

# **A REPORT ON**

## **Calculator along with DevOps Toolchain**



**International Institute Of Information  
Technology, Bangalore**

**Computer Science and Engineering**

**SUBMITTED TO:**  
***Prof. B.Thangaraju***

**SUBMITTED BY:**  
***Shashank Agarwal (MT2019100)***

## 1.0 Introduction :

We will use DevOps methodologies in order to facilitate continuous building, testing, deployment and monitoring. This is a JAVA based calculator project. This application provides user to add, subtract, divide or multiply two numbers.

## 2.0 Project Architecture

Language Used: -**JAVA**

IDE Used for JAVA: -**IntelliJ**

DevOps Tools Used:

**GitHub:** -SCM (Source Code Management)

**Maven:** -Build

**Jenkins:** - Continuous Integration

**Rundeck:** - Continuous Deployment

**ELK:** -Continuous Monitoring

## 3.0 Workflow:

### 3.1 Source Code Management (SCM):

GitHub is used for SCM (Source Code Management). It is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.

**GIT has various benefits as VCS(Version Control System) listed as follows :**

- 1)Revert the code files back to their previous state.
- 2)Recall and revert the entire project back to its previous state.
- 3)Compare code changes over specific durations of time.
- 4)Find who last modified a piece of code that might be causing an issue or a problem. Who introduced a particular issue and when.

### **Advantages of using git:**

- When multiple people collaborate on a project, it's hard to keep track revisions who changed what, when, and where those files are stored. GitHub takes care of this problem by keeping track of all the changes that have been pushed to the repository.

- Git branching model lets you have multiple local branches which are independent of each other. Having this also enables you to have friction-less context switching.

I pushed my whole project including Dockerfile to github.

### URL of the GIT repository:

GitHub: <https://github.com/shankei/calculator-devops.git>

### Steps used to add code to my git repository:

- git add .
- git remote add origin <https://github.com/shankei/calculator-devops.git>
- git commit -m "initial"
- git push origin master

The screenshot shows a GitHub repository page for 'Shankei initial'. At the top, it displays '2 commits', '1 branch', '0 packages', '0 releases', and '1 contributor'. Below this, there are buttons for 'Branch: master', 'New pull request', 'Create new file', 'Upload files', 'Find file', and 'Clone or download'. The main content area shows a list of files and folders: 'src', 'target', 'Dockerfile', 'README.md', 'calculator-devops.iml', and 'pom.xml'. Each item is listed with its commit hash 'initial' and the time '3 days ago'. Below the file list, the 'README.md' file is expanded, showing the title 'calculator-devops' and the description: 'This is a calculator based project built using various tools of devops, including jenkins, docker, Rundeck, ELK etc. Basic multiplication, subtraction, addition and division is handled by the java calculator project.'

### 3.2 Build :

Maven is used with Jenkins for building the application as well as for dependency management. Maven made the dependency management task quite easy.

For the build step we integrated Maven with Jenkins so that whenever the build step is triggered Jenkins first pulls the updated code from the Github repo and then invokes maven to build the application.

## Maven has the following advantages :

1. It makes a project easy to build.
2. It provides uniform build process (maven project can be shared by all the maven projects).
3. It provides project information (log document, cross referenced sources, mailing list, dependency list, unit test reports etc).
4. It is easy to migrate for new features of Maven.

Following is the pom.xml that we defined for our project.

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <project xmlns="http://maven.apache.org/POM/4.0.0"
3      xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4      xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
5      <modelVersion>4.0.0</modelVersion>
6      <groupId>org.example</groupId>
7      <artifactId>calculator-devops</artifactId>
8      <version>1.0-SNAPSHOT</version>
9      <build>
10         <plugins>
11             <plugin>
12                 <groupId>org.apache.maven.plugins</groupId>
13                 <artifactId>maven-assembly-plugin</artifactId>
14                 <executions>
15                     <execution>
16                         <phase>package</phase>
17                         <goals>
18                             <goal>single</goal>
19                         </goals>
20                         <configuration>
21                             <archive>
22                                 <manifest>
23                                     <mainClass>calculator.Calculator</mainClass>
24                                 </manifest>
25                             </archive>
26                             <descriptorRefs>
27                                 <descriptorRef>jar-with-dependencies</descriptorRef>
28                             </descriptorRefs>
29                         </configuration>
30                     </execution>
31                 </executions>
32             </plugin>
```

```
32         </plugin>
33     </plugins>
34 </build>
35
36 <dependencies>
37     <dependency>
38         <groupId>net.logstash.logback</groupId>
39         <artifactId>logstash-logback-encoder</artifactId>
40         <version>4.7</version>
41     </dependency>
42     <dependency>
43         <groupId>junit</groupId>
44         <artifactId>junit</artifactId>
45         <version>4.13</version>
46         <scope>test</scope>
47     </dependency>
48     <dependency>
49         <groupId>log4j</groupId>
50         <artifactId>log4j</artifactId>
51         <version>1.2.17</version>
52     </dependency>
53 </dependencies>
54 </project>
```

### 3.3 Continuous Integration:

In our project, we have used 'Jenkins' for continuous integration. Jenkins is an open source automation server written in Java. Jenkins helps to automate the non-human part of the software development process, with continuous integration and facilitating technical aspects of continuous delivery.

#### Advantages of using Jenkins:

- Easily Configurable. Jenkins can be easily modified and extended. It deploys code instantly, generates test reports. Jenkins can be configured according to the requirements for continuous integrations and continuous delivery.
- Most of the integration work is automated. Hence fewer integration issues. This saves both time and money over the lifespan of a project.

Jenkins is a highly extensible product whose functionality can be extended through the installation of plugins like Maven, Github, Docker, Rundeck are installed in this project.

#### Jenkins Pipeline:


Jenkins pipeline consist of three stages.


**In first stage, we are polling the project from github and building the project using maven.**


Enter an item name


Calculator-pipeline


» A job already exists with the name 'Calculator-pipeline'

 **Freestyle project**  
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

 **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.

OK

Write the script to run the pipeline.

**Pipeline**

Definition Pipeline script

Script

```
1 pipeline {
2   environment {
3     registry = "shashankagarwal2310/calculator-devops" // docker hub id
4     registryCredential = "docker-hub" // docker hub credential, stored in credential part
5     dockerImage = ''
6     dockerImageLatest = ''
7   }
8   agent any
9   stages {
10    stage('Cloning Git') {
11      steps {
12        git 'https://github.com/shankei/calculator-devops.git'
13      }
14    }
15    stage('Build Executable Jar'){
16      steps {
17        sh 'mvn clean test package'
18      }
19    }
20    stage('Building image') {
```

☒ Use Groovy Sandbox

[Pipeline Syntax](#)

Save Apply

**In second stage of the pipeline we are pushing the generated image to docker hub. Here, add docker hub credentials, i.e. username and password. In the “ID” field, the identity variable is the same used in the registryCredential above. i.e. docker-hub**

Global credentials (unrestricted) > shashankagarwal2310/\*\*\*\*\* (docker)

Scope Global (Jenkins, nodes, items, all child items, etc)

Username shashankagarwal2310

Password Concealed Change Password

ID docker-hub

Description docker

Save

**In third stage of the pipeline we are connecting jenkins to rundeck and providing the id of rundeck job.**

## Post-build Actions

Rundeck

Rundeck Instance:

Rundeck calculator devops

Job user (optional)

User password (optional)

Concealed

Change Password

Token (optional)

Concealed

Change Password

Job Identifier

b3fb877f-0220-4f69-b5d0-d5d2b6be5d03

Your Rundeck job is : [user:admin] b3fb877f-0220-4f69-b5d0-d5d2b6be5d03 [calculator-devops] Running docker image

Job options (optional)

Save

Apply

**Create another free style project. In the Post-build Actions, the credentials will be picked directly from the configuration stored in Rundeck. Here, paste the job ID in the job identifier.**

Job cache

☒ Enable Rundeck job cache

Cache statistics

Invalidate cache

Instances

Rundeck job cache configuration

Name

Rundeck calculator devops

URL

http://localhost:4440/


?

Login

admin

?

Password

 Concealed

Change Password

?

Auth Token

?

API Version

?

Test Connection

Delete Rundeck

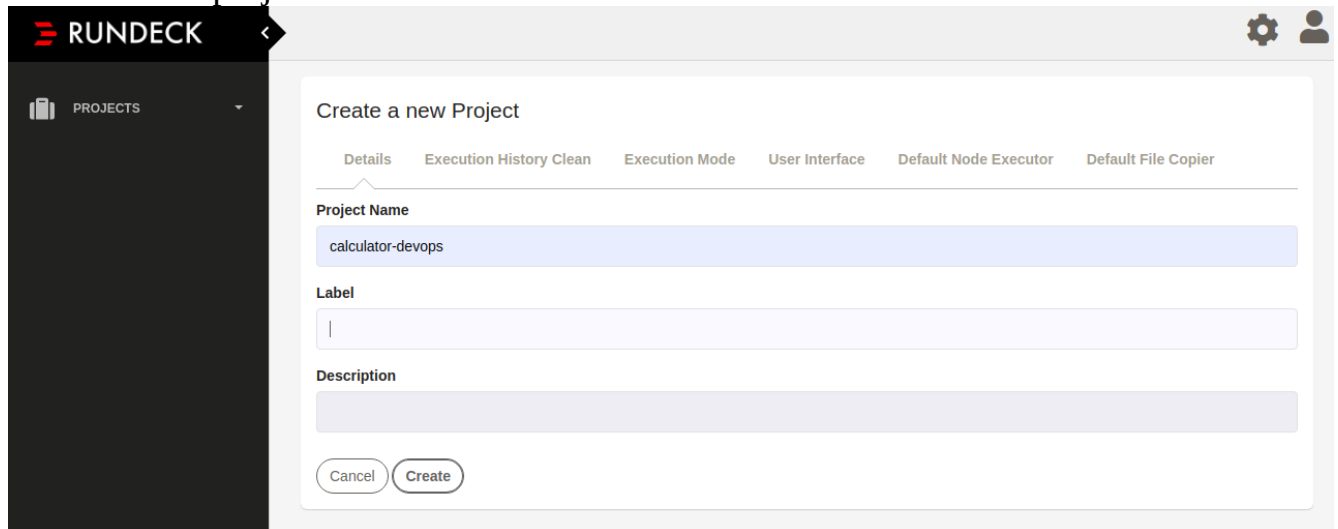
Save

Apply

### 3.4 Continuous Deployment

Rundeck is an orchestration tool for continuous deployment. It helps in standardising and automating complex release pipelines and deployment processes. It is an open source Web application written in Java and JavaScript. Rundeck source code is available on GitHub at <https://github.com/rundeck>. Many prominent multinational companies such as Walt Disney use Rundeck to carry out their operations. It is an easy-to-use tool that can be accessed via a Web interface, APIs or the command line.

Create a new project in Rundeck.



The screenshot shows the Rundeck web interface. On the left is a dark sidebar with the 'RUNDECK' logo and a 'PROJECTS' menu. The main area is titled 'Create a new Project' and has several tabs: 'Details' (selected), 'Execution History Clean', 'Execution Mode', 'User Interface', 'Default Node Executor', and 'Default File Copier'. Under the 'Details' tab, there are three input fields: 'Project Name' (containing 'calculator-devops'), 'Label' (empty), and 'Description' (empty). At the bottom of the form are 'Cancel' and 'Create' buttons.

Click on add a new node.



The screenshot shows the 'Node Sources' configuration page in Rundeck. At the top are tabs for 'Edit', 'Sources' (selected), 'Enhancers', and 'Configuration'. Below the tabs is a text block explaining that node sources are loaded in order and can use placeholders like `$(project.name)`. There is a list of sources, currently containing one item: '1. Local Provides the local node as the single resource'. This item has an 'Edit' button and a 'Delete' button with up/down arrows. At the bottom is a green button labeled 'Add a new Node Source +'.



Click on File to be added to create rundeck as a node.

## Add a new Node Source

+ AWS S3 remote model source

Obtain nodes information from a file located in a S3 bucket

+ Ansible Resource Model Source

Imports nodes from Ansible's inventory.

+ Directory

Scans a directory and loads all resource document files

+ File

Reads a file containing node definitions in a supported format

+ Local

Provides the local node as the single resource

+ Script


Run a script to produce resource model data


+ Stub

Generates stub nodes with stub node executors, useful for testing

+ URL Source

Specify path of rundeck to create xml file as devops.xml

1.  **Local** Provides the local node as the single resource

2.  **File** Reads a file containing node definitions in a supported format

Format

resourcexml

resourcexml

Format of the file. If unspecified, the format will be determined by the file extension.

File Path

/var/lib/rundeck

Path of the file

☒ **Generate**

Automatically generate the file if it is missing? Also creates missing directories.

☒ **Include Server Node**

Automatically include the server node in the generated file?

☐ **Require File Exists**

Require that the file exists

☒ **Writeable**

Allow this file to be editable.

Save

Cancel

Delete

↑

↓

we can see devops.xml has been created.

```
root@shashank:/var/lib/rundeck# cat devops.xml
<?xml version="1.0" encoding="UTF-8"?>

<project>
  <node name="localhost" description="Rundeck server node" tags="" hostname="localhost" osArch="amd64" osFamily="unix" osName="Linux" osVersion="5.3.0-42-generic" username="rundeck"/>
</project>
root@shashank:/var/lib/rundeck#
```

But here, if we need to run docker on Rundeck, we need to add Rundeck to the root group.

```
root@shashank:/var/lib/rundeck#
root@shashank:/var/lib/rundeck# usermod -aG docker rundeck
root@shashank:/var/lib/rundeck# sudo grep rundeck /etc/gshadow
docker:!:rundeck,jenkins
rundeck:!:
root@shashank:/var/lib/rundeck#
```

Specify a job in the new project in Rundeck.

Details

Workflow

Nodes

Schedule

Notifications

Other

Job Name

Running docker image

Group is a / separated path

Choose

Description

Edit

1

► The first line of the description will be shown in plain text, the rest will be rendered with Markdown. [More...](#)

Cancel

Save

Specify the command for the job.

**Workflow** If a step fails:

☒ Stop at the failed step. ☐ Run remaining steps before failing.

**Strategy:** **Node First** ▼

Execute all steps on a node before proceeding to the next node.

[Explain >](#)

---

**Global Log Filters** [+ add](#)

---

1.

**Command**

---

**Step Label**

[Cancel](#) [Save](#)

[Cancel](#) [Save](#)

Job can be seen executed.

**RUNDECK** calculator-devops

1 All Jobs [Search...](#)

[Expand All](#) [Collapse All](#) [Job Actions](#)

▶ Running docker image

**Activity for Jobs**

3 Executions [any time](#) [Save Filter...](#) [Bulk Delete](#)

✓	05/07/2020 5:49 PM <small>Last Thursday at 5:49 PM</small>	1 ok	1 seconds	by admin	Running docker image	#39
✓	05/07/2020 5:45 PM <small>Last Thursday at 5:45 PM</small>	1 ok	1 seconds	by admin	Running docker image	#38
✓	05/07/2020 5:42 PM <small>Last Thursday at 5:42 PM</small>	1 ok	3 seconds	by admin	Running docker image	#37

Rundeck 3.2.5-20200403

calculator-devops

Running docker image

b3fb877f-0220-4f69-b5d0-d5d2b6be5d03

Action

Definition

Follow execution

Nodes

Run Job Now

Stats

Activity

3 EXECUTIONS

100% SUCCESS RATE

1s AVG DURATION

## Pipeline in function

### Stage View



After following the steps, we can see the pipeline completely functional, following all the steps of a DevOps lifecycle.

```
root@shashank:/var/lib/rundeck# docker run -i -t shashankagarwal2310/calculator-devops
Calculator-DevOps, Choose to perform operation
Press 1 to Add
Press 2 to Subtract
Press 3 to Multiply
Press 4 to Divide
Press any other key to exit
Enter your choice:
1
Enter the first number : 5
Enter the second number : 4
[INFO ] 2020-05-10 13:48:29,001 method:calculator.Calculator.add(Calculator.java:63)
Adding two numbers 5.0 and 4.0
[INFO ] 2020-05-10 13:48:29,050 method:calculator.Calculator.add(Calculator.java:65)
Result of addition is 9.0
Addition result is : 9.0
Calculator-DevOps, Choose to perform operation
Press 1 to Add
Press 2 to Subtract
Press 3 to Multiply
Press 4 to Divide
Press any other key to exit
Enter your choice:
4
Enter the first number : 1
Enter the second number : 0
[INFO ] 2020-05-10 13:48:38,505 method:calculator.Calculator.divide(Calculator.java:87)
Dividing two numbers 1.0 and 0.0
[ERROR] 2020-05-10 13:48:38,507 method:calculator.Calculator.divide(Calculator.java:101)
Number cannot be divided by zero Case of Positive Infinity 1.0/0.0
[INFO ] 2020-05-10 13:48:38,508 method:calculator.Calculator.divide(Calculator.java:103)
Result of dividing is Infinity
Multiplication result is : Infinity
Calculator-DevOps, Choose to perform operation
Press 1 to Add
Press 2 to Subtract
Press 3 to Multiply
Press 4 to Divide
```

**This image shows how we call the container in which we deployed the image and access the container**

### 3.6 Continous Monitoring

Continuous Monitoring is the formal process of defining an agency's IT systems, categorizing each of these systems by the level of risk, application of the controls, continuous monitoring of the applied controls, and the assessment of the effectiveness of these controls against security threats.

In our project, we have used 'ELK Stack' for continuous monitoring. "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. Finally with the Kibana, we can generate visualisations using the log data.

```

input
{
  file {
    path => "/var/log/DevOps-Calculator/calculator.log"
    type => "logs"
    start_position => "beginning"
  }
}

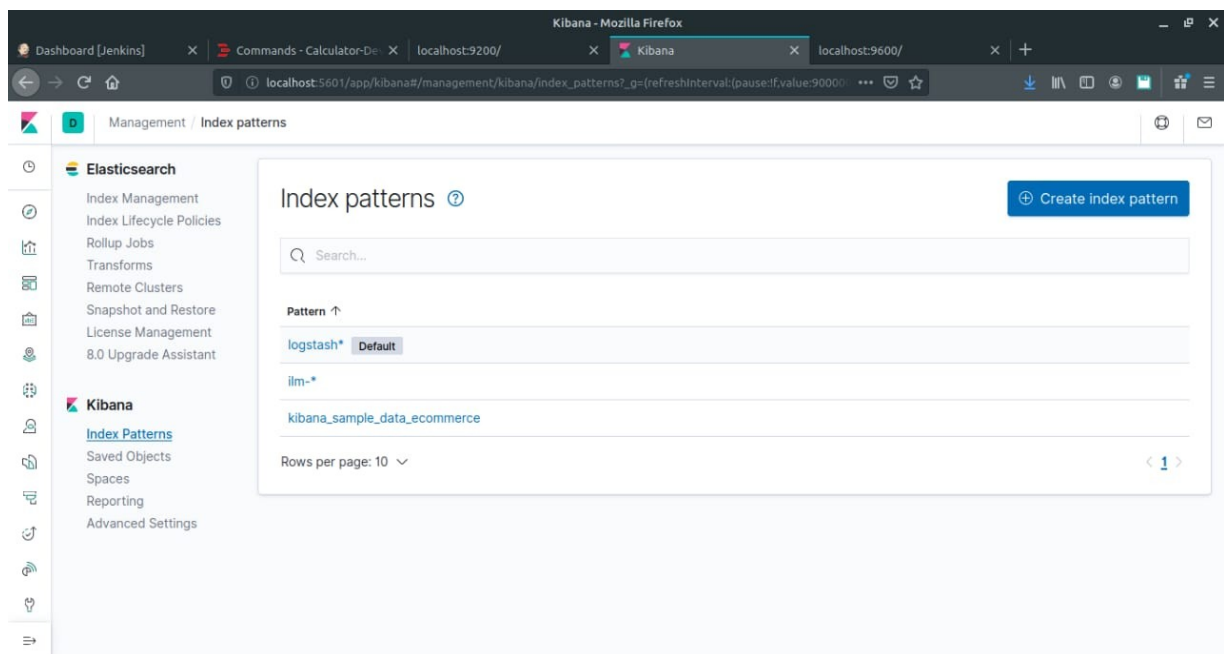
output
{
  elasticsearch {
    hosts => ["localhost:9200"]
  }

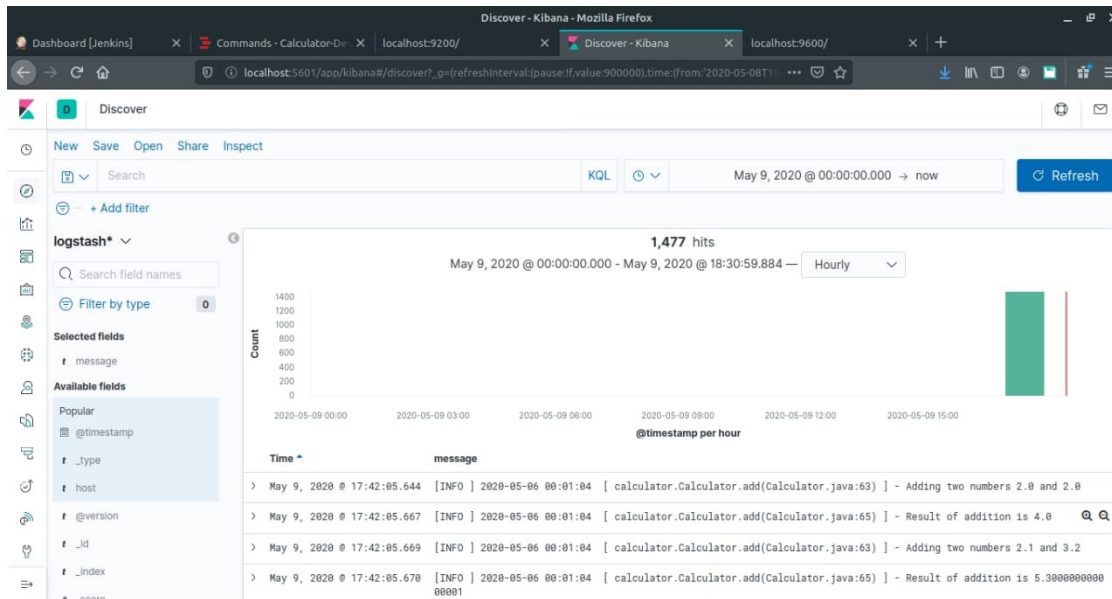
  stdout {
    codec => dots
  }
}

```

## Kibana Dashboard

In the management, toggle to Index Patterns under Kibana and create a new Create index pattern. Search for logstash-\* and select the log file of the calculator program fetched by ElasticSearch. Follow to the next slide for the dashboard generated by Kibana.





**Logs based on the @timestamp for the calculator program showing the INFO messages**

## 4.0 References

- [1]<https://medium.com/>
- [2]<https://www.youtube.com/>
- [3]<https://www.docker.com/>
- [4]<https://guides.github.com/>
- [5]<https://stackoverflow.com/>
- [6]<https://docs.docker.com/get-started/>
- [7]<https://jenkins.io/doc/tutorials/>
- [8]<https://maven.apache.org/guides/getting-started>
- [9]<https://docs.rundeck.com/docs/tutorials/>
- [10]<https://www.elastic.co/guide/en/elasticsearch/reference/current/gettingstarted.html>