CS 816 - Software Production Engineering
Mini Project - Scientific Calculator with DevOps

Hardeep Singh Arora - MT2022047

Links

GitHub:- https://github.com/hardeep0444/SPE_Calculator
Docker Hub:-
https://hub.docker.com/repository/docker/hardeep8011/junit-devops/general

Problem Statement

Create a scientific calculator program with user menu driven operations

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

Use DevOps toolchain to create a CI/CD pipeline using Jenkins.
Test code using JUnit, Selenium etc.
Build code using Maven, Gradel etc.
Containerize using Docker.
Configuration Management using Ansible, Chef etc.
Monitor (log files) using ELK stack.

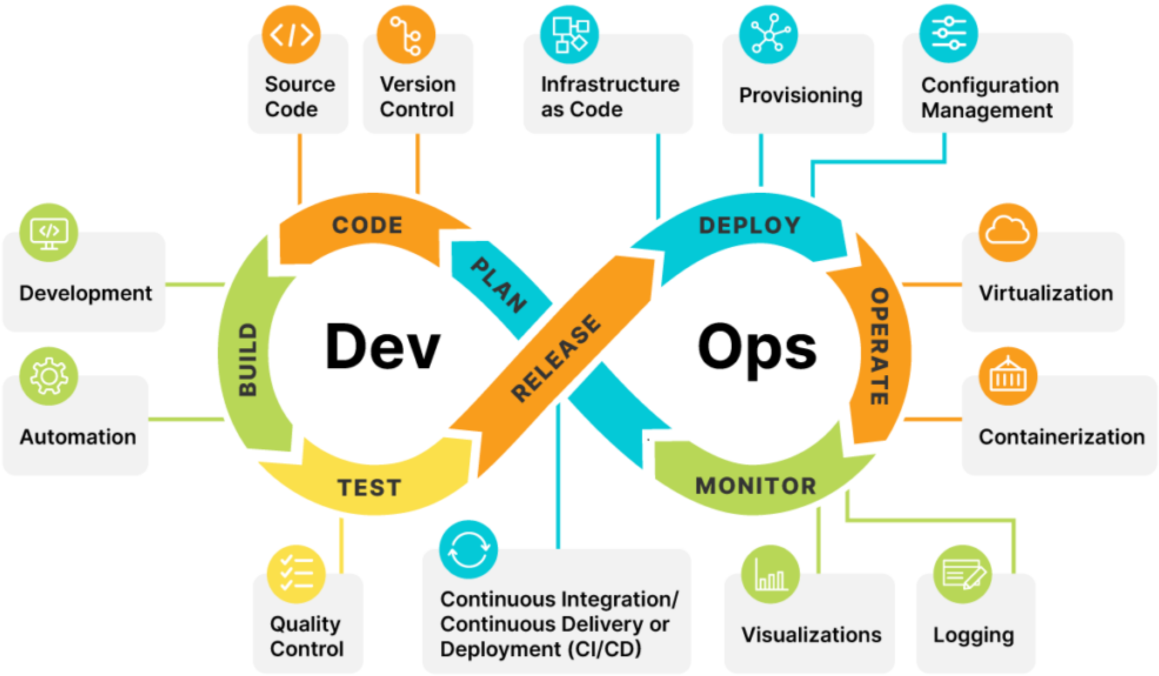
DevOps

- What is DevOps ?
    - DevOps is a set of cultural concepts, practices, and technologies that improves an organization's capacity to produce high-velocity applications and services, allowing it to evolve and improve products at a faster rate than traditional software development and infrastructure management methods. Organizations can better service their clients and compete in the market because of this quickness.

    - Development and operations teams are no longer "silos" in a DevOps architecture. These two teams are sometimes combined into a single team where the engineers work across the whole application lifecycle, from development and testing to deployment and operations, and develop a diverse set of abilities that aren't limited to a particular role.

- Quality assurance and security teams may become more closely linked with development and operations, as well as throughout the application lifecycle, in some DevOps models. When everyone in a DevOps team is focused on security, this is referred to as DevSecOps.

- These groups employ best practices to automate procedures that were previously manual and slow. They employ a technological stack and infrastructure that allows them to swiftly and reliably operate and evolve apps.These tools also assist engineers in independently completing tasks (such as deploying code or supplying infrastructure) that would ordinarily require assistance from other teams, hence increasing a team's velocity.



- Why DevOps ?
    - DevOps is important because it's a software development and operations approach that enables faster development of new products and easier maintenance of existing deployments.

      Five major benefits of using DevOps are :-
        1) Shorter Development Cycles, Faster Innovation
        2) Reduced Deployment Failures, Rollbacks, and Time to Recover
        3) Improved Communication and Collaboration
        4) Increased Efficiencies
        5) Reduced Costs and IT Headcount

Tools Used

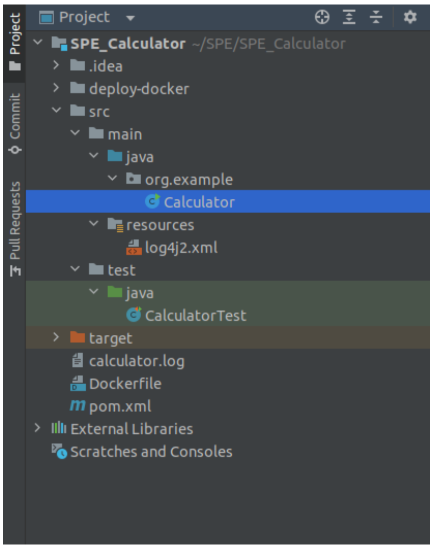
- ★ **Maven:** It's a Java-based application development tool that lets us add dependencies and build a jar file (a snapshot of our project) that can be run on any machine.
- ★ **GitHub:** Helps in automation through Jenkin Integration.
- ★ **Jenkins:** It is used for DevOps(for Continuous Integration and Continuous Deployment portion)
- ★ **Docker:** It is used to make images through containerization.
- ★ **Ansible:** It automates and simplifies repetitive, complex, and tedious operations. It saves a lot of time when we install packages or configure large numbers of servers.

Steps

- ➢ Install Java and IntelliJ.
- ➢ Write the Calculator code in Maven.
- ➢ Push the code into Github
- ➢ Create a repository in DockerHub for the project.
- ➢ Write Pipeline Script in Jenkins
    - ➔ Git Pull
    - ➔ Maven build
    - ➔ Docker Image creation
    - ➔ Pushing Image to Docker Hub
    - ➔ Ansible Deploy
- ➢ Build the project.
- ➢ Pull the image into the remote server
- ➢ Run the image

Development, Software Build, and Test

The code is developed in Java 11 and the IntelliJ IDE is utilised as the development environment. Log4j is used to keep track of logs for monitoring, and JUnit is used for unit testing.



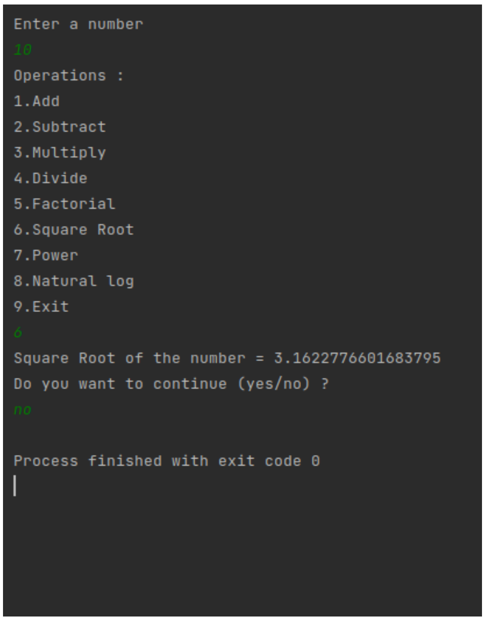
```
SPE_Calculator ~/SPE/SPE_Calculator
  .idea
  deploy-docker
  src
    main
      java
        org.example
          Calculator
      resources
        log4j2.xml
    test
      java
        CalculatorTest
  target
  calculator.log
  Dockerfile
  pom.xml
External Libraries
Scratches and Consoles
```

Calculator.java: It contains the main code of the project, which contains the following functions.

1. Addition
2. Subtraction
3. Multiplication
4. Division
5. Factorial
6. Power
7. Square Root
8. Natural log

CalculatorTest.java: It contains true and false positive test cases used to test the code when we build the project. It is performed using JUnit.

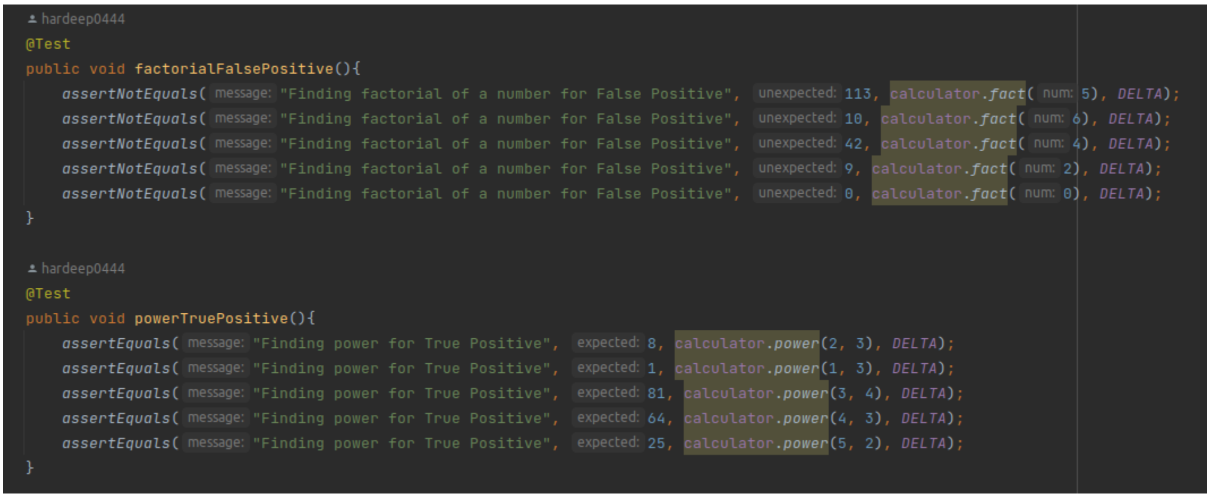
Output:



```
Enter a number
10
Operations :
1.Add
2.Subtract
3.Multiply
4.Divide
5.Factorial
6.Square Root
7.Power
8.Natural log
9.Exit
6
Square Root of the number = 3.1622776601683795
Do you want to continue (yes/no) ?
no

Process finished with exit code 0
```

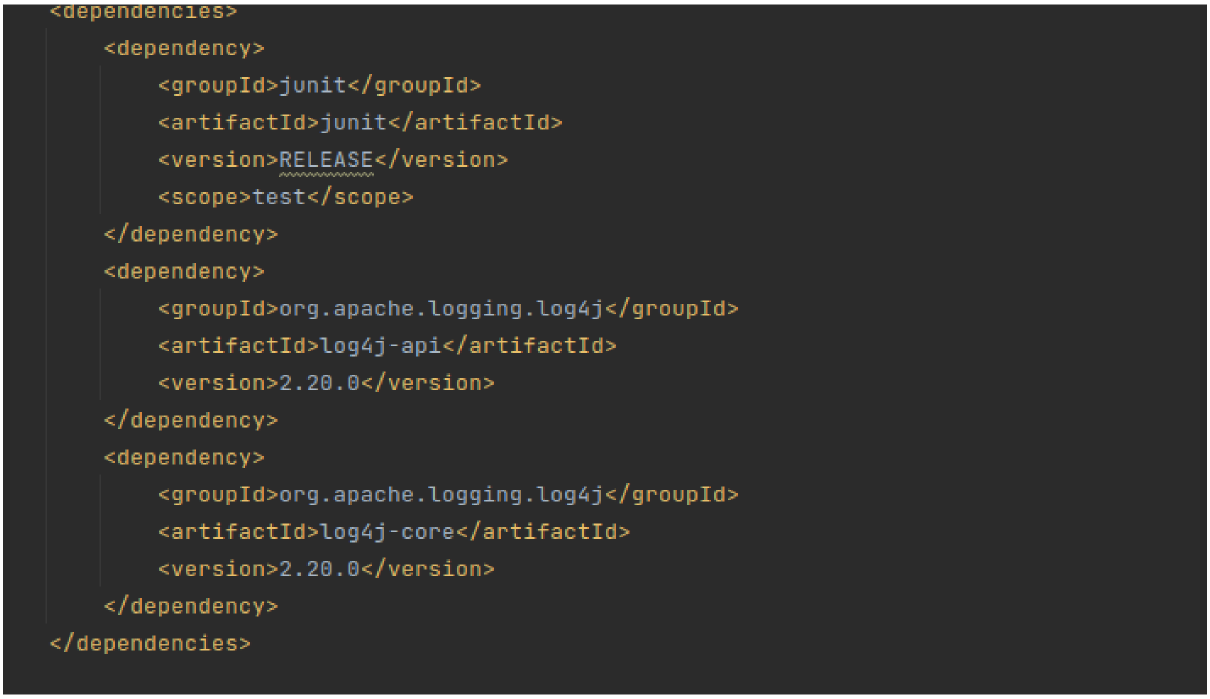
Test-Cases: For every functionality, two types of test cases are used, one is a True Positive and the other is a False Positive.



```
hardeep0444
@Test
public void factorialFalsePositive(){
    assertNotEquals( message: "Finding factorial of a number for False Positive",  unexpected: 113, calculator.fact( num: 5), DELTA);
    assertNotEquals( message: "Finding factorial of a number for False Positive",  unexpected: 10, calculator.fact( num: 6), DELTA);
    assertNotEquals( message: "Finding factorial of a number for False Positive",  unexpected: 42, calculator.fact( num: 4), DELTA);
    assertNotEquals( message: "Finding factorial of a number for False Positive",  unexpected: 9, calculator.fact( num: 2), DELTA);
    assertNotEquals( message: "Finding factorial of a number for False Positive",  unexpected: 0, calculator.fact( num: 0), DELTA);
}

hardeep0444
@Test
public void powerTruePositive(){
    assertEquals( message: "Finding power for True Positive",  expected: 8, calculator.power(2, 3), DELTA);
    assertEquals( message: "Finding power for True Positive",  expected: 1, calculator.power(1, 3), DELTA);
    assertEquals( message: "Finding power for True Positive",  expected: 81, calculator.power(3, 4), DELTA);
    assertEquals( message: "Finding power for True Positive",  expected: 64, calculator.power(4, 3), DELTA);
    assertEquals( message: "Finding power for True Positive",  expected: 25, calculator.power(5, 2), DELTA);
}
```

Project Dependencies: To use JUnit and log4j, we need to add certain jar files in the pom.xml file. So **Maven** will add those dependencies.



```
<dependencies>
    <dependency>
        <groupId>junit</groupId>
        <artifactId>junit</artifactId>
        <version>RELEASE</version>
        <scope>test</scope>
    </dependency>
    <dependency>
        <groupId>org.apache.logging.log4j</groupId>
        <artifactId>log4j-api</artifactId>
        <version>2.20.0</version>
    </dependency>
    <dependency>
        <groupId>org.apache.logging.log4j</groupId>
        <artifactId>log4j-core</artifactId>
        <version>2.20.0</version>
    </dependency>
</dependencies>
```

Now by doing **$ mvn clean install** the entire code builds and all the test cases will be verified. A new folder named “**target**” is automatically created, which contains the **.jar** file.

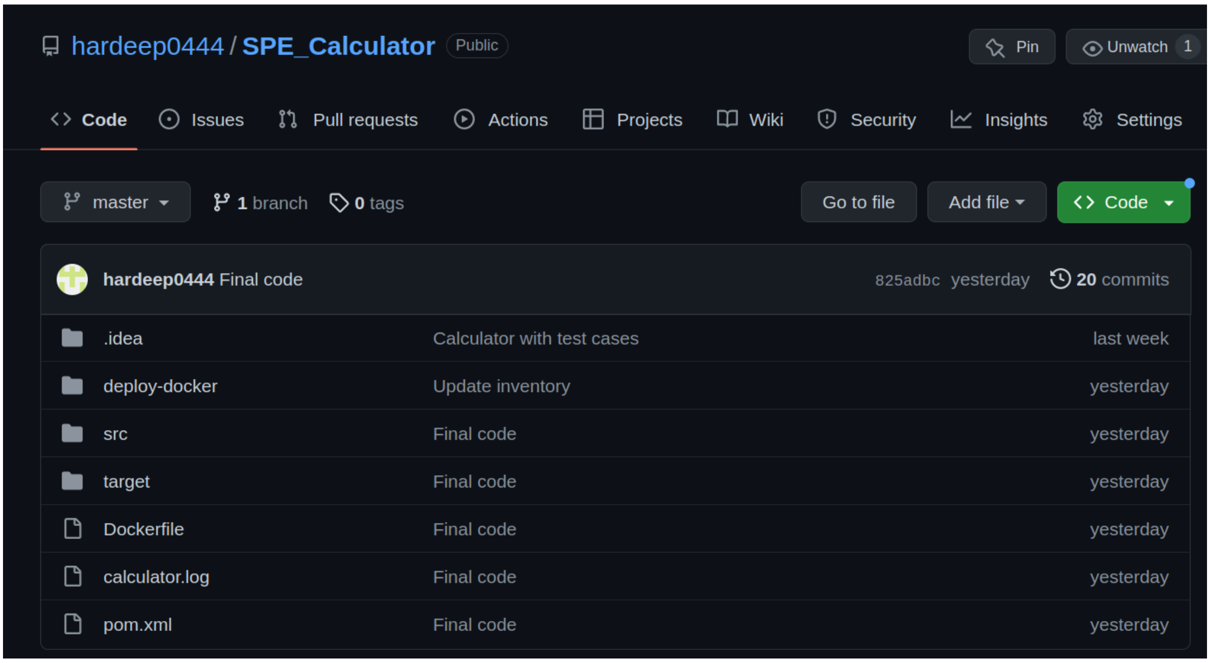
Source Code Management - GitHub

Source code management (SCM) is used to track modifications to a source code repository. SCM tracks a running history of changes to a code base and helps resolve conflicts when merging updates from multiple contributors. SCM is also synonymous with Version control.
GitHub is used to accomplish SCM.

Create a new repository at https://github.com/ to get started. We can build a new repository by providing it with a unique name connected with the user. The SCM, which will be connected to Jenkins as an input, will manage our code.

Steps:

1. Create a public repository.
2. $ git init
3. $ git add .
4. $ git remote add origin <github repository URL>
5. $ git commit -m “type a message here”
6. $ git push origin branch name (master or main)



hardeep0444 Final code 825adbc yesterday 20 commits

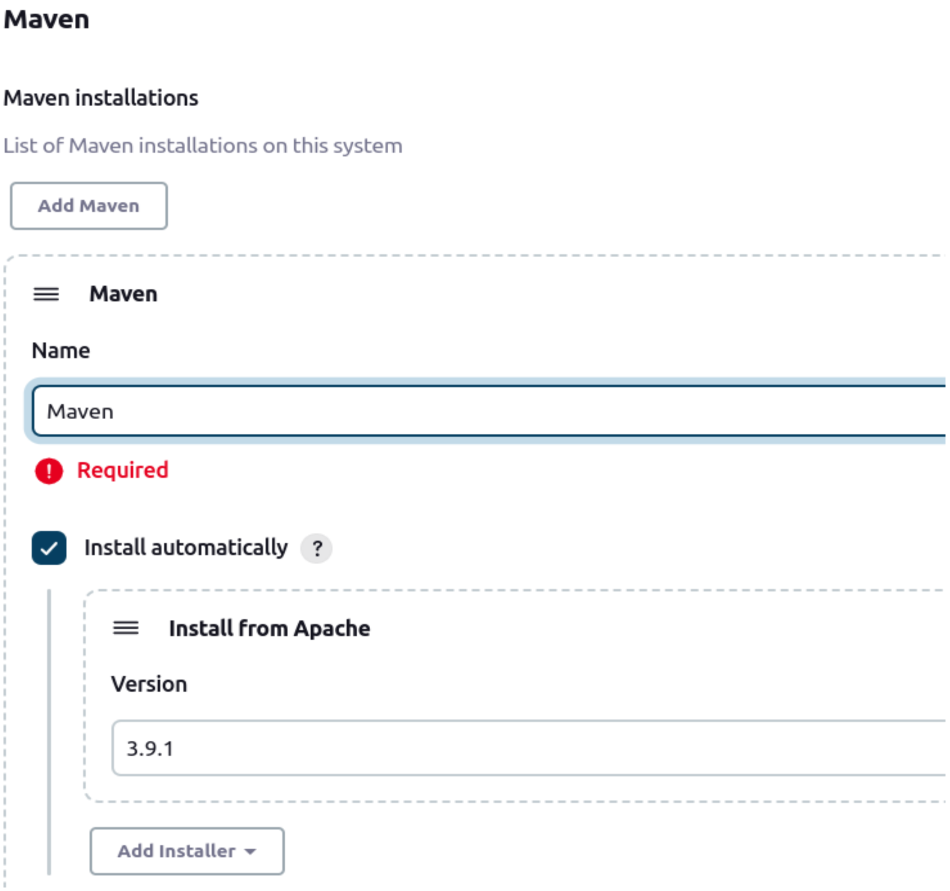
.idea	Calculator with test cases	last week
deploy-docker	Update inventory	yesterday
src	Final code	yesterday
target	Final code	yesterday
Dockerfile	Final code	yesterday
calculator.log	Final code	yesterday
pom.xml	Final code	yesterday

Jenkins

- Jenkins is an open source automation server. It helps automate the parts of software development related to building, testing, and deploying, facilitating continuous integration and continuous delivery. It is a server-based system that runs in servlet containers such as Apache Tomcat. It supports version control tools like Git,CVS etc.

  The Jenkins pipeline was utilised in this project to handle until delivery, i.e. continuous delivery. http://localhost:8080 is the URL for the Jenkins service.

- **Plugins**
  Install the following plugins from Plugin Manager:
  - ➢ Maven
  - ➢ Git
  - ➢ Ansible
  - ➢ Docker
- **Manage Jenkins –> Global Tool Configuration**



Maven

Maven installations

List of Maven installations on this system

Maven

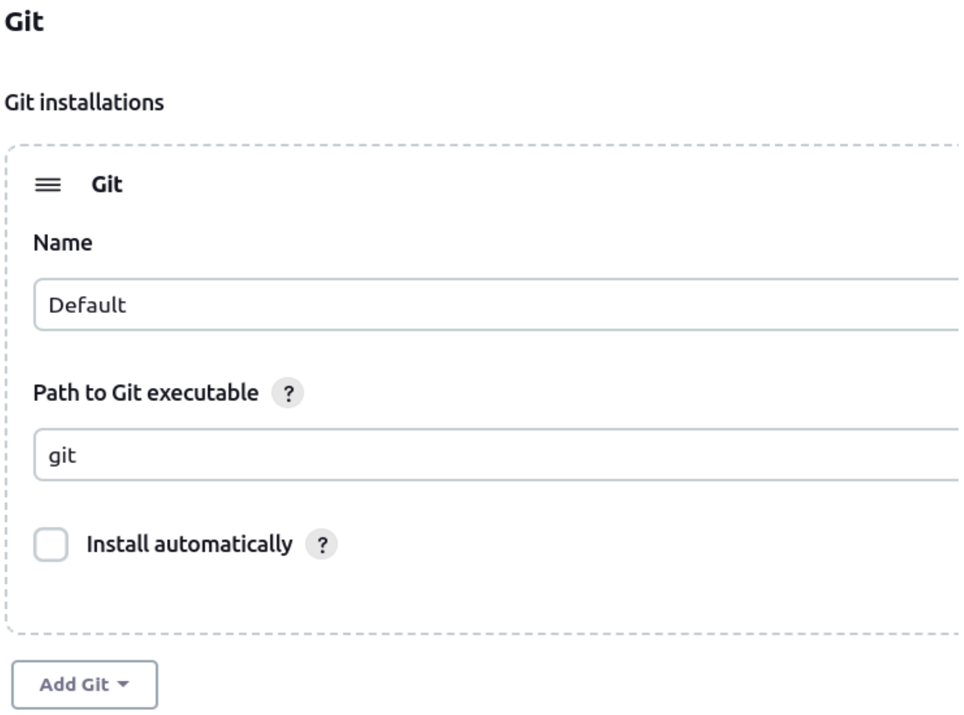
Name: Maven

Required

☑ Install automatically ?

Install from Apache

Version: 3.9.1



Git

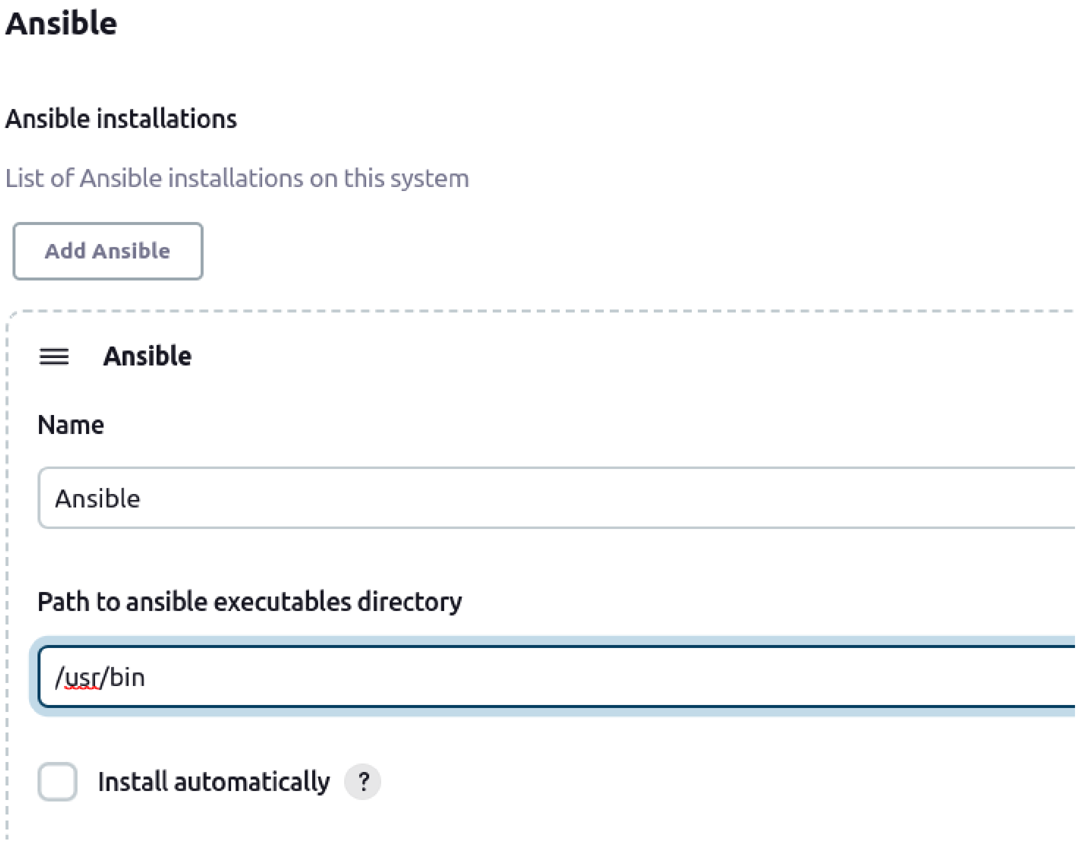
Git installations

Git

Name: Default

Path to Git executable: git

□ Install automatically



Ansible

Ansible installations

List of Ansible installations on this system

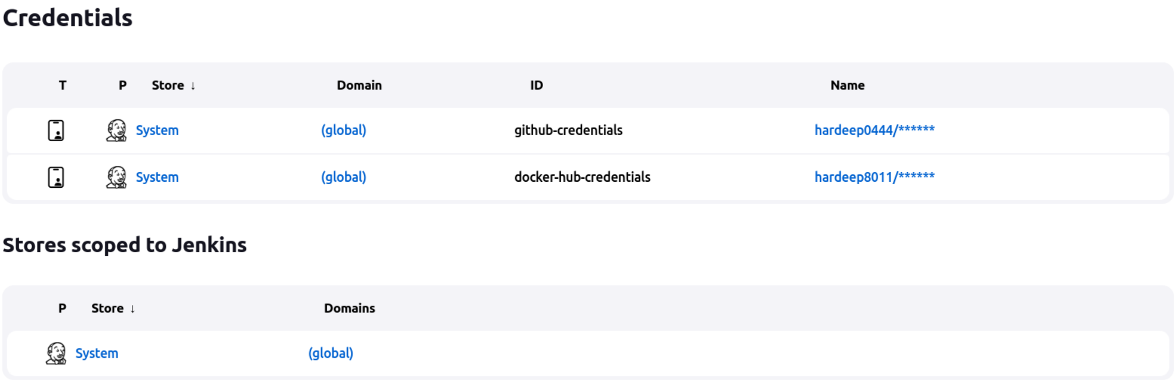
Ansible

Name: Ansible

Path to ansible executables directory: /usr/bin

□ Install automatically

- **Manage Jenkins –> Manage Credentials**



Credentials

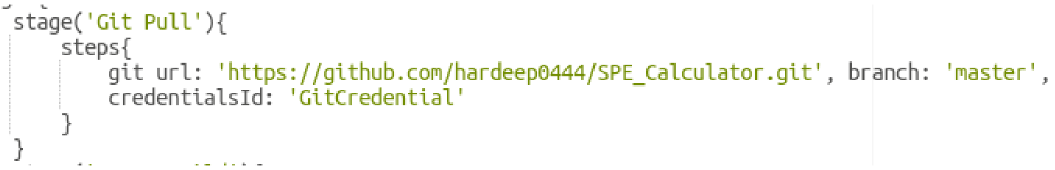
T	P	Store ↓	Domain	ID	Name
		System	(global)	github-credentials	hardeep0444/******
		System	(global)	docker-hub-credentials	hardeep8011/******

Stores scoped to Jenkins

P	Store ↓	Domains
	System	(global)

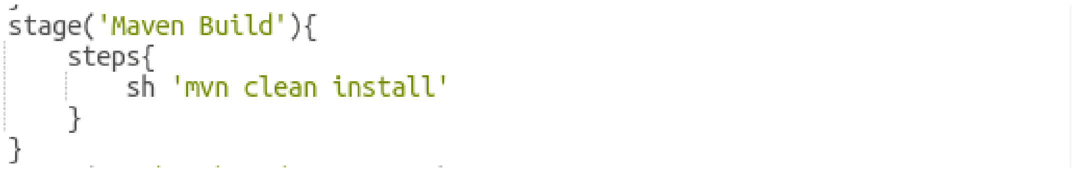
Jenkins Pipeline

1. **Git Pull:** Pulls the remote repository from github using Jenkins.



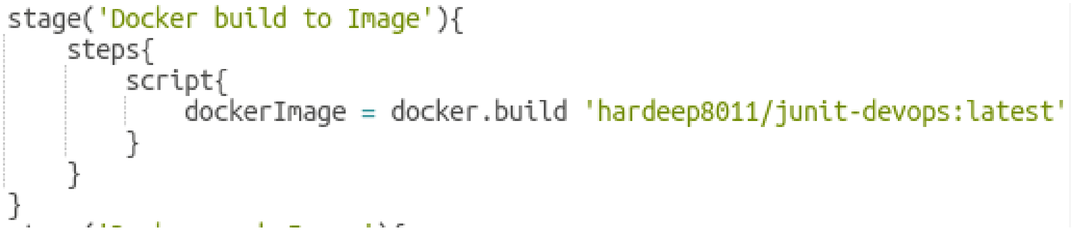
```
stage('Git Pull'){
    steps{
        git url: 'https://github.com/hardeep0444/SPE_Calculator.git', branch: 'master',
        credentialsId: 'GitCredential'
    }
}
```

2. **Maven Build:** It contains the jar file that contains our source code coupled with any dependencies. The existing target folder with old dependencies will be replaced, by a fresh target folder with the new jar.



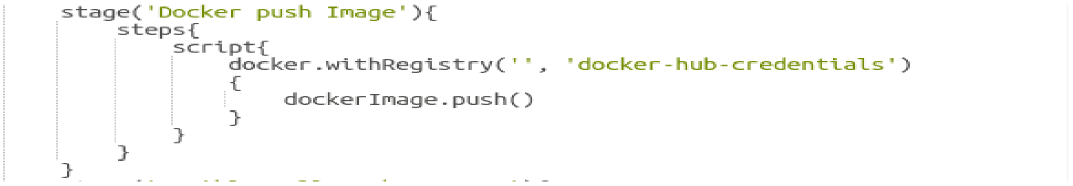
```
stage('Maven Build'){
    steps{
        sh 'mvn clean install'
    }
}
```

3. **Docker Image Creation:** It is used to create images on our local machine, which is later pushed in the docker hub. Pushing the image in the docker hub allows us to pull the image and run the application on other servers.



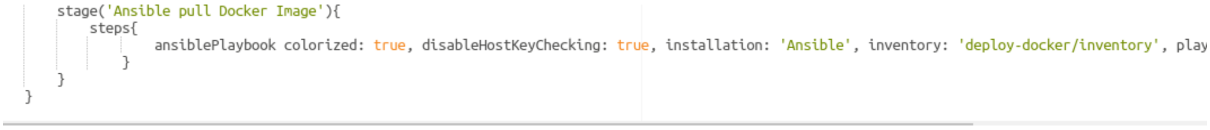
```
stage('Docker build to Image'){
    steps{
        script{
            dockerImage = docker.build 'hardeep8011/junit-devops:latest'
        }
    }
}
```

4. **Deploying Docker Image:** This is the part where we deploy the image into docker hub so that anyone could later pull the image.

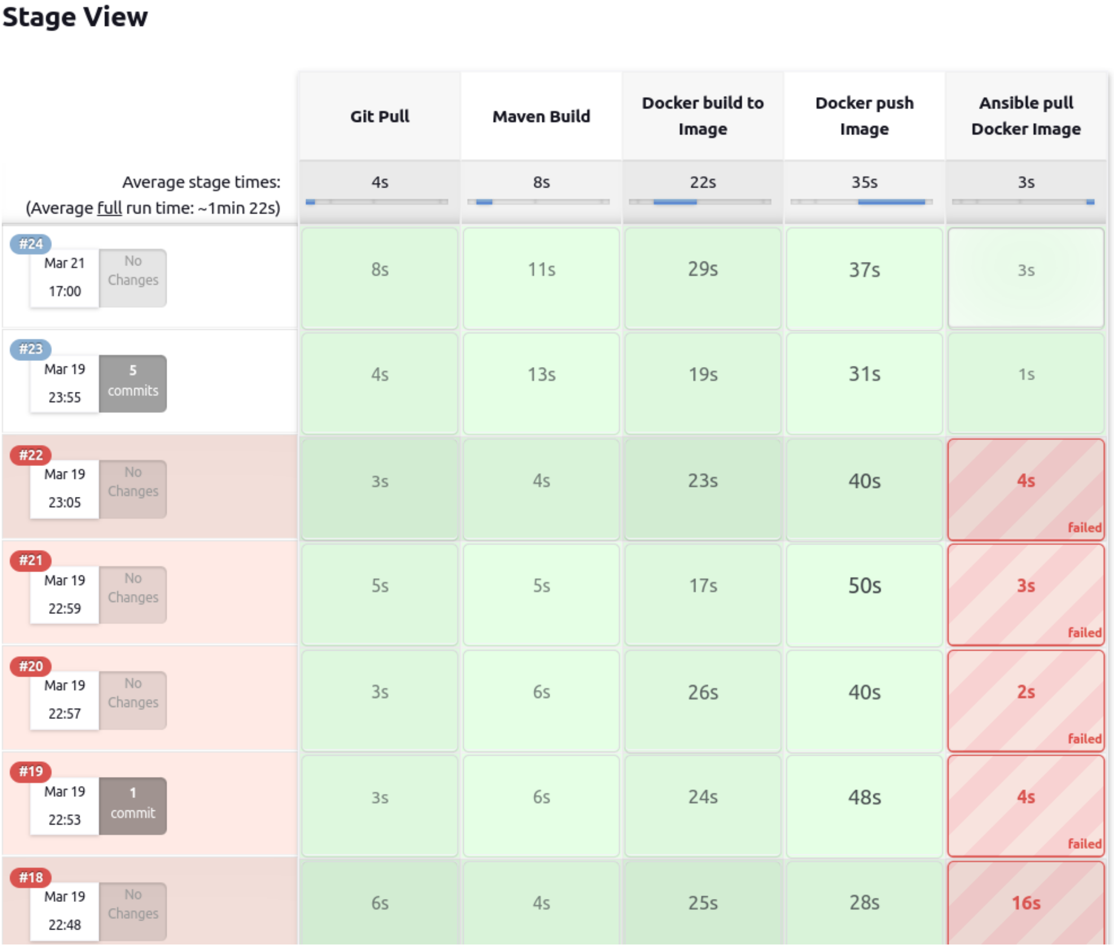


```
stage('Docker push Image'){
    steps{
        script{
            docker.withRegistry('', 'docker-hub-credentials')
            {
                dockerImage.push()
            }
        }
    }
}
```

5. **Ansible Deploy:** This is the part where we try to pull the docker image from the docker hub. The places where we need to pull the docker image are all written in the inventory file.



```
    stage('Ansible pull Docker Image'){
        steps{
            ansiblePlaybook colorized: true, disableHostKeyChecking: true, installation: 'Ansible', inventory: 'deploy-docker/inventory', play
        }
    }
}
```

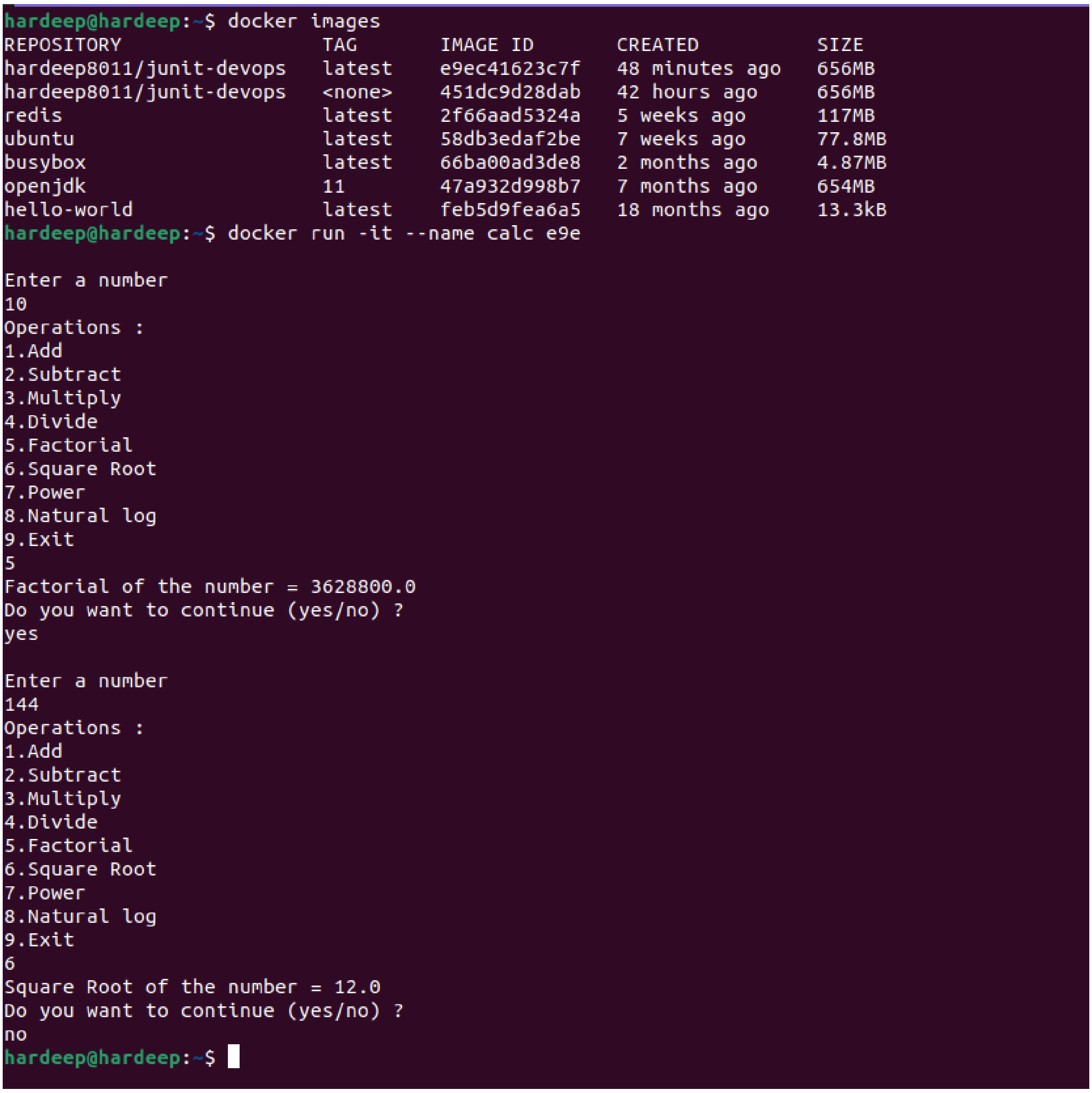


Stage View

	Git Pull	Maven Build	Docker build to Image	Docker push Image	Ansible pull Docker Image
Average stage times: (Average full run time: ~1min 22s)	4s	8s	22s	35s	3s
#24 Mar 21 17:00 No Changes	8s	11s	29s	37s	3s
#23 Mar 19 23:55 5 commits	4s	13s	19s	31s	1s
#22 Mar 19 23:05 No Changes	3s	4s	23s	40s	4s failed
#21 Mar 19 22:59 No Changes	5s	5s	17s	50s	3s failed
#20 Mar 19 22:57 No Changes	3s	6s	26s	40s	2s failed
#19 Mar 19 22:53 1 commit	3s	6s	24s	48s	4s failed
#18 Mar 19 22:48 No Changes	6s	4s	25s	28s	16s

After successful execution of the pipeline, a docker image can be found in the local machine.

We can execute the prog by running the command “**$ docker run -it –name <Container-Name> <Image-id>**”.



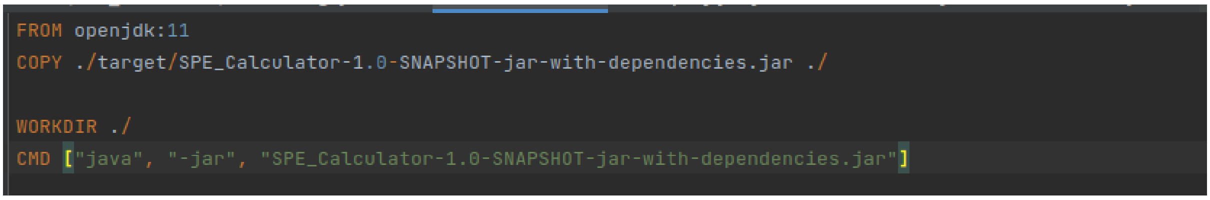
```
hardeep@hardeep:~$ docker images
REPOSITORY                 TAG       IMAGE ID       CREATED          SIZE
hardeep8011/junit-devops   latest    e9ec41623c7f   48 minutes ago   656MB
hardeep8011/junit-devops   <none>    451dc9d28dab   42 hours ago     656MB
redis                      latest    2f66aad5324a   5 weeks ago      117MB
ubuntu                     latest    58db3edaf2be   7 weeks ago      77.8MB
busybox                    latest    66ba00ad3de8   2 months ago     4.87MB
openjdk                    11        47a932d998b7   7 months ago     654MB
hello-world                latest    feb5d9fea6a5   18 months ago    13.3kB
hardeep@hardeep:~$ docker run -it --name calc e9e

Enter a number
10
Operations :
1.Add
2.Subtract
3.Multiply
4.Divide
5.Factorial
6.Square Root
7.Power
8.Natural log
9.Exit
5
Factorial of the number = 3628800.0
Do you want to continue (yes/no) ?
yes

Enter a number
144
Operations :
1.Add
2.Subtract
3.Multiply
4.Divide
5.Factorial
6.Square Root
7.Power
8.Natural log
9.Exit
6
Square Root of the number = 12.0
Do you want to continue (yes/no) ?
no
hardeep@hardeep:~$ 
```

Containerize

- Docker is an operating system virtualization platform that allows applications to be delivered in containers. As a result, rather than just supplying software, the full environment is provided as a Docker image, including all software dependencies.
- So, using open-JDK 11 and the "SPE_Calculator-1.0-SNAPSHOT-jar-with-dependencies.jar" file, we'll create a docker image. After that, the image will be posted to the Docker Hub (we need to create a public repository on the docker hub before pushing the image). Ansible will then fetch this image from Docker Hub and deploy it across many machines.
- To build the docker image, a Docker File is used in which the script is written.

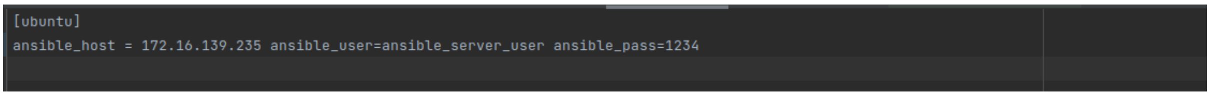


```
FROM openjdk:11
COPY ./target/SPE_Calculator-1.0-SNAPSHOT-jar-with-dependencies.jar ./

WORKDIR ./
CMD ["java", "-jar", "SPE_Calculator-1.0-SNAPSHOT-jar-with-dependencies.jar"]
```

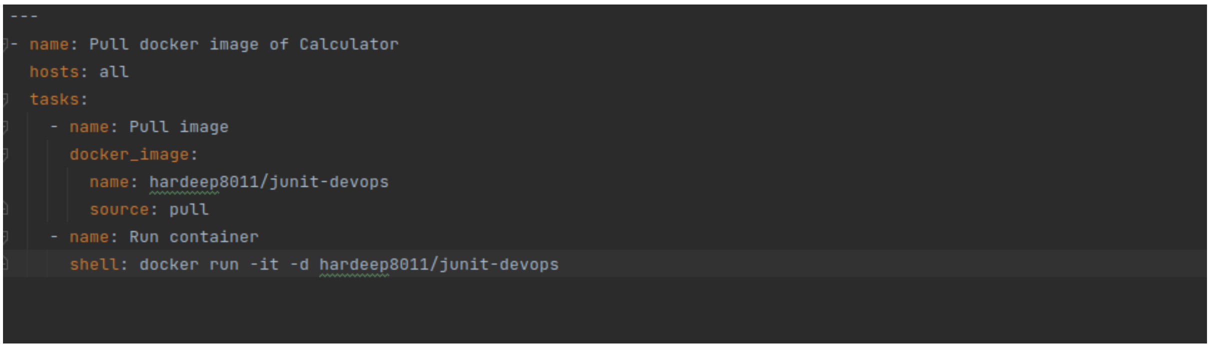
Deployment

- **Ansible** is a suite of software tools that enables infrastructure as code. It is open-source and the suite includes software provisioning, configuration management, and application deployment functionality.
- Steps to Install
    - ➢ Sudo apt install openssh-server
    - ➢ Ssh-keygen -t rsa
    - ➢ Ssh-copy-id <username>@<ip>
    - ➢ Sudo apt install ansible
- **Inventory file** - File that container data about ansible clients that are connected with the server. You can make a group of clients for developers , or testers to manage in a better way.



```
[ubuntu]
ansible_host = 172.16.139.235 ansible_user=ansible_server_user ansible_pass=1234
```

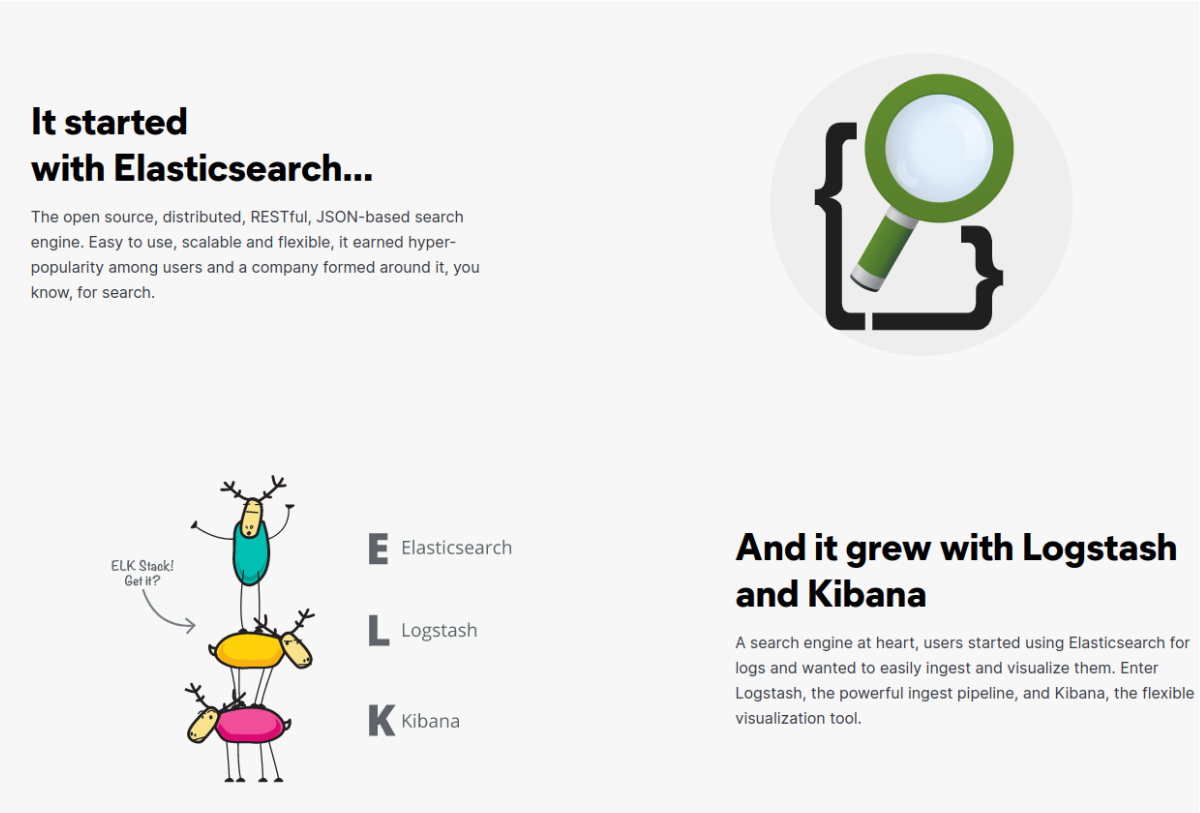
- **Playbook** - file which contains instructions or tasks to be executed in the client machine. It is written in YAML format.



```
---
- name: Pull docker image of Calculator
  hosts: all
  tasks:
    - name: Pull image
      docker_image:
        name: hardeep8011/junit-devops
        source: pull
    - name: Run container
      shell: docker run -it -d hardeep8011/junit-devops
```

Continuous Monitoring

- After completing the deployment, the following steps is used to monitor the system. Monitoring entails determining whether or not the project is performing as intended.
- "**ELK**" is the acronym for three open source projects: **Elasticsearch**, **Logstash**, and **Kibana**. Elasticsearch is a search and analytics engine. Logstash is a server-side data processing pipeline that ingests data from multiple sources simultaneously, transforms it, and then sends it to a "stash" like Elasticsearch. Kibana lets users visualize data with charts and graphs in Elasticsearch.



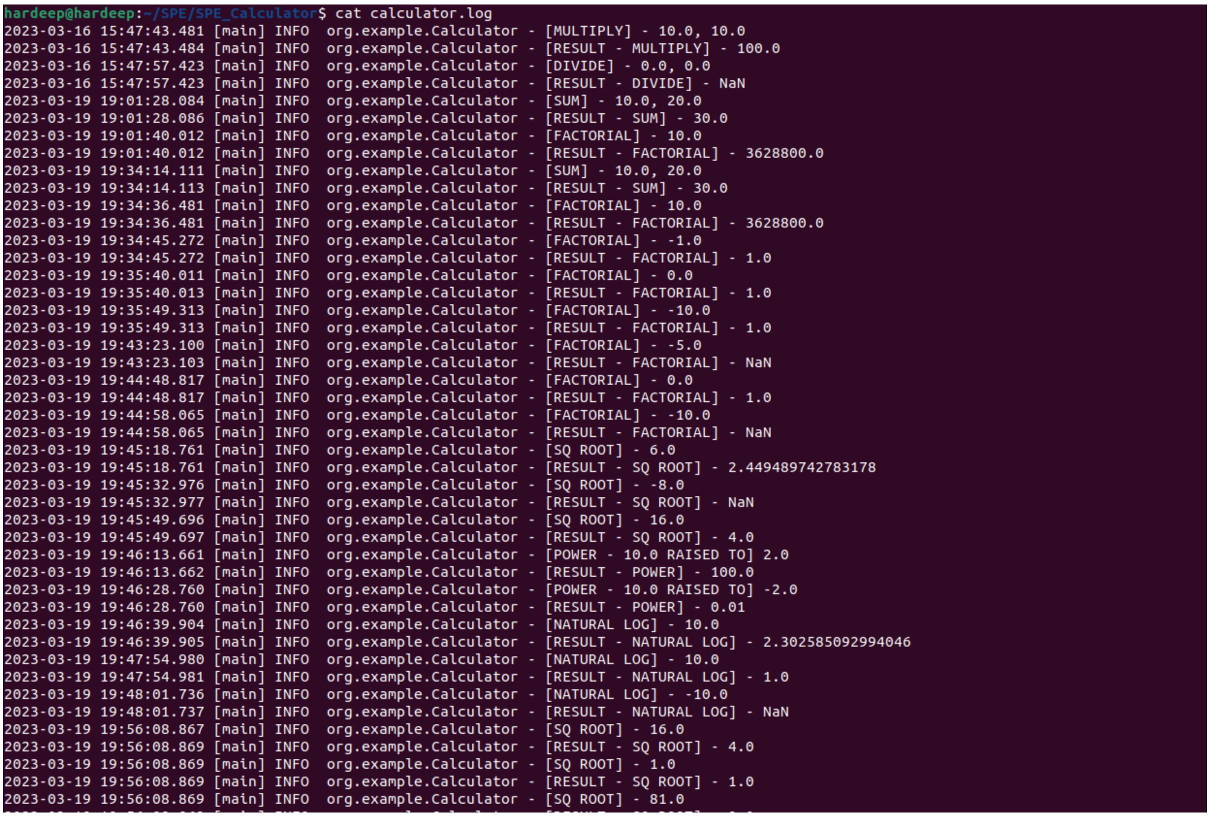
It started with Elasticsearch...

The open source, distributed, RESTful, JSON-based search engine. Easy to use, scalable and flexible, it earned hyper-popularity among users and a company formed around it, you know, for search.

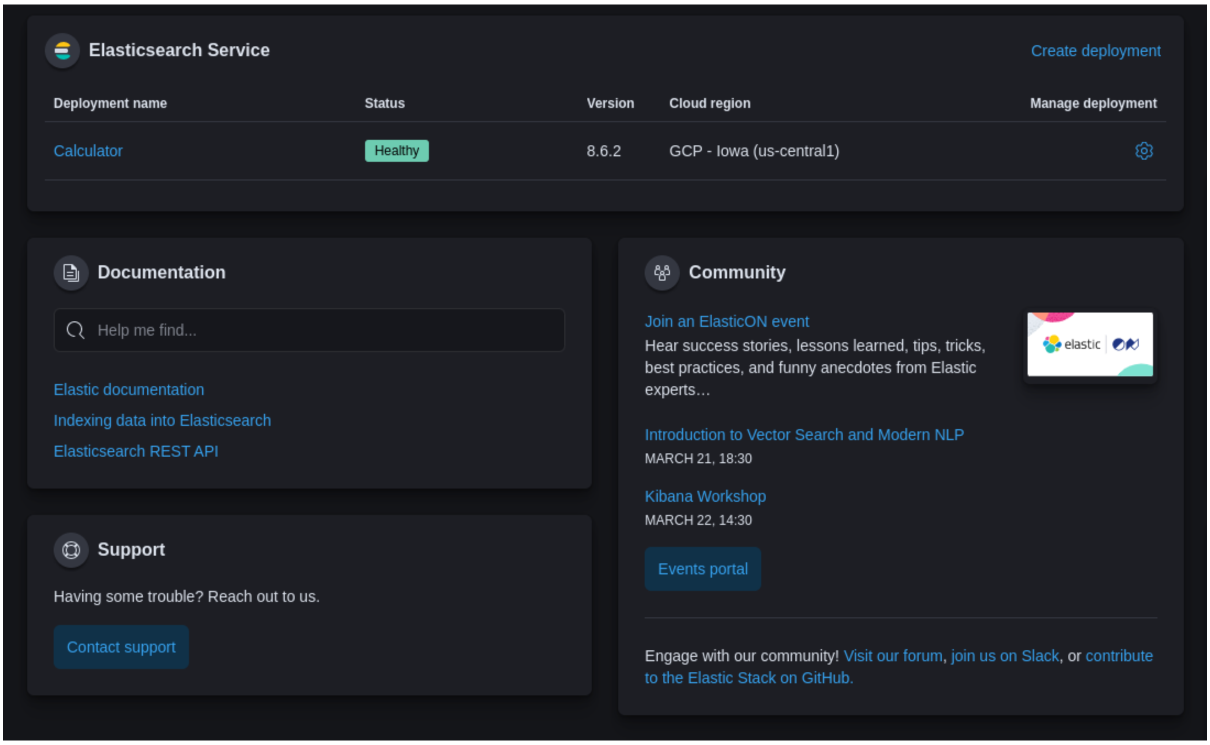
And it grew with Logstash and Kibana

A search engine at heart, users started using Elasticsearch for logs and wanted to easily ingest and visualize them. Enter Logstash, the powerful ingest pipeline, and Kibana, the flexible visualization tool.

The log file “**calculator.log**” generated by the program can be seen below:



```
hardeep@hardeep:~/SPE/SPE_Calculator$ cat calculator.log
2023-03-16 15:47:43.481 [main] INFO  org.example.Calculator - [MULTIPLY] - 10.0, 10.0
2023-03-16 15:47:43.484 [main] INFO  org.example.Calculator - [RESULT - MULTIPLY] - 100.0
2023-03-16 15:47:57.423 [main] INFO  org.example.Calculator - [DIVIDE] - 0.0, 0.0
2023-03-16 15:47:57.423 [main] INFO  org.example.Calculator - [RESULT - DIVIDE] - NaN
2023-03-19 19:01:28.084 [main] INFO  org.example.Calculator - [SUM] - 10.0, 20.0
2023-03-19 19:01:28.086 [main] INFO  org.example.Calculator - [RESULT - SUM] - 30.0
2023-03-19 19:01:40.012 [main] INFO  org.example.Calculator - [FACTORIAL] - 10.0
2023-03-19 19:01:40.012 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - 3628800.0
2023-03-19 19:34:14.111 [main] INFO  org.example.Calculator - [SUM] - 10.0, 20.0
2023-03-19 19:34:14.113 [main] INFO  org.example.Calculator - [RESULT - SUM] - 30.0
2023-03-19 19:34:36.481 [main] INFO  org.example.Calculator - [FACTORIAL] - 10.0
2023-03-19 19:34:36.481 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - 3628800.0
2023-03-19 19:34:45.272 [main] INFO  org.example.Calculator - [FACTORIAL] - -1.0
2023-03-19 19:34:45.272 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - 1.0
2023-03-19 19:35:40.011 [main] INFO  org.example.Calculator - [FACTORIAL] - 0.0
2023-03-19 19:35:40.013 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - 1.0
2023-03-19 19:35:49.313 [main] INFO  org.example.Calculator - [FACTORIAL] - -10.0
2023-03-19 19:35:49.313 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - 1.0
2023-03-19 19:43:23.100 [main] INFO  org.example.Calculator - [FACTORIAL] - -5.0
2023-03-19 19:43:23.103 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - NaN
2023-03-19 19:44:48.817 [main] INFO  org.example.Calculator - [FACTORIAL] - 0.0
2023-03-19 19:44:48.817 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - 1.0
2023-03-19 19:44:58.065 [main] INFO  org.example.Calculator - [FACTORIAL] - -10.0
2023-03-19 19:44:58.065 [main] INFO  org.example.Calculator - [RESULT - FACTORIAL] - NaN
2023-03-19 19:45:18.761 [main] INFO  org.example.Calculator - [SQ ROOT] - 6.0
2023-03-19 19:45:18.761 [main] INFO  org.example.Calculator - [RESULT - SQ ROOT] - 2.449489742783178
2023-03-19 19:45:32.976 [main] INFO  org.example.Calculator - [SQ ROOT] - -8.0
2023-03-19 19:45:32.977 [main] INFO  org.example.Calculator - [RESULT - SQ ROOT] - NaN
2023-03-19 19:45:49.696 [main] INFO  org.example.Calculator - [SQ ROOT] - 16.0
2023-03-19 19:45:49.697 [main] INFO  org.example.Calculator - [RESULT - SQ ROOT] - 4.0
2023-03-19 19:46:13.661 [main] INFO  org.example.Calculator - [POWER - 10.0 RAISED TO] 2.0
2023-03-19 19:46:13.662 [main] INFO  org.example.Calculator - [RESULT - POWER] - 100.0
2023-03-19 19:46:28.760 [main] INFO  org.example.Calculator - [POWER - 10.0 RAISED TO] -2.0
2023-03-19 19:46:28.760 [main] INFO  org.example.Calculator - [RESULT - POWER] - 0.01
2023-03-19 19:46:39.904 [main] INFO  org.example.Calculator - [NATURAL LOG] - 10.0
2023-03-19 19:46:39.905 [main] INFO  org.example.Calculator - [RESULT - NATURAL LOG] - 2.302585092994046
2023-03-19 19:47:54.980 [main] INFO  org.example.Calculator - [NATURAL LOG] - 10.0
2023-03-19 19:47:54.981 [main] INFO  org.example.Calculator - [RESULT - NATURAL LOG] - 1.0
2023-03-19 19:48:01.736 [main] INFO  org.example.Calculator - [NATURAL LOG] - -10.0
2023-03-19 19:48:01.737 [main] INFO  org.example.Calculator - [RESULT - NATURAL LOG] - NaN
2023-03-19 19:56:08.867 [main] INFO  org.example.Calculator - [SQ ROOT] - 16.0
2023-03-19 19:56:08.869 [main] INFO  org.example.Calculator - [RESULT - SQ ROOT] - 4.0
2023-03-19 19:56:08.869 [main] INFO  org.example.Calculator - [SQ ROOT] - 1.0
2023-03-19 19:56:08.869 [main] INFO  org.example.Calculator - [RESULT - SQ ROOT] - 1.0
2023-03-19 19:56:08.869 [main] INFO  org.example.Calculator - [SQ ROOT] - 81.0
```



Elasticsearch Service

Create deployment

Deployment name	Status	Version	Cloud region	Manage deployment
Calculator	Healthy	8.6.2	GCP - Iowa (us-central1)	⚙

Documentation

Help me find...

Elastic documentation
Indexing data into Elasticsearch
Elasticsearch REST API

Support

Having some trouble? Reach out to us.

Contact support

Community

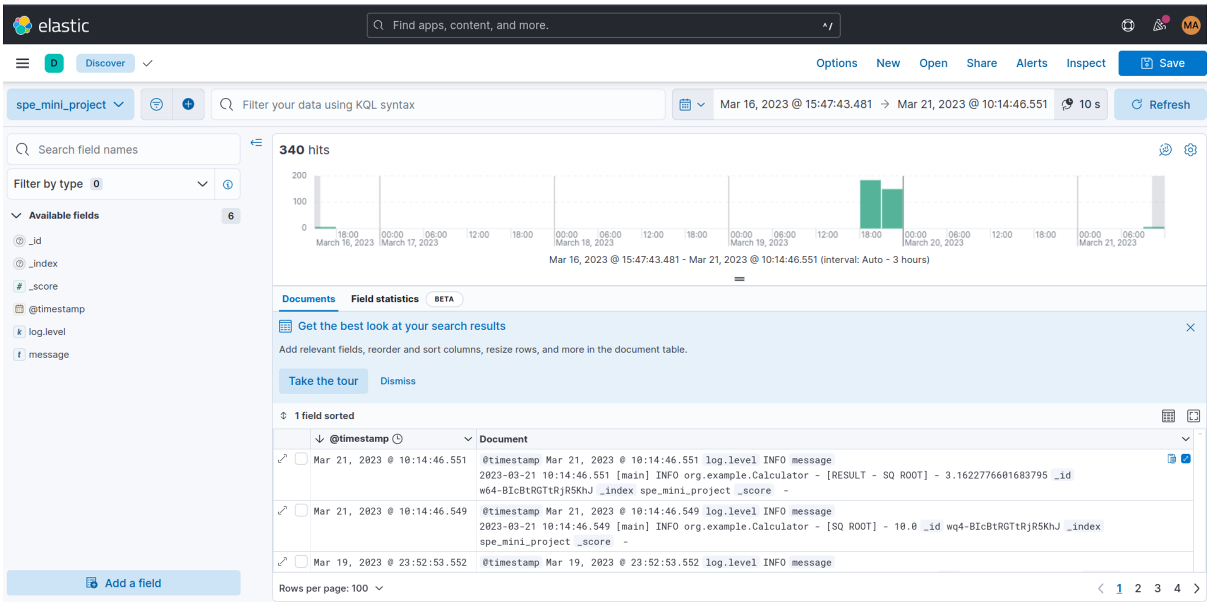
Join an ElasticON event
Hear success stories, lessons learned, tips, tricks, best practices, and funny anecdotes from Elastic experts…

Introduction to Vector Search and Modern NLP
MARCH 21, 18:30

Kibana Workshop
MARCH 22, 14:30

Events portal

Engage with our community! Visit our forum, join us on Slack, or contribute to the Elastic Stack on GitHub.



Get the best look at your search results

Add relevant fields, reorder and sort columns, resize rows, and more in the document table.

Take the tour Dismiss

1 field sorted

```
@timestamp                 Document
Mar 21, 2023 @ 10:14:46.551  @timestamp Mar 21, 2023 @ 10:14:46.551 log.level INFO message
                             2023-03-21 10:14:46.551 [main] INFO org.example.Calculator - [RESULT - SQ ROOT] - 3.1622776601683795 _id
                             w64-BIcBtRGTtRjR5KhJ _index spe_mini_project _score  -
Mar 21, 2023 @ 10:14:46.549  @timestamp Mar 21, 2023 @ 10:14:46.549 log.level INFO message
                             2023-03-21 10:14:46.549 [main] INFO org.example.Calculator - [SQ ROOT] - 10.0 _id wq4-BIcBtRGTtRjR5KhJ _index
                             spe_mini_project _score  -
Mar 19, 2023 @ 23:52:53.552  @timestamp Mar 19, 2023 @ 23:52:53.552 log.level INFO message
```

Rows per page: 100   1 2 3 4