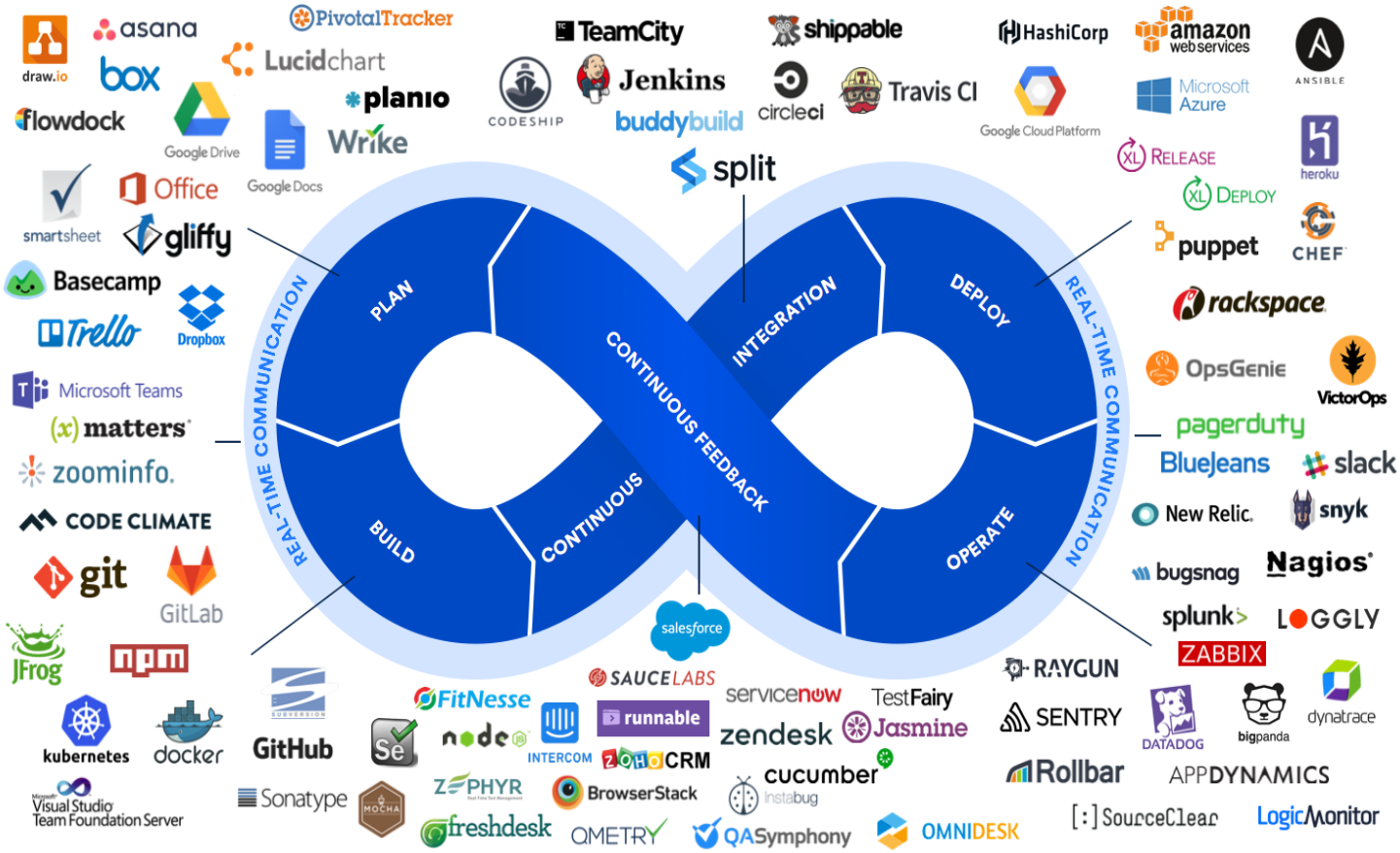
Scientific Calculator-Project Report

(Software Production Engineering)



Abhishek Garg

~MT2020021

Project Objective:

The objective of the project is to build a scientific calculator with the help of automation tools for the purpose of integration and deployment chain. This includes source control management(SCM), continuous integration, continuous deployment, configuration management,cloud based machine/container and continuous monitoring for the overall working of the code. This is achieved by using some of the devops tools that would help in the automation of the above processes. The project is about creating a scientific calculator program with operations such as natural log, square root, power function, and factorial, adding each functionality incrementally using devops tools like intellij, jenkins, ansible, docker, maven, github ,multipass and elk. The main objective of the project is to learn devops concept CI/CD/CM which is achieved by creating a Jenkins pipeline.

What is Devops:

1. DevOps is the practice of operations and development engineers participating together in the entire service lifecycle, from design through the development process to production support.
2. DevOps is also characterized by operations staff making use of many of the same techniques/tools as developers for their systems work.

Why Devops is required:

When development and operations teams are in separate silos, it's usually difficult to tell if an application is ready for operations. When development teams simply turn over an application, the operations' cycle times are extended needlessly.

With a combined development and operations team, applications are ready for use much more quickly. This is important, since companies succeed based on their ability to innovate faster than their competitors do. In fact, Kevin Murphy from Red Hat estimates that shorter development cycles translate to bringing an application to market 60 percent faster than with traditional approaches.

Tools used:

1. Github: version control system
2. Jenkins: CI/CD pipeline
3. Maven: Build tool
4. JUnit: Testing
5. Ansible: Configuration management and infrastructure as code
6. Docker, Multipass: Deployment/ containerisation
7. ELK: Continuous Monitoring

PROJECT STRUCTURE:

1. Java project with maven on Intellij
2. Add some changes to pom.xml
3. Jenkins Pipeline for continuous deployment and integration
4. Create a pipeline and write script :
    i. git clone
    ii. mvn clean install
    iii. Create docker image from docker file
    iv. Push the created docker image to the docker hub
5. Add jenkins user and other users to docker group
6. Run ansible script to push code on node machines

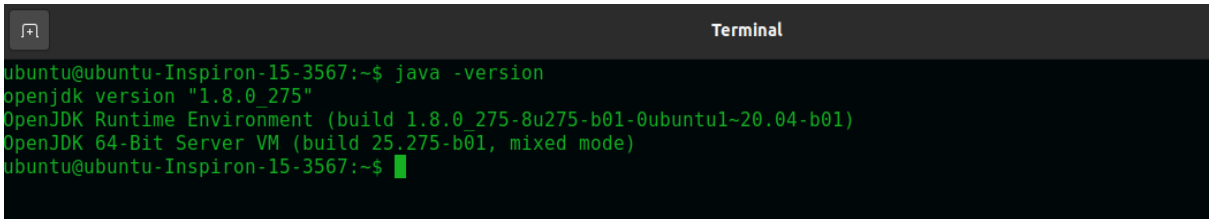
HOW I DID- Workflow:

- ❏ Setup Java environment for the project
- ❏ Using Intellij as IDE, created a maven web-app project
- ❏ Installed git and linked the project with github
- ❏ Using Github as a repository to store the project, added the github repository as an origin
- ❏ Create a jenkins pipeline that includes the task of git clone, maven build, create docker image, push image to dockerhub and pull images to node machines
- ❏ Install Multipass and launch a ubuntu machine(20.04)
- ❏ Configure SSH Connection in the newly launched Multipass VM
- ❏ Update the inventory with the IP address of the VM
- ❏ Sign up in Dockerhub in order to push the images to cloud
- ❏ Build the Jenkins Pipeline and wait for the successful running of all the stages in the pipeline
- ❏ The docker image is now pulled in the new VM then the user is able to run the scientific calculator code in all the node machines.

Environment Setup for Cloud/Node machine:

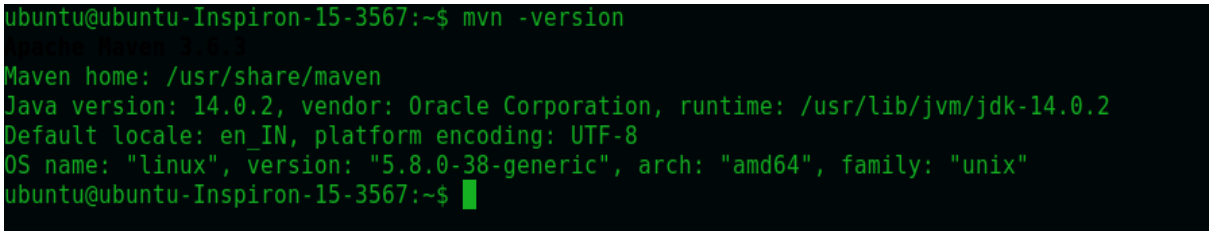
1. Installing Java:

```
ubuntu@ubuntu-Inspiron-15-3567:~$ sudo apt-get update

ubuntu@ubuntu-Inspiron-15-3567:~$ sudo apt install
openjdk-8-jre-headless

ubuntu@ubuntu-Inspiron-15-3567:~$ export
JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64

ubuntu@ubuntu-Inspiron-15-3567:~$ export
PATH=$PATH:$JAVA_HOME/bin
```



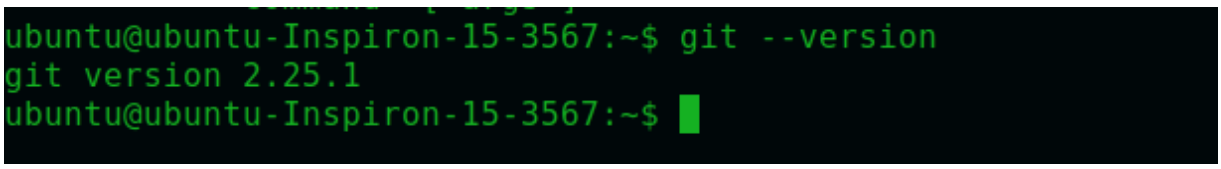
2. Maven Installation:

```
~$ sudo apt-get update

~$ sudo apt-get install maven
```



3. Installing Git:

```
~$ sudo apt-get update

~$ sudo apt-get install git
```



4. Installing Jenkins:

~$ wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -

~$ sudo sh -c 'echo deb https://pkg.jenkins.io/debian binary/ > /etc/apt/sources.list.d/jenkins.list'

~$ sudo apt-get update

~$ sudo apt-get install jenkins

```
ubuntu@ubuntu-Inspiron-15-3567:~$ systemctl start jenkins
ubuntu@ubuntu-Inspiron-15-3567:~$ systemctl status jenkins
● jenkins.service - LSB: Start Jenkins at boot time
     Loaded: loaded (/etc/init.d/jenkins; generated)
     Active: active (exited) since Fri 2021-03-12 18:56:36 IST; 20h ago
       Docs: man:systemd-sysv-generator(8)
    Process: 28526 ExecStart=/etc/init.d/jenkins start (code=exited, status=0/SUCCESS)

Mar 12 18:56:35 ubuntu-Inspiron-15-3567 systemd[1]: Starting LSB: Start Jenkins at boot time...
Mar 12 18:56:35 ubuntu-Inspiron-15-3567 jenkins[28526]: Correct java version found
Mar 12 18:56:35 ubuntu-Inspiron-15-3567 jenkins[28526]:  * Starting Jenkins Automation Server jenkins
Mar 12 18:56:35 ubuntu-Inspiron-15-3567 su[28565]: (to jenkins) root on none
Mar 12 18:56:35 ubuntu-Inspiron-15-3567 su[28565]: pam_unix(su-l:session): session opened for user jenkins by (uid=0)
Mar 12 18:56:35 ubuntu-Inspiron-15-3567 su[28565]: pam_unix(su-l:session): session closed for user jenkins
Mar 12 18:56:36 ubuntu-Inspiron-15-3567 jenkins[28526]:    ...done.
Mar 12 18:56:36 ubuntu-Inspiron-15-3567 systemd[1]: Started LSB: Start Jenkins at boot time.
ubuntu@ubuntu-Inspiron-15-3567:~$
```

5. Installing Ansible:

Installed and configure ansible using this link https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-ansible-on-ubuntu-18-04

```
ubuntu@ubuntu-Inspiron-15-3567:~$ ansible --version
ansible 2.9.6
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/ubuntu/.ansible/plugins/modules', '/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  executable location = /usr/bin/ansible
  python version = 3.8.5 (default, Jul 28 2020, 12:59:40) [GCC 9.3.0]
ubuntu@ubuntu-Inspiron-15-3567:~$
```

6. Installing Docker and Multipass:

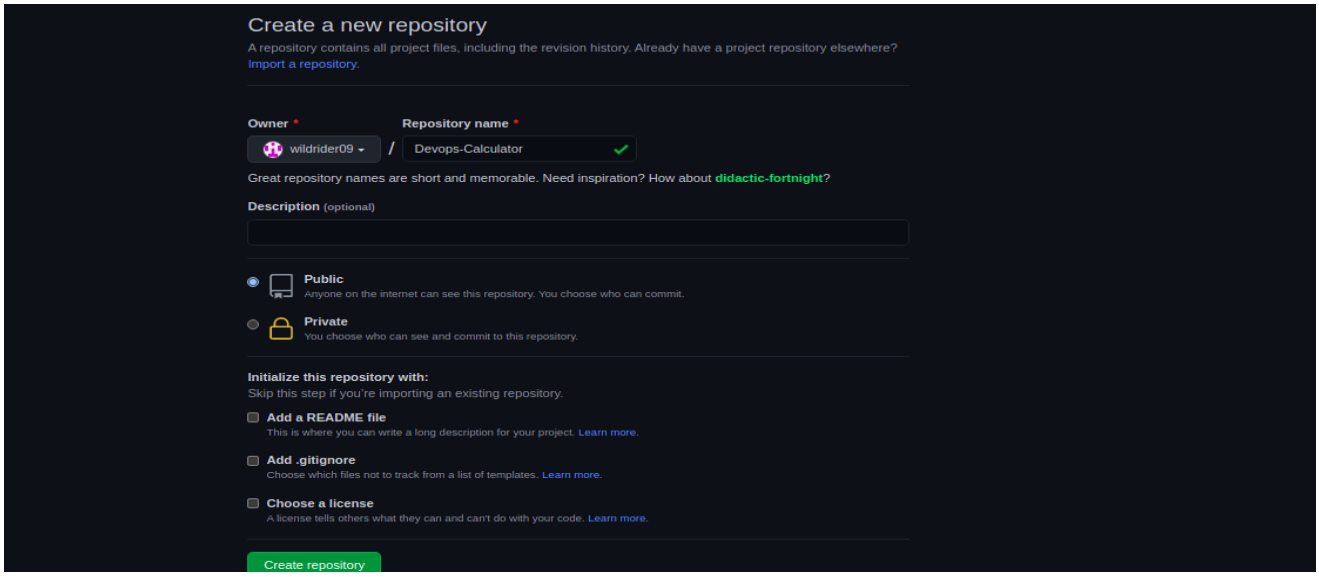
```
ubuntu@ubuntu-Inspiron-15-3567:~$ docker --version
Docker version 20.10.3, build 48d30b5
ubuntu@ubuntu-Inspiron-15-3567:~$ multipass --version
multipass  1.6.2
multipassd 1.6.2
ubuntu@ubuntu-Inspiron-15-3567:~$
```

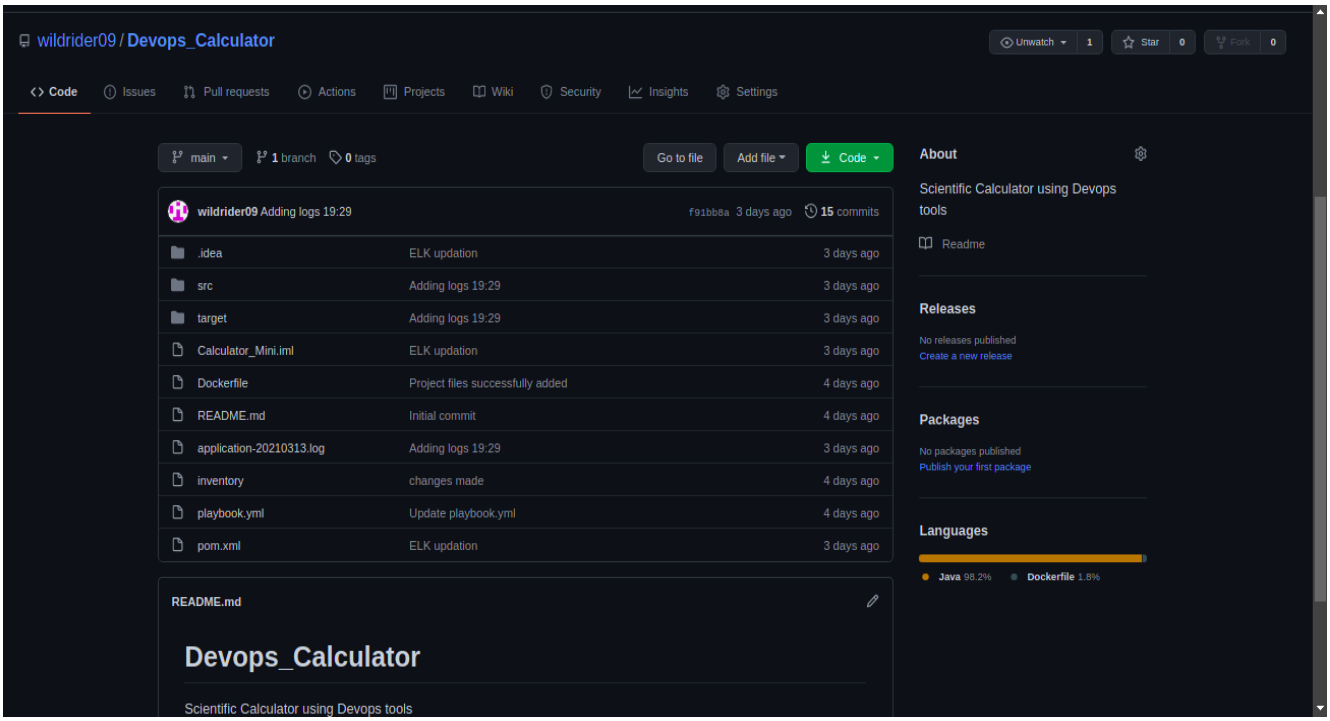
Source Control Management- Git/GitHub:

- First task is to create a new repository into a github account https://www.github.com. This includes adding repository name and description. The repository name should be unique to the signed in user. GitHub handles our code and is used to connect as input to Jenkins.
- Link for the calculator dev-ops project can be found at the following url : https://github.com/wildrider09/Devops_Calculator

- Now if you have already developed the project and want to push to yours create repository, you can initiate the current directory - git init ./ git remote add origin git push -f origin Or if you creating a new project I suggest doing a git clone *Here we are using java 8 since rundeck supports only java 8, you can configure it to java 8 with sudo update-alternatives --config java after installing it. To push the code on to repository follow following commands (only if you are on master branch, otherwise I suggest merging your branch with master first, resolving conflicts and then do git push)
- cd /directory
- Follow the commands given below:
    - git add .
    - git commit "Adding all the files of mini-calculator"
    - git config --global user.email "abhishek.garg@iiitb.org"
    - git config --global user.name "wildrider09"
    - git push origin main git config --global user.name
    - git push -u origin main
    - git branch -m main

```
cd Devops_Calculator
git add .
git commit "Adding all the files of mini-calculator"
git commit -m "Adding all the files of mini-calculator"
git config --global user.email "abhishek.garg@iiitb.org"
git config --global user.name "wildrider09"
git commit -m "Adding all the files of mini-calculator"
git push origin main git config --global user.name
git push -u origin main
```

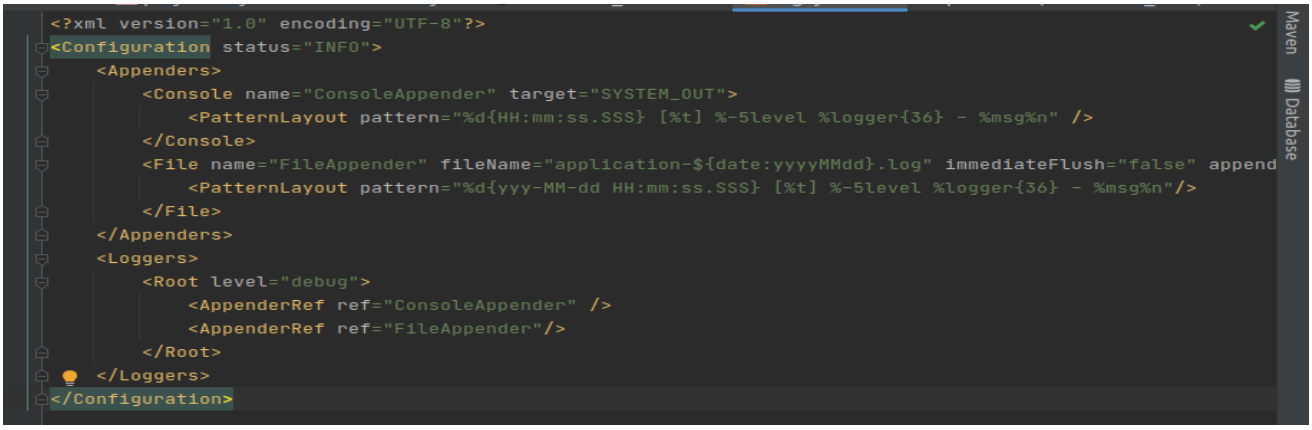


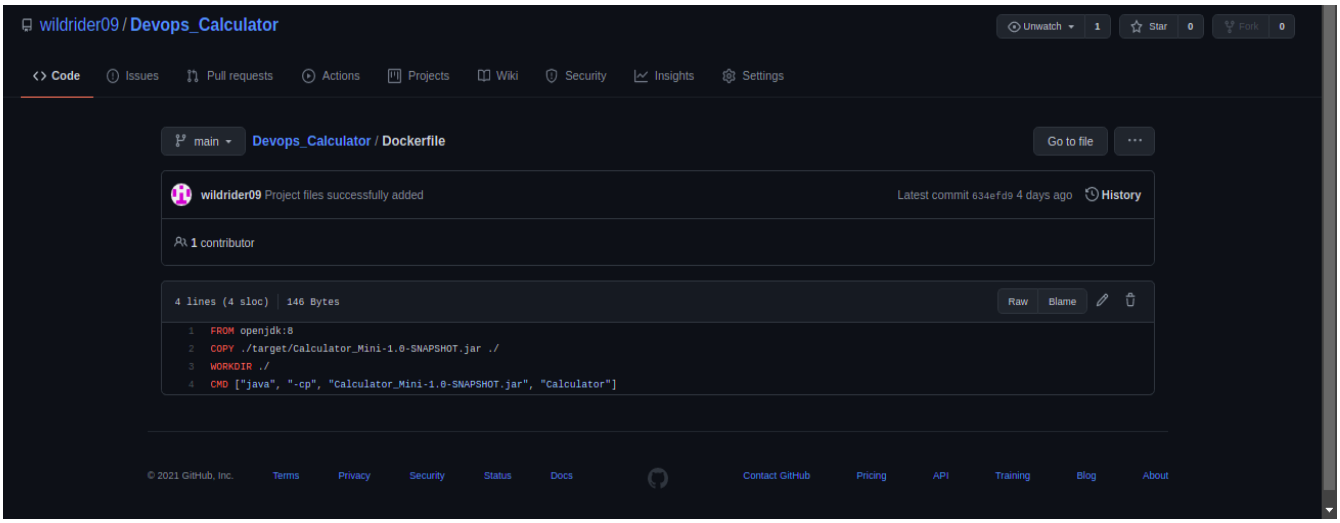


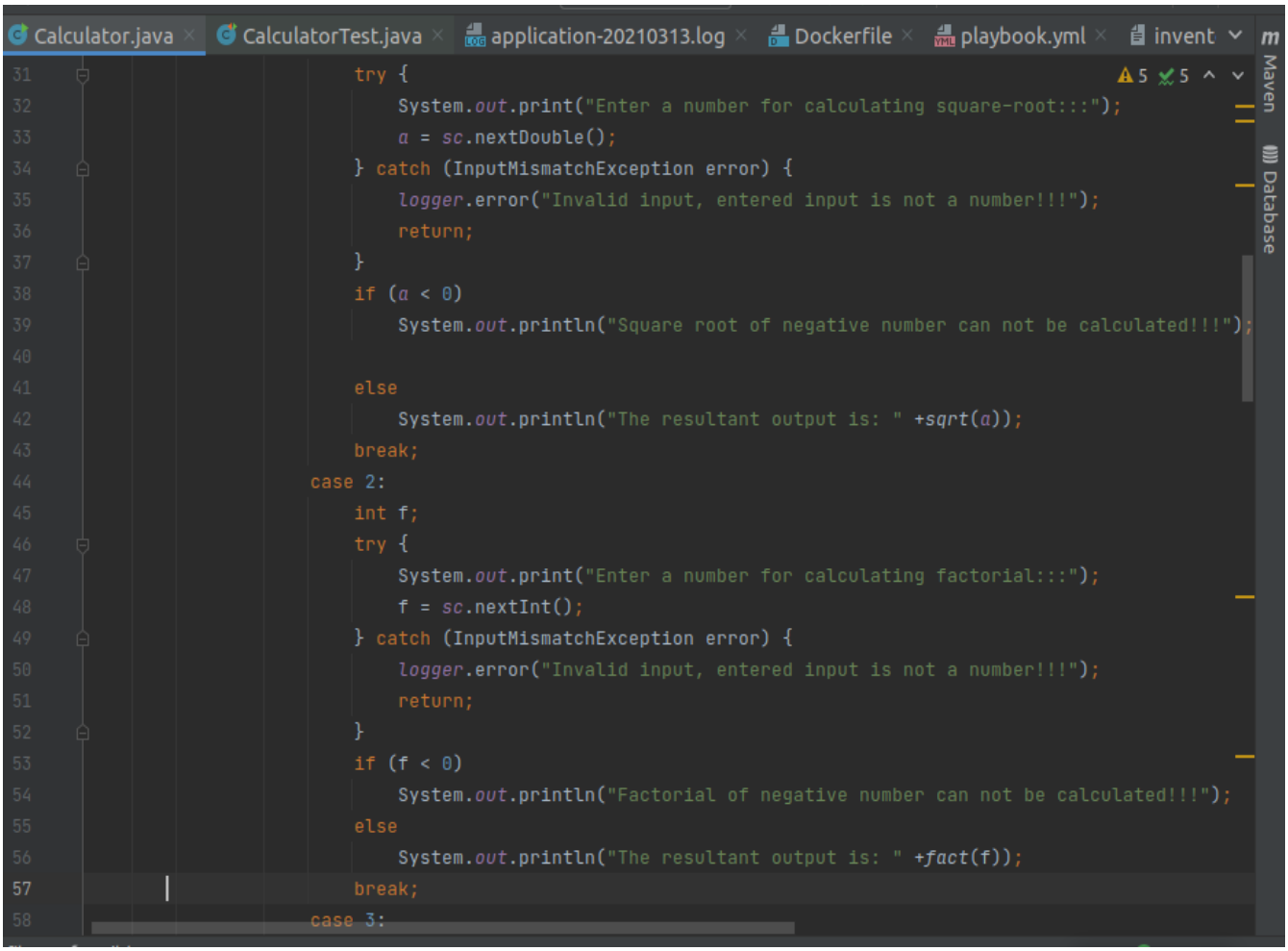
Development and Software Build- IntelliJ:

- The source code is built using Java on the IDE platform that is IntelliJ and integrating it with maven to resolve dependency problems in the code.Dependencies are external libraries which we use to integrate it within our code. So I started with a maven project and added basic functionalities of the calculator program. Also added log4j.properties which is a must file if using log4j apache jar. The log4j.properties handles projects log functionalities, be its structure or where it should be stored or append or create a new file based on date and time. I also created my own rollingappender file which creates a missing directory and file if the file or directory is not present within the specified path.

- The docker file tells the build should be built on what image, here it is openjdk 8, after that we copy the created jar file and copy it to the working directory, and what command should run when container is running. For unit testing I have included junit dependency and written assertEquals caseto test while we build our project.This test case helps to compare the actual output and expected output from the code.The necessary thing to do is to add a manifest within pom.xml which tells java where to look for main class in the project for an executable jar and since we are using dependencies.




wildrider09 / Devops_Calculator
Unwatch 1
Star 0
Fork 0
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
main
Devops_Calculator / Dockerfile
Go to file
wildrider09 Project files successfully added
Latest commit 634efd9 4 days ago
History
1 contributor
4 lines (4 sloc) 146 Bytes
Raw
Blame
1 FROM openjdk:8
2 COPY ./target/Calculator_Mini-1.0-SNAPSHOT.jar ./
3 WORKDIR ./
4 CMD ["java", "-cp", "Calculator_Mini-1.0-SNAPSHOT.jar", "Calculator"]
© 2021 GitHub, Inc.
Terms
Privacy
Security
Status
Docs
Contact GitHub
Pricing
API
Training
Blog
About


Calculator.java
CalculatorTest.java
application-20210313.log
Dockerfile
playbook.yml
invent
Maven
Database
31 try {
32 System.out.print("Enter a number for calculating square-root:::");
33 a = sc.nextDouble();
34 } catch (InputMismatchException error) {
35 logger.error("Invalid input, entered input is not a number!!!");
36 return;
37 }
38 if (a < 0)
39 System.out.println("Square root of negative number can not be calculated!!!");
40
41 else
42 System.out.println("The resultant output is: " +sqrt(a));
43 break;
44 case 2:
45 int f;
46 try {
47 System.out.print("Enter a number for calculating factorial:::");
48 f = sc.nextInt();
49 } catch (InputMismatchException error) {
50 logger.error("Invalid input, entered input is not a number!!!");
51 return;
52 }
53 if (f < 0)
54 System.out.println("Factorial of negative number can not be calculated!!!");
55 else
56 System.out.println("The resultant output is: " +fact(f));
57 break;
58 case 3:

Maven build & adding dependencies:

Maven is a project management and comprehension tool that provides developers a complete build lifecycle framework. Development team can automate the project's build infrastructure in almost no time as Maven uses a standard directory layout and a default build lifecycle.

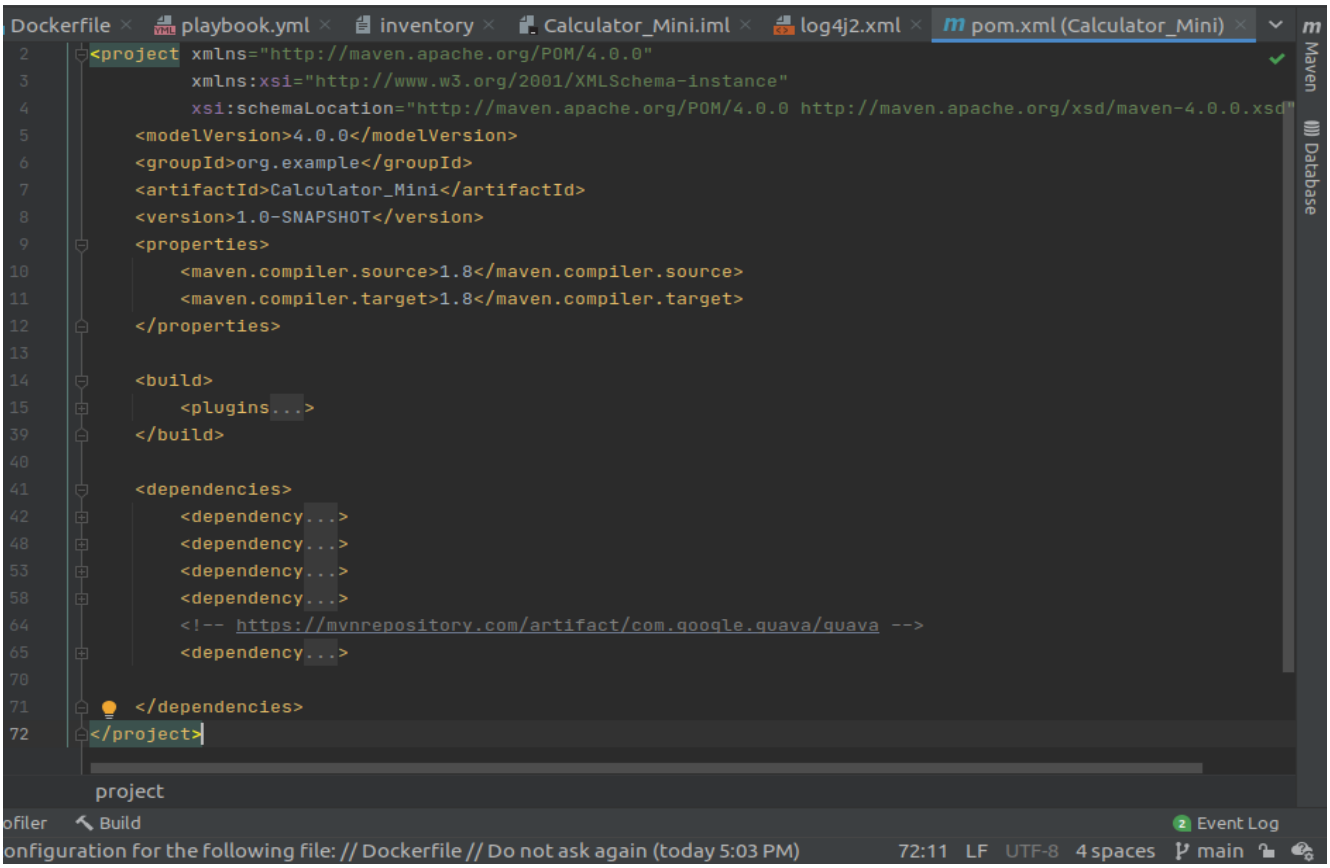
The main goal of Maven is to provide developer with the following:

- A comprehensive model for projects, which is reusable, maintainable, and easier to comprehend.
- Plugins or tools that interact with this declarative model.

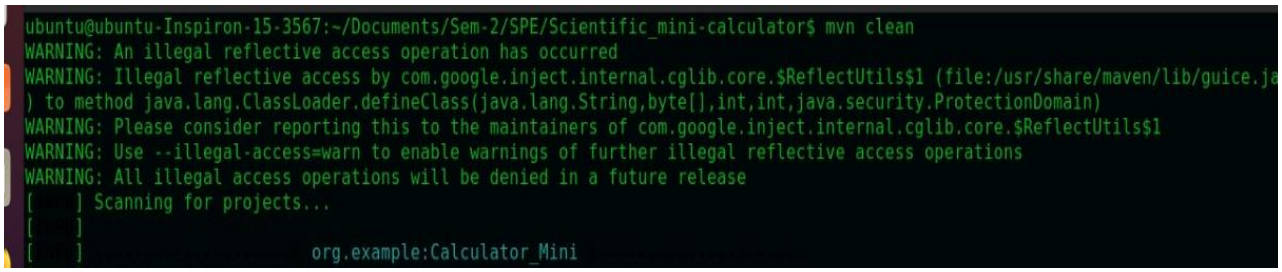
Maven project structure and contents are declared in an xml file, pom.xml, referred as Project Object Model (POM), which is the fundamental unit of the entire Maven system.

The maven enables the dependency management and helps the user to build the project to create the JAR file that can be transported to other systems to enable it to run to other systems also.

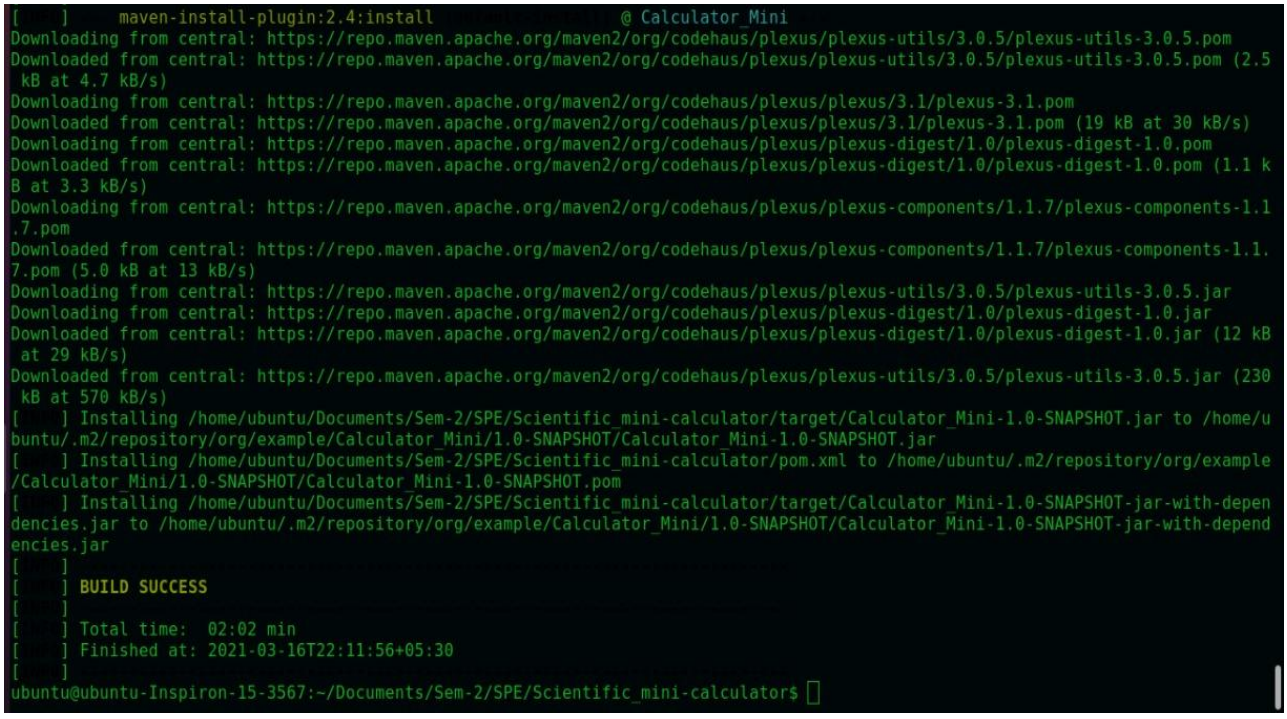
The maven maintains a pom.xml file that contains all the dependencies that are required for the project to run.



The above is the pom.xml file that contains all the dependencies that are required for the project. It also maintains the proper version of the dependency to ensure proper functioning of the applications.



mvn clean : The *clean* life cycle has only one phase named *clean* that is automatically bound to the only goal of the plugin with the same name. This goal can, therefore, be executed with the command *mvn clean*. This command cleans the maven project by deleting the target directory.



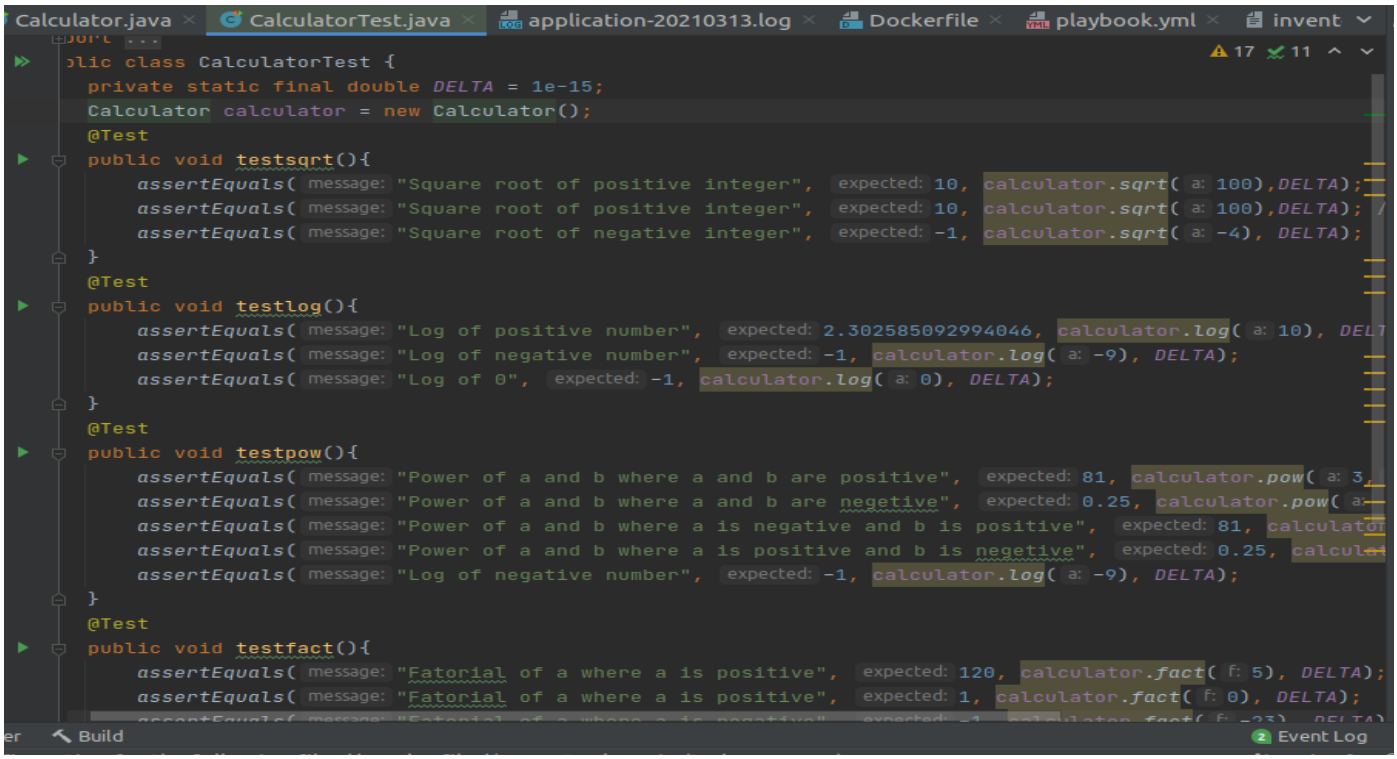
mvn install : This command builds the maven project and installs the project files (JAR, WAR, pom.xml, etc) to the local repository.It frames a dependency tree based on the project configuration pom. xml on all the sub projects under the super pom. xml (the root POM) and downloads/compiles all the needed components in a directory called.

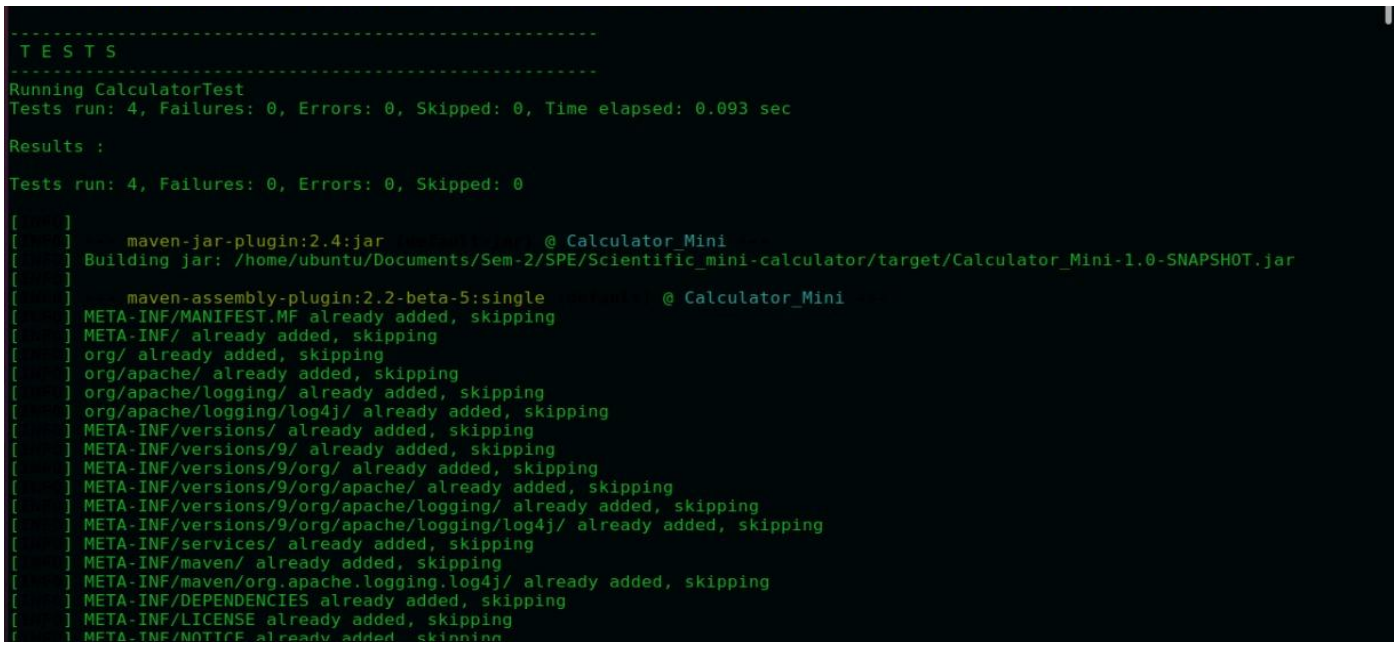
Unit Testing- Junit:

A *unit test* is a piece of code written by a developer that executes a specific functionality in the code to be tested and asserts a certain behavior or state. The percentage of code which is tested by unit tests is typically called *test coverage*. A unit test targets a small unit of code, e.g., a method or a class. External dependencies should be removed from unit tests, e.g., by replacing the dependency with a test implementation or a (mock) object created by a test framework. Unit tests are not suitable for testing complex user interfaces or component interaction. For this, you should develop integration tests.

A JUnit test is a method contained in a class which is only used for testing. This is called a Test class. To define that a certain method is a test method, annotate it with the `@Test` annotation.This method executes the code under test. You use an assert method, provided by JUnit or another assert framework, to check an expected result versus the actual result. These calls are typically called asserts or assert statements.Assert statements typically allow to define messages which are shown if the test fails. You should provide here meaningful messages to make it easier for the user to identify and fix the problem. This is especially true if someone looks at the problem, who did not write the code under test or the test code.

The below file contains the test cases that contain all types of inputs and expected outputs in order to completely test the application on the developers level. When we run **mvn install** then first junit runs all the test cases and verifies if all the test cases get passed and this ensures that the application is running fine.



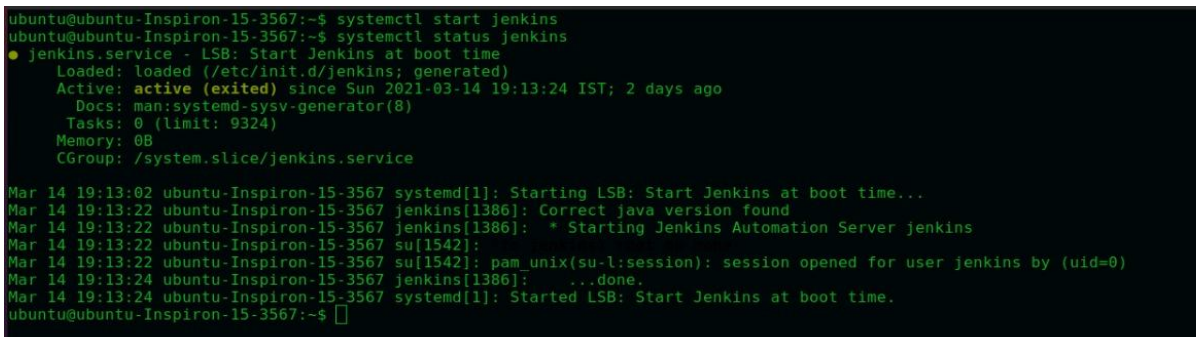


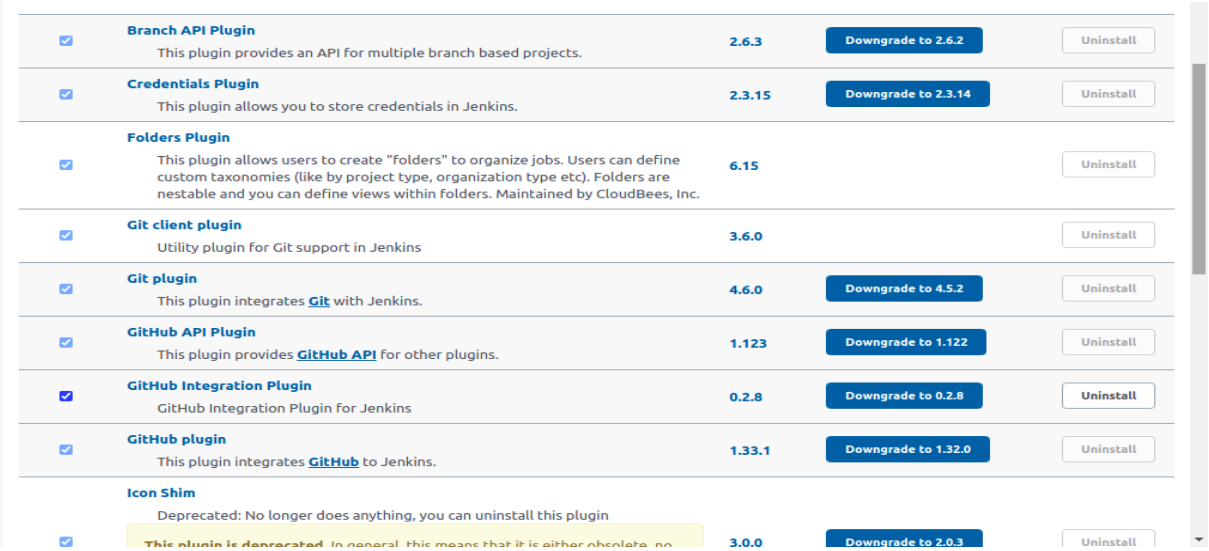
In the above image it shows the number of test cases that passed and the test cases that failed to pass. This enables the developer to work on the issues that arise in the code and makes them easy to be identified to get them fixed.

Continuous Integration - Jenkins:

The development process is complete and we have successfully pushed the code onto git, now we setup Jenkins and other plugins for it. Make sure to add Jenkins to the docker group, so that Jenkins can use docker for building docker images . sudo usermod -aG docker Jenkins, you can verify it with sudo grep jenkins /etc/gshadow We can now start Jenkins with, by command sudo systemctl start jenkins, jenkins starts at port number 8080, so login on to http://localhost:8080 onto your browser.

Now we manage the plugins of jenkins under manage plugins in manage jenkins. We download Build pipeline plugin, Docker plugin, GitHub, Maven integration plugin, Rundeck plugin. After its done downloading, jenkins will restart and now you can create a new job for jenkins, enter jenkins job name and choose pipeline as job functionality. Properties of the pipeline is it is script based and each stage of pipeline script runs one after another. Making it perfect for continuous integration and then deployment. Properties of continuous integration include SCM, unit testing and integration testing.

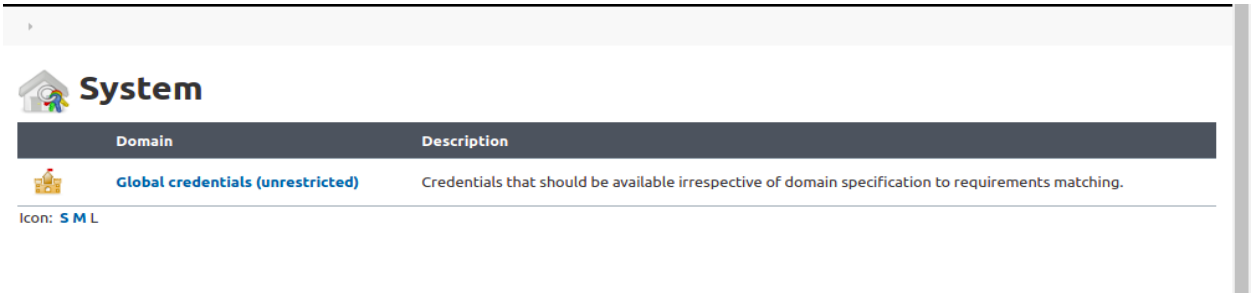


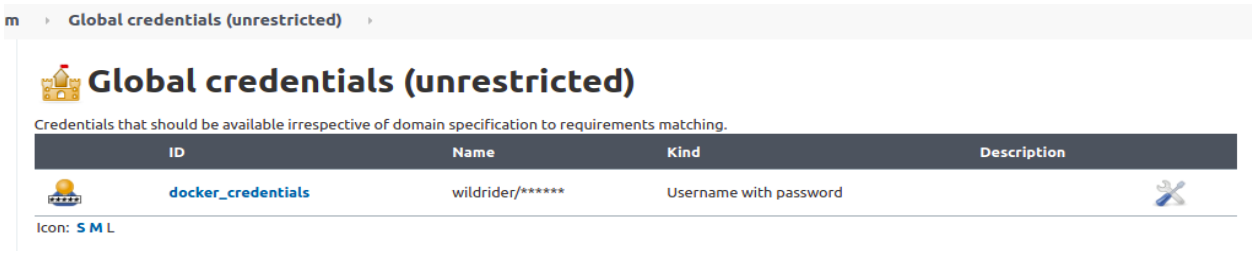


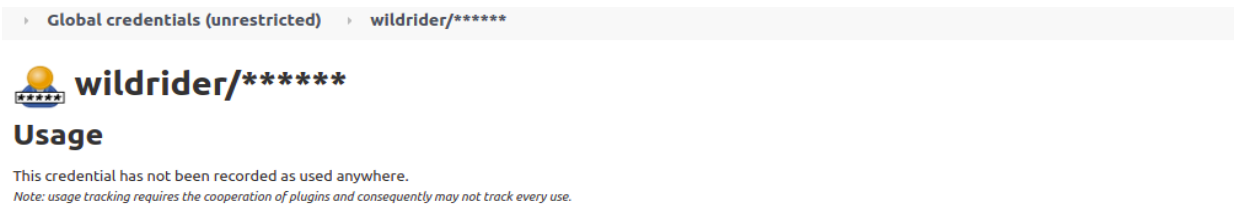
Pipeline script for Jenkins & setting credentials:

Script

```
pipeline {
    agent any

    stages {
        stage('Github Pull Code-Repository') {
            steps {
                git branch: 'main', url: 'https://github.com/wildrider09/Devops_Calculator'
            }
        }
        stage('Build the artifact') {
            steps {
                sh 'mvn clean install'
            }
        }
        stage('Creating docker image') {
            steps {
                script{
                    image_doc = docker.build "wildrider/devops_calculator:latest"
                }
            }
        }
        stage('Push docker image to DockerHUB'){
            steps{
                script{
                    docker.withRegistry('','docker_credentials'){
                        image_doc.push()
                    }
                }
            }
        }
        stage('Run ansible playbook-Pull docker image in vm and run the project'){
            steps{
                ansiblePlaybook becomeUser: null, installation: 'Ansible', inventory: 'inventory', playbook: 'playbook.yml', sudoUser: null
            }
        }
    }
}
```





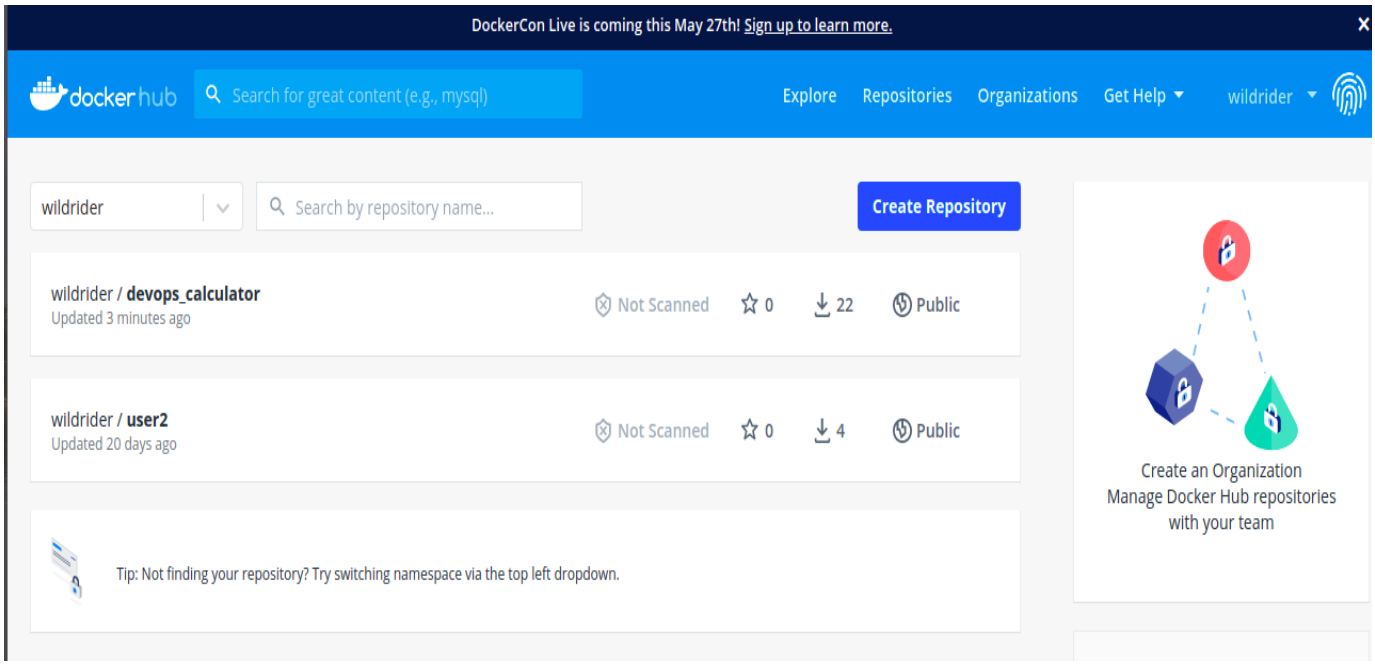


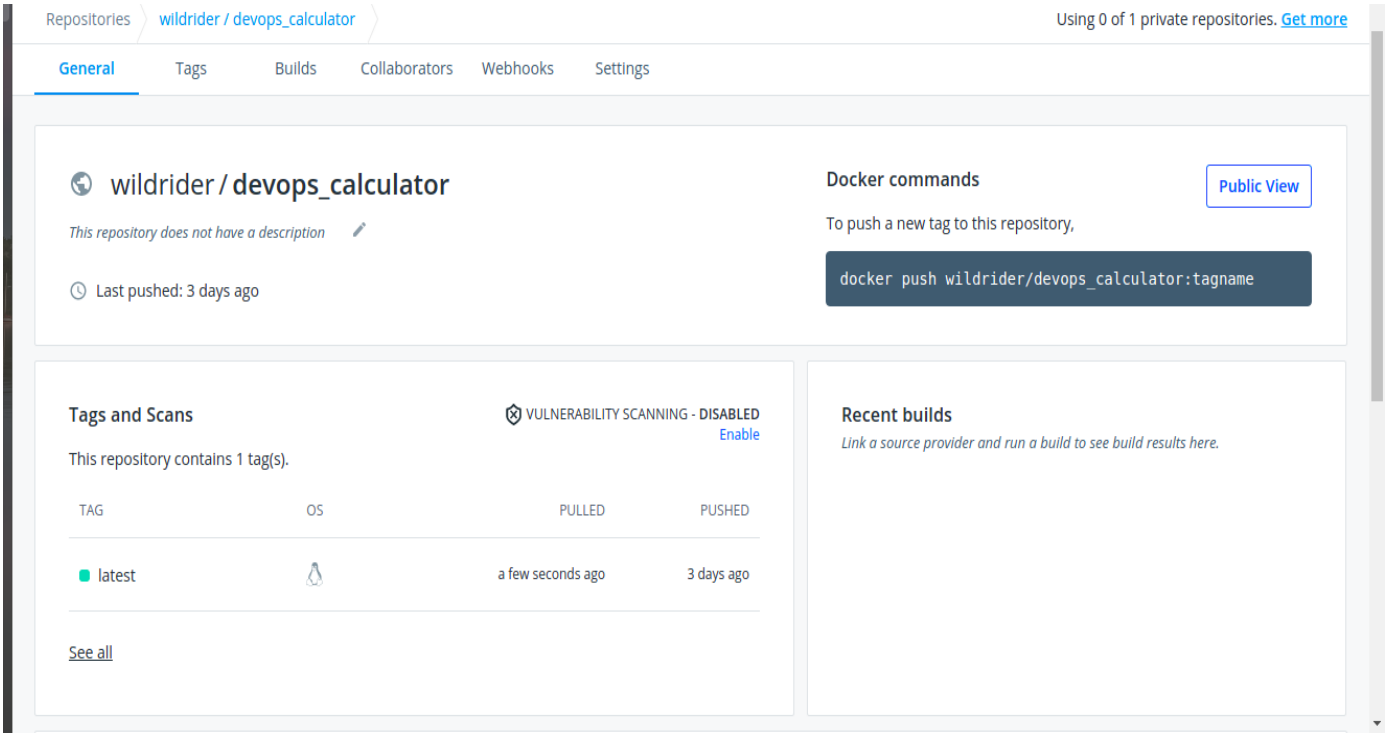
Containerization- DockerHub:

DockerHub is an open-source project launched in 2013, has helped popularize technology and has helped drive the trend towards containerization and microservices in software development that's known as cloud-native development.

Virtualization is the technique of importing a Guest operating system on top of a Host operating system. This technique was a revelation at the beginning because it allowed developers to run many operating systems in different virtual machines, all running on the same host. They have cancelled the need for extra hardware resources.Where as containerization is a form of operating system virtualization, through which it runs applications in secluded user spaces called containers, all using the same shared operating system (OS).

- ❏ So the plan is to create a docker image of the jar file created after the build process and push the latest image on to docker hub. The pushed procedure would be done after every build which would include removing previously pushed docker image and replace with latest built one.The pushed images can be found at https://hub.docker.com/repository/docker/wildrider/devops_calculator
- ❏ To run docker on your machine, either run it as root or as user where you have to enter a user to the group of docker which can be done using *sudo usermod -aG docker*.
- ❏ Run the docker on to terminal via command sudo systemctl start docker. To pull the docker image use command *docker pull*.
- ❏ To push the docker image use command *docker push /:tagname* The docker image is pushed from jenkins after the build stage and pulled on from ansible script.
- ❏ The docker image can be run using command *docker run -it --name mc_namae image_name*


DockerCon Live is coming this May 27th! Sign up to learn more.
docker hub
Search for great content (e.g., mysql)
Explore
Repositories
Organizations
Get Help
wildrider
wildrider
Search by repository name...
Create Repository
wildrider / devops_calculator
Updated 3 minutes ago
Not Scanned
0
22
Public
wildrider / user2
Updated 20 days ago
Not Scanned
0
4
Public
Tip: Not finding your repository? Try switching namespace via the top left dropdown.
Create an Organization
Manage Docker Hub repositories
with your team

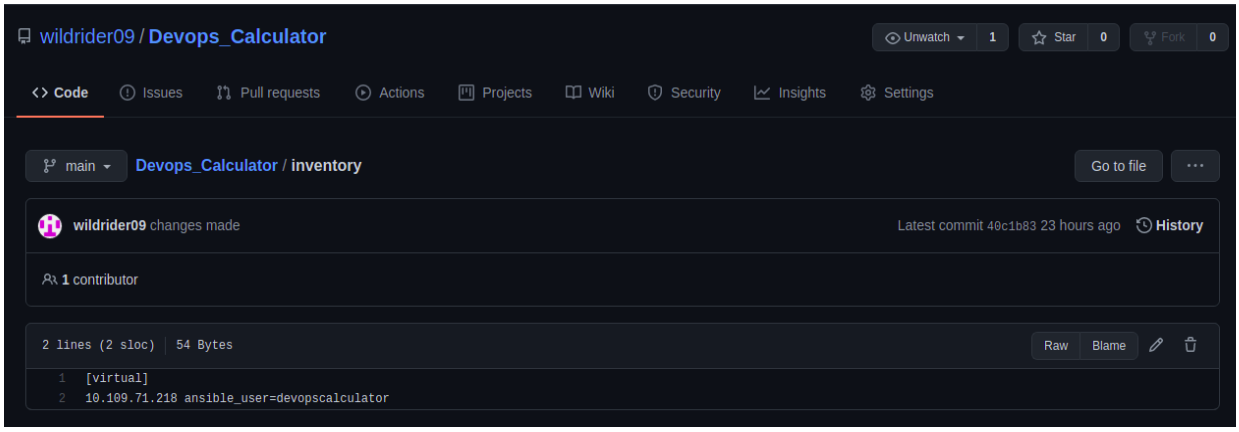

Repositories
wildrider / devops_calculator
Using 0 of 1 private repositories. Get more
General
Tags
Builds
Collaborators
Webhooks
Settings
wildrider / devops_calculator
This repository does not have a description
Last pushed: 3 days ago
Docker commands
Public View
To push a new tag to this repository,
docker push wildrider/devops_calculator:tagname
Tags and Scans
VULNERABILITY SCANNING - DISABLED
Enable
This repository contains 1 tag(s).
TAG
OS
PULLED
PUSHED
latest
a few seconds ago
3 days ago
See all
Recent builds
Link a source provider and run a build to see build results here.

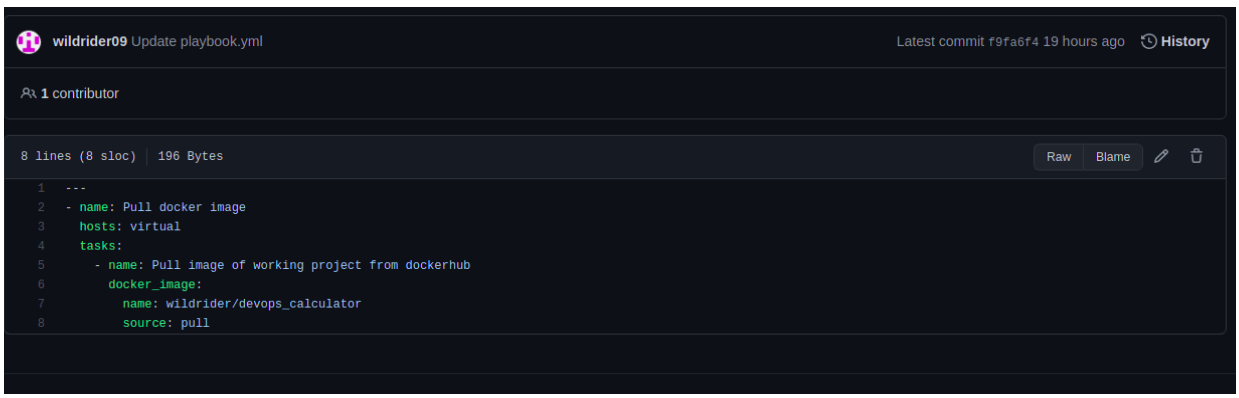
Configuration Management- Ansible:

Ansible is a radically simple IT automation engine that automates cloud provisioning, configuration management, application deployment, intra-service orchestration, and many other IT needs.Designed for multi-tier deployments since day one, Ansible models your IT infrastructure by describing how all of your systems interrelate, rather than just managing one system at a time.

It uses no agents and no additional custom security infrastructure, so it's easy to deploy - and most importantly, it uses a very simple language (YAML, in the form of Ansible Playbooks) that allow you to describe your automation jobs in a way that approaches plain English.

We are using Ansible to fetch the docker image pushed onto docker hub and then running the pulled image onto our container/virtual machine.Ansible is a config management tool & helps in orchestration.





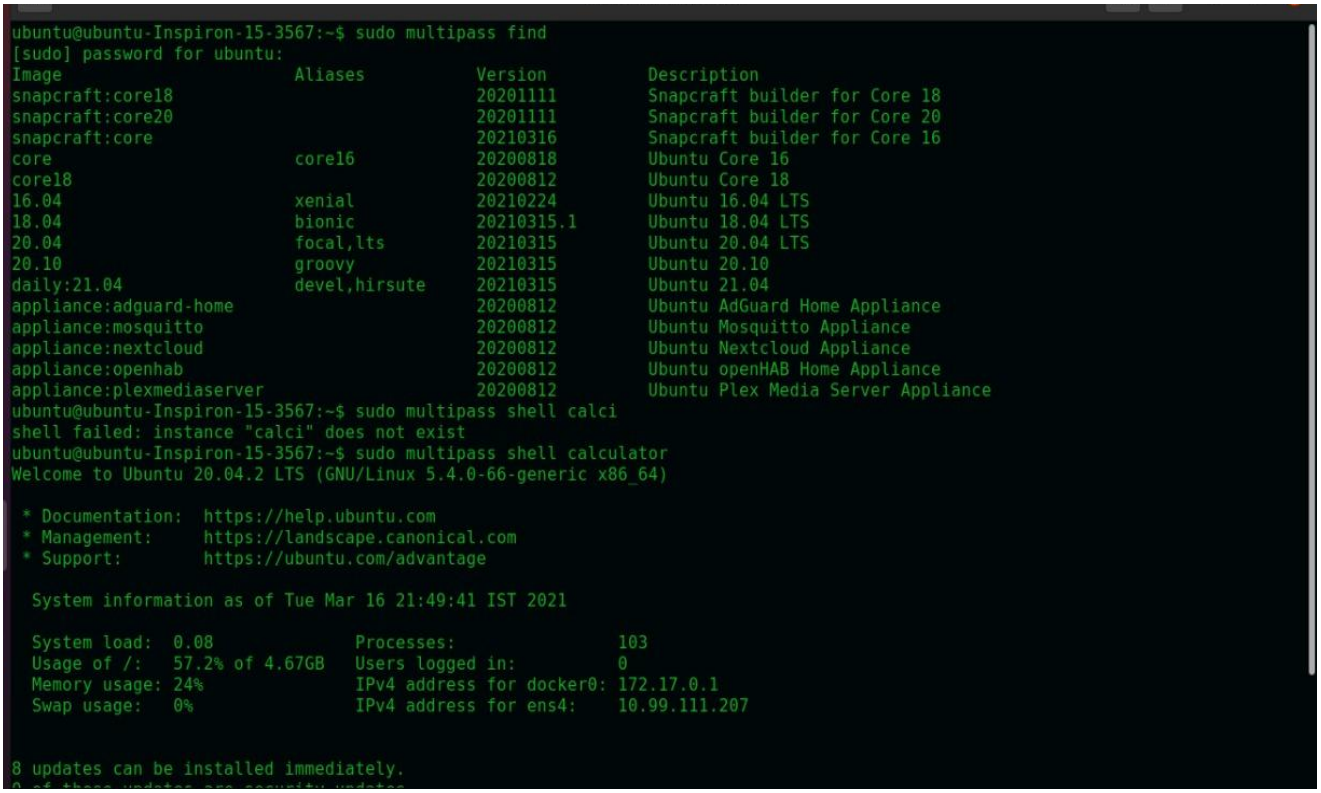
Multipass- A lightweight VM:

Multipass is a mini-cloud on your workstation using native hypervisors of all the supported platforms (Windows, macOS and Linux), it will give you an Ubuntu command line in just a click ("Open shell") or a simple multipass shell command, or even a keyboard shortcut. Find what images are available with multipass find and create new instances with multipass launch.

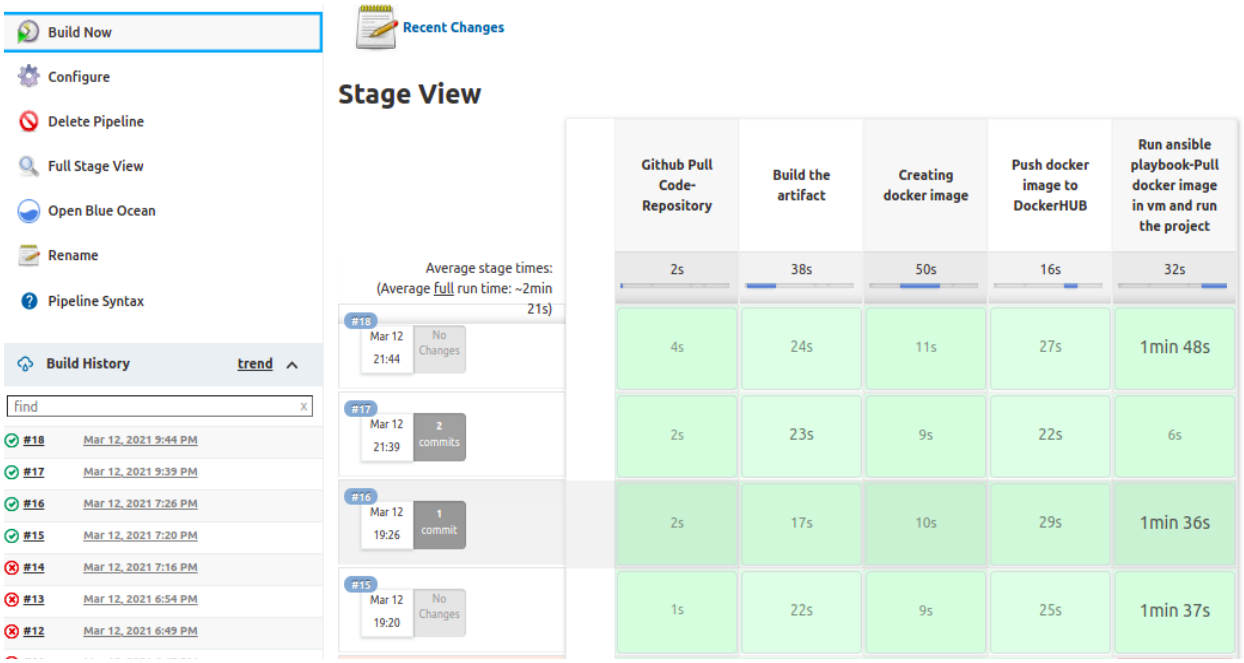
sudo snap install multipass --classic --stable

The above command can be used to install multipass.**multipass find** command enables the user to find the required OS from the list.

From the list, we can decide which OS we need to launch then by using the command :
multipass launch daily:20.04 -This launches a ubuntu 20.04 version as a lightweight VM. Once you're connected, you can start installing the tools you need or go right to developing and testing.



Working Of Pipeline:



Jenkins pipeline helps to make a proper flow for the devops and helps us in continuous integration and deployment on cloud platforms. After we are done setting up every aspect of our DevOps tool chain, we may now build the jenkins job. This job for now can be manually triggered but for other scm it can be triggered based on events such as a new push or new pull but for GitHub you might have set up a webhook. This can be done via port forwarding; you have to make your jenkins port 8080 open on the wide web so it can be triggered based on the event. For now, we'll skip it and we'll build it manually, every stage of the pipeline script works in sequence which includes :

1. Cloning Git
2. Build Executable Jar
3. Building image (docker)
4. Deploy Image on docker hub
5. Execute ansible script from git & push the docker image on the local machine
6. Running the jar file on a local machine manually(multipass shell)

```
#18
e551c18d1724: Layer already exists
fb0b25718816: Layer already exists
da654bc8bc80: Layer already exists
4ef81dc52d99: Layer already exists
7f03bfe4d6dc: Layer already exists
909e93c71745: Layer already exists
c8956c617e1f: Pushed
latest: digest: sha256:ffd87958db813cd5f4a3bde54fa39f086f9c1a9b713e186c0d21421d80c6f7be size: 2002
[Pipeline] }
[Pipeline] // withDockerRegistry
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // script
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Run ansible playbook-Pull docker image in vm and run the project)
[Pipeline] ansiblePlaybook
[Devops_Calculator] $ /usr/bin/ansible-playbook playbook.yml -i inventory

PLAY [Pull docker image] *******************************************************

TASK [Gathering Facts] *********************************************************
ok: [10.109.71.218]

TASK [Pull image of working project from dockerhub] ****************************
changed: [10.109.71.218]

PLAY RECAP *********************************************************************
10.109.71.218              : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0

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

Continuous Monitoring- ELK:

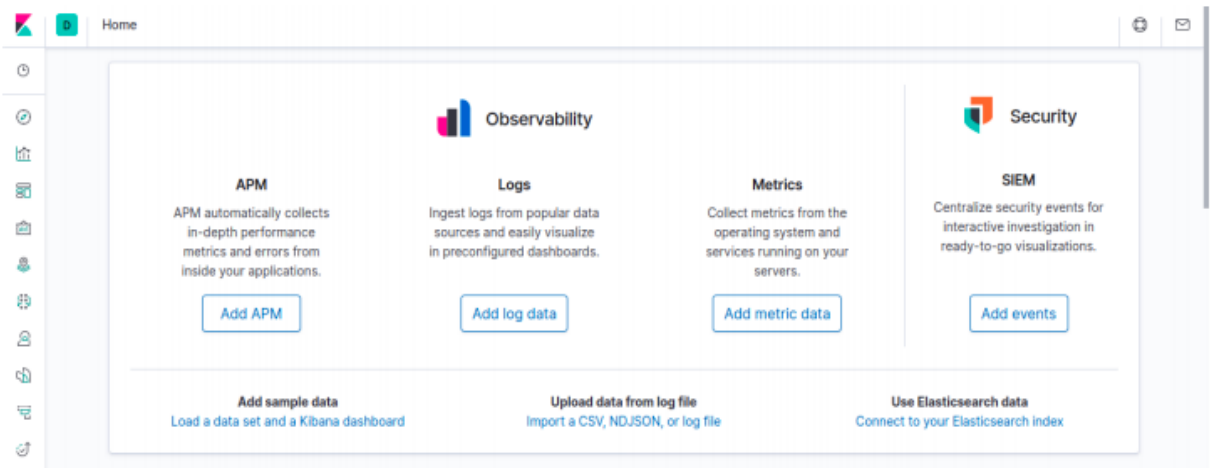
The ELK Stack is a log management platform- collection of three open-source products — Elasticsearch, Logstash, and Kibana — from Elastic. Elasticsearch is a NoSQL database that is based on the Lucene search engine. Logstash is a log pipeline tool that accepts inputs from various sources, executes different transformations, and exports the data to various targets. Kibana is a visualization layer that works on top of Elasticsearch.Together, these three different open source products are most commonly used in log analysis in IT environments (though there are many more use cases for the ELK Stack starting including business intelligence, security and compliance, and web analytics). Logstash collects and parses logs, and then Elasticsearch indexes and stores the information. Kibana then presents the data in visualizations that provide actionable insights into one's environment.

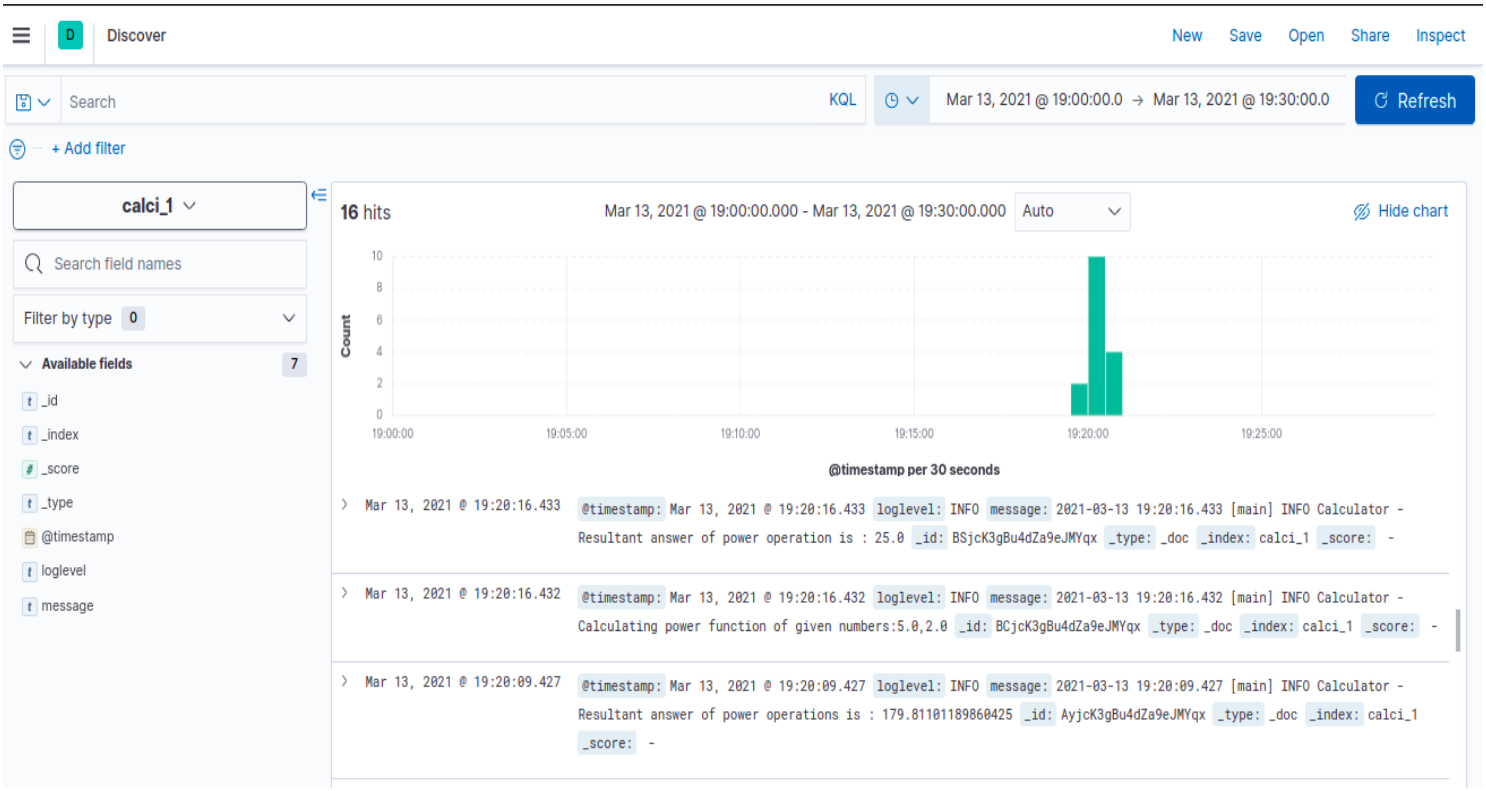
To start with download elastic search, logstash and kibana from https://www.elastic.co Run them side by side and feed the logstash your log set after configuring it configuration file. In the config file we provide the details for logs. Here the config file logstash.conf Elastic search starts at localhost:9200, and kibana starts at localhost:5601 Commands to run Elastic search hare as follows:

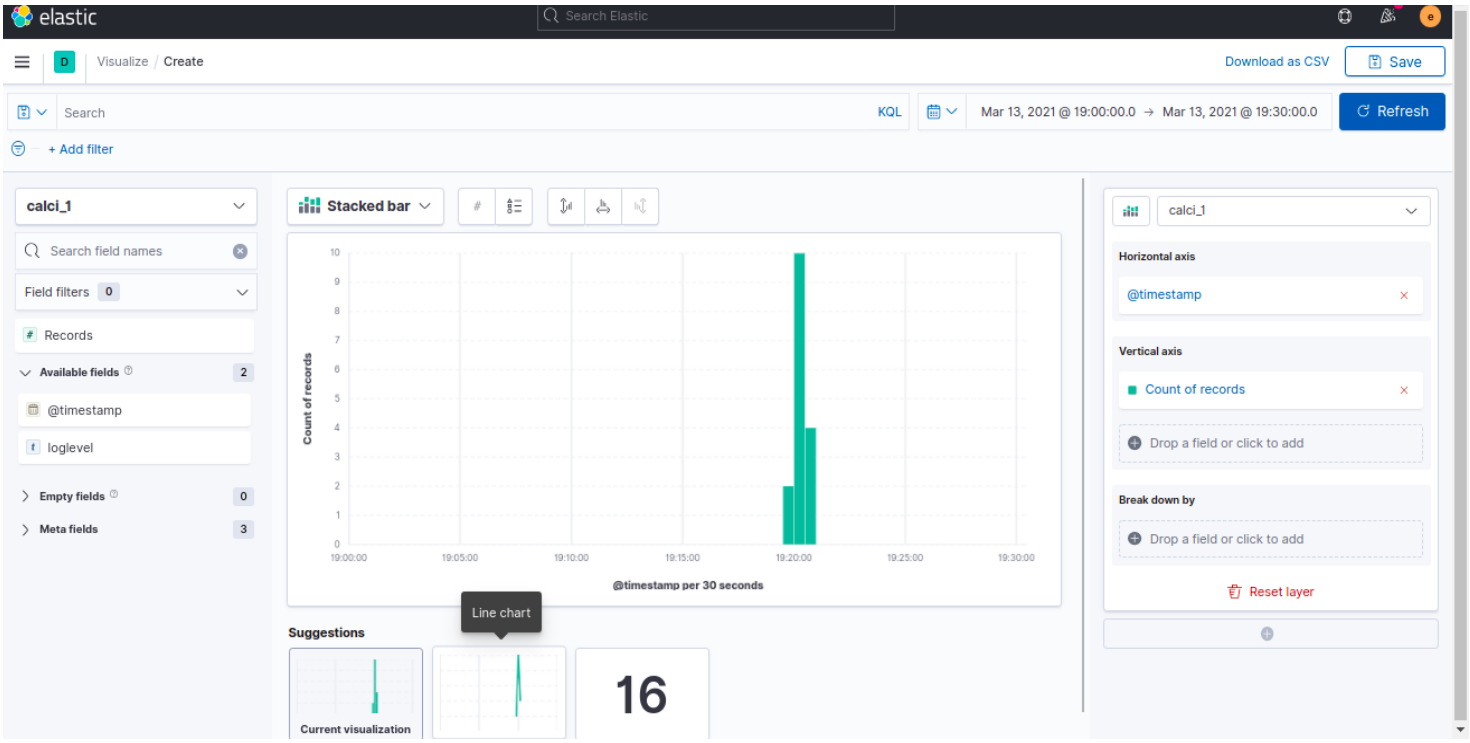
Elastic search - ./path_to_elastic_search/bin/elastic

Logstash - ./path_to_logstash/bin/logstash -f ./path_to_logstash.conf

Kibana - ./path_to_kibana/bin/kiba




Discover
New
Save
Open
Share
Inspect
Search
KQL
Mar 13, 2021 @ 19:00:00.0 → Mar 13, 2021 @ 19:30:00.0
Refresh
+ Add filter
calci_1
Search field names
Filter by type 0
Available fields 7
_id
_index
_score
_type
@timestamp
loglevel
message
16 hits
Mar 13, 2021 @ 19:00:00.000 - Mar 13, 2021 @ 19:30:00.000
Auto
Hide chart
Count
@timestamp per 30 seconds
Mar 13, 2021 @ 19:20:16.433 @timestamp: Mar 13, 2021 @ 19:20:16.433 loglevel: INFO message: 2021-03-13 19:20:16.433 [main] INFO Calculator - Resultant answer of power operation is : 25.0 _id: BSjcK3gBu4dZa9eJMYqx _type: _doc _index: calci_1 _score: -
Mar 13, 2021 @ 19:20:16.432 @timestamp: Mar 13, 2021 @ 19:20:16.432 loglevel: INFO message: 2021-03-13 19:20:16.432 [main] INFO Calculator - Calculating power function of given numbers:5.0,2.0 _id: BCjcK3gBu4dZa9eJMYqx _type: _doc _index: calci_1 _score: -
Mar 13, 2021 @ 19:20:09.427 @timestamp: Mar 13, 2021 @ 19:20:09.427 loglevel: INFO message: 2021-03-13 19:20:09.427 [main] INFO Calculator - Resultant answer of power operations is : 179.81101189860425 _id: AyjcK3gBu4dZa9eJMYqx _type: _doc _index: calci_1 _score: -


elastic
Search Elastic
Visualize / Create
Download as CSV
Save
Search
KQL
Mar 13, 2021 @ 19:00:00.0 → Mar 13, 2021 @ 19:30:00.0
Refresh
+ Add filter
calci_1
Search field names
Field filters 0
Records
Available fields 2
@timestamp
loglevel
Empty fields 0
Meta fields 3
Stacked bar
Count of records
@timestamp per 30 seconds
Line chart
Suggestions
Current visualization
16
calci_1
Horizontal axis
@timestamp
Vertical axis
Count of records
Drop a field or click to add
Break down by
Drop a field or click to add
Reset layer

Problems faced while working on the project:

- Docker within a docker can lead to file system issues - First pull the docker image onto system, but after running it within a docker container I was prompted with docker: Error response from daemon:/var/lib/docker/aufs/mnt/be6e52c353fa0fe27d9c30b6c14e0b596ebf5fa411a93b4661d66c867 756a012-init. Therefore,it's not safe to run docker within docker and could lead to file system errors.I created a multipass shell container for running docker locally, on a local node.

- Docker within a docker official image doesn't support ssh - To resolve this issue, I had an option either run docker image locally which would still create its own container,or change the docker image build of the code over Ubuntu docker image and openjdk and run our created docker image over it.

- Jenkins pipeline - While setting up jenkins there were many challenges I faced which included incorrect docker id and setting credentials for docker hub account, not connecting to git or ansible, and docker not working.

- Maven not working – In many cases maven integration plugin doesn't work, while building the code it would throw the error of mvn not found. While we build it on intellij it has inbuilt maven so building over it doesn't require maven to be installed on your machine. So if maven is not installed then mvn won't work on jenkins because its plugin is a maven integration plugin so maven should be there on your host machine, which can be downloaded via sudo apt maven (debian only).