Scientific Calculator with DevOps

Submitted By

Priyanshu Vaish, MT2023089

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY, BANGALORE.

26/C, Hosur Rd, Electronics City Phase 1, Electronic City, Bengaluru, Karnataka 560100,
Academic Year 2023-24

GitHub Link

https://github.com/Priyansuvaish/calculator_final.git

DockerHub Link

docker pull priyanshugupta753/calculator

Table of Contents

Introduction

What is DevOps?

DevOps is like a bridge that connects the world of software development (where applications and systems are created) with IT operations (where those applications are deployed and maintained in the real world). This connection is not just about making sure that the two sides talk to each other but also ensuring they work closely together throughout the entire lifecycle of a software system. By doing so, DevOps aims to make the process of developing, testing, deploying, and maintaining software much smoother and faster.

In other words, DevOps is all about making the process of delivering software better, faster, and more efficient. It's like turning a bumpy, winding dirt road into a smooth, straight highway, allowing for quicker and safer delivery of the end product to the customer.

Why DevOps?

There are many benefits of using DevOps some of them are as follows:

- With DevOps, companies can release new features, updates, and fixes more quickly and frequently. This rapid pace allows businesses to respond to market changes better and enhance customer satisfaction.

- Closer collaboration between teams means problems can be detected and resolved faster. This teamwork approach leads to a better understanding of the software and infrastructure, reducing downtime and improving service reliability.

- By automating repetitive tasks and streamlining processes, teams have more time to focus on innovation and improvement. DevOps practices like continuous integration and continuous deployment automate the software release process, from build to deployment.

- DevOps encourages the use of practices such as infrastructure as code (IaC), which allows the management of infrastructure using configuration files. This approach leads to more efficient use of resources, reduces errors, and enhances reproducibility.

Tools Used in DevOps

- Source Control Management (SCM): GitHub
- Build the code: Maven
- Testing: JUnit
- Continuous Integration/Continuous Deployment (CI/CD): Jenkins
- Containerization: Docker
- For Deployment: Ansible

Steps Explanation and Tool Configuration

Source Control Management with GitHub

SCM is the task of tracking and controlling changes in the software, part of the larger cross-disciplinary field of configuration management. GitHub, a cloud-based hosting service, lets you manage Git repositories.

Setup and Configurations:

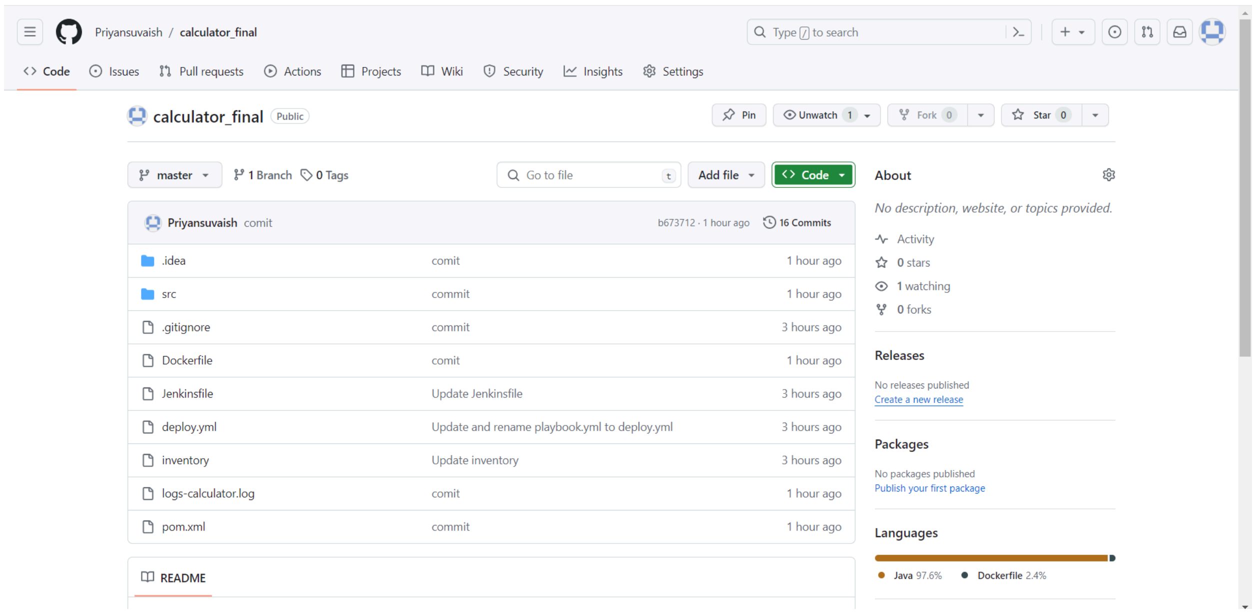
- Create a GitHub account and repository.
- Initialize Git in your project directory: git init

Commands Used:

- git add.
- git commit -m "<commit-message>"
- git push origin master
- git clone "https://github.com/Priyansuvaish/calculator_final.git"

ScreenShot:

Repository image:



calculator_final Public

Priyansuvaish comit b673712 · 1 hour ago 16 Commits

.idea	comit	1 hour ago
src	commit	1 hour ago
.gitignore	commit	3 hours ago
Dockerfile	comit	1 hour ago
Jenkinsfile	Update Jenkinsfile	3 hours ago
deploy.yml	Update and rename playbook.yml to deploy.yml	3 hours ago
inventory	Update inventory	3 hours ago
logs-calculator.log	comit	1 hour ago
pom.xml	commit	1 hour ago

About

No description, website, or topics provided.

Activity
0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

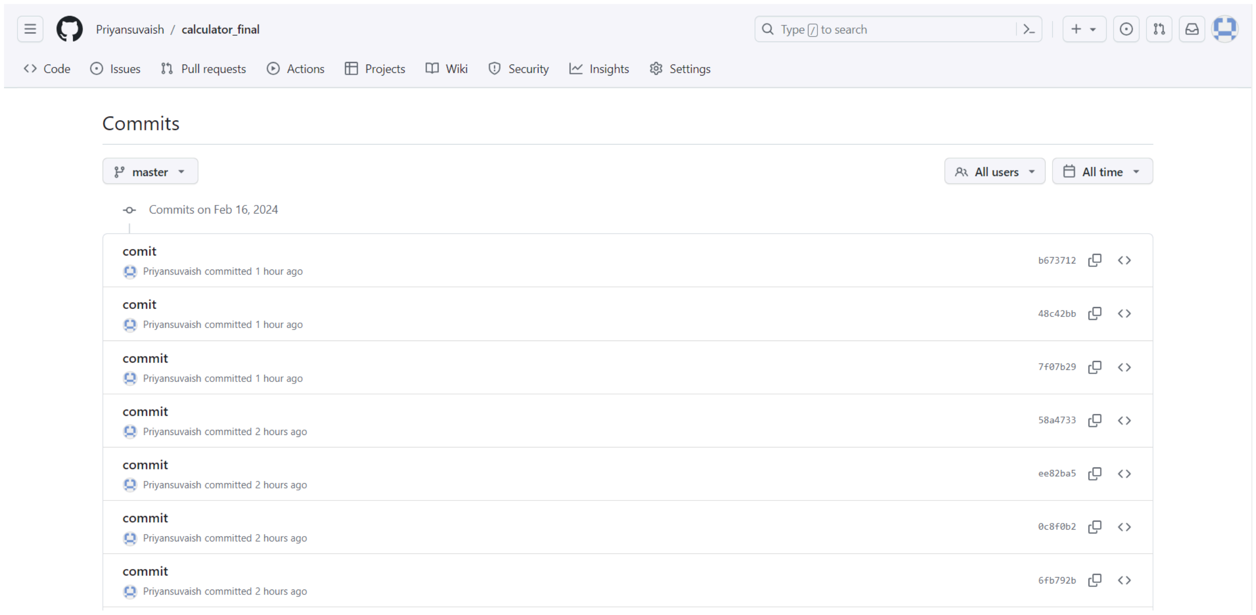
Packages

No packages published
Publish your first package

Languages

Java 97.6% Dockerfile 2.4%

Commit image:



Commits

Commits on Feb 16, 2024

comit
Priyansuvaish committed 1 hour ago — b673712

comit
Priyansuvaish committed 1 hour ago — 48c42bb

commit
Priyansuvaish committed 1 hour ago — 7f07b29

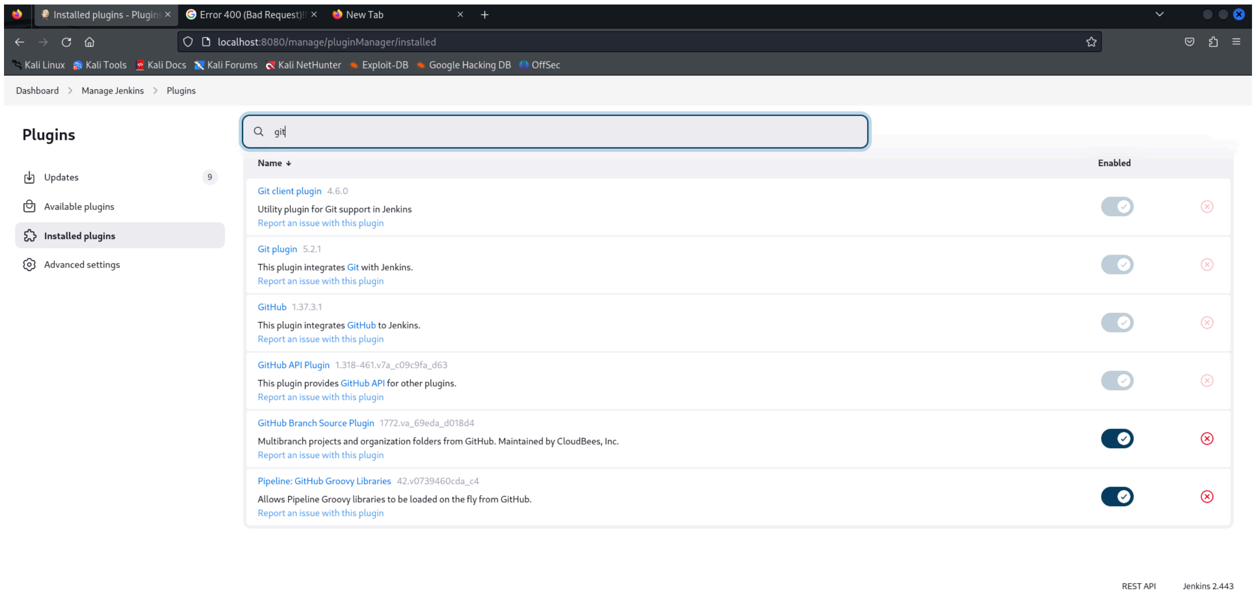
commit
Priyansuvaish committed 2 hours ago — 58a4733

commit
Priyansuvaish committed 2 hours ago — ee82ba5

commit
Priyansuvaish committed 2 hours ago — 0c8f0b2

commit
Priyansuvaish committed 2 hours ago — 6fb792b

Install the GitHub plugins:

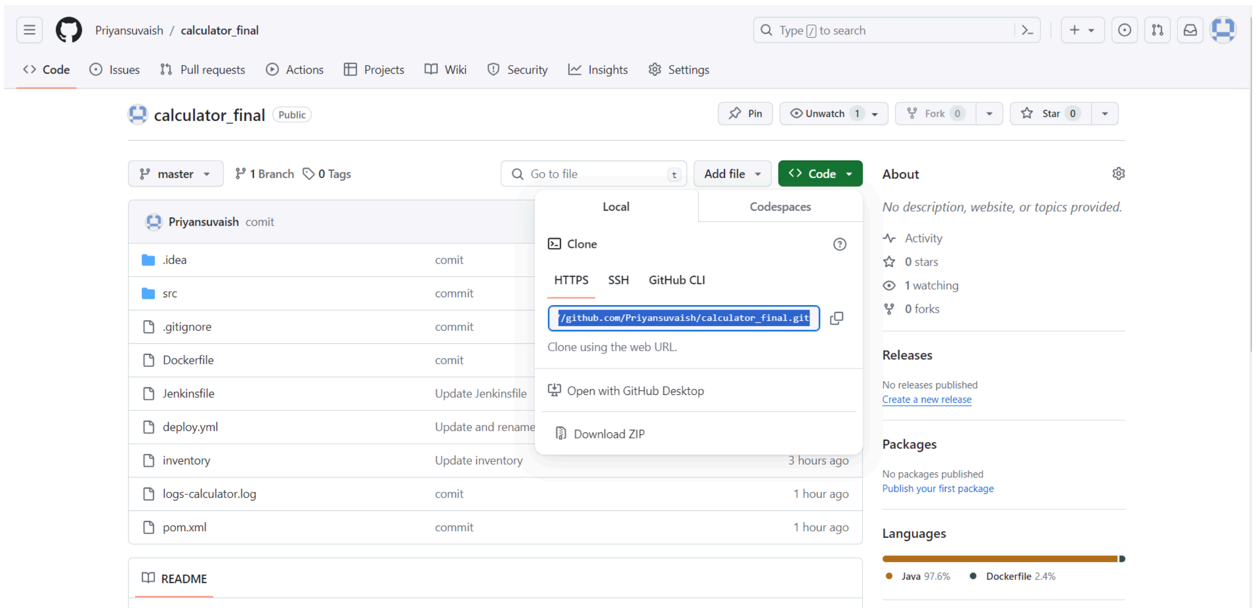


Plugins

git

Name ↓	Enabled
Git client plugin 4.6.0 Utility plugin for Git support in Jenkins Report an issue with this plugin	✓
Git plugin 5.2.1 This plugin integrates Git with Jenkins. Report an issue with this plugin	✓
GitHub 1.37.3.1 This plugin integrates GitHub to Jenkins. Report an issue with this plugin	✓
GitHub API Plugin 1.318-461.v7a_c09c9fa_d63 This plugin provides GitHub API for other plugins. Report an issue with this plugin	✓
GitHub Branch Source Plugin 1772.va_69eda_d018d4 Multibranch projects and organization folders from GitHub. Maintained by CloudBees, Inc. Report an issue with this plugin	✓
Pipeline: GitHub Groovy Libraries 42.v0739460cda_c4 Allows Pipeline Groovy libraries to be loaded on the fly from GitHub. Report an issue with this plugin	✓

Copy the GitHub repository URL:



/github.com/Priyansuvaish/calculator_final.git

Clone using the web URL.

.idea	comit	
src	commit	
.gitignore	commit	
Dockerfile	comit	
Jenkinsfile	Update Jenkinsfile	
deploy.yml	Update and rename	
inventory	Update inventory	3 hours ago
logs-calculator.log	comit	1 hour ago
pom.xml	commit	1 hour ago

No description, website, or topics provided.

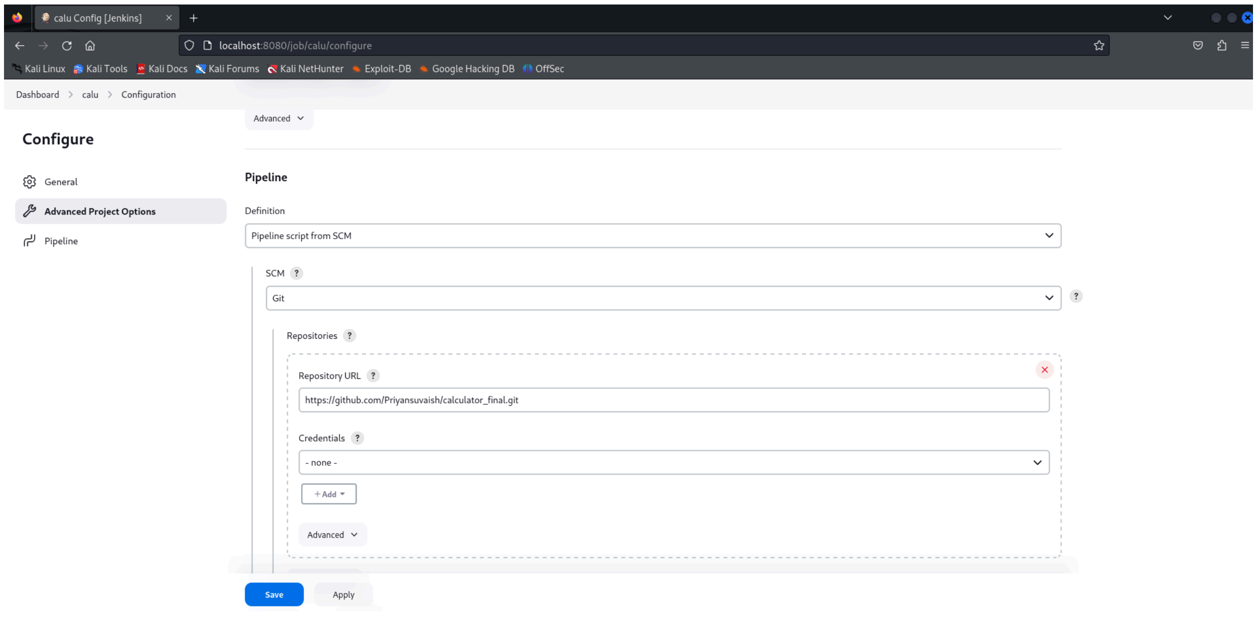
Activity
0 stars
1 watching
0 forks

No releases published
Create a new release

No packages published
Publish your first package

Java 97.6% Dockerfile 2.4%

Paste the URL in the Jenkins project:



Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/Priyansuvaish/calculator_final.git

Credentials: - none -

Building the code with Maven and Testing with JUnit

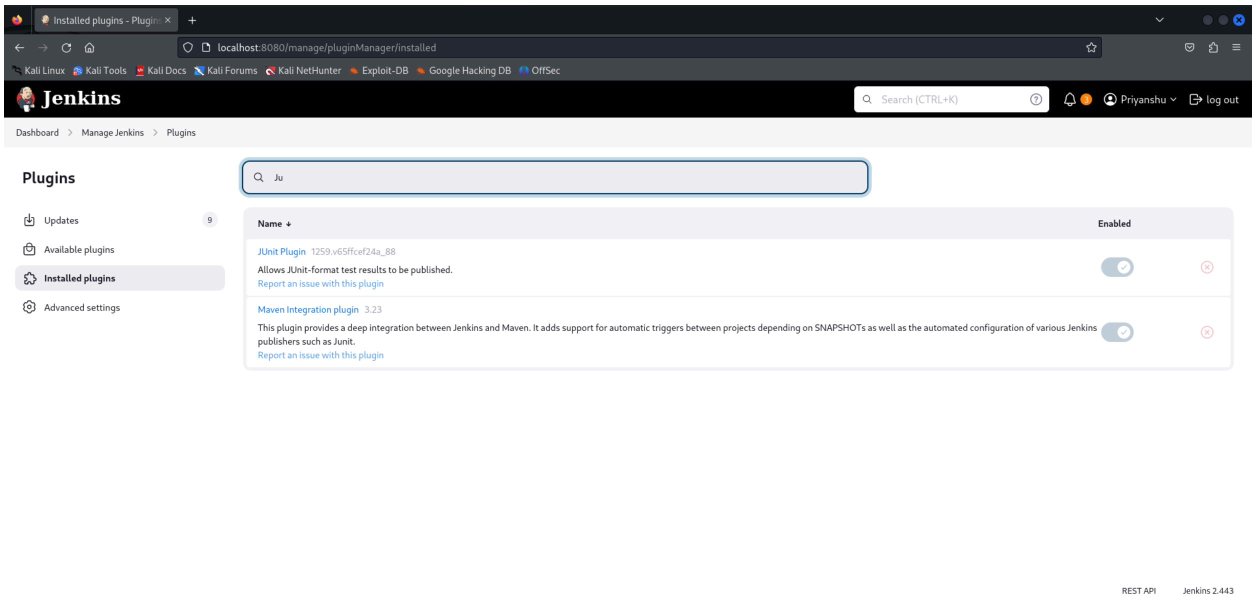
Maven is a powerful build tool for Java projects, simplifying the process of project building, dependency management, and more.

JUnit is a popular framework for testing Java applications, allowing you to write and run repeatable tests.

Setting Up Maven and JUnit

IntelliJ IDEA comes with Maven support out of the box. To configure Maven for a new or existing project, simply open the project in IntelliJ. IntelliJ should automatically recognize the pom.xml file for existing Maven projects and import the necessary configurations.

Download the maven integration plugin, JUnit plugin and Oracle Java SE Development Kit Installer Plugin in Jenkins:

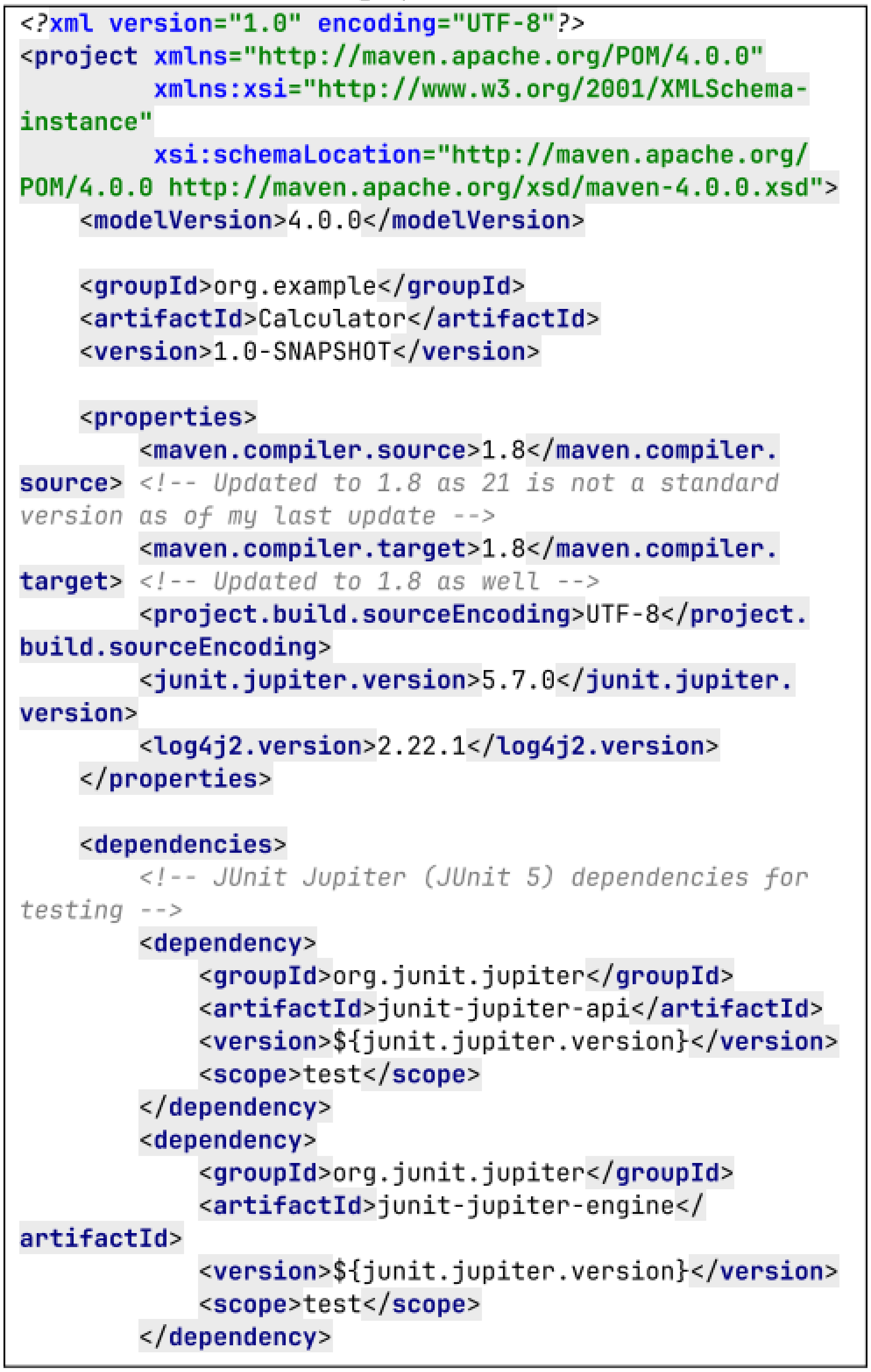


Name ↓	Enabled
JUnit Plugin 1259.v65ffcef24a_88 Allows JUnit-format test results to be published. Report an issue with this plugin	✓
Maven Integration plugin 3.23 This plugin provides a deep integration between Jenkins and Maven. It adds support for automatic triggers between projects depending on SNAPSHOTs as well as the automated configuration of various Jenkins publishers such as Junit. Report an issue with this plugin	✓

Update the pom file for Building and Testing

In the pom.xml file, we define the project's dependencies, plugins, and other configurations. Here, we specify the version of JUnit that we use for testing, among other things.

Screenshot of the Dependencies:



```xml
<?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>Calculator</artifactId>
    <version>1.0-SNAPSHOT</version>

    <properties>
        <maven.compiler.source>1.8</maven.compiler.
source> <!-- Updated to 1.8 as 21 is not a standard
version as of my last update -->
        <maven.compiler.target>1.8</maven.compiler.
target> <!-- Updated to 1.8 as well -->
        <project.build.sourceEncoding>UTF-8</project.
build.sourceEncoding>
        <junit.jupiter.version>5.7.0</junit.jupiter.
version>
        <log4j2.version>2.22.1</log4j2.version>
    </properties>

    <dependencies>
        <!-- JUnit Jupiter (JUnit 5) dependencies for
testing -->
        <dependency>
            <groupId>org.junit.jupiter</groupId>
            <artifactId>junit-jupiter-api</artifactId>
            <version>${junit.jupiter.version}</version>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>org.junit.jupiter</groupId>
            <artifactId>junit-jupiter-engine</
artifactId>
            <version>${junit.jupiter.version}</version>
            <scope>test</scope>
        </dependency>
        <!-- Log4j2 dependencies for logging -->
        <dependency>
            <groupId>org.apache.logging.log4j</groupId>
            <artifactId>log4j-api</artifactId>
            <version>${log4j2.version}</version>
        </dependency>
        <dependency>
            <groupId>org.apache.logging.log4j</groupId>
            <artifactId>log4j-core</artifactId>
            <version>${log4j2.version}</version>
        </dependency>
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>4.13.1</version>
            <scope>test</scope>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <!-- Maven Compiler Plugin to ensure Java 
version compatibility -->
            <plugin>
                <groupId>org.apache.maven.plugins</
groupId>
                <artifactId>maven-compiler-plugin</
artifactId>
                <version>3.8.1</version>
                <configuration>
                    <source>${maven.compiler.source}</
source>
                    <target>${maven.compiler.target}</
target>
                </configuration>
            </plugin>
            <!-- Maven Assembly Plugin for creating a 
jar with dependencies -->
            <plugin>
                <groupId>org.apache.maven.plugins</
groupId>
```

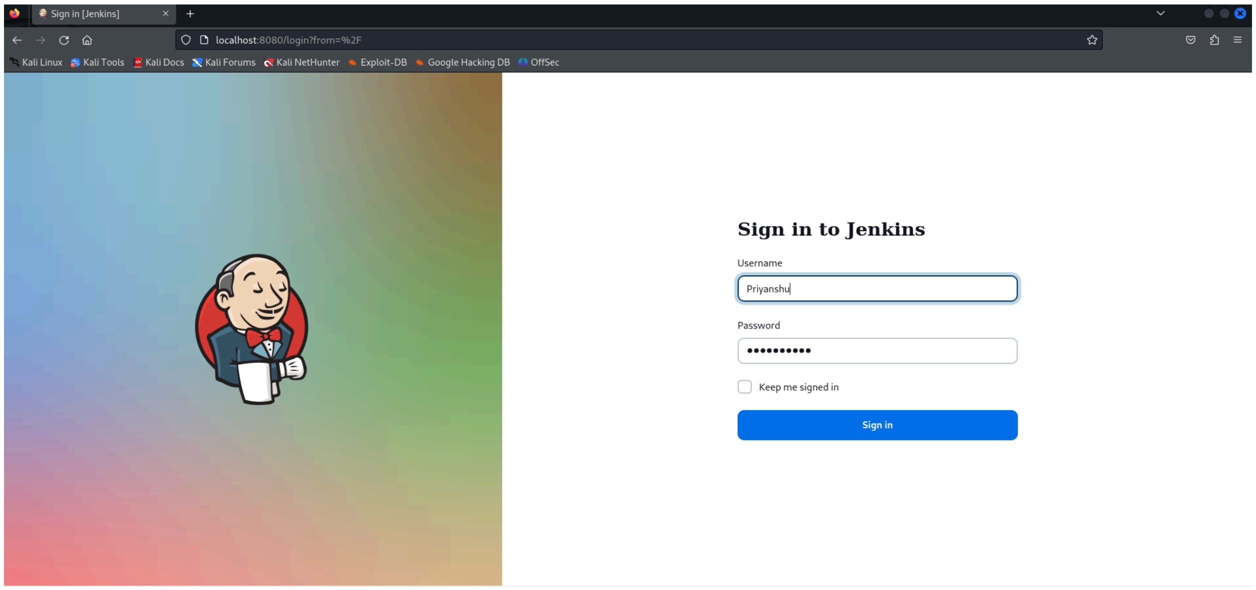
```
                <artifactId>maven-assembly-plugin</
artifactId>
                <version>3.3.0</version> <!-- Update to
 the latest version available -->
                <executions>
                    <execution>
                        <phase>package</phase>
                        <goals>
                            <goal>single</goal>
                        </goals>
                        <configuration>
                            <archive>
                                <manifest>
                                    <mainClass>org.
example.calculator</mainClass>
                                </manifest>
                            </archive>
                            <descriptorRefs>
                                <descriptorRef>jar-with
-dependencies</descriptorRef>
                            </descriptorRefs>
                        </configuration>
                    </execution>
                </executions>
            </plugin>
        </plugins>
    </build>
</project>
```

Continuous Integration/Continuous Deployment (CI/CD): Jenkins

Jenkins automates the parts of software development related to building, testing, and deploying, facilitating continuous integration and continuous delivery.

Setup and Configurations:

- Install Jenkins on the local machine.
- Login the Jenkins using the credentials.



Sign in to Jenkins

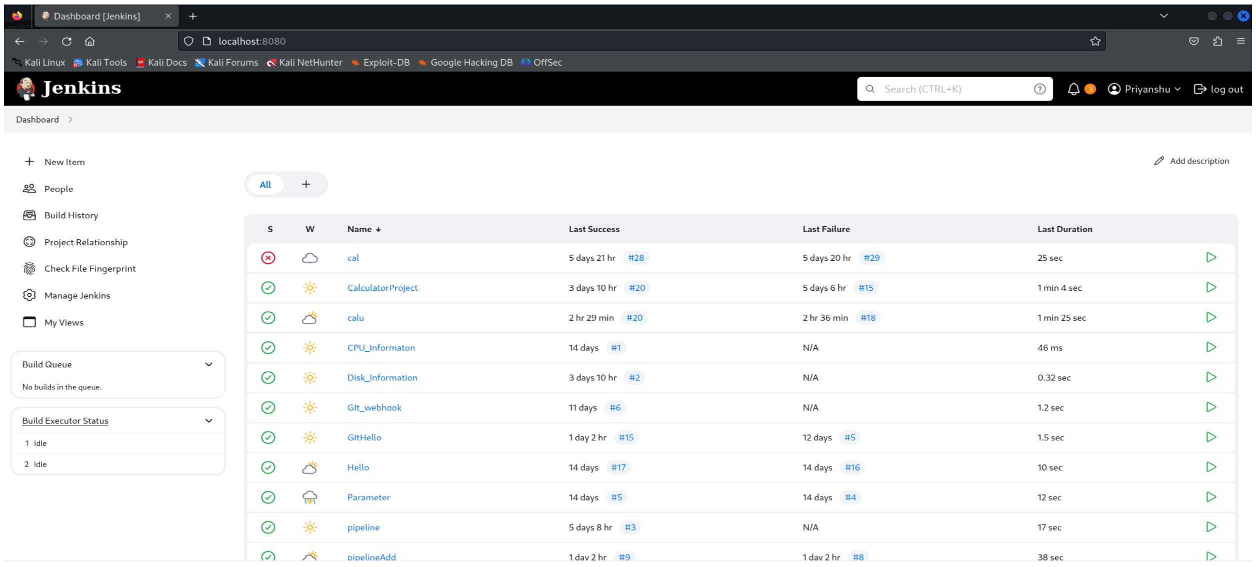
Username: Priyanshu

Password: ••••••••••

Keep me signed in

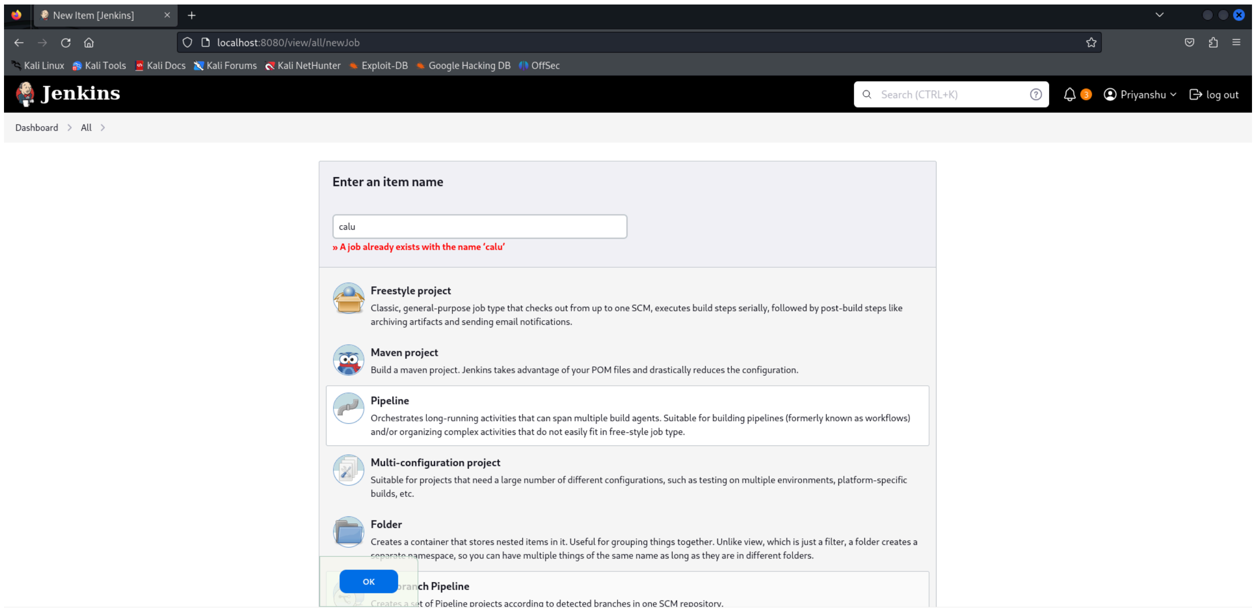
Sign in

- After login, the welcome screen or the previous jobs dashboard comes.



S	W	Name ↓	Last Success	Last Failure	Last Duration
✗	☁	cal	5 days 21 hr #28	5 days 20 hr #29	25 sec
✓	☀	CalculatorProject	3 days 10 hr #20	5 days 6 hr #15	1 min 4 sec
✓	⛅	calu	2 hr 29 min #20	2 hr 36 min #18	1 min 25 sec
✓	☀	CPU_Informaton	14 days #1	N/A	46 ms
✓	☀	Disk_Information	3 days 10 hr #2	N/A	0.32 sec
✓	☀	GIt_webhook	11 days #6	N/A	1.2 sec
✓	☀	GItHello	1 day 2 hr #15	12 days #5	1.5 sec
✓	⛅	Hello	14 days #17	14 days #16	10 sec
✓	⛈	Parameter	14 days #5	14 days #4	12 sec
✓	☀	pipeline	5 days 8 hr #3	N/A	17 sec
✓	⛅	pipelineAdd	1 day 2 hr #9	1 day 2 hr #8	38 sec

- Create the new job as the pipeline.



Enter an item name

calu

» A job already exists with the name 'calu'

Freestyle project
Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

Maven project
Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

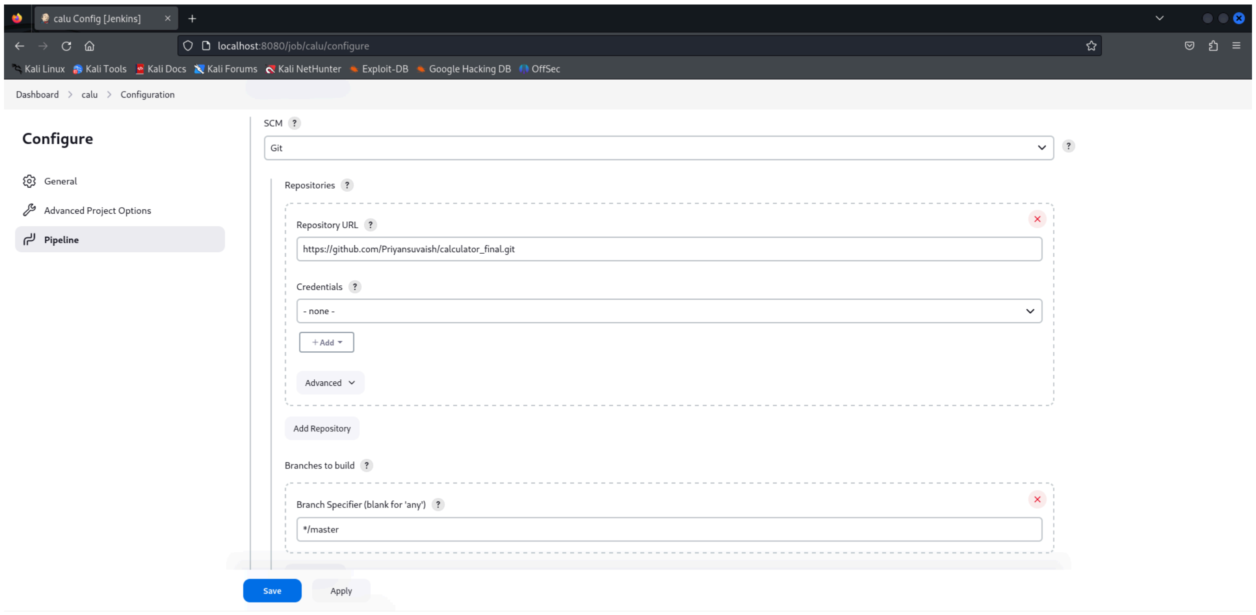
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

- Set up the git SCM and give the branch of the project in GitHub.



SCM: Git

Repositories

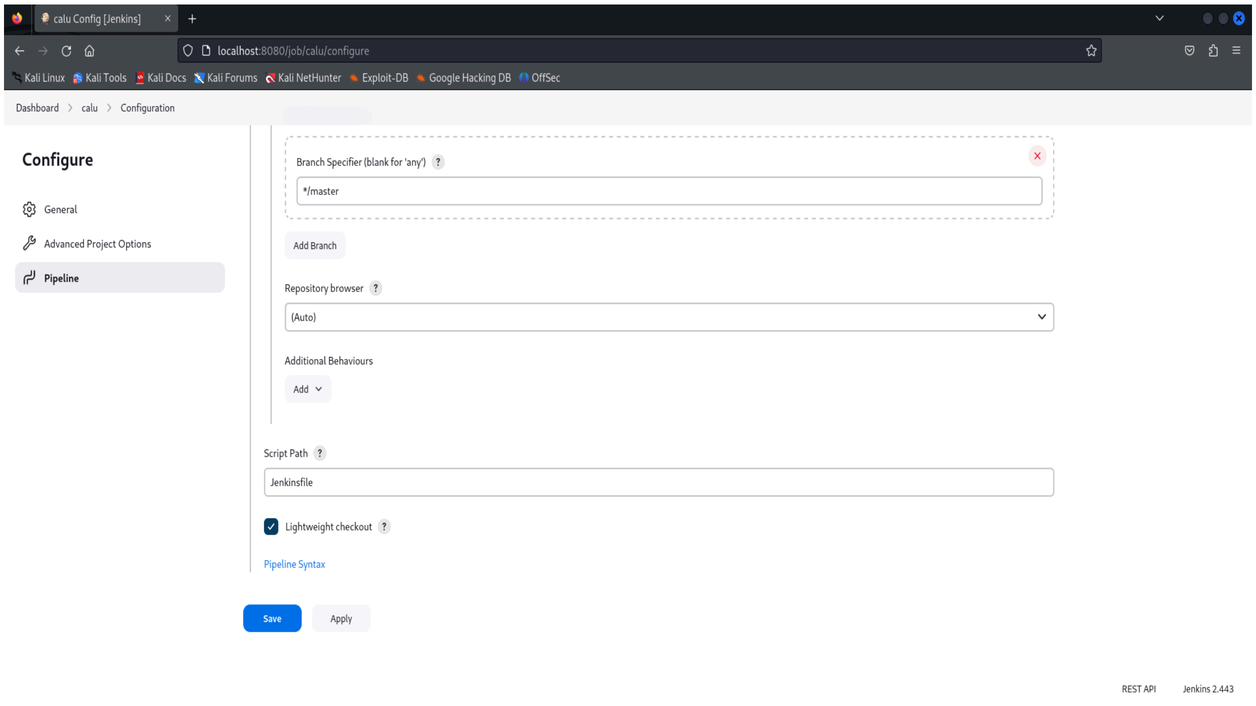
Repository URL: https://github.com/Priyansuvaish/calculator_final.git

Credentials: - none -

Branches to build

Branch Specifier (blank for 'any'): */master

- Give the Jenkinsfile name in the Script Path.



Branch Specifier (blank for 'any'): */master

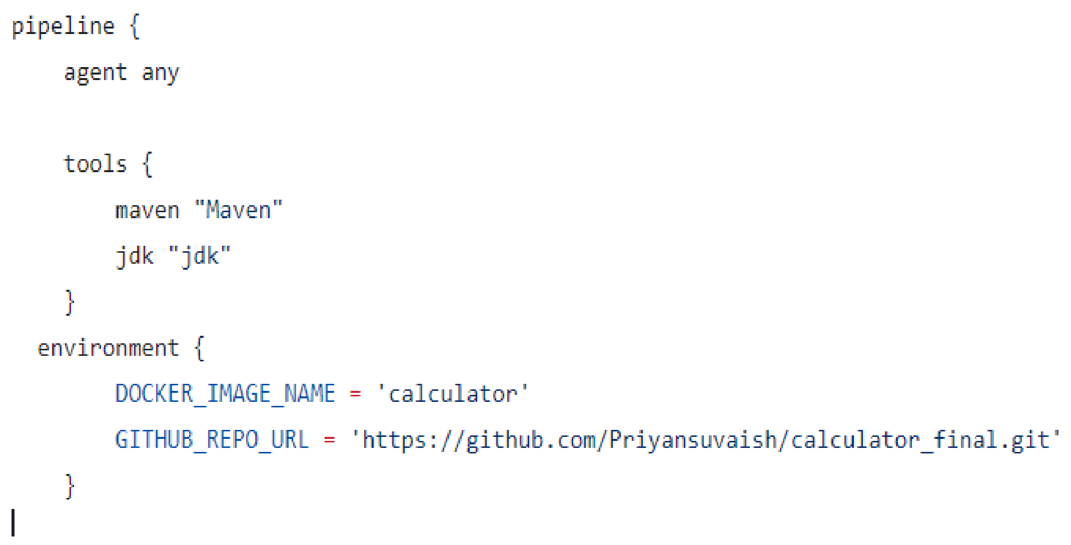
Repository browser: (Auto)

Additional Behaviours

Script Path: Jenkinsfile

☑ Lightweight checkout

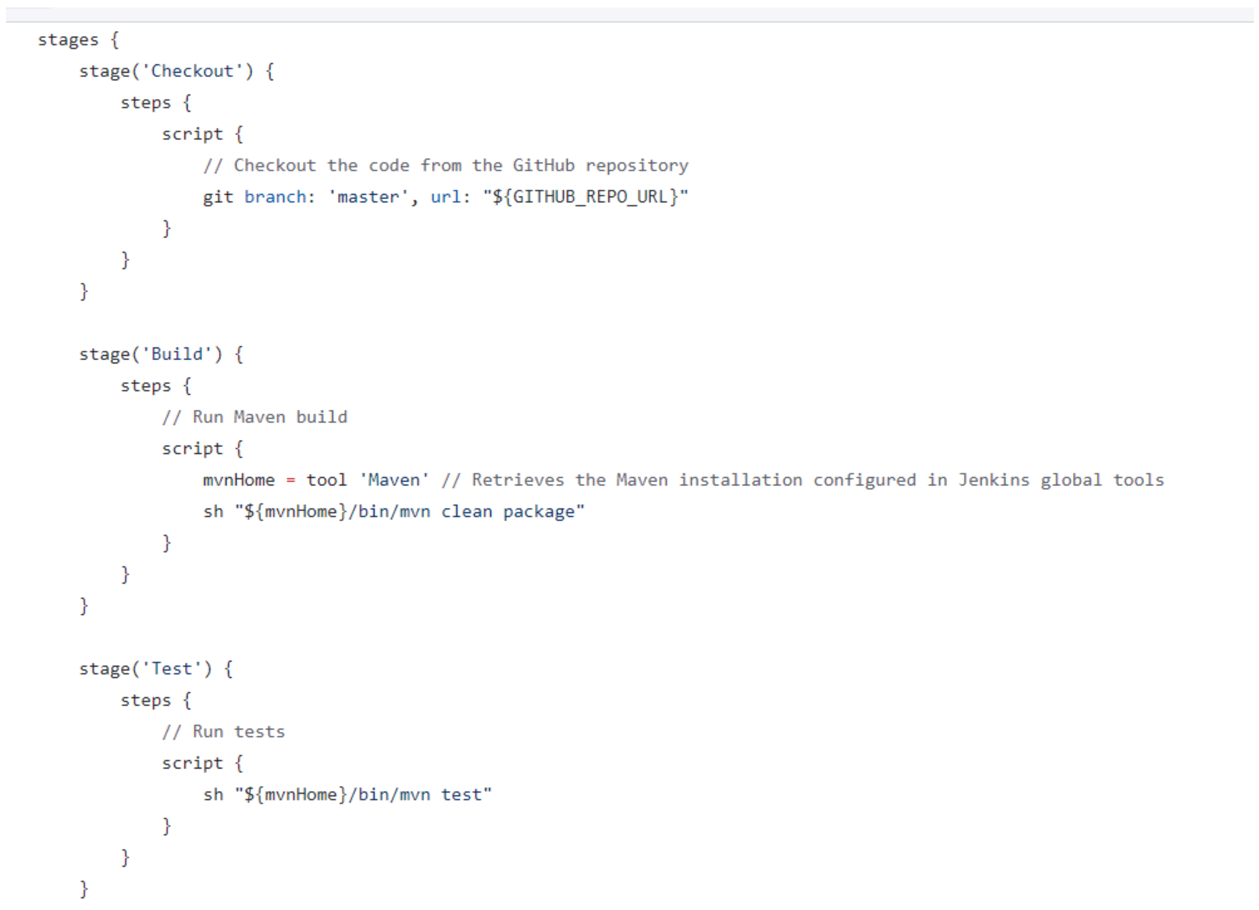
Screenshots of the Jenkins Pipeline Script



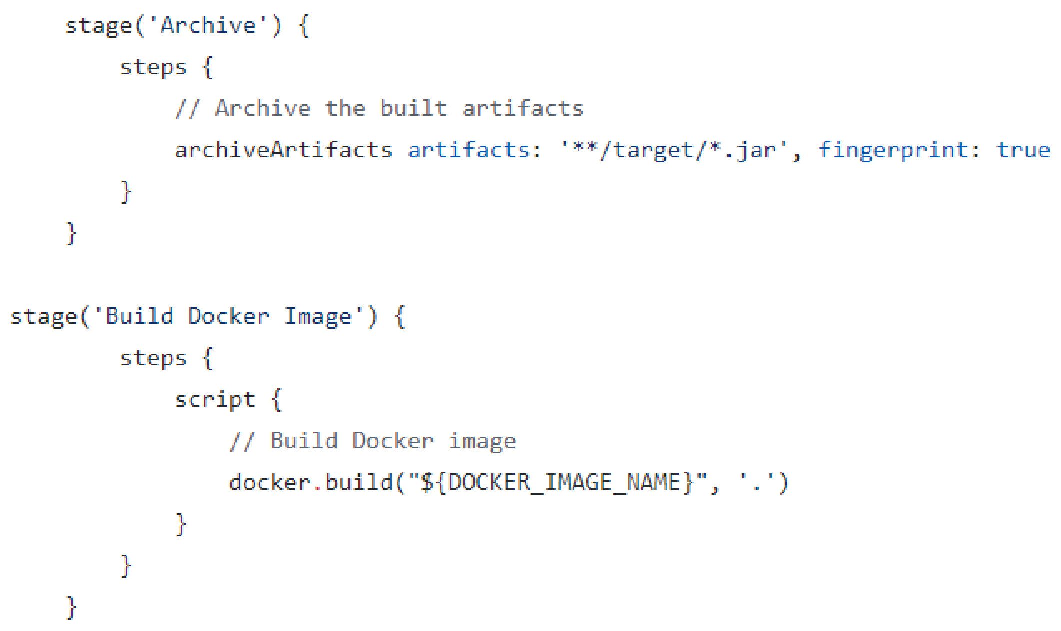
```
pipeline {
    agent any

    tools {
        maven "Maven"
        jdk "jdk"
    }
  environment {
        DOCKER_IMAGE_NAME = 'calculator'
        GITHUB_REPO_URL = 'https://github.com/Priyansuvaish/calculator_final.git'
    }
|
```

In this, we set up the tools and environment variables that we need in different stages.

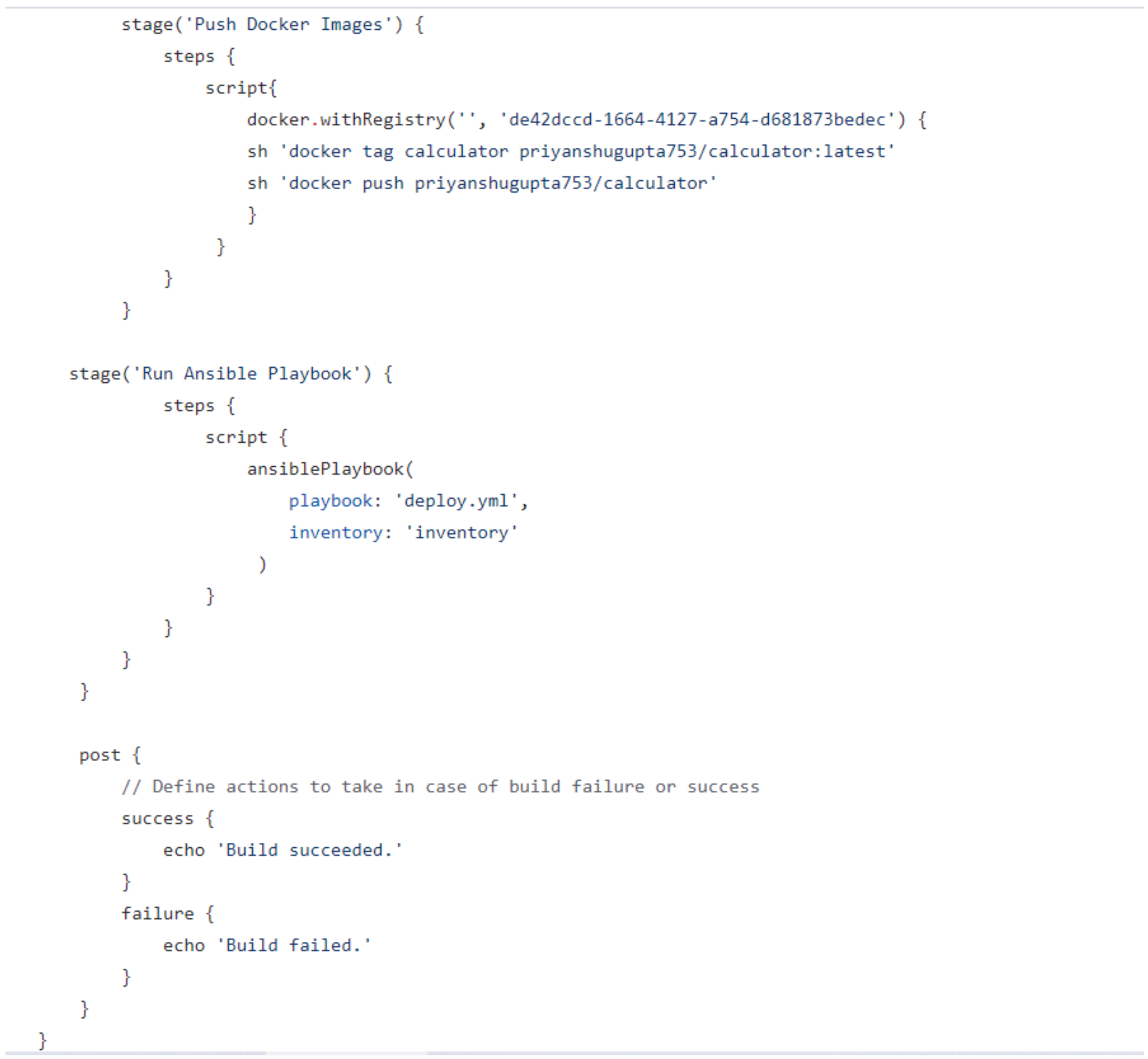
Now the stages of the Jenkins file are as follows:



```
stages {
    stage('Checkout') {
        steps {
            script {
                // Checkout the code from the GitHub repository
                git branch: 'master', url: "${GITHUB_REPO_URL}"
            }
        }
    }

    stage('Build') {
        steps {
            // Run Maven build
            script {
                mvnHome = tool 'Maven' // Retrieves the Maven installation configured in Jenkins global tools
                sh "${mvnHome}/bin/mvn clean package"
            }
        }
    }

    stage('Test') {
        steps {
            // Run tests
            script {
                sh "${mvnHome}/bin/mvn test"
            }
        }
    }
```



```
        stage('Archive') {
            steps {
                // Archive the built artifacts
                archiveArtifacts artifacts: '**/target/*.jar', fingerprint: true
            }
        }

    stage('Build Docker Image') {
            steps {
                script {
                    // Build Docker image
                    docker.build("${DOCKER_IMAGE_NAME}", '.')
                }
            }
        }
```



```
        stage('Push Docker Images') {
            steps {
                script{
                    docker.withRegistry('', 'de42dccd-1664-4127-a754-d681873bedec') {
                    sh 'docker tag calculator priyanshugupta753/calculator:latest'
                    sh 'docker push priyanshugupta753/calculator'
                    }
                 }
            }
        }

   stage('Run Ansible Playbook') {
            steps {
                script {
                    ansiblePlaybook(
                        playbook: 'deploy.yml',
                        inventory: 'inventory'
                     )
                }
            }
        }
    }

    post {
        // Define actions to take in case of build failure or success
        success {
            echo 'Build succeeded.'
        }
        failure {
            echo 'Build failed.'
        }
    }
}
```

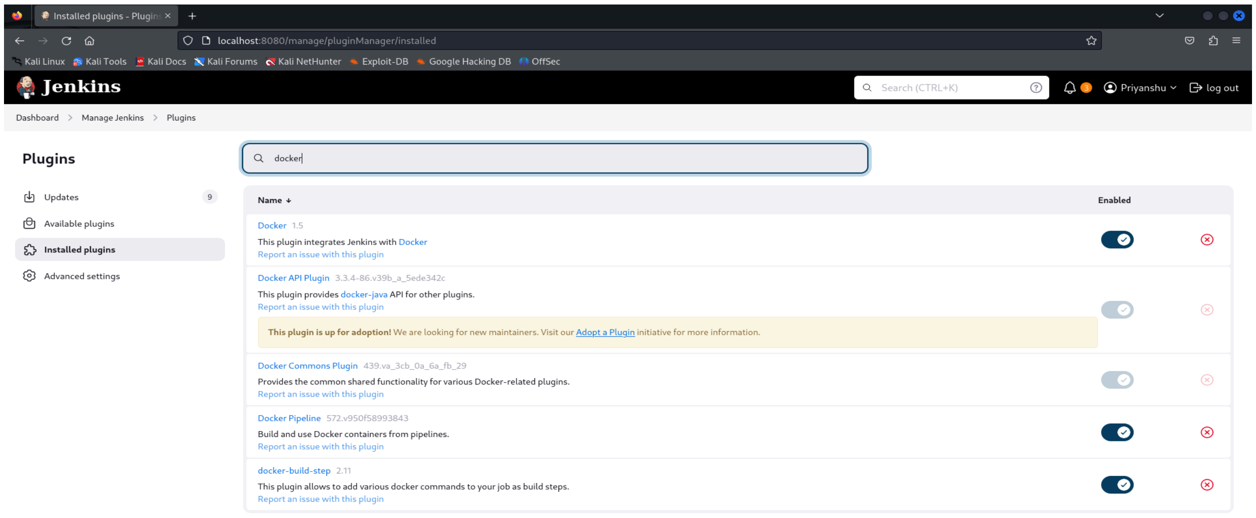
Containerization with Docker

Docker is a set of platform-as-a-service products that use OS-level virtualization to deliver software in packages called containers.

Setup and Configurations:

- Install Docker on the local machine.

- Install the plugins in Jenkins

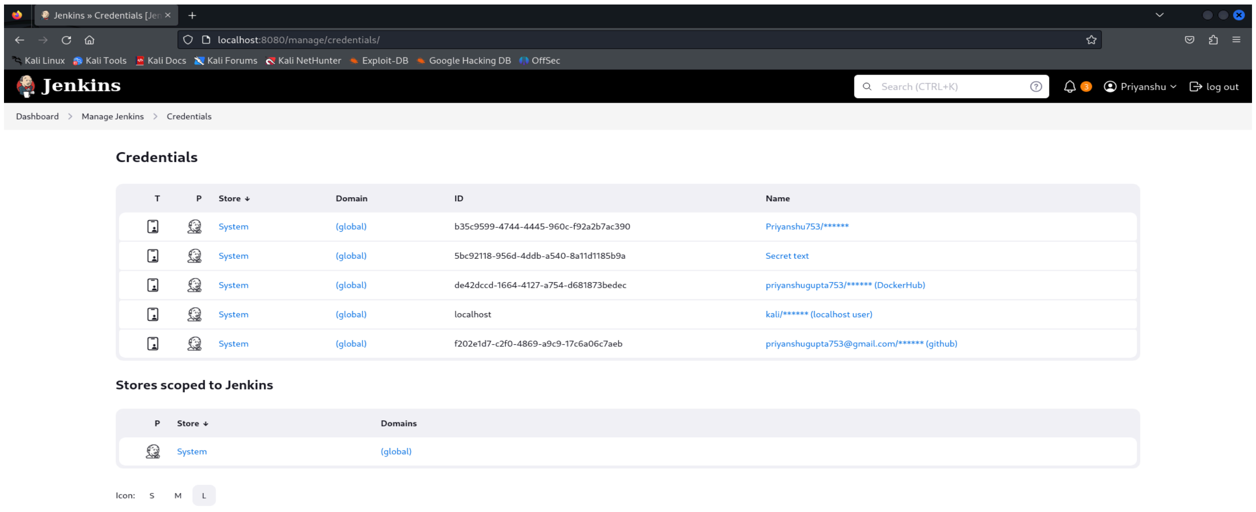


Dashboard > Manage Jenkins > Plugins

Plugins

Name ↓	Enabled
Docker 1.5 This plugin integrates Jenkins with Docker Report an issue with this plugin	✓
Docker API Plugin 3.3.4-86.v39b_a_5ede342c This plugin provides docker-java API for other plugins. Report an issue with this plugin **This plugin is up for adoption!** We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.	✓
Docker Commons Plugin 439.va_3cb_0a_6a_fb_29 Provides the common shared functionality for various Docker-related plugins. Report an issue with this plugin	✓
Docker Pipeline 572.v950f58993843 Build and use Docker containers from pipelines. Report an issue with this plugin	✓
docker-build-step 2.11 This plugin allows to add various docker commands to your job as build steps. Report an issue with this plugin	✓

- In the credential give the docker username and password.



Dashboard > Manage Jenkins > Credentials

Credentials

T	P	Store ↓	Domain	ID	Name
		System	(global)	b35c9599-4744-4445-960c-f92a2b7ac390	Priyanshu753/******
		System	(global)	5bc92118-956d-4ddb-a540-8a11d1185b9a	Secret text
		System	(global)	de42dccd-1664-4127-a754-d681873bedec	priyanshugupta753/****** (DockerHub)
		System	(global)	localhost	kali/****** (localhost user)
		System	(global)	f202e1d7-c2f0-4869-a9c9-17c6a06c7aeb	priyanshugupta753@gmail.com/****** (github)

Stores scoped to Jenkins

P	Store ↓	Domains
	System	(global)

Icon: S M L

- Copy the ID that you give in the credential and put it in the Jenkins file in the command:

```
stage('Push Docker Images') {
    steps {
        script{
            docker.withRegistry('', 'de42dccd-1664-4127-a754-d681873bedec') {
            sh 'docker tag calculator priyanshugupta753/calculator:latest'
            sh 'docker push priyanshugupta753/calculator'
            }
         }
    }
}
```

ScreenShot of Docker file script

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

Deployment using Ansible

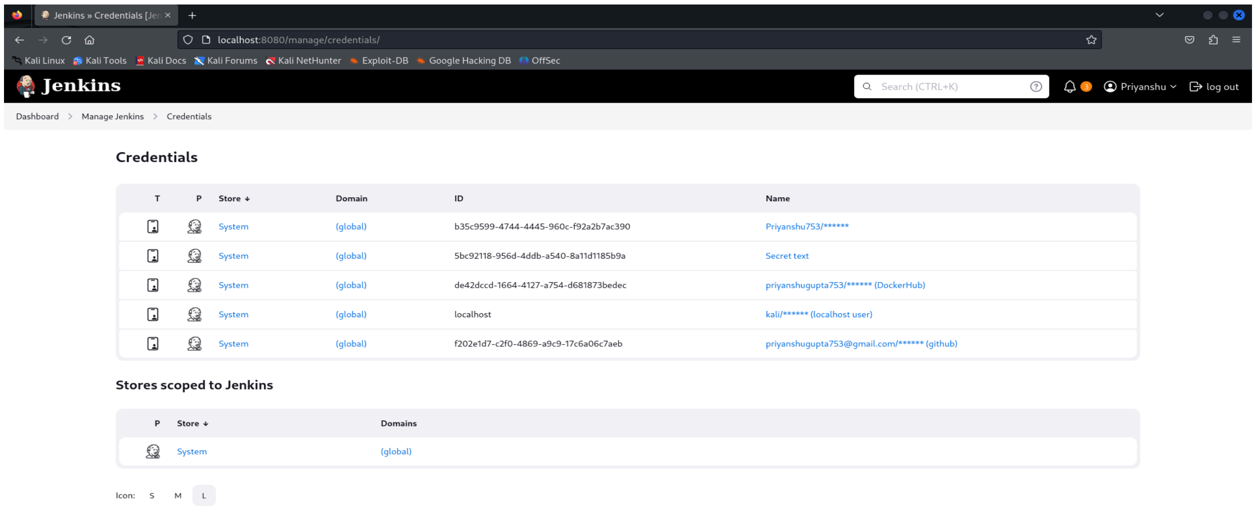
Ansible is an open-source software provisioning, configuration management, and application-deployment tool enabling infrastructure as code.

Setup and Configurations:

- Install Ansible on the local machine.
- Configure inventory file is as follows:

```
ansible_host=localhost ansible_user=kali ansible_ssh_pass=kali
# ansible_ssh_common_args = '-o StrictHostKeyChecking=no'
```

- In the credential give the local machine username and password.



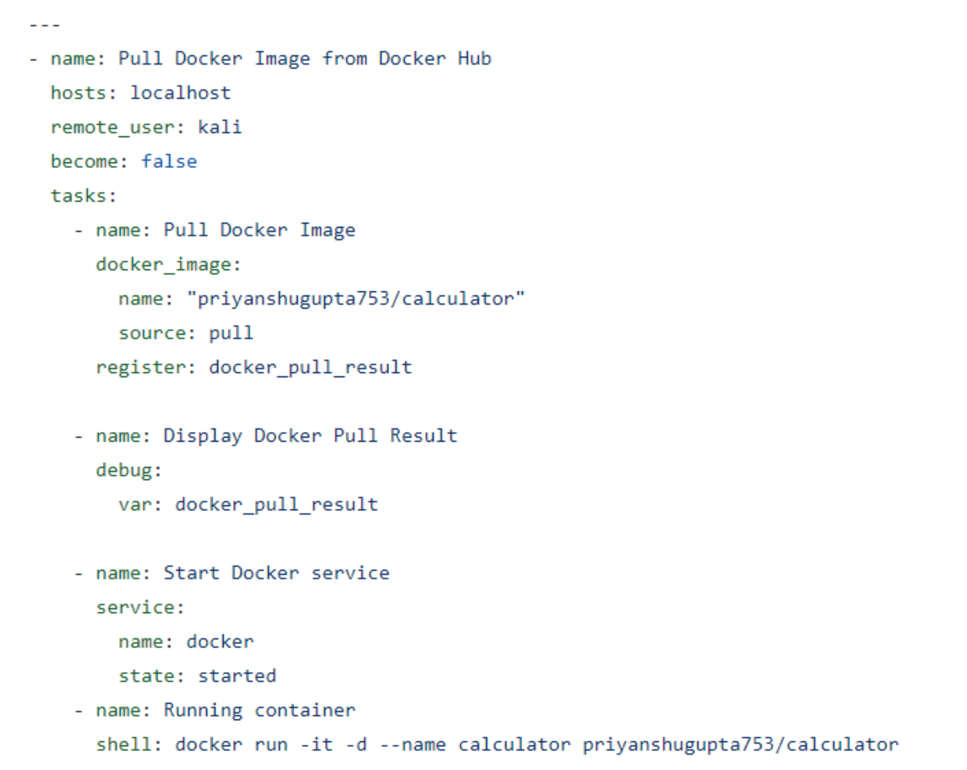
Credentials

T	P	Store ↓	Domain	ID	Name
		System	(global)	b35c9599-4744-4445-960c-f92a2b7ac390	Priyanshu753/******
		System	(global)	5bc92118-956d-4ddb-a540-8a11d1185b9a	Secret text
		System	(global)	de42dccd-1664-4127-a754-d681873bedec	priyanshugupta753/****** (DockerHub)
		System	(global)	localhost	kali/****** (localhost user)
		System	(global)	f202e1d7-c2f0-4869-a9c9-17c6a06c7aeb	priyanshugupta753@gmail.com/****** (github)

Stores scoped to Jenkins

P	Store ↓	Domains
	System	(global)

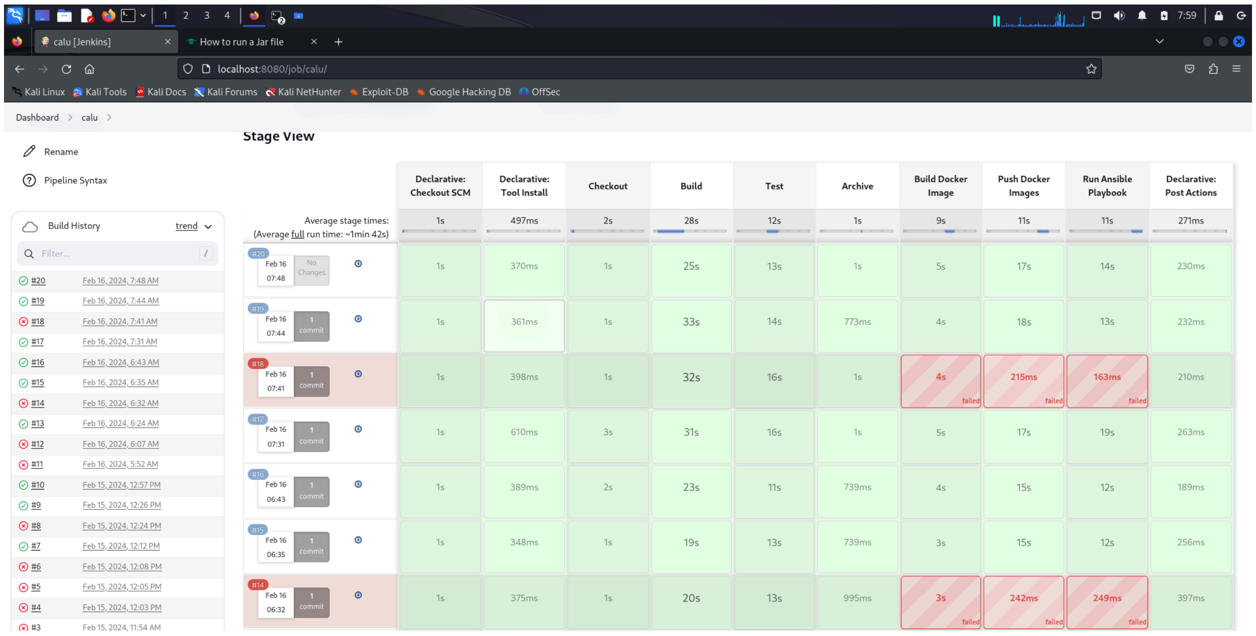
ScreenShot of the playbook script



```yaml
---
- name: Pull Docker Image from Docker Hub
  hosts: localhost
  remote_user: kali
  become: false
  tasks:
    - name: Pull Docker Image
      docker_image:
        name: "priyanshugupta753/calculator"
        source: pull
      register: docker_pull_result

    - name: Display Docker Pull Result
      debug:
        var: docker_pull_result

    - name: Start Docker service
      service:
        name: docker
        state: started
    - name: Running container
      shell: docker run -it -d --name calculator priyanshugupta753/calculator
```

Working of the application after the successfully deployed:

- All the stages are built successfully



Dashboard > calu >

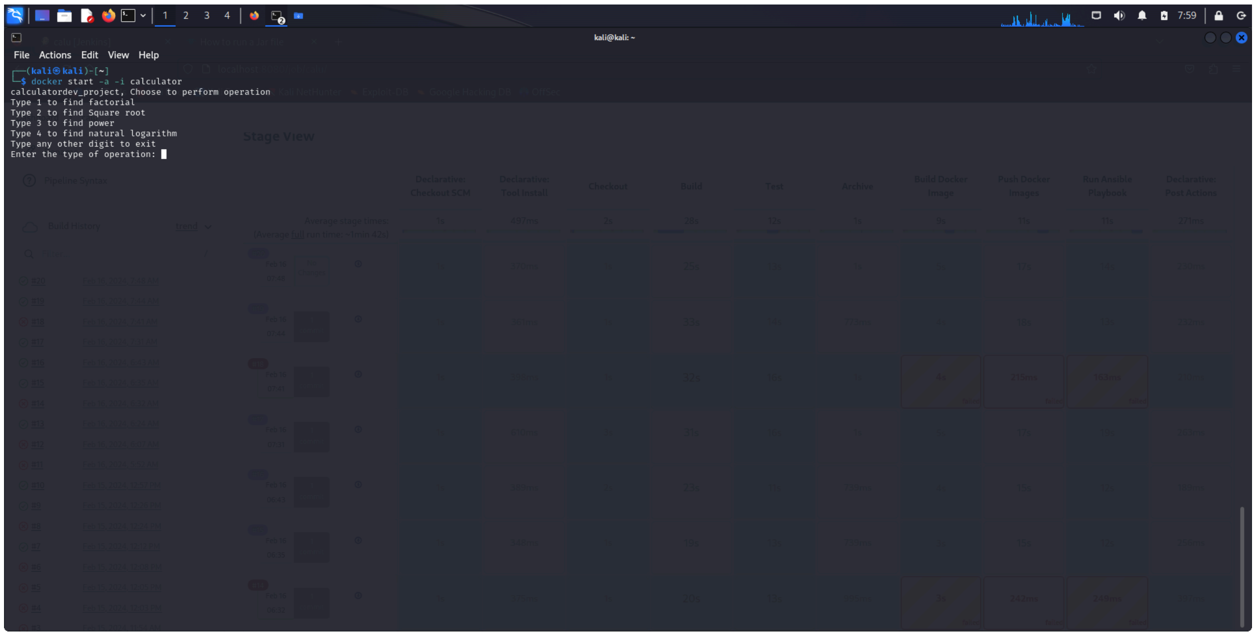
Stage View

Build History

✓ #20	Feb 16, 2024, 7:48 AM
✓ #19	Feb 16, 2024, 7:44 AM
✗ #18	Feb 16, 2024, 7:41 AM
✓ #17	Feb 16, 2024, 7:31 AM
✓ #16	Feb 16, 2024, 6:43 AM
✓ #15	Feb 16, 2024, 6:35 AM
✗ #14	Feb 16, 2024, 6:32 AM
✓ #13	Feb 16, 2024, 6:24 AM
✗ #12	Feb 16, 2024, 6:07 AM
✗ #11	Feb 16, 2024, 5:52 AM
✓ #10	Feb 15, 2024, 12:57 PM
✓ #9	Feb 15, 2024, 12:26 PM
✗ #8	Feb 15, 2024, 12:24 PM
✓ #7	Feb 15, 2024, 12:12 PM
✗ #6	Feb 15, 2024, 12:08 PM
✗ #5	Feb 15, 2024, 12:05 PM
✗ #4	Feb 15, 2024, 12:03 PM
✗ #3	Feb 15, 2024, 11:54 AM

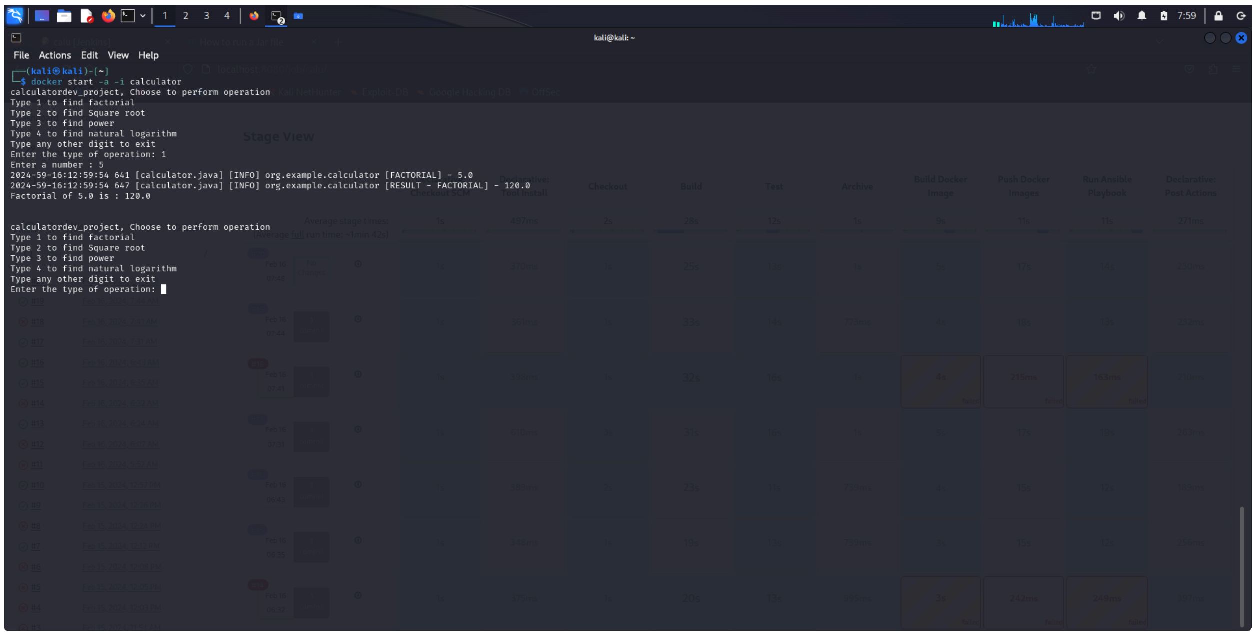
	Declarative: Checkout SCM	Declarative: Tool Install	Checkout	Build	Test	Archive	Build Docker Image	Push Docker Images	Run Ansible Playbook	Declarative: Post Actions
Average stage times: (Average full run time: ~1min 42s)	1s	497ms	2s	28s	12s	1s	9s	11s	11s	271ms
#20 Feb 16 07:48 No Changes	1s	370ms	1s	25s	13s	1s	5s	17s	14s	230ms
#19 Feb 16 07:44 1 commit	1s	361ms	1s	33s	14s	773ms	4s	18s	13s	232ms
#18 Feb 16 07:41 1 commit	1s	398ms	1s	32s	16s	1s	4s failed	215ms failed	163ms failed	210ms
#17 Feb 16 07:31 1 commit	1s	610ms	3s	31s	16s	1s	5s	17s	19s	263ms
#16 Feb 16 06:43 1 commit	1s	389ms	2s	23s	11s	739ms	4s	15s	12s	189ms
#15 Feb 16 06:35 1 commit	1s	348ms	1s	19s	13s	739ms	3s	15s	12s	256ms
#14 Feb 16 06:32 1 commit	1s	375ms	1s	20s	13s	995ms	3s failed	242ms failed	249ms failed	397ms

- Running the docker container:



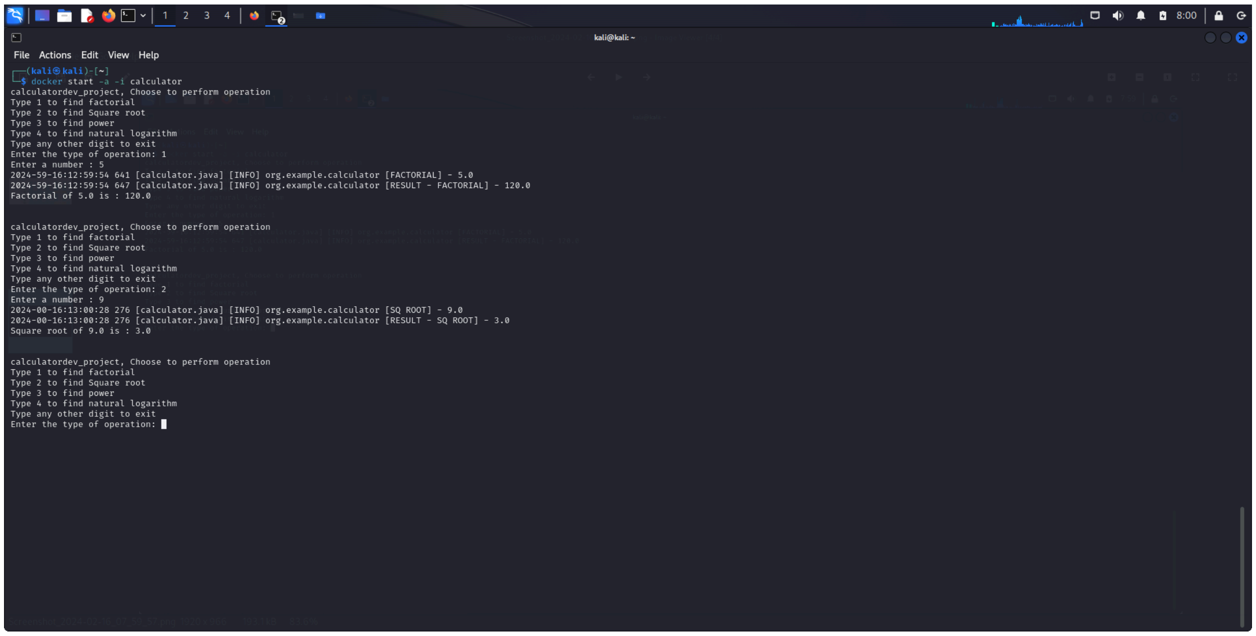
```
┌──(kali㉿kali)-[~]
└─$ docker start -a -i calculator
calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 
```

- Finding the factorial of the number



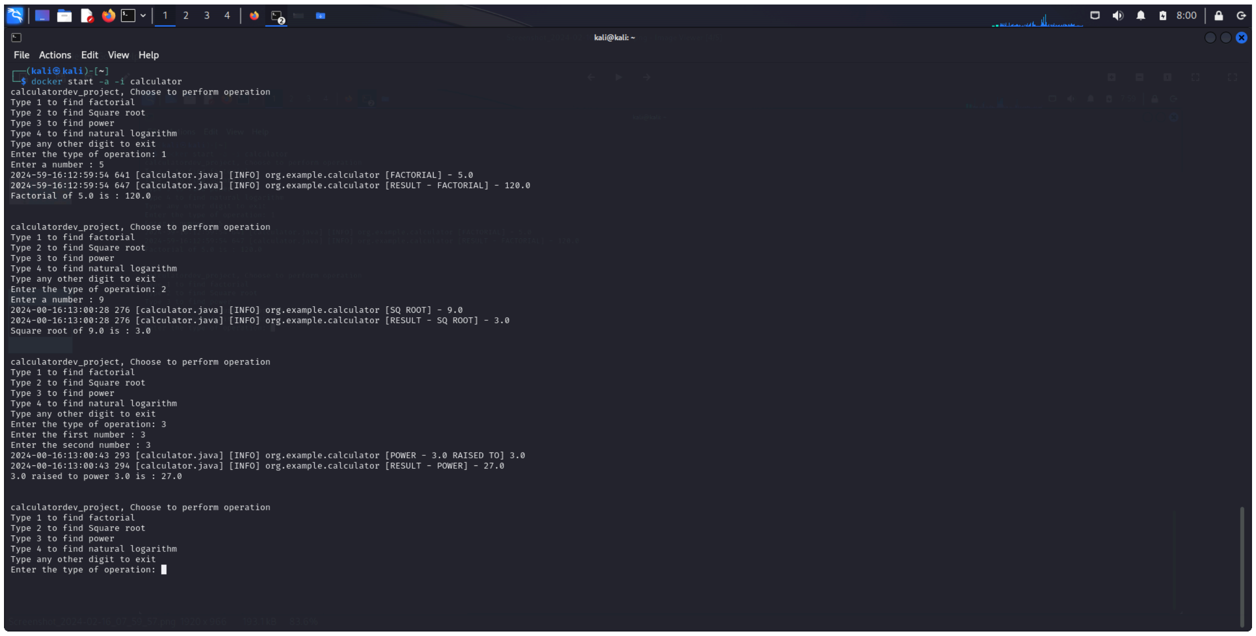
```
┌──(kali㉿kali)-[~]
└─$ docker start -a -i calculator
calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 1
Enter a number : 5
2024-59-16:12:59:54 641 [calculator.java] [INFO] org.example.calculator [FACTORIAL] - 5.0
2024-59-16:12:59:54 647 [calculator.java] [INFO] org.example.calculator [RESULT - FACTORIAL] - 120.0
Factorial of 5.0 is : 120.0


calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 
```

- Finding the square root of the number



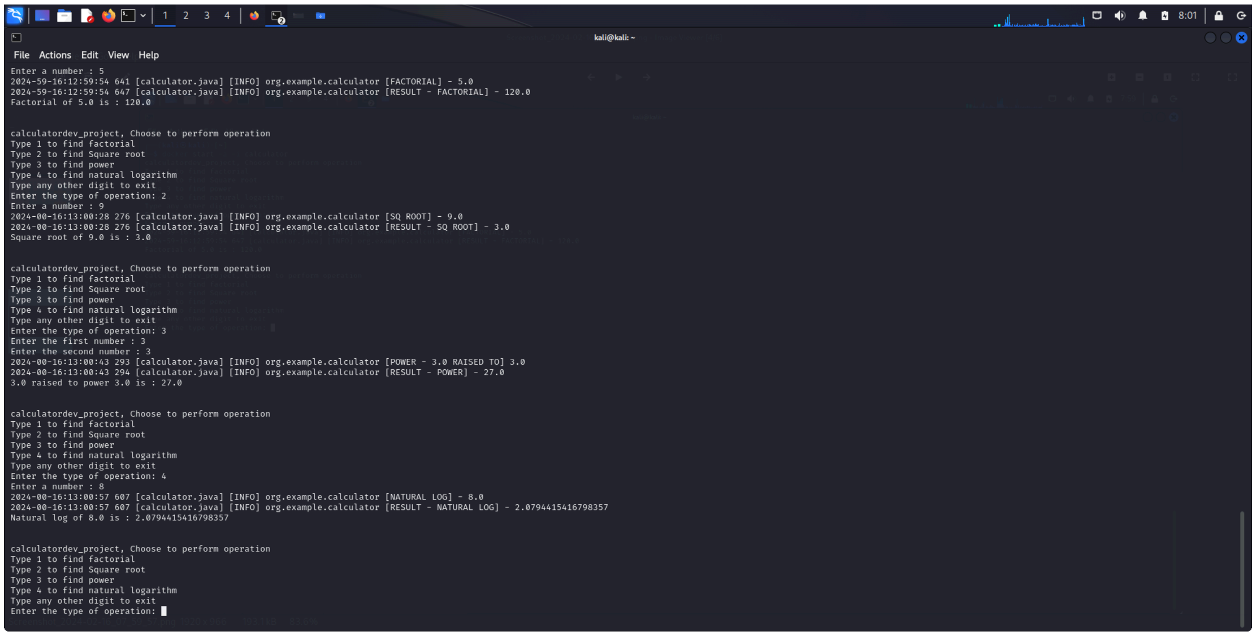
```
┌──(kali㉿kali)-[~]
└─$ docker start -a -i calculator
calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 1
Enter a number : 5
2024-59-16:12:59:54 641 [calculator.java] [INFO] org.example.calculator [FACTORIAL] - 5.0
2024-59-16:12:59:54 647 [calculator.java] [INFO] org.example.calculator [RESULT - FACTORIAL] - 120.0
Factorial of 5.0 is : 120.0


calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 2
Enter a number : 9
2024-00-16:13:00:28 276 [calculator.java] [INFO] org.example.calculator [SQ ROOT] - 9.0
2024-00-16:13:00:28 276 [calculator.java] [INFO] org.example.calculator [RESULT - SQ ROOT] - 3.0
Square root of 9.0 is : 3.0


calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 
```

- Finding the power of the number



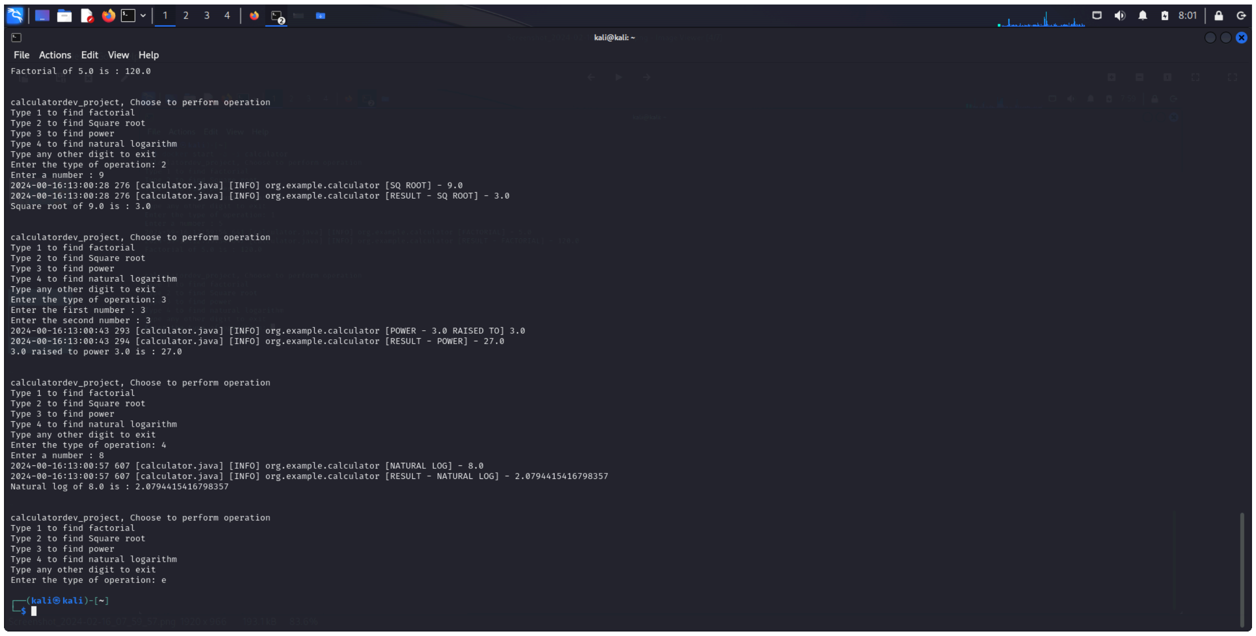
```
┌──(kali㉿kali)-[~]
└─$ docker start -a -i calculator
calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 1
Enter a number : 5
2024-59-16:12:59:54 641 [calculator.java] [INFO] org.example.calculator [FACTORIAL] - 5.0
2024-59-16:12:59:54 647 [calculator.java] [INFO] org.example.calculator [RESULT - FACTORIAL] - 120.0
Factorial of 5.0 is : 120.0


calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 2
Enter a number : 9
2024-00-16:13:00:28 276 [calculator.java] [INFO] org.example.calculator [SQ ROOT] - 9.0
2024-00-16:13:00:28 276 [calculator.java] [INFO] org.example.calculator [RESULT - SQ ROOT] - 3.0
Square root of 9.0 is : 3.0


calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 3
Enter the first number : 3
Enter the second number : 3
2024-00-16:13:00:43 293 [calculator.java] [INFO] org.example.calculator [POWER - 3.0 RAISED TO] 3.0
2024-00-16:13:00:43 294 [calculator.java] [INFO] org.example.calculator [RESULT - POWER] - 27.0
3.0 raised to power 3.0 is : 27.0


calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 
```

- Finding the natural log of the number



```
Enter a number : 5
2024-59-16:12:59:54 641 [calculator.java] [INFO] org.example.calculator [FACTORIAL] - 5.0
2024-59-16:12:59:54 647 [calculator.java] [INFO] org.example.calculator [RESULT - FACTORIAL] - 120.0
Factorial of 5.0 is : 120.0


calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 2
Enter a number : 9
2024-00-16:13:00:28 276 [calculator.java] [INFO] org.example.calculator [SQ ROOT] - 9.0
2024-00-16:13:00:28 276 [calculator.java] [INFO] org.example.calculator [RESULT - SQ ROOT] - 3.0
Square root of 9.0 is : 3.0


calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 3
Enter the first number : 3
Enter the second number : 3
2024-00-16:13:00:43 293 [calculator.java] [INFO] org.example.calculator [POWER - 3.0 RAISED TO] 3.0
2024-00-16:13:00:43 294 [calculator.java] [INFO] org.example.calculator [RESULT - POWER] - 27.0
3.0 raised to power 3.0 is : 27.0


calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 4
Enter a number : 8
2024-00-16:13:00:57 607 [calculator.java] [INFO] org.example.calculator [NATURAL LOG] - 8.0
2024-00-16:13:00:57 607 [calculator.java] [INFO] org.example.calculator [RESULT - NATURAL LOG] - 2.0794415416798357
Natural log of 8.0 is : 2.0794415416798357


calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 
```

- Type any other digit to exit.



```
Factorial of 5.0 is : 120.0


calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 2
Enter a number : 9
2024-00-16:13:00:28 276 [calculator.java] [INFO] org.example.calculator [SQ ROOT] - 9.0
2024-00-16:13:00:28 276 [calculator.java] [INFO] org.example.calculator [RESULT - SQ ROOT] - 3.0
Square root of 9.0 is : 3.0


calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 3
Enter the first number : 3
Enter the second number : 3
2024-00-16:13:00:43 293 [calculator.java] [INFO] org.example.calculator [POWER - 3.0 RAISED TO] 3.0
2024-00-16:13:00:43 294 [calculator.java] [INFO] org.example.calculator [RESULT - POWER] - 27.0
3.0 raised to power 3.0 is : 27.0


calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: 4
Enter a number : 8
2024-00-16:13:00:57 607 [calculator.java] [INFO] org.example.calculator [NATURAL LOG] - 8.0
2024-00-16:13:00:57 607 [calculator.java] [INFO] org.example.calculator [RESULT - NATURAL LOG] - 2.0794415416798357
Natural log of 8.0 is : 2.0794415416798357


calculatordev_project, Choose to perform operation
Type 1 to find factorial
Type 2 to find Square root
Type 3 to find power
Type 4 to find natural logarithm
Type any other digit to exit
Enter the type of operation: e

┌──(kali㉿kali)-[~]
└─$ 
```

Thank you