Mini Project Report

CS 836: Software Production Engineering
Spring 2023
March 21, 2023

Calculator Using DevOps Toolchain

Kautuk Raj
(IMT2019043)

Contents

1 Introduction and Tools Used **4**

2 What is DevOps? **4**

3 Why DevOps? **5**

4 Development **5**

5 Testing **6**

6 Logging **6**

7 Project Build **7**

8 Source Code Management **8**

8.1 Creation of GitHub repository 8

9 ngrok **9**

10 Webhooks **10**

11 Continuous Integration **13**

11.1 Plugins 14

11.2 Global Tool Configuration 14

11.3 Manage Credentials 15

11.4 Jenkins Pipeline 15

11.5 Jenkins Run 17

12 Ansible Deployment **17**

12.1 Dockerfile 17

12.2 Docker Hub 18

12.3 Ansible Installation 18

12.4 Inventory 19

12.5 Playbook 19

12.6 Ansible Run 19

13 Continous Monitoring **20**

1 Introduction and Tools Used

A calculator is a simple application that performs arithmetic operations like addition, subtraction, multiplication, and division. In our implementation, we consider the functions of square root, factorial, natural logarithm and power. It can be developed using a DevOps toolchain that involves the following steps:

1. **Version Control**: Use a version control system like Git and GitHub to keep track of changes in the codebase.
2. **Build**: Use a build tool like Maven to automate the build process. The build tool can compile the source code, create a binary file, and run unit tests.
3. **Test**: Use a testing framework like JUnit to automate the testing process. The testing framework can run unit tests and generate reports on test coverage and test results.
4. **Continuous Integration**: Use a CI/CD server like Jenkins to continuously integrate your code.
5. **Containerize**: Use a containerization tool like Docker to create a container and push it to the Docker Hub.
6. **Webhooks**: To use webhooks, the private IP address of the local machine must be changed to a public IP address using Ngrok.
7. **Deploy**: Use a deployment tool like Ansible to perform the configuration management and deployment.
8. **Logging**: Use a Java-based logging utility like log4j. It is a Java-based logging utility that provides a flexible logging infrastructure for Java applications. It is used to log messages from applications, allowing developers to track the flow of their code and diagnose problems that may occur during runtime.
9. **Monitoring**: Use a monitoring tool like the ELK Stack to monitor the application and generate alerts on errors and failures.

By using a DevOps toolchain, we can automate the entire software development process, from code changes to deployment and monitoring. This can help us deliver high-quality software faster and with fewer errors.

2 What is DevOps?

DevOps is a software development approach that aims to break down the barriers between development (Dev) and operations (Ops) teams by promoting collaboration, communication, and automation. The goal of DevOps is to enable organizations to deliver software products and services faster, more reliably, and with higher quality by ensuring that development and operations teams work together seamlessly.

DevOps involves a combination of practices, tools, and cultural norms, including:

- **Continuous Integration and Continuous Delivery/Deployment (CI/CD)**: This involves automating the building, testing, and deployment of software to ensure that changes can be delivered quickly and reliably.
- **Infrastructure as Code (IaC)**: This involves treating infrastructure as software code, which enables organizations to manage their infrastructure using the same tools and processes they use for application code.

- **Collaboration and communication**: This involves breaking down silos between teams and promoting communication and collaboration between development, operations, and other stakeholders.
- **Agile and Lean methodologies**: These approaches focus on iterative, incremental development and continuous improvement, which aligns well with the DevOps philosophy.

By adopting DevOps, organizations can improve their ability to innovate and respond quickly to changing market conditions, while reducing the risk of errors and outages.

3 Why DevOps?

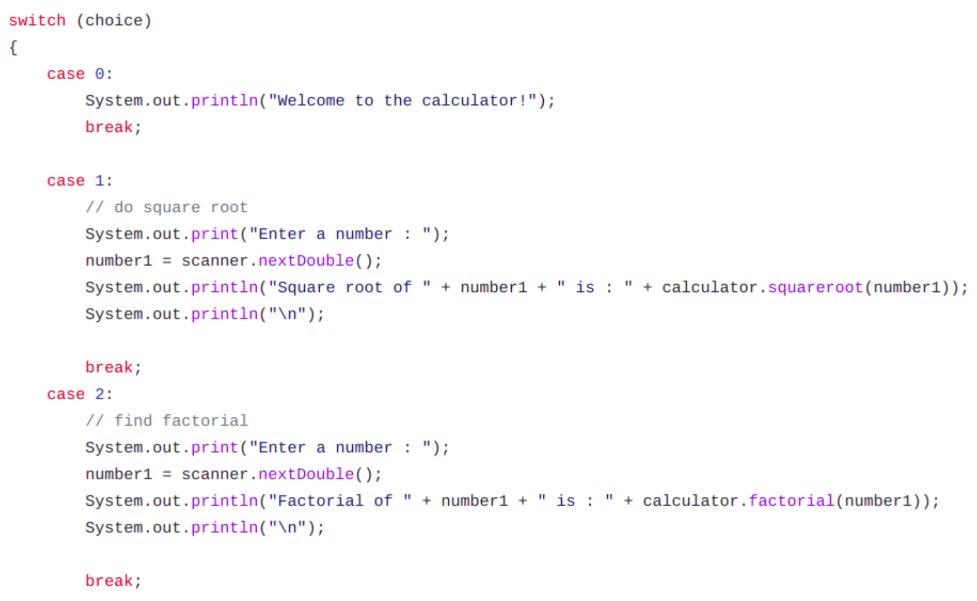
There are several reasons why organizations are adopting DevOps as a software development approach. Here are some of the main benefits:

- **Faster time-to-market**: DevOps enables organizations to release software products and services more frequently and with shorter lead times. This can help companies stay ahead of the competition and respond quickly to changing market conditions.
- **Increased collaboration and communication**: DevOps encourages collaboration and communication between teams, which can lead to better alignment, improved efficiency, and faster problem resolution.
- **Improved quality and reliability**: By automating testing and deployment, DevOps can reduce the risk of errors and outages, improving the overall quality and reliability of software products and services.
- **Better customer experience**: By delivering software products and services more frequently and with higher quality, DevOps can improve the customer experience and drive customer satisfaction.
- **Greater agility and innovation**: DevOps enables organizations to respond quickly to changing customer needs and market conditions, fostering innovation and agility.
- **Cost savings**: By automating processes and reducing manual effort, DevOps can lead to cost savings and increased efficiency.

Overall, DevOps can help organizations improve their software development and delivery processes, enabling them to innovate faster, reduce risk, and deliver value to their customers more quickly and reliably.

4 Development

The codebase is developed in Java 8. A **switch** case is used to implement the different functionalities of the calculator, with different functions for each feature, based on the choice input.

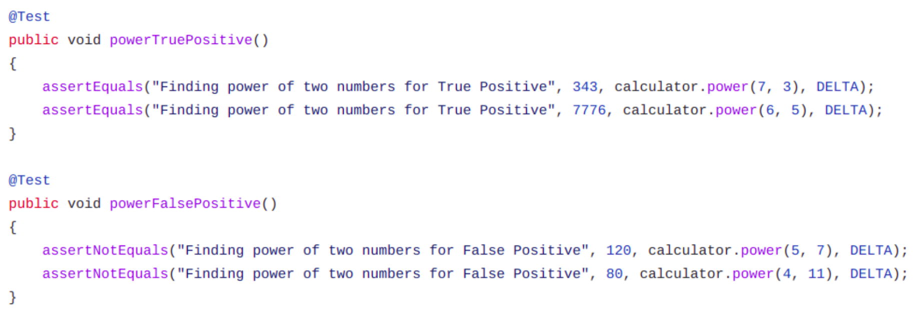


```
switch (choice)
{
    case 0:
        System.out.println("Welcome to the calculator!");
        break;

    case 1:
        // do square root
        System.out.print("Enter a number : ");
        number1 = scanner.nextDouble();
        System.out.println("Square root of " + number1 + " is : " + calculator.squareroot(number1));
        System.out.println("\n");

        break;
    case 2:
        // find factorial
        System.out.print("Enter a number : ");
        number1 = scanner.nextDouble();
        System.out.println("Factorial of " + number1 + " is : " + calculator.factorial(number1));
        System.out.println("\n");

        break;
```

Figure 1: A snippet of the **switch** case code

5 Testing

Testcases for the purposes of unit testing are written in JUnit, to test each functionality of the calculator. JUnit is a testing framework for Java that is used to write and run unit tests for Java code. Unit testing is a software testing technique in which individual units or components of a software application are tested to ensure that they are working correctly. The test file contains test cases that are both true and false positive and are used to check the code while the project is built.



```
@Test
public void powerTruePositive()
{
    assertEquals("Finding power of two numbers for True Positive", 343, calculator.power(7, 3), DELTA);
    assertEquals("Finding power of two numbers for True Positive", 7776, calculator.power(6, 5), DELTA);
}

@Test
public void powerFalsePositive()
{
    assertNotEquals("Finding power of two numbers for False Positive", 120, calculator.power(5, 7), DELTA);
    assertNotEquals("Finding power of two numbers for False Positive", 80, calculator.power(4, 11), DELTA);
}
```

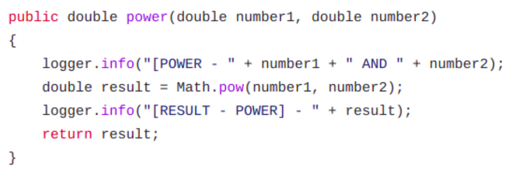
Figure 2: A snippet of the test cases code

The tests can be run using the command `mvn test -Dtest="TestClassName"`.

6 Logging

Log4j (short for Log for Java) is a popular Java-based logging utility that provides a flexible and configurable logging framework for Java applications. It allows developers to generate log statements from their code to capture runtime information about the application's behavior, performance, and errors.

Log4j allows developers to configure log statements to be written to various destinations such as console, files, email, or other custom destinations. Log4j also provides support for log levels, allowing developers to control the granularity of the information captured in the logs. Log levels include `DEBUG`, `INFO`, `WARN`, `ERROR`, and `FATAL`.



```
public double power(double number1, double number2)
{
    logger.info("[POWER - " + number1 + " AND " + number2);
    double result = Math.pow(number1, number2);
    logger.info("[RESULT - POWER] - " + result);
    return result;
}
```

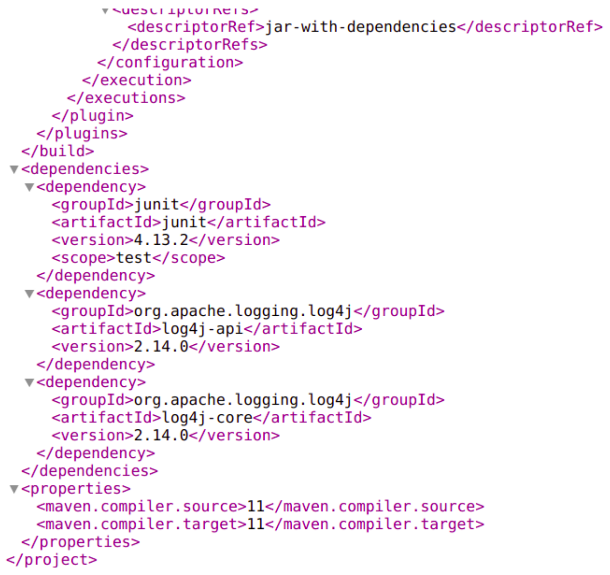
Figure 3: A snippet of the logging tasks

7 Project Build

Maven is a build automation tool used primarily for Java projects. It helps manage the build process by defining project structure, dependencies, and plugins. Some of the main uses of Maven:

- **Managing dependencies**: Maven helps manage dependencies by downloading required libraries and frameworks from remote repositories.
- **Building the project**: Maven builds the project by compiling source code, packaging it into an executable format (such as a JAR file), and running tests.
- **Managing project structure**: Maven defines a standard project structure that helps developers organize their code into logical components and provides a consistent build process.
- **Automating tasks**: Maven can automate tasks such as generating documentation, running tests, and deploying artifacts to a repository.

Overall, Maven simplifies the build process for Java projects, improves code quality, and makes it easier for teams to collaborate. To use certain dependencies like JUnit in our project, we define the `pom.xml` file, a part of the Maven build.



```
             ▼<descriptorRefs>
                <descriptorRef>jar-with-dependencies</descriptorRef>
              </descriptorRefs>
            </configuration>
          </execution>
        </executions>
      </plugin>
    </plugins>
  </build>
 ▼<dependencies>
   ▼<dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.13.2</version>
      <scope>test</scope>
    </dependency>
   ▼<dependency>
      <groupId>org.apache.logging.log4j</groupId>
      <artifactId>log4j-api</artifactId>
      <version>2.14.0</version>
    </dependency>
   ▼<dependency>
      <groupId>org.apache.logging.log4j</groupId>
      <artifactId>log4j-core</artifactId>
      <version>2.14.0</version>
    </dependency>
  </dependencies>
 ▼<properties>
    <maven.compiler.source>11</maven.compiler.source>
    <maven.compiler.target>11</maven.compiler.target>
  </properties>
</project>
```

Figure 4: A snippet of the pom.xml file

By executing the command `mvn clean install`, the codebase will be built, dependencies will be fetched, test cases will be run and reported. In the current directory, a new directory called `target` will be generated which will have the executable `.jar` files.

8 Source Code Management Using Git and GitHub

Source code management (SCM), also known as version control, is the practice of tracking and managing changes to software code over time. SCM allows developers to collaborate on a project, maintain a history of changes, and control access to different versions of the code.

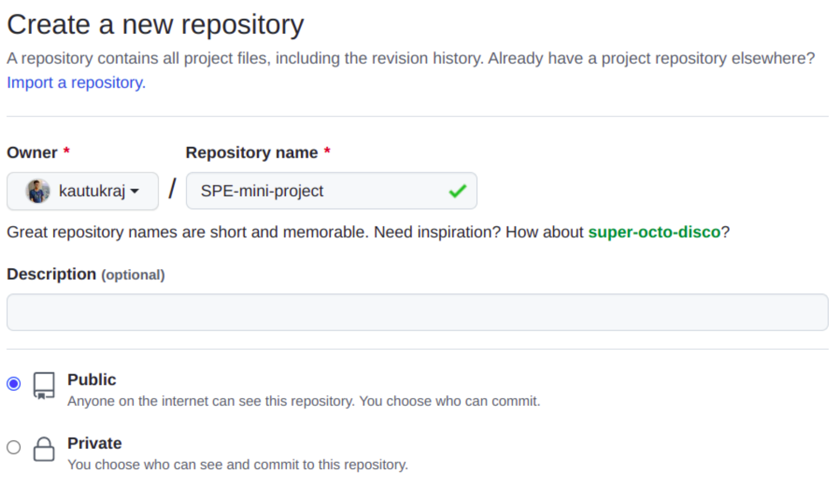
Git is a popular distributed version control system (DVCS) used for tracking changes in software code. It was created by Linus Torvalds in 2005 and is widely used by software developers and organizations around the world.

GitHub is a web-based platform that provides hosting for software development projects using the Git version control system. It is a popular platform for collaborating on open source software projects and for managing private projects within organizations.

The development work can be started locally and the directory can be added to git using `git init`, followed by `git add .`.

8.1 Creation of GitHub repository

The repository can be created on GitHub, as shown below.



Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? Import a repository.

Owner *: kautukraj / Repository name *: SPE-mini-project

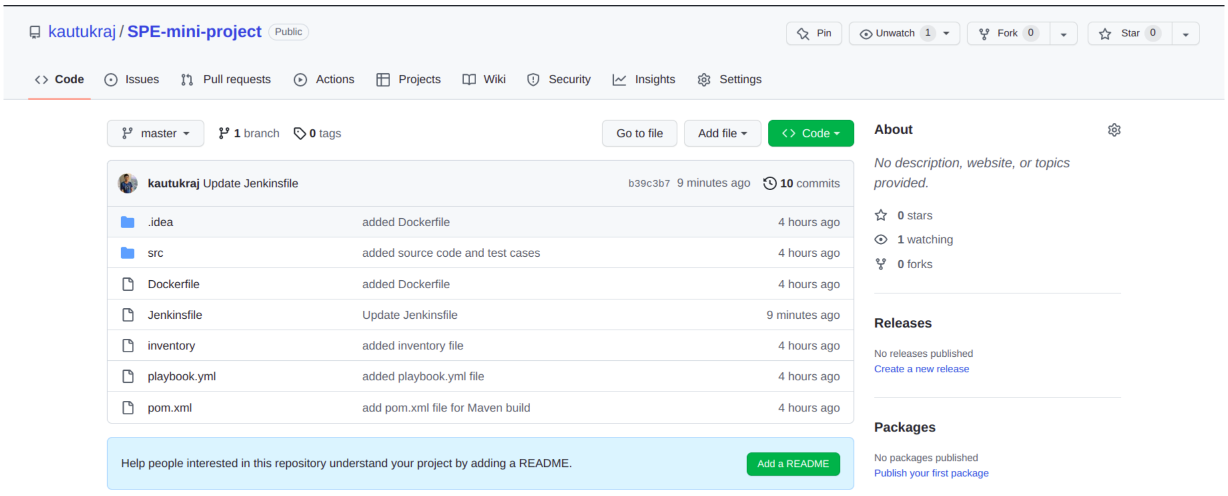
Great repository names are short and memorable. Need inspiration? How about **super-octo-disco**?

Description (optional)

[✓ Public] Anyone on the internet can see this repository. You choose who can commit.

Private — You choose who can see and commit to this repository.

Figure 5: Creation of a repository on GitHub



kautukraj Update Jenkinsfile b39c3b7 9 minutes ago 10 commits

.idea	added Dockerfile	4 hours ago
src	added source code and test cases	4 hours ago
Dockerfile	added Dockerfile	4 hours ago
Jenkinsfile	Update Jenkinsfile	9 minutes ago
inventory	added inventory file	4 hours ago
playbook.yml	added playbook.yml file	4 hours ago
pom.xml	add pom.xml file for Maven build	4 hours ago

Help people interested in this repository understand your project by adding a README. Add a README

About

No description, website, or topics provided.

0 stars
1 watching
0 forks

Releases

No releases published
Create a new release

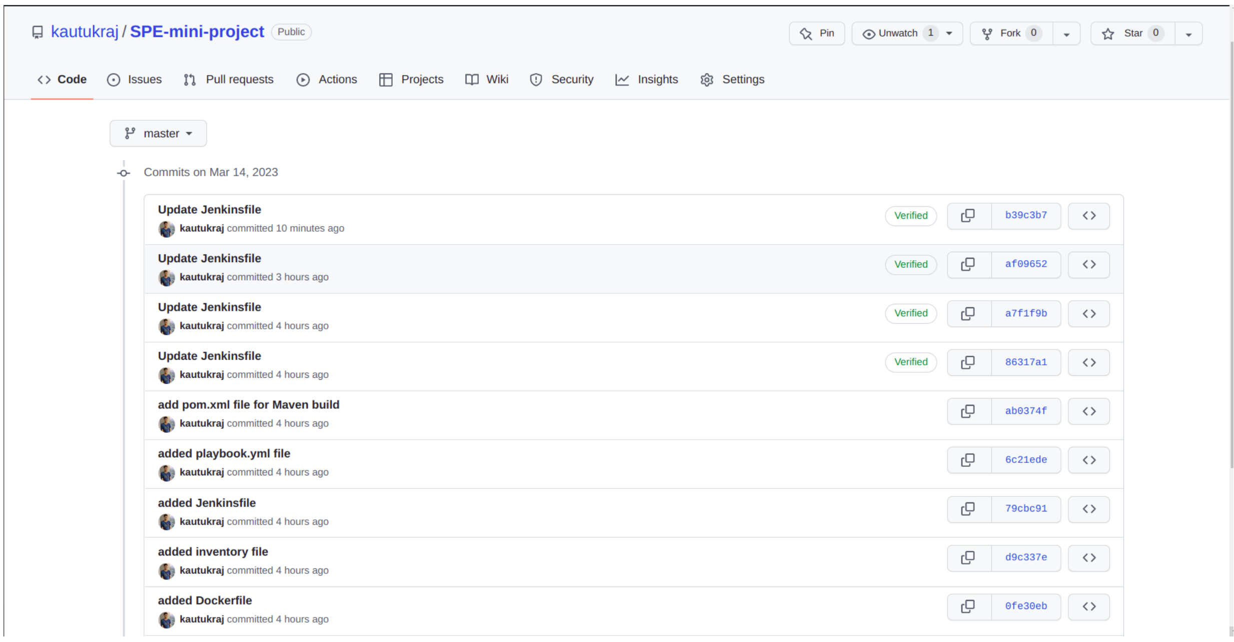
Packages

No packages published
Publish your first package

Figure 6: View of the repository on GitHub

The repository can be found at this URL: `https://github.com/kautukraj/SPE-mini-project`.

The GitHub URL can be added to our local git system using `git remote add origin <repo URL>`. A commit can be made using `git commit -m "a meaningful commit message"` and local filesystem changes can be sent to GitHub using `git push origin main`.



Commits on Mar 14, 2023

Update Jenkinsfile	kautukraj committed 10 minutes ago	Verified	b39c3b7
Update Jenkinsfile	kautukraj committed 3 hours ago	Verified	af09652
Update Jenkinsfile	kautukraj committed 4 hours ago	Verified	a7f1f9b
Update Jenkinsfile	kautukraj committed 4 hours ago	Verified	86317a1
add pom.xml file for Maven build	kautukraj committed 4 hours ago		ab0374f
added playbook.yml file	kautukraj committed 4 hours ago		6c21ede
added Jenkinsfile	kautukraj committed 4 hours ago		79cbc91
added inventory file	kautukraj committed 4 hours ago		d9c337e
added Dockerfile	kautukraj committed 4 hours ago		0fe30eb

Figure 7: A view of the commits on GitHub

9 ngrok

ngrok is a tool that allows developers to expose a web server running on their local machine to the internet. It creates a secure tunnel between the local machine and a public URL that can be accessed from anywhere, making it possible to test and share applications without the need for a public-facing server.

When ngrok is running, it provides a temporary public URL that can be used to access the local server. This URL can be shared with others, allowing them to view and interact with the application being developed. The ngrok tunnel is encrypted using HTTPS, ensuring that traffic between the local machine and the public URL is secure. Developers can use ngrok to receive webhooks from external services during development and testing.

Run the following command to start Ngrok: `./ngrok http <port-number>`. Replace `<port-number>` with the port number of the local web server we want to expose. Ngrok will generate a public URL that we can use to access our local web server from anywhere. We can find the URL in the Ngrok terminal window under the "Forwarding" section. Copy the URL and paste it into our web browser to test it. The local web server is now accessible from the internet using the Ngrok public URL. Keep in mind that the Ngrok session will end if we close the terminal window or stop the Ngrok process.

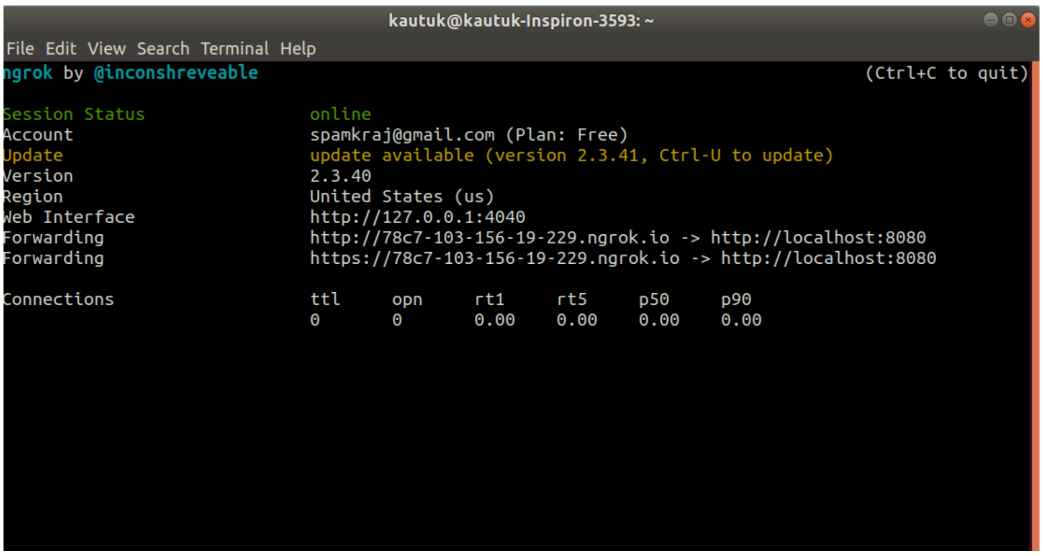
10 Webhooks

GitHub webhooks can be used in Jenkins to trigger builds or perform other actions in response to events in a GitHub repository, such as a new pull request being opened or a code push to a specific branch.

To set up GitHub webhooks in Jenkins, we need to follow these steps:

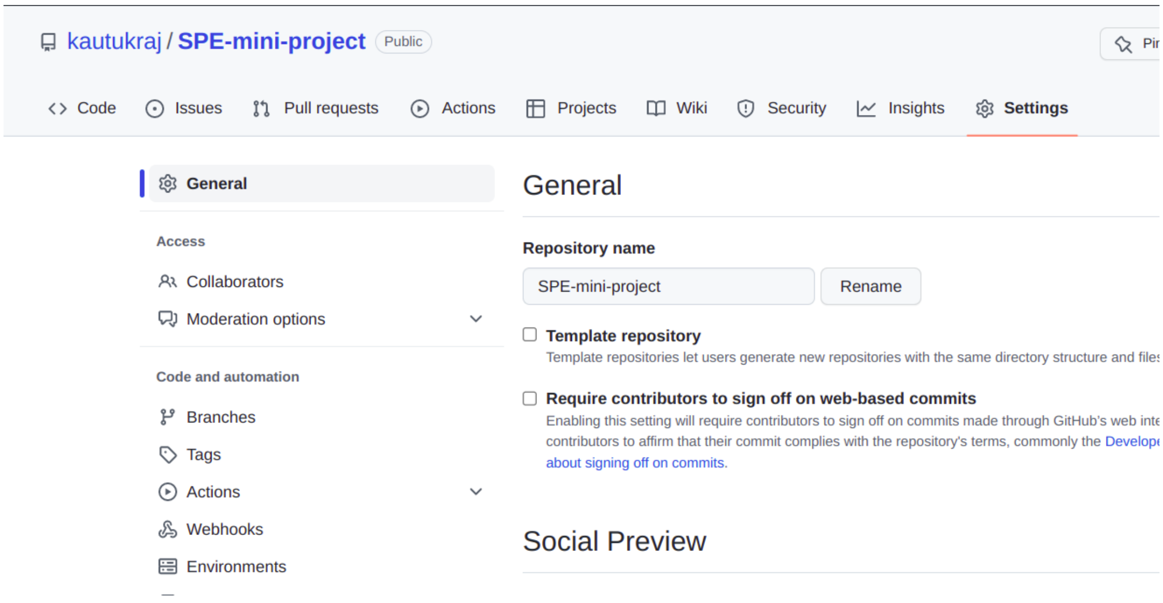
- Install the GitHub plugin in Jenkins, which allows Jenkins to interact with GitHub repositories and respond to webhook events.
- Create a new Jenkins job or configure an existing one to respond to a webhook event. Select the "GitHub hook trigger for GITScm polling" option in the Build Triggers section of the job configuration.
- In the GitHub repository settings, add a new webhook and specify the URL of the Jenkins server and the payload URL for the specific webhook event. You can choose to trigger the webhook event for all pushes, pull requests, or specific branches or tags.
- Save the webhook configuration and test the integration by triggering a webhook event in the GitHub repository. Jenkins should receive the event and trigger a build or perform the specified action.

GitHub webhooks in Jenkins can be used to automate various aspects of the development workflow, such as building and testing code, deploying applications, and notifying team members of new events. By using webhooks, we can streamline the development process and reduce manual effort.


kautuk@kautuk-Inspiron-3593: ~
File Edit View Search Terminal Help

```
ngrok by @inconshreveable                                                    (Ctrl+C to quit)

Session Status                online
Account                       spamkraj@gmail.com (Plan: Free)
Update                        update available (version 2.3.41, Ctrl-U to update)
Version                       2.3.40
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    http://78c7-103-156-19-229.ngrok.io -> http://localhost:8080
Forwarding                    https://78c7-103-156-19-229.ngrok.io -> http://localhost:8080

Connections                   ttl     opn     rt1     rt5     p50     p90
                              0       0       0.00    0.00    0.00    0.00
```

Figure 8: Running ngrok to obtain the forwarding URL


kautukraj / SPE-mini-project Public

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

General

Access
Collaborators
Moderation options

Code and automation
Branches
Tags
Actions
Webhooks
Environments

General

Repository name

SPE-mini-project Rename

Template repository
Template repositories let users generate new repositories with the same directory structure and files

Require contributors to sign off on web-based commits
Enabling this setting will require contributors to sign off on commits made through GitHub's web inte
contributors to affirm that their commit complies with the repository's terms, commonly the Develope
about signing off on commits.

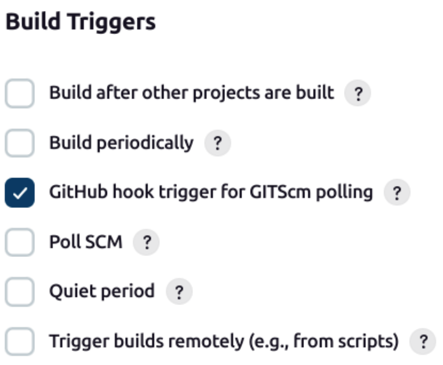
Social Preview

Figure 9: Choosing the webhooks options on the GitHub repository

In Jenkins, there are several types of build triggers available to automatically start a build process:

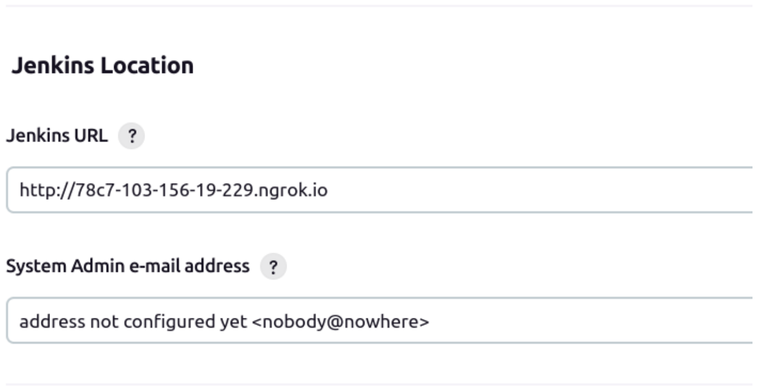
- **Polling SCM**: This trigger regularly checks the source code repository for changes and starts a build if new changes are detected.
- **Schedule**: This trigger starts a build at a specified time or on a recurring schedule.
- **Build after other projects are built**: This trigger starts a build after one or more specified projects are built successfully.
- **Trigger builds remotely**: This trigger allows remote programs or scripts to start a build through a URL or API call.
- **Build periodically**: This trigger starts a build at regular intervals, such as every hour or every day.
- **GitHub hook trigger for GITScm polling**: This trigger allows Jenkins to listen for incoming webhooks from GitHub and start a build when changes are pushed to the repository.
- **Pipeline trigger**: This trigger allows one pipeline to trigger another pipeline.

These triggers can be configured in the Jenkins project settings, and multiple triggers can be used in combination to start a build based on different conditions.


Build Triggers

- [ ] Build after other projects are built
- [ ] Build periodically
- [x] GitHub hook trigger for GITScm polling
- [ ] Poll SCM
- [ ] Quiet period
- [ ] Trigger builds remotely (e.g., from scripts)

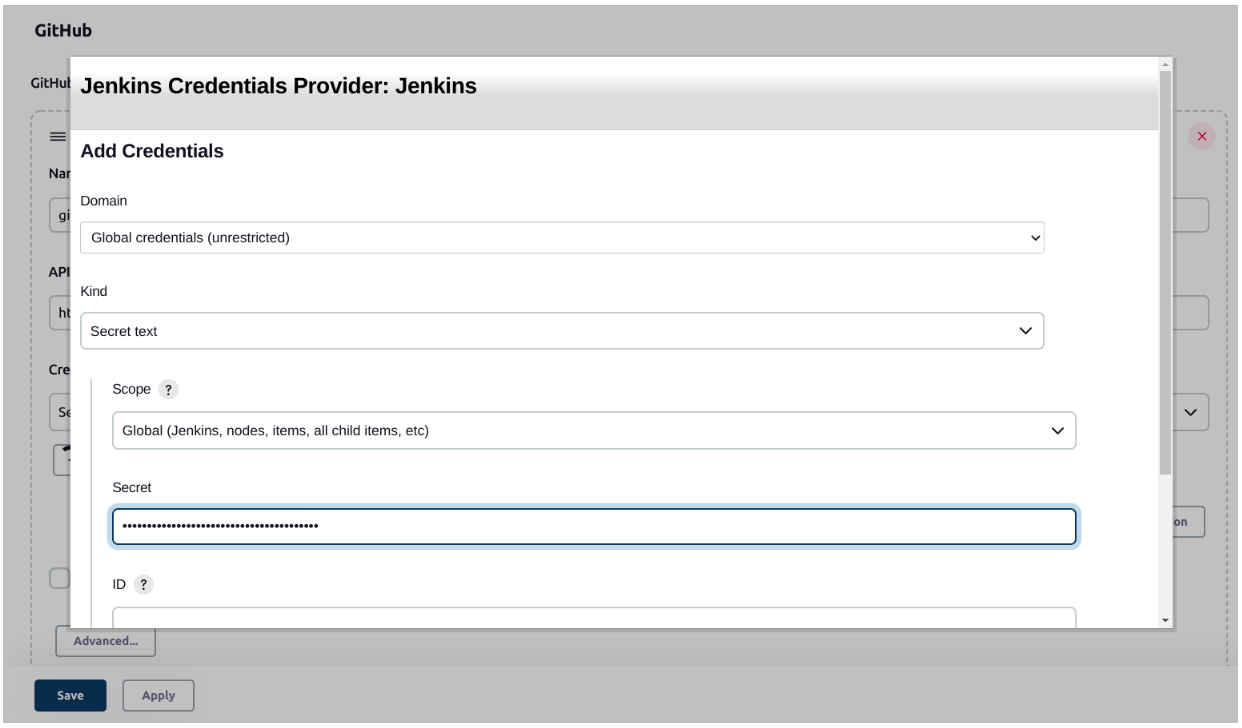
Figure 10: Choosing the GitSCM polling mechanism in Jenkins


Jenkins Location

Jenkins URL: http://78c7-103-156-19-229.ngrok.io

System Admin e-mail address: address not configured yet <nobody@nowhere>

Figure 11: Adding the ngrok forwarding URL to Jenkins


Jenkins Credentials Provider: Jenkins

Add Credentials

Domain: Global credentials (unrestricted)

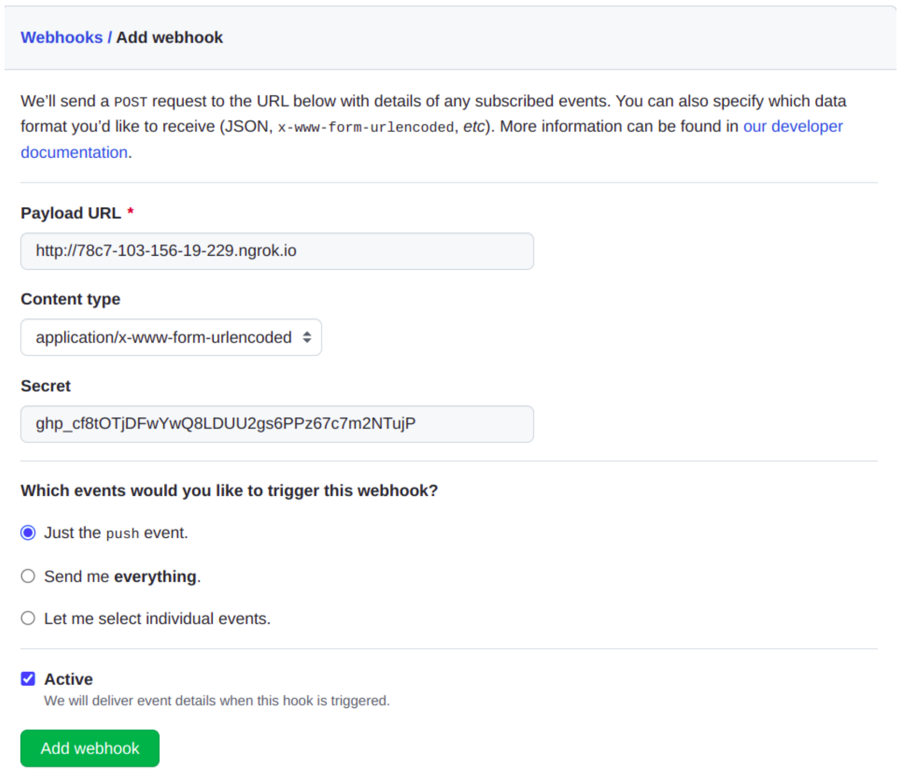
Kind: Secret text

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

Secret: ••

ID

Figure 12: Adding the secret text generated from GitHub to the Jenkins Credentials provider


Webhooks / Add webhook

We'll send a `POST` request to the URL below with details of any subscribed events. You can also specify which data format you'd like to receive (JSON, `x-www-form-urlencoded`, *etc*). More information can be found in our developer documentation.

Payload URL *

http://78c7-103-156-19-229.ngrok.io

Content type

application/x-www-form-urlencoded

Secret

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

Which events would you like to trigger this webhook?

[✓ Just the `push` event.]

Send me **everything**.

Let me select individual events.

[✓] **Active**

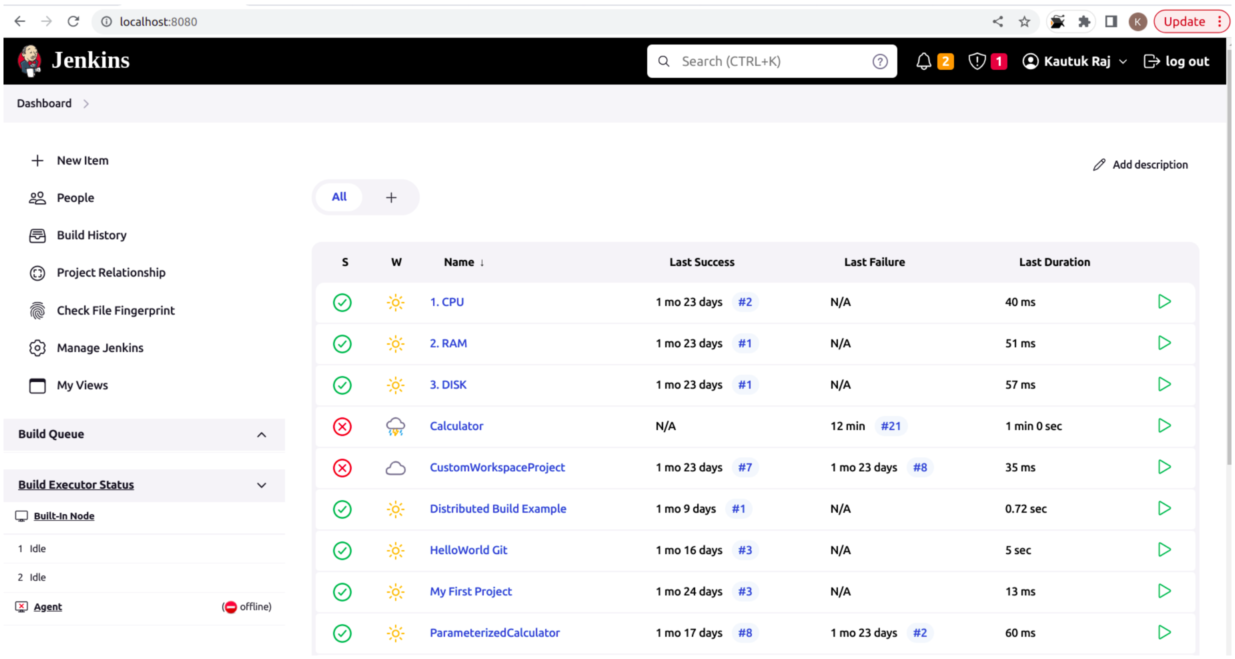
We will deliver event details when this hook is triggered.

Figure 13: Adding the ngrok forwarding URL to the GitHub repository

11 Continuous Integration Using Jenkins

Jenkins is an open-source automation server that allows developers to build, test, and deploy software applications. It was originally developed as a continuous integration (CI) tool for Java applications, but has since grown to support a wide range of programming languages and platforms. Jenkins automates the entire software development process, from building code to testing, and deploying the final application. It provides a user-friendly web interface for managing and configuring jobs, which are the individual tasks that make up the development pipeline. Jenkins can be installed by following these steps:

- `wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -`
- `sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.lis`
- `sudo apt install ca-certificates`
- `sudo apt-get update`
- `sudo apt-get install jenkins`


localhost:8080

Jenkins | Search (CTRL+K) | 2 | 1 | Kautuk Raj | log out

Dashboard >

- New Item
- People
- Build History
- Project Relationship
- Check File Fingerprint
- Manage Jenkins
- My Views

Build Queue

Build Executor Status

Built-In Node
1 Idle
2 Idle
Agent (offline)

Add description

All +

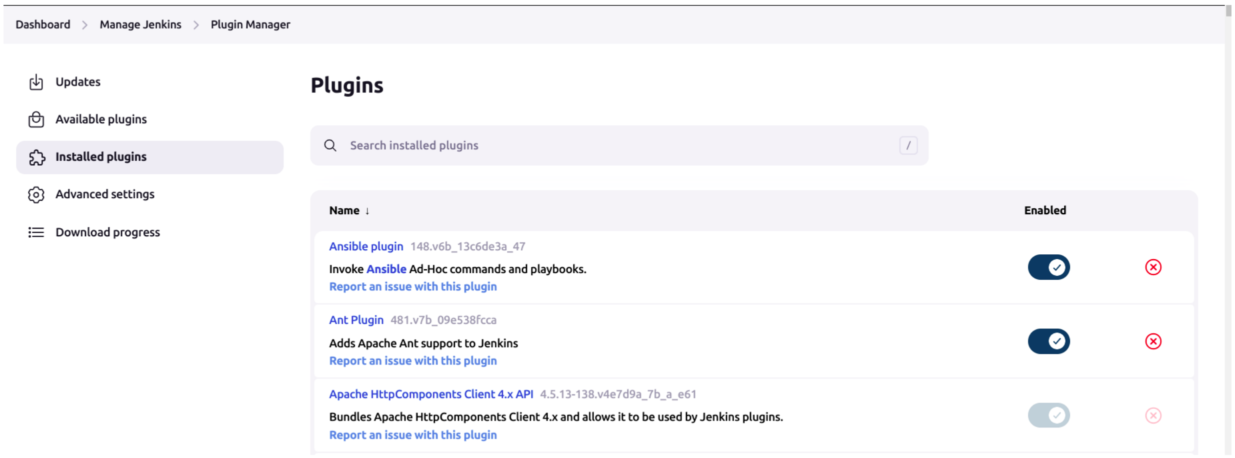
S	W	Name ↓	Last Success	Last Failure	Last Duration	
✓	☀	1. CPU	1 mo 23 days #2	N/A	40 ms	▷
✓	☀	2. RAM	1 mo 23 days #1	N/A	51 ms	▷
✓	☀	3. DISK	1 mo 23 days #1	N/A	57 ms	▷
✗	🌧	Calculator	N/A	12 min #21	1 min 0 sec	▷
✗	☁	CustomWorkspaceProject	1 mo 23 days #7	1 mo 23 days #8	35 ms	▷
✓	☀	Distributed Build Example	1 mo 9 days #1	N/A	0.72 sec	▷
✓	☀	HelloWorld Git	1 mo 16 days #3	N/A	5 sec	▷
✓	☀	My First Project	1 mo 24 days #3	N/A	13 ms	▷
✓	☀	ParameterizedCalculator	1 mo 17 days #8	1 mo 23 days #2	60 ms	▷

Figure 14: The Jenkins dashboard

11.1 Plugins

Plugins are a key feature of Jenkins that allow developers to extend the functionality of the software and customize it to meet their specific needs. Jenkins has a vast library of over 1,500 plugins, with new plugins being developed and added to the library on a regular basis.

The plugins which we need for this project are the ones for Git, Docker, Ansible and Maven. Plugins can be installed from the `Plugin Manager` under `Manage Jenkins`.


Dashboard > Manage Jenkins > Plugin Manager

- Updates
- Available plugins
- Installed plugins
- Advanced settings
- Download progress

Plugins

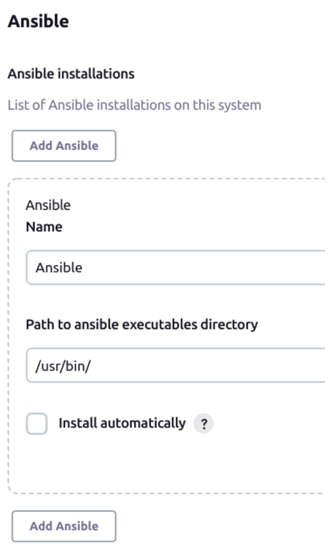
Search installed plugins

Name ↓	Enabled	
Ansible plugin 148.v6b_13c6de3a_47 Invoke Ansible Ad-Hoc commands and playbooks. Report an issue with this plugin	✓	⊗
Ant Plugin 481.v7b_09e538fcca Adds Apache Ant support to Jenkins Report an issue with this plugin	✓	⊗
Apache HttpComponents Client 4.x API 4.5.13-138.v4e7d9a_7b_a_e61 Bundles Apache HttpComponents Client 4.x and allows it to be used by Jenkins plugins. Report an issue with this plugin	✓	⊗

Figure 15: The Plugins page on Jenkins

11.2 Global Tool Configuration

Under the `Global Tool Configuration` options in `Manage Jenkins`, we have to specify the configuration for Maven, Git and Ansible. Though they are almost everytime automatically populated, it is a good idea to have a check.


Ansible

Ansible installations

List of Ansible installations on this system

Ansible
Name

Ansible

Path to ansible executables directory

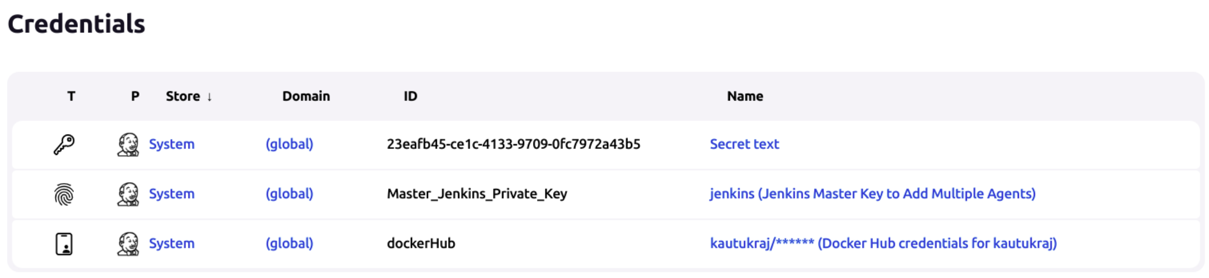
/usr/bin/

[ ] Install automatically

Figure 16: The Global Tool Configuration for Ansible

11.3 Manage Credentials

Under the `Manage Credentials` options in `Manage Jenkins`, we have to specify the login credentials for Docker Hub and GitHub.


Credentials

T	P	Store ↓	Domain	ID	Name
🔑		System	(global)	23eafb45-ce1c-4133-9709-0fc7972a43b5	Secret text
		System	(global)	Master_Jenkins_Private_Key	jenkins (Jenkins Master Key to Add Multiple Agents)
		System	(global)	dockerHub	kautukraj/****** (Docker Hub credentials for kautukraj)

Figure 17: The Manage Credentials page

11.4 Jenkins Pipeline

In Jenkins, a pipeline is a way to define and automate the steps required to build, test, and deploy a software application. A pipeline is typically defined in a Jenkinsfile, which is a text file that describes the stages, steps, and configurations required to execute a continuous delivery pipeline. Jenkins pipelines can be configured and managed through the Jenkins web interface or through the Jenkinsfile itself, which allows for version-controlled and automated pipeline management. The different stages in our pipeline script are:

1. **Git Pull**: Pull the codebase from a remote repository hosted on GitHub.

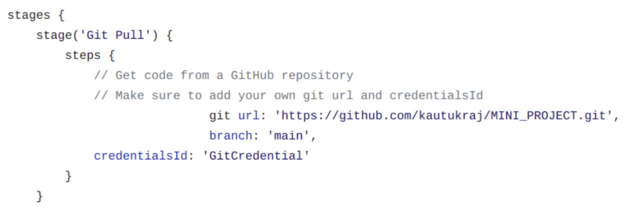
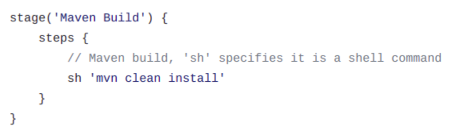

```
stages {
    stage('Git Pull') {
        steps {
            // Get code from a GitHub repository
            // Make sure to add your own git url and credentialsId
                            git url: 'https://github.com/kautukraj/MINI_PROJECT.git',
                            branch: 'main',
            credentialsId: 'GitCredential'
        }
    }
```

Figure 18: Pipeline script for Git Pull

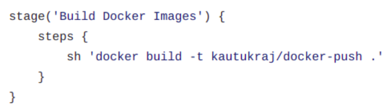
2. **Maven Build**: It creates a jar file with all of our source code's dependencies in it. A new target folder with the new jar file will be created when the original target folder with the previous dependencies is destroyed.



```
stage('Maven Build') {
    steps {
        // Maven build, 'sh' specifies it is a shell command
        sh 'mvn clean install'
    }
}
```

Figure 19: Pipeline script for Maven Build

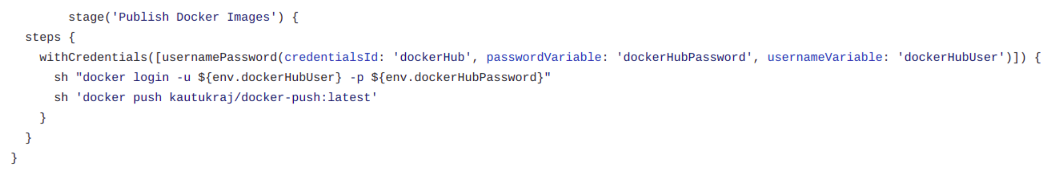
3. **Create Docker Image**: On our local system, it is used to create images that are subsequently uploaded to our Docker Hub, allowing us to fetch the image and execute the application on other systems. The tag for the image is `:latest`.



```
stage('Build Docker Images') {
    steps {
        sh 'docker build -t kautukraj/docker-push .'
    }
}
```

Figure 20: Pipeline script for Create Docker Image

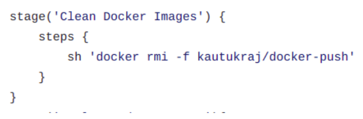
4. **Publish Docker Image**: We are uploading the image to our Docker Hub in this stage so that anyone may download it. To grant permissions, we must execute the command `sudo chmod 666 /var/run/docker.sock` on the localhost.



```
      stage('Publish Docker Images') {
  steps {
    withCredentials([usernamePassword(credentialsId: 'dockerHub', passwordVariable: 'dockerHubPassword', usernameVariable: 'dockerHubUser')]) {
      sh "docker login -u ${env.dockerHubUser} -p ${env.dockerHubPassword}"
      sh 'docker push kautukraj/docker-push:latest'
    }
  }
}
```

Figure 21: Pipeline script for Publish Docker Image

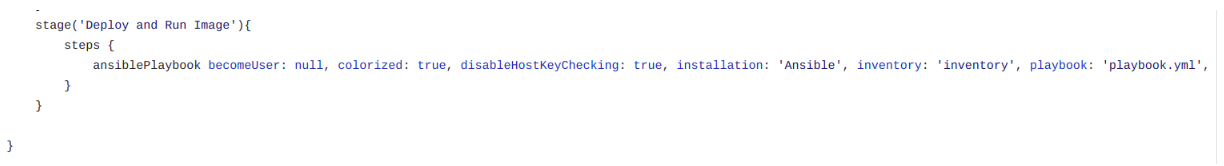
5. **Clean Docker Image**: In this step, we remove the Docker image from our local system.



```
stage('Clean Docker Images') {
    steps {
        sh 'docker rmi -f kautukraj/docker-push'
    }
}
```

Figure 22: Pipeline script for Clean Docker Image

6. **Ansible Deploy**: Fetching the image from the Docker Hub and running it on the hosts specified in the `inventory` file.

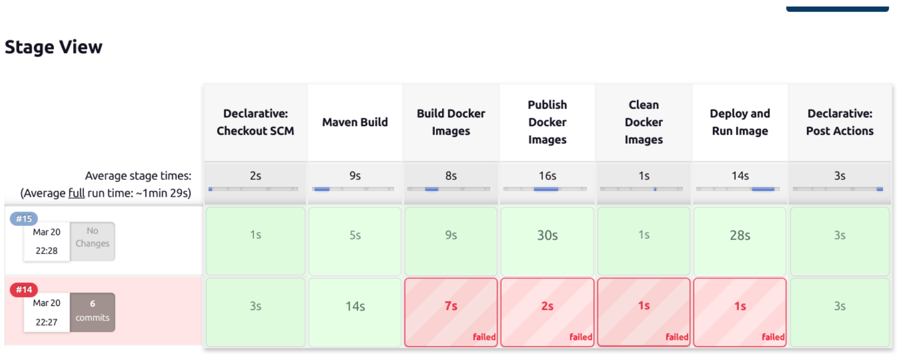


```
stage('Deploy and Run Image'){
    steps {
        ansiblePlaybook becomeUser: null, colorized: true, disableHostKeyChecking: true, installation: 'Ansible', inventory: 'inventory', playbook: 'playbook.yml',
    }
}

}
```

Figure 23: Pipeline script for Ansible Deploy

11.5 Jenkins Run

The pipeline script specifies all the steps and the build is triggered based on webhooks.


Stage View

	Declarative: Checkout SCM	Maven Build	Build Docker Images	Publish Docker Images	Clean Docker Images	Deploy and Run Image	Declarative: Post Actions
Average stage times: (Average full run time: ~1min 29s)	2s	9s	8s	16s	1s	14s	3s
#15 Mar 20 22:28 No Changes	1s	5s	9s	30s	1s	28s	3s
#14 Mar 20 22:27 6 commits	3s	14s	7s failed	2s failed	1s failed	1s failed	3s

Figure 24: The stage view on the Jenkins dashboard

12 Ansible Deployment

Ansible is an open-source automation tool that helps in configuring and managing computer systems. It is designed to be simple, agentless, and powerful, allowing users to automate repetitive tasks, such as software installations, configuration management, and application deployment. Ansible fetches the Docker image from the Docker Hub and deploys it across several machines.

12.1 Dockerfile

Dockerfile is a text file that contains a set of instructions used to build a Docker image. A Dockerfile typically contains instructions that describe the base image to use, any additional packages or software that need to be installed, configuration settings, and any other actions that need to be taken to prepare the image.

Using a Dockerfile allows developers to automate the process of building Docker images, making it easier to maintain consistency across different environments and platforms. Dockerfiles can also be version-controlled and shared among team members, making it easier to collaborate on building and deploying applications in a Dockerized environment.

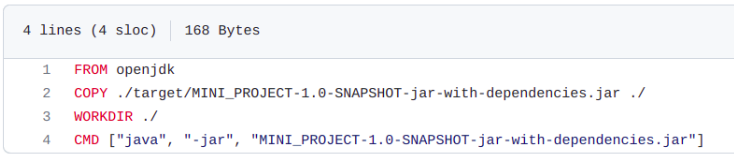
Once a Dockerfile is created, it can be used to build a Docker image using the docker build command. The resulting image can then be used to run containers that include the specified software and configurations.

The following are the main fields of a Dockerfile:

- `FROM`: Specifies the base image that the new image will be built upon.
- `RUN`: Runs a command inside the container while building the image.
- `COPY` or `ADD`: Copies files or directories from the host system into the container.
- `WORKDIR`: Sets the working directory for any `RUN`, `CMD`, `ENTRYPOINT`, `COPY`, and `ADD` instructions that follow.
- `EXPOSE`: Informs Docker that the container will listen on the specified network ports at runtime.
- `CMD or ENTRYPOINT`: Specifies the command to be executed when the container starts.
- `ENV`: Sets environment variables that will be available in the container.

- ARG: Defines arguments that can be passed to the FROM, RUN, and CMD instructions.
- LABEL: Adds metadata to the image.
- USER: Sets the user that will run the subsequent CMD, ENTRYPOINT, COPY, and ADD instructions.
- VOLUME: Creates a mount point for external storage volumes.

These fields allow you to define how the image should be built, what commands should be executed inside the container, and how it should behave when it's run.



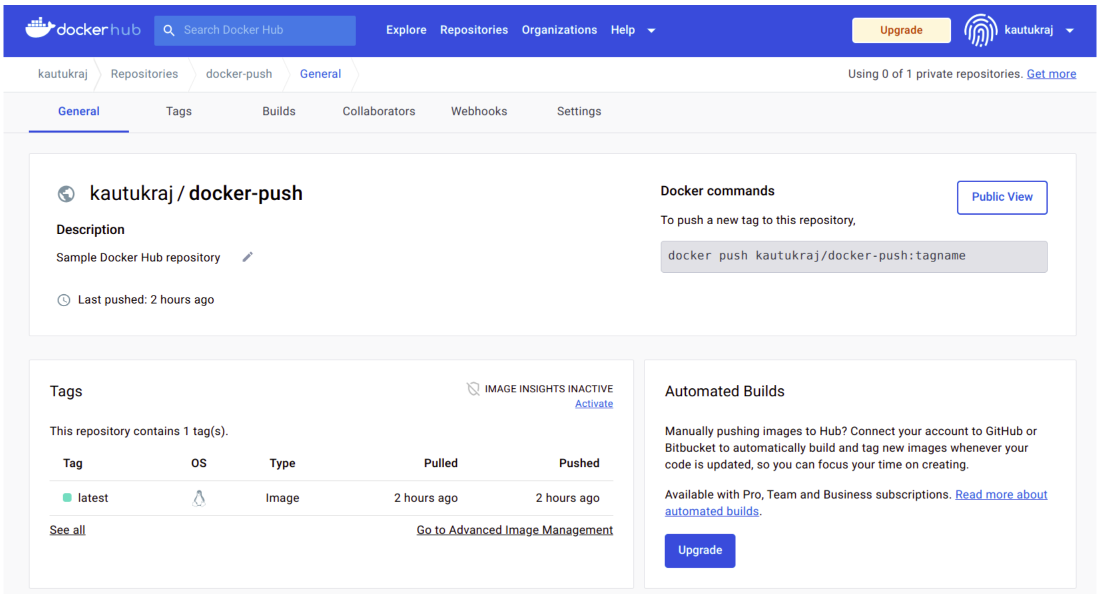
```
4 lines (4 sloc) | 168 Bytes
FROM openjdk
COPY ./target/MINI_PROJECT-1.0-SNAPSHOT-jar-with-dependencies.jar ./
WORKDIR ./
CMD ["java", "-jar", "MINI_PROJECT-1.0-SNAPSHOT-jar-with-dependencies.jar"]
```

Figure 25: The Dockerfile of the project

12.2 Docker Hub

The created image is pushed to the Docker Hub. Docker Hub is a cloud-based repository that allows developers to store and share their Docker container images with other developers or users. It is essentially a central hub for Docker images, where developers can find, download, and share pre-built Docker images that can be used to run their applications in containers.

The image can be found at this URL: `https://hub.docker.com/repository/docker/kautukraj/docker-push/general`.



kautukraj / docker-push

Description

Sample Docker Hub repository

Last pushed: 2 hours ago

Docker commands

Public View

To push a new tag to this repository,

```
docker push kautukraj/docker-push:tagname
```

Tags

IMAGE INSIGHTS INACTIVE
Activate

This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
latest		Image	2 hours ago	2 hours ago

See all

Go to Advanced Image Management

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. Read more about automated builds.

Upgrade

Figure 26: The The published image on the Docker Hub

12.3 Ansible Installation

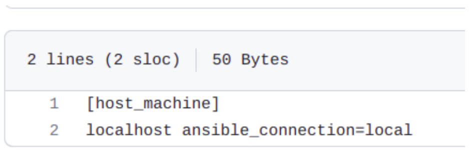
The following steps have to be followed for installing Ansible:

- `sudo apt install openssh-server`

- `service start ssh`
- `ssh-keygen -t rsa`
- `ssh-copy-id <username>@<IP>`
- `sudo apt update`
- `sudo apt install ansible`
- `ansible -version`

12.4 Inventory

In Ansible, an inventory file is a configuration file that specifies the hosts and groups of hosts that Ansible will manage. The inventory file contains a list of hostnames or IP addresses that Ansible will connect to and execute tasks on.



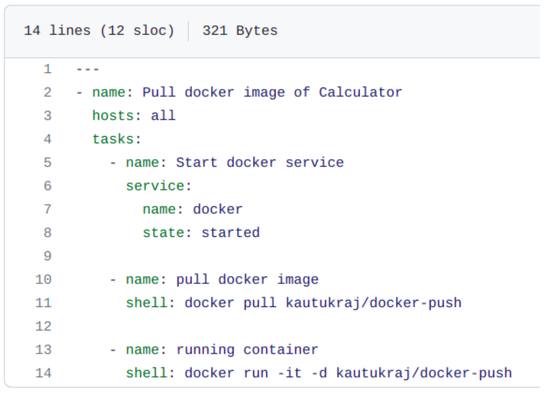
```
2 lines (2 sloc) | 50 Bytes
[host_machine]
localhost ansible_connection=local
```

Figure 27: The inventory file for the project

12.5 Playbook

In Ansible, a playbook is a file or set of files that defines a set of tasks to be executed on one or more hosts. Playbooks are written in YAML format and are used to automate complex tasks such as configuration management, application deployment, and orchestration.

A playbook consists of one or more “plays”, which are a set of tasks that are executed on a group of hosts. Each play specifies the hosts that it applies to, as well as any variables, tasks, and handlers that are required to perform the desired operations.

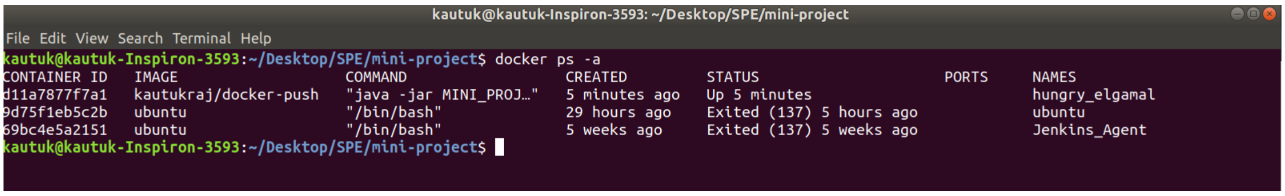


```
14 lines (12 sloc) | 321 Bytes
---
- name: Pull docker image of Calculator
  hosts: all
  tasks:
    - name: Start docker service
      service:
        name: docker
        state: started

    - name: pull docker image
      shell: docker pull kautukraj/docker-push

    - name: running container
      shell: docker run -it -d kautukraj/docker-push
```

Figure 28: The Ansible playbook for the project

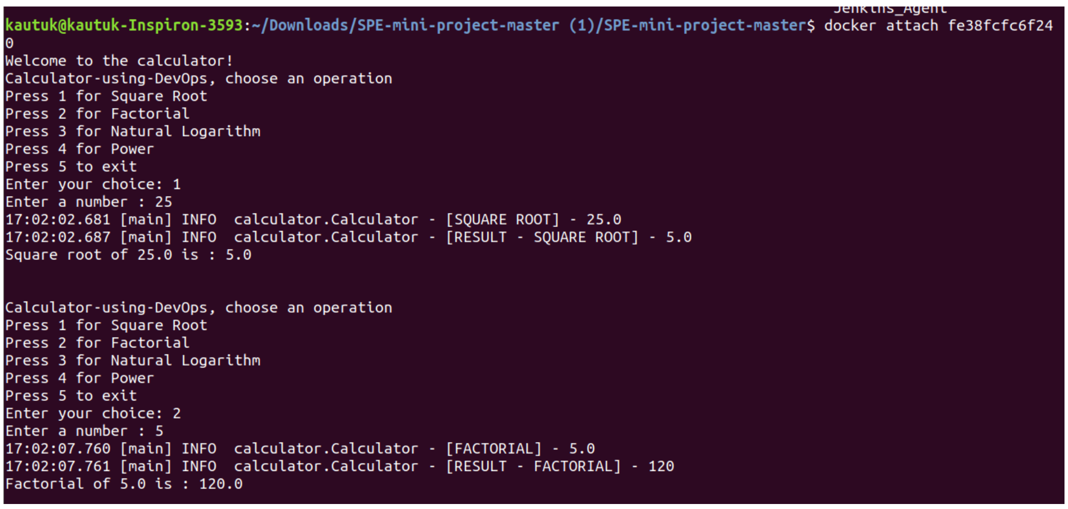
12.6 Ansible Run

Through the playbook, Ansible pulls the image from the Docker Hub and spwans a container on the target configuration(s), in this case, the localhost.



```
kautuk@kautuk-Inspiron-3593: ~/Desktop/SPE/mini-project
File  Edit  View  Search  Terminal  Help
kautuk@kautuk-Inspiron-3593:~/Desktop/SPE/mini-project$ docker ps -a
CONTAINER ID   IMAGE                    COMMAND                  CREATED          STATUS                    PORTS     NAMES
d11a7877f7a1   kautukraj/docker-push    "java -jar MINI_PROJ…"   5 minutes ago    Up 5 minutes                        hungry_elgamal
9d75f1eb5c2b   ubuntu                   "/bin/bash"              29 hours ago     Exited (137) 5 hours ago            ubuntu
69bc4e5a2151   ubuntu                   "/bin/bash"              5 weeks ago      Exited (137) 5 weeks ago            Jenkins_Agent
kautuk@kautuk-Inspiron-3593:~/Desktop/SPE/mini-project$
```

Figure 29: The list of containers, which includes the freshly spawned one, through Ansible



```
kautuk@kautuk-Inspiron-3593:~/Downloads/SPE-mini-project-master (1)/SPE-mini-project-master$ docker attach fe38fcfc6f24
0
Welcome to the calculator!
Calculator-using-DevOps, choose an operation
Press 1 for Square Root
Press 2 for Factorial
Press 3 for Natural Logarithm
Press 4 for Power
Press 5 to exit
Enter your choice: 1
Enter a number : 25
17:02:02.681 [main] INFO  calculator.Calculator - [SQUARE ROOT] - 25.0
17:02:02.687 [main] INFO  calculator.Calculator - [RESULT - SQUARE ROOT] - 5.0
Square root of 25.0 is : 5.0


Calculator-using-DevOps, choose an operation
Press 1 for Square Root
Press 2 for Factorial
Press 3 for Natural Logarithm
Press 4 for Power
Press 5 to exit
Enter your choice: 2
Enter a number : 5
17:02:07.760 [main] INFO  calculator.Calculator - [FACTORIAL] - 5.0
17:02:07.761 [main] INFO  calculator.Calculator - [RESULT - FACTORIAL] - 120
Factorial of 5.0 is : 120.0
```

Figure 30: The jar file can be run by the command docker attach <CONTAINER_ID>. The image shows the execution of the calculator program.

13 Continous Monitoring

The ELK stack is a set of open-source software tools that are commonly used for log management and analysis. ELK is an acronym that stands for Elasticsearch, Logstash, and Kibana.

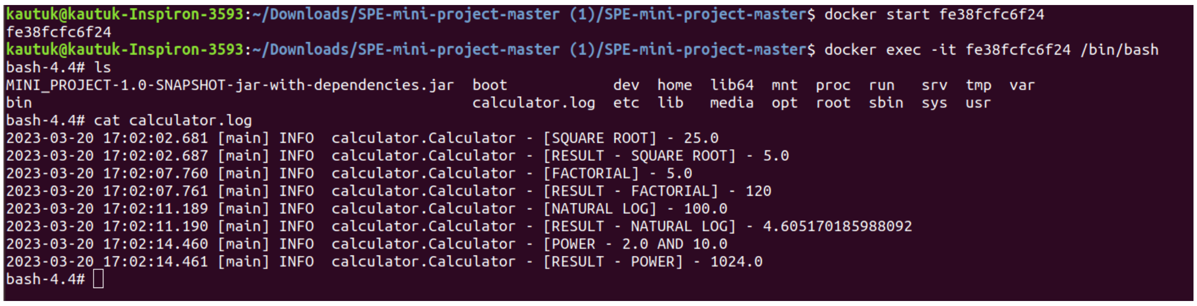
Elasticsearch is a distributed search and analytics engine that is designed to provide fast and efficient full-text search capabilities. It is used to store and index large amounts of data in real-time, and provides advanced querying capabilities.

Logstash is a data processing pipeline that is used to collect, process, and transform data from various sources. It is often used to collect log data from different sources and parse it into a structured format that can be easily analyzed.

Kibana is a data visualization tool that is used to create interactive dashboards and visualizations from data stored in Elasticsearch. It allows users to easily explore and analyze data using a web-based interface, and provides a range of visualization options, including charts, graphs, and maps.

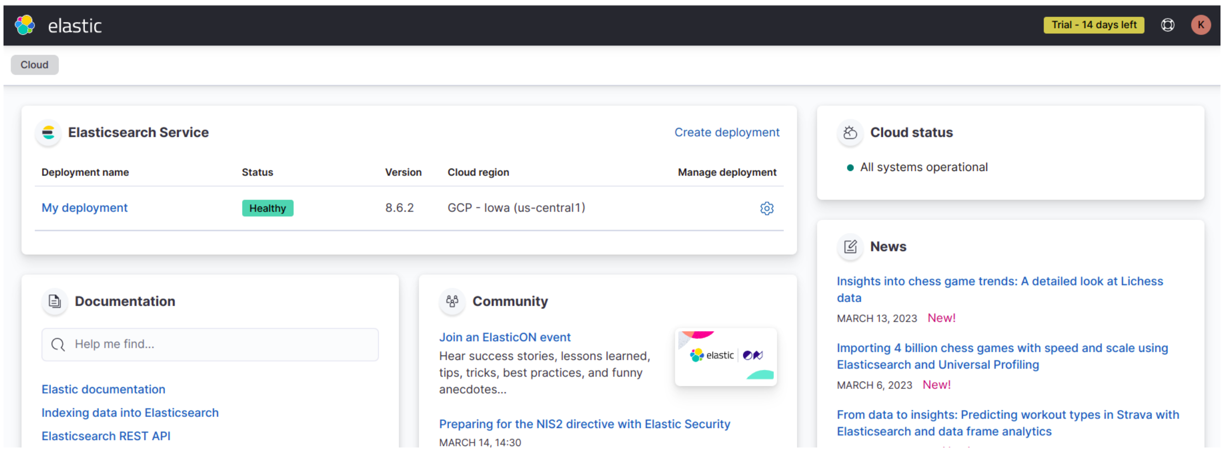
Together, the ELK stack provides a powerful and flexible platform for log management and analysis, and is widely used in a variety of industries, including IT, security, and business intelligence.

The log file can be obtained by running `docker start <CONTAINER_ID>`, `docker exec -it <CONTAINER_ID> /bin/bash`, and `cat calculator.log`.



```
kautuk@kautuk-Inspiron-3593:~/Downloads/SPE-mini-project-master (1)/SPE-mini-project-master$ docker start fe38fcfc6f24
fe38fcfc6f24
kautuk@kautuk-Inspiron-3593:~/Downloads/SPE-mini-project-master (1)/SPE-mini-project-master$ docker exec -it fe38fcfc6f24 /bin/bash
bash-4.4# ls
MINI_PROJECT-1.0-SNAPSHOT-jar-with-dependencies.jar  boot            dev  home  lib64  mnt  proc  run   srv  tmp  var
bin                                                  calculator.log  etc  lib   media  opt  root  sbin  sys  usr
bash-4.4# cat calculator.log
2023-03-20 17:02:02.681 [main] INFO  calculator.Calculator - [SQUARE ROOT] - 25.0
2023-03-20 17:02:02.687 [main] INFO  calculator.Calculator - [RESULT - SQUARE ROOT] - 5.0
2023-03-20 17:02:07.760 [main] INFO  calculator.Calculator - [FACTORIAL] - 5.0
2023-03-20 17:02:07.761 [main] INFO  calculator.Calculator - [RESULT - FACTORIAL] - 120
2023-03-20 17:02:11.189 [main] INFO  calculator.Calculator - [NATURAL LOG] - 100.0
2023-03-20 17:02:11.190 [main] INFO  calculator.Calculator - [RESULT - NATURAL LOG] - 4.605170185988092
2023-03-20 17:02:14.460 [main] INFO  calculator.Calculator - [POWER - 2.0 AND 10.0
2023-03-20 17:02:14.461 [main] INFO  calculator.Calculator - [RESULT - POWER] - 1024.0
bash-4.4# 
```

Figure 31: The log file generated through log4j, which shall be fed to the ELK stack



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

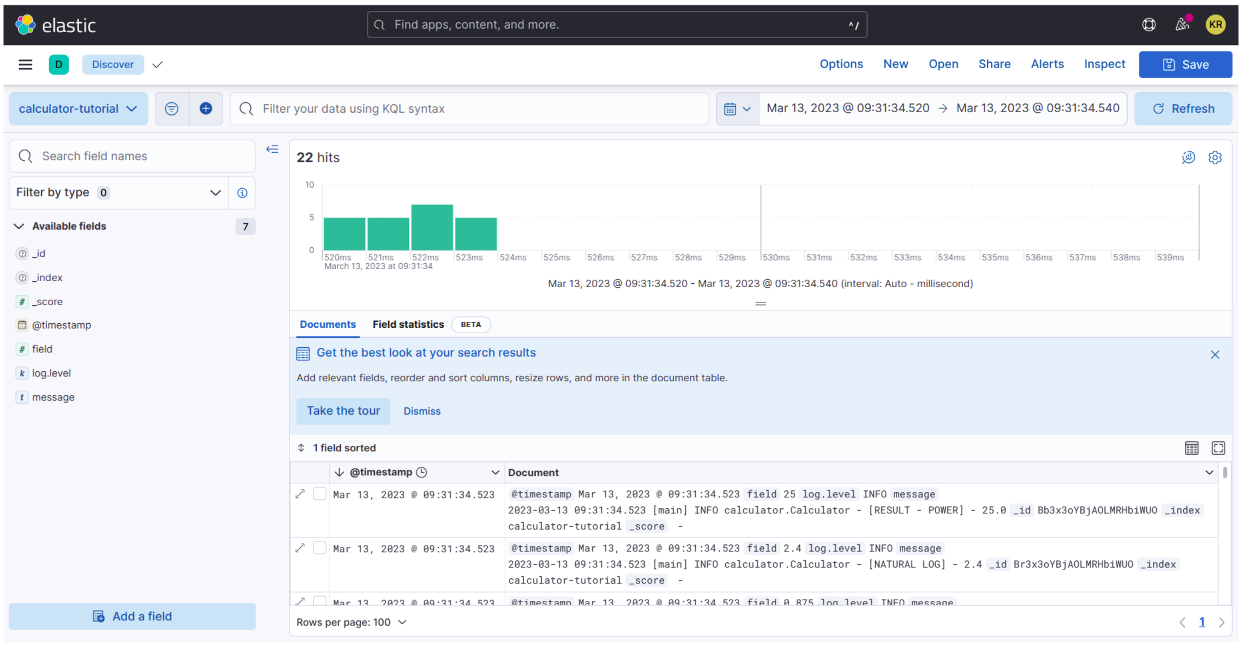
Hear success stories, lessons learned, tips, tricks, best practices, and funny anecdotes...

Insights into chess game trends: A detailed look at Lichess data
MARCH 13, 2023 New!

Importing 4 billion chess games with speed and scale using Elasticsearch and Universal Profiling
MARCH 6, 2023 New!

From data to insights: Predicting workout types in Strava with Elasticsearch and data frame analytics

Figure 32: The Elastic Cloud dashboard



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

@timestamp	Document
Mar 13, 2023 @ 09:31:34.523	@timestamp Mar 13, 2023 @ 09:31:34.523 field 25 log.level INFO message 2023-03-13 09:31:34.523 [main] INFO calculator.Calculator - [RESULT - POWER] - 25.0 _id Bb3x3oYBjAOLMRHbiWUO _index calculator-tutorial _score -
Mar 13, 2023 @ 09:31:34.523	@timestamp Mar 13, 2023 @ 09:31:34.523 field 2.4 log.level INFO message 2023-03-13 09:31:34.523 [main] INFO calculator.Calculator - [NATURAL LOG] - 2.4 _id Br3x3oYBjAOLMRHbiWUO _index calculator-tutorial _score -
Mar 13, 2023 @ 09:31:34.523	@timestamp Mar 13, 2023 @ 09:31:34.523 field 0.875 log.level INFO message

Figure 33: The monitoring dashboard on Kibana