example:

Create a file named `update_systems.yml`:

```
- update_cache: yes name: Update all systems hosts: all
become: yes tasks:

- name:  Update  apt
cache
(Debian/Ubuntu)
apt:


when: ansible_os_family == "Debian"
- name:    Upgrade    all
packages
(Debian/Ubuntu)apt:
upgrade: dist
when: ansible_os_family == "Debian"
- name:       Update
yum              cache
(CentOS/RHEL)
yum:
update_cache: yes
when: ansible_os_family == "RedHat"
- name: Upgrade all
packages
(CentOS/RHEL)
yum: name: '*' state:
latest
-
when: ansible_os_family == "RedHat"
```

Run the playbook:

```
ansible-playbook update_systems.yml
```

This playbook will update and upgrade all packages on your managed systems, regardless of whetherthey're Debian/Ubuntu or CentOS/RHEL based.

3. Roles:
Roles are ways of automatically loading certain vars_files, tasks, and handlers based

on a known file structure. They are great for organizing playbooks and making them reusable. Here's a basic structure of arole:

roles/ common/ tasks/ main.yml handlers/ main.yml files/ templates/ vars/ main.yml defaults/ main.yml meta/ main.yml

You can create roles using the `ansible-galaxy` command:

ansible-galaxy init rolename

5. Troubleshooting Common Issues

Even with careful setup, you may encounter issues. Here are some common problems and their solutions:

1. SSH Connection Issues:

1. Ensure that SSH key-based authentication is set up correctly.
2. Check that the SSH service is running on the managed nodes.
3. Verify that the correct SSH port is being used (especially if it's not the default port 22).

2. Privilege Escalation Errors:

1. Make sure the user has sudo privileges on the managed nodes.
2. Use the `become: yes` directive in your playbooks when necessary.

3. Python Interpreter Issues:

1. Ansible requires Python on the managed nodes. Ensure it's installed and specify the correct interpreter ifneeded: ansible_python_interpreter: /usr/bin/python3

4. Inventory Problems:

1. Double-check your inventory file for typos or incorrect IP addresses.
2. Use the `ansible-inventory --list` command to verify your inventory.

5. Module Not Found Errors:

1. Ensure you're using the correct module name and that it's available in your Ansible version.
2. Update Ansible if you're trying to use a module from a newer version.

Documentation:

Creating File on Ubuntu PCs using Ansible

Index

1. Install Ansible on Your Main PC
2. Set Up SSH Key-Based Authentication
3. Create an Ansible Inventory File
4. Create an Ansible Playbook
5. Run the Ansible Playbook

Step 1. Install Ansible on Your Main PC

a. Open a terminal on your main PC.
b. Update the package lists by running:
 sudo apt update
c. Install Ansible using the command:
 sudo apt install ansible
d. Verify the Ansible installation:
 ansible --version

Step 2. Set Up SSH Key-Based Authentication

a. Install SSH if it's not already installed:
 sudo apt install openssh-server
b. Generate an SSH key pair using:
 ssh-keygen -t rsa
c. Copy the public key to the remote Ubuntu PCs:
 ssh-copy-id username@hostname
 (Replace `username` and `hostname` with the actual user and hostnames of the remote PC)

Step 3. Create an Ansible Inventory File

a. Create an inventory file (hosts.ini) to define the target PCs:
 nano hosts.ini
b. Add the target hosts in the following format:
 [servers]

server1 ansible_host=10.0.1.164 ansible_ssh_user=dbit
c. Save the file and exit the text editor.

Step 4. Create an Ansible Playbook
a. Create a playbook file (create_file.yml) to specify the tasks for creating a file:
nano create_file.yml
b. Add the following YAML code to the playbook:

```
---
- name: Create a file on remote PCs
 hosts: servers
 tasks:
 - name: Ensure a file is created
 file:
 path: /home/dbit/testfile.txt
 state: touch
```

c. Save the playbook and exit.

Step 5. Run the Ansible Playbook
a. Open a terminal on your main PC.
b. Navigate to the directory containing your playbook and inventory file:
cd /path/to/your/playbook
c. Run the playbook using the following command:
ansible-playbook -i hosts.ini create_file.yml

Conclusion -

This guide has walked you through the process of installing Ansible on both Linux systems and AWS EC2 instances. We've covered the basic configuration, usage, and troubleshooting tips to help you get started with Ansible. Remember, Ansible is a powerful tool with many features and capabilities beyond what we've discussed here. As you become more comfortable with these basics, explore Ansible's extensive documentation to learn about more advanced topics like dynamic inventories, custom modules, and complex playbooks.

By mastering Ansible, you'll be able to automate repetitive tasks, manage configurations across multipleservers, and streamline your IT operations

References:

https://www.youtube.com/watch?v=VANub3AhZpI&t=1s
https://www.youtube.com/watch?v=Egnsb89T0Rs
https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-ansible-on- ubuntu-20-04
https://www.coachdevops.com/2020/04/install-ansible-on-ubuntu-how-to-setup.html