Software Production Engineering

Dev-Ops Mini Project: Scientific Calculator

ROOPAM PATIL || MT2021110

Github link: https://github.com/Roopam10/miniproject_MT2021110.git

Dockerhub link:
https://hub.docker.com/repository/docker/cristiano10/calculator_roopam

Contents

Introduction

This report is aimed towards defining the details of a complete Dev-ops enables application:**Scientific Calculator**. This aim and plan is to automate the development, testing and
deployment pipeline with the help of certain dev-ops tools. The **Scientific Calculator** application that is developed here is a Terminal based application where user can perform
following operations:

1. Square root function
2. Factorial function
3. logarithm function(base 10)
4. Power function

Devops Pipeline

The cycle will include -> Once a developer pushes the code onto a code repository CI/CD pipeline comes into the pictures. A tool named Jenkins will build the code create a Docker
image and will perform deployment operation with the help of Ansible.

- Development IDE: Intellij
- Language: JAVA
- SCM – Github - https://github.com/Roopam10/miniproject_MT2021110.git
- Building and Packaging - Maven
- Docker image - https://hub.docker.com/repository/docker/cristiano10/calculator_roopam
- Continous Integration – Jenkins
- Continous Deployment – Ansible
- Continous Monitoring – ELK Stack

Source Code Management

Version Control : Github

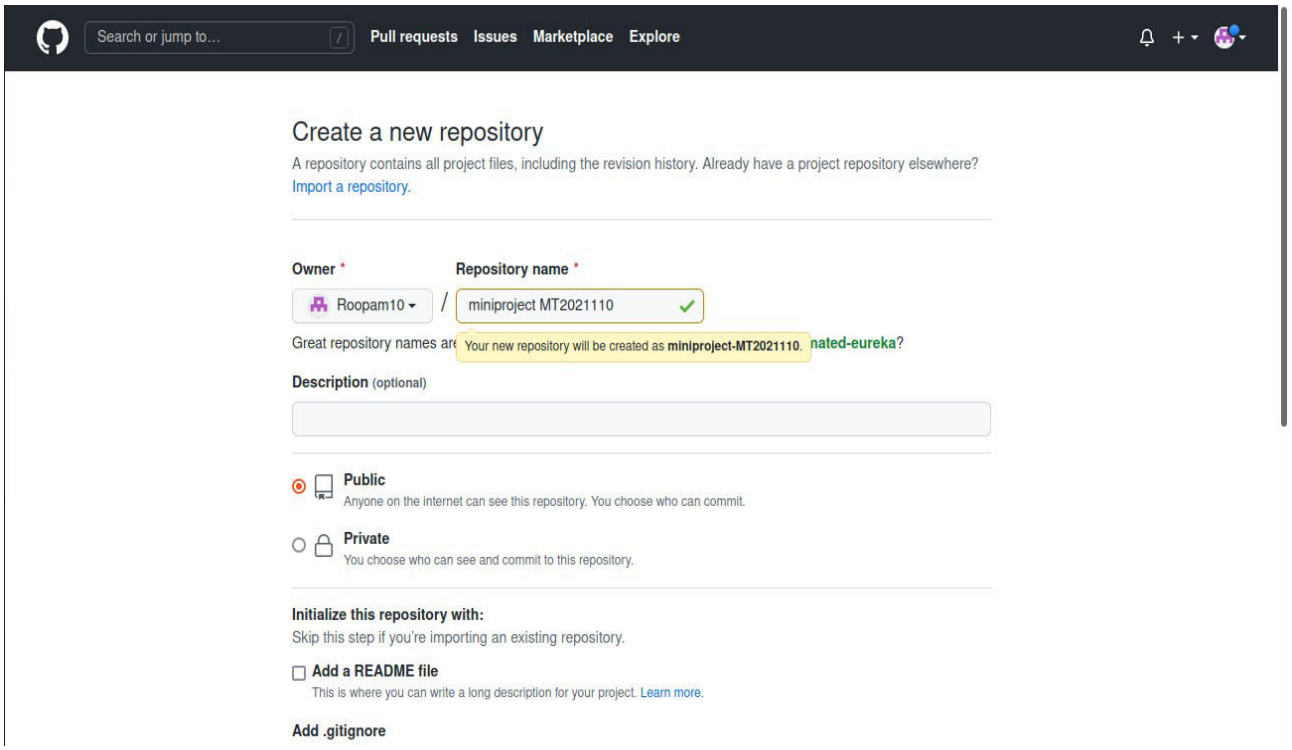


Fig:1

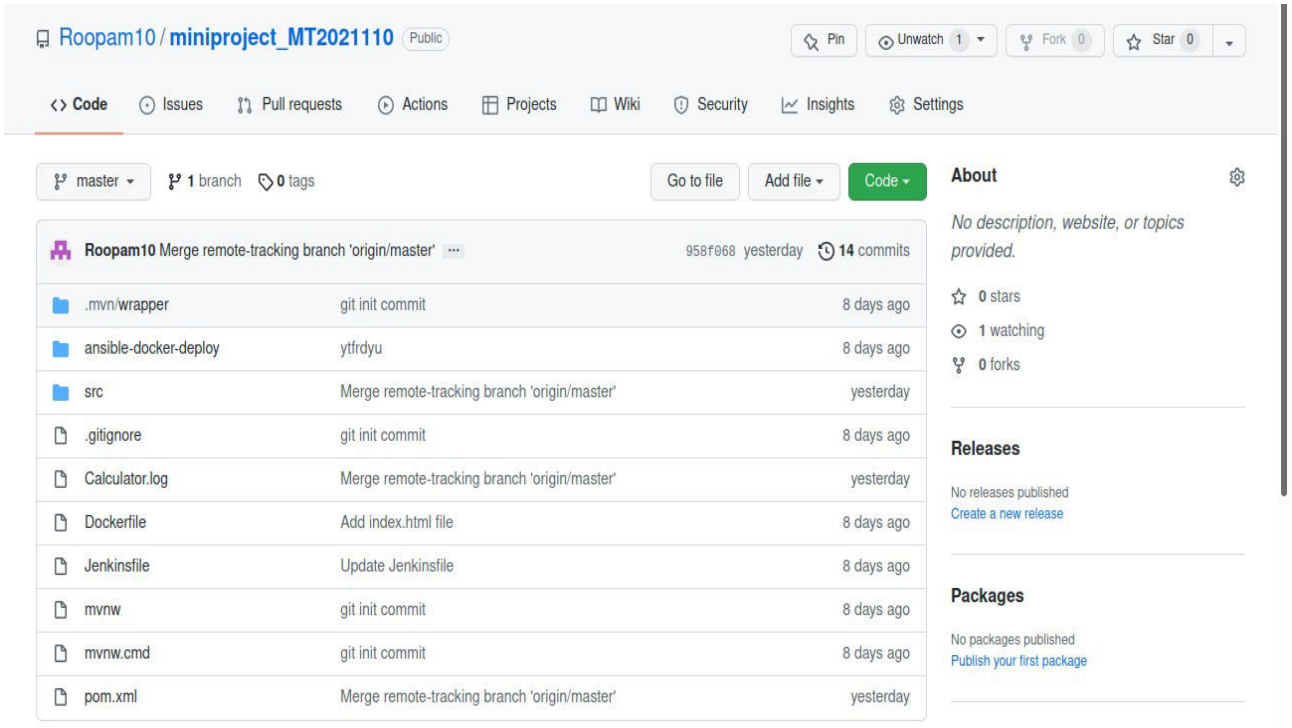


Fig:2

Create a new repository on to https://www.github.com.

1.Give appropriate name to your repository
2.Give description about the repository.

After clicking on submit/Ok button a new repo will get generated in the logged-in user's
account.

Now create a Maven project in Intellij IDEA ->
File -> New -> Project -> Maven -> Give the project Name-> Finish

Now you have a project directory in your local system. Commit and Push it to Github remote
repository by issuing following commands in the project directory:

1. git init
2. git add
3. git commit -m "first commit"
4. git branch -M master
5. git remote add origin https://github.com/Roopam10/miniproject_MT2021110.git
6. git push -u origin master

Code Build and test

In the calculator program, Apache Maven is responsible for managing dependencies and building the project. It is Maven who finally outputs the SNAPSHOT jar of the project that has
the compiled classes along with other classes the project depends on. The pom.xml file of the project is shown in below screenshot.

Pom.xml

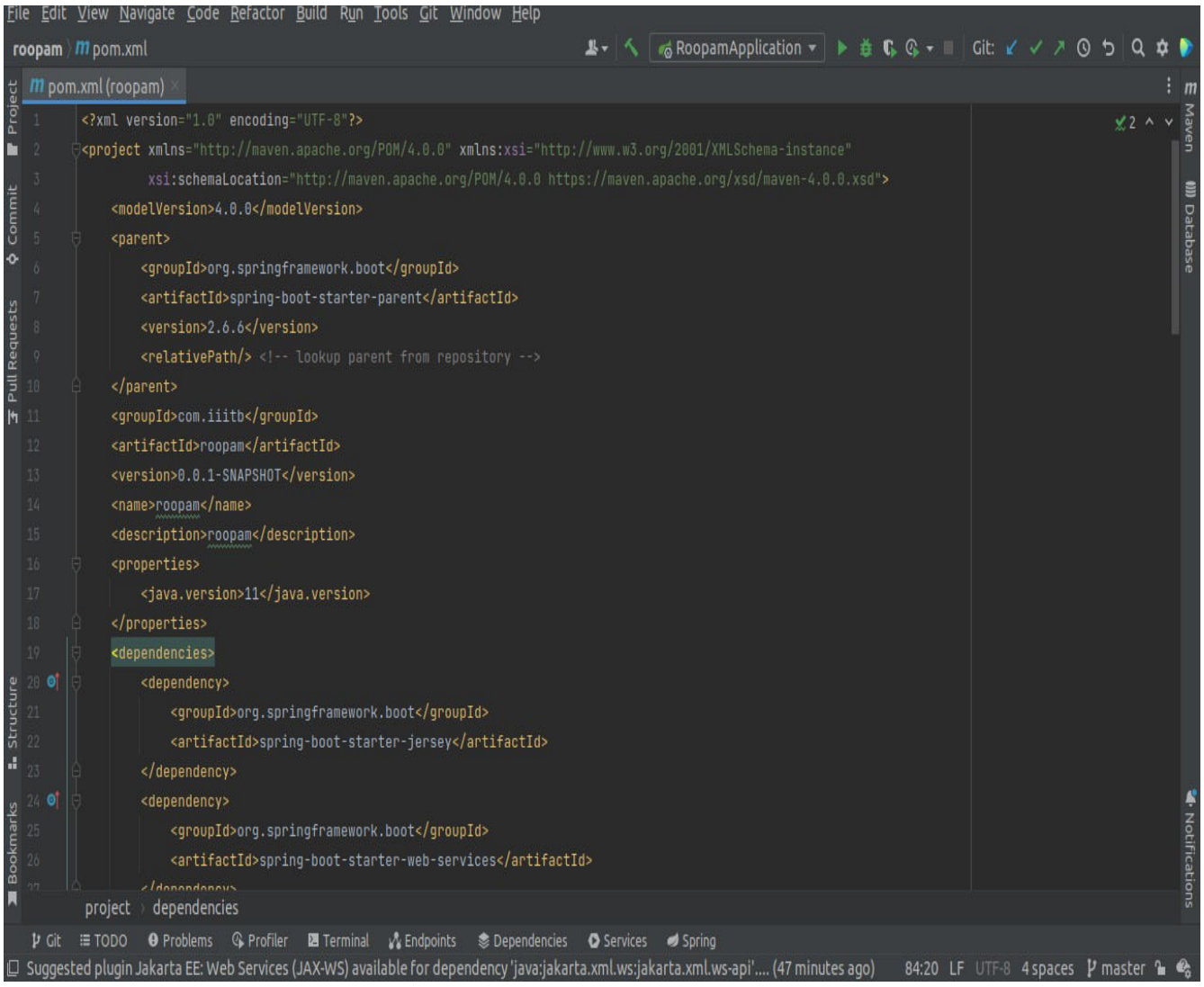


Fig:3

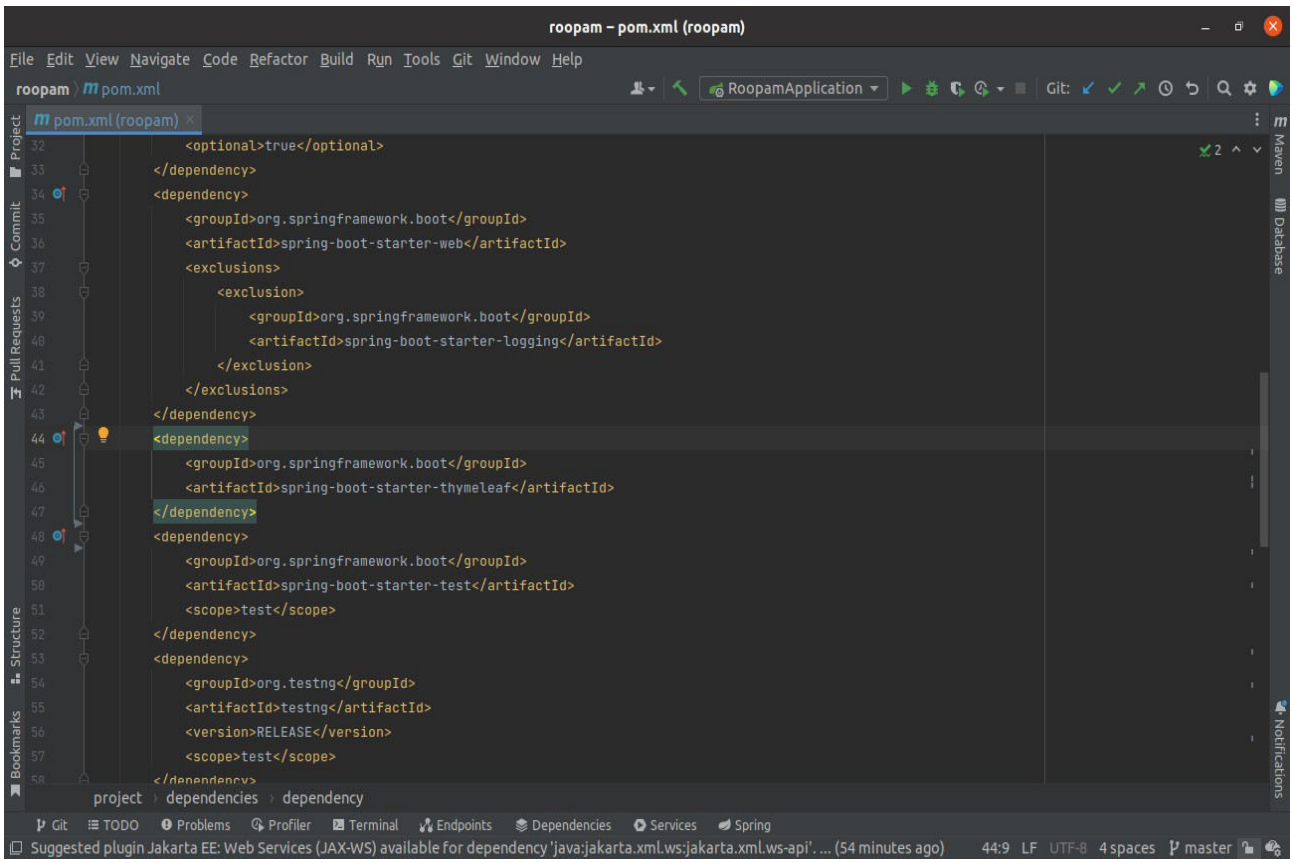

roopam – pom.xml (roopam)
File Edit View Navigate Code Refactor Build Run Tools Git Window Help
roopam › pom.xml
RoopamApplication
Git:
pom.xml (roopam)
<optional>true</optional>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-web</artifactId>
<exclusions>
<exclusion>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-logging</artifactId>
</exclusion>
</exclusions>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-thymeleaf</artifactId>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-test</artifactId>
<scope>test</scope>
</dependency>
<dependency>
<groupId>org.testng</groupId>
<artifactId>testng</artifactId>
<version>RELEASE</version>
<scope>test</scope>
project › dependencies › dependency
Git TODO Problems Profiler Terminal Endpoints Dependencies Services Spring
Suggested plugin Jakarta EE: Web Services (JAX-WS) available for dependency 'java:jakarta.xml.ws:jakarta.xml.ws-api'. ... (54 minutes ago)
44:9 LF UTF-8 4 spaces master

Fig:4

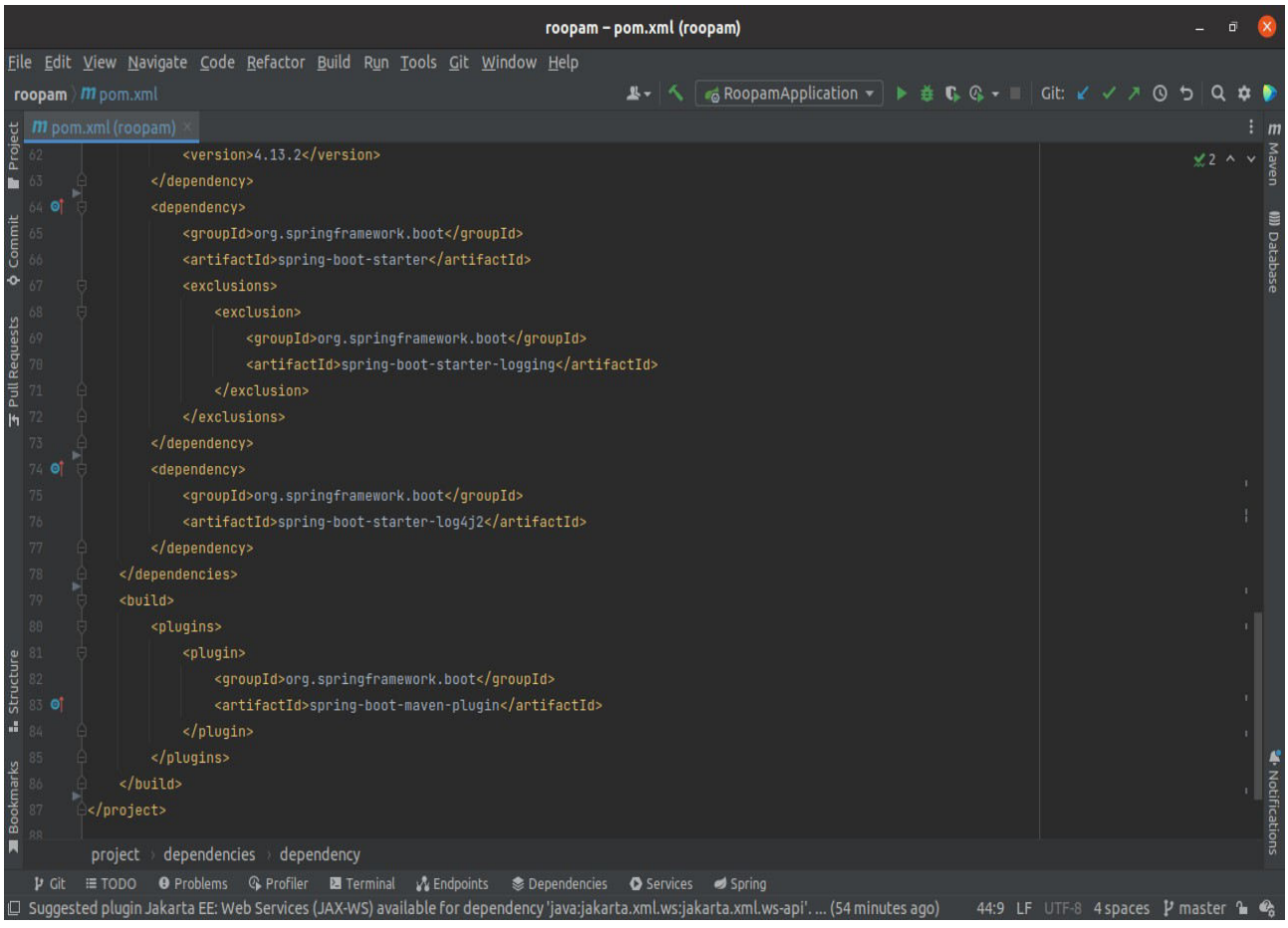

roopam – pom.xml (roopam)
File Edit View Navigate Code Refactor Build Run Tools Git Window Help
roopam › pom.xml
RoopamApplication
Git:
pom.xml (roopam)
<version>4.13.2</version>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter</artifactId>
<exclusions>
<exclusion>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-logging</artifactId>
</exclusion>
</exclusions>
</dependency>
<dependency>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-starter-log4j2</artifactId>
</dependency>
</dependencies>
<build>
<plugins>
<plugin>
<groupId>org.springframework.boot</groupId>
<artifactId>spring-boot-maven-plugin</artifactId>
</plugin>
</plugins>
</build>
</project>
project › dependencies › dependency
Git TODO Problems Profiler Terminal Endpoints Dependencies Services Spring
Suggested plugin Jakarta EE: Web Services (JAX-WS) available for dependency 'java:jakarta.xml.ws:jakarta.xml.ws-api'. ... (54 minutes ago)
44:9 LF UTF-8 4 spaces master

Fig:5

Source Code and Output :

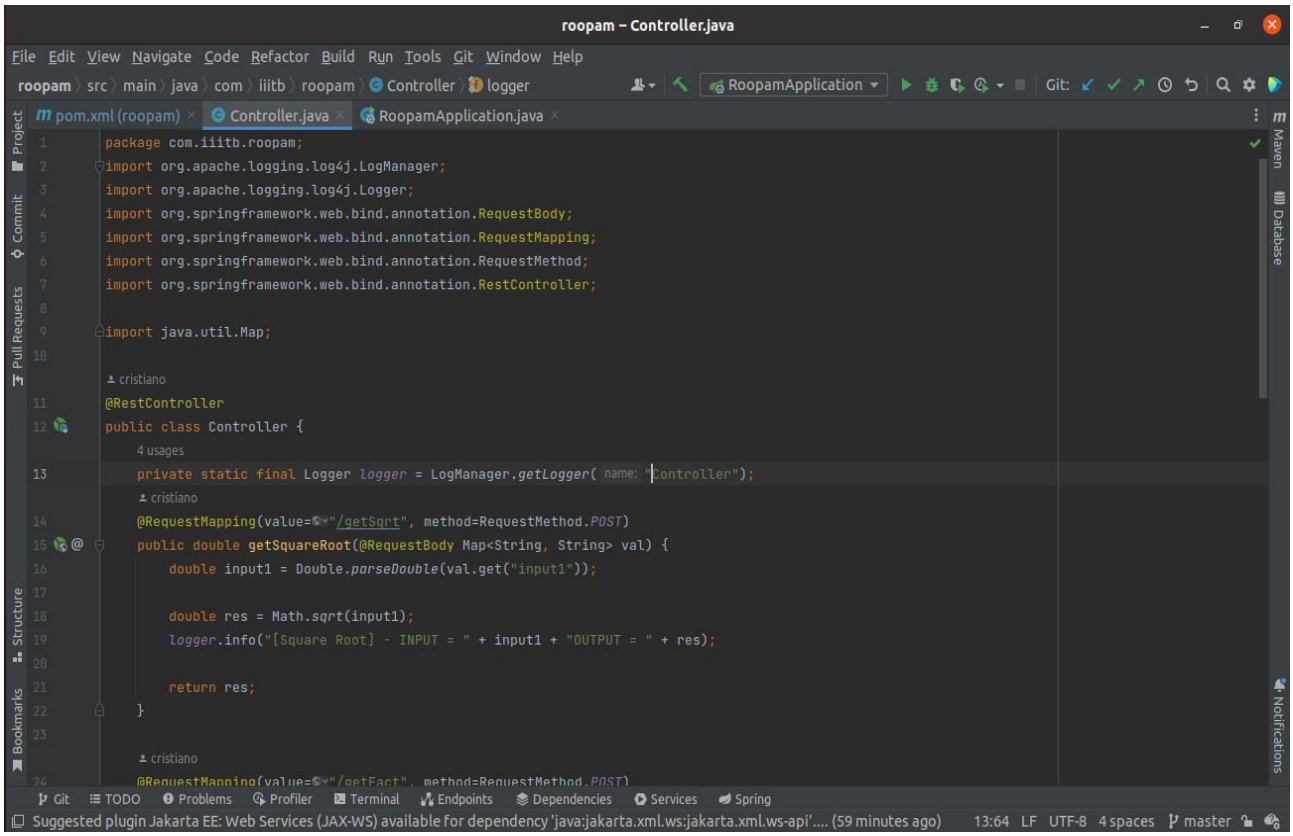


Fig:6

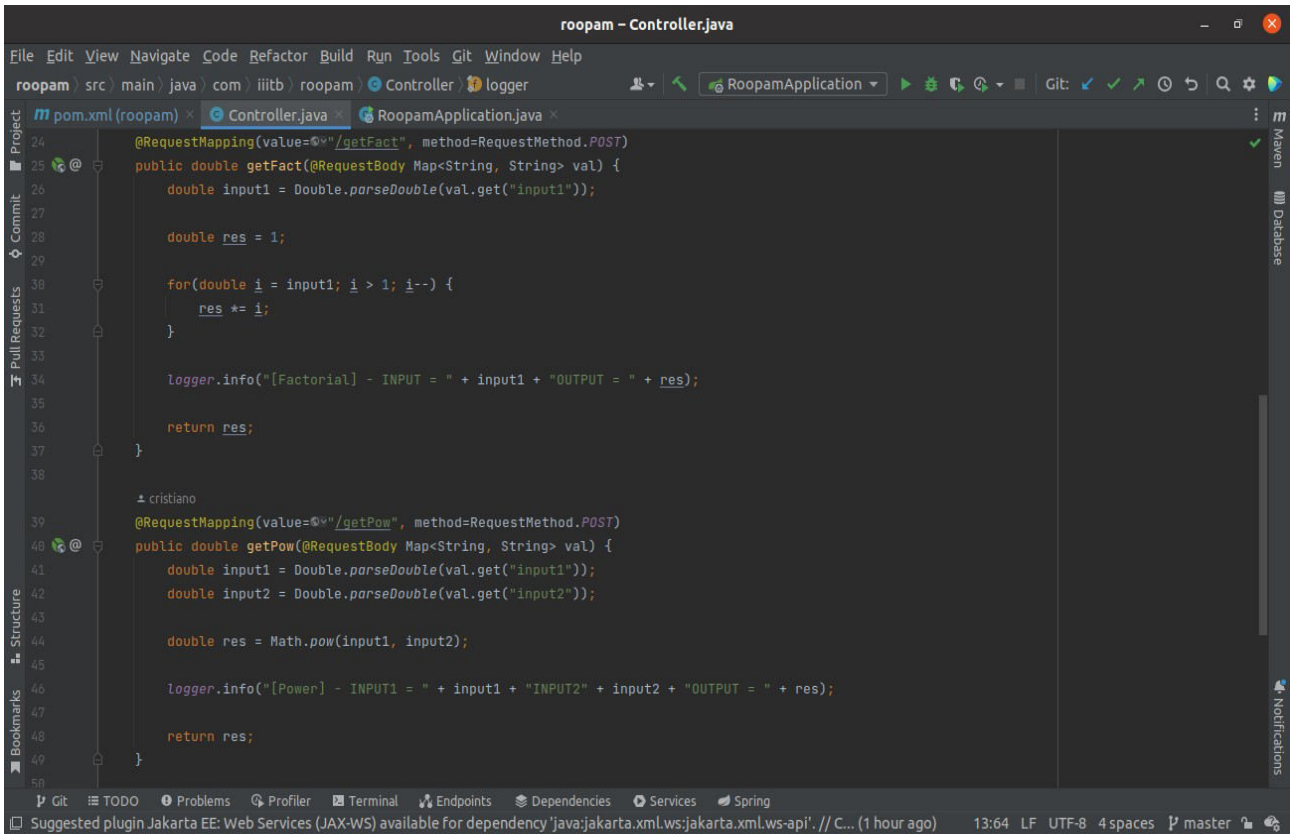


Fig: 7

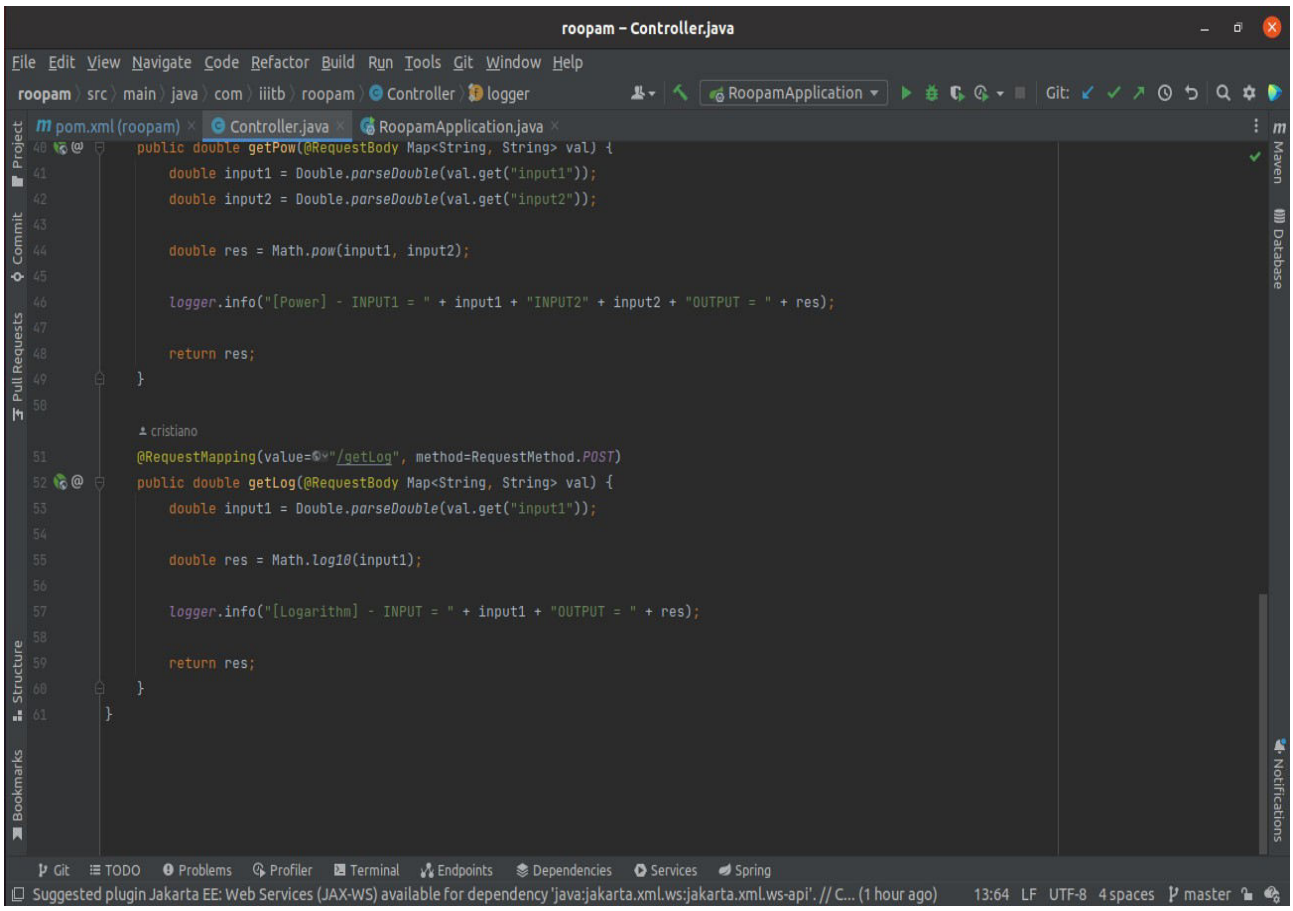

roopam – Controller.java
File Edit View Navigate Code Refactor Build Run Tools Git Window Help
roopam src main java com iiitb roopam Controller logger
RoopamApplication
Git:
pom.xml (roopam)
Controller.java
RoopamApplication.java
public double getPow(@RequestBody Map<String, String> val) {
double input1 = Double.parseDouble(val.get("input1"));
double input2 = Double.parseDouble(val.get("input2"));
double res = Math.pow(input1, input2);
logger.info("[Power] - INPUT1 = " + input1 + "INPUT2" + input2 + "OUTPUT = " + res);
return res;
}
cristiano
@RequestMapping(value="/getLog", method=RequestMethod.POST)
public double getLog(@RequestBody Map<String, String> val) {
double input1 = Double.parseDouble(val.get("input1"));
double res = Math.log10(input1);
logger.info("[Logarithm] - INPUT = " + input1 + "OUTPUT = " + res);
return res;
}
}
Git TODO Problems Profiler Terminal Endpoints Dependencies Services Spring
Suggested plugin Jakarta EE: Web Services (JAX-WS) available for dependency 'java:jakarta.xml.ws:jakarta.xml.ws-api'. // C... (1 hour ago)
13:64 LF UTF-8 4 spaces master

Fig: 8

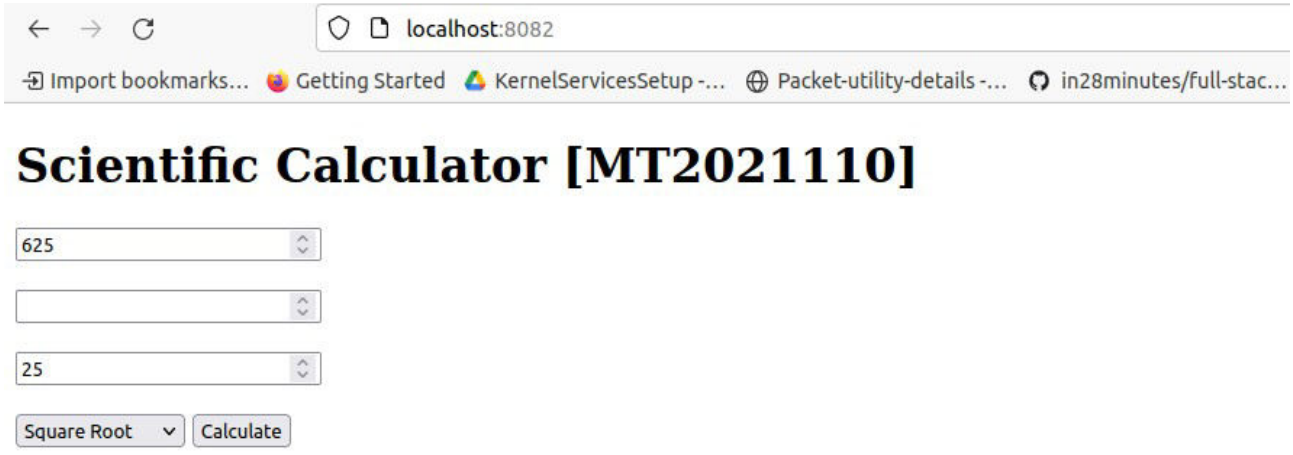

localhost:8082
Import bookmarks...
Getting Started
KernelServicesSetup -...
Packet-utility-details -...
in28minutes/full-stac...
Scientific Calculator [MT2021110]
625
25
Square Root
Calculate

Fig : 9

Logger :

Logging is an important feature that helps developers to trace out the errors. In our application we are using log4j2 to generate the loggers.

1. To set it up we need to add “org.apache.logging.log4j” dependency in pom.xml.
2. Then create a file with name: “log4j2.xml” under the resources folder.
3. And then write the logger statements inside the code.

After all these steps a log file will get generated (with name and path as specified in
log4j2.xml).

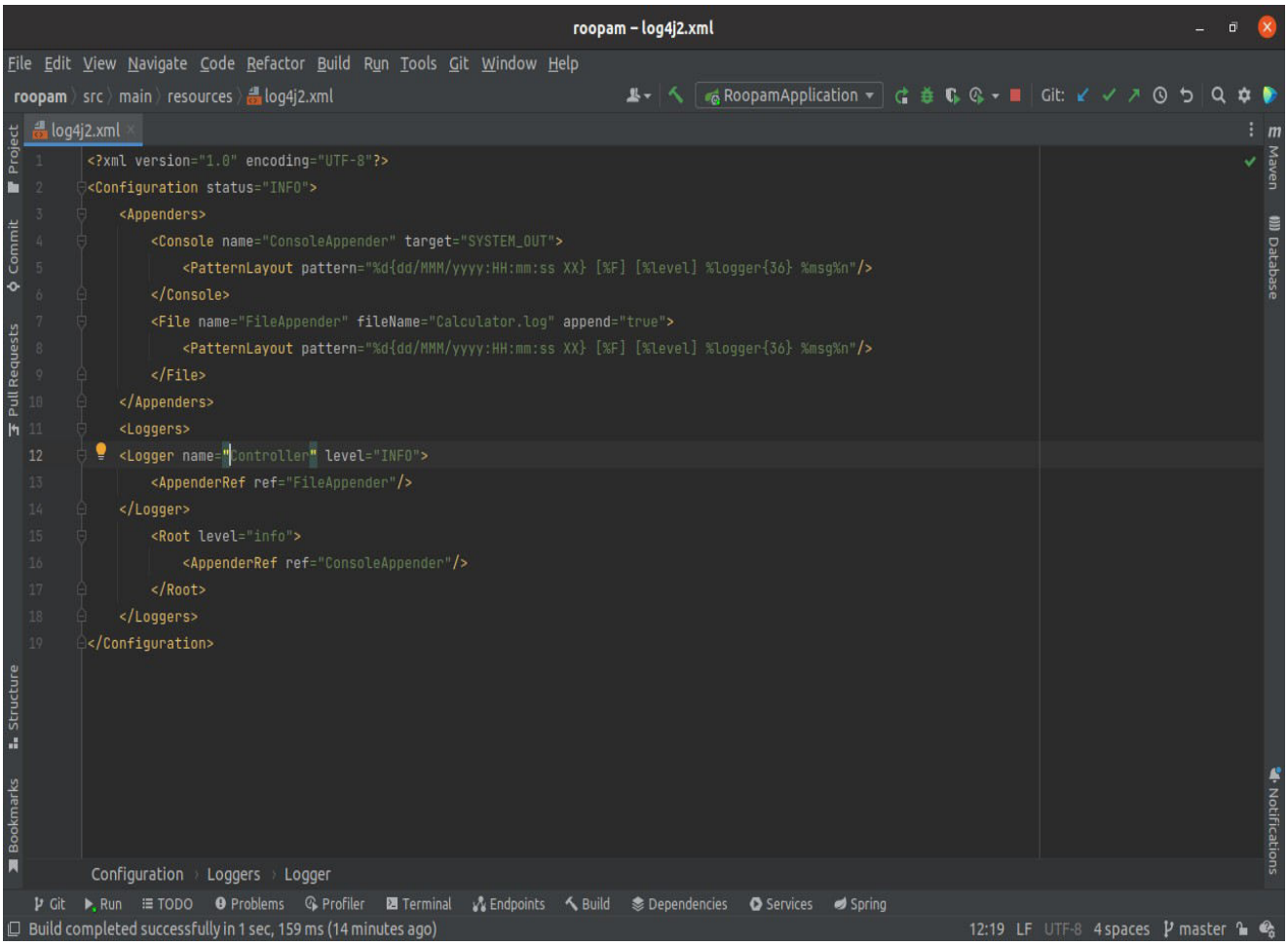


Fig: 10

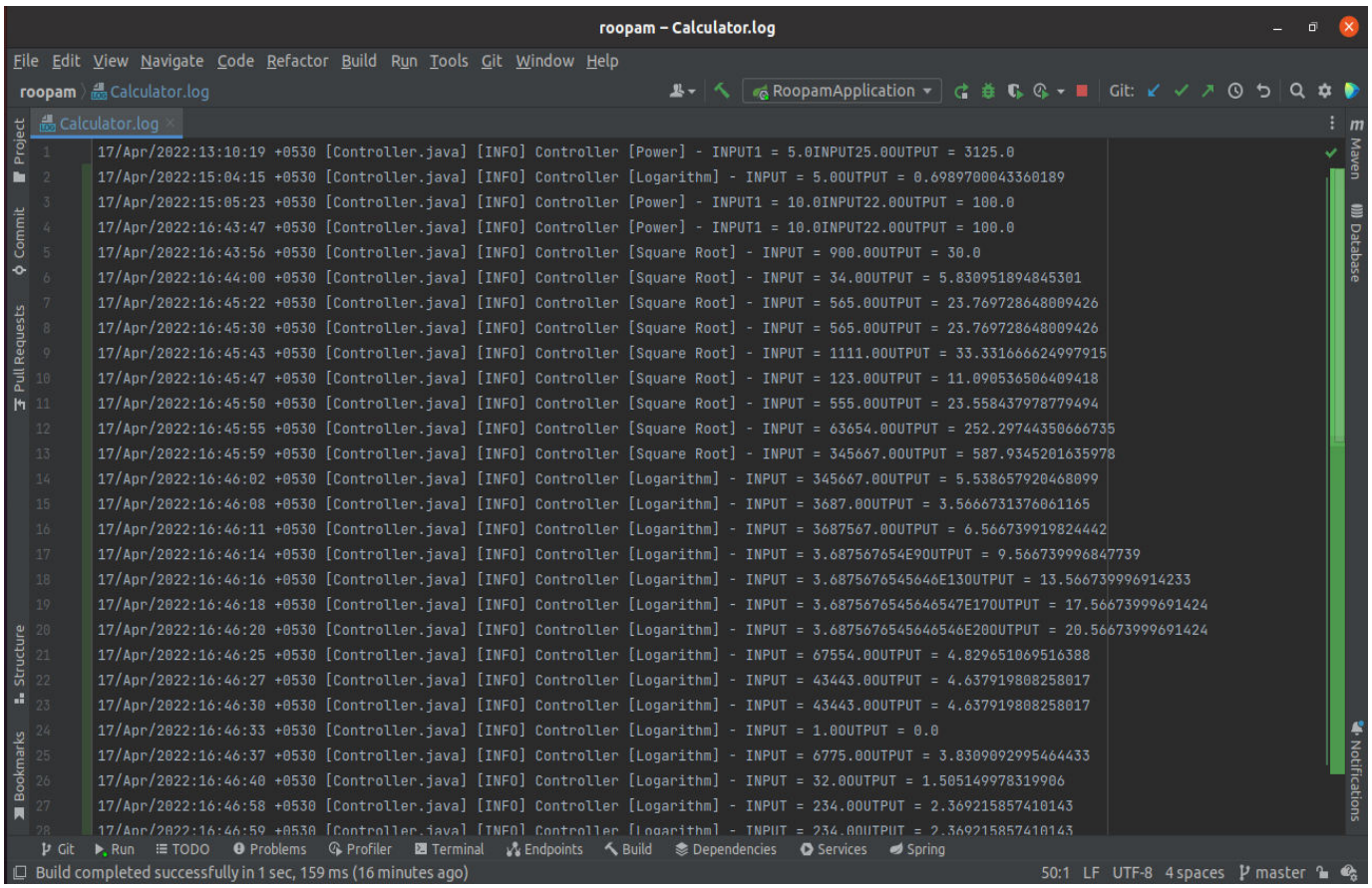


Fig : 11

Build Maven:

Maven Command to create the JAR locally: mvn clean test package.

1. Clean – It will clear older snapshots (if any)
2. test – It will run unit tests.
3. Package – It will create the jar file.

Alternate to this we can also run “mvn install” this will execute all the required steps and a target folder will get generated which will contain the JAR file.

Continuous Integration

Continuous Integration (CI) refers to the practice of integrating code changes with the existing code as and when it is written. This includes building the project and running the test cases too automatically as described above. Jenkins is the tool that I have used for Continuous Integration. It keeps watching the SCM system for changes in the code and builds it as and
when it detects the changes. Apache Maven is integrated with Jenkins so that Jenkins can trigger maven builds.

Jenkins Installation

To install jenkins, follow the steps given below:

1. Download and install the necessary GPG key> wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -
2. Add the necessary repository > sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'
3. Add the universe repository > sudo add-apt-repository universe.
4. Update apt > sudo apt update
5. Install Jenkins > sudo apt-get install jenkins –y
6. Start Jenkins > sudo systemctl start jenkins
7. Install Git, Maven, JUnit, Ansible, Docker plugin by going to Manage Jenkins -> Manage Plugins

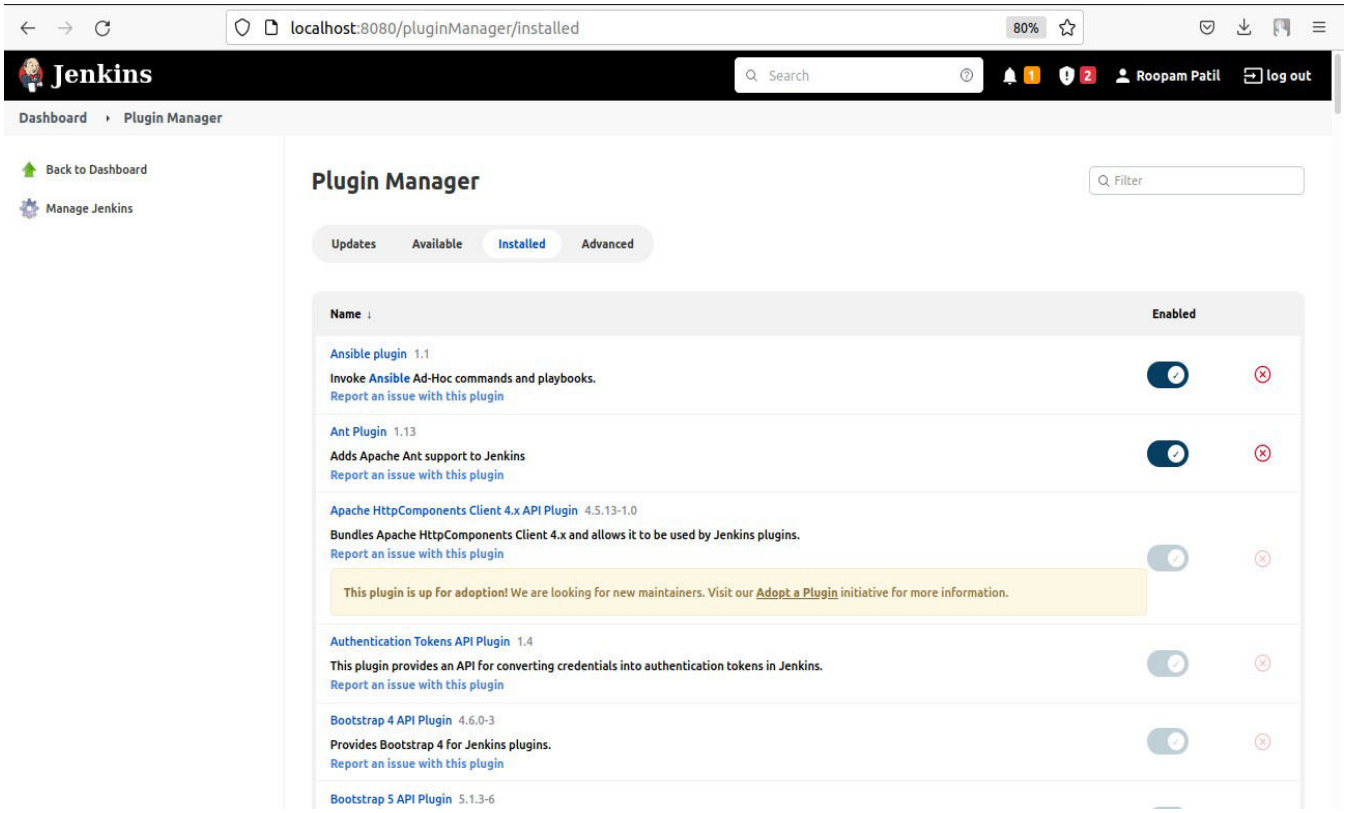

localhost:8080/pluginManager/installed
80%
Jenkins
Search
Roopam Patil
log out
Dashboard
Plugin Manager
Back to Dashboard
Manage Jenkins
Plugin Manager
Filter
Updates
Available
Installed
Advanced
Name
Enabled
Ansible plugin 1.1
Invoke Ansible Ad-Hoc commands and playbooks.
Report an issue with this plugin
Ant Plugin 1.13
Adds Apache Ant support to Jenkins
Report an issue with this plugin
Apache HttpComponents Client 4.x API Plugin 4.5.13-1.0
Bundles Apache HttpComponents Client 4.x and allows it to be used by Jenkins plugins.
Report an issue with this plugin
This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.
Authentication Tokens API Plugin 1.4
This plugin provides an API for converting credentials into authentication tokens in Jenkins.
Report an issue with this plugin
Bootstrap 4 API Plugin 4.6.0-3
Provides Bootstrap 4 for Jenkins plugins.
Report an issue with this plugin
Bootstrap 5 API Plugin 5.1.3-6

Fig : 12

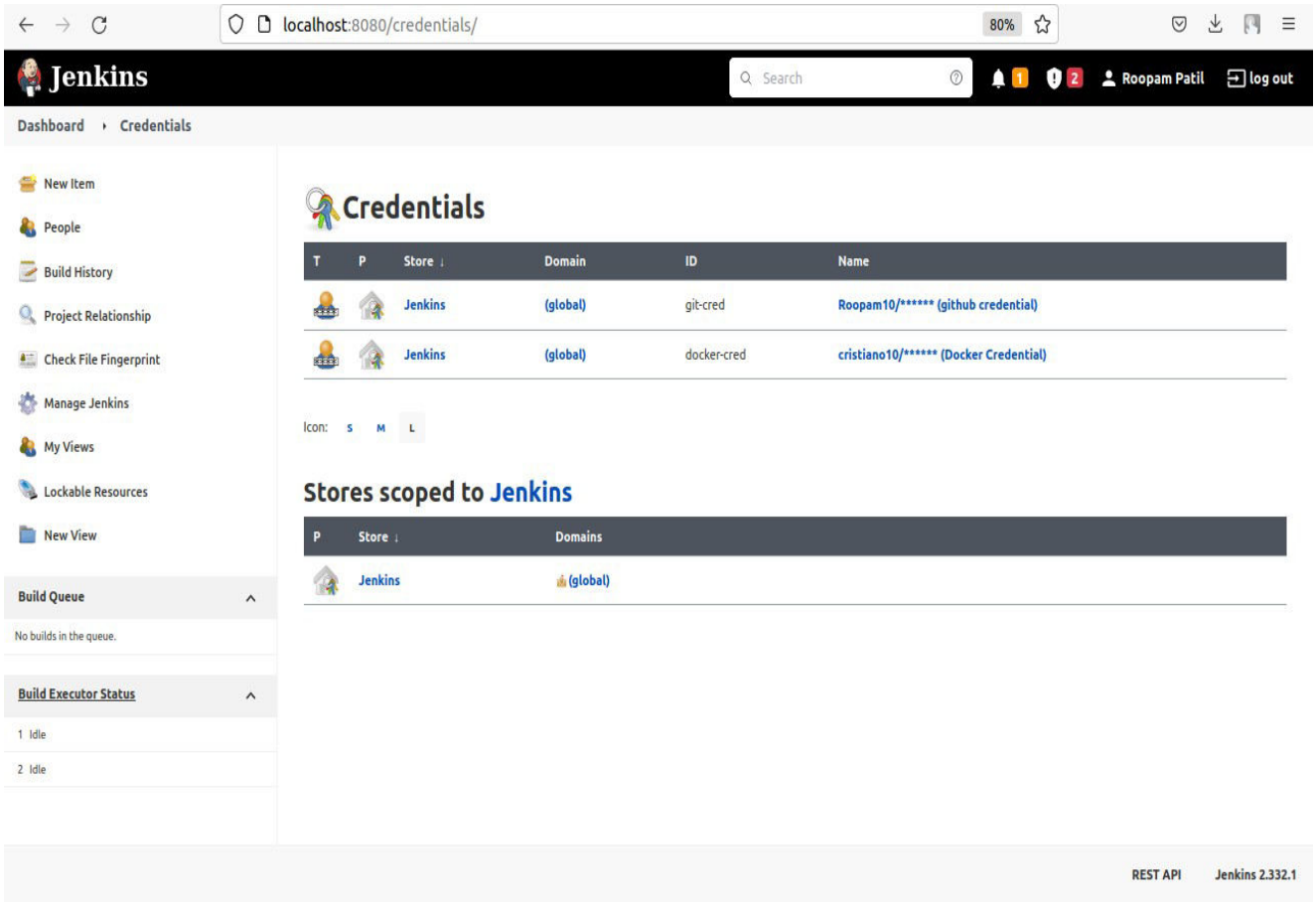

localhost:8080/credentials/
80%
Jenkins
Search
Roopam Patil
log out
Dashboard
Credentials
New Item
People
Build History
Project Relationship
Check File Fingerprint
Manage Jenkins
My Views
Lockable Resources
New View
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
Credentials
T
P
Store
Domain
ID
Name
Jenkins
(global)
git-cred
Roopam10/****** (github credential)
Jenkins
(global)
docker-cred
cristiano10/****** (Docker Credential)
Icon: S M L
Stores scoped to Jenkins
P
Store
Domains
Jenkins
(global)
REST API
Jenkins 2.332.1

Fig : 13

Jenkins Pipeline

A Jenkins pipeline gives us a graphical view of the various steps of a CI/CD pipeline. It allows us to link different Jenkins jobs and allows us to check their progress during execution. To create
Jenkins pipeline, follow the given steps:

1. Go to jenkins Dashboard.
2. Click on new item.
3. Enter Project pipeline name.
4. Click on pipeline.
5. Click OK.

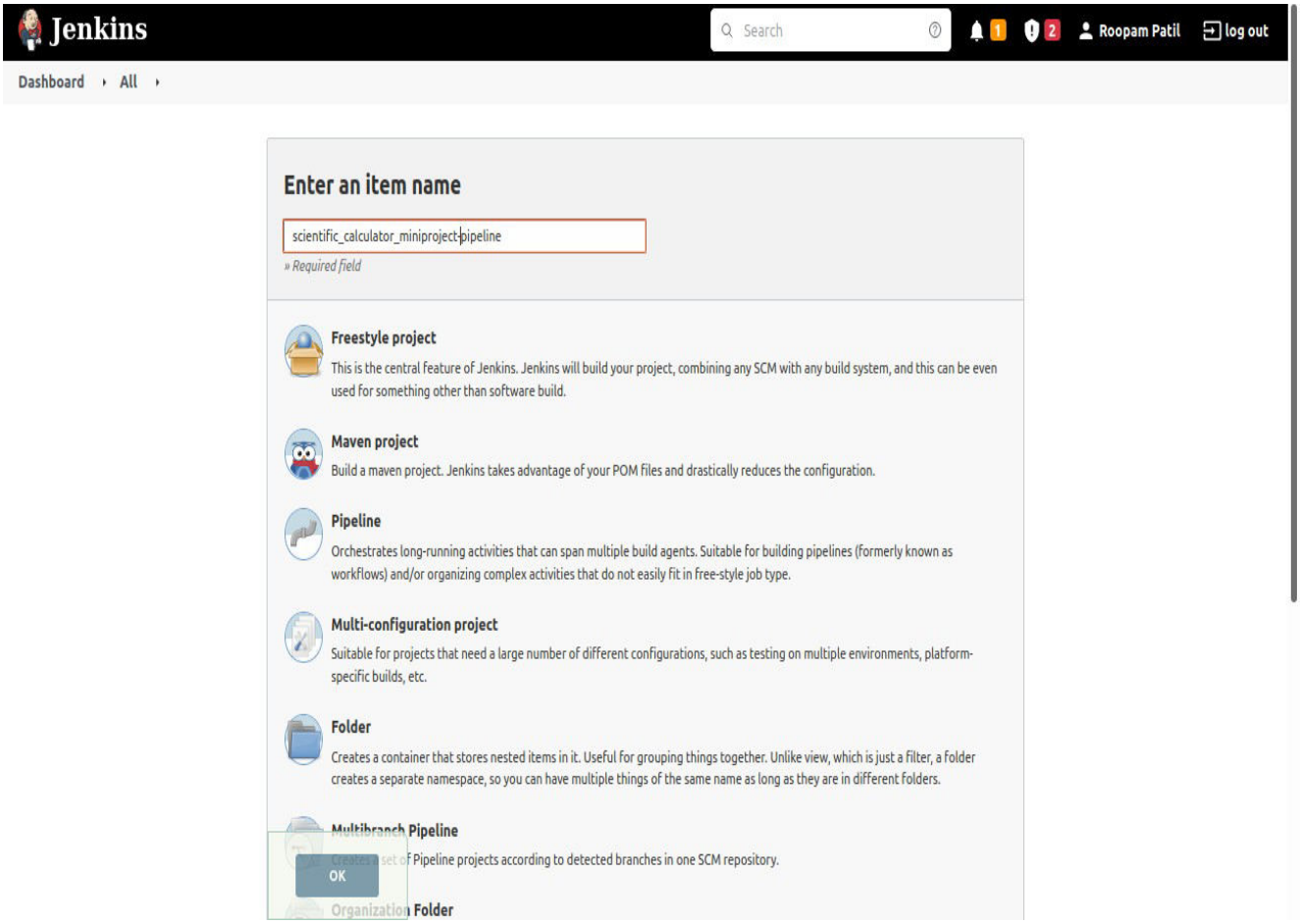


Fig : 14

6. After this a new window will get opened. Provide description.

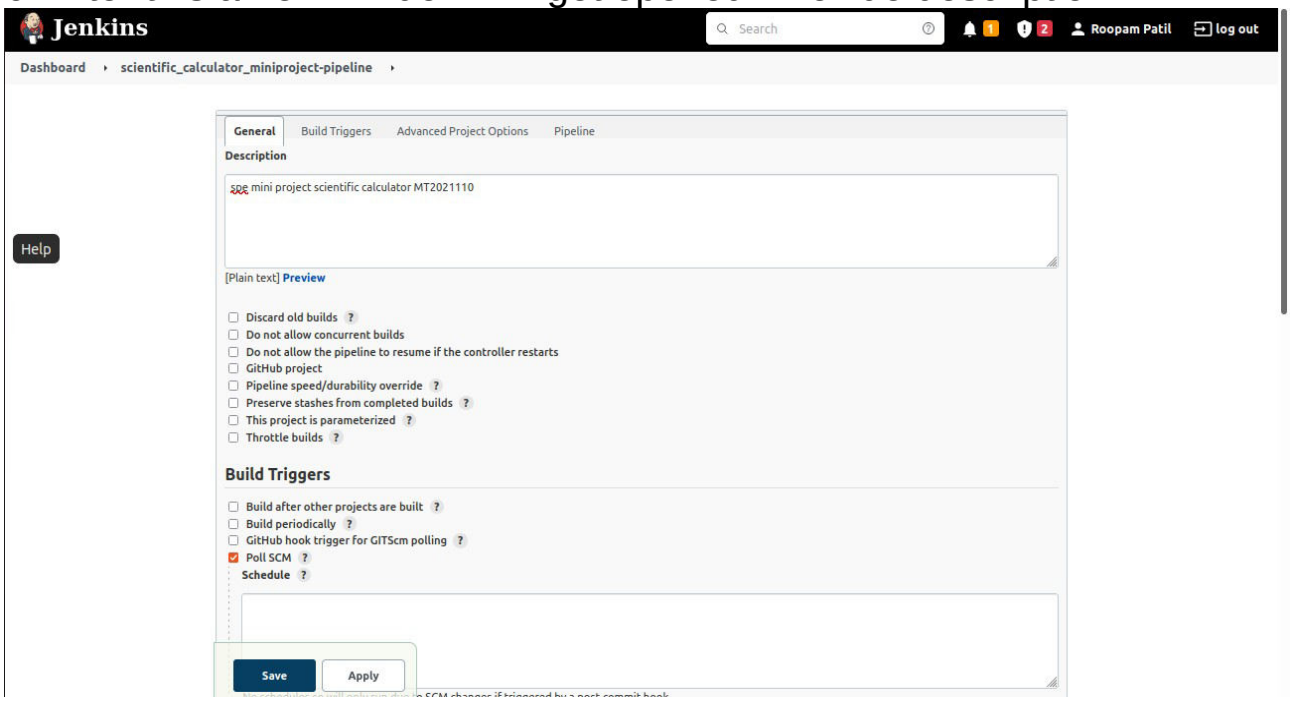


Fig : 15

7. Select poll SCM and give input like: * * * * * to poll for the SCM changes .

8. We can provide pipeline scripts either in this page as well or we can have it in our code base. For this project we'll be using Jenkins file. So provide details like below:

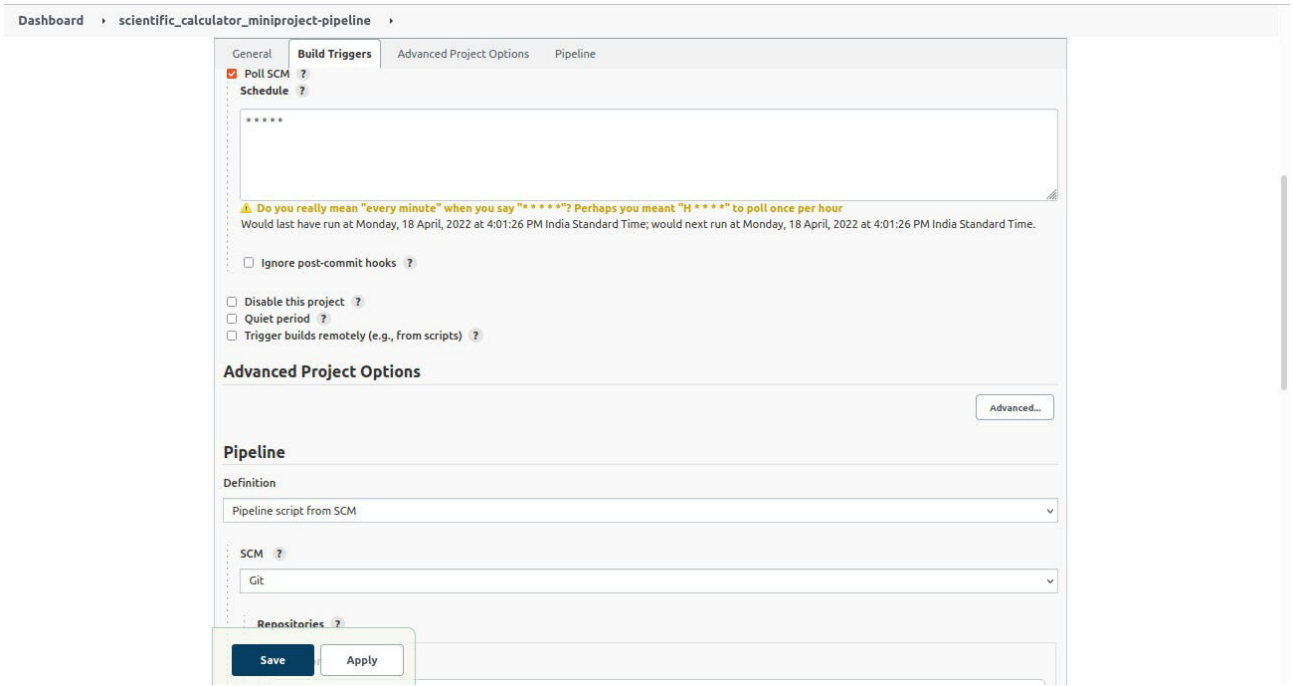


Fig : 16

9. Jenkins File:

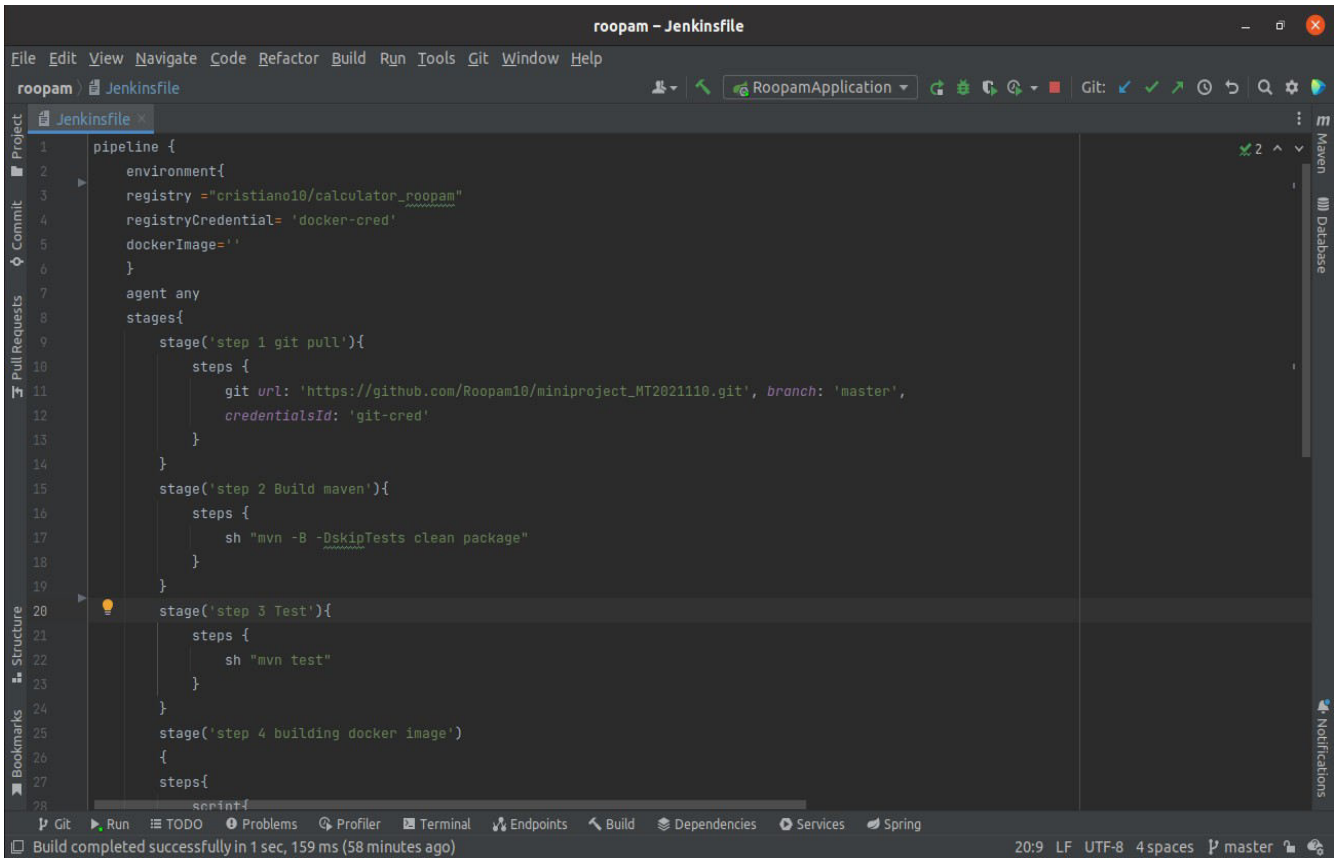


Fig : 17

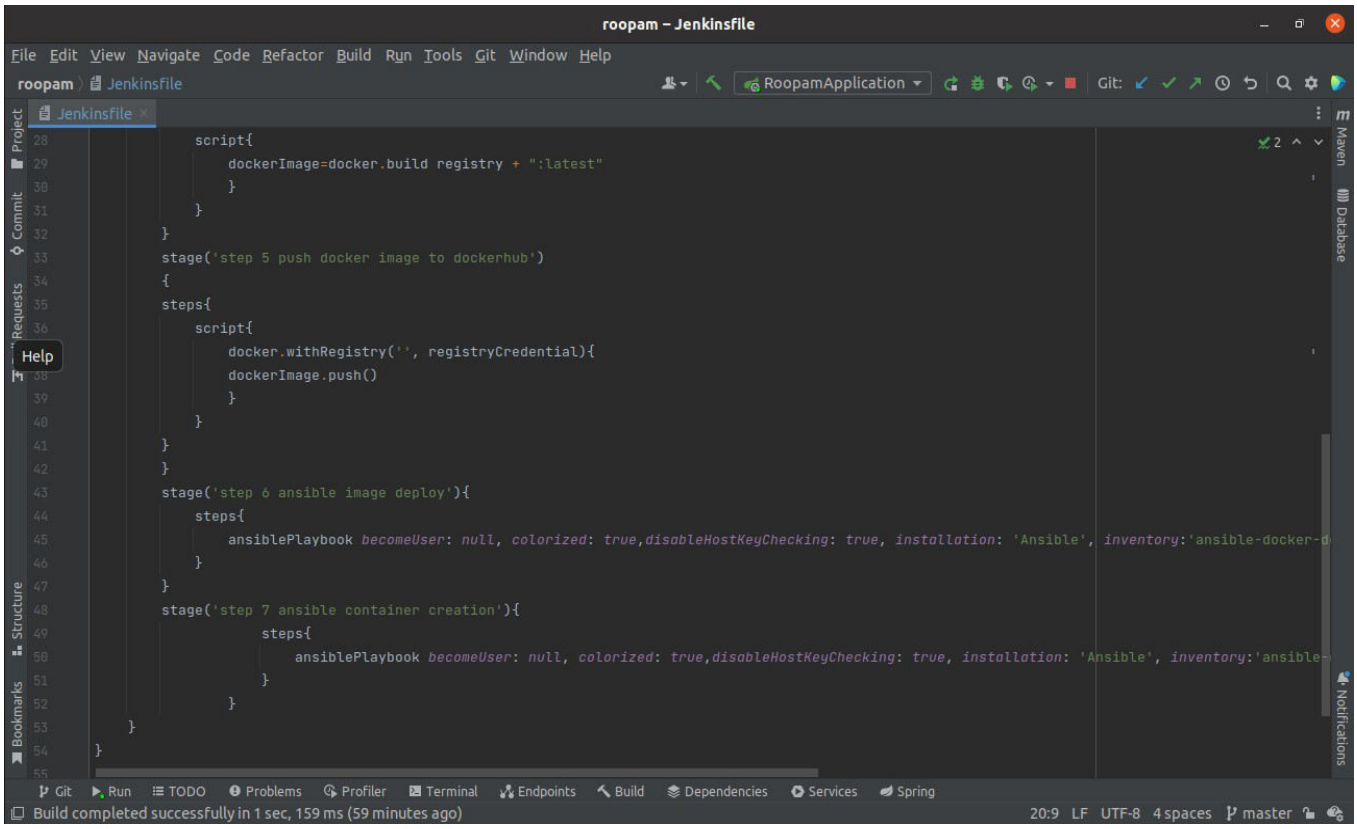


Fig : 18

Steps involved in pipeline:

1. SCM Checkout
2. Build Executable jar
3. Building Docker Image
4. Push Docker Image to DockerHub
5. Remove Unused docker images
6. Deploy Code to Host

10. To Build the project -> Click on build now.

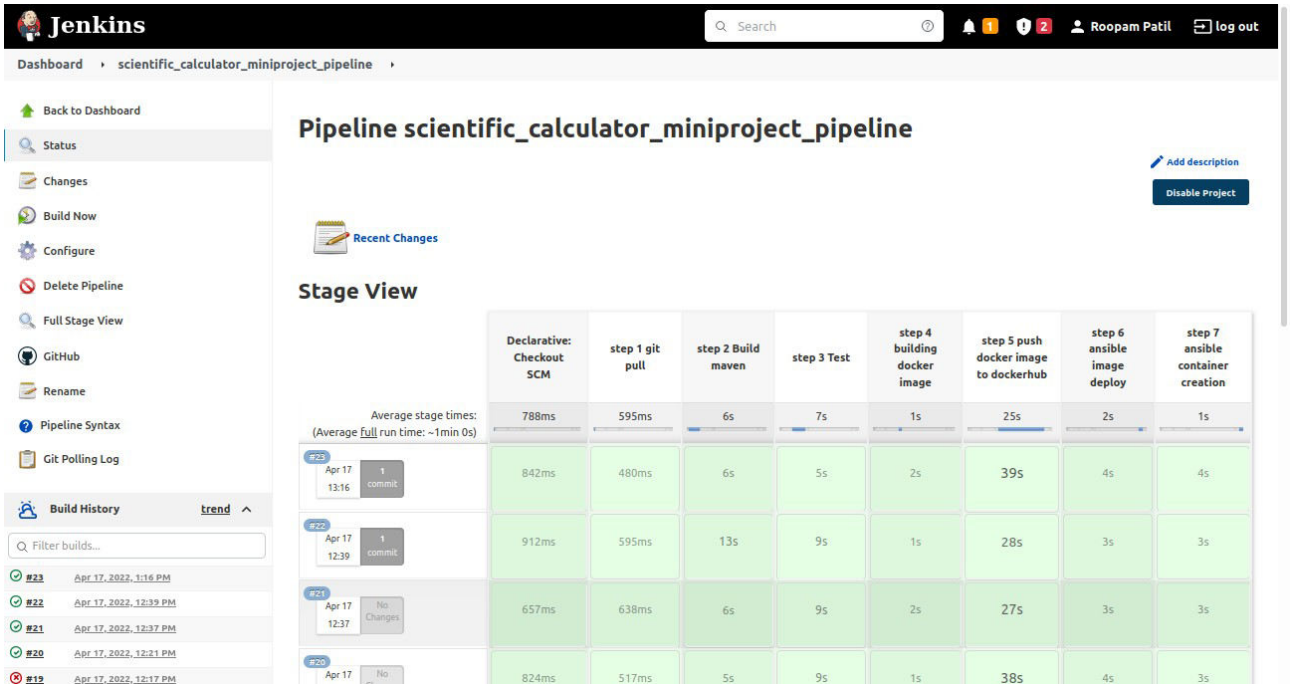


Fig : 19

Here we can see the overall status of our pipeline:
Blue dot: Pipeline succeeded, Red Dot: Pipeline failed, Black Dot: Pipeline aborted.

11.To view the detailed outline of pipeline: Click on build number then console output.

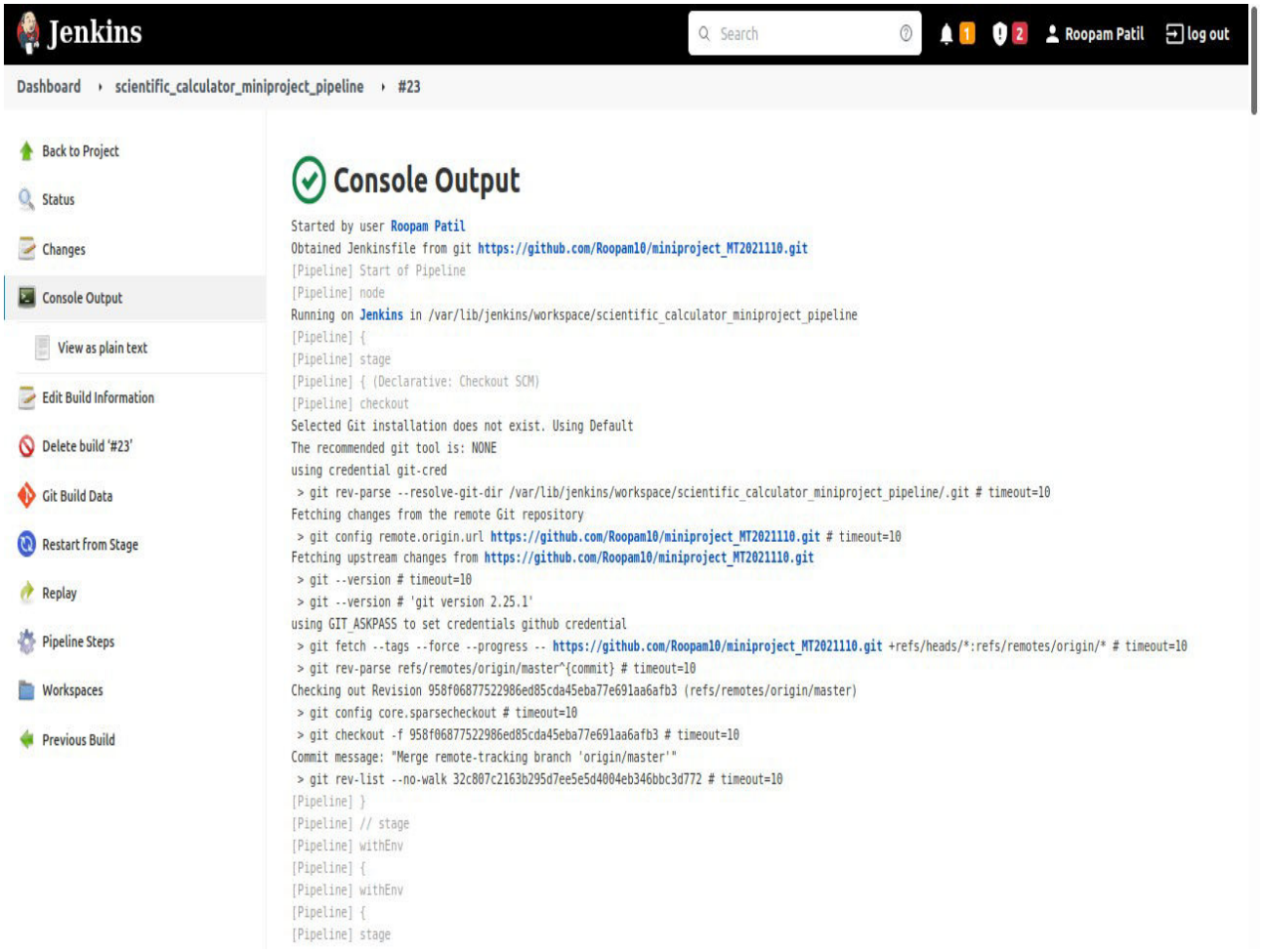

Jenkins
Search
1
2
Roopam Patil
log out
Dashboard
scientific_calculator_miniproject_pipeline
#23
Back to Project
Status
Changes
Console Output
View as plain text
Edit Build Information
Delete build '#23'
Git Build Data
Restart from Stage
Replay
Pipeline Steps
Workspaces
Previous Build
Console Output
Started by user Roopam Patil
Obtained Jenkinsfile from git https://github.com/Roopam10/miniproject_MT2021110.git
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/scientific_calculator_miniproject_pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
using credential git-cred
> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/scientific_calculator_miniproject_pipeline/.git # timeout=10
Fetching changes from the remote Git repository
> git config remote.origin.url https://github.com/Roopam10/miniproject_MT2021110.git # timeout=10
Fetching upstream changes from https://github.com/Roopam10/miniproject_MT2021110.git
> git --version # timeout=10
> git --version # 'git version 2.25.1'
using GIT_ASKPASS to set credentials github credential
> git fetch --tags --force --progress -- https://github.com/Roopam10/miniproject_MT2021110.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 958f06877522986ed85cda45eba77e691aa6afb3 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 958f06877522986ed85cda45eba77e691aa6afb3 # timeout=10
Commit message: "Merge remote-tracking branch 'origin/master'"
> git rev-list --no-walk 32c807c2163b295d7ee5e5d4004eb346bbc3d772 # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage

Fig : 20

Continuous Delivery

A deliverable in Software Engineering is an artifact that is ready to be delivered to the client. It thus is the end product of a SDLC. Continuous Delivery (CD) is the practice of generating
deliverable as soon as code changes happen. CD requires that CI pipeline be in place first. Hence, CI is a prerequisite of CD. Here, the deliverable is in the form of a docker image. The
image consists of everything that our project requires to run including OS, OpenJDK and Tomcat server. The tools used by me for creating a CD pipeline are Docker and Jenkins.

Docker

Docker is a tool designed to make it easier to create, deploy, and run applications by using containers. Containers allow a developer to package up an application with all the parts it
needs, such as libraries and other dependencies, and deploy it as one package.

To install Docker, follow the given steps:

1. Type the following commands on the terminal
    > sudo apt-get update
    > sudo apt-get install apt-transport-https ca-certificate curl gnupg-agent software-properties-common
    > curl –fsSL ttps://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -> sudo add-apt repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu
$(lsb_release -cs) stable"
    > sudo apt-get update
    > sudo apt-get docker-ce docker-ce-cli containerd.io
2.Check whether Docker has been installed or not
    >sudo docker run hello-world
3.Configure Docker to run without sudo and also give Jenkins permission to run docker commands without sudo
    >sudo groupadd docker
    > sudo usermod -aG docker cristiano
    > sudo usermod -aG docker jenkins

Here we are going to create a docker image of our application's generated jar file from Jenkins - Maven on top of Open-jdk-8 as a base image and pushing the latest along with build number wise images to Dockerhub. The steps to push the image is mentioned in Jenkins file as a stage.

• The docker file tells the Image should be built/created using openjdk 8, after that we copy the created jar file and copy it to the working directory, and specify the command that should get triggered whenever a container is getting started
• Pushed images can be found here: https://hub.docker.com/repository/docker/cristiano10/calculator_roopam

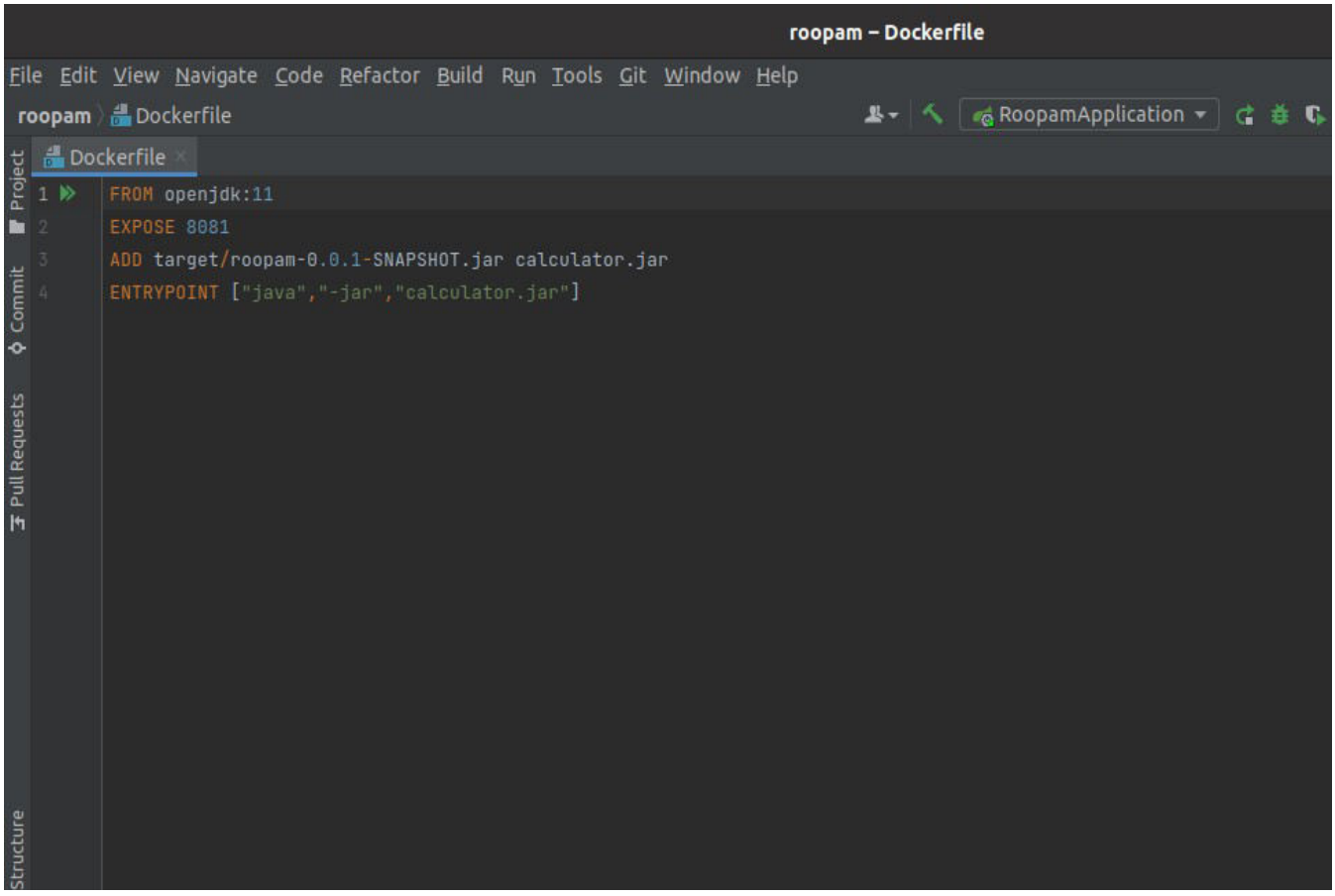


Fig : 21

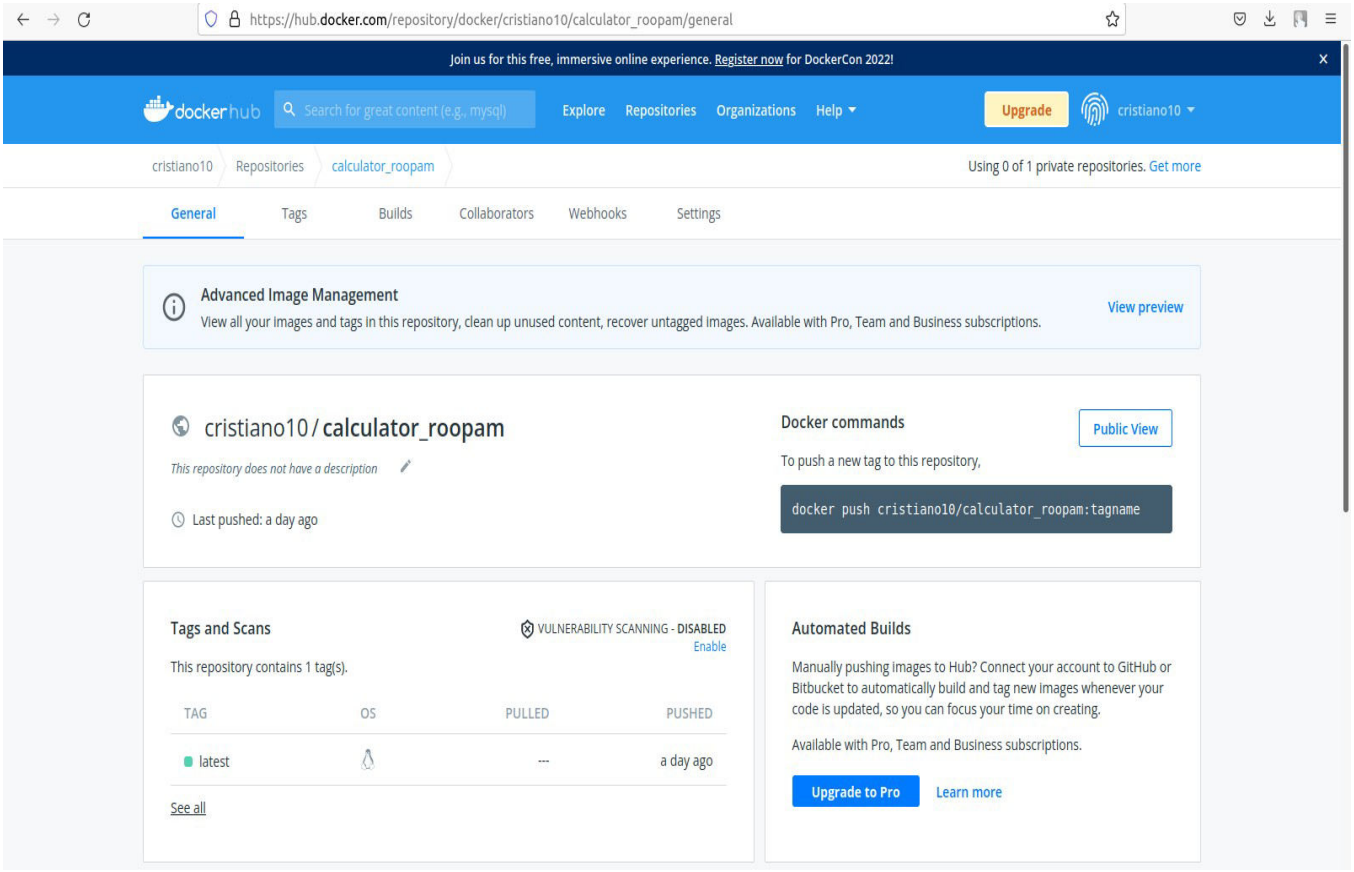
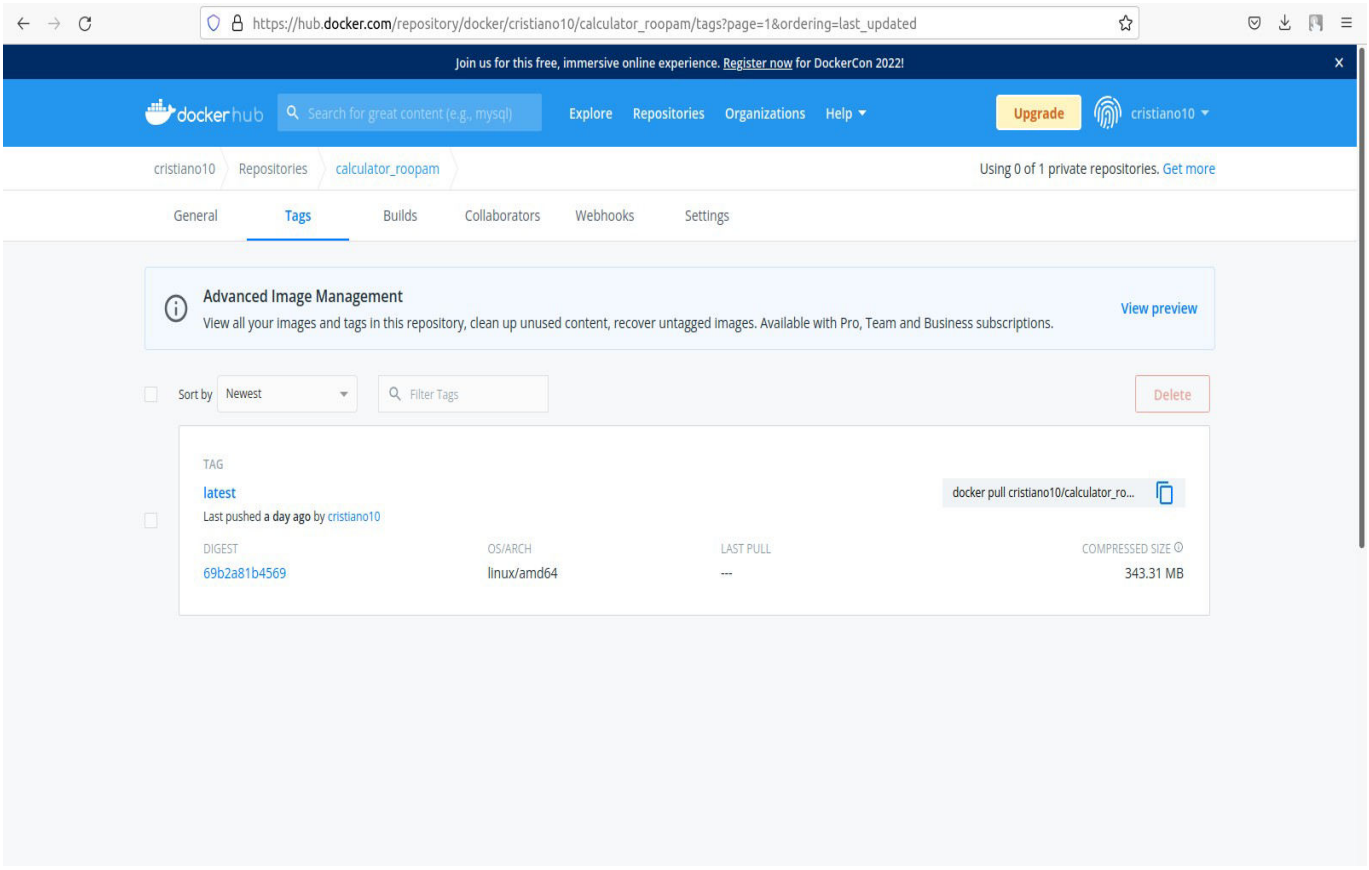

https://hub.docker.com/repository/docker/cristiano10/calculator_roopam/general
Join us for this free, immersive online experience. Register now for DockerCon 2022!
dockerhub
Search for great content (e.g., mysql)
Explore
Repositories
Organizations
Help
Upgrade
cristiano10
cristiano10
Repositories
calculator_roopam
Using 0 of 1 private repositories. Get more
General
Tags
Builds
Collaborators
Webhooks
Settings
Advanced Image Management
View all your images and tags in this repository, clean up unused content, recover untagged images. Available with Pro, Team and Business subscriptions.
View preview
cristiano10 / calculator_roopam
This repository does not have a description
Last pushed: a day ago
Docker commands
Public View
To push a new tag to this repository,
docker push cristiano10/calculator_roopam:tagname
Tags and Scans
VULNERABILITY SCANNING - DISABLED
Enable
This repository contains 1 tag(s).
TAG
OS
PULLED
PUSHED
latest
---
a day ago
See all
Automated Builds
Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.
Available with Pro, Team and Business subscriptions.
Upgrade to Pro
Learn more

Fig : 22


https://hub.docker.com/repository/docker/cristiano10/calculator_roopam/tags?page=1&ordering=last_updated
Join us for this free, immersive online experience. Register now for DockerCon 2022!
dockerhub
Search for great content (e.g., mysql)
Explore
Repositories
Organizations
Help
Upgrade
cristiano10
cristiano10
Repositories
calculator_roopam
Using 0 of 1 private repositories. Get more
General
Tags
Builds
Collaborators
Webhooks
Settings
Advanced Image Management
View all your images and tags in this repository, clean up unused content, recover untagged images. Available with Pro, Team and Business subscriptions.
View preview
Sort by
Newest
Filter Tags
Delete
TAG
latest
Last pushed a day ago by cristiano10
docker pull cristiano10/calculator_ro...
DIGEST
69b2a81b4569
OS/ARCH
linux/amd64
LAST PULL

COMPRESSED SIZE
343.31 MB

Fig : 23

Ansible

Ansible is an open-source automation tool, or platform, used for IT tasks such as configuration management, application deployment, intra-service orchestration, and provisioning.

Steps to install Ansible:

1. Install puthon 3.
2. Install ssh
   - sudo apt install openssh-server
   - ssh-keygen-t rsa
   - sudo apt update
   - sudo apt install ansible

3.We need to allow Jenkins user to login to the remote server and execute the specified playbook on that remote server, we need to generate ssh-keys. So, switch to Jenkins user by running sudo su jenkins. After that run the ssh-keygen –t rsa command to generate our remote server key.

4.Authenticate and authorise Jenkins and remote server using ssh-copy-id cristiano@localhost. This will copy the remote user's ssh keys to the known hosts in our Jenkins ssh directory and now Jenkins will be able login to the remote server without any need of the password, and we can run our ansible playbook without any password.

An Ansible playbook is a blueprint of automation tasks—which are complex IT actions executed
with limited or no human involvement. Ansible playbooks are executed on a set, group, or
classification of hosts, which together make up an Ansible inventory.

For our project we are going to specify following steps in the playbook:
1. Pulling the Scientific Calculator Image
2. Removing previous container.
3. Creating new Container using the pulled image
4. Remove unused and unnecessary Images

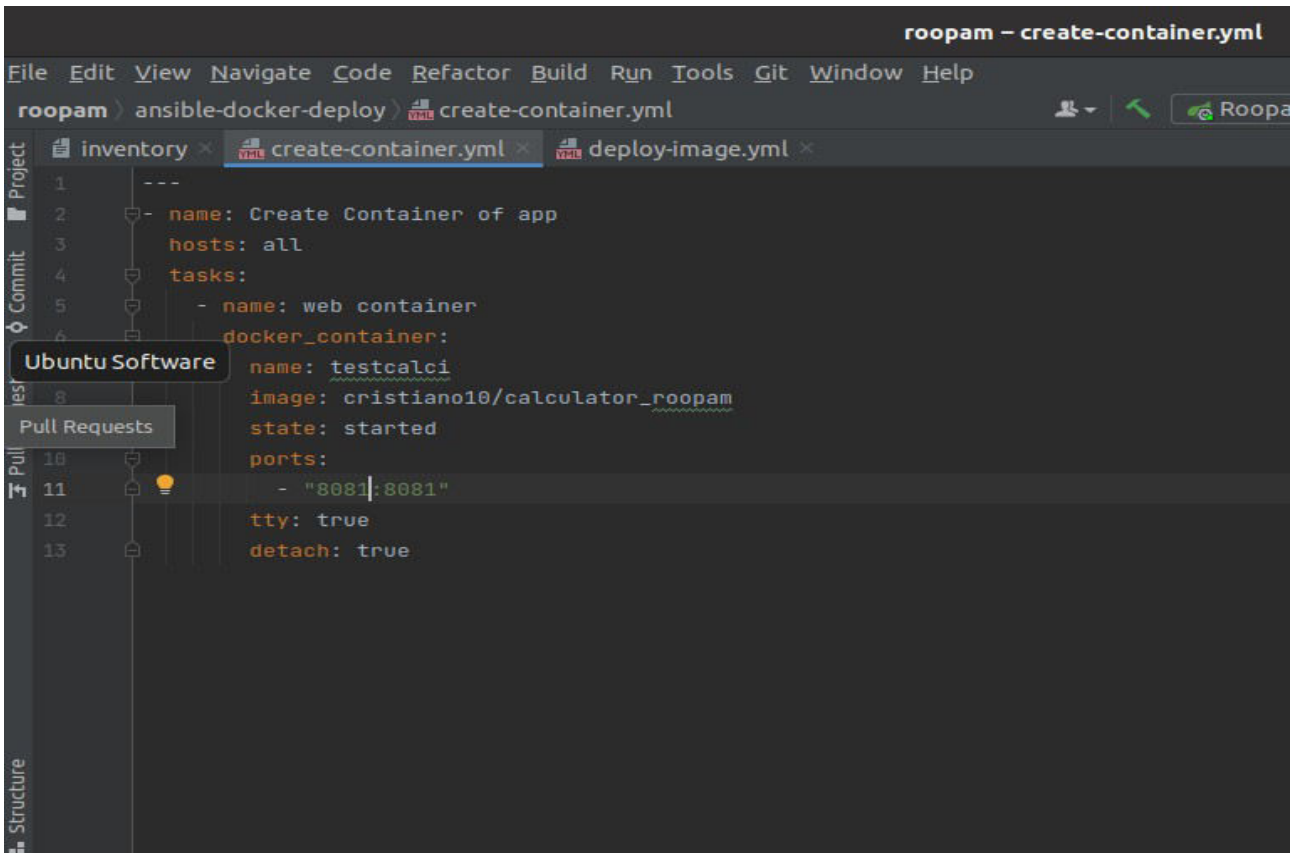


Fig : 24

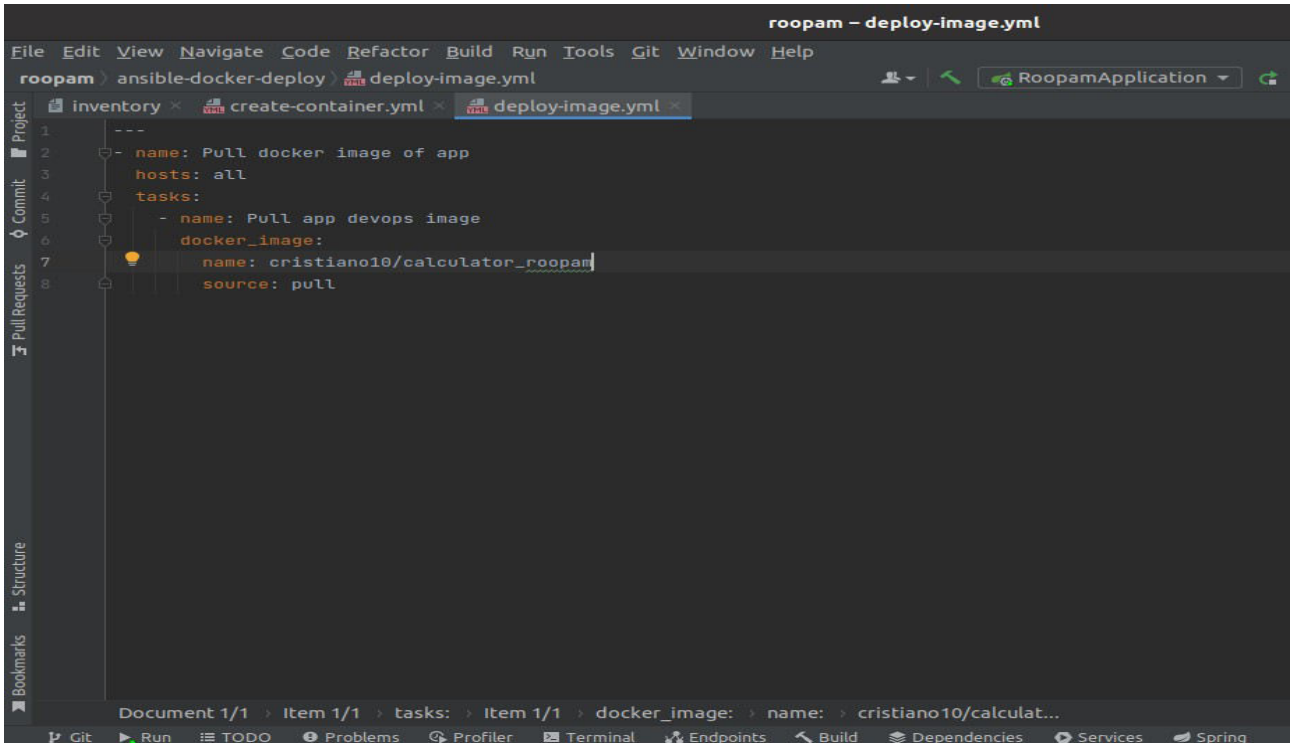


Fig : 25

When we are creating a container using docker image we are applying volume mapping so that the logs that are getting generated using our application should also be available in our dockerv host machine.

Host Details:

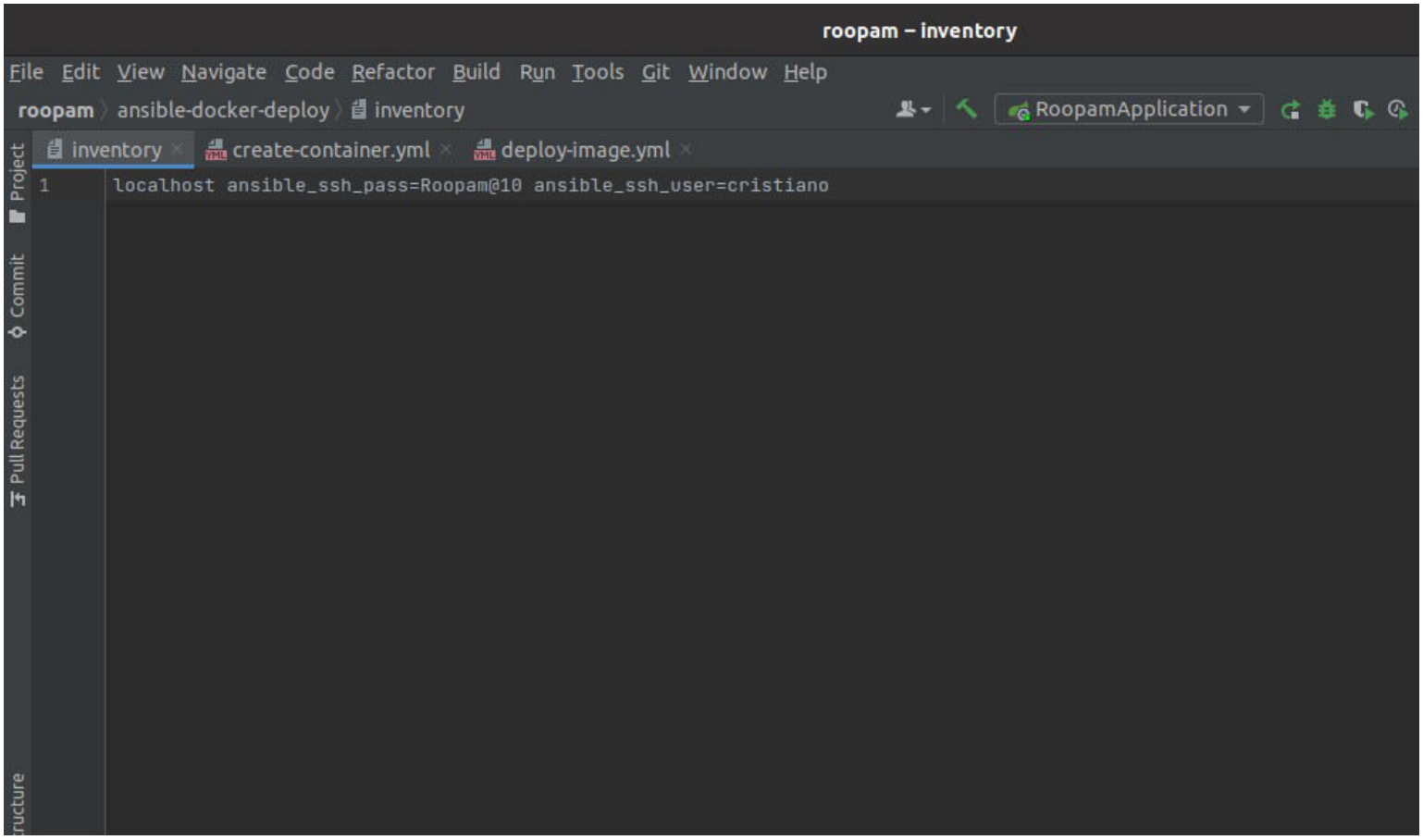


Fig : 26

```
PLAY [Pull Docker image of app] ************************************************

TASK [Gathering Facts] *********************************************************
[0;32mok: [localhost][0m
[0;32m[0m
TASK [Pull app devops image] ***************************************************
[0;32mok: [localhost][0m
[0;32m[0m
PLAY RECAP *********************************************************************
[0;32mlocalhost[0m                 : [0;32mok=2   [0m changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (step 7 ansible container creation)
[Pipeline] ansiblePlaybook
[scientific_calculator_miniproject_pipeline] $ ansible-playbook ansible-docker-deploy/create-container.yml -i ansible-docker-deploy/inventory

PLAY [Create Container of app] *************************************************

TASK [Gathering Facts] *********************************************************
[0;32mok: [localhost][0m
[0;32m[0m
TASK [web container] ***********************************************************
[0;33mchanged: [localhost][0m
[0;33m[0m
PLAY RECAP *********************************************************************
[0;33mlocalhost[0m                 : [0;32mok=2   [0m [0;33mchanged=1   [0m unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Fig : 27

After successful execution of the Ansible playbook pipeline stage: A new container got
generated at the specified host.

Continuous Monitoring

Continuous Monitoring is used to measure the performance and availability of application services. It is a process to monitor and identify compliance issues and security risks throughout
each phase of DevOps and IT operations lifecycles. Continuous Monitoring and observability can be considered as a final step of the DevOps pipeline. This is one of the most crucial steps in
a DevOps lifecycle and will help to achieve true efficiency and scalability.

In this project we have used ELK Stack to monitor the application on various fronts.

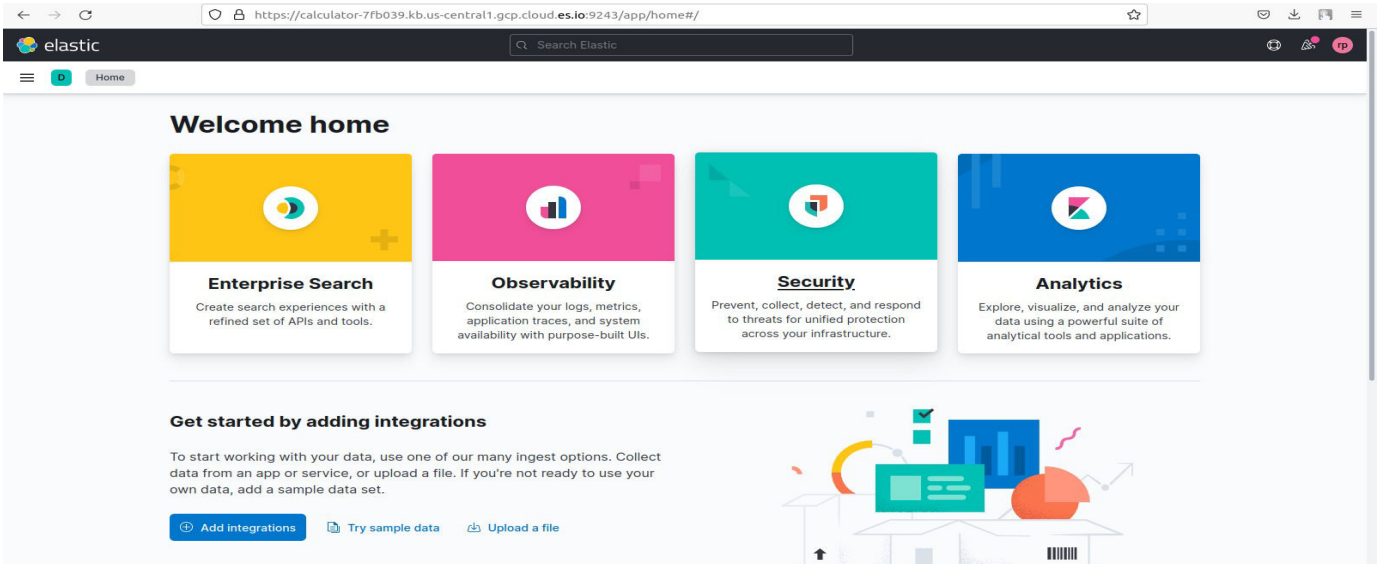
ELK Stack

"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.

Visualizing Logs in Kibana :

1. We need to go to Kibana home page: "https://www.elastic.co/what-is/elk-stack"

Fig : 28



Select your log file:

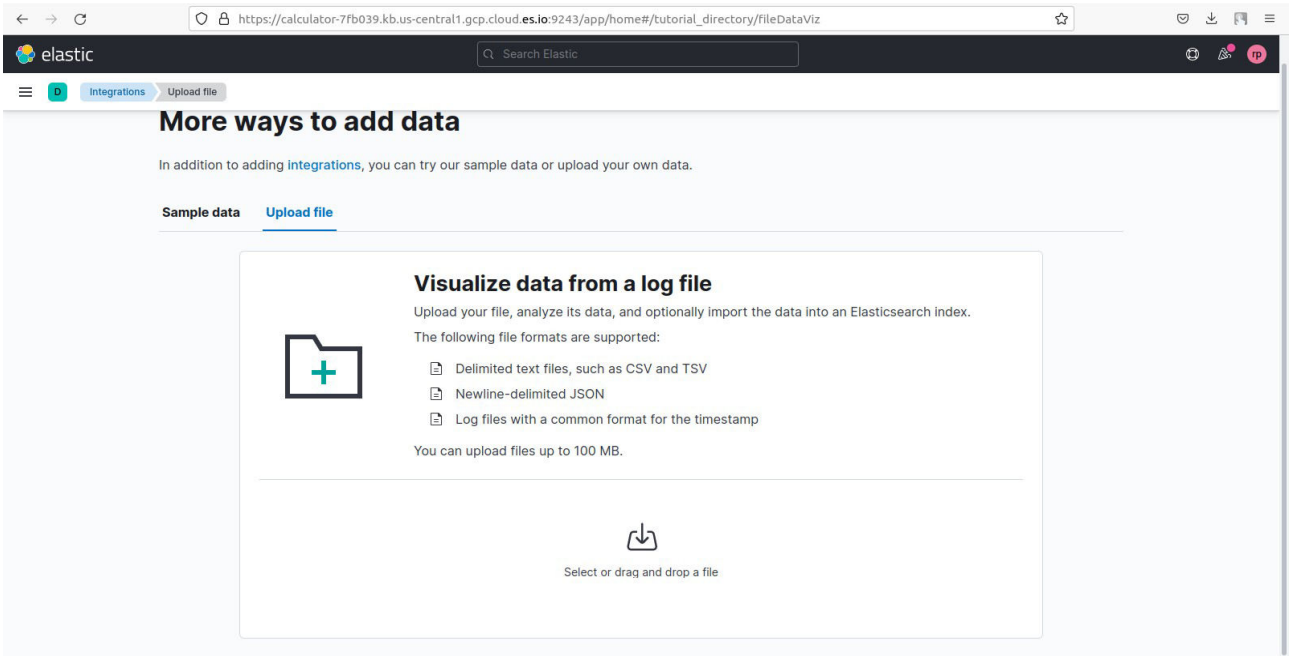

https://calculator-7fb039.kb.us-central1.gcp.cloud.es.io:9243/app/home#/tutorial_directory/fileDataViz
elastic
Search Elastic
Integrations
Upload file
More ways to add data
In addition to adding integrations, you can try our sample data or upload your own data.
Sample data
Upload file
Visualize data from a log file
Upload your file, analyze its data, and optionally import the data into an Elasticsearch index.
The following file formats are supported:
Delimited text files, such as CSV and TSV
Newline-delimited JSON
Log files with a common format for the timestamp
You can upload files up to 100 MB.
Select or drag and drop a file

Fig :29

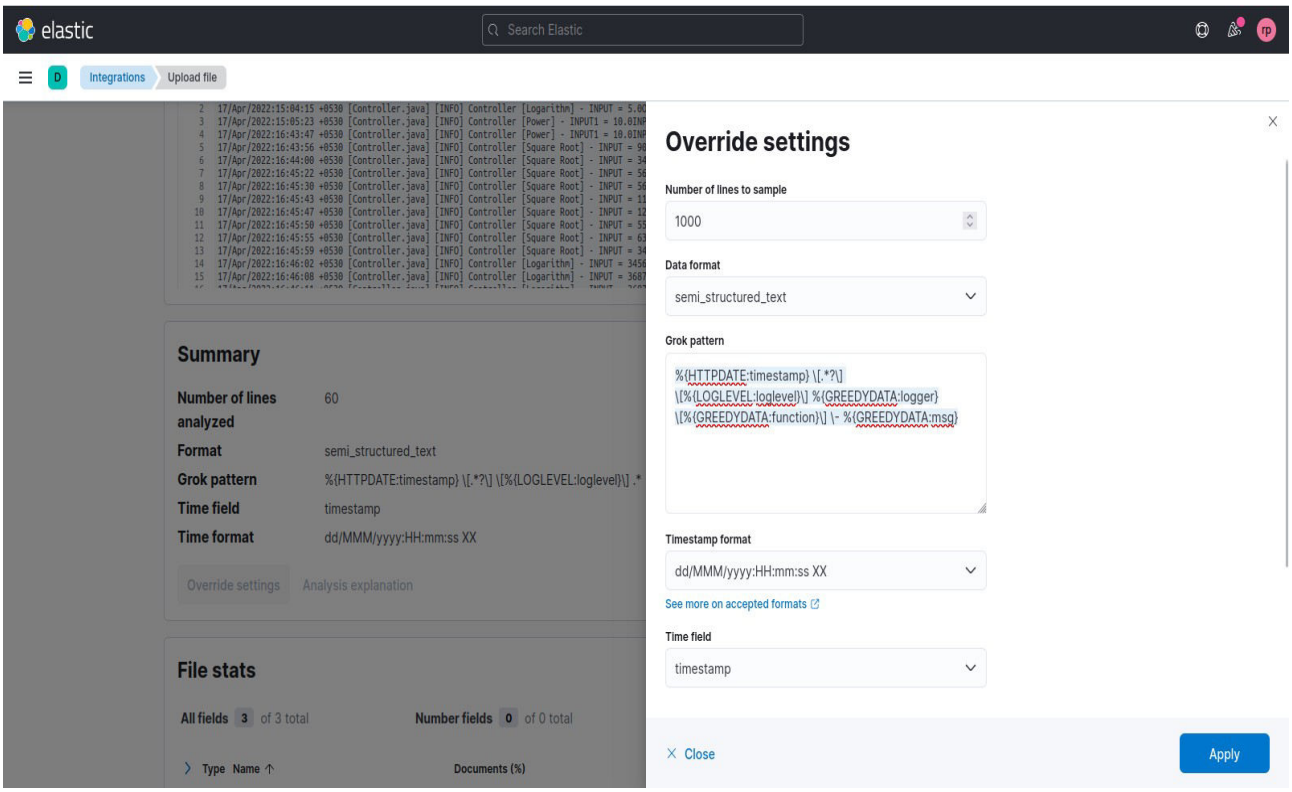

elastic
Search Elastic
Integrations
Upload file
Override settings
Number of lines to sample
1000
Data format
semi_structured_text
Grok pattern
%{HTTPDATE:timestamp} \[.*?\] \[%{LOGLEVEL:loglevel}\] %{GREEDYDATA:logger} \[%{GREEDYDATA:function}\] \- %{GREEDYDATA:msg}
Timestamp format
dd/MMM/yyyy:HH:mm:ss XX
See more on accepted formats
Time field
timestamp
Close
Apply
Summary
Number of lines analyzed 60
Format semi_structured_text
Grok pattern %{HTTPDATE:timestamp} \[.*?\] \[%{LOGLEVEL:loglevel}\] .*
Time field timestamp
Time format dd/MMM/yyyy:HH:mm:ss XX
Override settings
Analysis explanation
File stats
All fields 3 of 3 total
Number fields 0 of 0 total
Type
Name
Documents (%)

Fig : 30

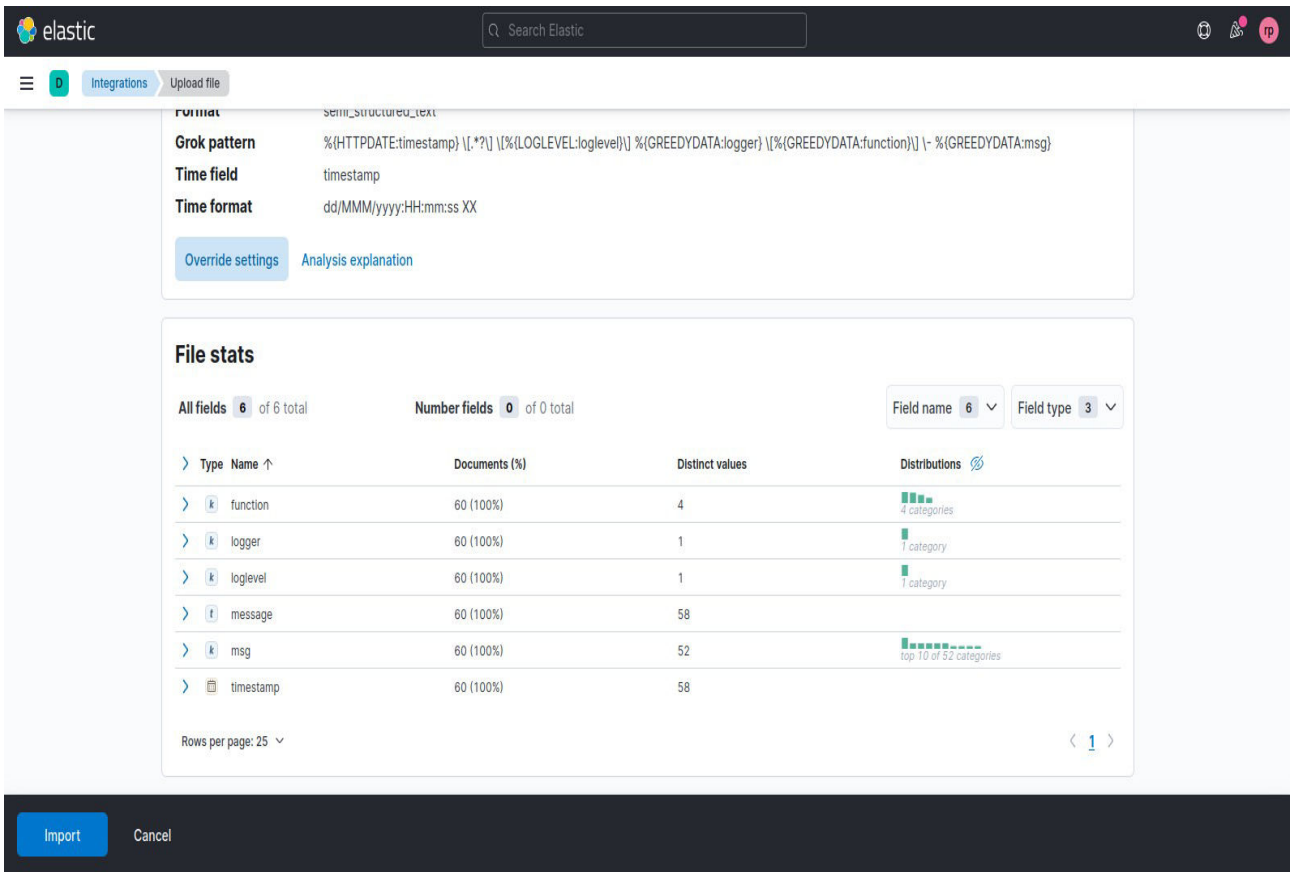

elastic
Search Elastic
Integrations
Upload file
Grok pattern
%{HTTPDATE:timestamp} \[.*?\] \[%{LOGLEVEL:loglevel}\] %{GREEDYDATA:logger} \[%{GREEDYDATA:function}\] \- %{GREEDYDATA:msg}
Time field
timestamp
Time format
dd/MMM/yyyy:HH:mm:ss XX
Override settings
Analysis explanation
File stats
All fields 6 of 6 total
Number fields 0 of 0 total
Field name 6
Field type 3
Type Name
Documents (%)
Distinct values
Distributions
function 60 (100%) 4 4 categories
logger 60 (100%) 1 1 category
loglevel 60 (100%) 1 1 category
message 60 (100%) 58
msg 60 (100%) 52 top 10 of 52 categories
timestamp 60 (100%) 58
Rows per page: 25
Import
Cancel

Fig : 31

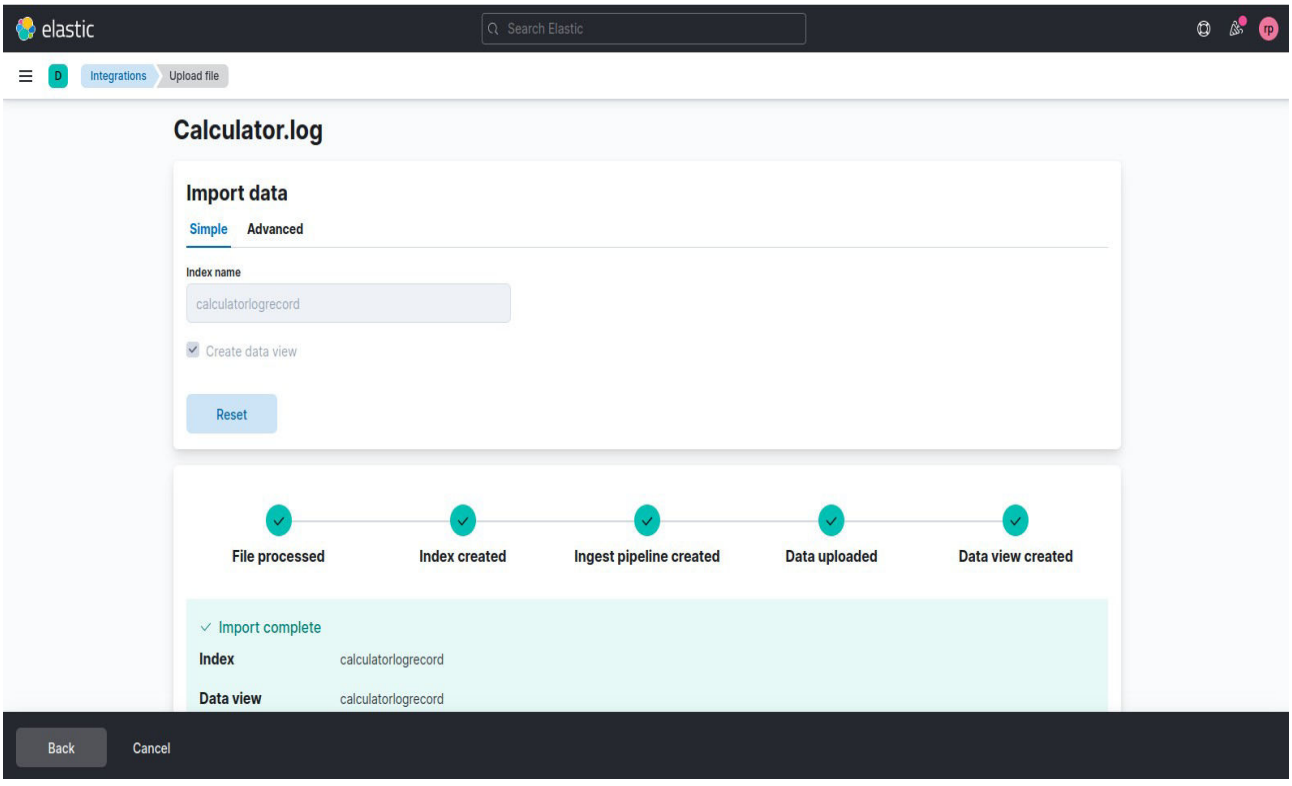

elastic
Search Elastic
Integrations
Upload file
Calculator.log
Import data
Simple
Advanced
Index name
calculatorlogrecord
Create data view
Reset
File processed
Index created
Ingest pipeline created
Data uploaded
Data view created
Import complete
Index
calculatorlogrecord
Data view
calculatorlogrecord
Back
Cancel

Fig : 32

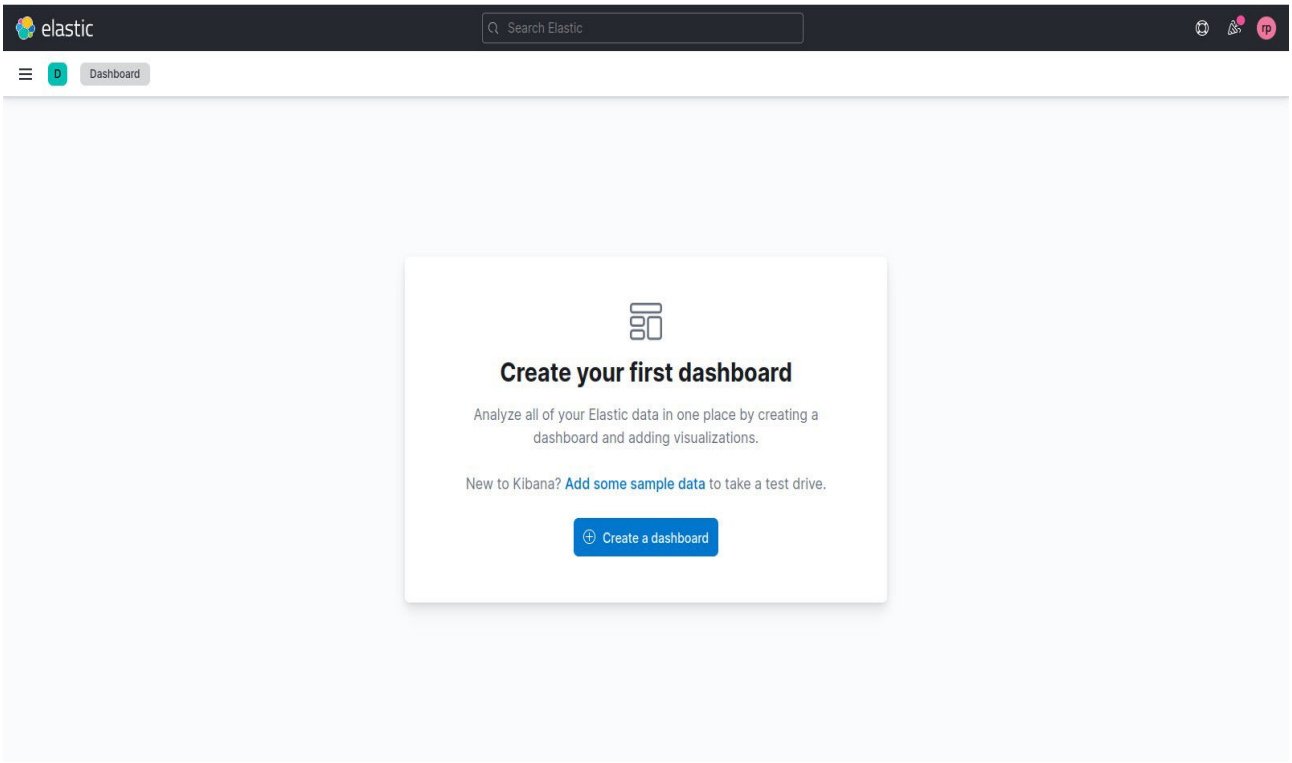

elastic
Search Elastic
Dashboard
Create your first dashboard
Analyze all of your Elastic data in one place by creating a dashboard and adding visualizations.
New to Kibana? Add some sample data to take a test drive.
Create a dashboard

Fig : 33

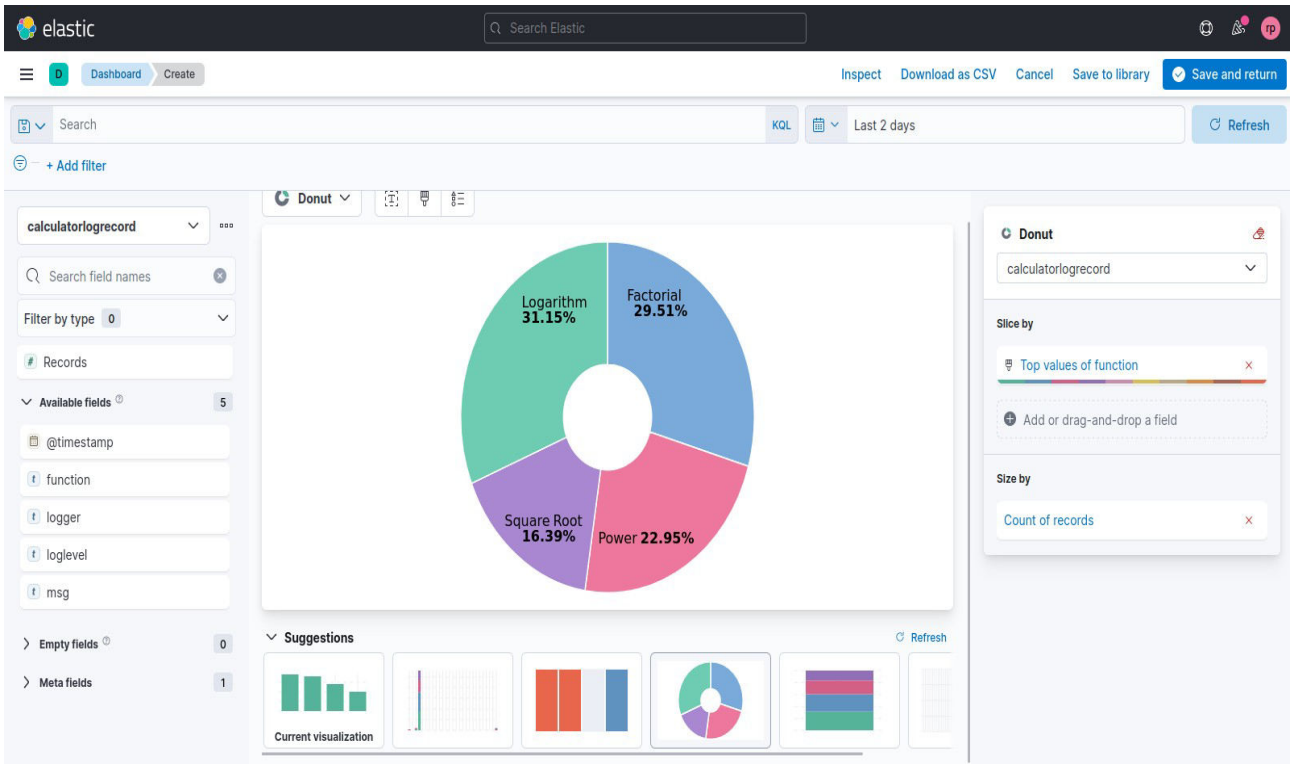

elastic
Search Elastic
Dashboard
Create
Inspect
Download as CSV
Cancel
Save to library
Save and return
Search
KQL
Last 2 days
Refresh
+ Add filter
Donut
calculatorlogrecord
Search field names
Filter by type 0
Records
Available fields 5
@timestamp
function
logger
loglevel
msg
Empty fields 0
Meta fields 1
Logarithm 31.15%
Factorial 29.51%
Square Root 16.39%
Power 22.95%
Suggestions
Refresh
Current visualization
Donut
calculatorlogrecord
Slice by
Top values of function
Add or drag-and-drop a field
Size by
Count of records

Fig :34

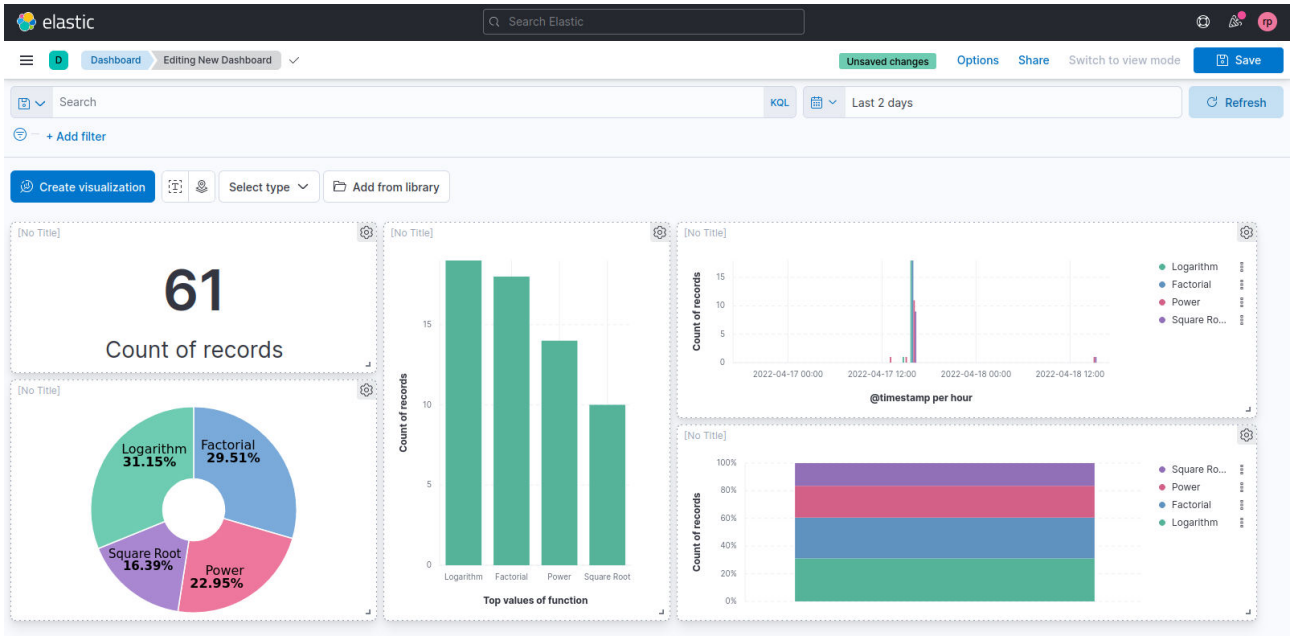


Fig 35

Here you can view all the details of your logs.

Conclusion

In this project, I automated the entire SDLC using DevOps tools and approaches. With the help of which the task of development team and operations team becomes easy as the DevOps pipeline gives the comfort of making code changes easily and also reduces the chances of post production release errors. The toolchain allows software companies to quickly integrate, build, test and deploy newer versions of their products to the production hosts. These kinds of tools and approaches are very helpful in companies where the need for daily deployment is very high.

References

1. https://docs.docker.com/get-started/
2. https://docs.ansible.com/ansible/latest/user_guide/index.html
3. https://www.jenkins.io/doc/developer/guides/
4. https://guides.github.com/activities/hello-world/
5. https://logz.io/learn/complete-guide-elk-stack/#installing-elk
6. https://stackoverflow.com/