MINI PROJECT

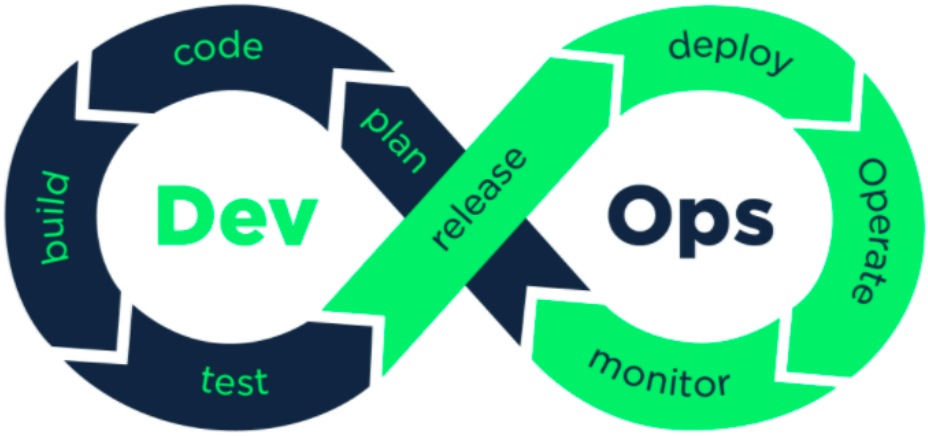
SCIENTIFIC CALCULATOR

Building and Deploying using DevOps tools

DEBMALYA SEN
MT2021036

Github Profile
https://github.com/DEBMALYASEN/mini

DockerHub Profile
https://hub.docker.com/u/bony211298



Project Statement

Creating a Scientific Calculator program with operations
such as :

Square root function - √x
Factorial function - x!
Natural logarithm (base e) - ln(x)
Power function - (x,y)

Adding each functionality incrementally using devops tools like git,jenkins,docker, ansible and ELK stack.

DevOps Pipeline

The calculator application is a small project developed to better understand source control management, continuous integration and deployment.It includes creating a small calculator that has functions like square root, natural log, power and factorial.
Includes pushing the latest changes to git (GitHub), building code using maven, pushing build docker image onto docker hub. This entire process is called continuous integration.
Later we will integrate Ansible with Jenkins which will support continuous deployment.
Ansible will pull the image from docker hub to managed nodes and will deploy it to the docker container.

The tools used: -

1. Software Configuration Management - GitHub
2. Testing - Junit
3. Build - Maven
4. Containerize - Docker
5. Continuous Integration – Jenkins
6. Continuous Deployment – Ansible
7. Continuous Monitoring - ELK

Detailed description of all the steps used during development of the entire calculator application:

1. Software Configuration Management

The main role of SCM is to maintain the current state of the software (called the “baseline”), while enabling developers to work on new versions for features or fixes. This has been implemented through GitHub. Initially a new repository has to be created on https://www.github.com . We can create a new repository by mentioning a unique name signed in to the user. The code for the calculator application can be found at mini.
The SCM handles our code and will be used to connect as input to jenkins. Various other SCM are available like GitLab, BitBucket etc.

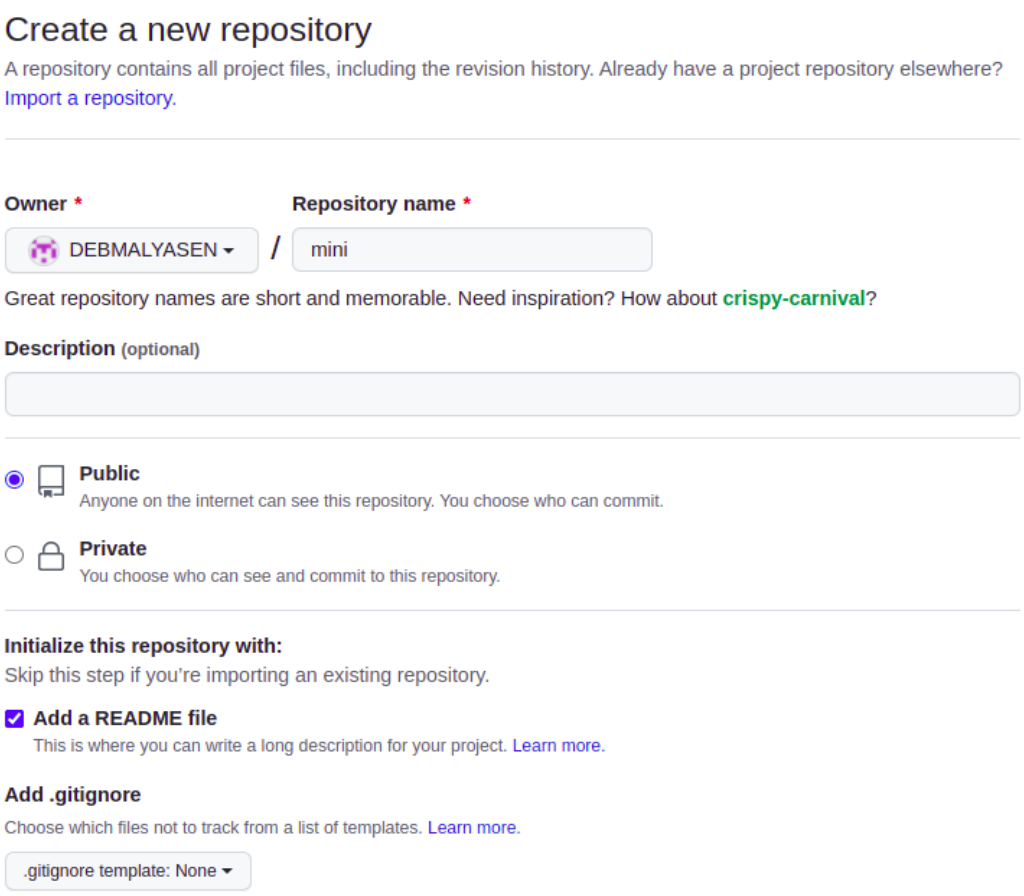


Figure 1: Creating a new repository

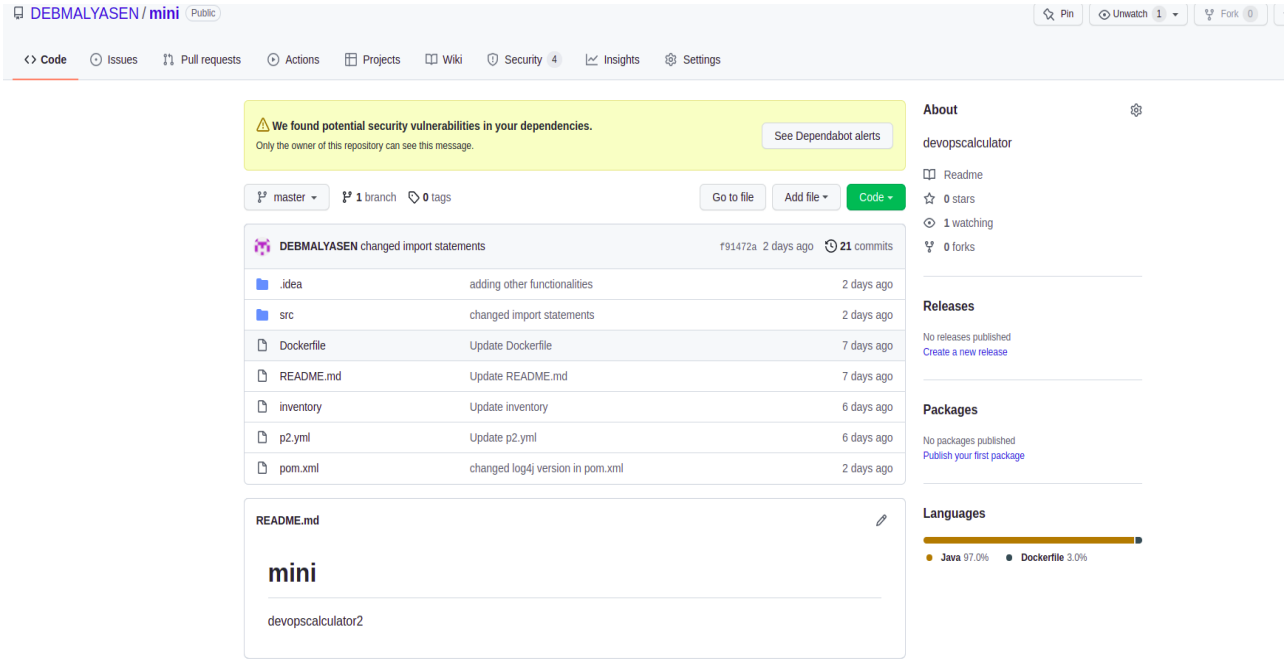


Figure 2: Source Code of the Calculator project

After creating the project , since a new repository has to be created the steps you need to follow on git terminal are :

git init

git add README.md

git commit -m "Commit message"

git remote add origin https://github.com/DEBMALYASEN/mini

git push -u origin master

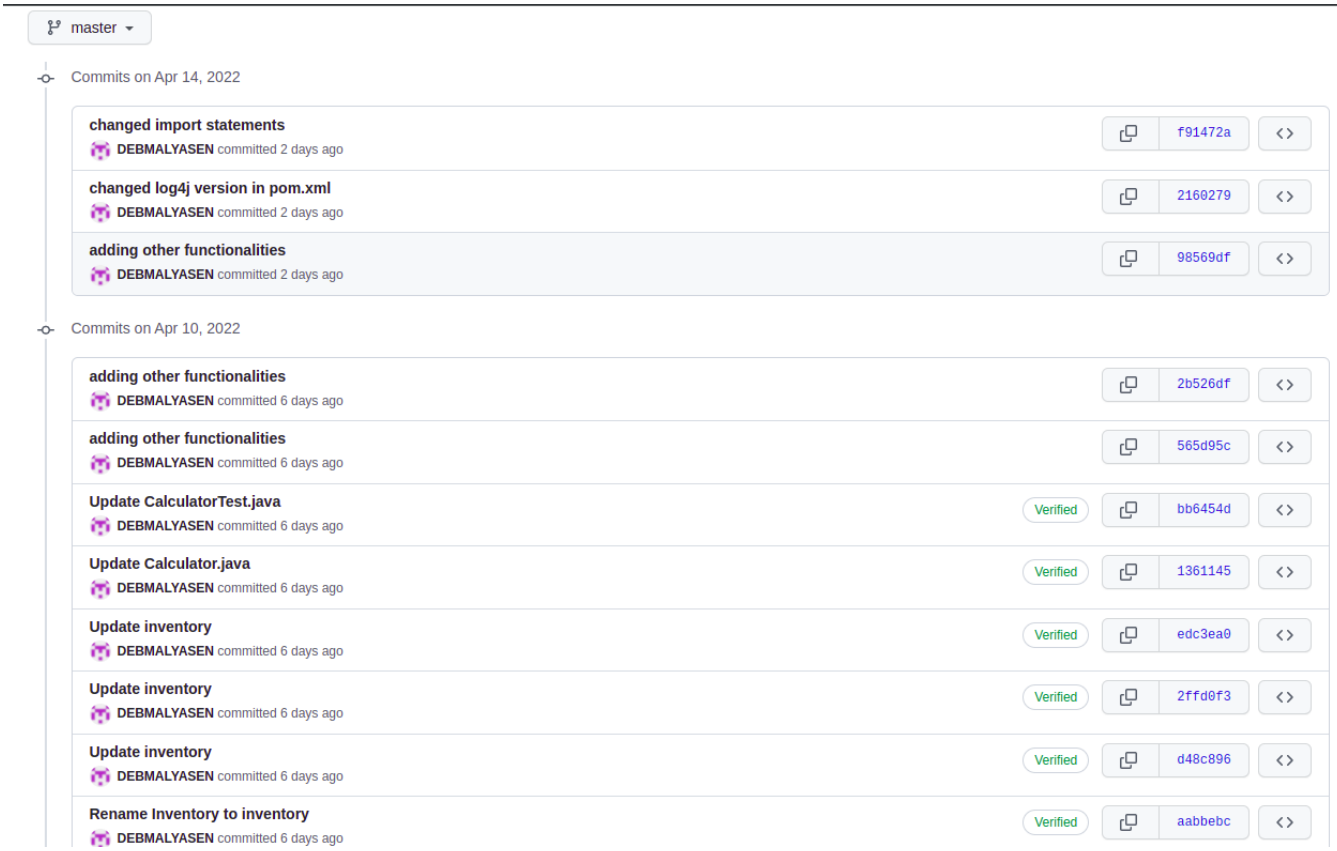


Figure 3: List of all subsequent commits while creation of the project

2. Code Development and Build

For code development and build Intellij has been used. Build of the code is done using Maven.

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-dependenc y-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.

```
<dependencies>
    <dependency>
        <groupId>net.logstash.logback</groupId>
        <artifactId>logstash-logback-encoder</artifactId>
        <version>4.7</version>
    </dependency>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>4.13.1</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.apache.logging.log4j</groupId>
        <artifactId>log4j-api</artifactId>
        <version>2.14.0</version>
    </dependency>
    <dependency>
        <groupId>org.apache.logging.log4j</groupId>
        <artifactId>log4j-core</artifactId>
        <version>2.14.0</version>
    </dependency>
```

Figure 4: List of dependencies added in pom.xml

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

Figure 5: Log4j2.xml

lo4j2.xml creates a log file for test cases.

These log4j2 properties are must for log4j apache jar. The log4j2.properties handles projects log functionalities, be its structure or where it should be stored or append or create a new file based on date and time. Along with it Dockerfile which would be used to create docker image of the build has also been added. The docker file tells the build should be built on which image. For unit testing Junit dependencies have been added and test cases have been written for false(assertNotEquals) as well as true assert(assertEquals).

3. Testing:

Since there are four major functions that this calculator performs, for different test case files have been created.
Screenshots of the same have been added below.

```
@Test
public void factorialTruePositive()
{
    assertEquals( message: "Finding factorial of a number for True Positive",  expected: 720, calculator.fact( num: 6), DELTA);
    assertEquals( message: "Finding factorial of a number for True Positive",  expected: 24, calculator.fact( num: 4), DELTA);
}

@Test
public void factorialFalsePositive()
{
    assertNotEquals( message: "Finding factorial of a number for False Positive",  unexpected: 120, calculator.fact( num: 6), DELTA);
    assertNotEquals( message: "Finding factorial of a number for False Positive",  unexpected: 24, calculator.fact( num: 2), DELTA);
}
```

Figure 6: Test cases for factorial function

```
@Test
public void logTruePositive()
{
    assertEquals( message: "Finding natural log for True Positive",  expected: 0, calculator.naturalLog( number1: 1), DELTA);
    assertEquals( message: "Finding natural log for True Positive",  expected: 0, calculator.naturalLog( number1: 1), DELTA);
}

@Test
public void logFalsePositive()
{
    assertNotEquals( message: "Finding natural log for False Positive",  unexpected: 6, calculator.naturalLog( number1: 2.4), DELTA);
    assertNotEquals( message: "Finding natural log for False Positive",  unexpected: 7.3, calculator.naturalLog( number1: 2.1), DELTA);
}
```

Figure 7: Test cases for Natural Log function

```
@Test
public void powerTruePositive()
{
    assertEquals( message: "Finding power for True Positive",  expected: 16, calculator.power(2, 4), DELTA);
    assertEquals( message: "Finding power for True Positive",  expected: 64, calculator.power(4, 3), DELTA);
}

@Test
public void powerFalsePositive()
{
    assertNotEquals( message: "Finding power for False Positive",  unexpected: 8, calculator.power(2, 2), DELTA);
    assertNotEquals( message: "Finding power for False Positive",  unexpected: 10, calculator.power(2, 3), DELTA);
}
```

Figure 8: Test cases for Power function

```
@Test
public void sqrootTruePositive()
{
    assertEquals( message: "Finding square root for True Positive",  expected: 4, calculator.sqroot( number1: 16), DELTA);
    assertEquals( message: "Finding square root for True Positive",  expected: 9, calculator.sqroot( number1: 81), DELTA);
}

@Test
public void sqrootFalsePositive()
{
    assertNotEquals( message: "Finding square root for False Positive",  unexpected: 1, calculator.sqroot( number1: 3), DELTA);
    assertNotEquals( message: "Finding square root for False Positive",  unexpected: 0, calculator.sqroot( number1: 4), DELTA);

}
```

Figure 9: Test cases for square Root function

These test cases have been included for both assert negative as well positive cases

After adding all the test cases, the entire project is build using maven.
The command for build is: mvn clean test
The command entered in terminal shows that build was successful.

```
08:08:33.109 [Calculator.java] INFO  calculator.Calculator - [SQ ROOT] - 4.0
08:08:33.110 [Calculator.java] INFO  calculator.Calculator - [RESULT - SQ ROOT] - 2.0
08:08:33.110 [Calculator.java] INFO  calculator.Calculator - [NATURAL LOG] - 1.0
08:08:33.111 [Calculator.java] INFO  calculator.Calculator - [RESULT - NATURAL LOG] - 0.0
08:08:33.111 [Calculator.java] INFO  calculator.Calculator - [NATURAL LOG] - 1.0
08:08:33.112 [Calculator.java] INFO  calculator.Calculator - [RESULT - NATURAL LOG] - 0.0
08:08:33.113 [Calculator.java] INFO  calculator.Calculator - [POWER - 2.0 RAISED TO] 2.0
08:08:33.114 [Calculator.java] INFO  calculator.Calculator - [RESULT - POWER] - 4.0
08:08:33.114 [Calculator.java] INFO  calculator.Calculator - [POWER - 2.0 RAISED TO] 3.0
08:08:33.114 [Calculator.java] INFO  calculator.Calculator - [RESULT - POWER] - 8.0
08:08:33.115 [Calculator.java] INFO  calculator.Calculator - [POWER - 2.0 RAISED TO] 4.0
08:08:33.116 [Calculator.java] INFO  calculator.Calculator - [RESULT - POWER] - 16.0
08:08:33.116 [Calculator.java] INFO  calculator.Calculator - [POWER - 4.0 RAISED TO] 3.0
08:08:33.116 [Calculator.java] INFO  calculator.Calculator - [RESULT - POWER] - 64.0
08:08:33.117 [Calculator.java] INFO  calculator.Calculator - [NATURAL LOG] - 2.4
08:08:33.117 [Calculator.java] INFO  calculator.Calculator - [RESULT - NATURAL LOG] - 0.8754687373538999
08:08:33.118 [Calculator.java] INFO  calculator.Calculator - [NATURAL LOG] - 2.1
08:08:33.118 [Calculator.java] INFO  calculator.Calculator - [RESULT - NATURAL LOG] - 0.7419373447293773
08:08:33.119 [Calculator.java] INFO  calculator.Calculator - [SQ ROOT] - 16.0
08:08:33.119 [Calculator.java] INFO  calculator.Calculator - [RESULT - SQ ROOT] - 4.0
08:08:33.120 [Calculator.java] INFO  calculator.Calculator - [SQ ROOT] - 81.0
08:08:33.120 [Calculator.java] INFO  calculator.Calculator - [RESULT - SQ ROOT] - 9.0
Tests run: 8, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.442 sec

Results :

Tests run: 8, Failures: 0, Errors: 0, Skipped: 0

[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  4.989 s
[INFO] Finished at: 2022-04-17T08:08:33+05:30
[INFO] ------------------------------------------------------------------------
(base) debmalya@debmalya-Nitro-AN515-52:~/Music/mini$
```

Figure 10: Build Success

4. Continuous Integration

The development process is complete and we have successfully pushed the code on to git,and built through Maven , Jenkins will be required for continuous integration and configuration management and continuous deployment. Now we setup Jenkins and other plugins for it.
Setting up Jenkins will require entering few commands on Linux terminal.

- wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add –
- sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'
- sudo apt update
- sudo apt install Jenkins

Then Create new account and login to Jenkins.

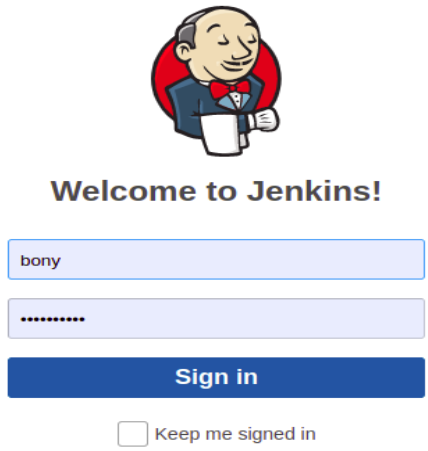


Figure 10: Sign in to Jenkins Page

after that we click on manage jenkins and then on manage Plugins.It will show multiple plugins which has to be downloaded before creating the pipeline.

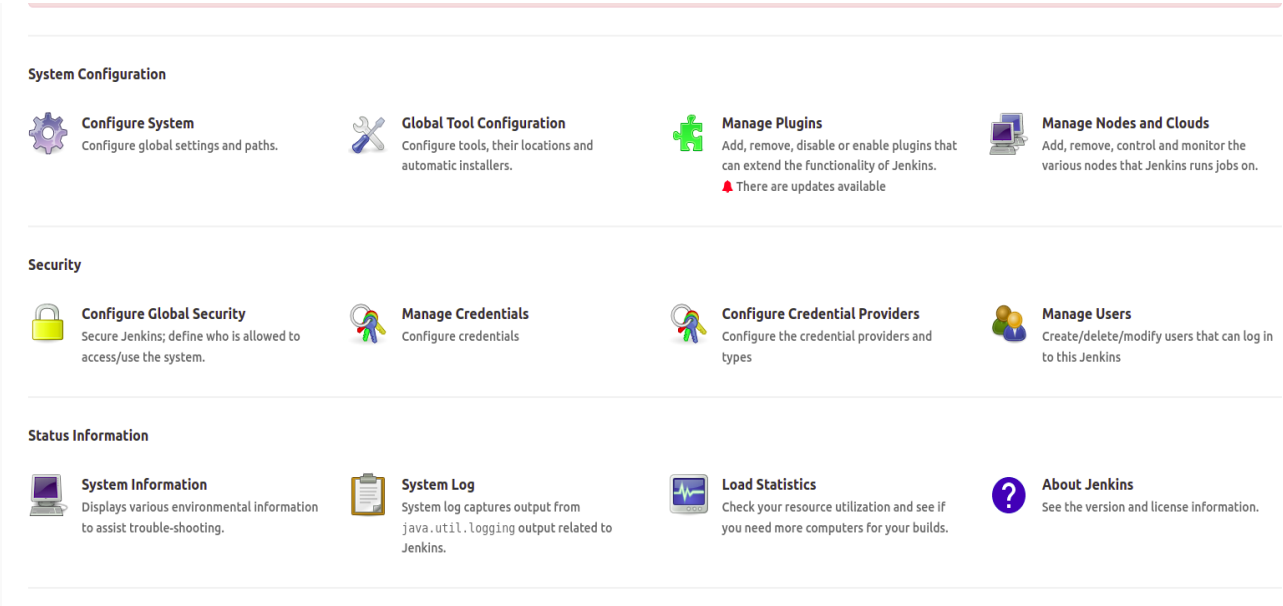


Figure 11: Adding Plugin

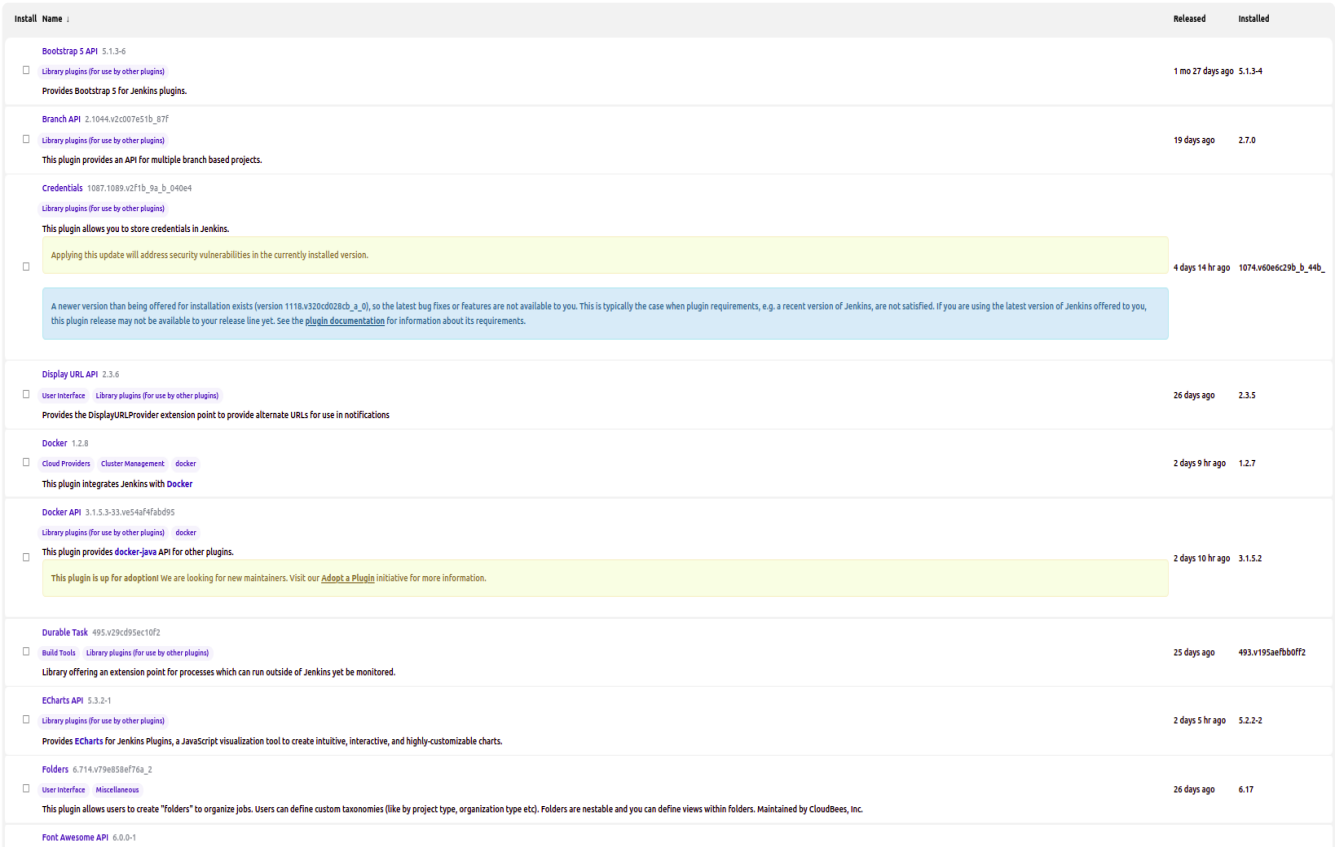


Figure 12: Plugins required and installed

Jenkins will download all the related dependencies required for these plugins automatically.

A new pipeline is created for creating tasks in multiple stages and for smooth functioning.
After it's done downloading, jenkins will restart and now you can create a new job for jenkins, enter jenkins job name and choose pipeline as job functionality.

Properties of the pipeline is it is script based and each stage of pipeline script runs one after another. Making it perfect for continuous integration and then deployment. Properties of continuous integration include SCM, unit testing and integration testing.

Setup of docker hub credentials in Jenkins.

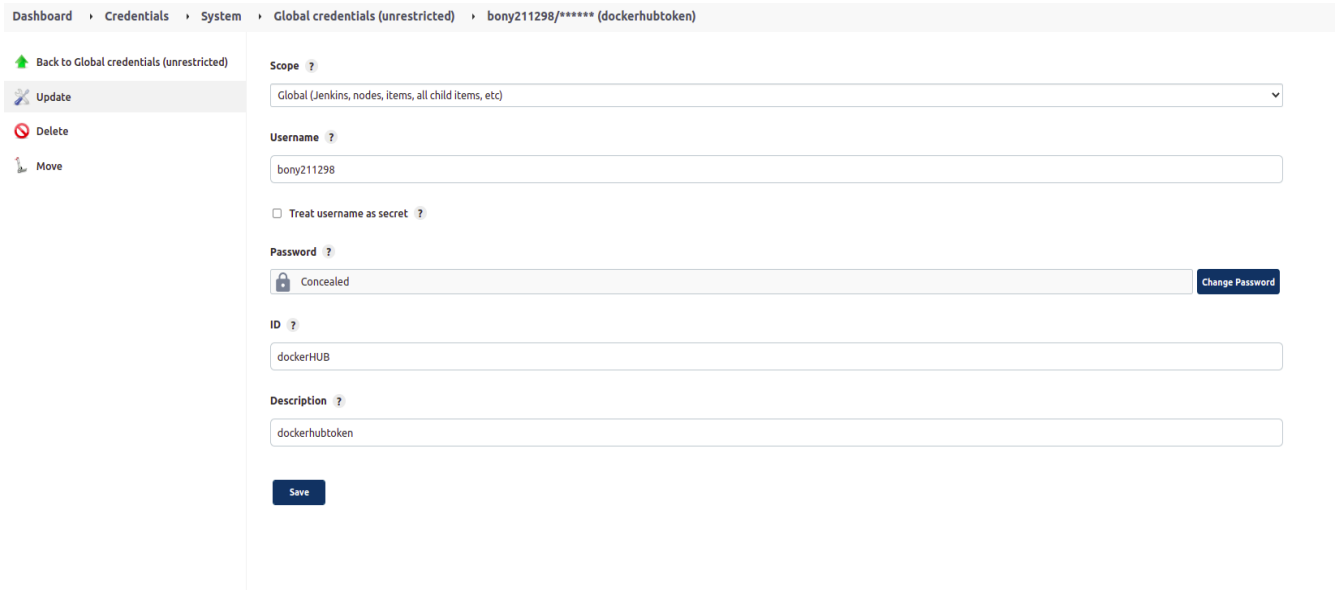


Figure 13: Docker Hub Credential

The steps taken for creating mentioned pipeline are:

Pull the project from GitHub

After clicking on Generate Pipeline Script , we get the script to add in the first stage of pipeline.

Next stage is Maven Build and for this script command mvn clean install has to be added.
After build Docker image has to be build. The main role of this command is to build image using docker file. Script command used is :

imageName = docker.build “bony211298/devopscalculator”
This image name is mentioned in p2.yml file.

The image built is pushed to docker hub.

Docker Hub repositories allow you share container images with your team, customers, or the Docker community at large. Docker images are pushed to Docker Hub through the docker push command. A single Docker Hub repository can hold many Docker images.
This docker image is pulled to a managed node, from docker hub
using p2.yml file.
For ansible pipeline stages can be generated as given in below screenshots.

Figure 14: Creating ansible pipeline stage

Here we have fill every field then automatically its get created which can be used in pipeline

The final script :

```
stage('Cloning Git')
{
  steps
  {
    git 'https://github.com/DEBMALYASEN/mini.git'
  }
}
```

Figure 15: Cloning Git

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

Figure 16:Building

```
stage('Test')
{
    steps
    {
        sh 'mvn test'
    }
}
```

Figure 17:Test stage

```
stage('Building image') {
  steps
  {
    sh 'docker build -t bony211298/devopscalculator:latest .'

  }
}
```

Figure 18:Building Image

```
stage('Deploy Image')
{
  steps
  {

      withDockerRegistry([credentialsId:"dockerHUB",url:""])
      {
        sh 'docker push bony211298/devopscalculator:latest '
      }
  }
}
```

Figure 19:Deploy Image

```
stage("Clean Docker Image")
{
    steps
    {
        sh 'docker rmi  -f bony211298/devopscalculator:latest '
    }
}
```

Figure 20 :Clean Docker Image

```
stage(' Ansible Deploy') {
        steps {

            withDockerRegistry([ credentialsId: 'dockerhub', url: "" ]) {
                ansiblePlaybook colorized: true, disableHostKeyChecking: true, installation: 'Ansible', inventory: 'inventory', playbook: 'p2.yml'
            }
        }
    }
```

Figure 21:Ansible Deploy

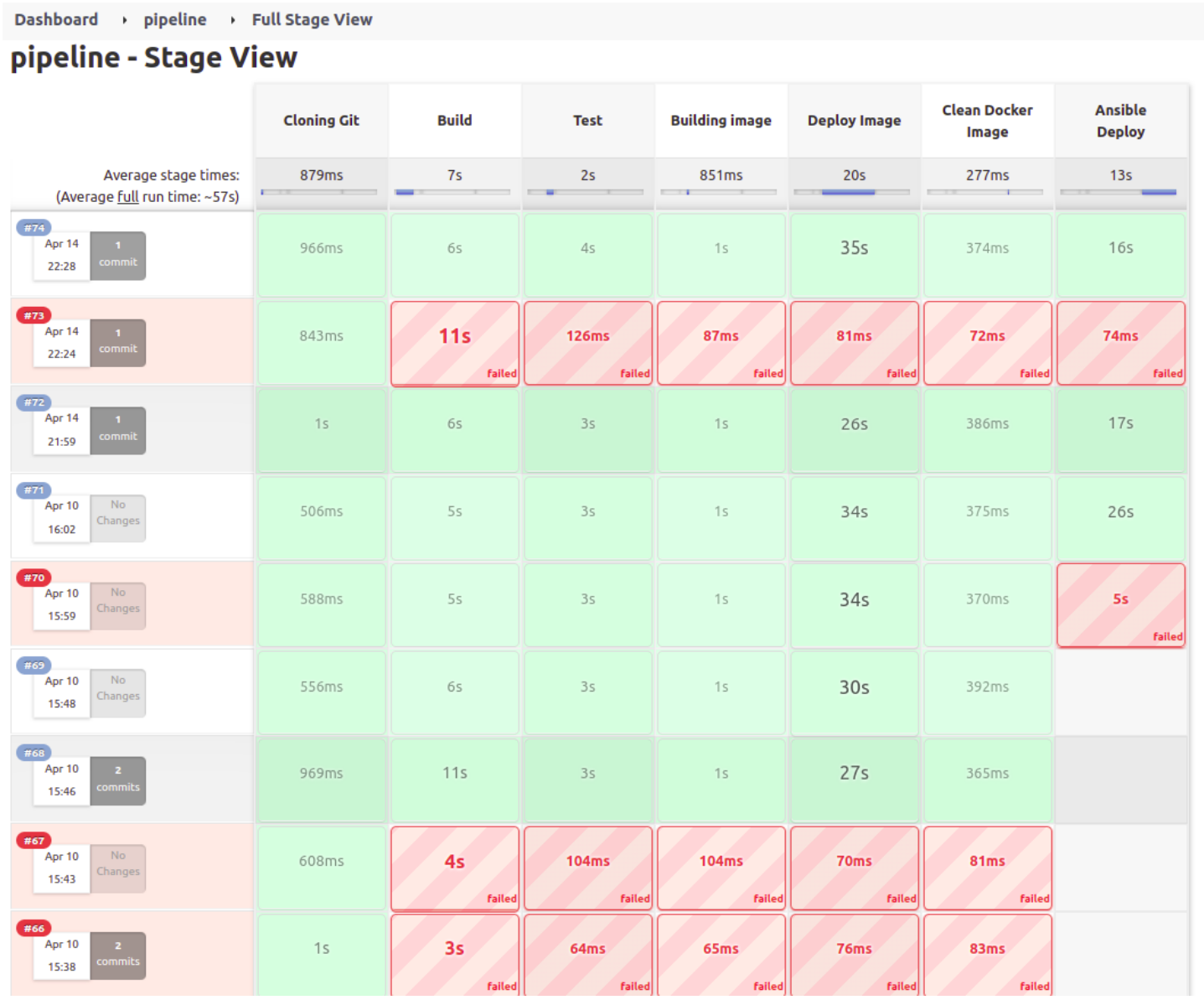


Figure 22: Stage view of complete pipeline

5. **Containerize**:

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications.

Steps to install Docker:

- sudo apt update
- Sudo apt install docker-ce
- sudo systemctl status docker // to check if the docker system is active or not

To run docker on your machine, either run it as root or as user. To run it as user you have to enter user to the group of docker which can be done using **sudo usermod –aG docker <username>**
Run the docker on to terminal via command **sudo systemctl start docker**
To pull the docker image use command **docker pull <docker_image_name:tag>**
To push the docker image use command :-
docker push <username>/<repository_name>:tagname

The docker image is pushed from jenkins after the build stage and pulled on from ansible job. The docker image can be run using command **docker run –i –t <image_name>**

Docker File:
Initially openjdk image was pulled and jar file was copied to the root directory of docker.

workDir./

The WORKDIR command is used to define the working directory of a Docker container at any given time. The command is specified in the Dockerfile.
Any RUN, CMD, ADD, COPY, or ENTRYPOINT command will be executed in the specified working directory.
The below command will run as soon as docker starts and it will run the jar file.
CMD ["java", "-jar", "mini-1.0-SNAPSHOT-jar-with-dependencies.jar", "Main"]

The docker images can be found on the link:
https://hub.docker.com/repository/docker/bony211298/devopscalculator

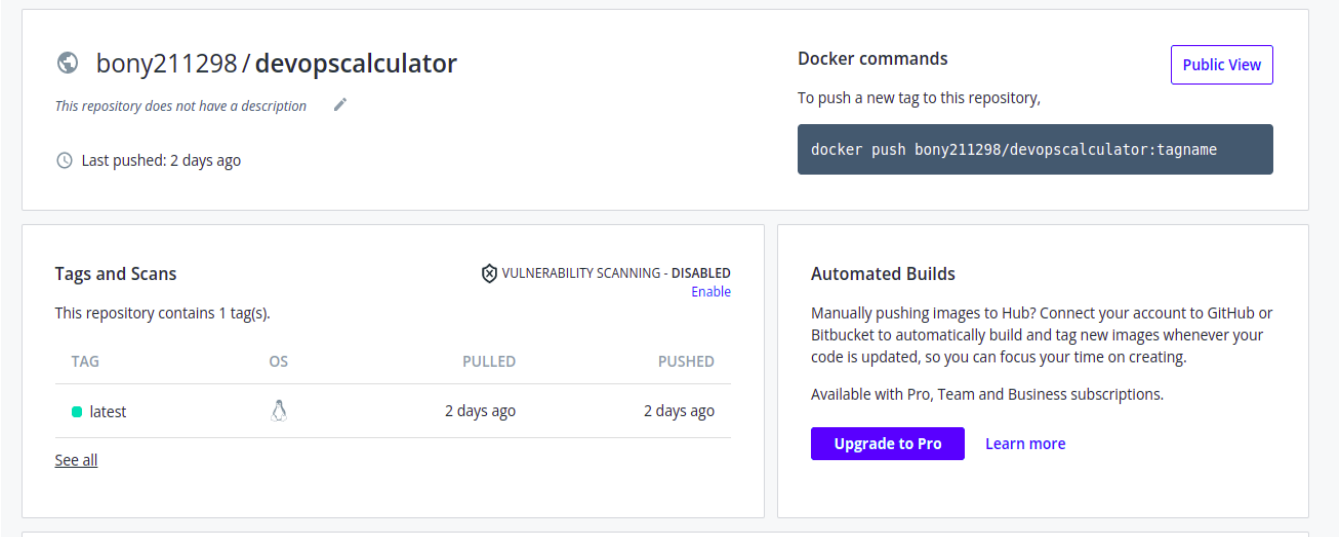


Figure 23: Docker Hub

6. Deployment

Ansible is an open-source automation platform. It is automation engine that runs ansible playbooks, playbooks are defined tasks, where we define environments and workflows.

Steps to install Ansible:

- Sudo apt install openssh-server
- Ssh-keygen -t rsa
- Ssh-copy-id <username>@<ip>
- Sudo apt install ansible

Ansible playbook

```
---
- name: Pull docker image of Calculator
  hosts: all
  tasks:
    - name: Pull image
      docker_image:
        name: bony211298/devopscalculator
        source: pull
```

Figure 24: Ansible Playbook

Inventory File :

The Ansible inventory file defines the hosts and groups of hosts upon which commands, modules, and tasks in a playbook operate. The file can be in one of many formats depending on your Ansible environment and plugins.

```
[ubuntu18]
172.16.133.201 ansible_user=debmalya ansible_pthon_interpreter=/usr/bin/python3
```

Figure 25: Inventory File

```
(base) debmalya@debmalya-Nitro-AN515-52:~$ sudo chmod 666 /var/run/docker.sock
(base) debmalya@debmalya-Nitro-AN515-52:~$ docker images
REPOSITORY                    TAG         IMAGE ID       CREATED        SIZE
bony211298/devopscalculator   latest      1d0a25183e97   2 days ago     530MB
bony211298/devopscalculator   <none>      6e813d476f52   2 days ago     529MB
bony211298/devopscalculator   <none>      f72f9b685554   6 days ago     529MB
bony211298/devopscalculator   <none>      5571fd0187b7   6 days ago     529MB
<none>                        <none>      12e29bfdc65f   6 days ago     529MB
bony211298/devopscalculator   <none>      343a5b54e15e   6 days ago     529MB
<none>                        <none>      5d5ed6cc3ee3   6 days ago     529MB
<none>                        <none>      fc61e999b2d5   7 days ago     529MB
<none>                        <none>      a26b8e07c1a6   7 days ago     529MB
<none>                        <none>      51f79737e23d   7 days ago     529MB
<none>                        <none>      57a96566bbad   7 days ago     529MB
<none>                        <none>      88e9e13a877d   7 days ago     529MB
<none>                        <none>      3639e70299e8   7 days ago     529MB
<none>                        <none>      51c23753e2c5   7 days ago     529MB
<none>                        <none>      19ab3a12dfba   7 days ago     529MB
<none>                        <none>      6fb768194956   7 days ago     529MB
openjdk                       8           18fbe41f975e   2 weeks ago    526MB
jenkins/jenkins               lts-jdk11   ba37b97921de   5 weeks ago    464MB
hello-world                   latest      feb5d9fea6a5   6 months ago   13.3kB
(base) debmalya@debmalya-Nitro-AN515-52:~$ docker run -it bony211298/devopscalculator /bin/bash
root@e75cdb33ae39:/#
exit
(base) debmalya@debmalya-Nitro-AN515-52:~$ ls-lrt
ls-lrt: command not found
(base) debmalya@debmalya-Nitro-AN515-52:~$ docker run -it bony211298/devopscalculator /bin/bash
root@fab924fac710:/# ls -lrt
total 4352
drwxr-xr-x   2 root root    4096 Mar 19 13:46 home
drwxr-xr-x   2 root root    4096 Mar 19 13:46 boot
drwxr-xr-x   1 root root    4096 Mar 28 00:00 var
drwxr-xr-x   1 root root    4096 Mar 28 00:00 usr
drwxr-xr-x   2 root root    4096 Mar 28 00:00 srv
drwxr-xr-x   3 root root    4096 Mar 28 00:00 run
drwxr-xr-x   2 root root    4096 Mar 28 00:00 opt
drwxr-xr-x   2 root root    4096 Mar 28 00:00 mnt
drwxr-xr-x   2 root root    4096 Mar 28 00:00 media
drwxr-xr-x   2 root root    4096 Mar 28 00:00 lib64
drwxr-xr-x   1 root root    4096 Mar 28 00:00 lib
drwxr-xr-x   1 root root    4096 Mar 29 17:29 sbin
drwxr-xr-x   1 root root    4096 Mar 29 23:11 bin
drwx------   1 root root    4096 Mar 29 23:13 root
drwxrwxrwt   1 root root    4096 Mar 29 23:13 tmp
-rw-r--r--   1 root root 4388759 Apr 14 16:58 mini-1.0-SNAPSHOT-jar-with-dependencies.jar
drwxr-xr-x   1 root root    4096 Apr 17 03:08 etc
dr-xr-xr-x  13 root root       0 Apr 17 03:08 sys
dr-xr-xr-x 435 root root       0 Apr 17 03:08 proc
drwxr-xr-x   5 root root     360 Apr 17 03:08 dev
root@fab924fac710:/# java -jar  mini-1.0-SNAPSHOT-jar-with-dependencies.jar
```

```
root@fab924fac710:/# java -jar  mini-1.0-SNAPSHOT-jar-with-dependencies.jar
Calculator-DevOps, Choose to perform operation
Press 1 to find factorial
Press 2 to find Square root
Press 3 to find power
Press 4 to find natural logarithm
Press 5 to exit
Enter your choice: 1
Enter a number : 5
03:10:16.524 [Calculator.java] INFO  calculator.Calculator - [FACTORIAL] - 5.0
03:10:16.527 [Calculator.java] INFO  calculator.Calculator - [RESULT - FACTORIAL] - 120.0
Factorial of 5.0 is : 120.0


Calculator-DevOps, Choose to perform operation
Press 1 to find factorial
Press 2 to find Square root
Press 3 to find power
Press 4 to find natural logarithm
Press 5 to exit
Enter your choice: 2
Enter a number : 64
03:10:42.720 [Calculator.java] INFO  calculator.Calculator - [SQ ROOT] - 64.0
03:10:42.720 [Calculator.java] INFO  calculator.Calculator - [RESULT - SQ ROOT] - 8.0
Square root of 64.0 is : 8.0


Calculator-DevOps, Choose to perform operation
Press 1 to find factorial
Press 2 to find Square root
Press 3 to find power
Press 4 to find natural logarithm
Press 5 to exit
Enter your choice: 3
Enter the first number : 2
Enter the second number : 3
03:10:57.197 [Calculator.java] INFO  calculator.Calculator - [POWER - 2.0 RAISED TO] 3.0
03:10:57.198 [Calculator.java] INFO  calculator.Calculator - [RESULT - POWER] - 8.0
2.0 raised to power 3.0 is : 8.0


Calculator-DevOps, Choose to perform operation
Press 1 to find factorial
Press 2 to find Square root
Press 3 to find power
Press 4 to find natural logarithm
Press 5 to exit
Enter your choice: 4
Enter a number : 10
03:11:06.853 [Calculator.java] INFO  calculator.Calculator - [NATURAL LOG] - 10.0
03:11:06.854 [Calculator.java] INFO  calculator.Calculator - [RESULT - NATURAL LOG] - 2.302585092994046
Natural log of 10.0 is : 2.302585092994046
```

Figure 26: Deployed docker image on local system

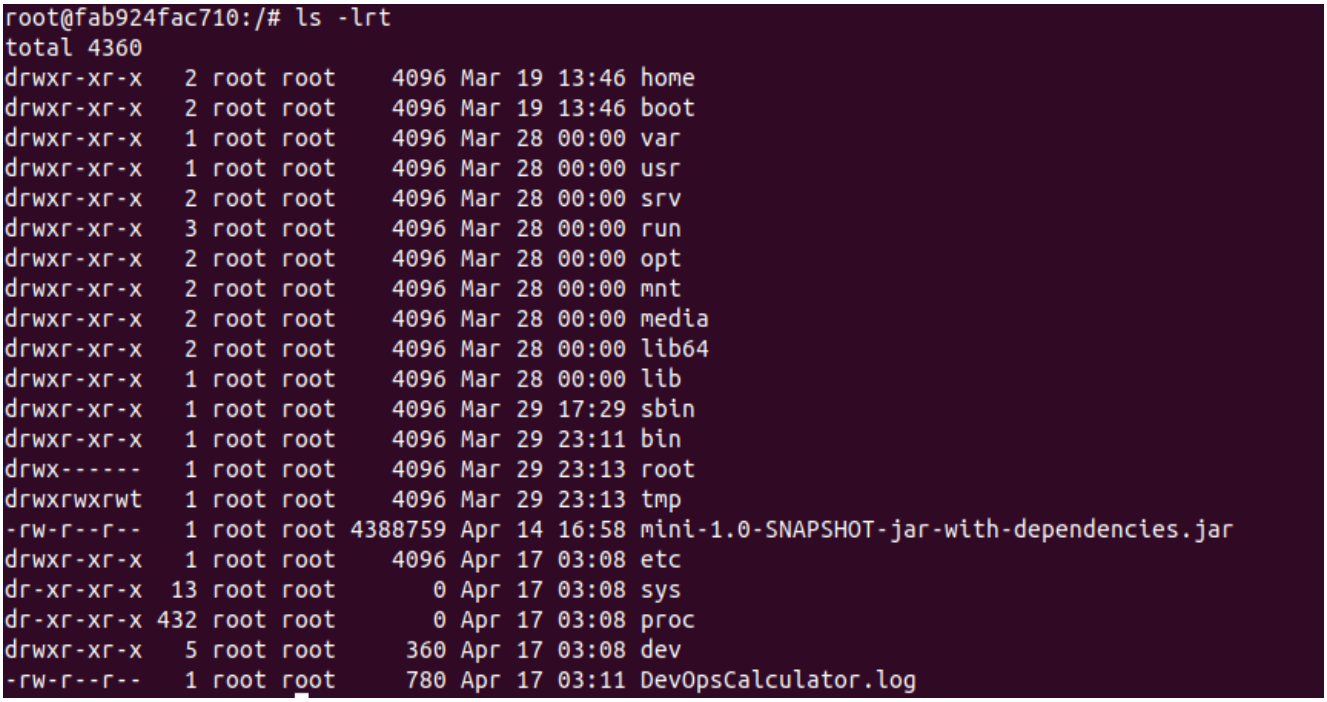


Figure 27:Formation of DevOpsCalculator.log file

7. Monitoring

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. ElasticSearch, LogStash and Kibana are all developed, managed ,and maintained by the company named Elastic.

To start with download elastic search, logstash and kibana from https://www.elastic.co Run them side by side and feed the logstash your log set after configuring it config file. In the config file we provide the details for logs. Here the config file logstash.conf.

Elastic search starts at localhost:9200, and kibana starts at localhost:5601.

Commands to run

- Elastic search - **./path_to_elastic_search/bin/elastic**
- Logstash - **./path_to_logstash/bin/logstash –f ./path_to_logstash.conf**
- Kibana - **./path_to_kibana/bin/kibana**

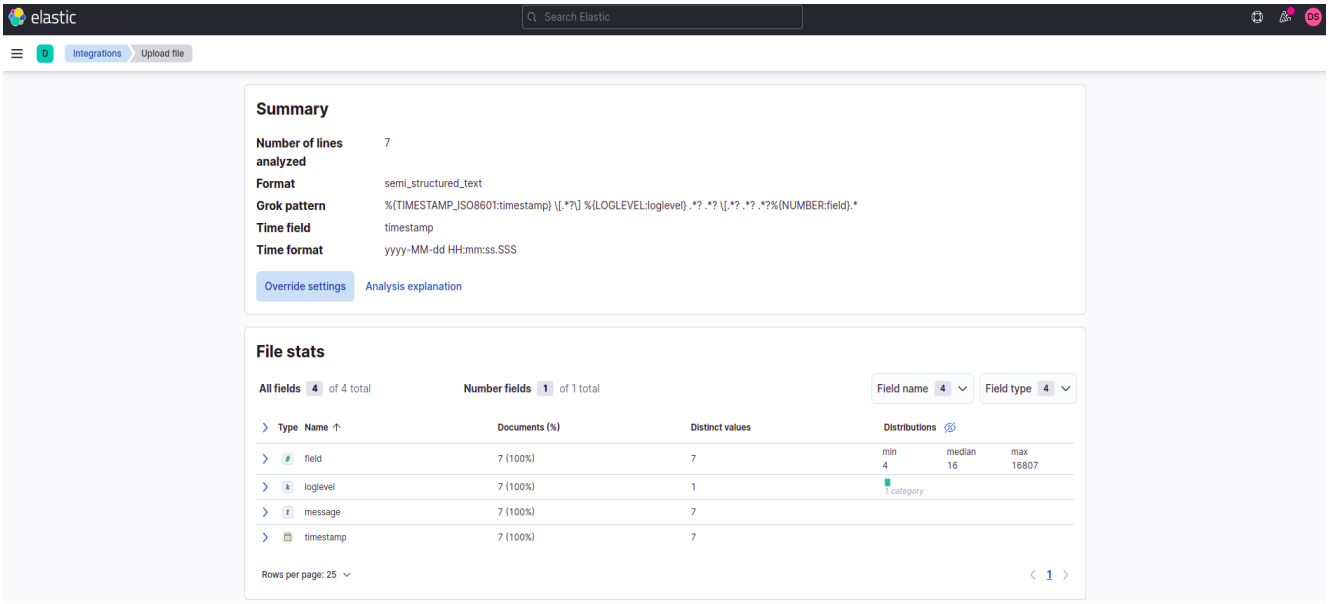


Figure 28: Log file update in ELK Cloud

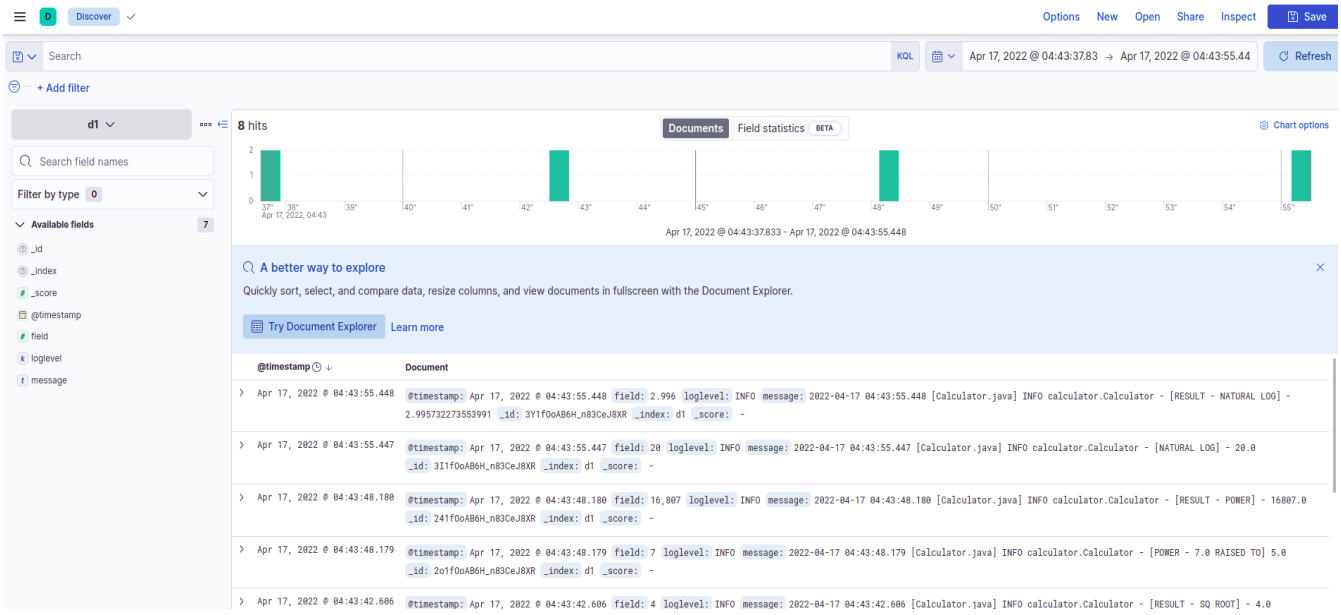


Figure 29: Visualization in ELK

8. Challenges and solutions

- Maven Build Error :
  While building the project in intellij, an error was generated “are you trying to run JRE instead of JDK.
  So I found out that there were two versions of JDK installed in my system and I uninstalled one of them, which resolved the error.

- Permission denied (docker daemon)

```
PLAY [Pull docker image of Calculator] ***************************************

TASK [Gathering Facts] *******************************************************
[0;32mok: [172.16.133.201][0m
[0;32m[0m
TASK [Pull image] ************************************************************
[0;31mfatal: [172.16.133.201]: FAILED! => {"changed": false, "msg": "Error connecting: Error while fetching server API version: ('Connection aborted.', PermissionError(13, 'Permission denied'))"}[0m
```

  Got permission denied while trying to connect to the Docker daemon socket at unix:///var/run/docker.sock
  This usually happens if your docker is not running, you can check it via sudo systemctl status docker, it also occurs if your jenkins user doesn’t have docker permission which can be easily given via sudo usermod –aG jenkins docker.
  Don’t forget to restart jenkins job after that using sudo systemctl restart Jenkins.
  OR
  You can use sudo chmod 666 /var/run/docker.soc

- Ansible pipeline syntax error: Had to add “ installation: ’Ansible’ “ explicity.

- MalformedURLException:Solved this by removing space in the url=” “ in docker deploy stage
- The DevopsCalculator.log file was not getting created ,so I had to update log4j dependencies .
- Ansible failed to connect host via ssh -

  UNREACHABLE! => {"changed": false, "msg": "Failed to connect to the host via ssh: Warning: Permanently added '172.16.133.201’ (ECDSA) to the list of known hosts.\r\debmalya@172.16.133.201: Permission denied (publickey,password).", "unreachable": true}.

  Ansible was using the Jenkins user, while it had to connect to the managed host via SSH.
  I used the ssh-copy-id command on Jenkins user to reach the managed host.