International Institute of Information Technology (IIIT), Bangalore

Calculator with Devops Tool Chain

Mentor
Prof.Thangaraju B
Professor,IIIT-Bangalore

MUKESH KUMAR PILANIYA
MTECH 1ST YEAR
MT2019068

Title: - Calculator with Devops Tool Chain

DockerHub Profile: https://hub.docker.com/r/pilaniya1337/calculator
GitHub Profile: https://github.com/mukeshpilaniya/calculator

Required Tools: -

- Git (source code management)
- Docker (container node)
- Eclipse /IntelliJ (Project IDE)
- Jenkins (Continuous Integration: git, Continuous testing: Junit)
- Maven (Continuous Build)
- Rundeck (continuous deployment)
- ELK (elastic search, Logstash, Kibana: continuous monitoring)

Installing Git: - *sudo apt-get install git*

Installing Docker: -

1. *sudo apt-get update*
2. *sudo apt install apt-transport-https ca-certificates curl software-properties-common*
3. *curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add*
4. *sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu bionic stable"*
5. *sudo apt update*
6. *sudo apt install docker-ce*
7. *sudo usermod -aG docker ${USER}*

Installing Eclipse IDE:- https://www.eclipse.org/downloads/

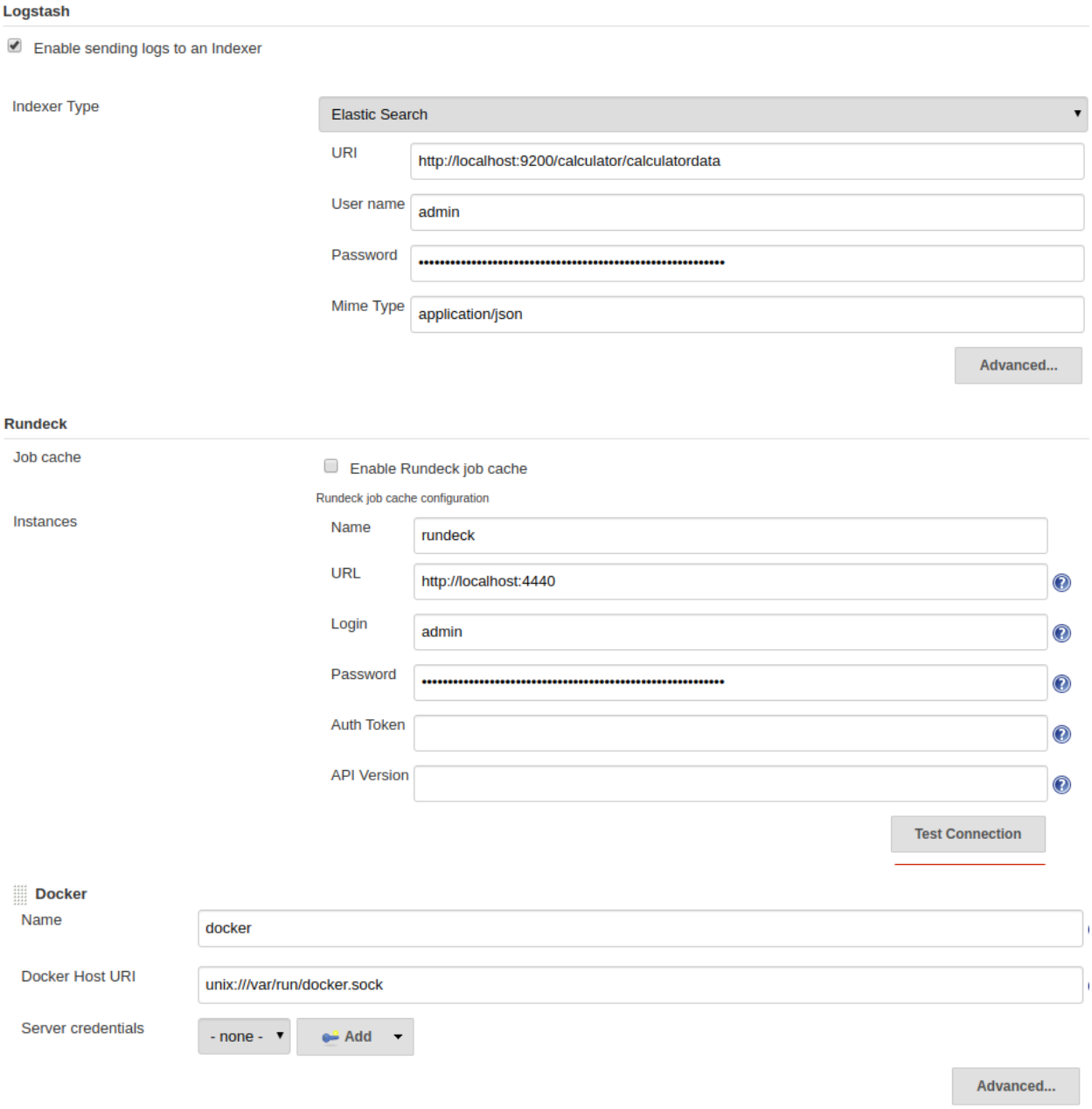
Installing Jenkins: -

1. Download the war file
   http://mirrors.jenkins.io/war-stable/latest/jenkins.war
2. Run the war file
   java -jar jenkins.war
3. Got to url http://localhost:8080

Jenkins Plugins: -

1. Rundeck plugin
2. Docker plugin
3. Logstash plugin

Configure plugin: -Logstash plugin automatically create calculator index in elasticsearch



Installing Maven: -

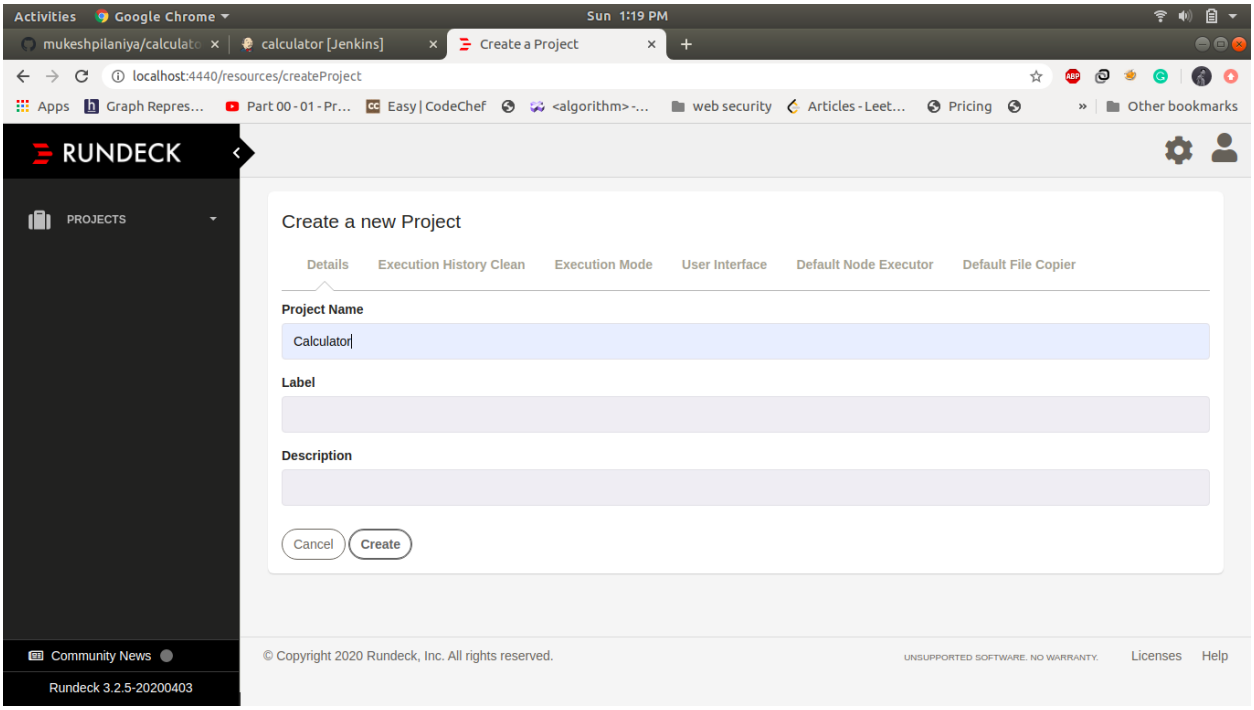
1. *sudo apt install maven*
2. *mvn -version*

Installing Rundeck: -

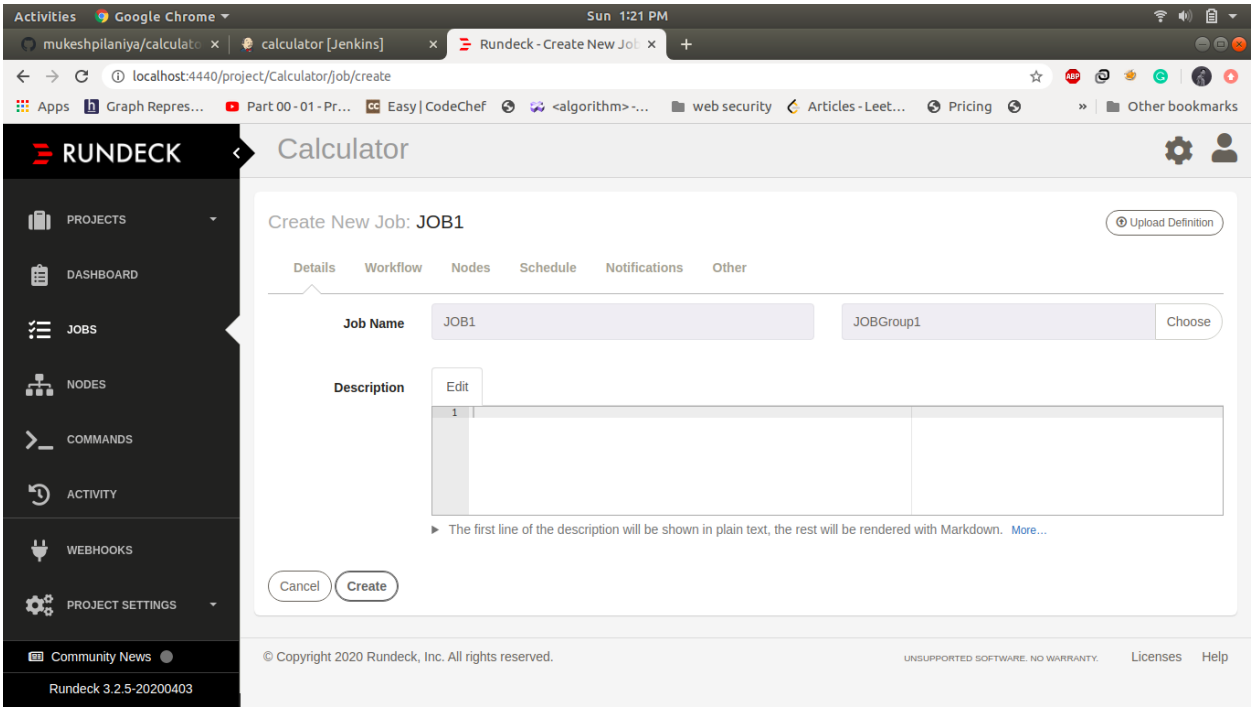
1. Download rundeck from
   https://download.rundeck.org/deb/rundeck_3.2.6.20200427-1_all.deb
2. install using dpkg
   dpkg -i rundeck_3.2.6.20200427-1_all.deb
3. Rundeck start and stop command
   sudo service rundeckd start
   sudo service rundeckd stop
4. Default address, username, password
   Default address: http://localhost:4440
   default username and password: admin
5. Allow rundeck to execute sudo commands on system terminal without password enter super user mode
   1. open file visudo
      sudo visudo
   2. *Add following lines at end of the file*
      rundeck ALL=(ALL) NOPASSWD: ALL
      Localhost ALL=(ALL) NOPASSWD: ALL

Create a new Project and job in Rundeck:-

1. Go to the url http://localhost:4440 and create a new project name as calculator and save it.

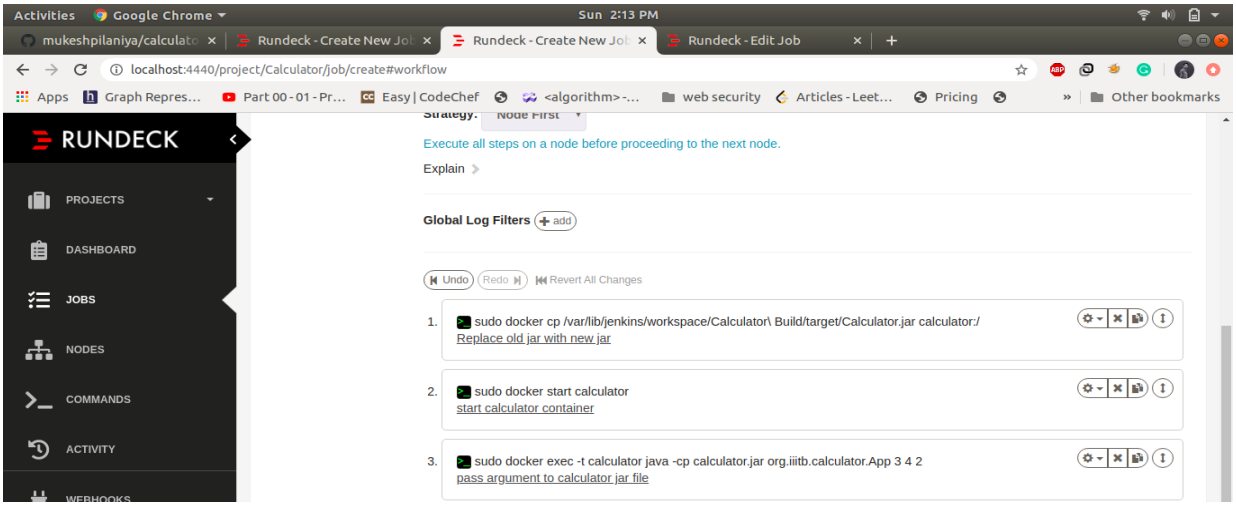


2. Create a new job in the same project, enter jobname and job group

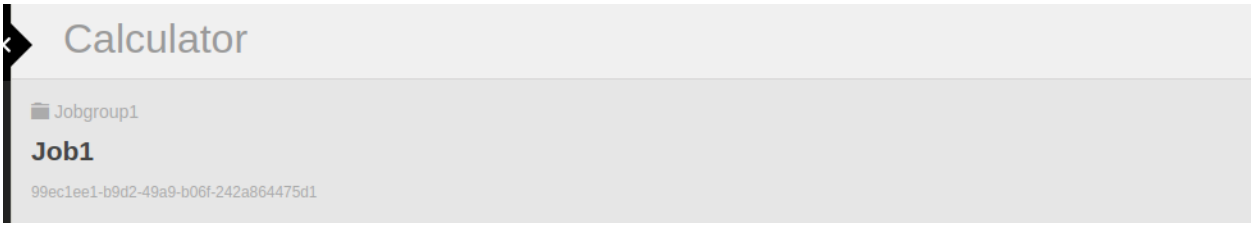


3. Navigate to workflow section and add the following commands to execute on registered node
   3.1 command 1 to replace old jar file with new jar file
   sudo docker cp /var/lib/jenkins/workspace/Calculator\ Build/target/Calculator.jar calculator:/
   3.2 Command 2 to start calculator container
   sudo docker start calculator
   3.3 Command 3 to pass argument to calculator jar file
   sudo docker exec -t calculator java -cp calculator.jar org.iiitb.calculator.App 3 4 2

4. Under Nodes select execute locally because all these commands will be executed on the local system.



5. Make note of UUID for future reference and save the job.



ELK Installing: -

Step 1 installing elasticsearch: -

1. Set java 8 as default java versionAllow rundeck to execute sudo commands on system terminal without password
2. enter super user mode
3. open file visudo
4. sudo visudo
5. Add following lines at end of the file
6. rundeck ALL=(ALL) NOPASSWD: ALL
7. Localhost ALL=(ALL) NOPASSWD: ALL
8. download Elasticsearch followed by public signing key
   sudo wget -qO - https://artifacts.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
9. install the apt-transport-https package
   sudo apt-get install apt-transport-https
10. Add the repository:
    echo "deb https://artifacts.elastic.co/packages/6.x/apt stable main" | sudo tee -a /etc/apt/sources.list.d/elastic-6.x.list
11. Update the repo list and install the package:
    sudo apt-get update
    sudo apt-get install elasticsearch
12. Update the repo list and install the package
    sudo vim /etc/elasticsearch/elasticsearch.yml
13. Uncomment “network.host” and “http.port”. Following configuration should be added:
    network.host: localhost
    http.port: 9200
14. Start ElasticSearch
    sudo systemctl start elasticsearch.service

Step 2 installing kibana:-

1. Let’s start installing Kibana now and modify Kibana settings
   sudo apt-get install kibana
   sudo vim /etc/kibana/kibana.yml

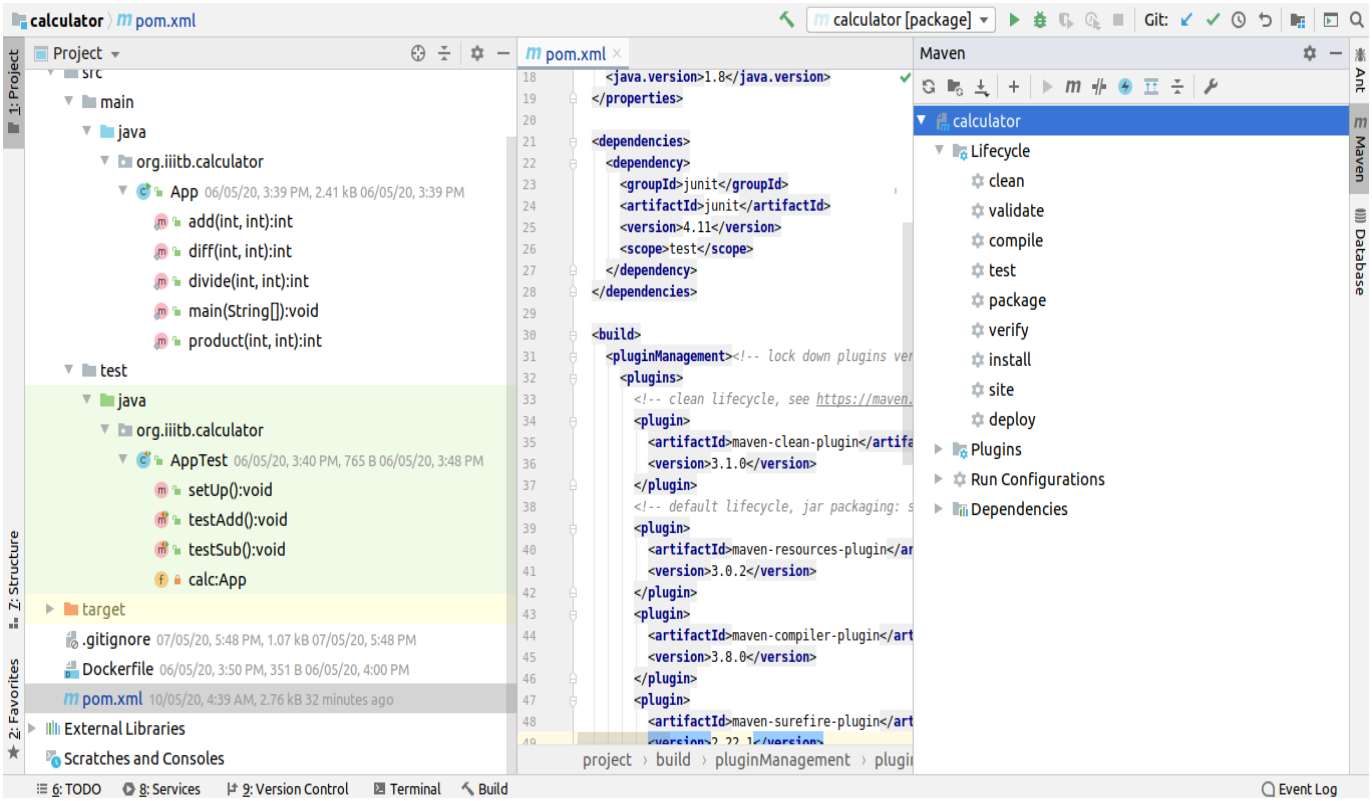
2. Uncomment following lines:
   server.port: 5601
   server.host: "localhost"
   elasticsearch.url: http://localhost:9200
3. start Kibana service
   sudo systemctl start kibana.service
4. Goto http://localhost:5601

Step 3 installing logstash: -

1. sudo apt-get install logstash
2. sudo service logstash start
3. Got to http://localhost:4440
   http://localhost:9200

SDLC: -

- **Development Phase: -** The development of this project is happened in java and it is a maven-based project. The src/main/java directory contains the project source code and the src/test/java directory contains the test cases like unit testing.



The next step is executing these commands: -
mvn clean- command attempt to clean target folder files that are generated during the build by maven
mvn package- command convert the entire maven project into an executable jar package

Pom xml file: - To perform unit testing we have to add Junit dependency and maven-jar-plugin for creating a package. It will create a package with a name calculator.

```
</plugin>
<plugin>
  <artifactId>maven-jar-plugin</artifactId>
  <configuration>
    <finalName>calculator</finalName>
  </configuration>
  <version>3.0.2</version>
</plugin>
```

```
<dependencies>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.11</version>
    <scope>test</scope>
  </dependency>
</dependencies>
```

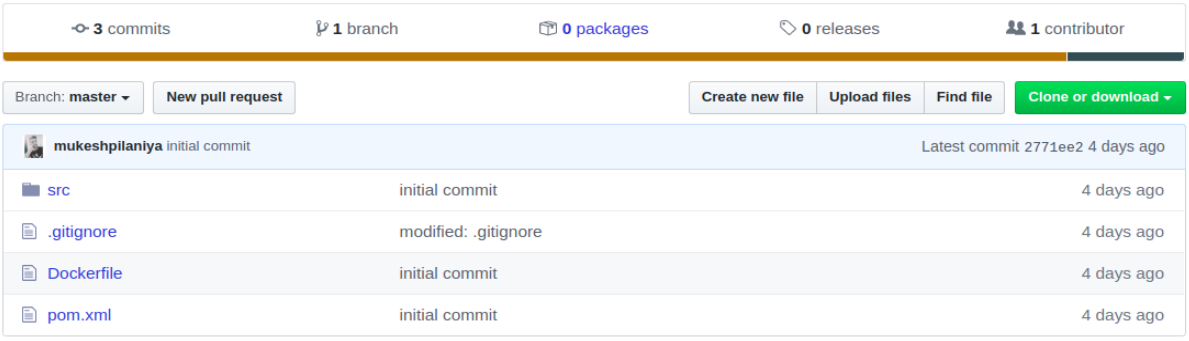
After executing these commands, a target folder is generated automatically which contains our artifacts file calculator.jar. To test this artifact, copy this artifacts file and run the below command in the same directory.

java -cp calculator.jar org.iiitb.calculator.App
org.iiitb.calculator is a package name and *App* is a class name where calculator methods are defined.
Docker file: - create a 'Dockerfile' under project level (at same level of pom.xml)

```
# Start with a base image containing Java runtime
FROM openjdk:8
# Add Maintainer Info
LABEL maintainer="github.com/mukeshpilaniya"
# Make port 8080 available to the world outside this container
EXPOSE 8080
# Add the application's jar to the container
ADD /target/calculator.jar calculator.jar
# Run the jar file
ENTRYPOINT ["java","-cp","calculator.jar","org.iiitb.calculator.App"]
```

The next step is, create a repository(calculator) in github and push project code into calculator repository. The following set of commands will push the code into github repository.

git init
git remote add origin "https://github.com/mukeshpilaniya/calculator.git"
git add .
git commit -m "initial commit"
git push origin master.



Build a docker Image: - Enter the following command in the terminal with the home directory of project
sudo docker build -t calculator_image .

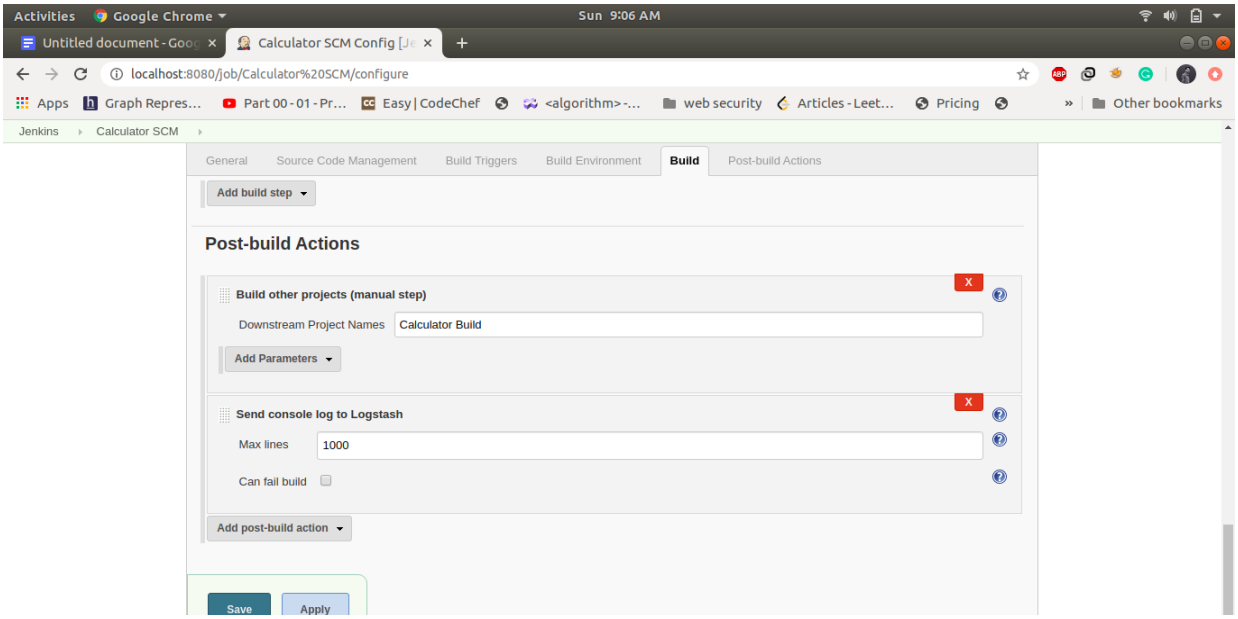
This command will create project specific docker image(calculator_image), now create a container of this image using the following command
sudo run --name calculator_container -d calculator_image

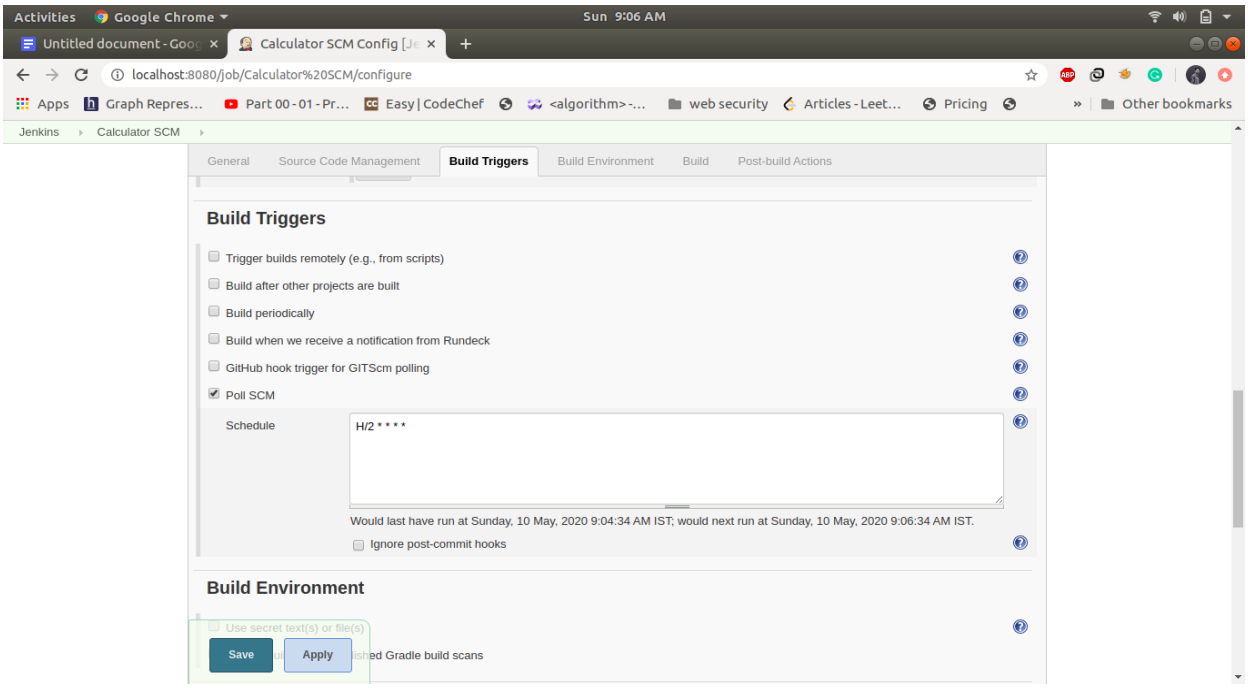
Software Development life cycle: -The whole project is developed following the DevOps model and using various tools. The software development Life Cycle of this project includes six stages

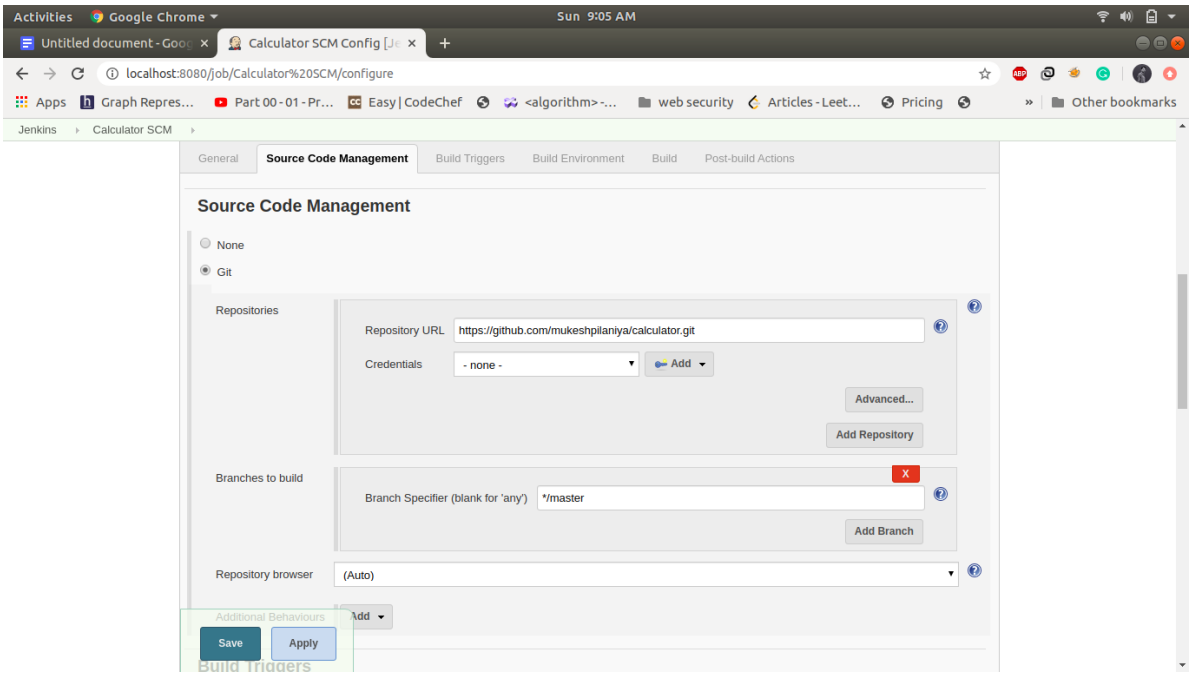
(1) Source Control Management (git)
(2) Code Building (maven)
(3) Code testing (Junit)
(4) Build and Publish Docker image (Docker)
(5) Deploying (Rundeck)
(6) Monitoring (ELK)

Setup jenkins Pipeline: -

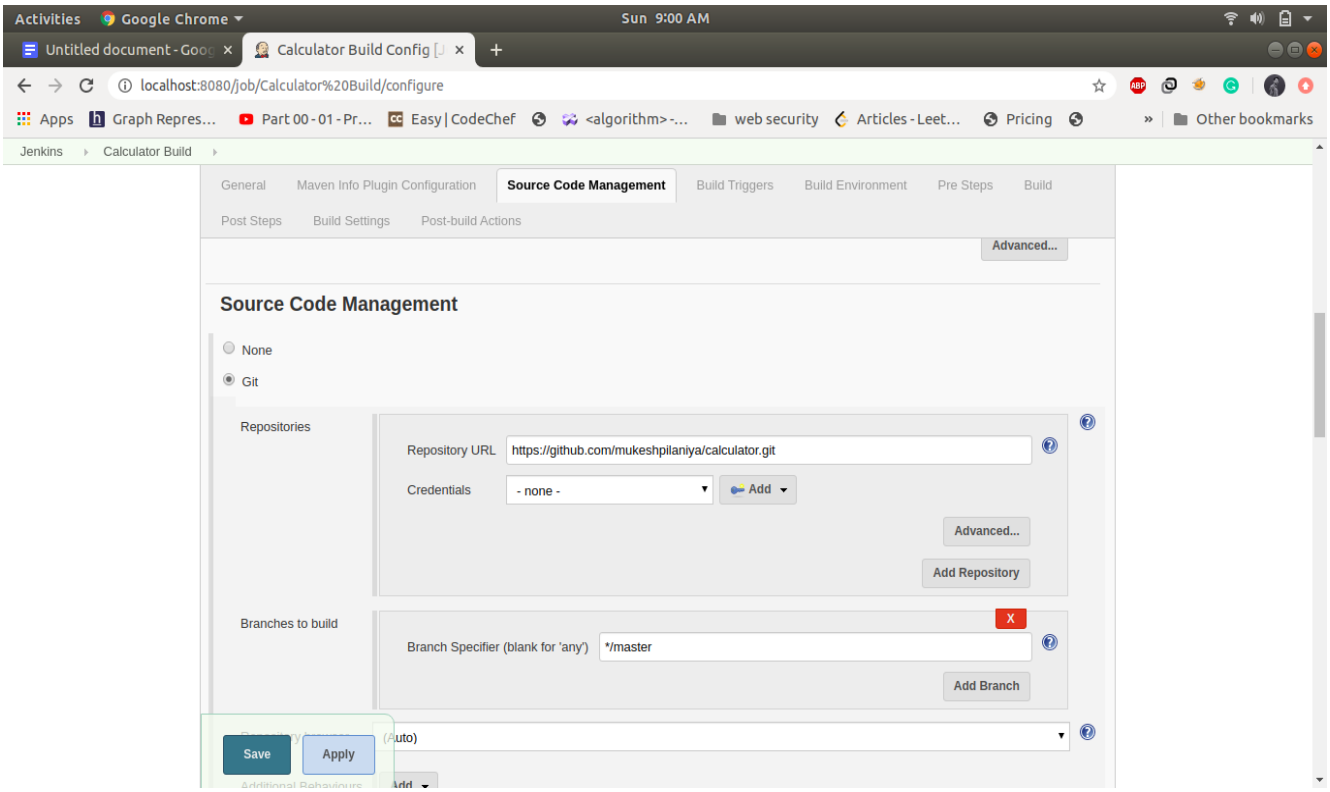
1. **Job1: Calculator SCM: -** SCM stands for source code management and used for managing the source code of the application, for this project source code is stored in a git repository hosted on GitHub at mukeshpilaniya/calculator. Here we are using pollSCM which checks the git repository after an interval and if there is a change in the code it triggers the pipeline otherwise it doesn't do anything. Create a FreeStyle project names as calculator SCM and following is the configuration in this step

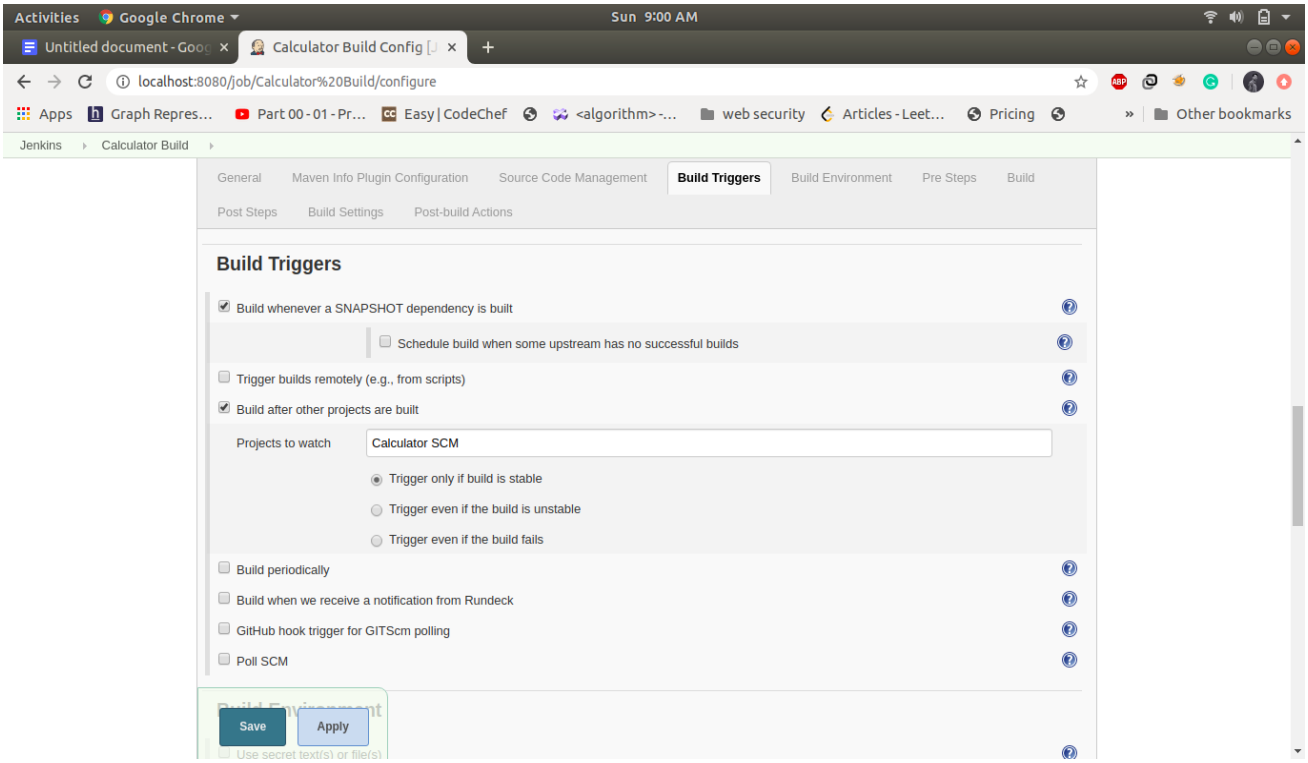


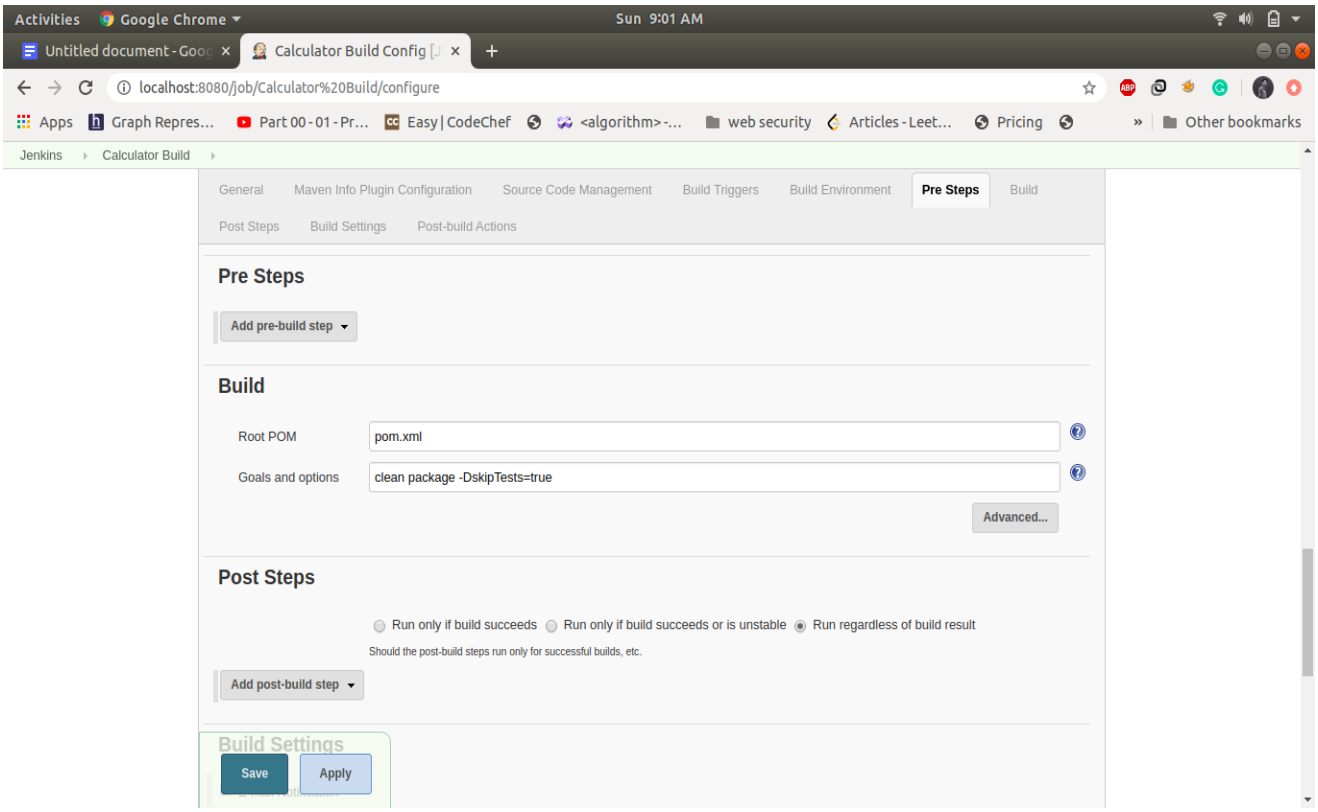


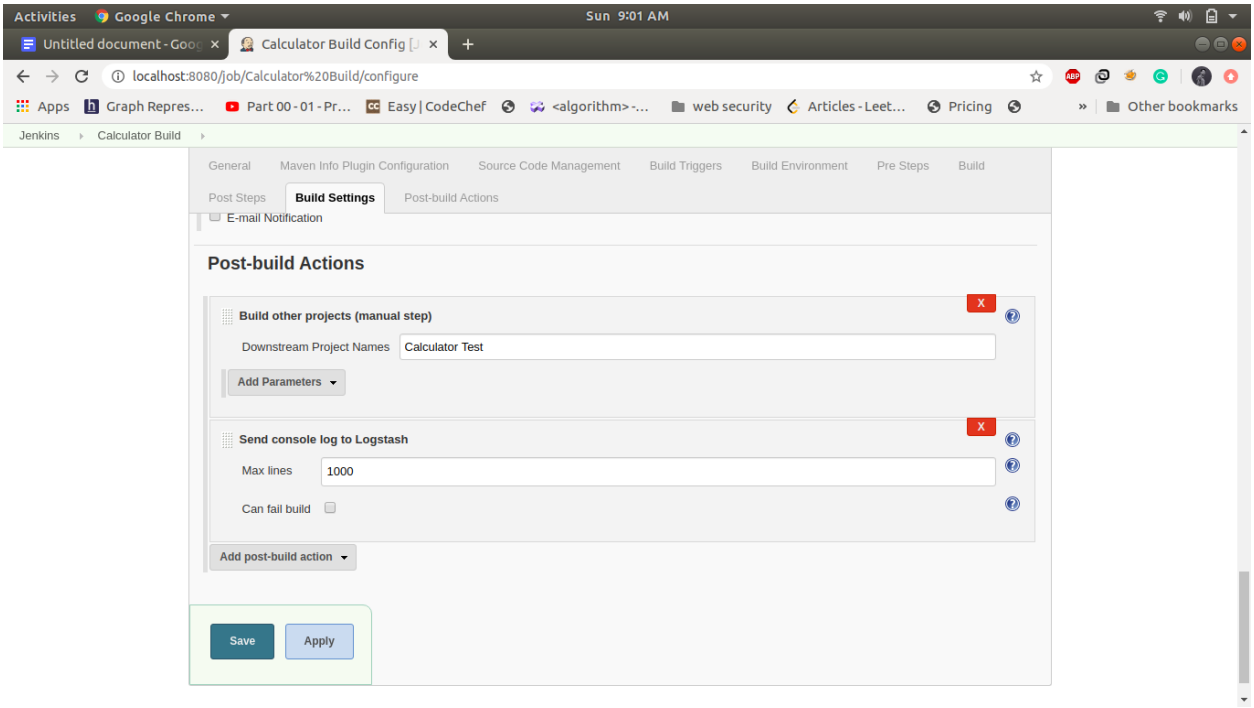


2. **Job2: Calculator Build: -** This step build triggers automatically when the first job is finished and it will build a jar file in the jenkins working directory if the build is successful it will automatically trigger the calculator Test job. Create a maven project name as calculator Build and following is the configuration in this step.

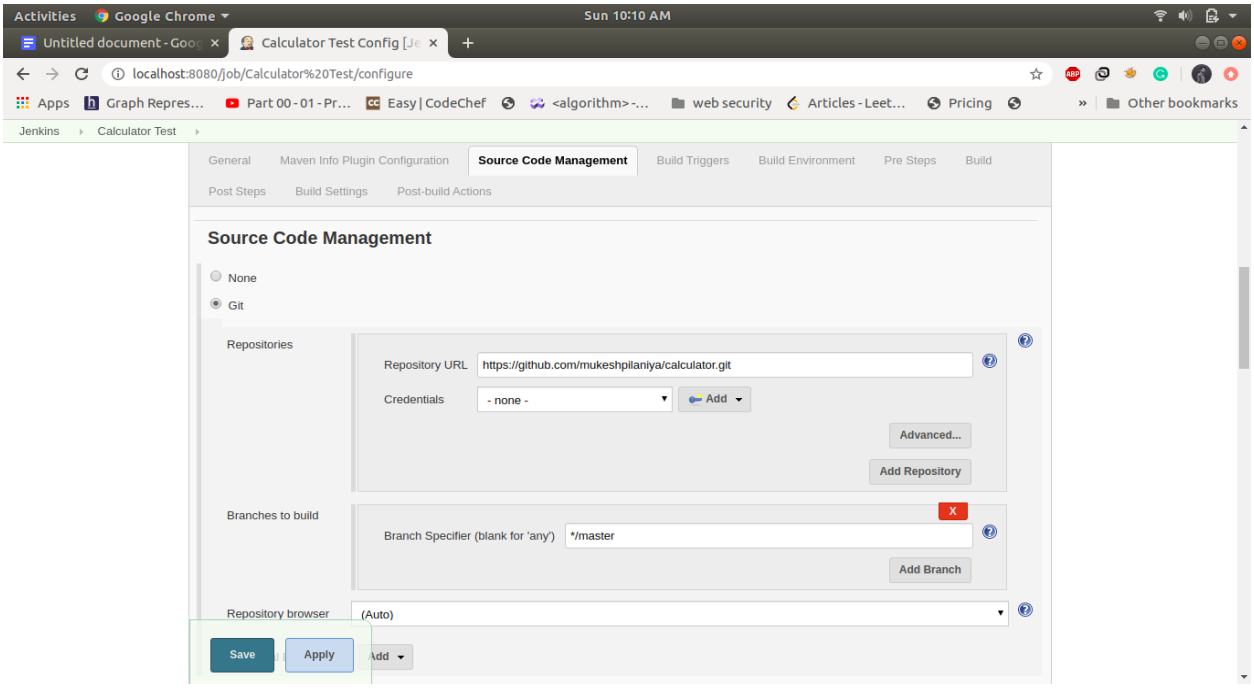


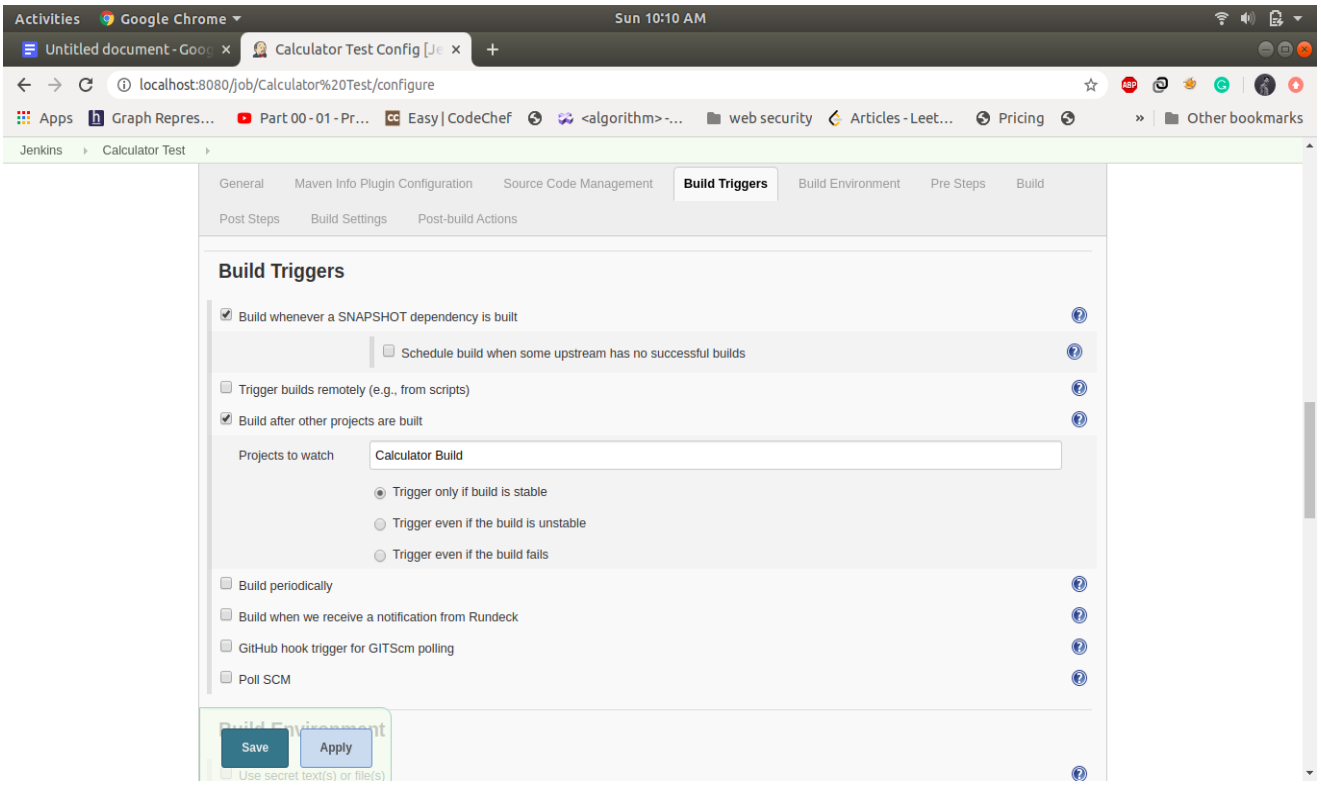

Activities
Google Chrome
Sun 9:00 AM
Untitled document - Goog
Calculator Build Config [J
localhost:8080/job/Calculator%20Build/configure
Apps
Graph Repres...
Part 00 - 01 - Pr...
Easy | CodeChef
<algorithm>-...
web security
Articles - Leet...
Pricing
Other bookmarks
Jenkins
Calculator Build
General
Maven Info Plugin Configuration
Source Code Management
Build Triggers
Build Environment
Pre Steps
Build
Post Steps
Build Settings
Post-build Actions
Build Triggers
Build whenever a SNAPSHOT dependency is built
Schedule build when some upstream has no successful builds
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Projects to watch
Calculator SCM
Trigger only if build is stable
Trigger even if the build is unstable
Trigger even if the build fails
Build periodically
Build when we receive a notification from Rundeck
GitHub hook trigger for GITScm polling
Poll SCM
Save
Apply

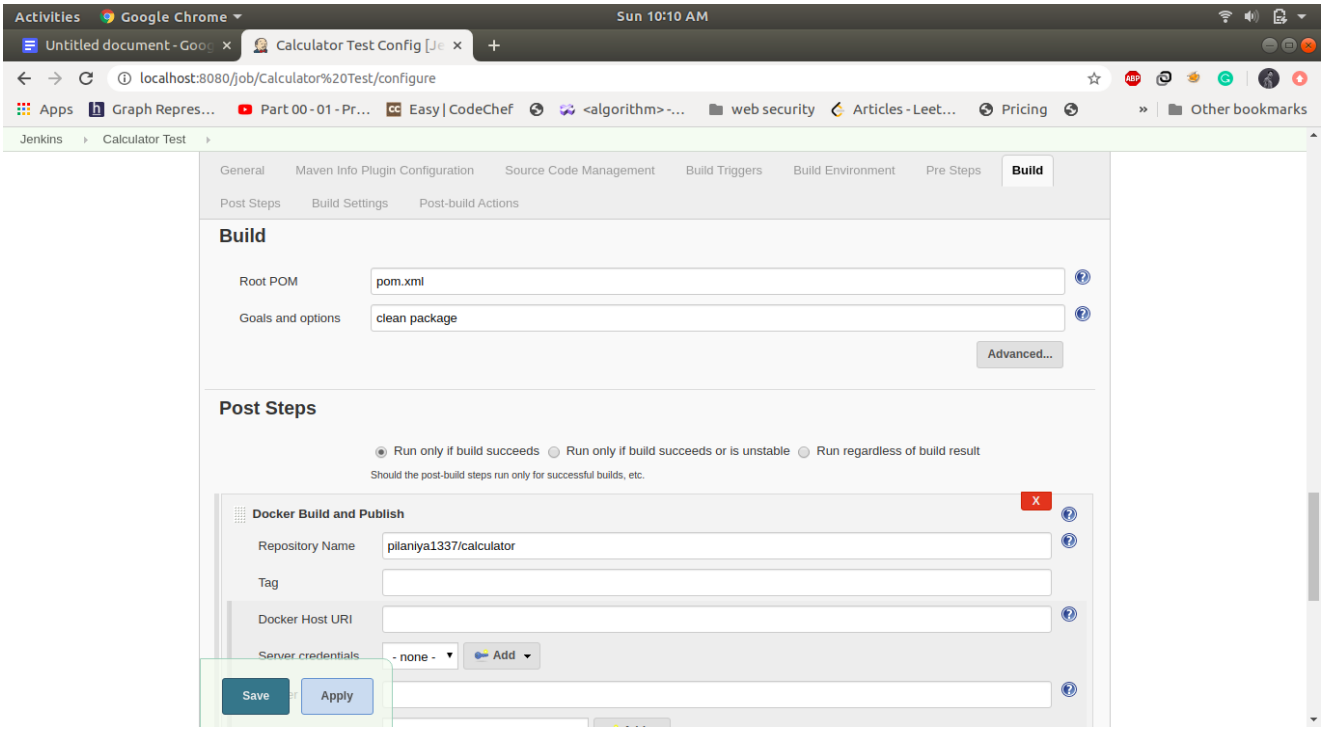

Activities
Google Chrome
Sun 9:01 AM
Untitled document - Goog
Calculator Build Config [J
localhost:8080/job/Calculator%20Build/configure
Apps
Graph Repres...
Part 00 - 01 - Pr...
Easy | CodeChef
<algorithm>-...
web security
Articles - Leet...
Pricing
Other bookmarks
Jenkins
Calculator Build
General
Maven Info Plugin Configuration
Source Code Management
Build Triggers
Build Environment
Pre Steps
Build
Post Steps
Build Settings
Post-build Actions
Pre Steps
Add pre-build step
Build
Root POM
pom.xml
Goals and options
clean package -DskipTests=true
Advanced...
Post Steps
Run only if build succeeds
Run only if build succeeds or is unstable
Run regardless of build result
Should the post-build steps run only for successful builds, etc.
Add post-build step
Build Settings
Save
Apply

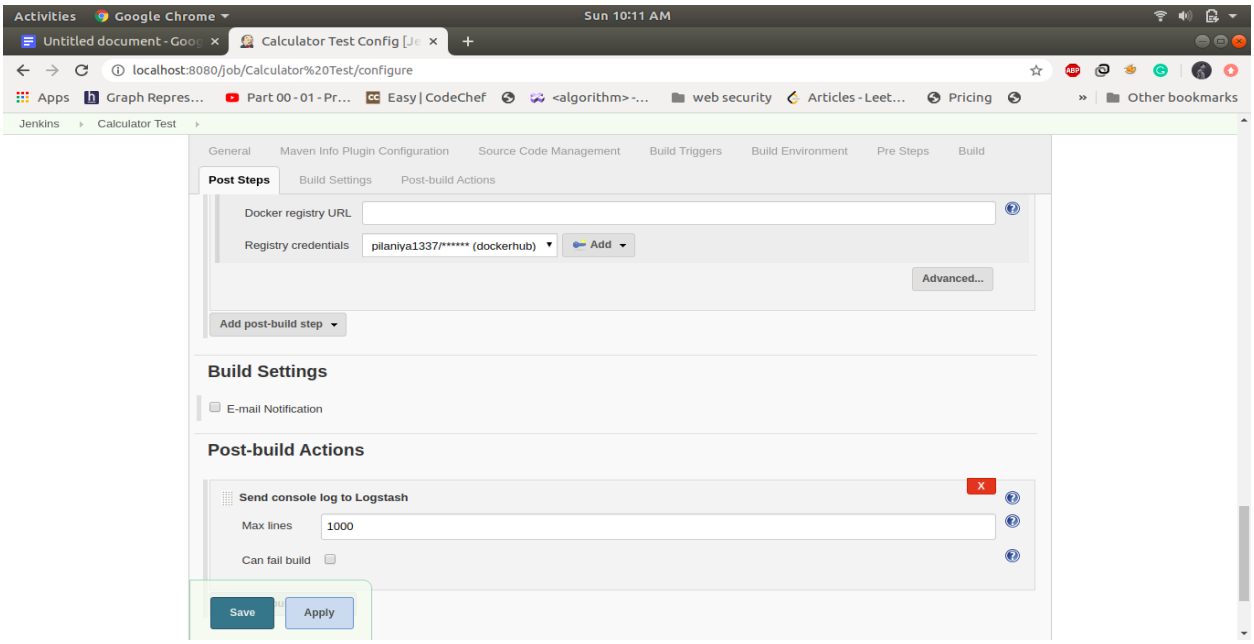


3. **Job3: CalculatorTest:** - If the build is successful then the calculator test job will automatically be triggered. it will build the docker image and push it into dockerhub. This job will run test cases and send the control to the calculator deploy. Create a maven project name as calculator Test and Configuration of this step is as follows.
   Create a Maven Project and name its calculator Test



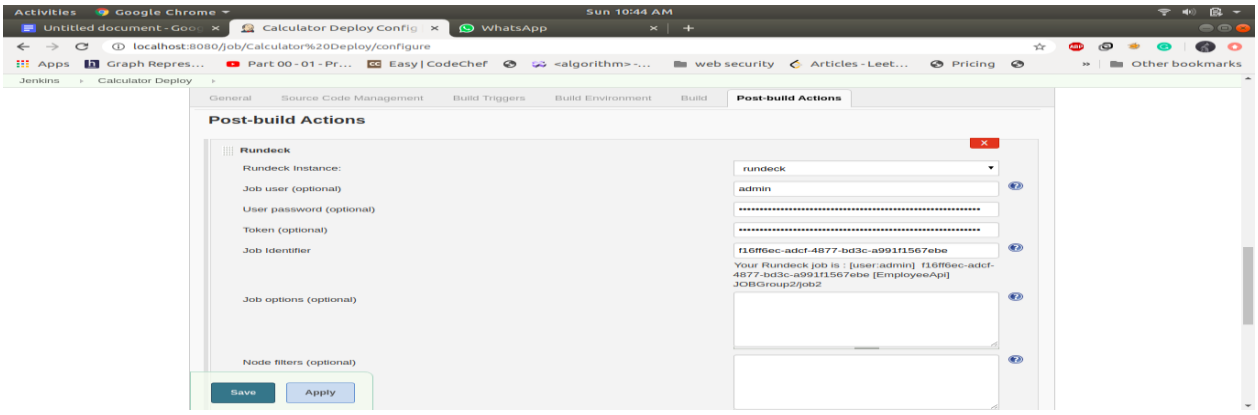

Activities
Google Chrome
Sun 10:10 AM
Untitled document - Goog
Calculator Test Config [Je
localhost:8080/job/Calculator%20Test/configure
Apps
Graph Repres...
Part 00 - 01 - Pr...
Easy | CodeChef
<algorithm>-...
web security
Articles - Leet...
Pricing
Other bookmarks
Jenkins
Calculator Test
General
Maven Info Plugin Configuration
Source Code Management
Build Triggers
Build Environment
Pre Steps
Build
Post Steps
Build Settings
Post-build Actions
Build Triggers
Build whenever a SNAPSHOT dependency is built
Schedule build when some upstream has no successful builds
Trigger builds remotely (e.g., from scripts)
Build after other projects are built
Projects to watch
Calculator Build
Trigger only if build is stable
Trigger even if the build is unstable
Trigger even if the build fails
Build periodically
Build when we receive a notification from Rundeck
GitHub hook trigger for GITScm polling
Poll SCM
Save
Apply


Activities
Google Chrome
Sun 10:10 AM
Untitled document - Goog
Calculator Test Config [Je
localhost:8080/job/Calculator%20Test/configure
Apps
Graph Repres...
Part 00 - 01 - Pr...
Easy | CodeChef
<algorithm>-...
web security
Articles - Leet...
Pricing
Other bookmarks
Jenkins
Calculator Test
General
Maven Info Plugin Configuration
Source Code Management
Build Triggers
Build Environment
Pre Steps
Build
Post Steps
Build Settings
Post-build Actions
Build
Root POM
pom.xml
Goals and options
clean package
Advanced...
Post Steps
Run only if build succeeds
Run only if build succeeds or is unstable
Run regardless of build result
Should the post-build steps run only for successful builds, etc.
Docker Build and Publish
Repository Name
pilaniya1337/calculator
Tag
Docker Host URI
Server credentials
- none -
Add
Save
Apply



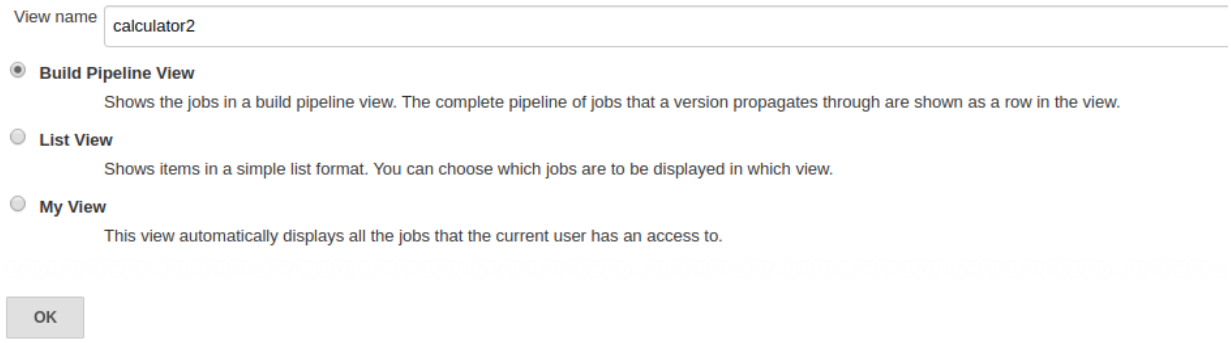
4. **Job4: calculator deploy:** -This job will automatically trigger if the calculator test job is successfully executed and it will trigger the specified rundeck job.
   Create a FreeStyle project and name it as a calculator deploy. Configuration of this step is as follows.

   Rundeck instance: rundeck
   Copy the job UUID in job identifier id in Post Build Actions → Rundeck

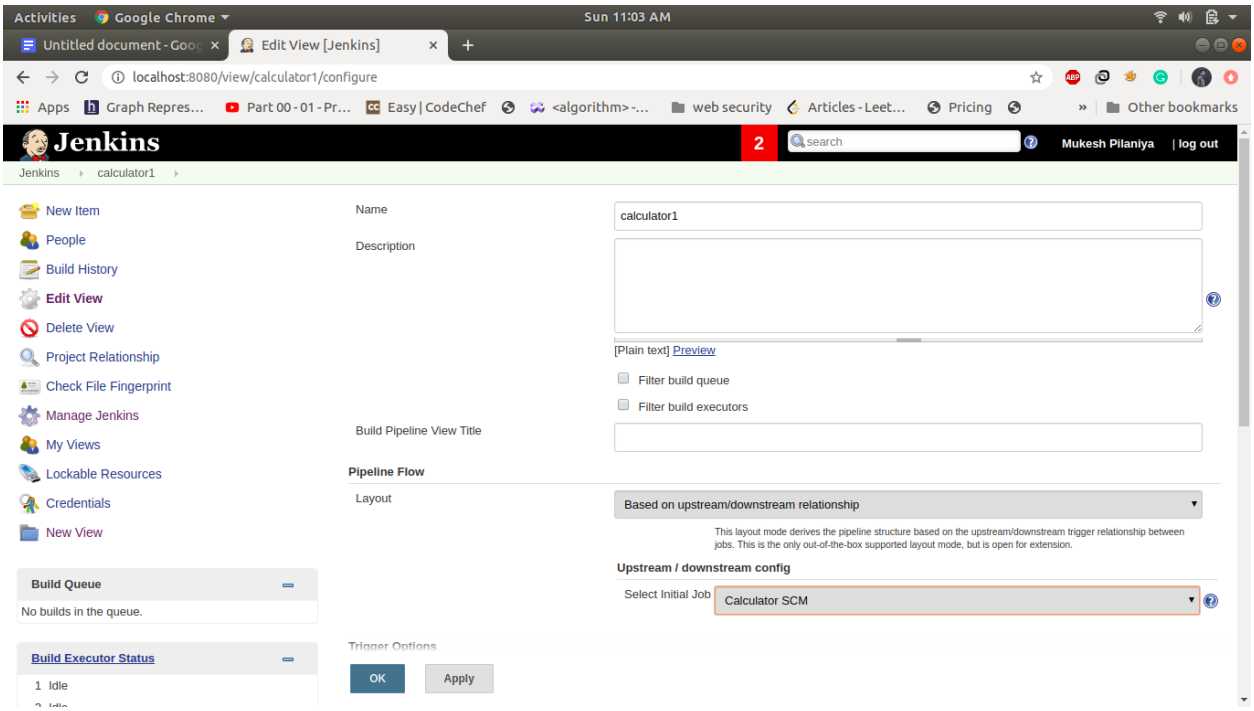


Create Pipeline View: -

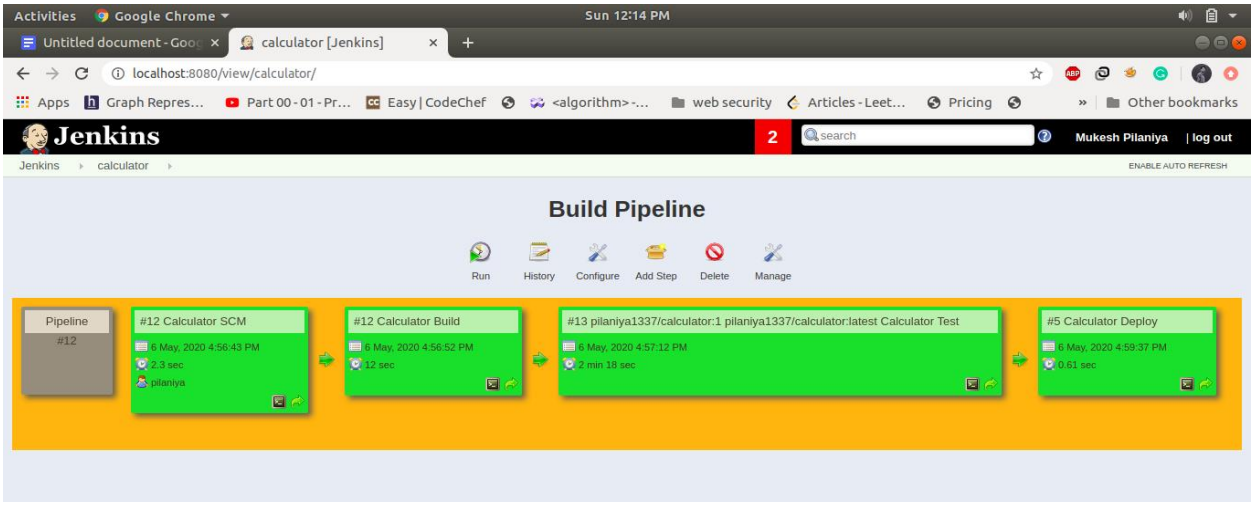
Click on + icon and do following configuration



Click Ok and select calculator SCM as Initial Job under Pipeline Flow. Then click save.

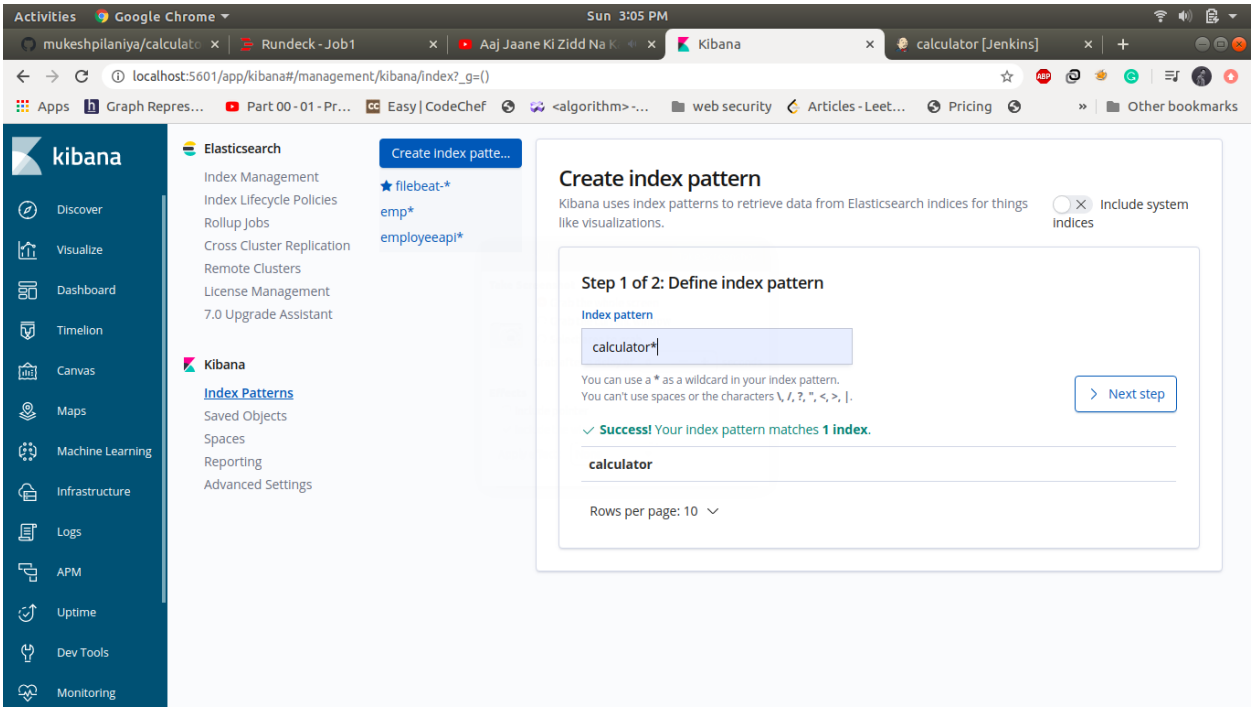


Pipeline View Layout: -

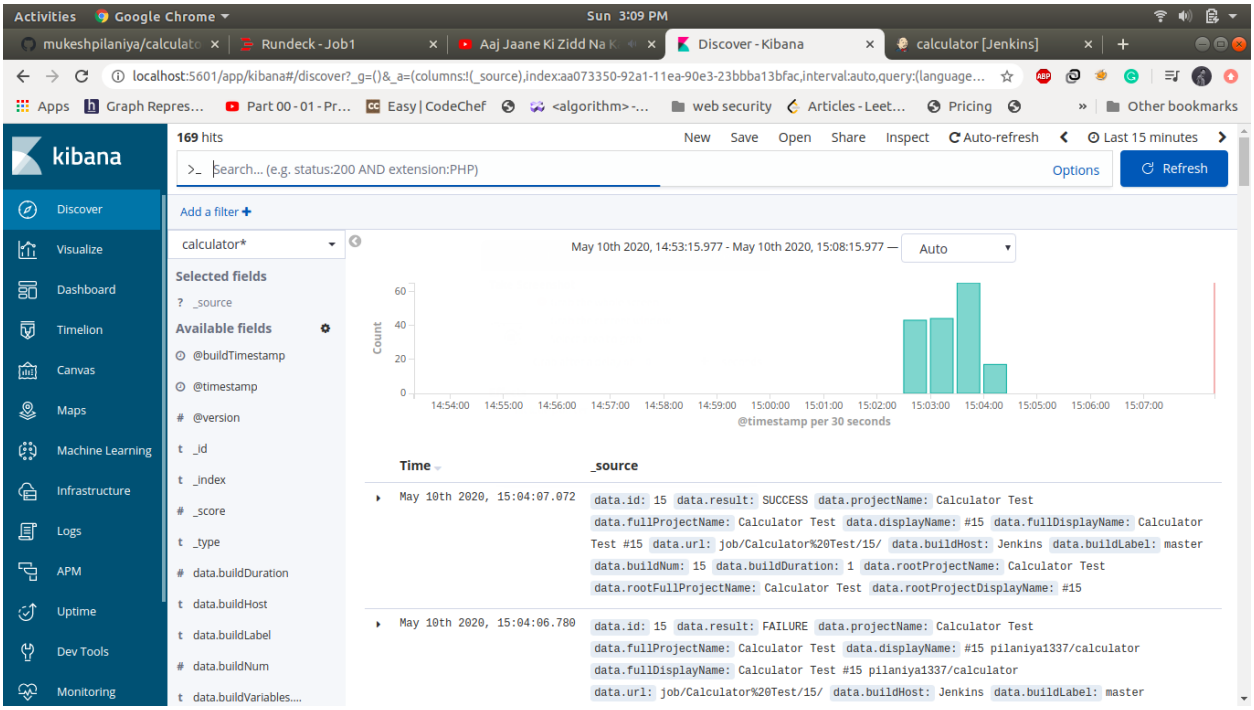


Create index in kibana and Visualize through graph: -

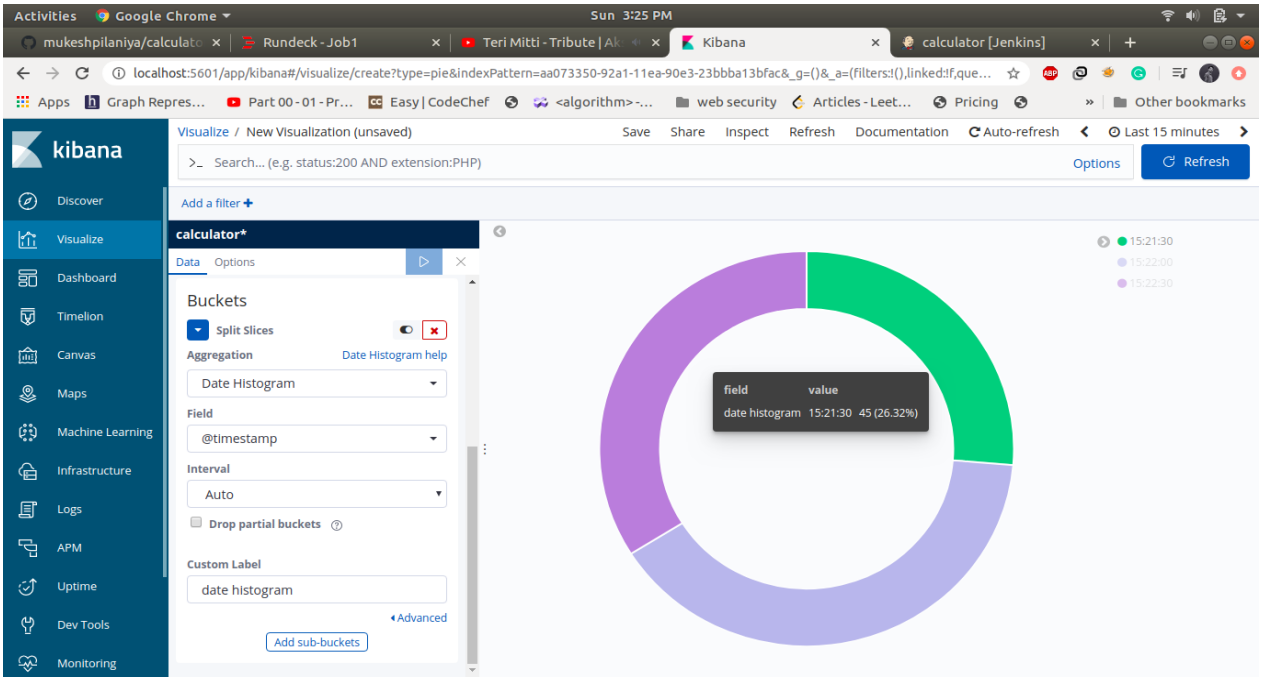
1. Go to url http://localhost:5601
2. To create kibana index pattern navigate to Management->under kibana section choose index pattern and create new index pattern name as “calculator*”. Click on next step and choose @Timestamp options



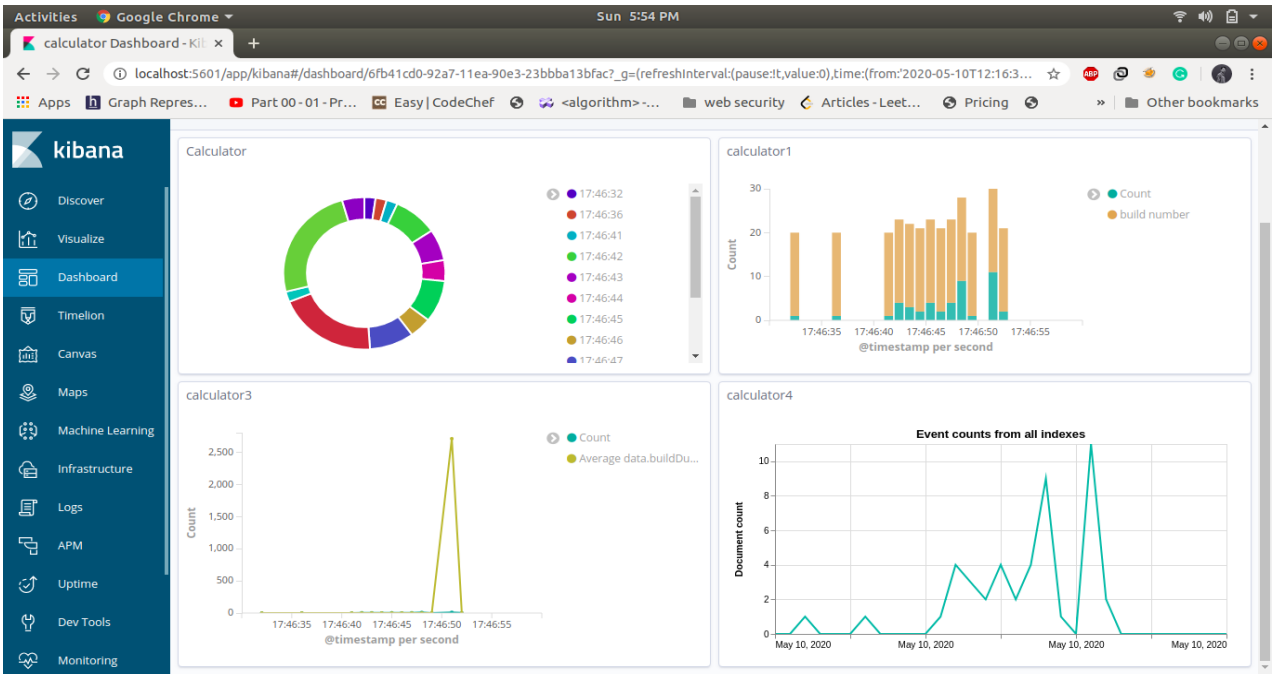
3. To see the log navigate to discover -> select calculator as index pattern



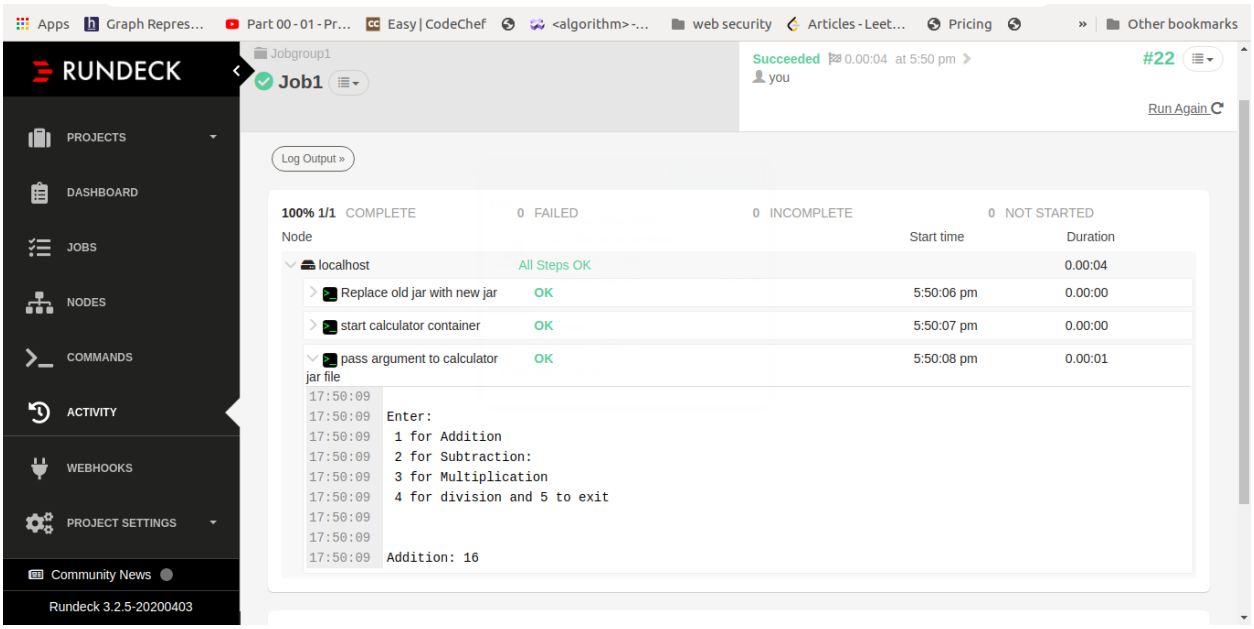
4. To visualize logs navigate to Visualize->click on + icon ->select one of map type->select calculator* index ->select appropriate fields->click on run->save ->name as calculator 1
   Create 3-4 graph and save as calculator 1, calculator 2, calculator 3

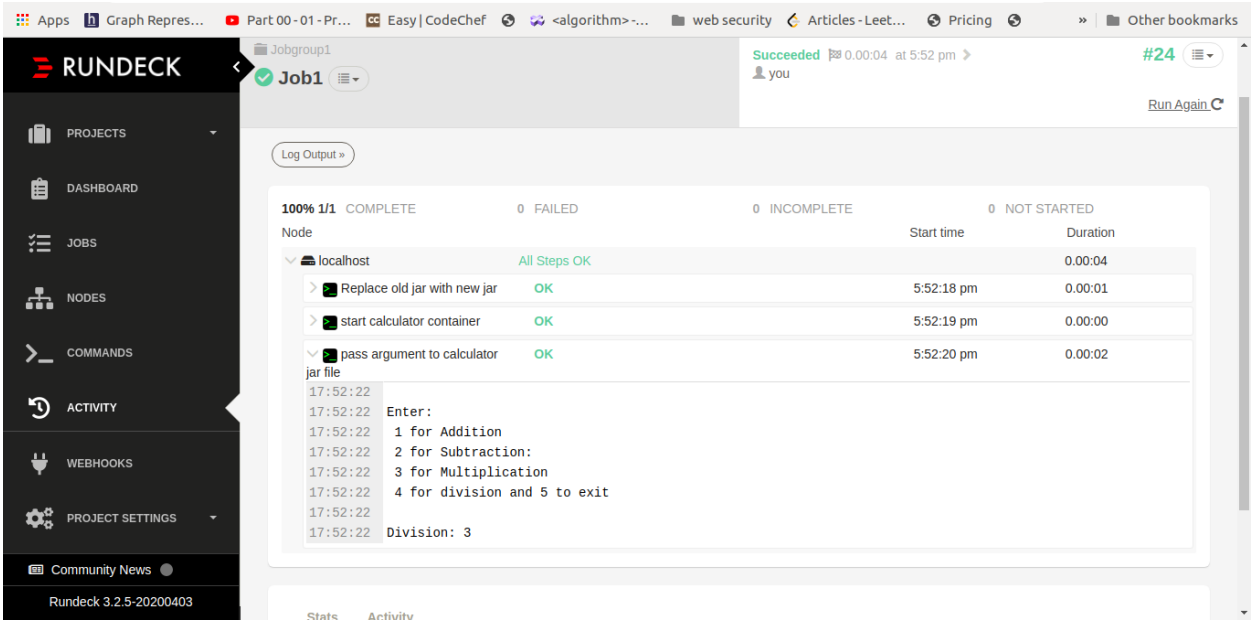


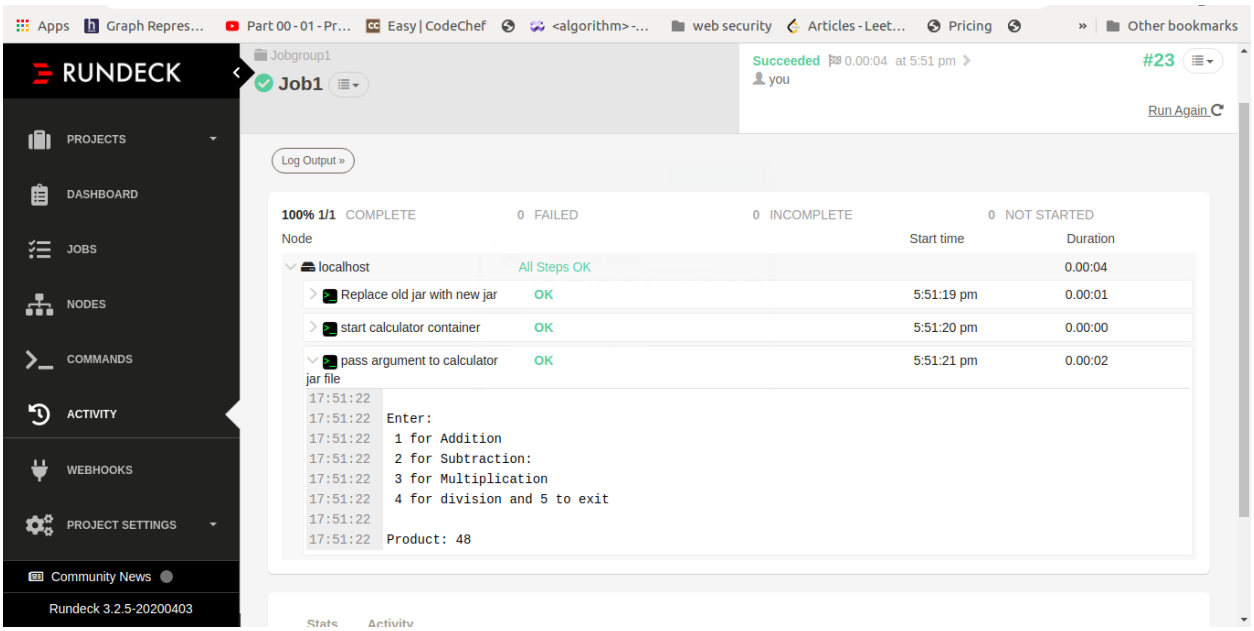
5. To create a Dashboard of calculator, navigate to Dashboard >click on add->in search bar type calculator it will show calculator 1,2 and 3 select all of these and save the dashboard name as Calculator.

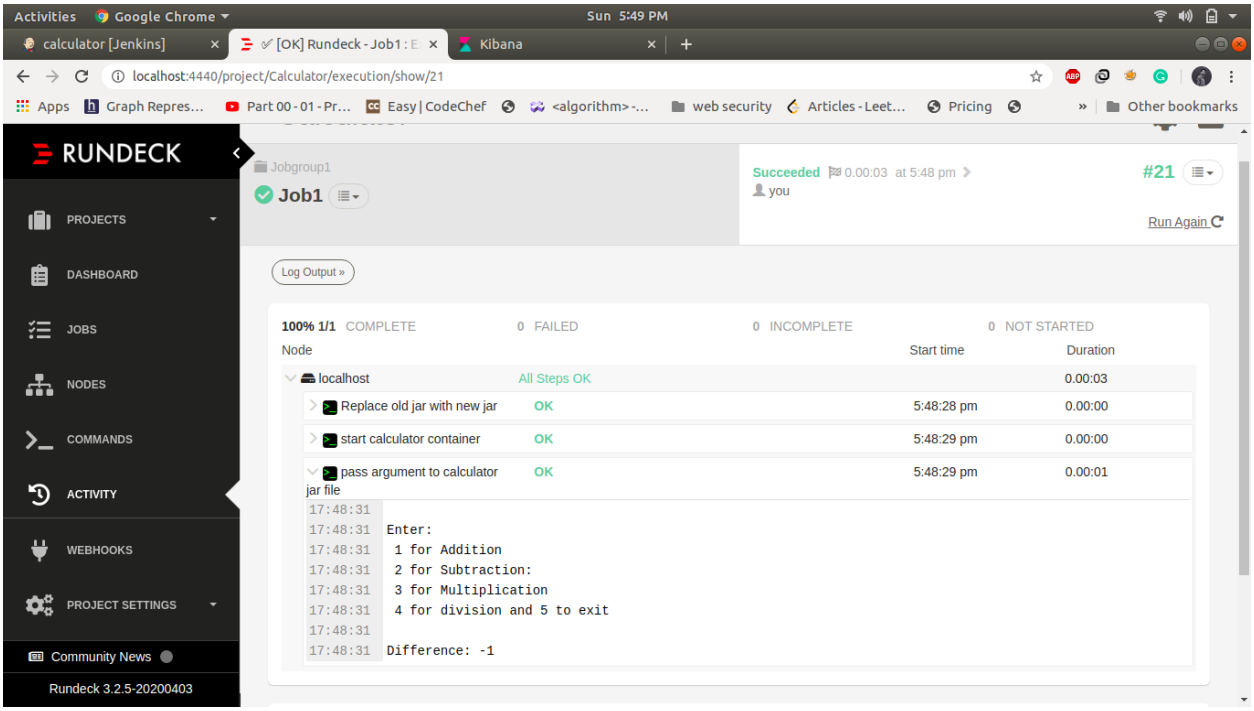


Results and execution: -When new features are introduced in a project, then from its building, testing, deployment and monitoring is done in an automated manner.
We first write code for addition method and latter add subtract, multiplication and division method. output of each method as we added in our project code




Apps
Graph Repres...
Part 00 - 01 - Pr...
Easy | CodeChef
<algorithm>-...
web security
Articles - Leet...
Pricing
Other bookmarks
RUNDECK
PROJECTS
DASHBOARD
JOBS
NODES
COMMANDS
ACTIVITY
WEBHOOKS
PROJECT SETTINGS
Community News
Rundeck 3.2.5-20200403
Jobgroup1
Job1
Succeeded 0.00:04 at 5:52 pm
you
#24
Run Again
Log Output »
100% 1/1 COMPLETE
0 FAILED
0 INCOMPLETE
0 NOT STARTED
Node
Start time
Duration
localhost
All Steps OK
0.00:04
Replace old jar with new jar
OK
5:52:18 pm
0.00:01
start calculator container
OK
5:52:19 pm
0.00:00
pass argument to calculator jar file
OK
5:52:20 pm
0.00:02
17:52:22
17:52:22 Enter:
17:52:22 1 for Addition
17:52:22 2 for Subtraction:
17:52:22 3 for Multiplication
17:52:22 4 for division and 5 to exit
17:52:22
17:52:22 Division: 3
Stats
Activity


Apps
Graph Repres...
Part 00 - 01 - Pr...
Easy | CodeChef
<algorithm>-...
web security
Articles - Leet...
Pricing
Other bookmarks
RUNDECK
PROJECTS
DASHBOARD
JOBS
NODES
COMMANDS
ACTIVITY
WEBHOOKS
PROJECT SETTINGS
Community News
Rundeck 3.2.5-20200403
Jobgroup1
Job1
Succeeded 0.00:04 at 5:51 pm
you
#23
Run Again
Log Output »
100% 1/1 COMPLETE
0 FAILED
0 INCOMPLETE
0 NOT STARTED
Node
Start time
Duration
localhost
All Steps OK
0.00:04
Replace old jar with new jar
OK
5:51:19 pm
0.00:01
start calculator container
OK
5:51:20 pm
0.00:00
pass argument to calculator jar file
OK
5:51:21 pm
0.00:02
17:51:22
17:51:22 Enter:
17:51:22 1 for Addition
17:51:22 2 for Subtraction:
17:51:22 3 for Multiplication
17:51:22 4 for division and 5 to exit
17:51:22
17:51:22 Product: 48
Stats
Activity



Conclusion: -DevOps tools help in automating the task of building, testing, releasing, deploying, operating and monitoring in a convenient and efficient way with enormous speed. Manual intervention prone to errors but automated environments are not. Data sharing techniques are used effectively to connect Devs with Ops.

References: -

[1] GitHub project: -https://github.com/mukeshpilaniya/calculator
[2] DockerHub profile: -https://hub.docker.com/u/pilaniya1337