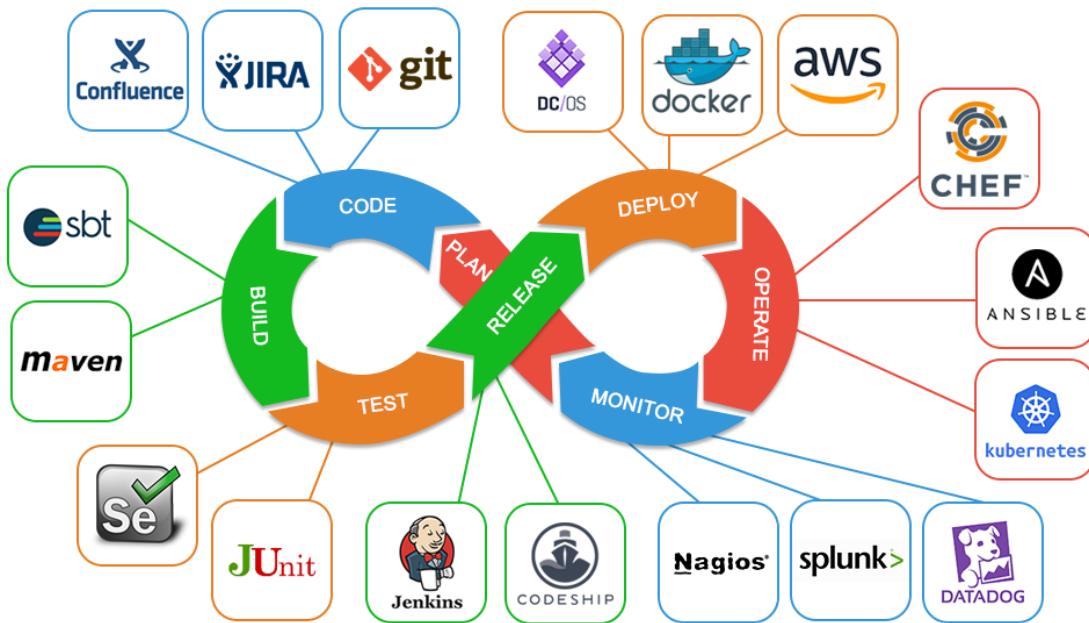


Scientific Calculator using Devops Methodologies

HIMANK JAIN

MT2021054

Software Production Engineering (CS 816)



Github Profile: https://github.com/himankjn/MiniProject_SPE

DockerHub Profile: <https://hub.docker.com/repository/docker/himankjn/check>

Table Of Contents:

[Problem Statement:](#)

[DevOps:](#)

[DevOps tool chain:](#)

[Jenkins Prerequisites:](#)

[Plugins:](#)

[Credentials:](#)

[Tools Configuration:](#)

[Jenkins Configuration:](#)

[Docker Hub:](#)

[Steps:](#)

[Testing out the code on local machine:](#)

[Push to Github:](#)

[Jenkins Project:](#)

[Stages in Pipeline:](#)

[Application Screenshots:](#)

[Summary:](#)

[Dependencies & Code:](#)

[Log4j2.xml:](#)

[Unit Testing:](#)

[CHALLENGES AND SOLUTIONS:](#)

Problem Statement:

Creating a calculator program with operations such as

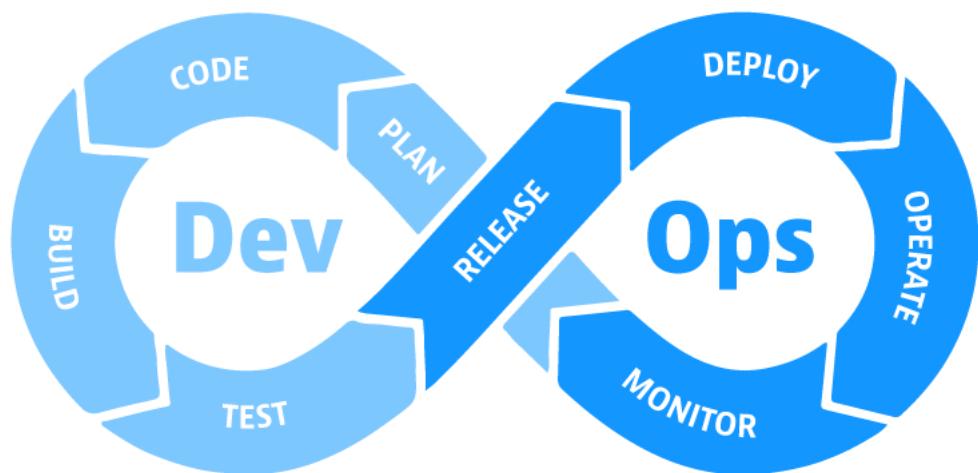
Square root function - \sqrt{x}

Factorial function - $x!$

Natural logarithm (base e) - $\ln(x)$

Power function - x^y

DevOps:



DevOps is a set of practices that combines software development and IT operations. It aims to shorten the systems development life cycle and provide continuous delivery with high software quality. DevOps is complementary with Agile software development; several DevOps aspects came from the Agile methodology.

The cultural philosophies, practices, and tools in DevOps increase an organisation's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organisations using traditional software development and infrastructure management processes. It enables faster development of new products and easier maintenance of existing deployments. This speed enables organisations to better serve their customers and compete more effectively in the market.

DevOps tool chain:

The project pipeline includes:

1. **Github**: Source control management tool
2. **Junit**: Testing
3. **Maven**: Build Automation & Dependency Management
4. **Jenkins, Github CLI**: Continuous Integration
5. **IntelliJ**: Integrated Development Environment
6. **Docker**: Containerization
7. **Docker Hub**: Docker Images
8. **Ansible**: Configuration Management and Remote Deployment
9. **VirtualBox**: Deployment on VirtualMachine
10. **ELK Stack**: Logs & Monitoring
11. **Java 11**: Language
12. **ngrok**: Exposing local machine port to the internet

We use two machines:

1. **Host** [Ubuntu20]
2. **Virtual Machine Remote Server** [Ubuntu18]

Setup:

- Download and Install the following on your host machine.

- **Java 11**: `sudo apt install openjdk-11-jdk`

```
java --version
```

```
himank@himank:~$ java --version
openjdk 11.0.14.1 2022-02-08
OpenJDK Runtime Environment (build 11.0.14.1+1-Ubuntu-0ubuntu1.20.04)
OpenJDK 64-Bit Server VM (build 11.0.14.1+1-Ubuntu-0ubuntu1.20.04, mixed mode, sharing)
himank@himank:~$
```

- **Git**: `sudo apt install git`

```
git --version
```

```
himank@himank:~$ git --version  
git version 2.25.1  
himank@himank:~$ █
```

- **Maven:** `sudo apt install maven`

```
mvn --version
```

```
himank@himank:~$ mvn --version  
Apache Maven 3.6.3  
Maven home: /usr/share/maven  
Java version: 11.0.14.1, vendor: Ubuntu, runtime: /usr/lib/jvm/java-11-openjdk-amd64  
Default locale: en_IN, platform encoding: UTF-8  
OS name: "linux", version: "5.13.0-39-generic", arch: "amd64", family: "unix"  
himank@himank:~$ █
```

- **Jenkins:** `sudo apt install jenkins`

```
jenkins --version
```

```
himank@himank:~$ jenkins --version  
2.332.2  
himank@himank:~$ █
```

- **Ansible:** `sudo apt install ansible`

```
ansible --version
```

```
himank@himank:~$ ansible --version  
ansible 2.9.6  
  config file = /etc/ansible/ansible.cfg  
  configured module search path = ['/home/himank/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']  
  ansible python module location = /usr/lib/python3/dist-packages/ansible  
  executable location = /usr/bin/ansible  
  python version = 3.8.10 (default, Mar 15 2022, 12:22:08) [GCC 9.4.0]  
himank@himank:~$ █
```

- **IntelliJ:** `sudo snap install intelliij-idea-ultimate --classic`

- **Docker:** `sudo apt install docker.io`, `sudo snap install docker`

```
docker --version
```

```
himank@himank:~$ docker --version  
Docker version 20.10.7, build 20.10.7-0ubuntu5~20.04.2  
himank@himank:~$ █
```

- **ngrok:** `sudo snap install ngrok`

```
ngrok --version
```

```
himank@himank:~$ ngrok --version
ngrok version 2.3.40
himank@himank:~$
```

- o **openssh:** `sudo apt-get install openssh-server`

```
ssh -V
```

```
himank@himank:~$ ssh -V
OpenSSH_8.2p1 Ubuntu-4ubuntu0.4, OpenSSL 1.1.1f 31 Mar 2020
himank@himank:~$
```

- Download and install **Docker** and **openssh** on your remote server(virtual machine):

- o **pip:** `sudo apt install python-pip`
- o **docker:** `pip install docker`
- o **openssh:** `sudo apt-get install openssh-server`

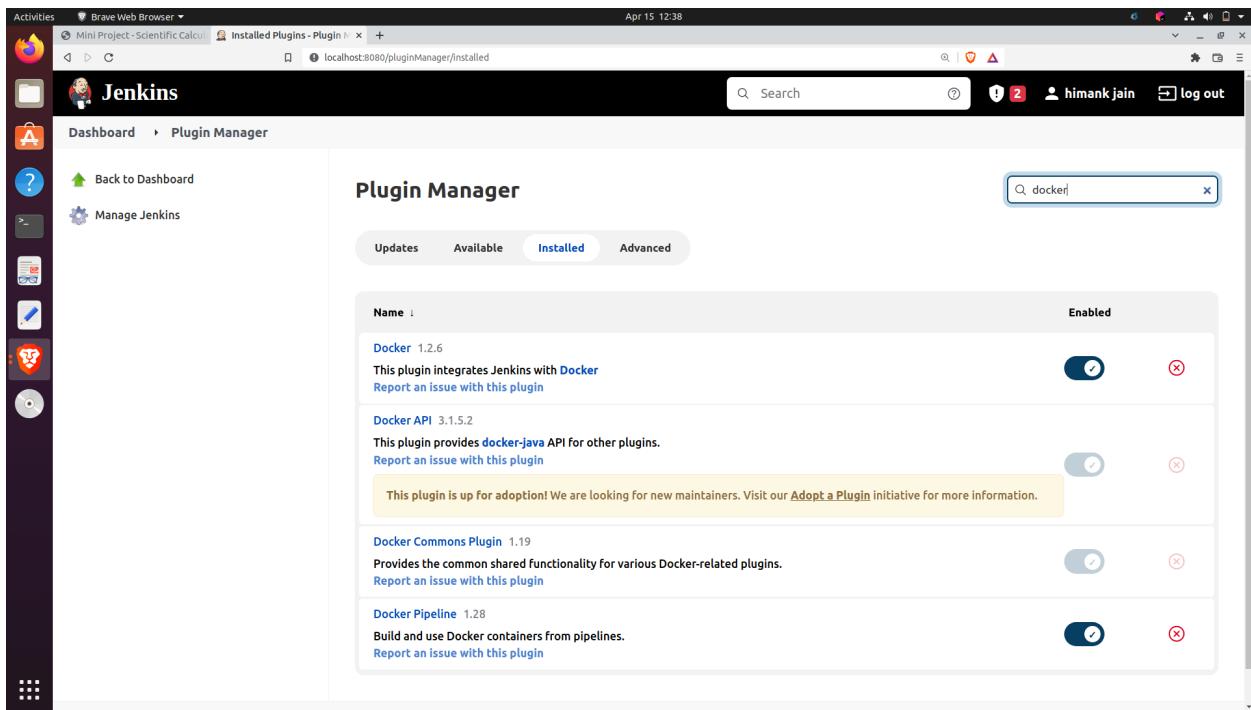
Jenkins Prerequisites:

Plugins:

Navigate to **Dashboard** → **Manage Jenkins** → **Manage Plugins**

Download & install the following plugins:

- **Docker**
- **Docker pipeline**
- **Ansible plugin**
- **Github Integration Plugin**
- **Maven Integration**
- **Pipeline**



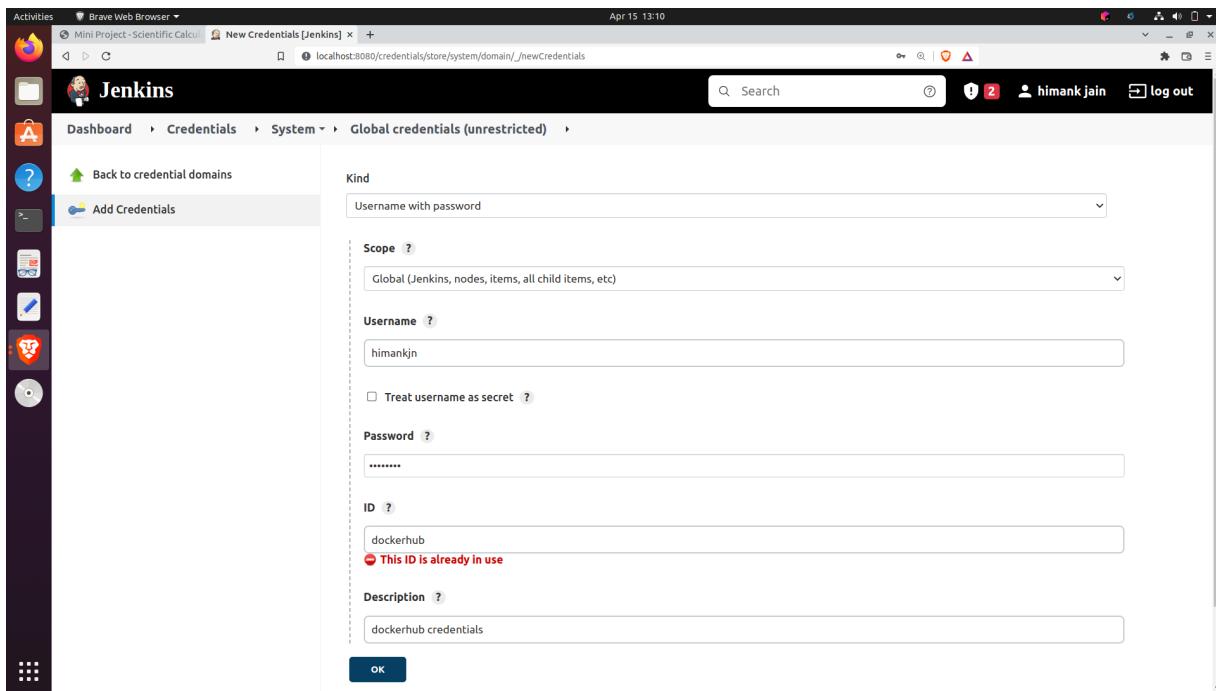
Credentials:

Add Dockerhub credentials:

Navigate to Manage Jenkins → Manage Credentials → Add Credentials

Docker hub:

1. **Kind:** Username and Password
2. **Username:** DockerHub username
3. **Password:** DockerHub Password
4. **ID:** Identification ["dockerhub"]
5. **Description**



Tools Configuration:

- Navigate to **Manage Jenkins → Global Tool Configuration → Git**:

Name: git

Path to Git Executable: /usr/bin/git

Git

Git installations

Git	Name
<input type="text" value="git"/>	
Path to Git executable ?	
<input type="text" value="/usr/bin/git"/>	
<input type="checkbox"/> Install automatically ?	
Delete Git	
Add Git ▾	

- Navigate to **Manage Jenkins → Global Tool Configuration → Maven**

Name: Maven3

Add Installer: Install from Apache

Maven

Maven installations

[Add Maven](#)

	Maven	Name
	<input type="text" value="Maven3"/>	

Install automatically ?

Install from Apache

Version

[Delete Installer](#)

[Add Installer ▾](#)

[Delete Maven](#)

- Navigate to **Manage Jenkins → Global Tool Configuration → Ansible**

Name: Ansible

Path to ansible executable directory: /usr/bin/

Ansible

Ansible installations

[Add Ansible](#)

	Ansible	Name
	<input type="text" value="Ansible"/>	

Path to ansible executables directory

Install automatically ?

[Delete Ansible](#)

[Add Ansible](#)

List of Ansible installations on this system

Jenkins Configuration:

- Navigate to **Manage Jenkins → Configure System → Jenkins URL**
 - Add your jenkins address with port
 - <http://192.168.100.51:8080/>
- Navigate to **Manage Jenkins → Configure System → System Admin e-mail address:**
 - Add your admin email address
 - himank369123@gmail.com

Jenkins Location

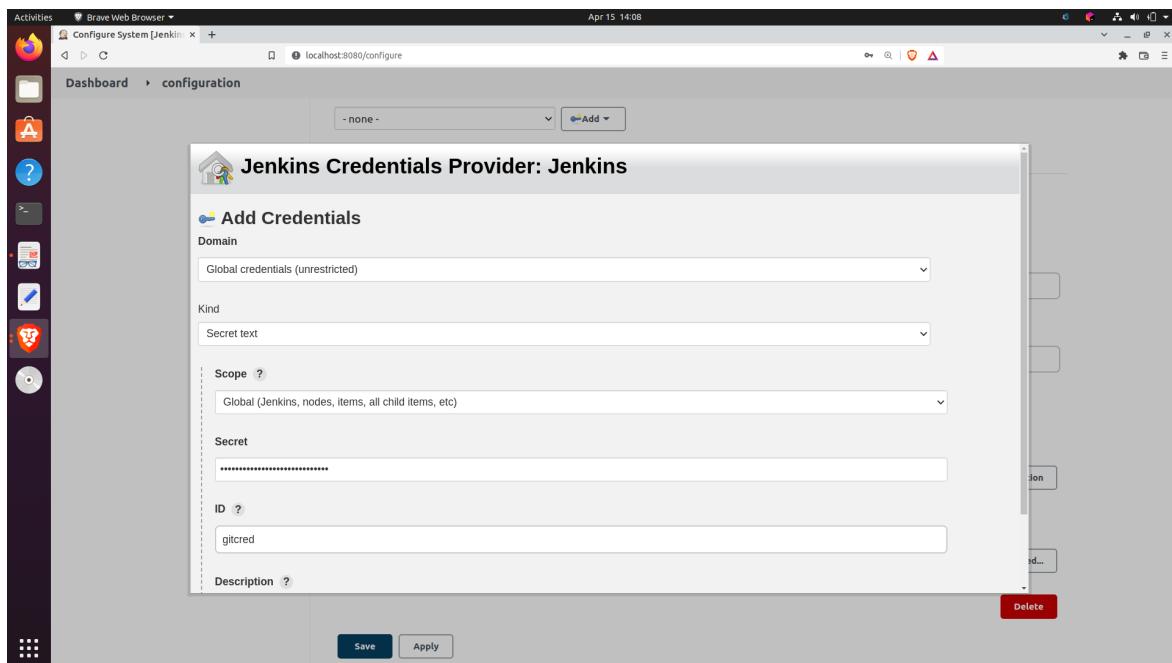
Jenkins URL ?

http://192.168.100.51:8080/

System Admin e-mail address ?

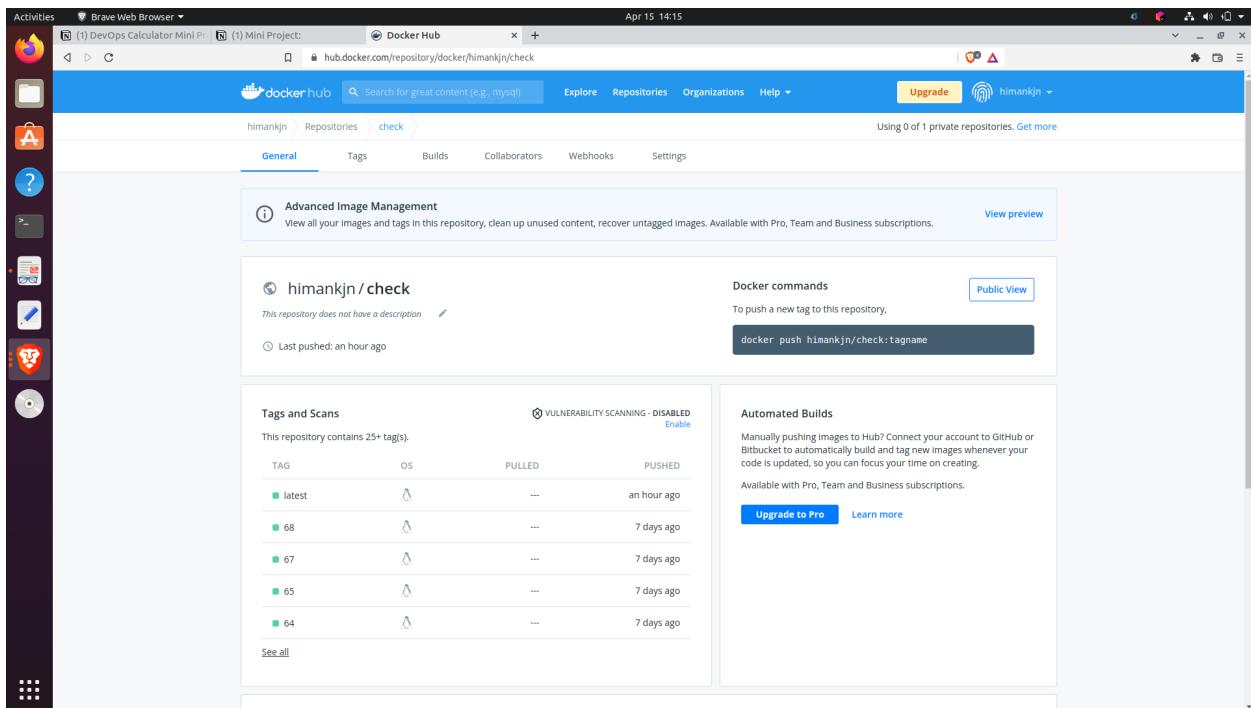
himank369123@gmail.com

- Navigate to Manage Jenkins → Configure System → Github → Github Server:
 - **Name:** servername [gitserver]
 - **API URL:** <https://api.github.com>
 - **Credentials:** Add Credential
 - **Kind:** SecretText
 - **Secret:** Personal Access Token
 - **ID:** Identification ["gitcred"]
 - **Description**



Docker Hub:

- Create a new public repository with format : username/dockerimage
- We will use this repository for pushing out docker image and pulling it in remote server.



Steps:

SSH Key:

- Generate ssh public key on local host and copy it to remote server to ensure login without password. Then test login using ssh.
- NOTE:** Ensure remote server is up and running.

```
ssh-keygen -t rsa
ssh-copy-id himank@10.0.2.15
#10.0.2.15 is remote server[ubuntu18's] ip address
ssh himank@10.0.2.15
```

```
Activities Terminal April 15 14:31
himank@himank:~$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/home/himank/.ssh/id_rsa):
/home/himank/.ssh/id_rsa already exists.
Overwrite (y/n)? y
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /home/himank/.ssh/id_rsa
Your public key has been saved in /home/himank/.ssh/id_rsa.pub
The key fingerprint is:
SHA256:IxvFUGqNL5M+nTcMF4OL3Ag8q6SnkyLf0FIw9ASD5BM himank@himank
The key's randomart image is:
+---[RSA 3072]---+
|oE. +o o+ |
|o.*o ooo o |
|.oo=+++.+ |
|o..oo+=oB |
|o+ o = S |
|o o . = o |
|= . |
|[B ..o |
|o+... |
+---[SHA256]---+
himank@himank:~$ ssh-copy-id himank@172.16.134.134
/usr/bin/ssh-copy-id: INFO: attempting to log in with the new key(s), to filter out any that are already installed
/usr/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you are prompted now it is to install the new keys
himank@172.16.134.134's password:

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'himank@172.16.134.134'"
and check to make sure that only the key(s) you wanted were added.

himank@himank:~$ ssh himank@172.16.134.134
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.15.0-175-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage
```

```
Activities Terminal ▾ April 15 14:31
himank@himank: ~

Number of key(s) added: 1

Now try logging into the machine, with: "ssh 'himank@172.16.134.134'" and check to make sure that only the key(s) you wanted were added.

himank@himank:~$ ssh himank@172.16.134.134
Welcome to Ubuntu 18.04.6 LTS (GNU/Linux 4.15.0-175-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

System information as of Fri Apr 15 09:00:36 UTC 2022

System load: 0.5 Processes: 96
Usage of /: 63.5% of 8.79GB Users logged in: 1
Memory usage: 21% IP address for enp0s3: 172.16.134.134
Swap usage: 0% IP address for docker0: 172.17.0.1

* Super-optimized for small spaces - read how we shrank the memory footprint of MicroK8s to make it the smallest full K8s around.

https://ubuntu.com/blog/microk8s-memory-optimisation

23 updates can be applied immediately.
To see these additional updates run: apt list --upgradable

New release '20.04.4 LTS' available.
Run 'do-release-upgrade' to upgrade to it.

Last login: Fri Apr 15 09:00:16 2022
himank@ubuntu10:~$ exit
=exit: command not found
himank@ubuntu10:~$ exit
logout
Connection to 172.16.134.134 closed.
himank@himank:~$
```

- Create new IntelliJ Maven project.

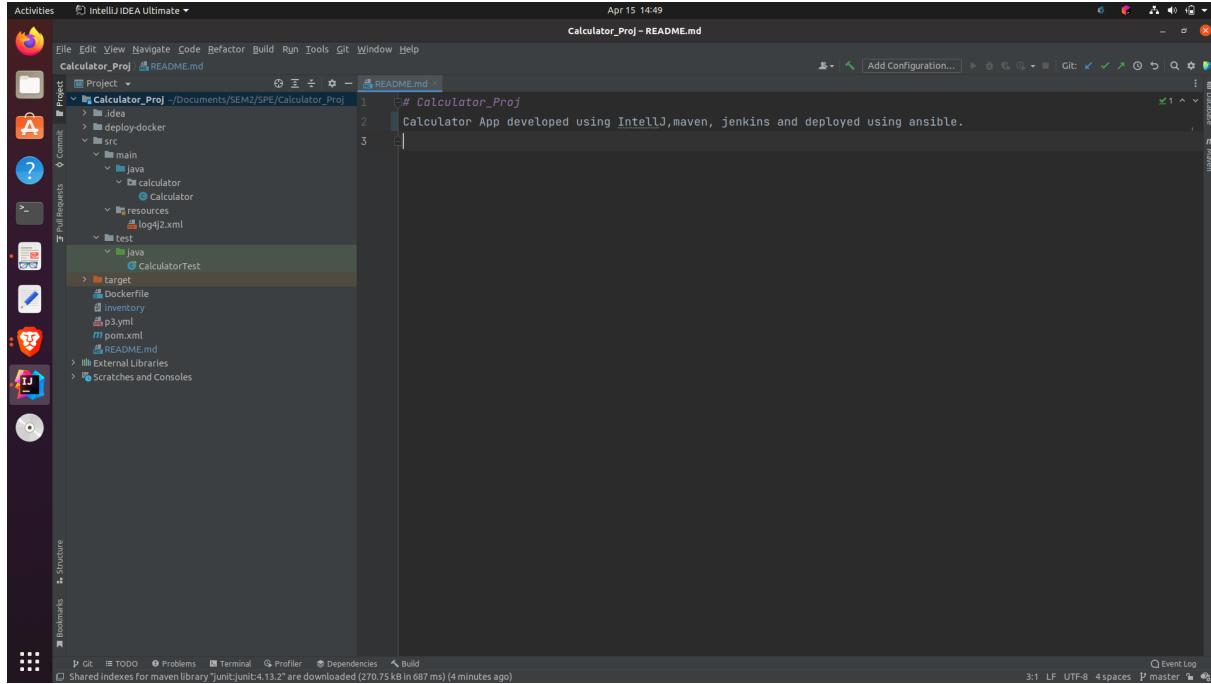
IntelliJ → New Project → Maven → [Give it a name] → Finish

- Ensure a project folder structure like this

- **p3.yml**: Playbook for ansible remote deployment

- **inventory:** file for ansible connection to remote server

- **pom.xml:** Project Object Model i.e. maven file for dependencies.
- **Dockerfile:** docker file containing instructions to build the image with final jar file and run it.
- **src/main/java/calculator/Calculator.java:** Java file containing code for scientific calculator
- **src/test/java/CalculatorTest.java:** Java file containing testing code for scientific calculator



DockerFile: Docker file containing steps to build image.

```
FROM openjdk:8
COPY ./target/calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar .
WORKDIR .
CMD ["java", "-jar", "calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar"]
```

- uses openjdk:8 as base image for our new image
- copies the jar file from local file system to image
- changes the working directory
- executes java -jar command to run the jar file.

p3.yml: Ansible Playbook specifying the commands to execute on remote server.

```
---
- name: Pull docker image of Calculator
  hosts: all
  tasks:
    - name: Start docker service
      service:
        name: docker
        state: started
    - name: pull docker image
      shell: docker pull himankjn/check
```

```
- name: running container
  shell: docker run -it -d himankjn/check
```

- Starts docker in remote server.
- pulls the docker image himankjn/check from dockerhub
- runs the image and creates a container in remote server.

inventory: ansible file with information about remote server

```
[ubuntu18]
172.16.134.134 ansible_user=himank
```

- specifies remote server ip address and user account.

src/main/java/calculator/Calculator.java: Java file containing code for scientific calculator

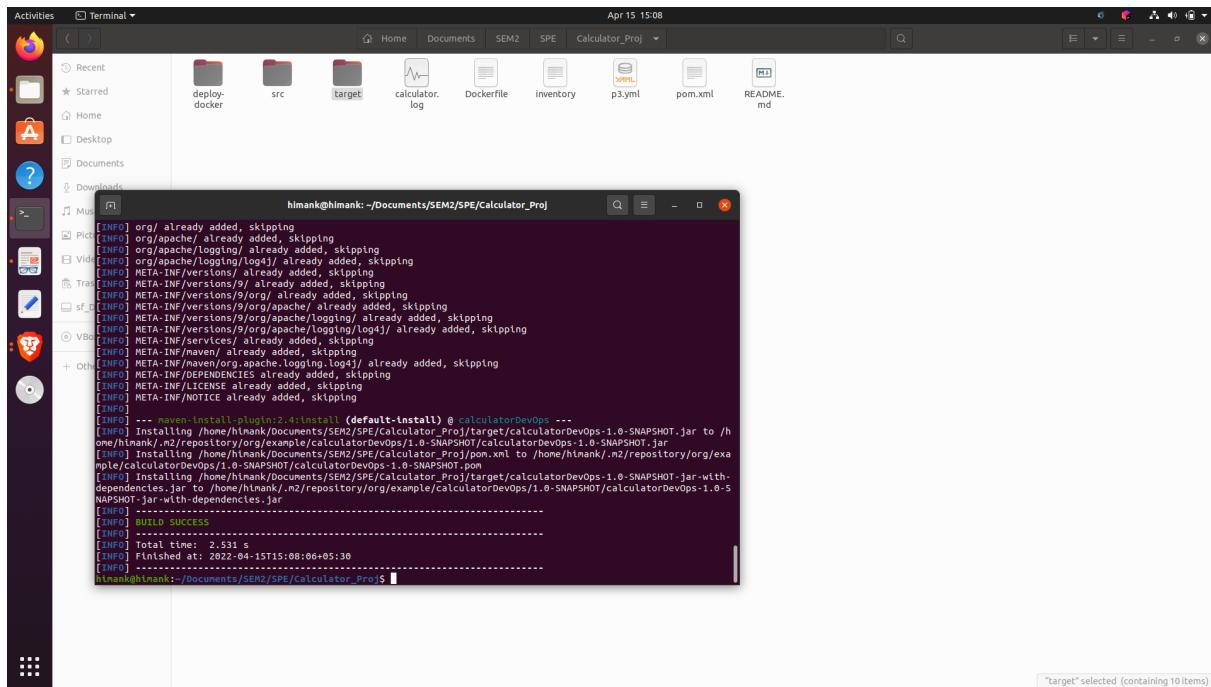
src/test/java/CalculatorTest.java: Java file containing testing code for scientific calculator

Build project using maven install command

pom.xml: Maven file containing all dependencies.

Testing out the code on local machine:

- `mvn clean install` Takes the project code, compiles it, tests it and converts it into a jar executable. Run this command inside the project folder.



The screenshot shows a Linux desktop environment with a terminal window open. The terminal window title is "himank@himank: ~/Documents/SEM2/SPE/Calculator_Proj". The window displays the output of a Maven 'clean install' command. The output shows various logs and messages indicating the build process, including skipping of already added dependencies and the final successful build message: "BUILD SUCCESS". The terminal window is positioned over a desktop background with icons for deploy-docker, src, target, Dockerfile, inventory, p3.yml, pom.xml, and README.md. The desktop environment includes a dock with icons for Home, Documents, SEM2, SPE, and Calculator_Proj.

```
[INFO] org/ already added, skipping
[INFO] org/apache/ already added, skipping
[INFO] org/apache/logging/log4j/ already added, skipping
[INFO] META-INF/versions/9/org/apache/logging/log4j/ already added, skipping
[INFO] META-INF/versions/9/org/apache/logging/log4j/already added, skipping
[INFO] META-INF/versions/9/org/apache/logging/log4j/ already added, skipping
[INFO] META-INF/maven/ already added, skipping
[INFO] META-INF/maven/org.apache.logging.log4j/ already added, skipping
[INFO] META-INF/DEPENDENCIES already added, skipping
[INFO] META-INF/LICENSE already added, skipping
[INFO] META-INF/NOTICE already added, skipping
[INFO] ...
[INFO] --- maven-install-plugin:2.4:install (default-install) @ calculatorDevOps ---
[INFO] Installing /home/himank/Documents/SEM2/SPE/Calculator_Proj/target/calculatorDevOps-1.0-SNAPSHOT.jar to /home/himank/.m2/repository/org/example/calculatorDevOps/1.0-SNAPSHOT/calculatorDevOps-1.0-SNAPSHOT.jar
[INFO] Installing /home/himank/Documents/SEM2/SPE/Calculator_Proj/pom.xml to /home/himank/.m2/repository/org/example/calculatorDevOps/1.0-SNAPSHOT/calculatorDevOps-1.0-SNAPSHOT.pom
[INFO] Installing /home/himank/Documents/SEM2/SPE/Calculator_Proj/target/calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar to /home/himank/.m2/repository/org/example/calculatorDevOps/1.0-SNAPSHOT/calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 2.531 s
[INFO] Finished at: 2022-04-15T15:08:06+05:30
[INFO] -----
[INFO] [helpмаг:~/Documents/SEM2/SPE/Calculator_Proj]
```

- `java -jar ./target/calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar` Tests the snapshot jar file created under target folder on local machine.

```

himank@himank:~/Documents/SEM2/SPE/Calculator_Pro$ java -jar ./target/calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar
Scientific Calculator using DevOps.
Choose operation:
1. Factorial
2. Square root
3. Power
4. Natural Logarithm
5. Exit
Enter your choice: 1
Enter a number : 5
15:09:59.026 [main] INFO  calculator.Calculator - [FACTORIAL] - 5.0
15:09:59.028 [main] INFO  calculator.Calculator - [RESULT - FACTORIAL] - 120.0
Factorial of 5.0 is : 120.0

Scientific Calculator using DevOps.
Choose operation:
1. Factorial
2. Square root
3. Power
4. Natural Logarithm
5. Exit
Enter your choice: ■

```

- Once your code is working flawlessly in local machine, push it to [GitHub](#).

Push to Github:

- push the code repository to [GitHub](#) for version control and automation.

```

#run the following in project folder
git init
git add .
git commit -m "project ready"
git remote add origin https://github.com/himankjn/MiniProject_SPE.git
git push origin master

```

```

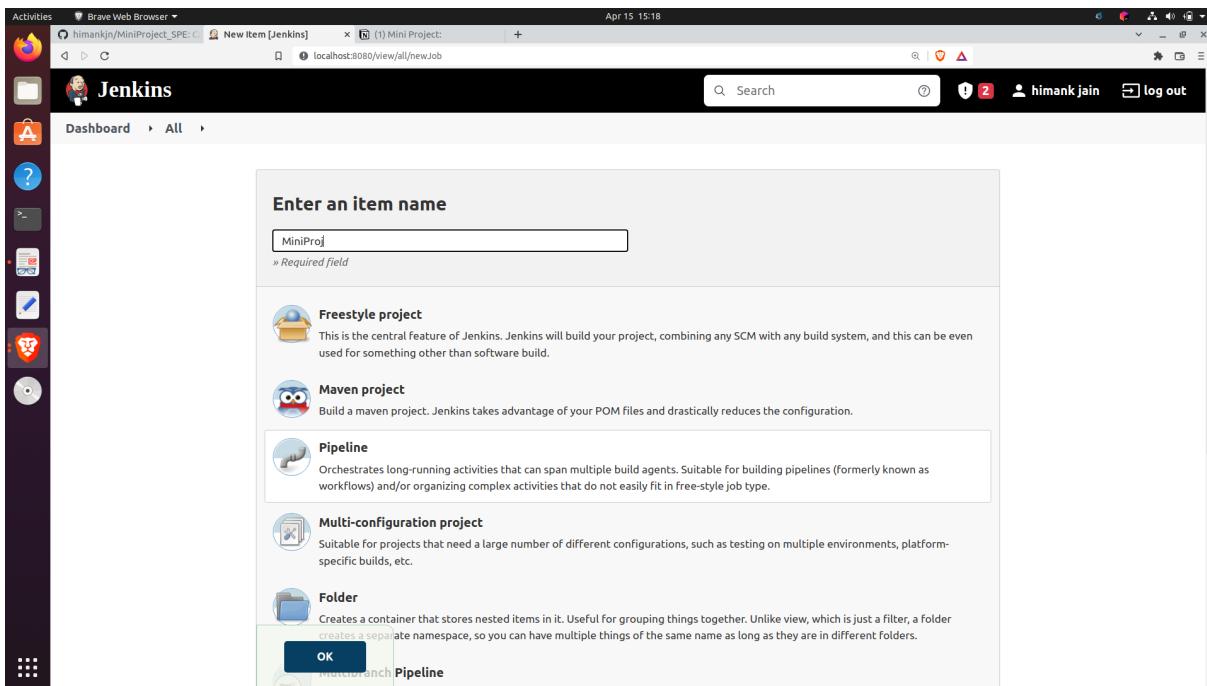
himank@himank:~/Documents/SEM2/SPE/MiniProject_SPE$ git add .
himank@himank:~/Documents/SEM2/SPE/MiniProject_SPE$ git commit -m "project ready"
On branch master
Your branch is ahead of 'origin/master' by 1 commit.
  (use "git push" to publish your local commits)

nothing to commit, working tree clean
himank@himank:~/Documents/SEM2/SPE/MiniProject_SPE$ git push origin master
Username for 'https://github.com': himankjn
Password for 'https://himankjn@github.com':
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 4 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 302 bytes | 302.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To https://github.com/himankjn/MiniProject_SPE
  0b80269..2b54a11  master -> master
himank@himank:~/Documents/SEM2/SPE/MiniProject_SPE$ ■

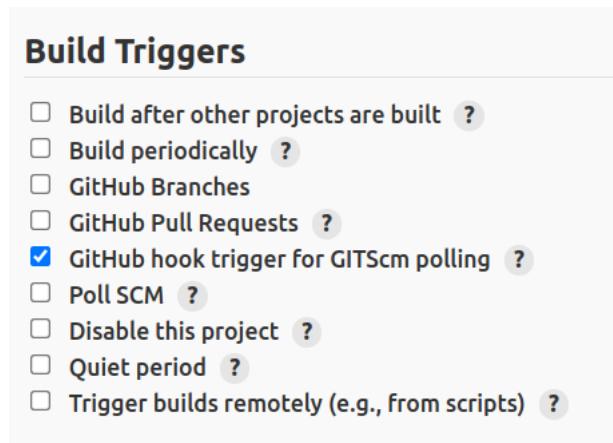
```

Jenkins Project:

- Create a new pipeline project in jenkins with pipeline script and make specific changes according to your credentials.
- **New Item → Pipeline**



- Give Project a name and click ok
- Give it a description.
- Enable **GitHub hook trigger for GITScm polling** under **Build Triggers**



- Under **Pipeline** add following script with required changes

```
pipeline {
    // The "agent" section configures on which nodes the pipeline can be run.
    // Specifying "agent any" means that Jenkins will run the job on any of the
    // available nodes.
    agent any

    environment {
```

```

registry = "himankjn/check"
registryCredential = 'dockerhub'
dockerImage = ''
}

stages {
    stage('Git Pull') {
        steps {
            // Get code from a GitHub repository
            // Make sure to add your own git url and credentialsId
            git url: 'https://github.com/himankjn/MiniProject_SPE.git', branch: 'master'
        }
    }
    stage('Maven Build') {
        steps {
            // Maven build
            sh 'mvn clean install'
        }
    }
    stage('Building our image') {
        steps{
            script {
                dockerImage = docker.build registry + ":latest"
            }
        }
    }
    stage('Deploy our image') {
        steps{
            script {
                docker.withRegistry( '', registryCredential ) {
                    dockerImage.push()
                }
            }
        }
    }
}

stage('Ansible Deploy') {
    steps {
        //Ansible Deploy to remote server (managed host)
        ansiblePlaybook colorized: true, disableHostKeyChecking: true, installation: 'Ansible', inventory: 'inventory', playbook: 'playbook'
    }
}
}
}

```

Stages in Pipeline:

1. Git Pull

It is used to pull remote repository from your github using Jenkins

```

stage('Git pull'){
    steps{
        git url: 'https://github.com/himankjn/MiniProject_SPE.git', branch: 'master'
    }
}

```

2. Maven Build

It is used to create the jar file which will contain our source code along with dependencies.

```

stage('Maven Build'){
    steps{
        sh 'mvn clean install'
    }
}

```

3. Docker Image Creation

It is used to create images in our local system which is pushed onto our docker hub so that we can pull the image and run the application on different servers.

```
environment{
    registry = 'himankjn/calculator'
    dockerImage = ''
}
stage('Docker Image Creation'){
    steps{
        script{
            dockerImage = docker.build registry + ':latest'
        }
    }
}
```

`:latest` is the tag name of the image

`docker build -t imagename .` will build an image according to the DockerFile contents in current directory.

4. Deploying the Docker Image to Docker Hub

use the above created docker credentials and image to push the image to docker hub.

```
environment{
    registryCredential: dockerhubcred
}
```

```
stage("Pushing Image to Docker Hub"){
    steps{
        script{
            docker.withRegistry( '', registryCredential ){
                dockerImage.push()
            }
        }
    }
}
```

5. Ansible Deploy

It builds Infrastructure for remote servers using the playbook.

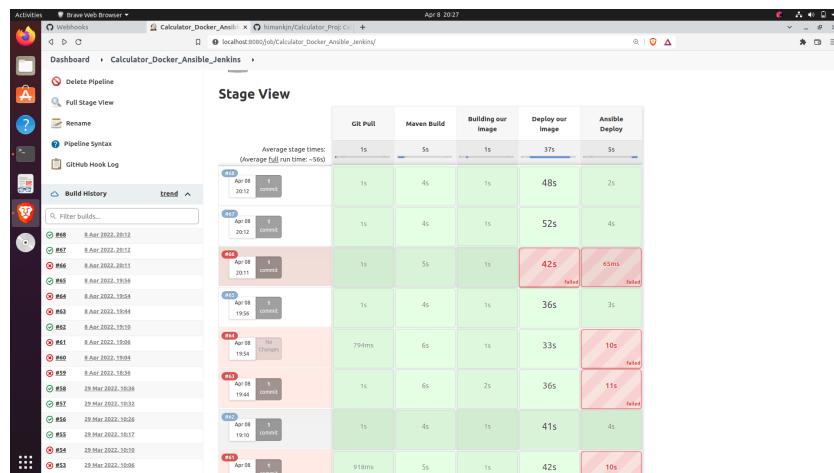
ansible-playbook command uses 3 files:

- playbook **[p3.yml]**: Steps to perform in remote server
- **inventory**: IP address and username of remote server

```
stage('Ansible Deploy') {
    steps {
        ansiblePlaybook colorized: true, disableHostKeyChecking: true, installation: 'Ansible', inventory: 'inventory', playbook: 'p3.yml'
    }
}
```

6. Jenkins Build:

Click on Build Now and let the pipeline run.



7. Run the image in a remote server

Once the pipeline executes ansible playbook would have pulled the image from docker hub to remote server.

check the image availability in remote server using: `docker images`

```
File Machine View Input Devices Help
himank@ubuntu18:~$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
himankjn/check latest fcc56fe19e9d 3 hours ago 528MB
himankjn/check <none> ed458c021449 4 hours ago 528MB
himank@ubuntu18:~$
```

Check if the container is created and running in background: `docker ps`

```
himank@ubuntu18:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
e5a9b46b3854 himankjn/check "java -jar calculato..." 3 seconds ago Up 2 seconds
vial_wing
himank@ubuntu18:~$
```

Run the docker container by attaching it to a terminal using : `docker exec -it e5 /bin/bash`

Tip: we can specify just the first 2-3 letters of the container id and it still works.

```
himank@ubuntu18:~$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
e5a9b46b3854 himankjn/check "java -jar calculato..." About a minute ago Up About a minute
jovial_wing
himank@ubuntu18:~$ docker exec -it e5 /bin/bash
root@e5a9b46b3854:/# ls
bin dev lib64 proc srv var
boot etc media root sys
calculator.log home mnt run tmp
calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar lib opt sbin usr
root@e5a9b46b3854:/#
```

once inside the container, List the files using `ls`

run the executable jar using `java -jar calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar`

```
jovial_wing
himank@ubuntu18:~$ docker exec -it e5 /bin/bash
root@e5a9b46b3854:/# ls
bin                               dev   lib64  proc  srv  var
boot                             etc   media  root  sys
calculator.log                    home  mnt   run   tmp
calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar  lib   opt   sbin  usr
root@e5a9b46b3854:/# ls
bin                               dev   lib64  proc  srv  var
boot                             etc   media  root  sys
calculator.log                    home  mnt   run   tmp
calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar  lib   opt   sbin  usr
root@e5a9b46b3854:/# java -jar calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar
Scientific Calculator using DevOps.
Choose operation:
1. Factorial
2. Square root
3. Power
4. Natural Logarithm
5. Exit
Enter your choice: 1
Enter a number : 5
10:44:11.737 [main] INFO calculator.Calculator - [FACTORIAL] - 5.0
10:44:11.741 [main] INFO calculator.Calculator - [RESULT - FACTORIAL] - 120.0
Factorial of 5.0 is : 120.0
```

check the log file `cat calculator.log`

```
Scientific Calculator using DevOps.
Choose operation:
1. Factorial
2. Square root
3. Power
4. Natural Logarithm
5. Exit
Enter your choice: 5
Exiting...
root@e5a9b46b3854:/# cat calculator.log
2022-04-15 10:44:11.737 [main] INFO calculator.Calculator - [FACTORIAL] - 5.0
2022-04-15 10:44:11.741 [main] INFO calculator.Calculator - [RESULT - FACTORIAL] - 120.0
2022-04-15 10:44:59.322 [main] INFO calculator.Calculator - [FACTORY] - 5.0
2022-04-15 10:44:59.327 [main] INFO calculator.Calculator - [RESULT - FACTORIAL] - 120.0
2022-04-15 10:45:01.405 [main] INFO calculator.Calculator - [SQ ROOT] - 4.0
2022-04-15 10:45:01.406 [main] INFO calculator.Calculator - [RESULT - SQ ROOT] - 2.0
2022-04-15 10:45:04.121 [main] INFO calculator.Calculator - [POWER - 1.0 RAISED TO] 2.0
2022-04-15 10:45:04.122 [main] INFO calculator.Calculator - [RESULT - POWER] - 1.0
root@e5a9b46b3854:/#
```

8. ELK STACK:

ELK Stack is a collection of three open-source products — Elasticsearch, Logstash, and Kibana. ELK stack provides centralized logging in order to identify problems with servers or applications. It allows you to search all the logs in a single place. It also helps to find issues in multiple servers by connecting logs during a specific time frame.

E stands for ElasticSearch: used for storing logs

L stands for LogStash : used for both shipping as well as processing and storing logs

K stands for Kibana: is a visualization tool (a web interface) which is hosted through Nginx or Apache.

- Upload your log file by clicking on Upload a file

calculator-mini-project.kb.us-central1.gcp.cloud.es.io:9243/app/home/

Welcome home

Enterprise Search
Create search experiences with a refined set of APIs and tools.

Observability
Consolidate your logs, metrics, application traces, and system availability with purpose-built UIs.

Security
Prevent, collect, detect, and respond to threats for unified protection across your infrastructure.

Analytics
Explore, visualize, and analyze your data using a powerful suite of analytical tools and applications.

Get started by adding integrations

To start working with your data, use one of our many ingest options. Collect data from an app or service, or upload a file. If you're not ready to use your own data, add a sample data set.

[Add integrations](#) [Try sample data](#) [Upload a file](#)

Management [Dev Tools](#) [Stack Management](#)

calculator-mini-project.kb.us-central1.gcp.cloud.es.io:9243/app/home#/tutorial_directory/fileDataViz

More ways to add data

In addition to adding [integrations](#), you can try our sample data or upload your own data.

[Sample data](#) [Upload file](#)

calculator.log

File contents
First 29 lines

```

1 2022-04-08 20:04:53.198 [main] INFO calculator.Calculator : [FACTORIAL] - 5.0
2 2022-04-08 20:04:53.282 [main] INFO calculator.Calculator : [RESULT - FACTORIAL] - 120.0
3 2022-04-08 20:04:53.283 [main] INFO calculator.Calculator : [LOG] - 4.386852819441458
4 2022-04-08 20:04:53.283 [main] INFO calculator.Calculator : [RESULT - FACTORIAL] - 24.0
5 2022-04-08 20:04:53.286 [main] INFO calculator.Calculator : [LOG] - 3.40123918954054
6 2022-04-08 20:04:53.286 [main] INFO calculator.Calculator : [RESULT - SQ ROOT] - 2.0
7 2022-04-08 20:04:53.286 [main] INFO calculator.Calculator : [LOG] - 0.6931471805599453
8 2022-04-08 20:04:53.286 [main] INFO calculator.Calculator : [RESULT - SQ ROOT] - 0.6
9 2022-04-08 20:04:53.286 [main] INFO calculator.Calculator : [LOG] - 0.0
10 2022-04-08 20:04:53.210 [main] INFO calculator.Calculator : [RESULT - NATURAL LOG] - 0.0
11 2022-04-08 20:04:53.210 [main] INFO calculator.Calculator : [LOG] - 0.0
12 2022-04-08 20:04:53.221 [main] INFO calculator.Calculator : [RESULT - FACTORIAL] - 120.0
13 2022-04-08 20:04:53.221 [main] INFO calculator.Calculator : [LOG] - 4.386852819441458
14 2022-04-08 20:04:53.221 [main] INFO calculator.Calculator : [RESULT - FACTORIAL] - 24.0
15 2022-04-08 20:04:53.226 [main] INFO calculator.Calculator : [LOG] - 3.40123918954054
16 2022-04-08 20:04:53.226 [main] INFO calculator.Calculator : [RESULT - SQ ROOT] - 2.0
17 2022-04-08 20:04:53.226 [main] INFO calculator.Calculator : [LOG] - 0.6931471805599453
18 2022-04-08 20:04:53.226 [main] INFO calculator.Calculator : [RESULT - NATURAL LOG] - 0.0
19 2022-04-08 20:04:53.230 [main] INFO calculator.Calculator : [LOG] - 0.0
20 2022-04-08 20:04:53.230 [main] INFO calculator.Calculator : [RESULT - NATURAL LOG] - 0.0
21 2022-04-08 20:04:53.230 [main] INFO calculator.Calculator : [LOG] - 0.0
22 2022-04-08 20:04:53.230 [main] INFO calculator.Calculator : [RESULT - FACTORIAL] - 120.0
23 2022-04-08 20:04:53.230 [main] INFO calculator.Calculator : [LOG] - 4.386852819441458
24 2022-04-08 20:04:53.230 [main] INFO calculator.Calculator : [RESULT - FACTORIAL] - 24.0
25 2022-04-08 20:04:53.230 [main] INFO calculator.Calculator : [LOG] - 3.40123918954054
26 2022-04-08 20:04:53.230 [main] INFO calculator.Calculator : [RESULT - SQ ROOT] - 2.0
27 2022-04-08 20:04:53.230 [main] INFO calculator.Calculator : [LOG] - 0.6931471805599453
28 2022-04-08 20:04:53.230 [main] INFO calculator.Calculator : [RESULT - NATURAL LOG] - 0.0
29 2022-04-08 20:04:53.237 [main] INFO calculator.Calculator : [LOG] - 0.0

```

Summary

Number of lines analyzed	29
Format	semi_structured_text
Grok pattern	%{TIMESTAMP_ISO8601:timestamp} %{.*?} %{LOGLEVEL:loglevel}:.*? .? .*? .? .*%{NUMBER:field},*
Time field	timestamp
Time format	yyyy-MM-dd HH:mm:ss.SSS

[Import](#) [Cancel](#)

b. Click on Import.

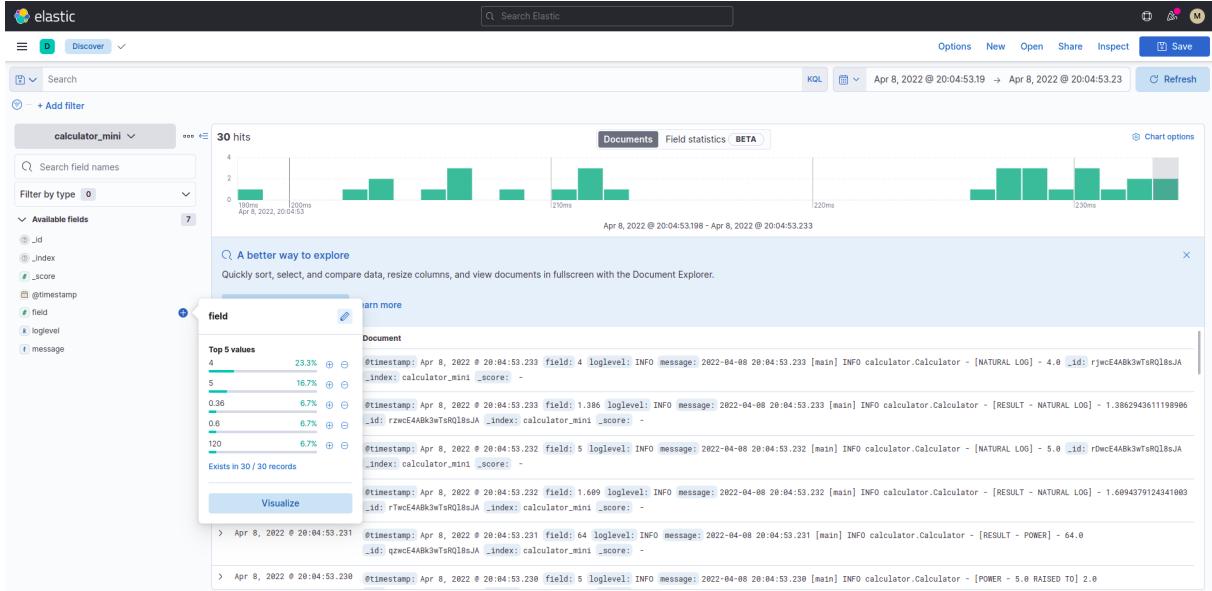
c. Add index name and click on import

The screenshot shows the 'calculator.log' file being imported into the 'calculator_min' index. The 'Simple' import method is selected. The 'Index name' field contains 'calculator_min'. The 'Create data view' checkbox is checked. A blue 'Import' button is visible at the bottom.

d. Click on View index to discover

The screenshot shows the import process completed successfully. The status bar indicates: File processed, Index created, Ingest pipeline created, Data uploaded, and Data view created. Below this, a summary table shows: Index (calculator_min), Data view (calculator_min), Ingest pipeline (calculator_mini-pipeline), and Documents ingested (30). At the bottom, four buttons are available: View index in Discover, Index Management, Data View Management, and Create Filebeat configuration. Navigation buttons Back and Cancel are at the very bottom.

e. click on visualize

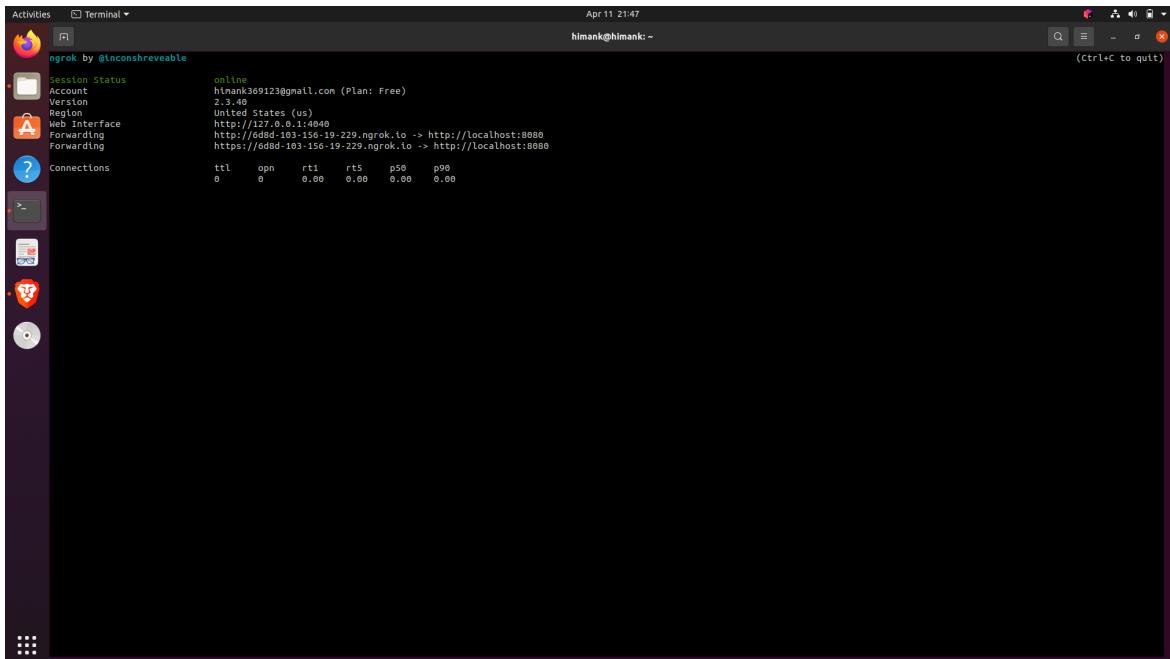


Using Github Webhook for automated builds:

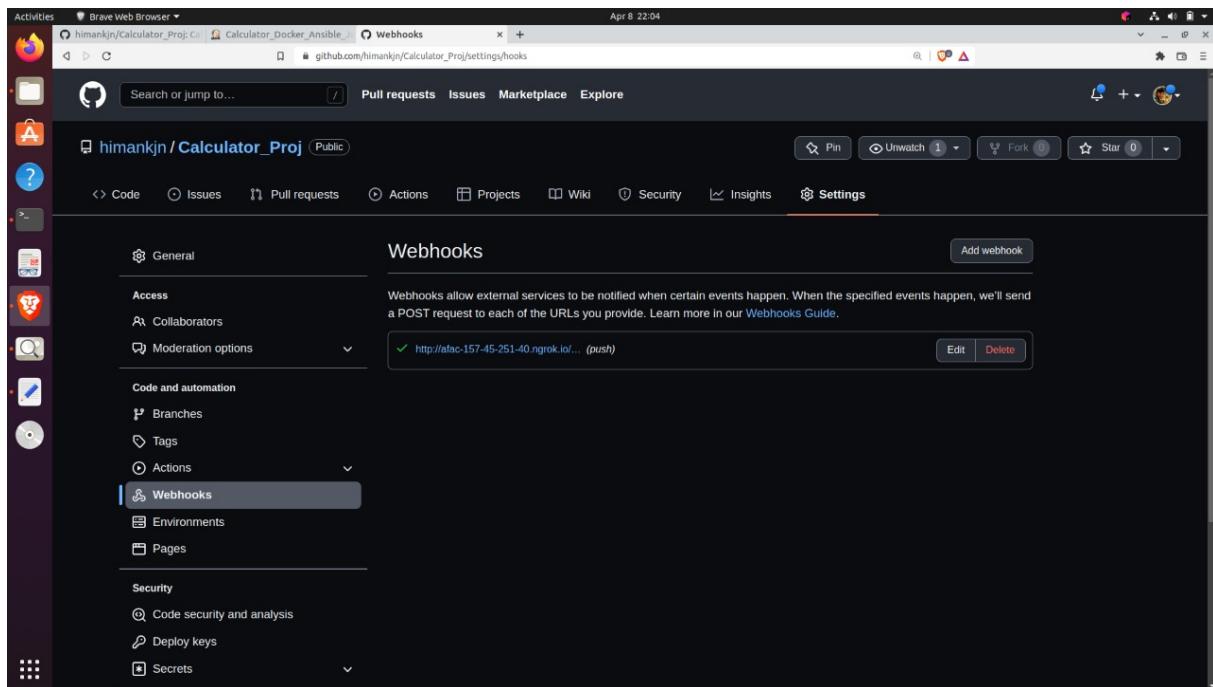
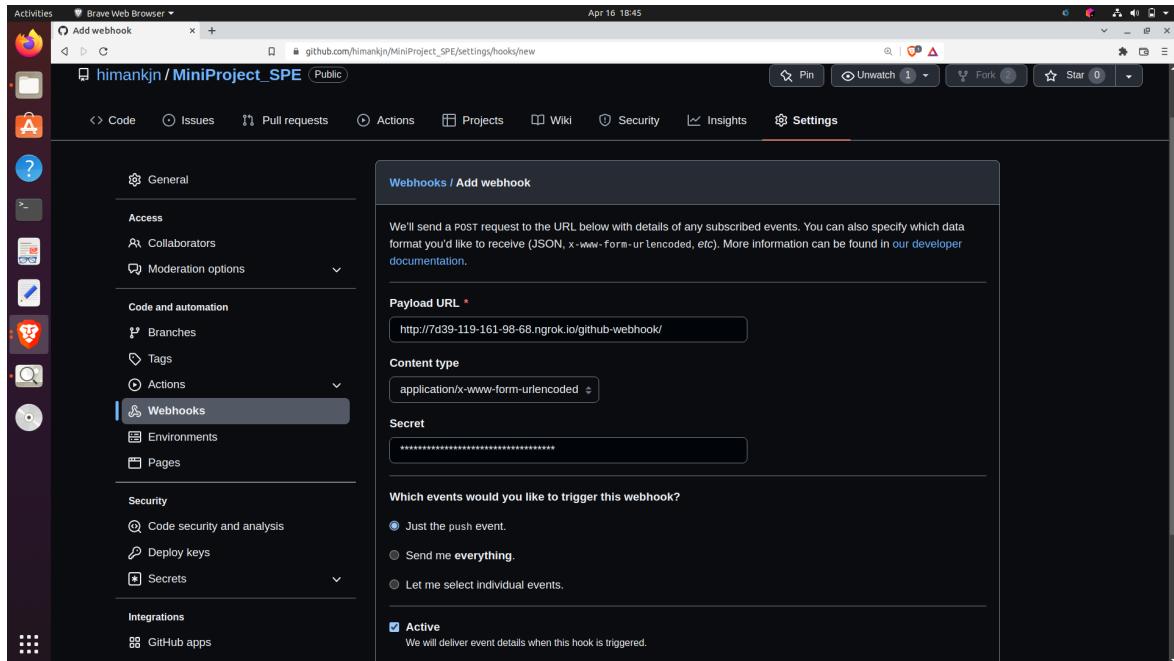
Expose the jenkins address to the internet by making it public using ngrok.

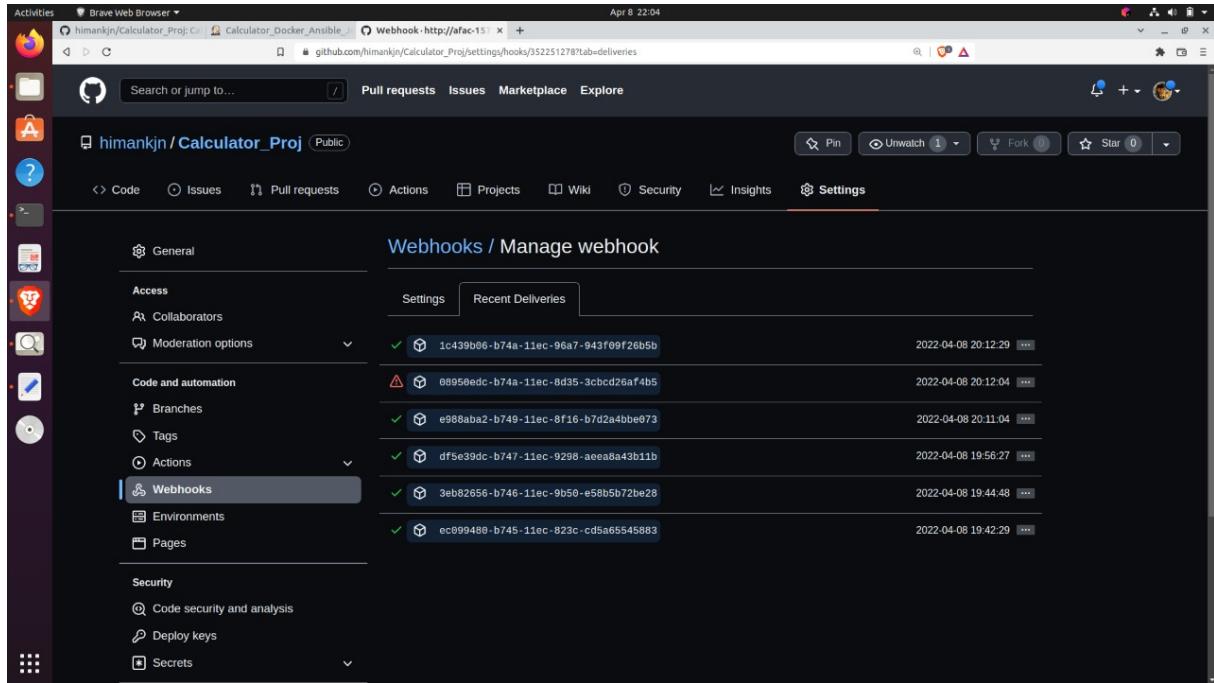
- Sign up in <https://ngrok.com/>
- Download ngrok from: <https://bin.equinox.io/c/4VmDzA7iaHb/ngrok-stable-linux-amd64.tgz>
- Extract ngrok from the terminal: \$sudo tar xvzf ~/Downloads/ngrok-stable-linux-amd64.tgz -C /usr/local/bin
- Copy Authtoken from: <https://dashboard.ngrok.com/get-started/your-authtoken>
- Add Authtoken: \$ngrok authtoken <token>
- Execute `ngrok http 172.16.134.225:8080;`

This generates a public address that redirects to localhost:8080



- Navigate to Github Project Repo → Settings → Webhooks:
 - Payload URL: ngrok_generated_IP/github-webhook
<http://7d39-119-161-98-68.ngrok.io/github-webhook/>
 - Secret: Personal Access Token





Application Screenshots:

The Calculator Application has a CLI interface and provides following options:

```
Activities Terminal Apr 18 12:07
himank@himank:~/Documents/SEM2/SPE/MiniProject_SPE/target
calculatorDevOps-1.0-SNAPSHOT.jar
calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar
calculator.log
calculator
himank@himank:~/Documents/SEM2/SPE/MiniProject_SPE/target$ java -jar calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar
Scientific calculator using DevOps.
Choose operation:
1. Factorial
2. Square root
3. Power
4. Natural Logarithm
5. Exit
Enter your choice: 1
Enter a number : 3
12:07:04.191 [main] INFO calculator.Calculator - [FACTORIAL] - 3.0
12:07:04.193 [main] INFO calculator.Calculator - [RESULT - FACTORIAL] - 6.0
Factorial of 3.0 is : 6.0

Scientific calculator using DevOps.
Choose operation:
1. Factorial
2. Square root
3. Power
4. Natural Logarithm
5. Exit
Enter your choice: 2
Enter a number : 4
12:07:04.417 [main] INFO calculator.Calculator - [SQ ROOT] - 4.0
12:07:04.417 [main] INFO calculator.Calculator - [RESULT - SQ ROOT] - 2.0
Square root of 4.0 is : 2.0

Scientific Calculator using DevOps.
Choose operation:
1. Factorial
2. Square root
3. Power
4. Natural Logarithm
5. Exit
Enter your choice: 3
Enter the first number : 2
Enter the second number : 3
12:07:07.429 [main] INFO calculator.Calculator - [POWER - 2.0 RAISED TO] 3.0
12:07:07.429 [main] INFO calculator.Calculator - [RESULT - POWER] - 8.0
2.0 raised to power 3.0 is : 8.0

Scientific Calculator using DevOps.
Choose operation:
1. Factorial
2. Square root
3. Power
4. Natural Logarithm
5. Exit
Enter your choice: 4
```

```

Activities Terminal ▾ Apr 18 12:08
himank@himank: ~/Documents/SEM2/SPE/MiniProject_SPE/target

Scientific Calculator using DevOps.
Choose operation:
1. Factorial
2. Square root
3. Power
4. Natural Logarithm
5. Exit
Enter your choice: 2
Enter a number : 4
12:07:04.417 [main] INFO calculator.Calculator - [SQ ROOT] - 4.0
12:07:04.417 [main] INFO calculator.Calculator - [RESULT - SQ ROOT] - 2.0
Square root of 4.0 is : 2.0

Scientific Calculator using DevOps.
Choose operation:
1. Factorial
2. Square root
3. Power
4. Natural Logarithm
5. Exit
Enter your choice: 3
Enter the first number : 2
Enter the second number : 3
12:07:07.429 [main] INFO calculator.Calculator - [POWER - 2.0 RAISED TO] 3.0
12:07:07.429 [main] INFO calculator.Calculator - [RESULT - POWER] - 8.0
2.0 raised to power 3.0 is : 8.0

Scientific Calculator using DevOps.
Choose operation:
1. Factorial
2. Square root
3. Power
4. Natural Logarithm
5. Exit
Enter your choice: 4
Enter a number : 5
12:07:14.312 [main] INFO calculator.Calculator - [NATURAL LOG] - 5.0
12:07:14.313 [main] INFO calculator.Calculator - [RESULT - NATURAL LOG] - 1.6094379124341003
Natural log of 5.0 is : 1.6094379124341003

Scientific Calculator using DevOps.
Choose operation:
1. Factorial
2. Square root
3. Power
4. Natural Logarithm
5. Exit
Enter your choice: 5
Exiting...

```

Summary:

- Download and install required tools in host and server machines.
- Generate & Exchange ssh keys so that host and remote server can communicate without password.
- Create Maven Project containing mainly the following files:
 - **p3.yml:** Playbook for ansible remote deployment
 - **inventory:** file for ansible connection to remote server
 - **pom.xml:** Project Object Model i.e. maven file for dependencies.
 - **Dockerfile:** docker file containing instructions to build the image with final jar file and run it.
 - **src/main/java/calculator/Calculator.java:** Java file containing code for scientific calculator
 - **src/test/java/CalculatorTest.java:** Java file containing testing code for scientific calculator
- Push your code to [Github](#)
- Create [DockerHub](#) Repository
- Create Jenkins Pipeline Project, write the pipeline script containing the following steps:
 1. **Git Pull**
 2. **Maven** build
 3. **Docker** Image creation
 4. Pushing Image to **Docker Hub**
 5. **Ansible** Deploy
- Setup Jenkins Configuration for **Github** Webhook using **ngrok**.
- Build the project.
- Run the docker image as a container in remote server.

- Upload the generated log file in **ELK Stack**.

GitHub - himankjn/Calculator_Proj: Calculator App developed using maven & jenkins and deployed using ansible
You can't perform that action at this time. You signed in with another tab or window. You signed out in another tab or window.
Reload to refresh your session. Reload to refresh your session.

https://github.com/himankjn/Calculator_Proj

Dependencies & Code:

Dependencies in the code can be resolved using Maven. Dependencies are the external libraries which we use to integrate it within our code. All of the dependencies can be searched from <https://mvnrepository.com/artifact/org.apache.maven.plugins/maven-dependency-plugin> and these can be included in POM.xml. Maven will automatically download the dependencies and will also indicate if there are any more requirements.

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>org.example</groupId>
    <artifactId>calculatorDevOps</artifactId>
    <version>1.0-SNAPSHOT</version>
    <build>
        <plugins>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-assembly-plugin</artifactId>
                <executions>
                    <execution>
                        <phase>package</phase>
                        <goals>
                            <goal>single</goal>
                        </goals>
                        <configuration>
                            <archive>
                                <manifest>
                                    <mainClass>calculator.Calculator</mainClass>
                                </manifest>
                            </archive>
                            <descriptorRefs>
                                <descriptorRef>jar-with-dependencies</descriptorRef>
                            </descriptorRefs>
                        </configuration>
                    </execution>
                </executions>
            </plugin>
        </plugins>
    </build>
</project>
```

Log4j2.xml:

We have added lo4j2.xml which will create a log file for test cases. For creation of the same, some build properties also have to be added in pom.xml.

The log4j2.properties handles projects log functionalities.

```

<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="INFO">
    <Appenders>
        <Console name="ConsoleAppender" target="SYSTEM_OUT">
            <PatternLayout pattern="%d{HH:mm:ss.SSS} [%t] %-5level %logger{36} - %msg%n" />
        </Console>
        <File name="FileAppender" fileName="calculator.log" immediateFlush="false" append="true">
            <PatternLayout pattern="%d{yy-MM-dd HH:mm:ss.SSS} [%t] %-5level %logger{36} - %msg%n"/>
        </File>
    </Appenders>
    <Loggers>
        <Root level="debug">
            <AppenderRef ref="ConsoleAppender" />
            <AppenderRef ref="FileAppender" />
        </Root>
        <Logger name="CalculatorTest" level="info" additivity="false">
            <AppenderRef ref="FileAppender" />
        </Logger>
    </Loggers>
</Configuration>

```

Unit Testing:

For unit testing Junit dependencies have been added and test cases have been written for false as well as true assertions.

```

import ...
public class CalculatorTest {
    private static final double DELTA = 1e-15;
    Calculator calculator = new Calculator();

    @Test
    public void factorialTruePositive() {
        assertEquals("Finding factorial of a number for True Positive", expected:120, calculator.factorial(5));
        assertEquals("Finding factorial of a number for True Positive", expected:24, calculator.factorial(4));
    }

    @Test
    public void factorialFalsePositive() {
        assertNotEquals("Finding factorial of a number for False Positive", unexpected:120, calculator.factorial(6));
        assertNotEquals("Finding factorial of a number for False Positive", unexpected:24, calculator.factorial(5));
    }

    @Test
    public void powerTruePositive() {
        assertEquals("Finding power for True Positive", expected:8, calculator.power(2, 3), DELTA);
        assertEquals("Finding power for True Positive", expected:64, calculator.power(4, 3), DELTA);
    }

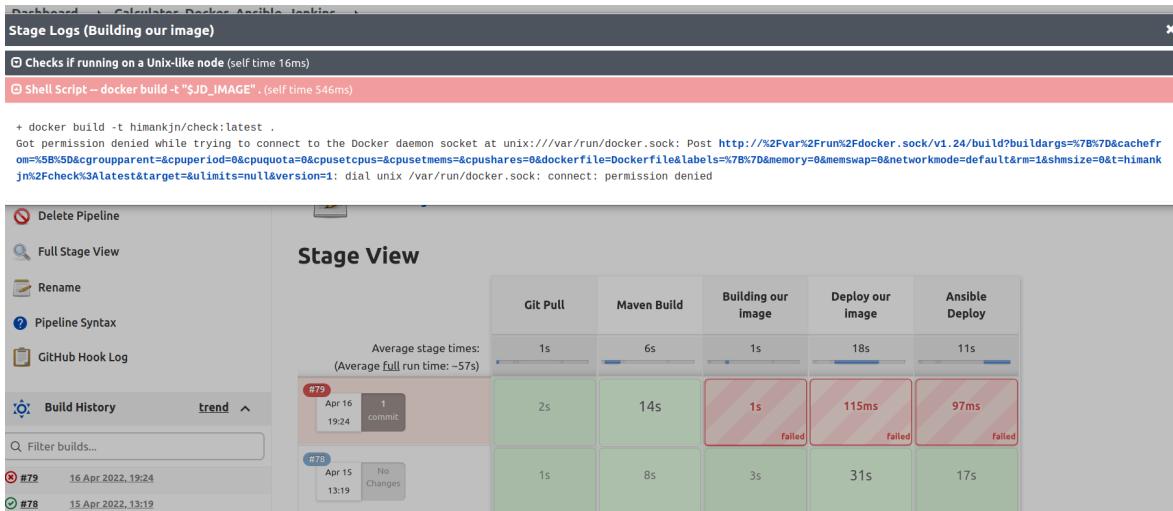
    @Test
    public void powerFalsePositive() {
        assertNotEquals("Finding power for False Positive", unexpected:6, calculator.power(2, 2), DELTA);
        assertNotEquals("Finding power for False Positive", unexpected:-7.3, calculator.power(2, 3), DELTA);
    }
}

```

CHALLENGES AND SOLUTIONS:

1. Docker Daemon Permission

- **Error:** permission denied while trying to connect to Docker daemon socket at unix:///var/run/docker.sock
- **Solution:** `sudo chmod 666 varrun/docker.sock` on both server and local machine



2. JDK version mismatch

- **Error:** Image creation using java 8 base image but installing java 11 in remote server
- **Solution:** changing the base image in Dockerfile from openjdk:8 to openjdk:11

The screenshot shows a GitHub repository page for 'himankjn / MiniProject_SPE'. The 'Code' tab is selected. The Dockerfile content is displayed, showing four lines of code:

```
FROM openjdk:8
COPY ./target/calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar .
WORKDIR .
CMD ["java", "-jar", "calculatorDevOps-1.0-SNAPSHOT-jar-with-dependencies.jar"]
```

The first line, 'FROM openjdk:8', is circled in red.

3. Docker Hub default image tag

- **Error:** The default image tag being pulled by remote server was `:latest` but the one we pushed through pipeline was the `:$BUILD_NUMBER`

- **Solution:** Changing the image tag in pipeline to :latest

4. pip and pip-ansible in remote server:

- **Error:** remote server was unable to pull the image using ansible and suggested to install ansible through pip
- **Solution:** Install python-pip and `pip install ansible`

5. Webhook:

- **Error:** ngrok webhook through github threw 404 error.
- **Solution:** Ensure that ngrok is running on localhost ip:8080

6. Remote Host Connectivity:

- **Error:** Ansible was unable to connect to remote server using ssh.
- **Solution:** Ensure SSH keys generated and copied to remote server using ssh-copy-id

7. Virtual Machines IP address:

- **Error:** Virtual Box assigned same IP to both machines.
- **Solution:** Use bridge adapter in network settings of virtual box to ensure different ip addresses