Course: DevOps | Name: Billipati Sai Teja
Module: Docker & Docker Hub | Batch no: 115
Topic: SonarQube, Nexus, Tomcat | Assignment no: 04
Trainer Name: Mr. Madhukar sir | Date of submission: 15th – Dec – 2023
Mail-ID: (BILLIPATISAITEJA@GMAIL.COM)

Assignment: SonarQube do code quality analysis, Store in nexus artifacts and Deploy in tomcat using project GitHub link: https://github.com/Venna12/dockerjenkin.git

Pre-requirements:

1. Project link from GitHub (Venna12).
2. Install Java jdk-11 and maven.
3. Jenkins and setup Jenkins (Install require plugs).
4. Install SonarQube and setup environment.
5. Install Nexus Artifacts and setup environment.
6. Install Tomcat and setup environment.

---------------------------- EC2 Connect ----------------------------

➔ Launch the EC2 instances, take instance type (t2.medium or t2.large), select Ubuntu OS for the project and wait the instances to change status pending to running state.

➔ Edit the Security Bounds and add the security bound (use port no or use All traffic and Anywhere 0.0.0./0) to run the servers in the Google.

➔ Connect the cloud command line interface.

➔ Change normal user to root user (using sudo -i).

➔ Update the Linux server using command (apt update -y).

------------------------- Install Java jdk-11 and Maven -------------------------

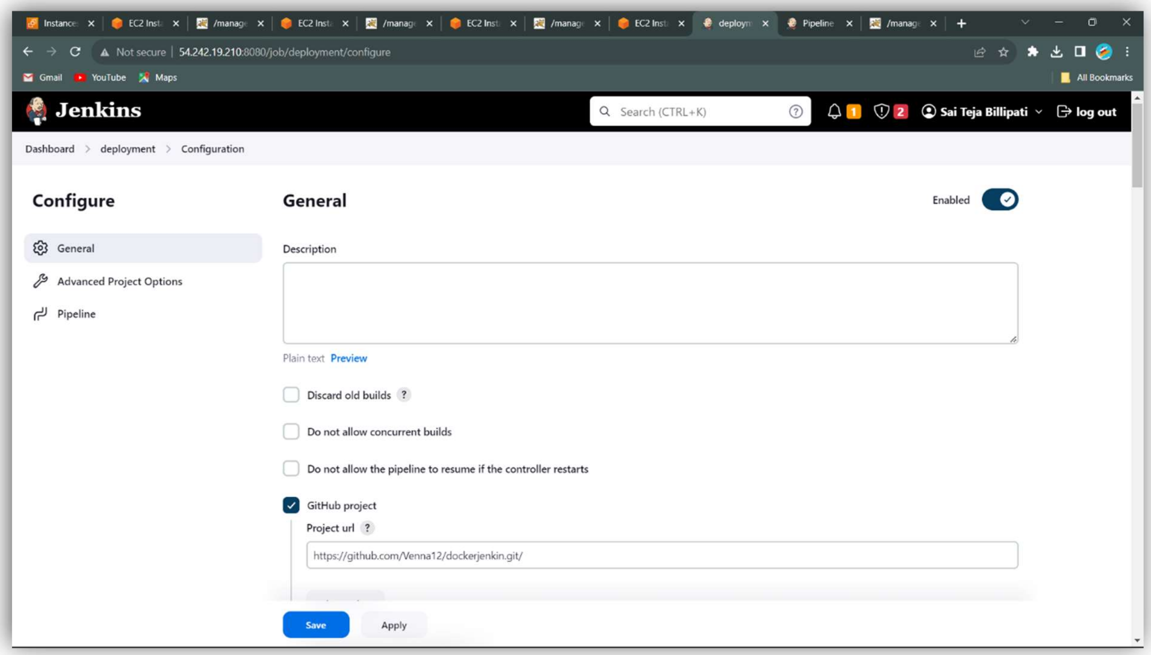
-➔ Use the command to install Java jdk-11 in the server (sudo apt install default-jdk -y or apt install openjdk-11-jdk -y).

-➔ Use the command to install Maven in the server (sudo apt install maven -y).

To Check install Java and Maven: java --version, mvn --version.

--------------- Creating a Pipeline in the Jenkins for Project ---------------

- Open the Jenkins using the AWS Public IPV4 and using port no 8080.
- Create the Job using pipeline. Job name Dockerassgin.
- We got configure dashboard, and click on the Git, add the clone link to given box.



Not secure | 54.242.19.210:8080/job/deployment/configure

Jenkins

Dashboard > deployment > Configuration

Configure

- General
- Advanced Project Options
- Pipeline

General — Enabled

Description

Plain text Preview

- [ ] Discard old builds
- [ ] Do not allow concurrent builds
- [ ] Do not allow the pipeline to resume if the controller restarts
- [✓] GitHub project

Project url: https://github.com/Venna12/dockerjenkin.git/

Save | Apply

Project URL: https://github.com/Venna12/dockerjenkin.git

------------------------------ SonarQube ------------------------------

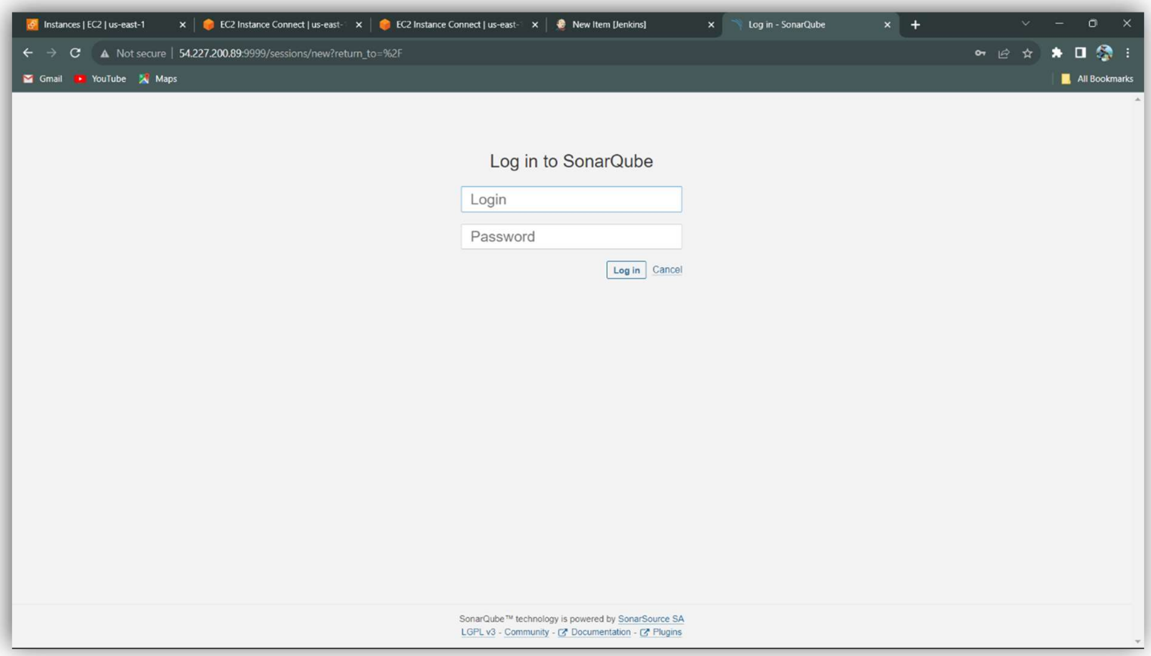
- SonarQube is a static code analysis tool, a tool that scans your code and tries to detect flaws, bugs, security vulnerabilities, etc. It can also measure test coverage of your code if provided with proper reports. All these features focus on direct code development and help developers build better products. SonarQube is an open-source platform developed by Sonar Source for continuous inspection of code quality to perform automatic reviews with static analysis of code to detect bugs and code smells on 29 programming languages.
- SonarQube is start using port no 9000. IP:9000. To Sonarqube software.
- We can install SonarQube manual or Using Docker we can easily install SonarQube.

Using Docker: Install Docker: sudo apt install docker.io -y

Install SonarQube: docker run --name "Name of the container" -d -p 9000:9000 sonarqube:latest.

Default → Username: admin

Password: admin


Instances | EC2 | us-east-1
EC2 Instance Connect | us-east-
EC2 Instance Connect | us-east-
New Item [Jenkins]
Log in - SonarQube
Not secure | 54.227.200.89:9999/sessions/new?return_to=%2F
Gmail
YouTube
Maps
All Bookmarks
Log in to SonarQube
Login
Password
Log in
Cancel
SonarQube™ technology is powered by SonarSource SA
LGPL v3 - Community - Documentation - Plugins

Interface of the SonarQube for Log in to SonarQube.

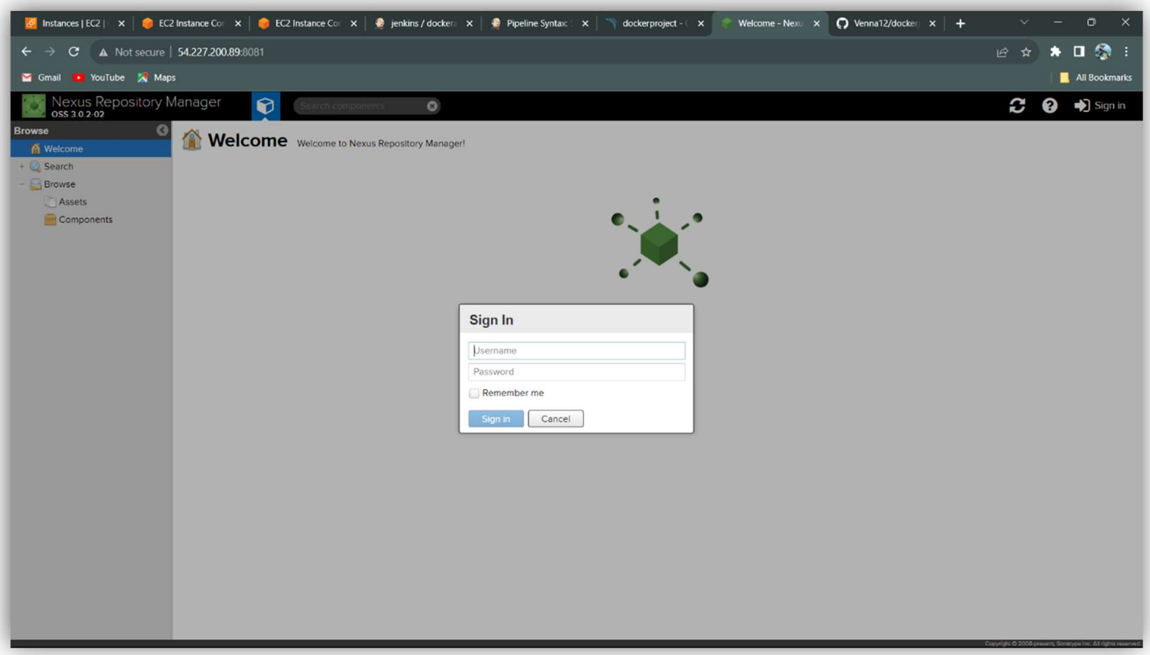
-------------------------------- Nexus --------------------------------

- ➢ Nexus by Sonatype is a repository manager that organizes, stores and distributes artifacts needed for development. With Nexus, developers can completely control access to, and deployment of, every artifact in an organization from a single location, making it easier to distribute software.
- ➢ Nexus is pure written in the Java. But it can store the 29 other programming languages artifacts in it.
- ➢ Nexus is run on Java jdk-8 version only.
- ➢ To run the Nexus use server IP address and Nexus Port no: 8081.

  Server Ip:8081 to open Nexus on the any search engine.

  Default → Username: admin

  Password: admin123.



User interface of the Nexus.

------------------------------ Apache Tomcat ------------------------------

- Tomcat Apache Tomcat is a free and open-source implementation of the Jakarta Servlet, Jakarta Expression Language, and WebSocket technologies. It provides a "pure Java" HTTP web server environment in which Java code can also run. Thus, it is a Java web application server, although not a full JEE application server.
- By default, Tomcat starts up on HTTP connector 8080. If another application on the install machine is already using port 8080 (for example, if you have another instance of Tomcat on the machine), then change the default startup port by modifying the conf/server. xml file.

  To sign in the Tomcat, we are config the details of the user and password.

  >>>Username: deployer

  >>>Password: deployer

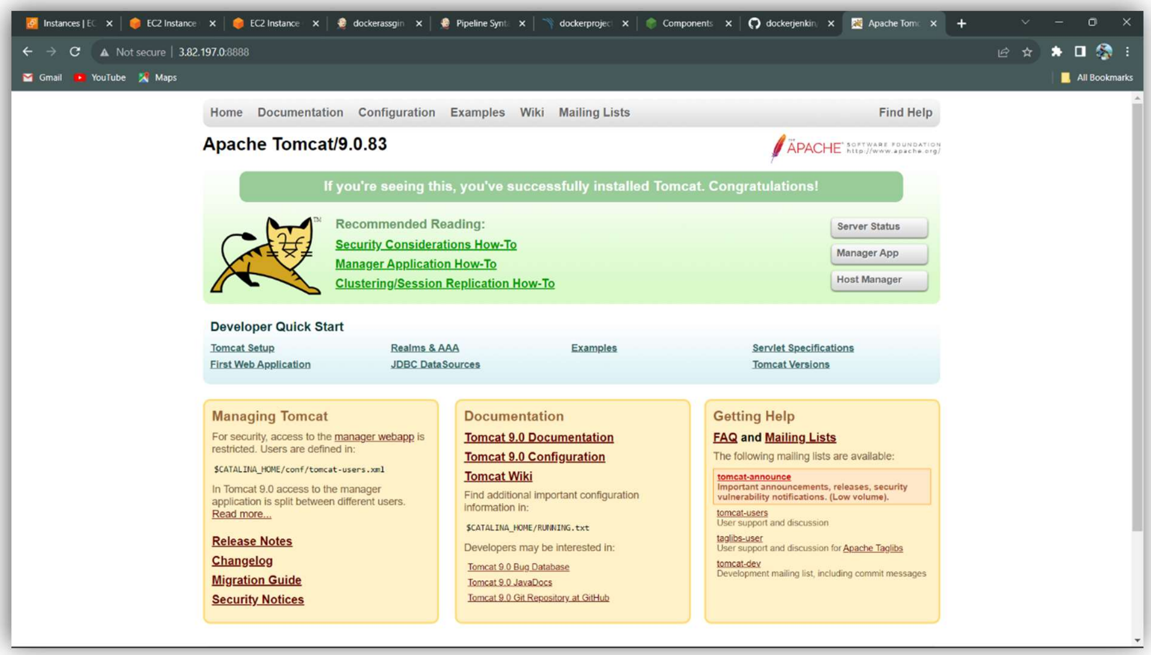
  Login details of the Apache Tomcat server.

  >>>Login: tomcat

  >>>Password: s3cret

  To open Tomcat server use IP address and port of Tomcat.

  → IP Address:8080 (Port no).



Apache Tomcat/9.0.83

If you're seeing this, you've successfully installed Tomcat. Congratulations!

Recommended Reading:
Security Considerations How-To
Manager Application How-To
Clustering/Session Replication How-To

Developer Quick Start
Tomcat Setup
First Web Application
Realms & AAA
JDBC DataSources
Examples
Servlet Specifications
Tomcat Versions

Managing Tomcat
For security, access to the manager webapp is restricted. Users are defined in:
`$CATALINA_HOME/conf/tomcat-users.xml`
In Tomcat 9.0 access to the manager application is split between different users. Read more...
Release Notes
Changelog
Migration Guide
Security Notices

Documentation
Tomcat 9.0 Documentation
Tomcat 9.0 Configuration
Tomcat Wiki
Find additional important configuration information in:
`$CATALINA_HOME/RUNNING.txt`
Developers may be interested in:
Tomcat 9.0 Bug Database
Tomcat 9.0 JavaDocs
Tomcat 9.0 Git Repository at GitHub

Getting Help
FAQ and Mailing Lists
The following mailing lists are available:
tomcat-announce
Important announcements, releases, security vulnerability notifications. (Low volume).
tomcat-users
User support and discussion
taglibs-user
User support and discussion for Apache Taglibs
tomcat-dev
Development mailing list, including commit messages

User Interface of the Apache Tomcat/9.0.83.

-→ Write a pipeline script for the Git clone for the project using link

Project URL: https://github.com/Venna12/dockerjenkin.git

1. Create a pipeline for Dockerjenkin:
   a. Clone the Project
   b. Validate the Project
   c. Compile the Project
   d. Test the Project
2. Pipeline script for the Project

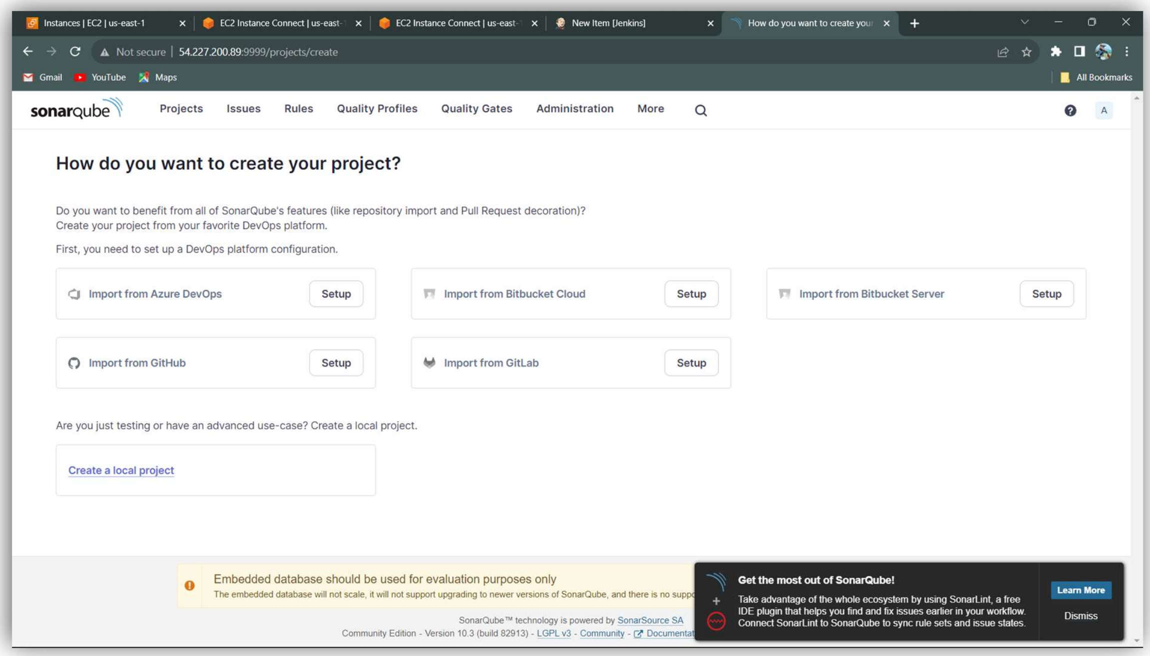
   Script:

```
pipeline{
    agent any
    stages{
        stage('Cloning the Project'){
            steps{
                checkout scmGit(branches: [[name: '*/master']], extensions: [], userRemoteConfigs: [[url: 'https://github.com/Venna12/dockerjenkin.git']])
            }
        }
        stage('validate the Project'){
            steps{
                sh 'mvn validate'
            }
        }
```

```
stage('compile the Project'){
    steps{
        sh 'mvn compile'
    }
}
stage('test the Project'){
    steps{
        sh 'mvn test'
    }
}
```

Configuration of SonarQube to Jenkins pipeline:

1. Open the SonarQube using Server IP address with port no.
    a. Server Ip:port no (9000)
2. Login to SonarQube using default credentials.
    a. Username: admin
    b. Password: admin
3. Once login to SonarQube to change default password.
4. It redirects to SonarQube Dashboard.



How do you want to create your project?

Do you want to benefit from all of SonarQube's features (like repository import and Pull Request decoration)?
Create your project from your favorite DevOps platform.

First, you need to set up a DevOps platform configuration.

Are you just testing or have an advanced use-case? Create a local project.

Embedded database should be used for evaluation purposes only
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no suppo

SonarQube™ technology is powered by SonarSource SA
Community Edition - Version 10.3 (build 82913) - LGPL v3 - Community - Documentat

Get the most out of SonarQube!
Take advantage of the whole ecosystem by using SonarLint, a free IDE plugin that helps you find and fix issues earlier in your workflow. Connect SonarLint to SonarQube to sync rule sets and issue states.

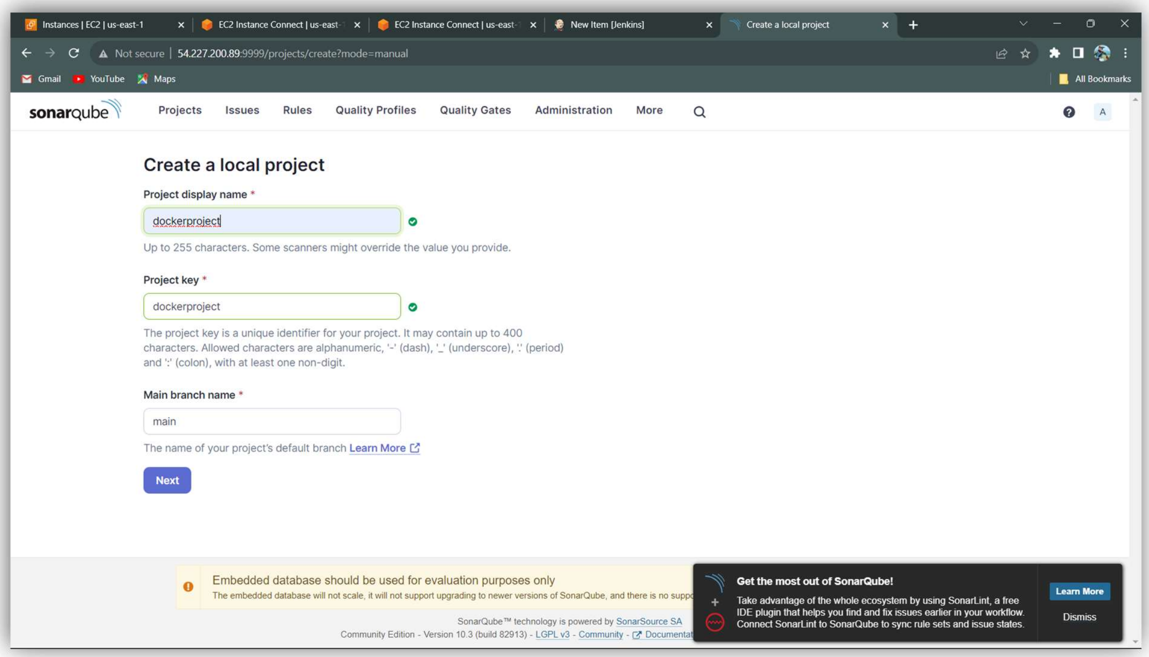
SonarQube Dashboard

5. Click on the Create a local project.

    a. Project display name: dockerproject

    b. Project key: dockerproject

    c. Main branch name: main



Create a local project

Project display name *

dockerproject

Up to 255 characters. Some scanners might override the value you provide.

Project key *

dockerproject

The project key is a unique identifier for your project. It may contain up to 400 characters. Allowed characters are alphanumeric, '-' (dash), '_' (underscore), '.' (period) and ':' (colon), with at least one non-digit.

Main branch name *

main

The name of your project's default branch Learn More

Embedded database should be used for evaluation purposes only

The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no supp

Get the most out of SonarQube!

Take advantage of the whole ecosystem by using SonarLint, a free IDE plugin that helps you find and fix issues earlier in your workflow. Connect SonarLint to SonarQube to sync rule sets and issue states.

SonarQube™ technology is powered by SonarSource SA

Community Edition - Version 10.3 (build 82913) - LGPL v3 - Community - Documentat

6. It shows different options for the options

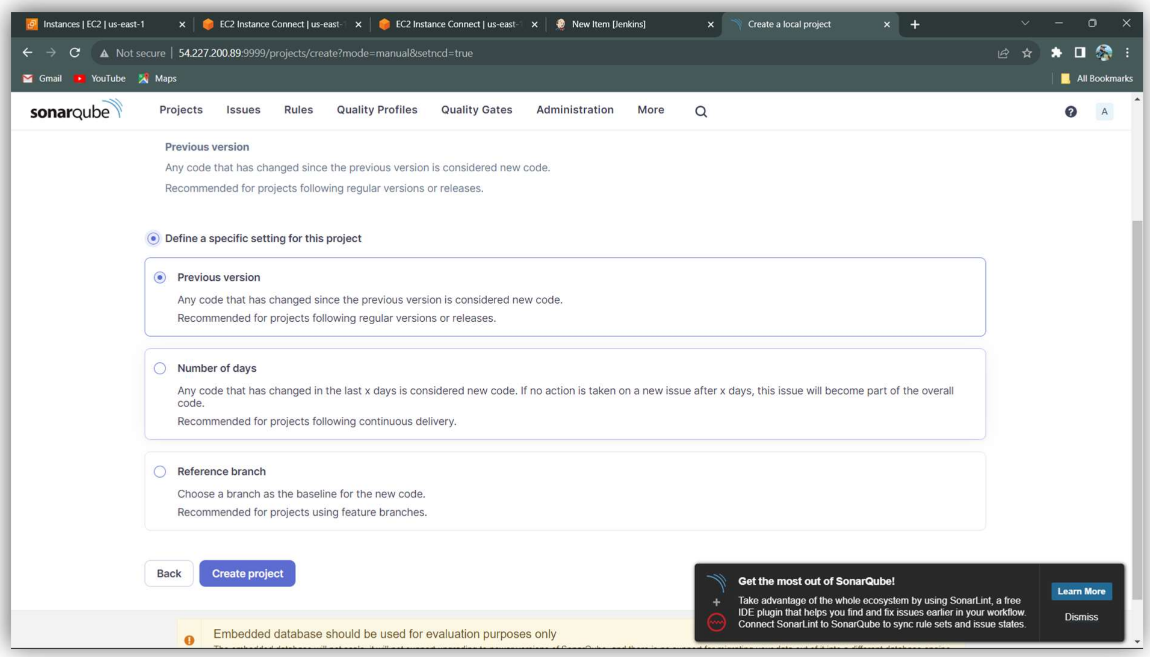
    a. Previous version

    b. Number of days

    c. Reference branch

7. Select the Previous version for this project

8. Click on the create project.


Instances | EC2 | us-east-1
EC2 Instance Connect | us-east-
EC2 Instance Connect | us-east-
New Item [Jenkins]
Create a local project
Not secure | 54.227.200.89:9999/projects/create?mode=manual&setncd=true
Gmail
YouTube
Maps
All Bookmarks
sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
More

Previous version

Any code that has changed since the previous version is considered new code.

Recommended for projects following regular versions or releases.

(●) Define a specific setting for this project

(●) Previous version

Any code that has changed since the previous version is considered new code.

Recommended for projects following regular versions or releases.

( ) Number of days

Any code that has changed in the last x days is considered new code. If no action is taken on a new issue after x days, this issue will become part of the overall code.

Recommended for projects following continuous delivery.

( ) Reference branch

Choose a branch as the baseline for the new code.

Recommended for projects using feature branches.

Get the most out of SonarQube!

Take advantage of the whole ecosystem by using SonarLint, a free IDE plugin that helps you find and fix issues earlier in your workflow. Connect SonarLint to SonarQube to sync rule sets and issue states.

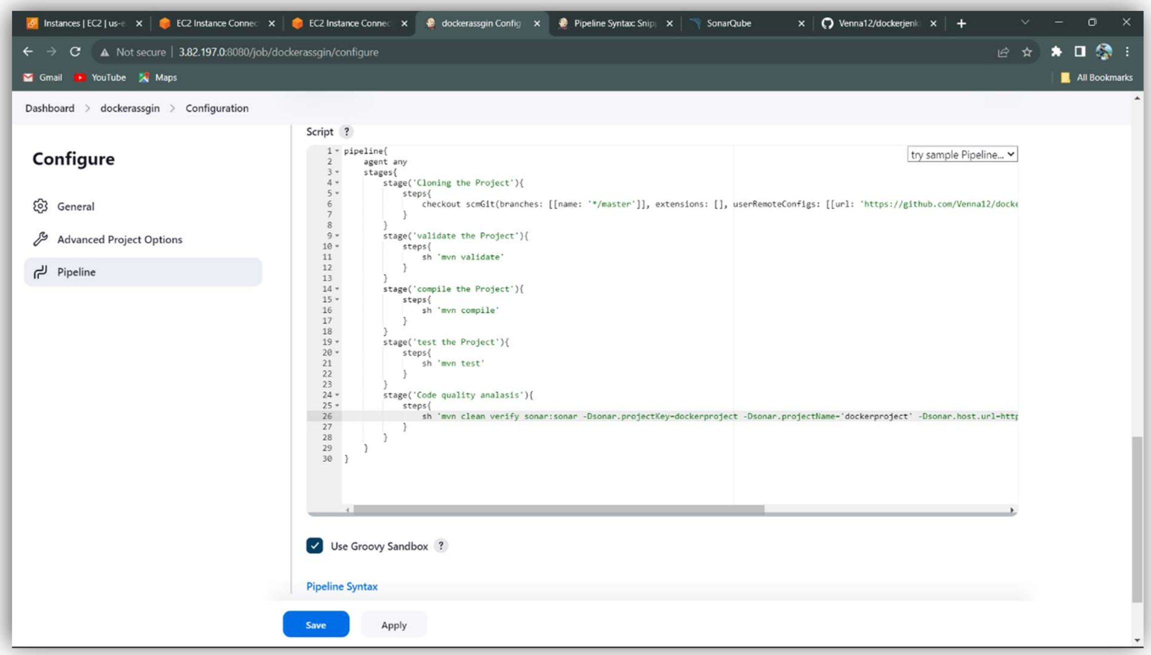
Embedded database should be used for evaluation purposes only

9. It generates Token name “dockerproject”, Expires in 30days, Click on Generate.
10. Run analysis on the project – select the maven it generates the maven code.

```
mvn clean verify sonar:sonar -Dsonar.projectKey=dockerproject
-Dsonar.projectName='dockerproject'
-Dsonar.host.url=http://54.227.200.89:9999
-Dsonar.token=sqp_7e026e6c571382aca2856db896ab7d99a5d9dc52
```

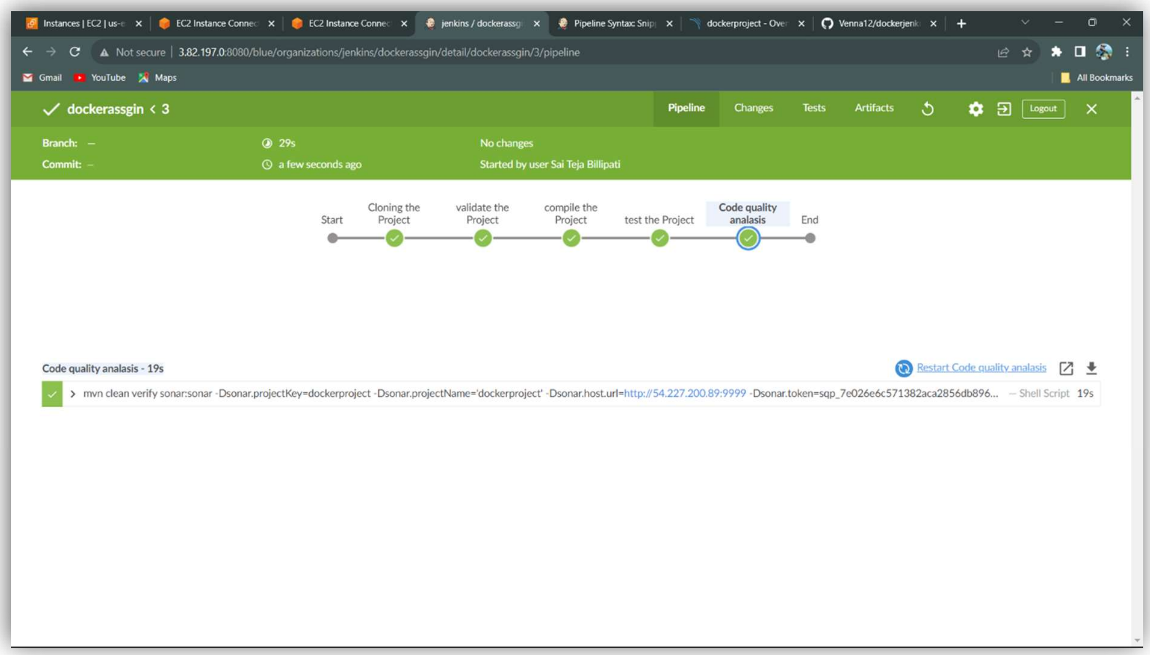
11. Copy the maven script and add to Jenkins Pipeline.

```
stage('Code quality analasis'){
    steps{
        sh "mvn clean verify sonar:sonar -Dsonar.projectKey=dockerproject -Dsonar.projectName='dockerproject' -Dsonar.host.url=http://54.227.200.89:9999 -Dsonar.token=sqp_7e026e6c571382aca2856db896ab7d99a5d9dc52"
    }
}
```



```
pipeline{
    agent any
    stages{
        stage('Cloning the Project'){
            steps{
                checkout scmGit(branches: [[name: '*/master']], extensions: [], userRemoteConfigs: [[url: 'https://github.com/Venna12/docke
            }
        }
        stage('validate the Project'){
            steps{
                sh 'mvn validate'
            }
        }
        stage('compile the Project'){
            steps{
                sh 'mvn compile'
            }
        }
        stage('test the Project'){
            steps{
                sh 'mvn test'
            }
        }
        stage('Code quality analasis'){
            steps{
                sh 'mvn clean verify sonar:sonar -Dsonar.projectKey=dockerproject -Dsonar.projectName='dockerproject' -Dsonar.host.url=http
            }
        }
    }
}
```

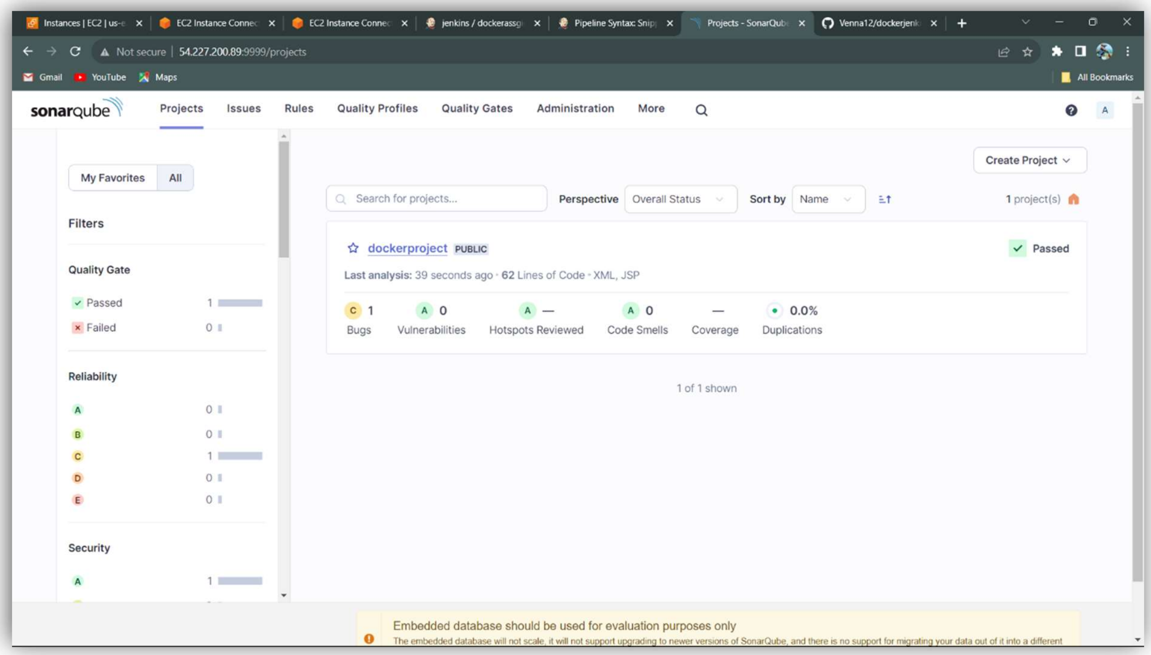
12. Apply and save the pipeline. Build the Pipeline.



Code quality analasis - 19s

```
> mvn clean verify sonar:sonar -Dsonar.projectKey=dockerproject -Dsonar.projectName='dockerproject' -Dsonar.host.url=http://54.227.200.89:9999 -Dsonar.token=sqp_7e026e6c571382aca2856db896... — Shell Script 19s
```

13. Code Quality analysis is build Successfully with the SonarQube.
14. It shows the Detect flaws, Bugs, Security Vulnerabilities, Code coverage, in the SonarQube.



dockerproject PUBLIC — Passed

Last analysis: 39 seconds ago · 62 Lines of Code · XML, JSP

Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications
C 1	A 0	A —	A 0	—	0.0%

1 of 1 shown

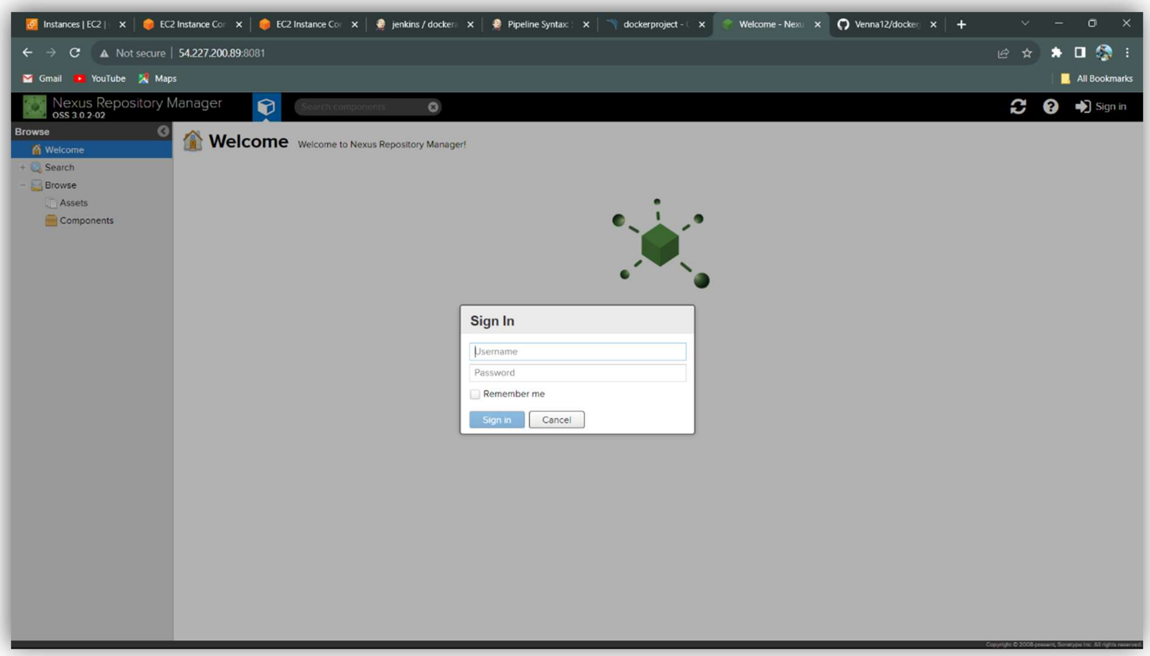
Embedded database should be used for evaluation purposes only

The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different

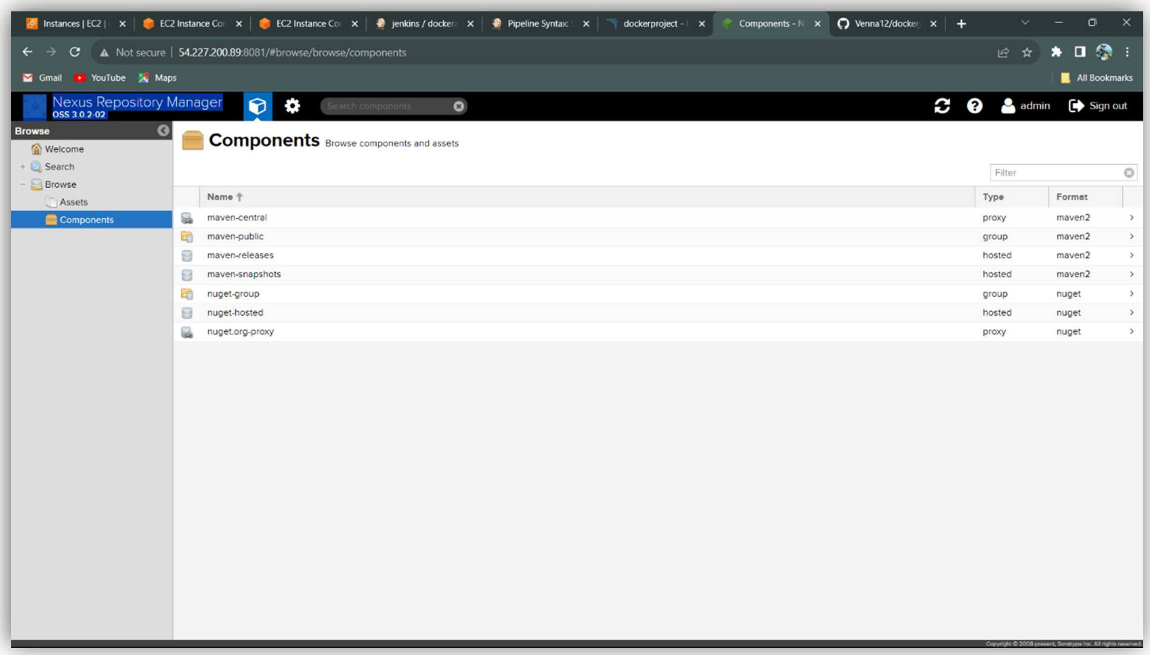
Code Quality Analysis with SonarQube.

Configuration of Nexus to Jenkins pipeline:

1. Open Nexus server using IP address with Port no.
    a. IP address:Port no (8081).
2. Login nexus server using admin credentials.
    a. Username: admin
    b. Password: admin123



3. Open Components and Click on Maven-Snapshots.
4. Open Maven-snapshots repository.



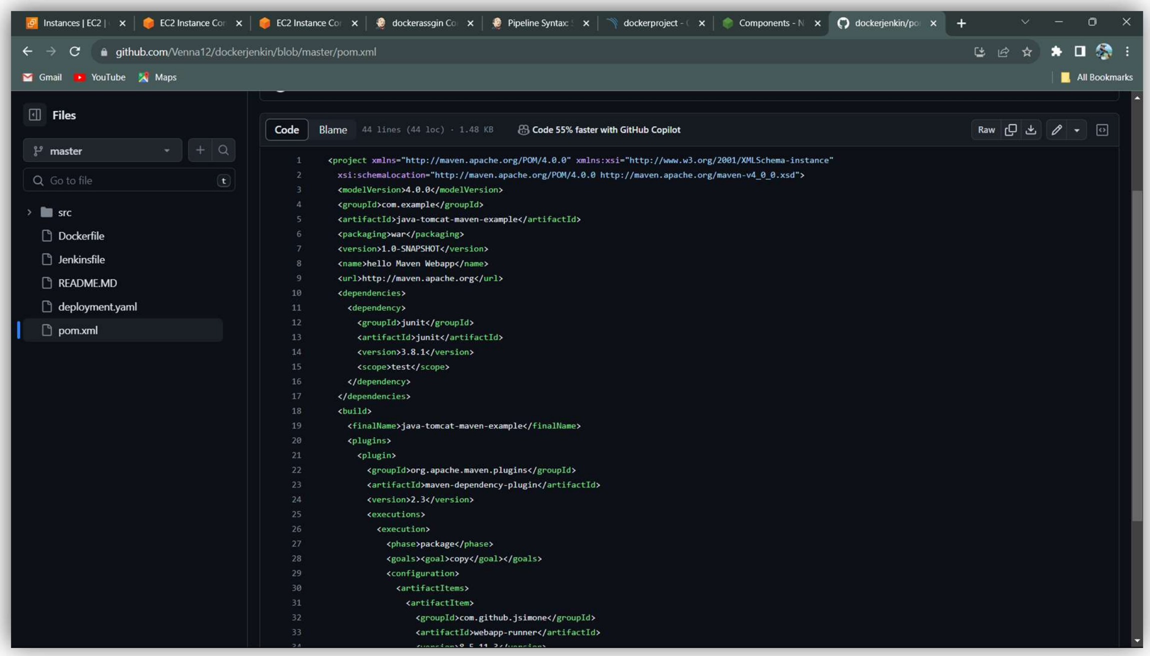
Components Browse components and assets

Name	Type	Format
maven-central	proxy	maven2
maven-public	group	maven2
maven-releases	hosted	maven2
maven-snapshots	hosted	maven2
nuget-group	group	nuget
nuget-hosted	hosted	nuget
nuget.org-proxy	proxy	nuget

5. Open Jenkins pipeline and pipeline script.
    a. Add script, to pipeline after SonarQube.
6. Add script of maven to pipeline using stage.

```
stage('Package'){
        steps{
            sh 'mvn package'
        }
    }
```

7. Go to console output, after build success.
8. Copy Building war file from the console output.
    a. /var/lib/jenkins/workspace/dockerassgin/target/java-tomcat-maven-example.war
9. Save it to notepad and open Jenkins pipeline.
10. Open pipeline syntax on other tab.
11. Open pom.xml file for the artifact information.



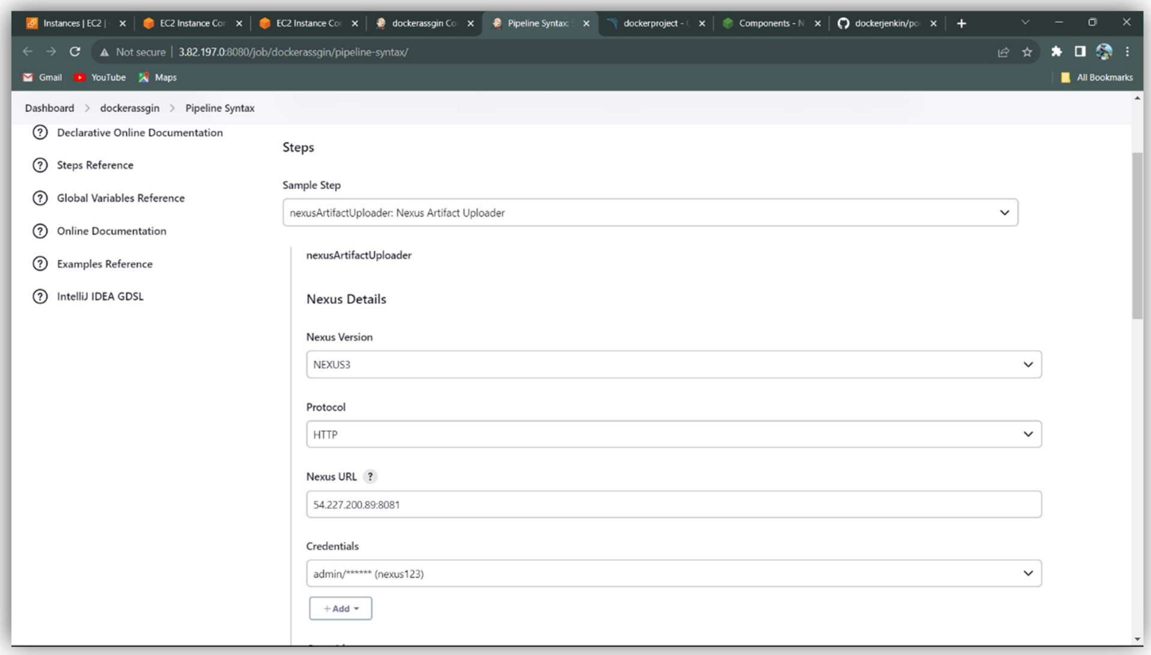
```xml
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/maven-v4_0_0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.example</groupId>
  <artifactId>java-tomcat-maven-example</artifactId>
  <packaging>war</packaging>
  <version>1.0-SNAPSHOT</version>
  <name>hello Maven Webapp</name>
  <url>http://maven.apache.org</url>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>3.8.1</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
  <build>
    <finalName>java-tomcat-maven-example</finalName>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-dependency-plugin</artifactId>
        <version>2.3</version>
        <executions>
          <execution>
            <phase>package</phase>
            <goals><goal>copy</goal></goals>
            <configuration>
              <artifactItems>
                <artifactItem>
                  <groupId>com.github.jsimone</groupId>
                  <artifactId>webapp-runner</artifactId>
```

Pom.xml file

12. Open pipeline syntax.
    a. Sample step
        i. nexusArtifactUploader: Nexus Artifact Uploader
    b. Nexus Version
        i. NEXUS 3
    c. Protocol
        i. HTTP
    d. Nexus URL
        i. 54.227.200.89:8081
    e. Credentials
        i. Create Jenkins for credentials using
            1. Username: admin
            2. Password: admin123
    f. GroupId
        i. Com.example
    g. Version
        i. 1.0-SNAPSHOT
    h. Repository
        i. Maven-snapshots (Repository from the Nexus)
    i. Add Artifact
    j. Artifact Id
        i. Java-tomcat-maven-example
    k. Type
        i. War
    l. File
        i. /var/lib/jenkins/workspace/dockerassgin/target/java-tomcat-maven-example.war (Building war from console output)
13. Generate Pipeline and copy the code

14. Paste the generate pipeline syntax script to pipeline.

```
stage('Store in nexus artifacts'){
        steps{
            nexusArtifactUploader artifacts: [[artifactId: 'java-tomcat-maven-
example', classifier: '', file:
'/var/lib/jenkins/workspace/dockerassgin/target/java-tomcat-maven-
example.war', type: 'war']], credentialsId: 'nexus123', groupId: 'com.example',
nexusUrl: '54.227.200.89:8081', nexusVersion: 'nexus3', protocol: 'http',
repository: 'maven-snapshots', version: '1.0-SNAPSHOT'
        }
    }
```



Steps

Sample Step: nexusArtifactUploader: Nexus Artifact Uploader

nexusArtifactUploader

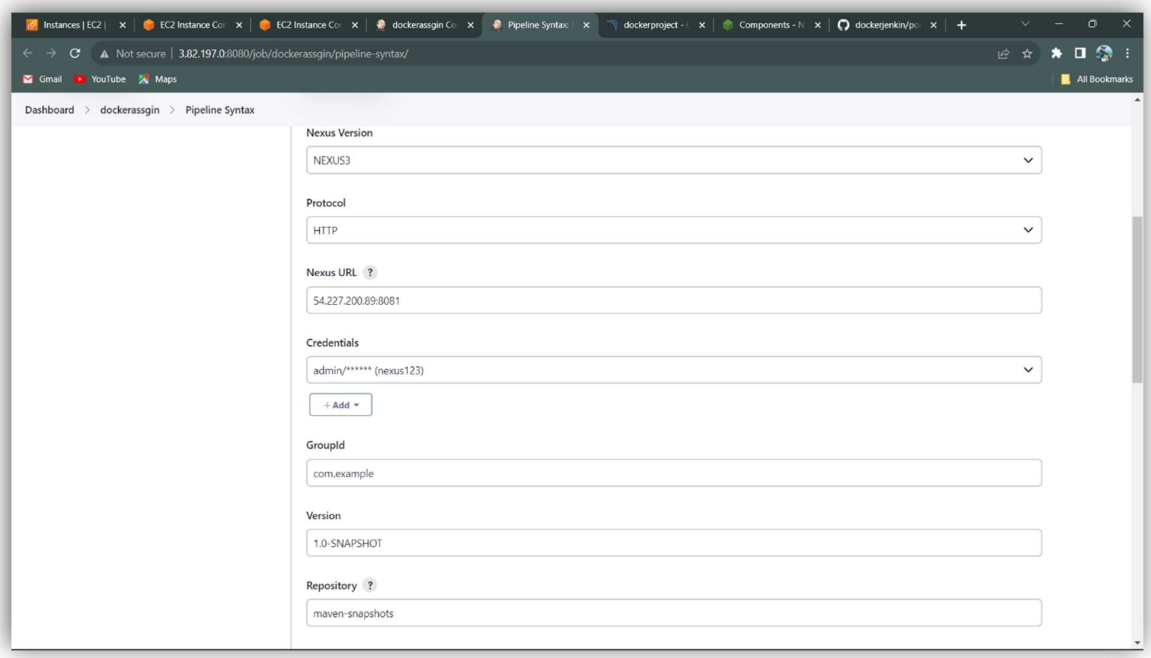
Nexus Details

Nexus Version: NEXUS3

Protocol: HTTP

Nexus URL: 54.227.200.89:8081

Credentials: admin/****** (nexus123)



Nexus Version: NEXUS3

Protocol: HTTP

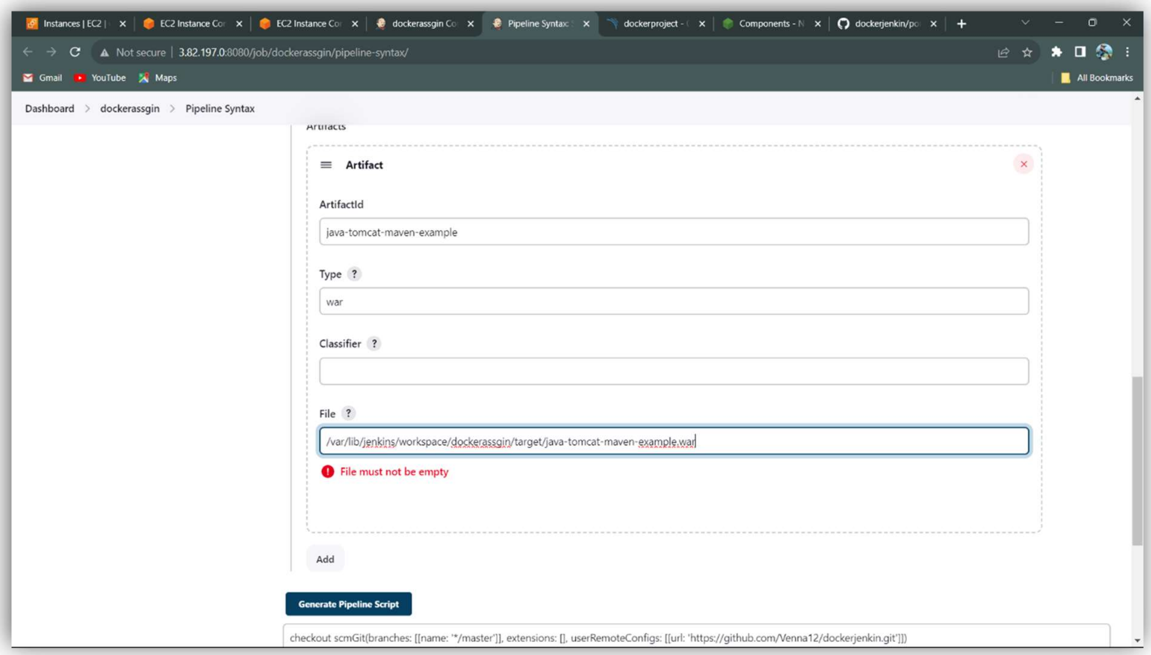
Nexus URL: 54.227.200.89:8081

Credentials: admin/****** (nexus123)

GroupId: com.example

Version: 1.0-SNAPSHOT

Repository: maven-snapshots



ArtifactId: java-tomcat-maven-example

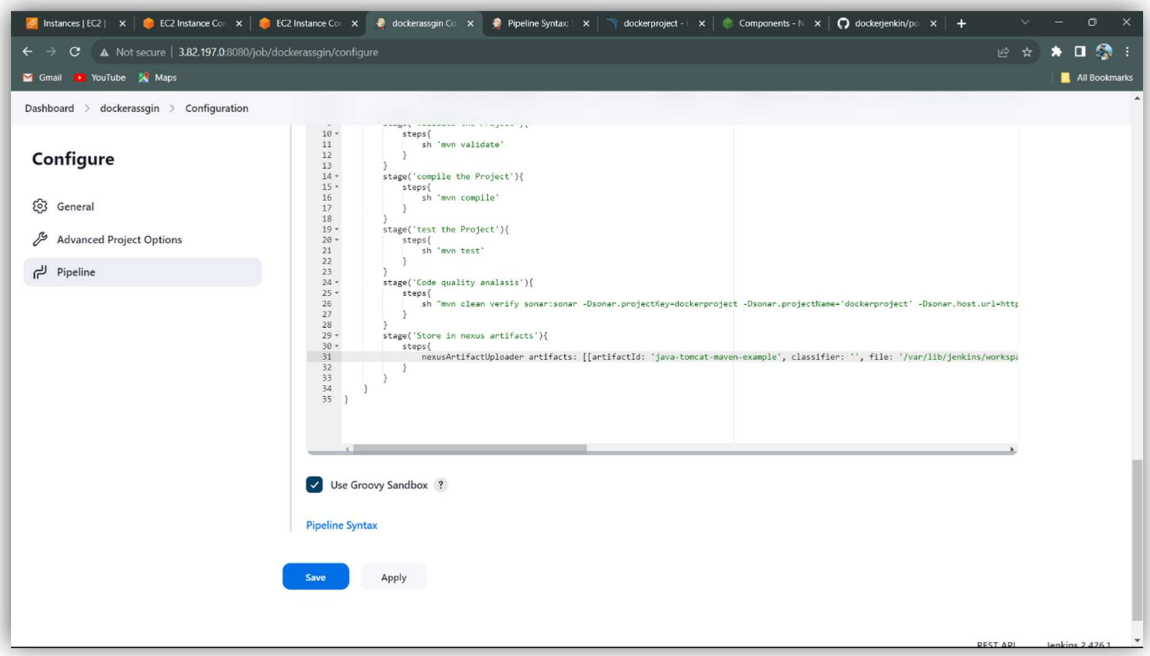
Type: war

Classifier:

File: /var/lib/jenkins/workspace/dockerassgin/target/java-tomcat-maven-example.war

File must not be empty

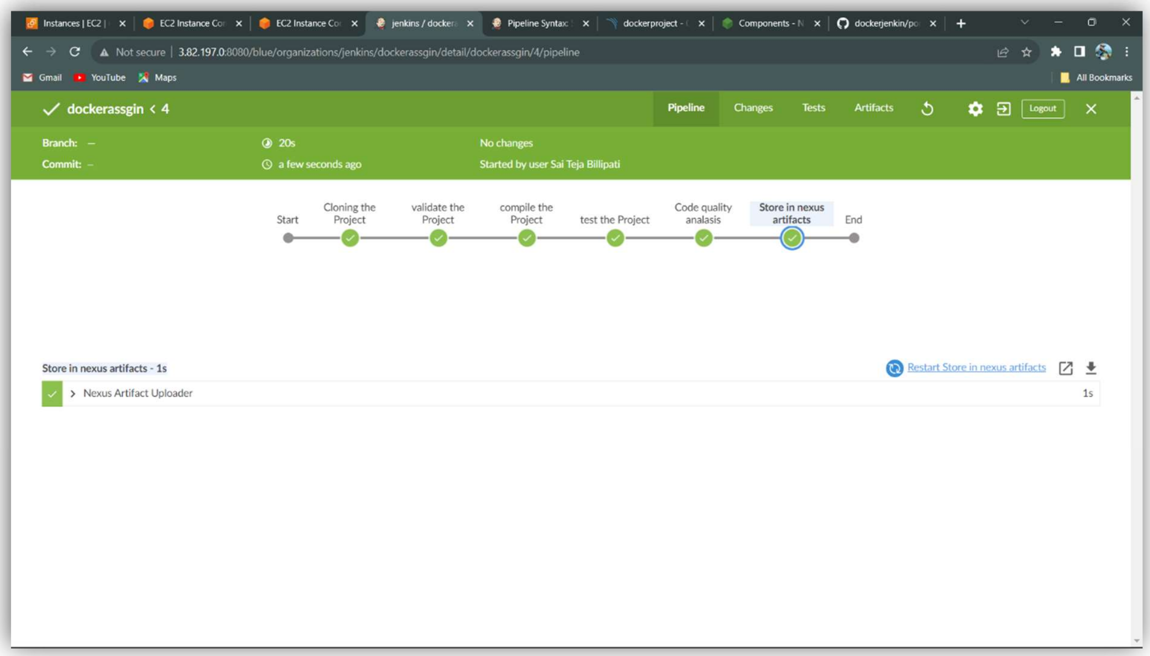
```
checkout scmGit(branches: [[name: '*/master']], extensions: [], userRemoteConfigs: [[url: 'https://github.com/Venna12/dockerjenkin.git']])
```



```
                steps{
                    sh 'mvn validate'
                }
            }
            stage('compile the Project'){
                steps{
                    sh 'mvn compile'
                }
            }
            stage('test the Project'){
                steps{
                    sh 'mvn test'
                }
            }
            stage('Code quality analasis'){
                steps{
                    sh "mvn clean verify sonar:sonar -Dsonar.projectKey=dockerproject -Dsonar.projectName='dockerproject' -Dsonar.host.url=http
                }
            }
            stage('Store in nexus artifacts'){
                steps{
                    nexusArtifactUploader artifacts: [[artifactId: 'java-tomcat-maven-example', classifier: '', file: '/var/lib/jenkins/workspa
                }
            }
        }
}
```

[✓] Use Groovy Sandbox

Pipeline Script for the Nexus Artifact in Jenkins.



```
3.82.197.0:8080/blue/organizations/jenkins/dockerassgin/detail/dockerassgin/4/pipeline
```

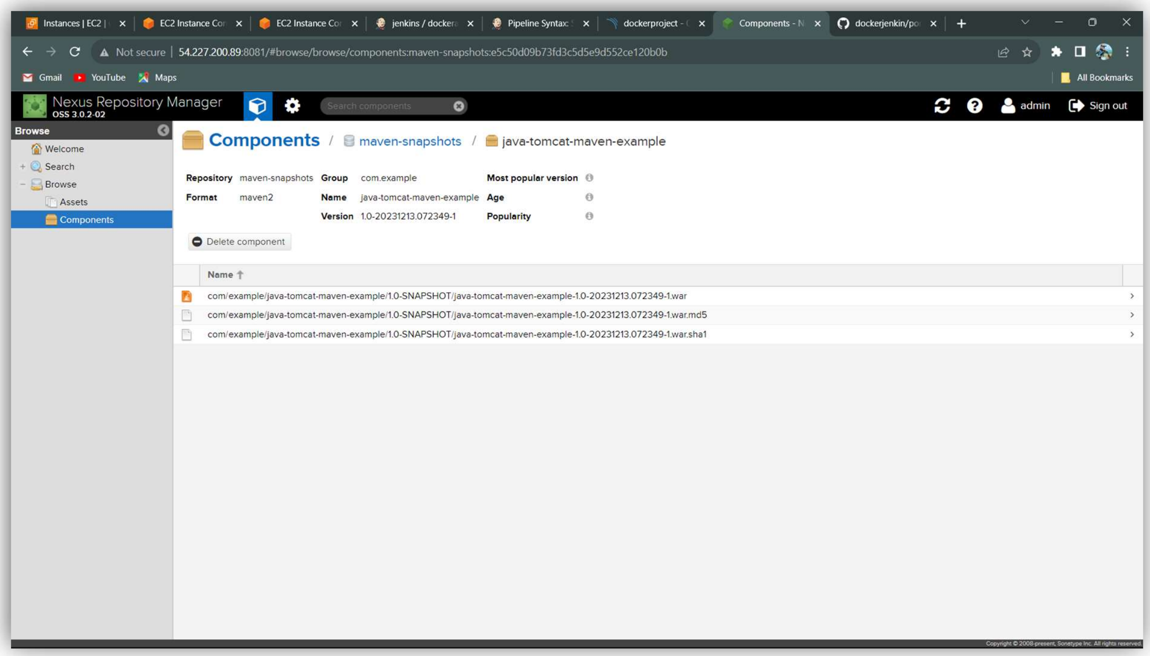
dockerassgin < 4

Branch: — Commit: — 20s a few seconds ago No changes Started by user Sai Teja Billipati

Store in nexus artifacts - 1s

Nexus Artifact Uploader 1s

The Pipeline is Successfully Build and The Artifact is stored in the Nexus Repository.



```
54.227.200.89:8081/#browse/browse/components:maven-snapshots:e5c50d09b73fd3c5d5e9d552ce120b0b
```

Components / maven-snapshots / java-tomcat-maven-example

Repository	maven-snapshots	Group	com.example	Most popular version	
Format	maven2	Name	java-tomcat-maven-example	Age	
		Version	1.0-20231213.072349-1	Popularity	

Name
com/example/java-tomcat-maven-example/1.0-SNAPSHOT/java-tomcat-maven-example-1.0-20231213.072349-1.war
com/example/java-tomcat-maven-example/1.0-SNAPSHOT/java-tomcat-maven-example-1.0-20231213.072349-1.war.md5
com/example/java-tomcat-maven-example/1.0-SNAPSHOT/java-tomcat-maven-example-1.0-20231213.072349-1.war.sha1

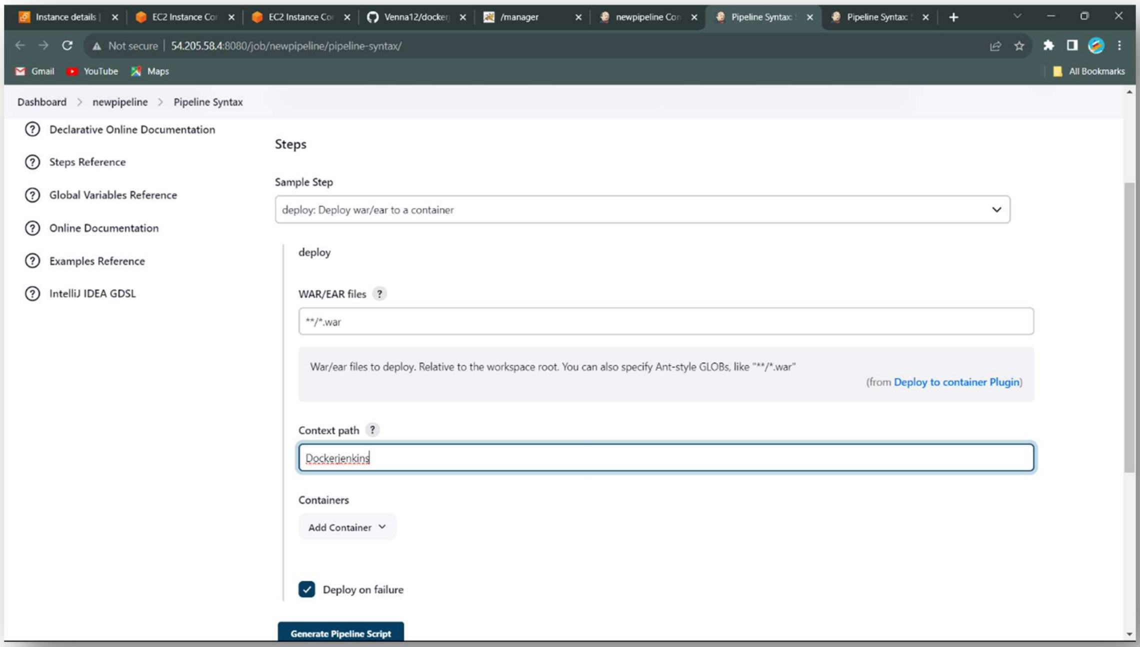
Artifact is stored in the maven-snapshots as java-tomcat-maven-example.

Deploying project into Apache Tomcat using Jenkins:

1. Open the Apache Tomcat Server using IP Address with Port no.
   a. IP address:Port no (8080).
2. Login to Tomcat using
   a. Username: Tomcat
   b. Password: s3cret
3. Install plugin for the deployment in the tomcat.

   ➢ Deploy Container

To get tomcat script, we use pipeline syntax



Not secure | 54.205.58.4:8080/job/newpipeline/pipeline-syntax/

Dashboard > newpipeline > Pipeline Syntax

- Declarative Online Documentation
- Steps Reference
- Global Variables Reference
- Online Documentation
- Examples Reference
- IntelliJ IDEA GDSL

Steps

Sample Step: deploy: Deploy war/ear to a container

deploy

WAR/EAR files: **/*.war

War/ear files to deploy. Relative to the workspace root. You can also specify Ant-style GLOBs, like "**/*.war"
(from Deploy to container Plugin)

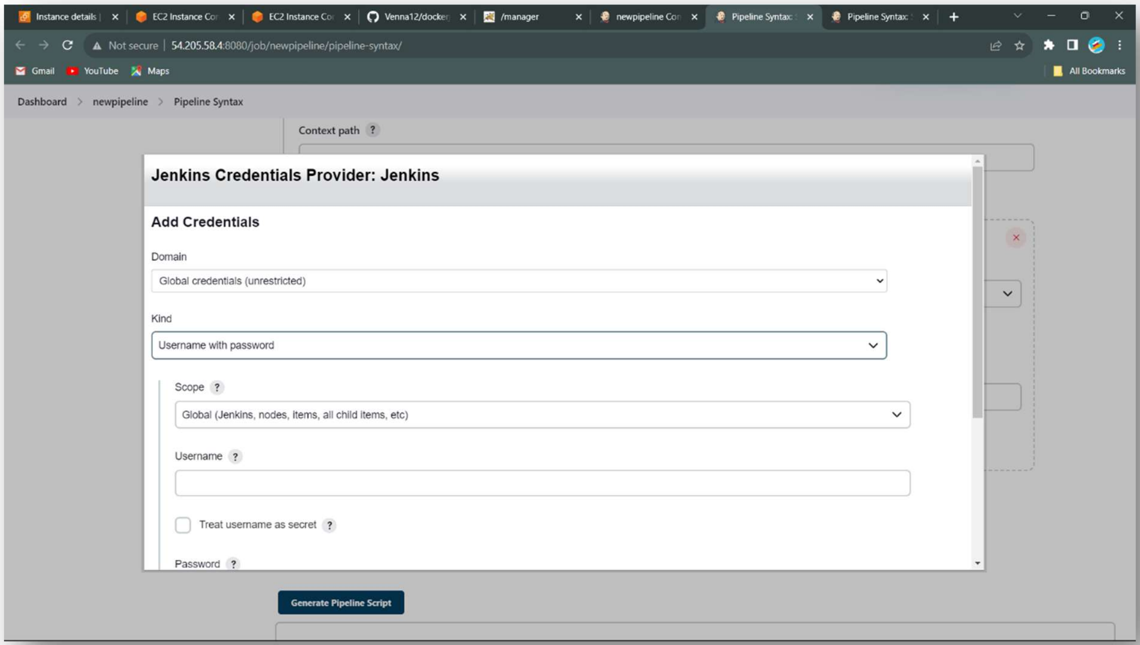
Context path: Dockerjenkins

Containers: Add Container

✓ Deploy on failure

Generate Pipeline Script

4. Create Jenkins id and password, it is add to the container.


Dashboard > newpipeline > Pipeline Syntax

Jenkins Credentials Provider: Jenkins

Add Credentials

Domain: Global credentials (unrestricted)

Kind: Username with password

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

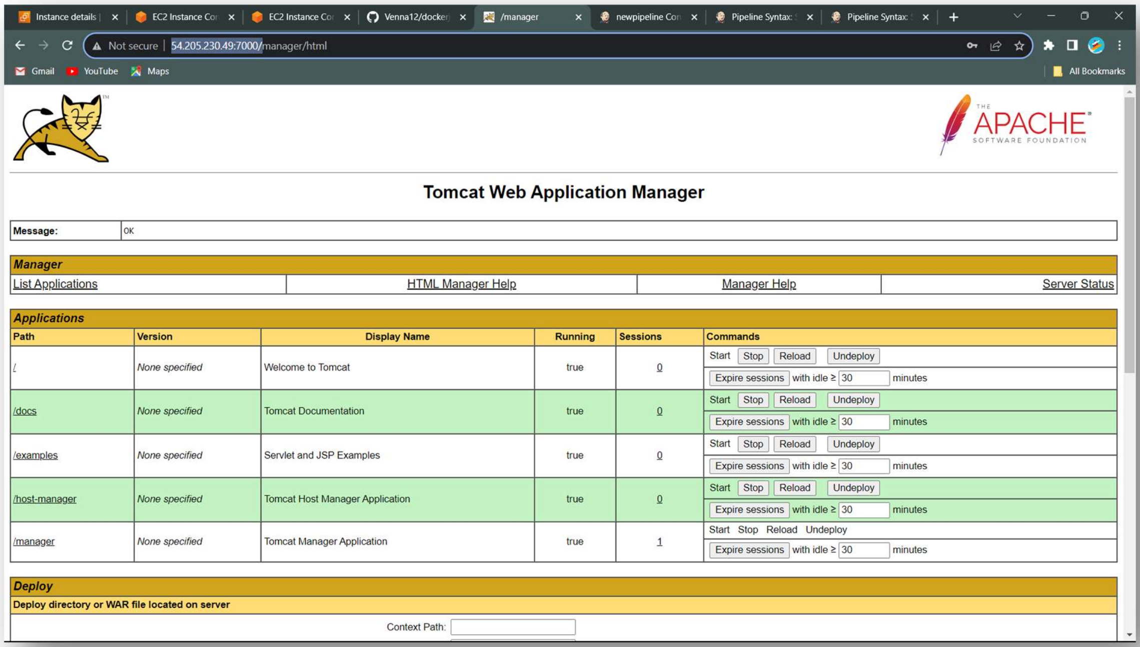
Username:

Treat username as secret

Password

Generate Pipeline Script

5. Open the Tomcat and host manager. It seems like that


54.205.230.49:7000/manager/html

Tomcat Web Application Manager

Message:	OK

Manager

List Applications	HTML Manager Help	Manager Help	Server Status

Applications

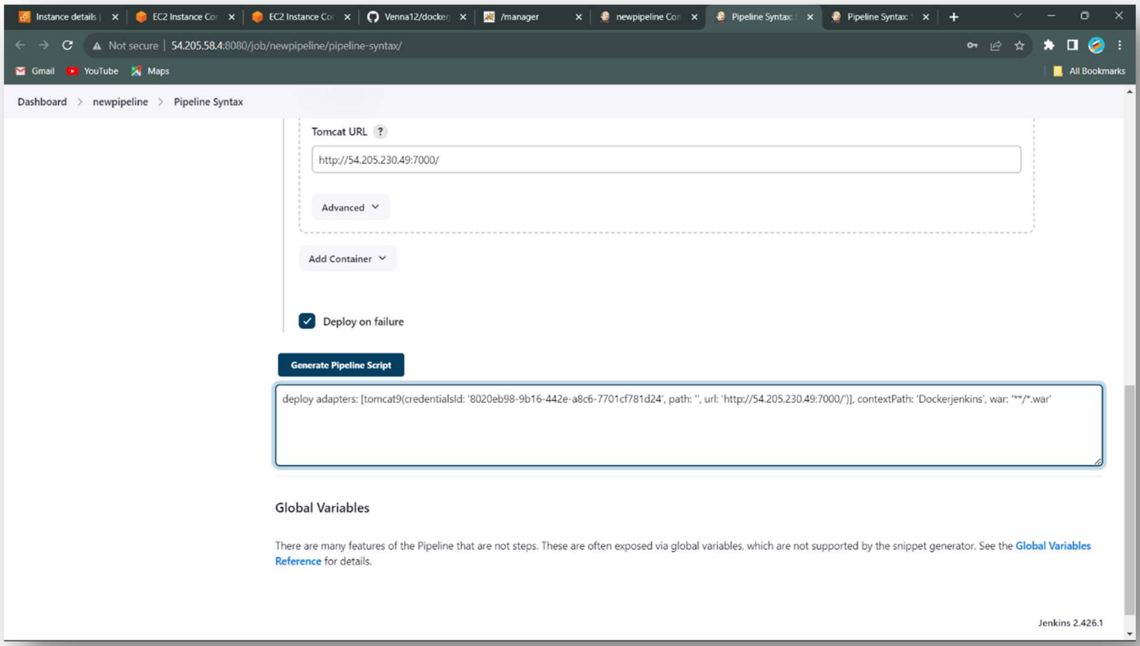
Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy; Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy; Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy; Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy; Expire sessions with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy; Expire sessions with idle ≥ 30 minutes

Deploy

Deploy directory or WAR file located on server

Context Path:

6. Copy the URL of the Tomcat with Ip address and port number. Copy the link address to the given box and we got script and copy the script the and add to the script to configure pipeline.


Dashboard > newpipeline > Pipeline Syntax

Tomcat URL: http://54.205.230.49:7000/

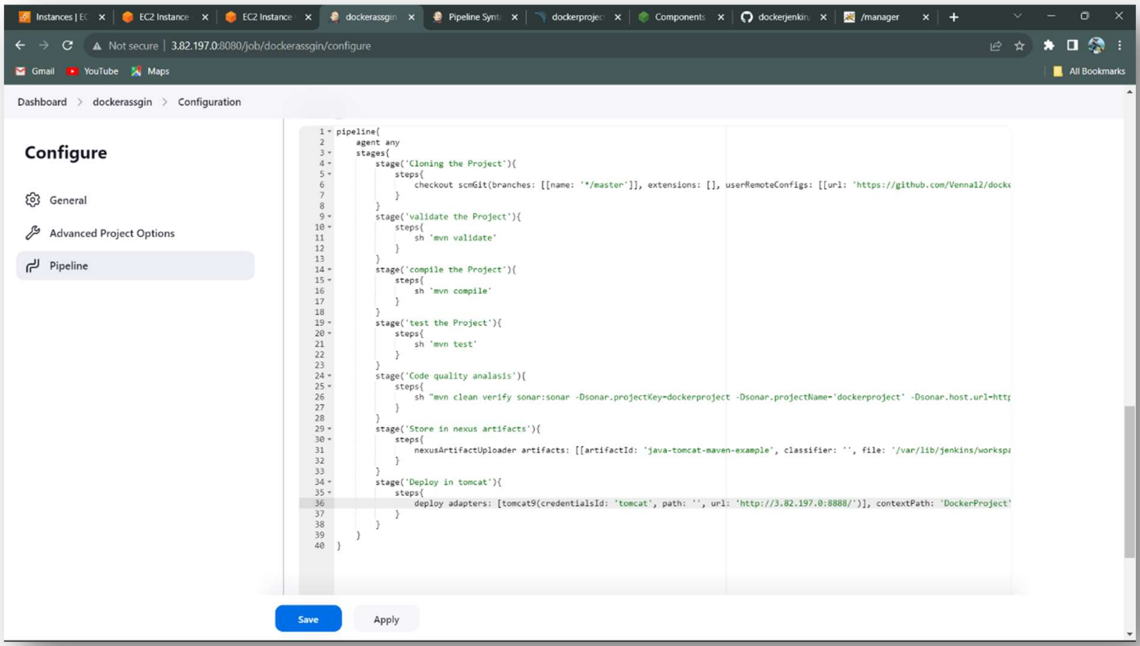
Deploy on failure ✓

```
deploy adapters: [tomcat9(credentialsId: '8020eb98-9b16-442e-a8c6-7701cf781d24', path: '', url: 'http://54.205.230.49:7000/')], contextPath: 'Dockerjenkins', war: '**/*.war'
```

Global Variables

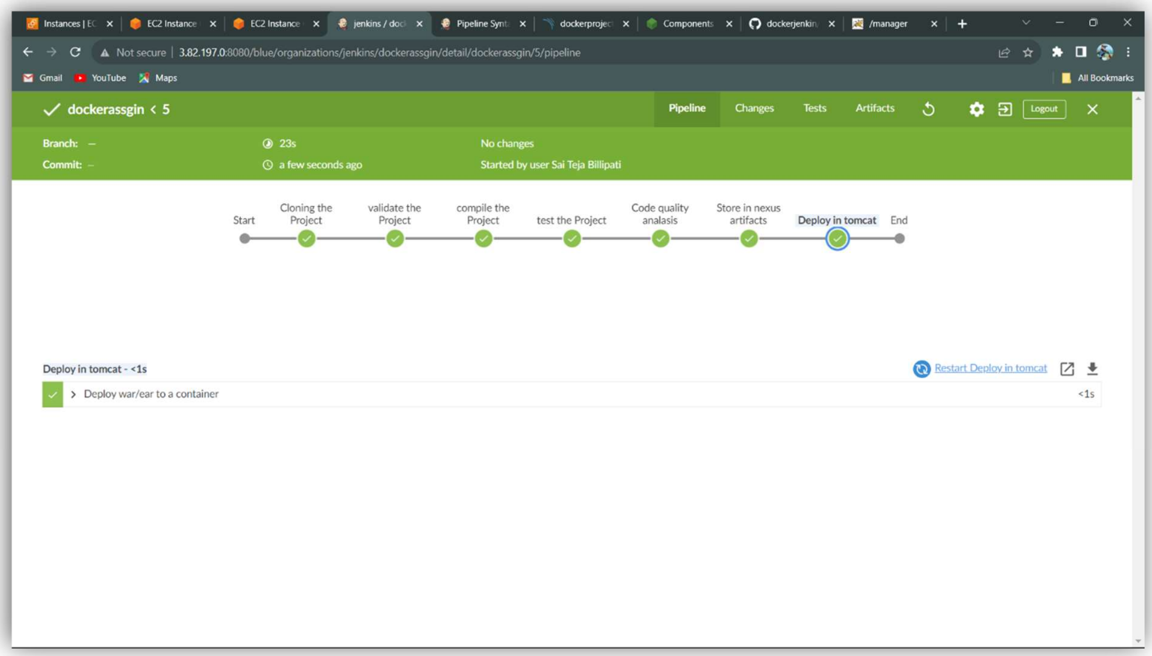
There are many features of the Pipeline that are not steps. These are often exposed via global variables, which are not supported by the snippet generator. See the Global Variables Reference for details.

7. Script add to pipeline script


Dashboard > dockerassgin > Configuration

```
pipeline{
    agent any
    stages{
        stage('Cloning the Project'){
            steps{
                checkout scmGit(branches: [[name: '*/master']], extensions: [], userRemoteConfigs: [[url: 'https://github.com/Venna12/docke
            }
        }
        stage('validate the Project'){
            steps{
                sh 'mvn validate'
            }
        }
        stage('compile the Project'){
            steps{
                sh 'mvn compile'
            }
        }
        stage('test the Project'){
            steps{
                sh 'mvn test'
            }
        }
        stage('Code quality analasis'){
            steps{
                sh "mvn clean verify sonar:sonar -Dsonar.projectKey=dockerproject -Dsonar.projectName='dockerproject' -Dsonar.host.url=http
            }
        }
        stage('Store in nexus artifacts'){
            steps{
                nexusArtifactUploader artifacts: [[artifactId: 'java-tomcat-maven-example', classifier: '', file: '/var/lib/jenkins/workspa
            }
        }
        stage('Deploy in tomcat'){
            steps{
                deploy adapters: [tomcat9(credentialsId: 'tomcat', path: '', url: 'http://3.82.197.0:8888/')], contextPath: 'DockerProject'
            }
        }
    }
}
```

8. Build the Project.



Not secure | 3.82.197.0:8080/blue/organizations/jenkins/dockerassgin/detail/dockerassgin/5/pipeline

✓ dockerassgin < 5

Branch: –
Commit: –
23s
a few seconds ago
No changes
Started by user Sai Teja Billipati

Start — Cloning the Project — validate the Project — compile the Project — test the Project — Code quality analasis — Store in nexus artifacts — Deploy in tomcat — End

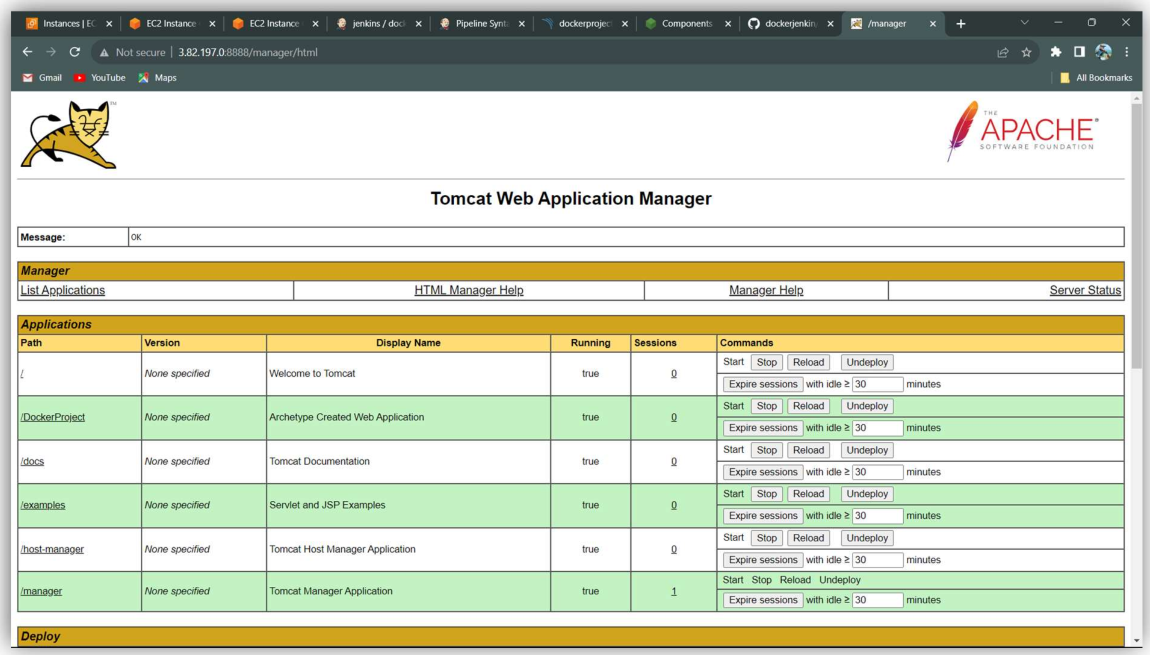
Deploy in tomcat - <1s

Restart Deploy in tomcat

✓ > Deploy war/ear to a container <1s

Project is Successfully Deployed in the tomcat.

9. Open Tomcat server, we can see the DockerProject in the Tomcat Dashboad.



Not secure | 3.82.197.0:8888/manager/html

Tomcat Web Application Manager

Message:	OK

Manager

List Applications	HTML Manager Help	Manager Help	Server Status

Applications

Path	Version	Display Name	Running	Sessions	Commands
/	None specified	Welcome to Tomcat	true	0	Start Stop Reload Undeploy / Expire sessions with idle ≥ 30 minutes
/DockerProject	None specified	Archetype Created Web Application	true	0	Start Stop Reload Undeploy / Expire sessions with idle ≥ 30 minutes
/docs	None specified	Tomcat Documentation	true	0	Start Stop Reload Undeploy / Expire sessions with idle ≥ 30 minutes
/examples	None specified	Servlet and JSP Examples	true	0	Start Stop Reload Undeploy / Expire sessions with idle ≥ 30 minutes
/host-manager	None specified	Tomcat Host Manager Application	true	0	Start Stop Reload Undeploy / Expire sessions with idle ≥ 30 minutes
/manager	None specified	Tomcat Manager Application	true	1	Start Stop Reload Undeploy / Expire sessions with idle ≥ 30 minutes

Deploy

-- END --