DevOps Practical Sets 1, 2, 3

SET :


```
Q1. Create a new Git repository locally, add a Java file (HelloWorld.java), commit it, and push it t
sudo apt update
sudo apt install git -y
mkdir DevOpsProject
cd DevOpsProject
git init
nano HelloWorld. java
# public class HelloWorld {
     public static void main(String[] args) {
         System.out.println("Hello World from Ubuntu!");
# }
git add HelloWorld.java
git commit -m "Initial commit - HelloWorld.java"
git branch -M main
git remote add origin https://github.com/<username>/<repo-name>.git
git push -u origin main
```


02. Create a simple Mayen project using the command line and build it using myn clean package.

```
sudo apt update
sudo apt install openjdk-17-jdk maven -y
java -version
mvn -v
mvn archetype:generate -DgroupId=com.example -DartifactId=myapp -DarchetypeArtifactId=maven-archetyp
cd myapp
mvn clean package
ls target/
```


Q3. Create a Dockerfile for your Maven project and build a Docker image using the command line.

```
sudo apt update
sudo apt install docker.io -y
sudo systemctl start docker
sudo systemctl enable docker
cd ~/myapp
nano Dockerfile
# FROM openjdk:17
# COPY target/myapp-1.0-SNAPSHOT.jar app.jar
# ENTRYPOINT ["java","-jar","app.jar"]
sudo docker build -t myapp:latest .
sudo docker images
```


SET 2

Q1. Create a new Jenkins Freestyle project that pulls code from your GitHub repository and builds it

```
sudo apt update
sudo apt install openjdk-17-jdk git maven -y
wget -q -0 - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -
sudo sh -c 'echo deb http://pkg.jenkins.io/debian/ binary/ > /etc/apt/sources.list.d/jenkins.list'
sudo apt update
sudo apt install jenkins -y
sudo systemctl start jenkins
sudo systemctl enable jenkins
# Open Jenkins at http://localhost:8080 → Create Freestyle Project → Configure Git URL → Add build
```

Q2. Configure Jenkins to automatically trigger a build whenever new code is pushed to GitHub using a

```
sudo apt update
sudo apt install git openjdk-17-jdk maven -y
wget -q -0 - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -
sudo sh -c 'echo deb http://pkg.jenkins.io/debian/ binary/ > /etc/apt/sources.list.d/jenkins.list'
sudo apt update
sudo apt install jenkins -y
sudo systemctl start jenkins
sudo systemctl enable jenkins
git clone https://github.com/<username>/<repo-name>.git
cd <repo-name>
echo "// Test webhook" >> HelloWorld.java
git add HelloWorld.java
git commit -m "Testing GitHub webhook"
git push
 \texttt{\# In GitHub} \rightarrow \texttt{Settings} \rightarrow \texttt{Webhooks} \rightarrow \texttt{Add webhook} \rightarrow \texttt{URL: http://<jenkins-server>:8080/github-webhook} 
\# In Jenkins job 	o Build Triggers 	o GitHub hook trigger for GITScm polling
Q3. Run a container from a Docker image and verify it using docker ps and container logs.
sudo apt update
sudo apt install docker.io -y
sudo systemctl start docker
sudo systemctl enable docker
mkdir DockerTest && cd DockerTest
nano HelloWorld.java
# public class HelloWorld {
     public static void main(String[] args) {
#
         System.out.println("Hello Docker!");
#
mkdir -p src/main/java
mv HelloWorld.java src/main/java/
sudo apt install maven -y
mvn clean package
nano Dockerfile
# FROM openjdk:17
# COPY target/myapp-1.0-SNAPSHOT.jar app.jar
# ENTRYPOINT ["java","-jar","app.jar"]
sudo docker build -t myapp:latest .
sudo docker run -d --name mycontainer myapp:latest
sudo docker ps
sudo docker logs mycontainer
Q1. Clone your GitHub repository, modify one Java file, commit and push the changes.
sudo apt update
sudo apt install git openjdk-17-jdk -y
git clone https://github.com/<username>/<repo-name>.git
cd <repo-name>
nano HelloWorld.java
git add HelloWorld.java
git commit -m "Updated HelloWorld.java"
git push
Q2. Create a Jenkins pipeline with stages: Checkout \rightarrow Build \rightarrow Test.
sudo apt update
sudo apt install git openjdk-17-jdk maven -y
wget -q -0 - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -
sudo sh -c 'echo deb http://pkg.jenkins.io/debian/ binary/ > /etc/apt/sources.list.d/jenkins.list'
sudo apt update
sudo apt install jenkins -y
sudo systemctl start jenkins
sudo systemctl enable jenkins
```

Jenkinsfile content:

agent any

stages {

pipeline {

#

```
# stage('Checkout') { steps { git 'https://github.com/<username>/<repo-name>.git' } }
# stage('Build') { steps { sh 'mvn clean package' } }
# stage('Test') { steps { sh 'mvn test' } }
# }
# }
```


Q3. Build a Docker image for your Java project and verify its creation.

```
sudo apt update
sudo apt install docker.io maven -y
sudo systemctl start docker
sudo systemctl enable docker
mkdir DockerJavaProject && cd DockerJavaProject
nano HelloWorld.java
# public class HelloWorld {
     public static void main(String[] args) {
#
          System.out.println("Hello Docker World!");
#
      }
# }
mkdir -p src/main/java
mv HelloWorld.java src/main/java/
nano pom.xml
# Minimal Maven configuration
mvn clean package
nano Dockerfile
# FROM openjdk:17
# COPY target/myapp-1.0-SNAPSHOT.jar app.jar
# ENTRYPOINT ["java","-jar","app.jar"]
sudo docker build -t myapp:latest .
sudo docker images
```