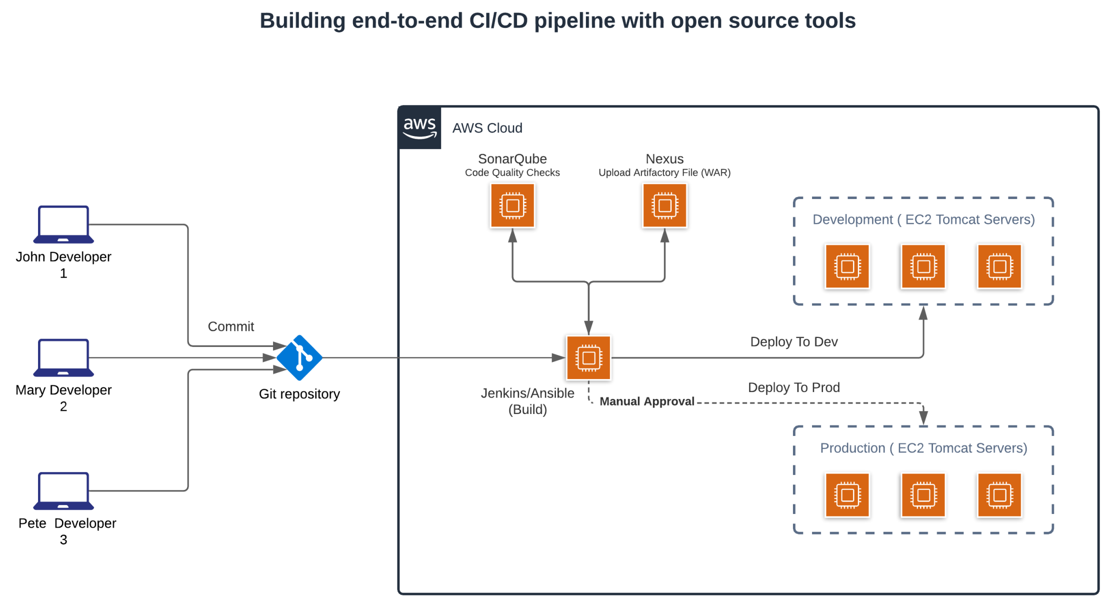
Building an end-to-end CI/CD pipeline with open source tools

Services and tools

In this section, we discuss the various AWS services and third-party tools used in this solution.

Jenkins	→ Jenkins is an open source automation server which enables developers around the world to reliably build, test, and deploy their software
SonarQube	→ Catches bugs and vulnerabilities in your app, with thousands of automated Static Code Analysis rules.
Nexus	→ Manage Binaries and build artifacts across your software supply chain
Git	→ **Git** is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency
AWS EC2	→ Amazon Elastic cloud-computing platform, Amazon Web Services, that allows users to rent virtual computers on which to run their own computer applications

Pipeline Architecture:



Main Steps:

1. When a user commits a code to a Github repository, Jenkins job will get triggered
2. Jenkins pipeline consists of SIX stages:
    a. Building the artifact out of Java Code
    b. SonarQube scans the code for any vulnerabilities
    c. Pushes the artifact to Sonatype Nexus (Artifact Repository)
    d. Deploys to DEV servers
    e. Manual Approval
    f. Deploys to PROD servers

Prerequisites:

Before getting started, make sure you have the following prerequisites:

→ Launch 5 EC2 Instances with the EC2 type t2.medium
→ 1 : Jenkins/Ansible
→ 2 : SonarQube
→ 3 : Nexus
→ 4 : Test DEV server to deploy
→ 5 : Test PROD server to deploy

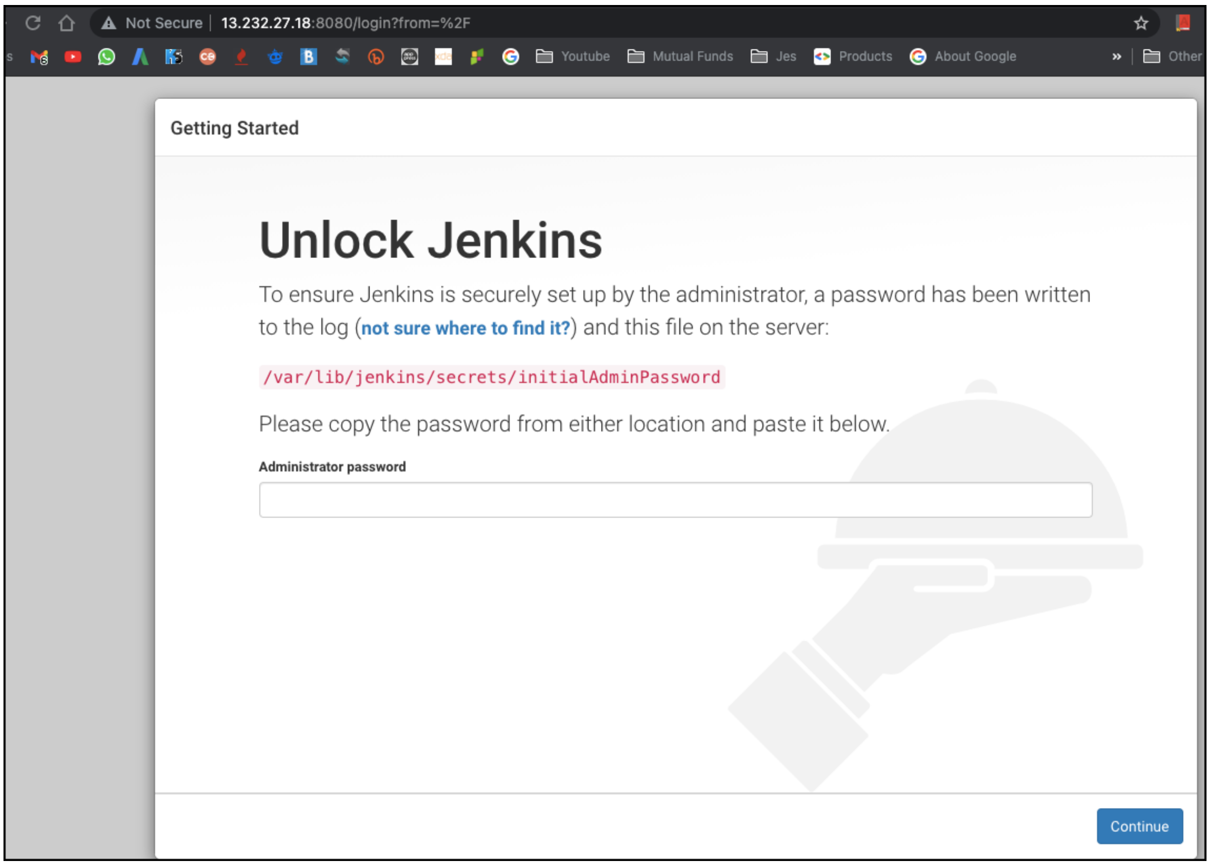
Section-1: Installation of Jenkins (Instance 1)

1.Create an Amazon Linux EC2 instance run below commands one you ssh

```
#!/bin/bash
cd /home/ec2-user

sudo yum install java-1.8* -y

sudo yum install wget –y

sudo yum install git –y

sudo wget -O /etc/yum.repos.d/jenkins.repo https://pkg.jenkins.io/redhat-stable/jenkins.repo

sudo rpm --import https://pkg.jenkins.io/redhat-stable/jenkins.io.key

sudo amazon-linux-extras install epel -y

sudo yum update -y

sudo yum install jenkins java-1.8.0-openjdk-devel

# Start jenkins service

sudo systemctl start jenkins

# Setup Jenkins to start at boot

sudo systemctl enable jenkins

# Setup Jenkins to check the status

sudo systemctl status jenkins
```

2. Once we are good with Jenkins Installation; we can access it using the url:
   http://public-ip:8080

3.Jenkins webpage should look like the one below



Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

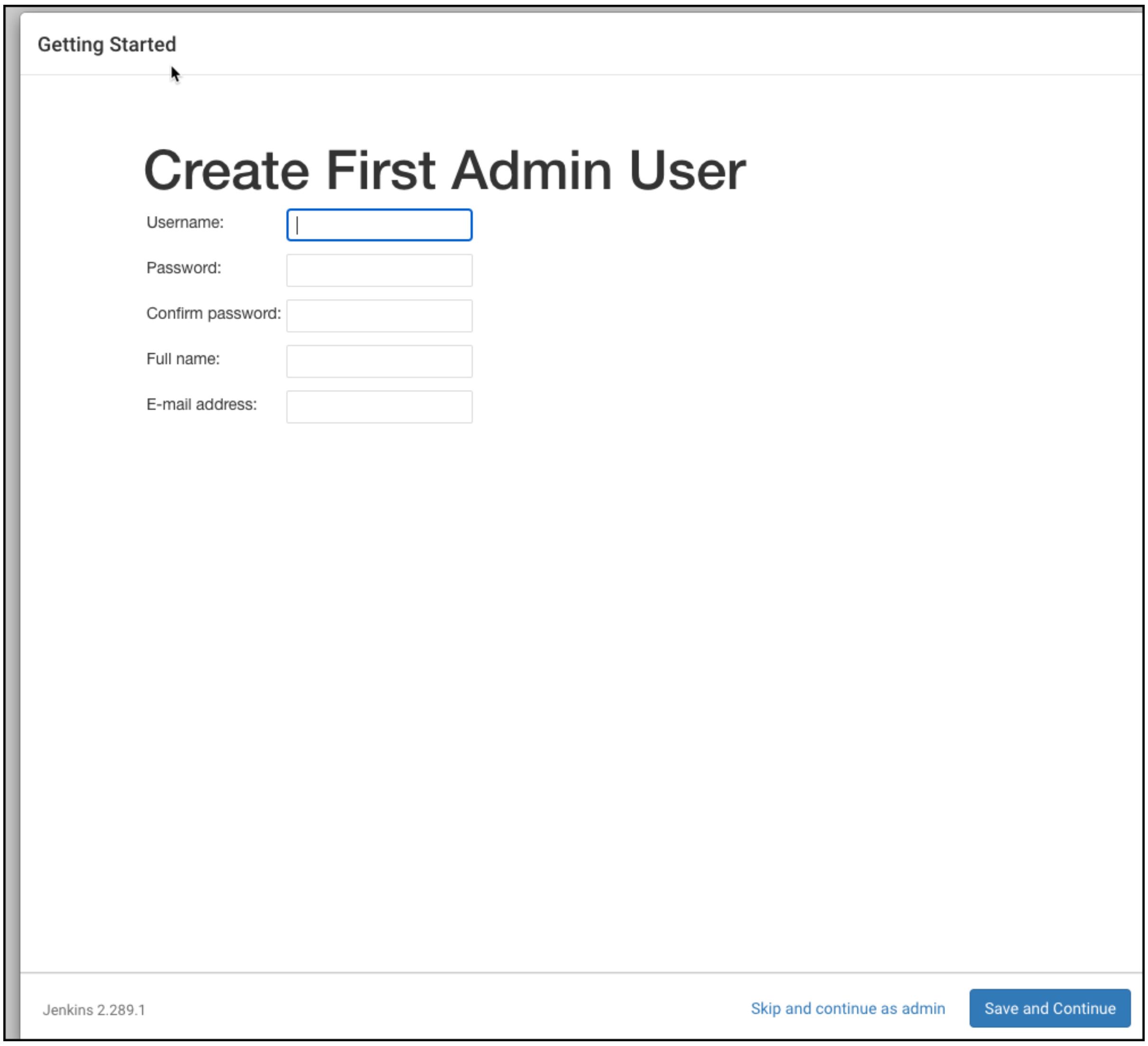
```
/var/lib/jenkins/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

Administrator password

4. Run the command sudo cat /var/lib/jenkins/secrets/initialAdminPassword
on the Jenkins EC2 instance terminal to get the admin password, enter it on the prompt above and continue.

5.Choose Install suggested plugins in the next step
6.



Getting Started

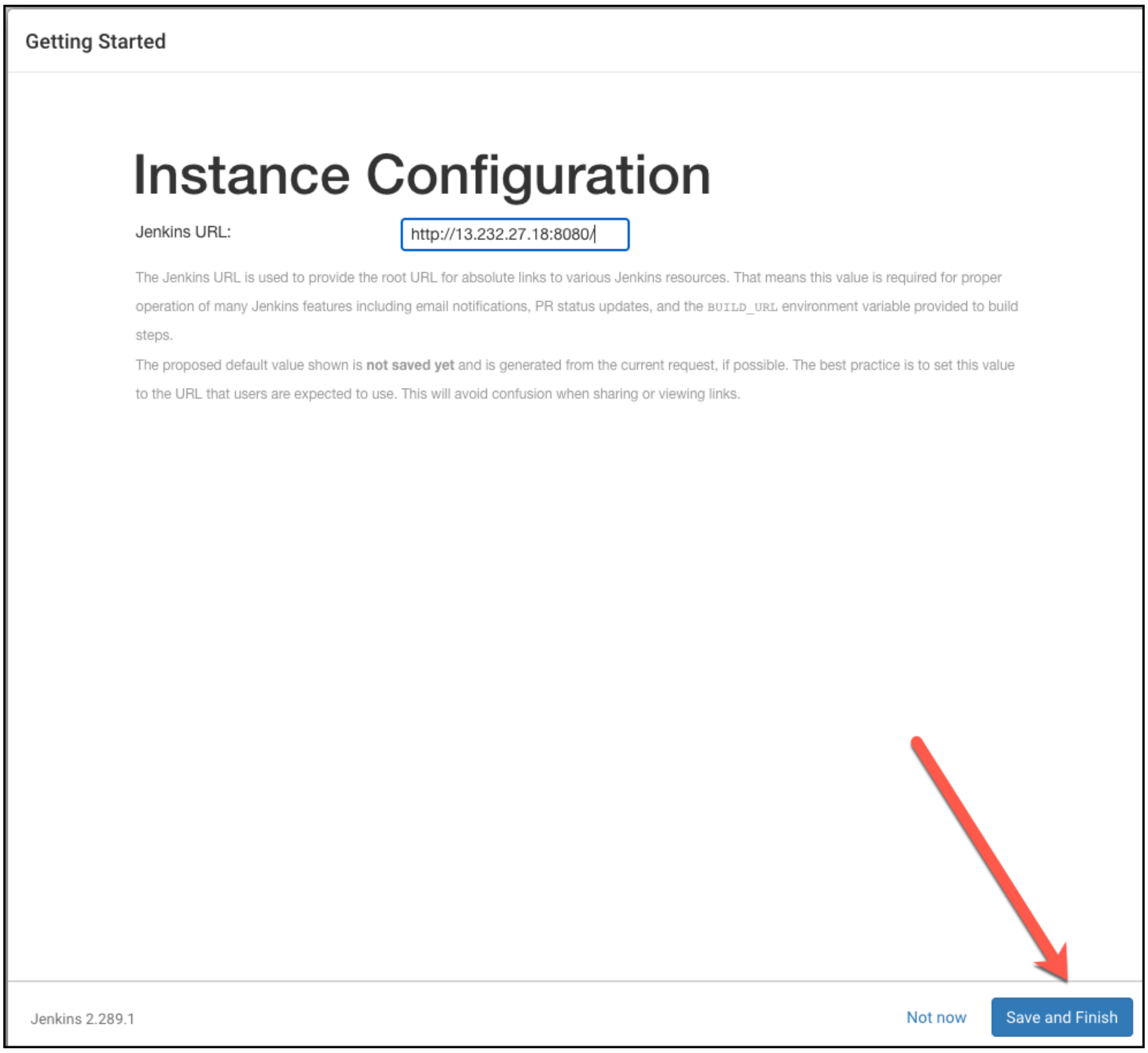
Create First Admin User

Username:	
Password:	
Confirm password:	
Full name:	
E-mail address:	

Jenkins 2.289.1

Enter all details, save and continue.

7.



Instance Configuration

Jenkins URL: http://13.232.27.18:8080/

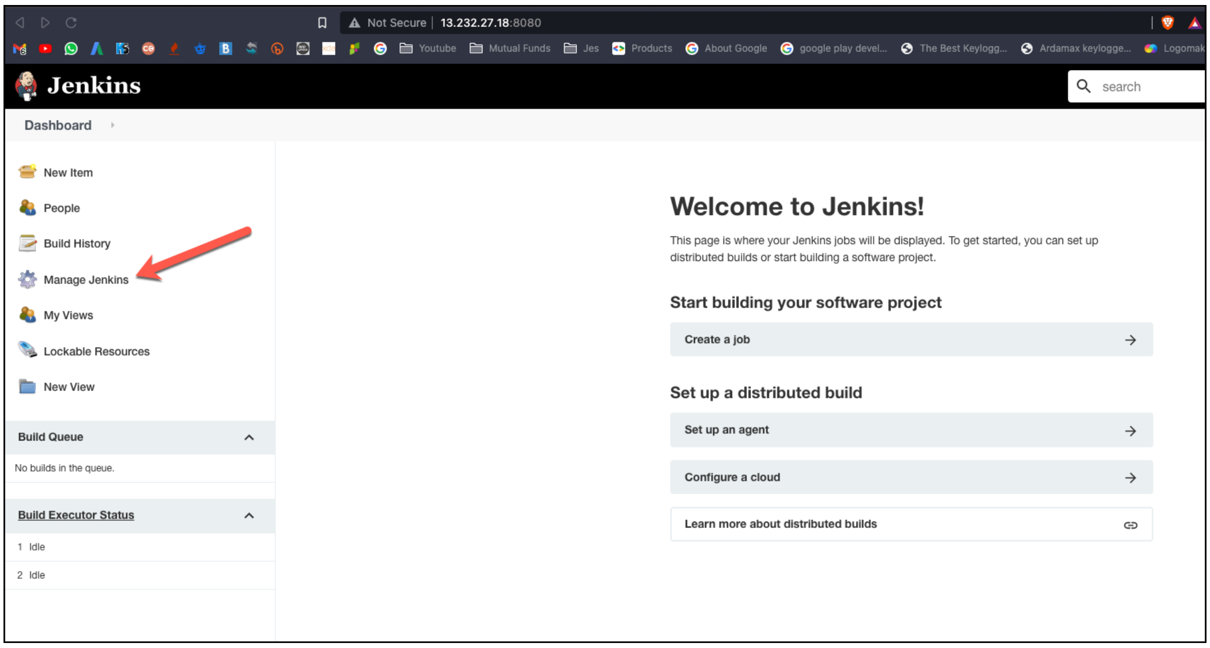
The Jenkins URL is used to provide the root URL for absolute links to various Jenkins resources. That means this value is required for proper operation of many Jenkins features including email notifications, PR status updates, and the `BUILD_URL` environment variable provided to build steps.

The proposed default value shown is **not saved yet** and is generated from the current request, if possible. The best practice is to set this value to the URL that users are expected to use. This will avoid confusion when sharing or viewing links.

8. You can always access Jenkins using url below
   http://public-ip:8080

9. Then click start using Jenkins

10. Navigate to manage jenkins


Not Secure | 13.232.27.18:8080

Jenkins

Dashboard

- New Item
- People
- Build History
- Manage Jenkins
- My Views
- Lockable Resources
- New View

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Welcome to Jenkins!

This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job

Set up a distributed build

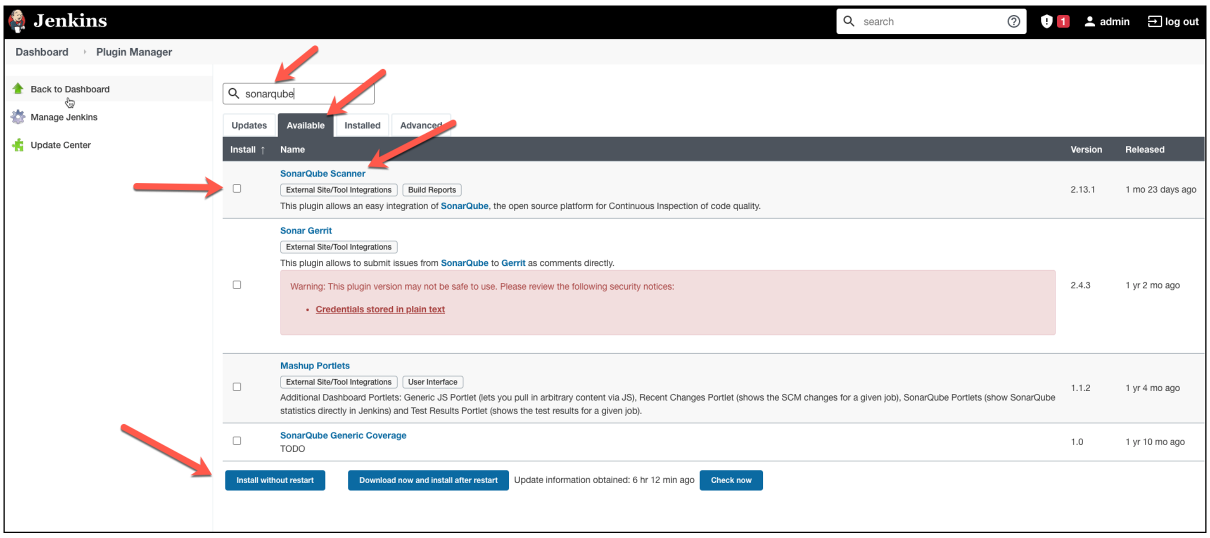
Set up an agent

Configure a cloud

Learn more about distributed builds

11. Please select Manage Plugins

12. That looks like below and search for “sonarqube” and Maven integration


Jenkins

Dashboard › Plugin Manager

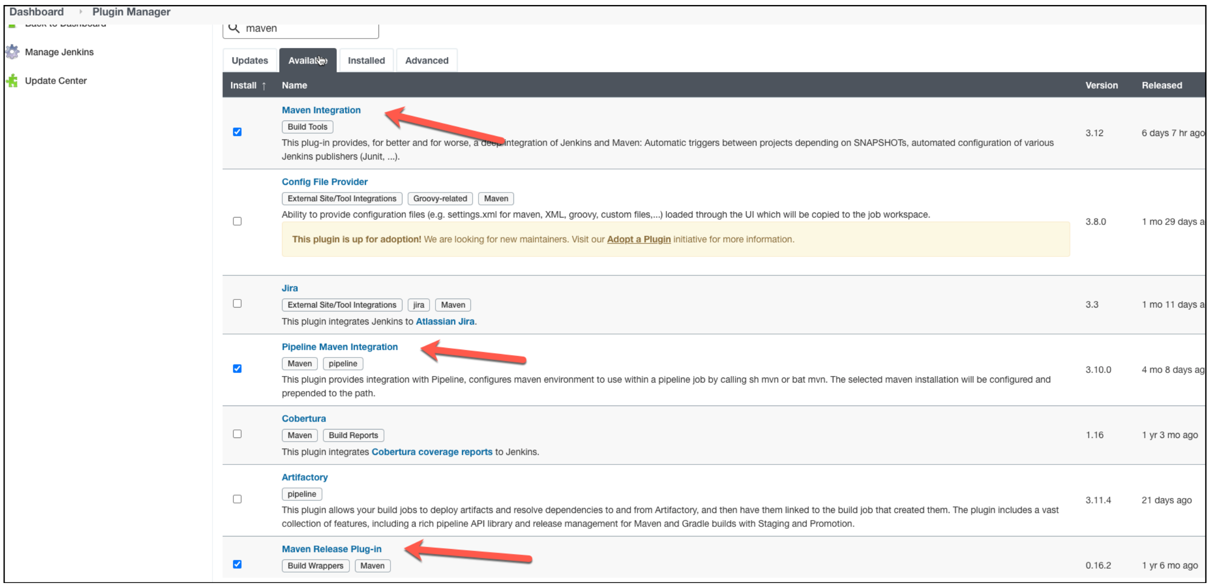
- Back to Dashboard
- Manage Jenkins
- Update Center

sonarqube

Updates | Available | Installed | Advanced

Install ↑	Name	Version	Released
☐	SonarQube Scanner External Site/Tool Integrations · Build Reports This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.	2.13.1	1 mo 23 days ago
☐	Sonar Gerrit External Site/Tool Integrations This plugin allows to submit issues from SonarQube to Gerrit as comments directly. Warning: This plugin version may not be safe to use. Please review the following security notices: • Credentials stored in plain text	2.4.3	1 yr 2 mo ago
☐	Mashup Portlets External Site/Tool Integrations · User Interface Additional Dashboard Portlets: Generic JS Portlet (lets you pull in arbitrary content via JS), Recent Changes Portlet (shows the SCM changes for a given job), SonarQube Portlets (show SonarQube statistics directly in Jenkins) and Test Results Portlet (shows the test results for a given job).	1.1.2	1 yr 4 mo ago
☐	SonarQube Generic Coverage TODO	1.0	1 yr 10 mo ago

Install without restart | Download now and install after restart | Update information obtained: 6 hr 12 min ago | Check now


Dashboard › Plugin Manager

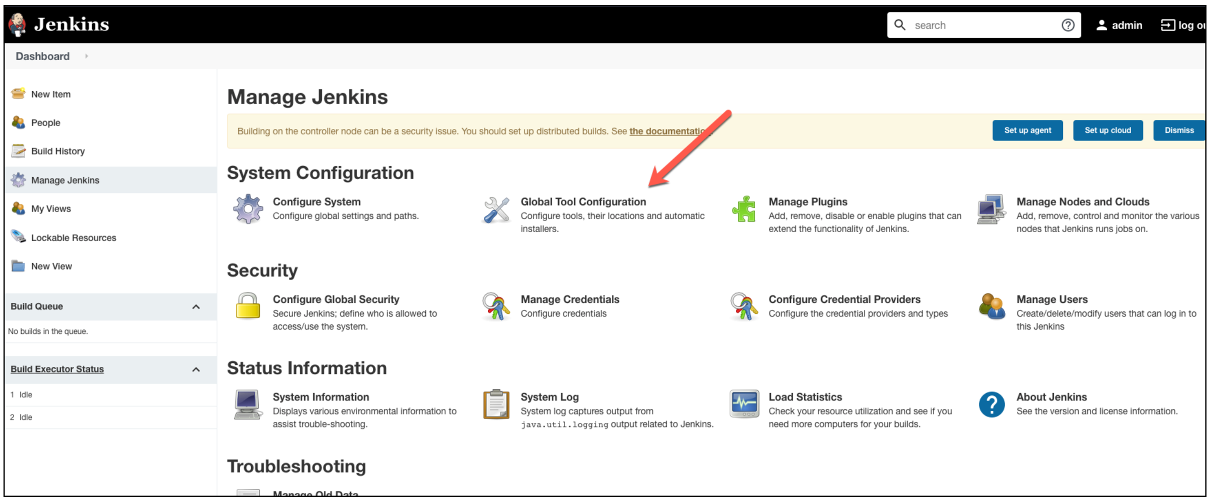
Manage Jenkins
Update Center

maven

Updates | Available | Installed | Advanced

Install ↓	Name	Version	Released
✓	Maven Integration Build Tools This plug-in provides, for better and for worse, a deep integration of Jenkins and Maven: Automatic triggers between projects depending on SNAPSHOTs, automated configuration of various Jenkins publishers (Junit, ...).	3.12	6 days 7 hr ago
☐	Config File Provider External Site/Tool Integrations · Groovy-related · Maven Ability to provide configuration files (e.g. settings.xml for maven, XML, groovy, custom files,...) loaded through the UI which will be copied to the job workspace. **This plugin is up for adoption!** We are looking for new maintainers. Visit our **Adopt a Plugin** initiative for more information.	3.8.0	1 mo 29 days a
☐	Jira External Site/Tool Integrations · jira · Maven This plugin integrates Jenkins to Atlassian Jira.	3.3	1 mo 11 days a
✓	Pipeline Maven Integration Maven · pipeline This plugin provides integration with Pipeline, configures maven environment to use within a pipeline job by calling sh mvn or bat mvn. The selected maven installation will be configured and prepended to the path.	3.10.0	4 mo 8 days ag
☐	Cobertura Maven · Build Reports This plugin integrates Cobertura coverage reports to Jenkins.	1.16	1 yr 3 mo ago
☐	Artifactory pipeline This plugin allows your build jobs to deploy artifacts and resolve dependencies to and from Artifactory, and then have them linked to the build job that created them. The plugin includes a vast collection of features, including a rich pipeline API library and release management for Maven and Gradle builds with Staging and Promotion.	3.11.4	21 days ago
✓	Maven Release Plug-in Build Wrappers · Maven	0.16.2	1 yr 6 mo ago

13. Navigate to Global tool configuration


Jenkins   search   admin   log ou

Dashboard ›

New Item
People
Build History
Manage Jenkins
My Views
Lockable Resources
New View

Build Queue
No builds in the queue.

Build Executor Status
1 Idle
2 Idle

Manage Jenkins

Building on the controller node can be a security issue. You should set up distributed builds. See the documentation.   Set up agent   Set up cloud   Dismiss

System Configuration

Configure System — Configure global settings and paths.
Global Tool Configuration — Configure tools, their locations and automatic installers.
Manage Plugins — Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
Manage Nodes and Clouds — Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

Security

Configure Global Security — Secure Jenkins; define who is allowed to access/use the system.
Manage Credentials — Configure credentials
Configure Credential Providers — Configure the credential providers and types
Manage Users — Create/delete/modify users that can log in to this Jenkins

Status Information

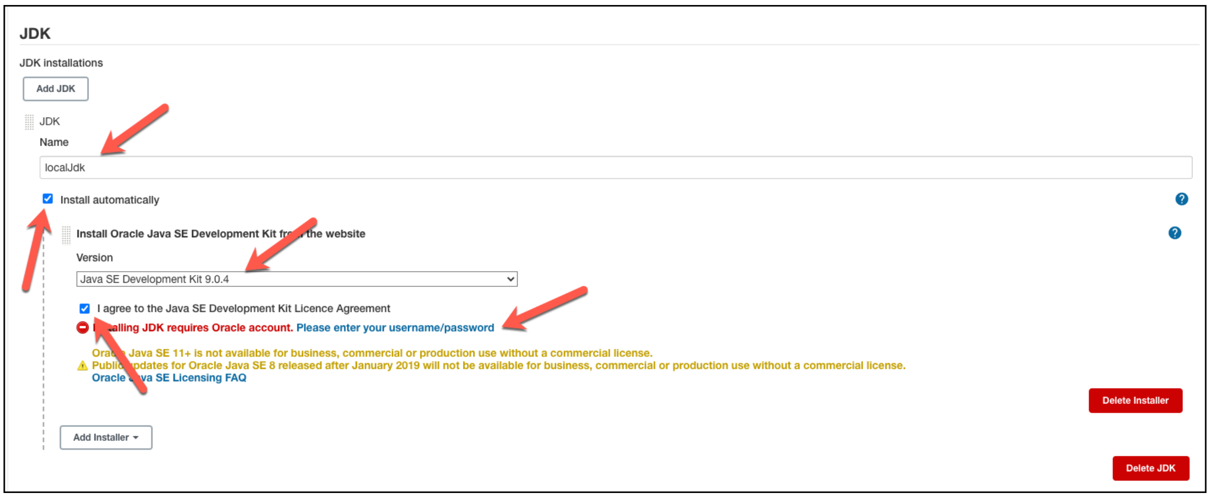
System Information — Displays various environmental information to assist trouble-shooting.
System Log — System log captures output from `java.util.logging` output related to Jenkins.
Load Statistics — Check your resource utilization and see if you need more computers for your builds.
About Jenkins — See the version and license information.

Troubleshooting

Manage Old Data

12. We have to configure JDK, Git, Sonarqube and Maven

a. Configure JDK.
b. Enter Name “localJdk
c. Select Install Automatically check button


JDK

JDK installations

Add JDK

JDK

Name

localJdk

☑ Install automatically

Install Oracle Java SE Development Kit from the website

Version

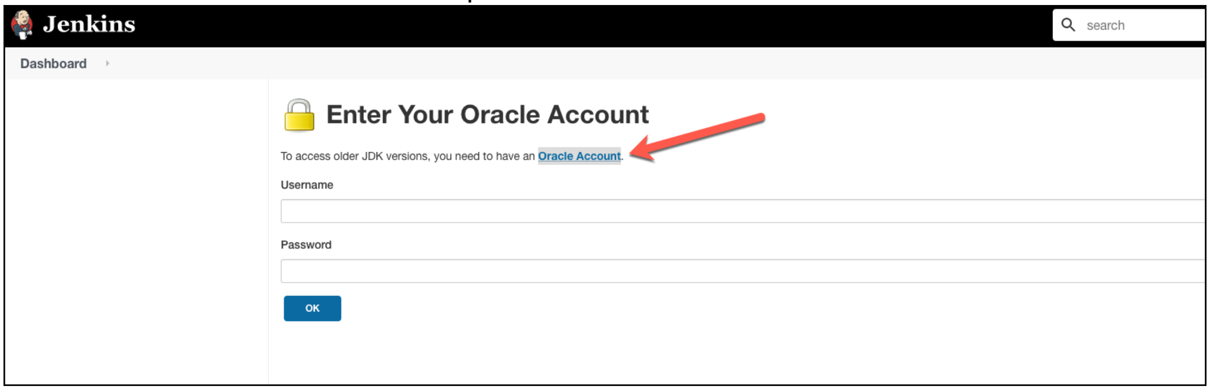
Java SE Development Kit 9.0.4

☑ I agree to the Java SE Development Kit Licence Agreement

Installing JDK requires Oracle account. Please enter your username/password

Oracle Java SE 11+ is not available for business, commercial or production use without a commercial license.
Public updates for Oracle Java SE 8 released after January 2019 will not be available for business, commercial or production use without a commercial license.
Oracle Java SE Licensing FAQ

d. Pelase select “Please enter your username/password”; then create an oracle account. Then enter username and password. OK



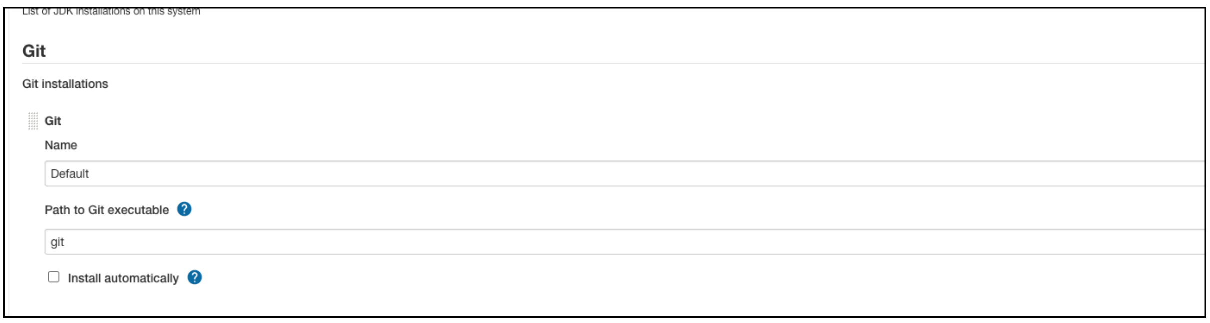
Enter Your Oracle Account

To access older JDK versions, you need to have an Oracle Account.

Username

Password

e. Enter details of Git


List of JDK installations on this system

Git

Git installations

Git

Name

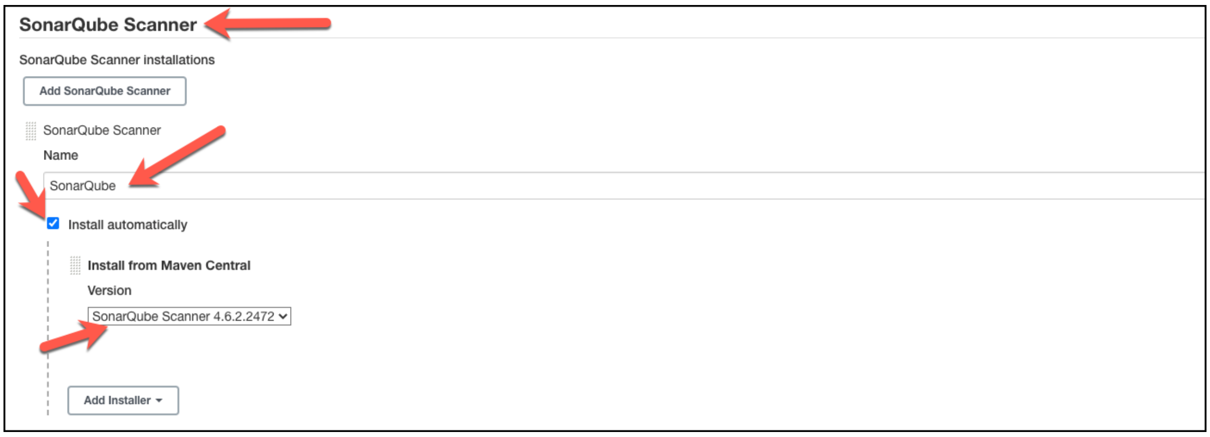
Default

Path to Git executable

git

☐ Install automatically

f. Select Sonarqube scanner


SonarQube Scanner

SonarQube Scanner installations

SonarQube Scanner

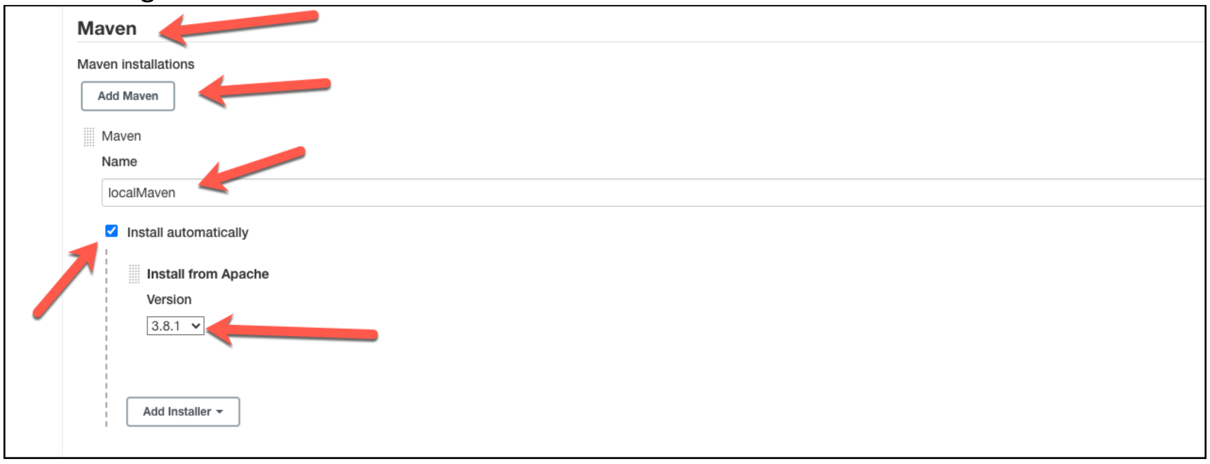
Name: SonarQube

Install automatically [✓]

Install from Maven Central

Version: SonarQube Scanner 4.6.2.2472

g. Then configure Maven


Maven

Maven installations

Maven

Name: localMaven

Install automatically [✓]

Install from Apache

Version: 3.8.1

h. Apply once you configure as shown above.

Section-2: Setup Sonarqube (Instance 2)

1. Create an Amazon Linux EC2 instance
2. Install Java

```
sudo yum install java-1.8.0 -y
```

3. Check java installation with the command below
   a. `java –version`

4. First, add repo using below command

```
sudo wget -O /etc/yum.repos.d/sonar.repo http://downloads.sourceforge.net/project/sonar-pkg/rpm/sonar.repo
```

5. Install SonarQube by running:

```
sudo yum install sonar -y
```

6. Start SonarQube

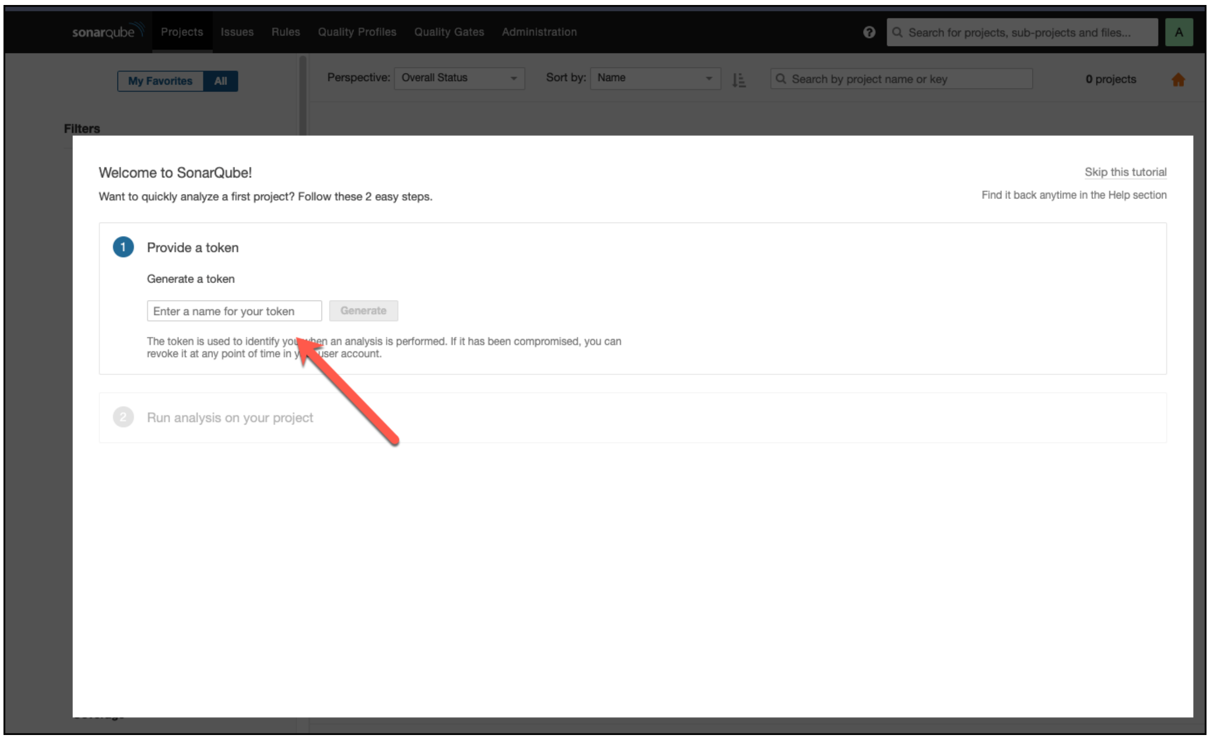
   sudo service sonar start

7. Check status SonarQube

   sudo service sonar status

8. Access SonarQube from browser
   http://<ipaddress>:9000
9. Default user name and password is “admin”

10. Once you login you will find this screen. Please enter a name and generate token


sonarqube
Projects
Issues
Rules
Quality Profiles
Quality Gates
Administration
Search for projects, sub-projects and files...
My Favorites
All
Perspective: Overall Status
Sort by: Name
Search by project name or key
0 projects
Filters
Skip this tutorial
Find it back anytime in the Help section
Enter a name for your token
Generate

Welcome to SonarQube!

Want to quickly analyze a first project? Follow these 2 easy steps.

1 Provide a token

Generate a token

The token is used to identify you when an analysis is performed. If it has been compromised, you can revoke it at any point of time in your user account.

2 Run analysis on your project

11. We have to copy the above code and update it in JenkinsFile

Section-3: Setup Nexus (Instance 3)

1. Login to your Linux server and update the yum packages. Also install required utilities.

   sudo yum update -y

   sudo yum install wget -y

2. Install OpenJDK 1.8

   ```
   sudo yum install java-1.8.0-openjdk.x86_64 -y
   ```

3. Create a directory named app and cd into the directory.

   ```
   sudo mkdir /app && cd /app
   ```

4. Download the latest nexus. You can get the latest download links for nexus from here.

   ```
   sudo wget -O nexus.tar.gz https://download.sonatype.com/nexus/3/latest-
   unix.tar.gz
   ```

5. Untar the downloaded file.

   ```
   sudo tar -xvf nexus.tar.gz
   ```

6. Rename the untared file to nexus

   ```
   sudo mv nexus-3.32.0-03/ nexus
   ```

7. As a good security practice, it is not advised to run nexus service with root privileges. So create a new user named nexus to run the nexus service.

   ```
   sudo adduser nexus
   ```

8. Change the ownership of nexus files and nexus data directory to nexus user.

   ```
   sudo chown -R nexus:nexus /app/nexus
   ```

   ```
   sudo chown -R nexus:nexus /app/sonatype-work
   ```

9. Open /app/nexus/bin/nexus.rc file

```
sudo vi  /app/nexus/bin/nexus.rc
```

10. Uncomment run_as_user parameter and set it as following.

```
run_as_user="nexus"
```

11. Create a nexus systemd unit file.

```
sudo vi /etc/systemd/system/nexus.service
```

12.
```
[Unit]
Description=nexus service
After=network.target

[Service]
Type=forking
LimitNOFILE=65536
User=nexus
Group=nexus
ExecStart=/app/nexus/bin/nexus start
ExecStop=/app/nexus/bin/nexus stop
User=nexus
Restart=on-abort

[Install]
WantedBy=multi-user.target
```

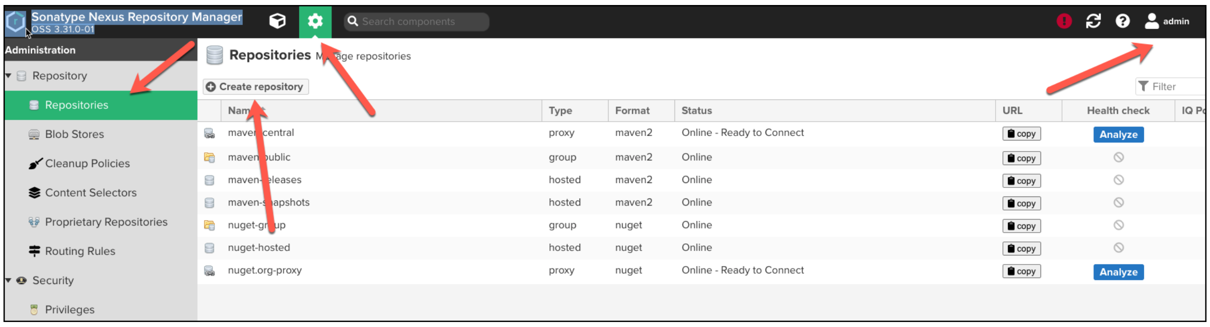
13. Execute the following command to add nexus service to boot.

```
sudo chkconfig nexus on
```

14. To start the Nexus service, use the following command.

```
sudo systemctl start nexus
```

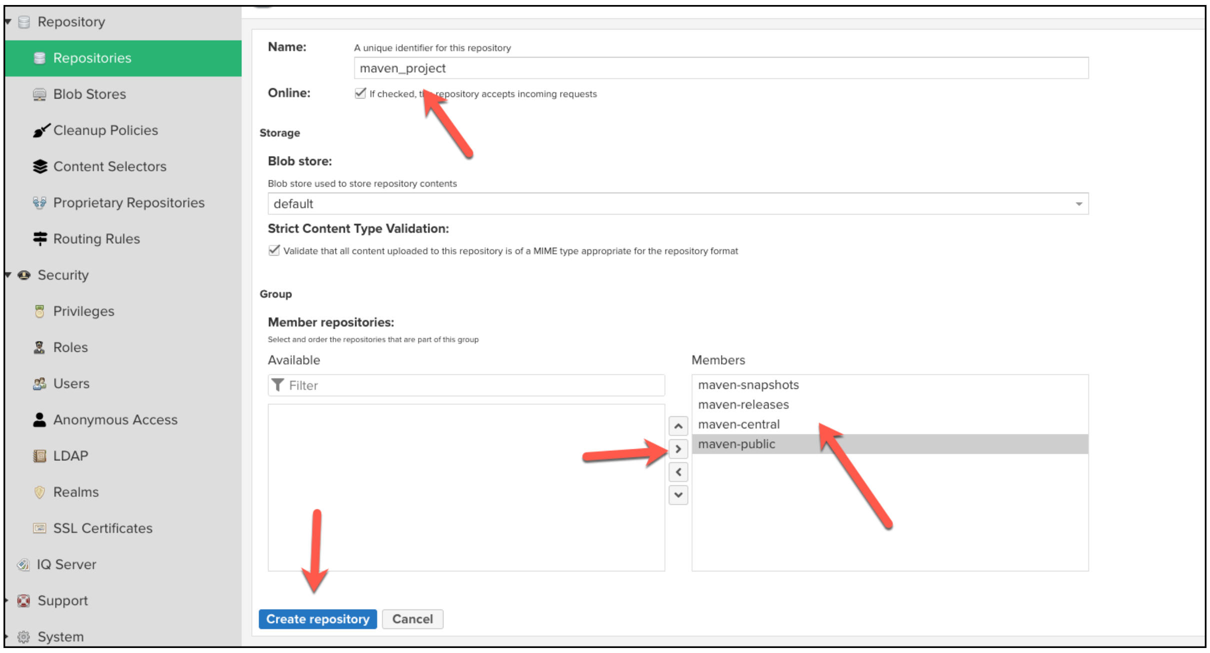
15. The above command will start the nexus service on port 8081. To access the nexus dashboard, visit http://:8081. You will be able to see the nexus homepage as shown below.
    http://public-ip:8081
16. Default username is admin
17. You can find the default admin password in /app/sonatype-work/nexus3/admin.password file.
    cat /app/sonatype-work/nexus3/admin.password
18. Once you login, you will be prompted to reset the password.



Repositories Manage repositories

Name	Type	Format	Status	URL	Health check	IQ Po
maven-central	proxy	maven2	Online - Ready to Connect	copy	Analyze	
maven-public	group	maven2	Online	copy		
maven-releases	hosted	maven2	Online	copy		
maven-snapshots	hosted	maven2	Online	copy		
nuget-group	group	nuget	Online	copy		
nuget-hosted	hosted	nuget	Online	copy		
nuget.org-proxy	proxy	nuget	Online - Ready to Connect	copy	Analyze	

Once you select create repository and select maven2(group)



Name: A unique identifier for this repository
maven_project

Online: ✓ If checked, the repository accepts incoming requests

Storage

Blob store:
Blob store used to store repository contents
default

Strict Content Type Validation:
✓ Validate that all content uploaded to this repository is of a MIME type appropriate for the repository format

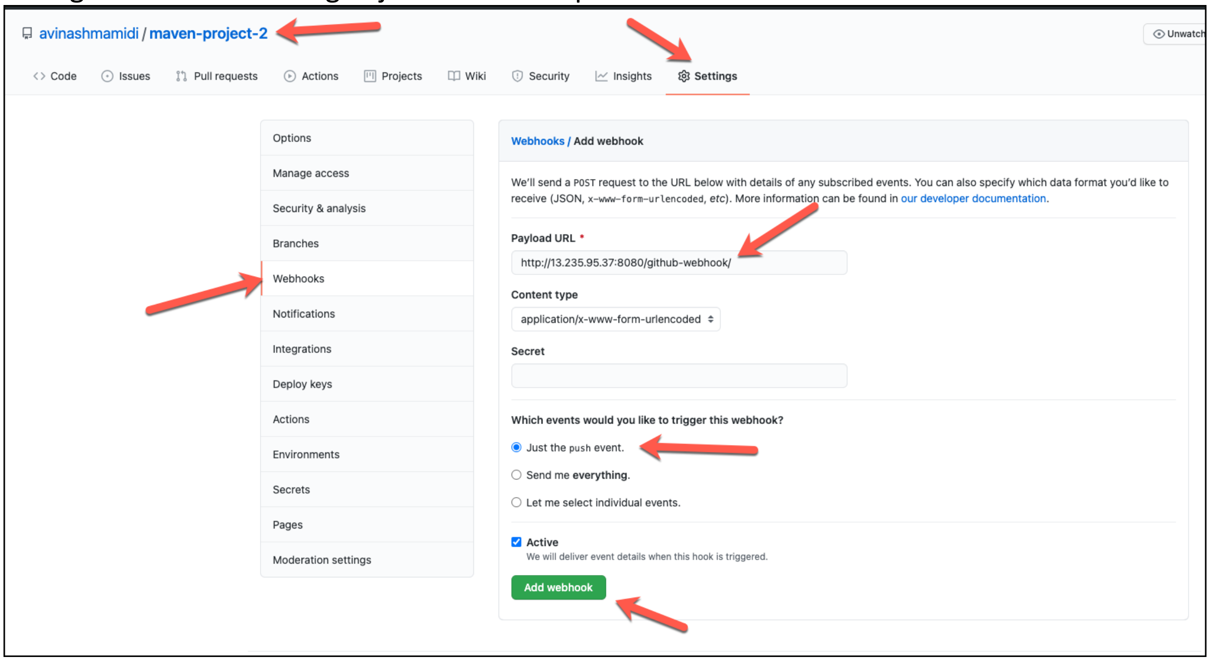
Group

Member repositories:
Select and order the repositories that are part of this group

Available

Members
maven-snapshots
maven-releases
maven-central
maven-public

Section-4: Clone the repo and do changes and push to your Github repo

1. Clone this repo https://github.com/avinashmamidi/maven-project-2
2. Change ip of nexus in pom.xml line 32 and 36
3. Change SonarQube config in jenkinsfile and push it back
4. 

avinashmamidi / maven-project-2

Code · Issues · Pull requests · Actions · Projects · Wiki · Security · Insights · Settings

Options · Manage access · Security & analysis · Branches · Webhooks · Notifications · Integrations · Deploy keys · Actions · Environments · Secrets · Pages · Moderation settings

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://13.235.95.37:8080/github-webhook/

Content type: application/x-www-form-urlencoded

Secret:

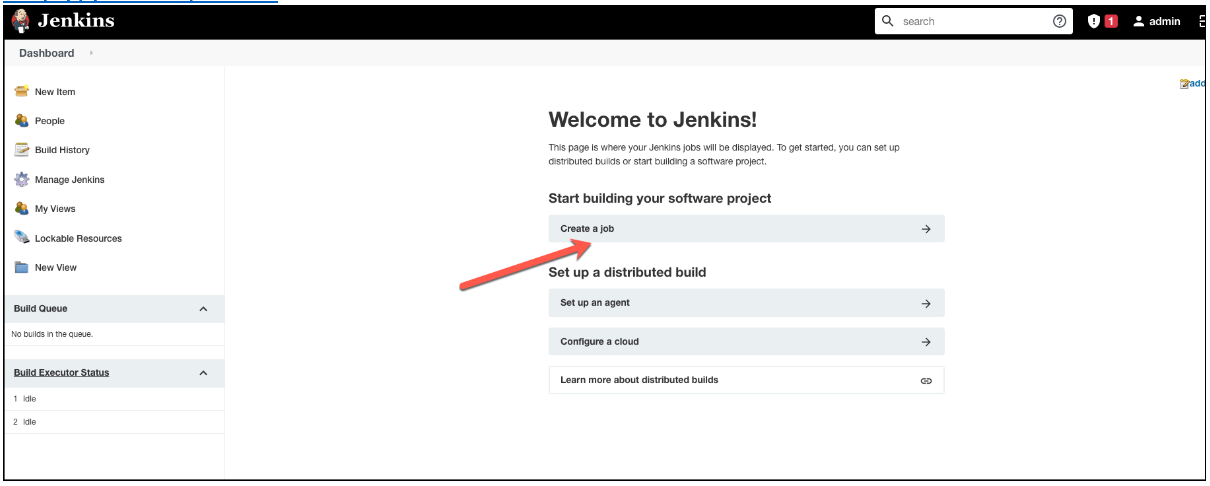
Which events would you like to trigger this webhook?

- [✓ Just the push event.]
- Send me everything.
- Let me select individual events.

[✓ Active] We will deliver event details when this hook is triggered.

Add webhook

Section-5: Navigate to Jenkins Dashboard

1. We can access Jenkins using the url below http://public-ip:8080
2. 

Jenkins · search · admin

Dashboard

New Item · People · Build History · Manage Jenkins · My Views · Lockable Resources · New View

Build Queue: No builds in the queue.

Build Executor Status: 1 Idle · 2 Idle

Welcome to Jenkins!

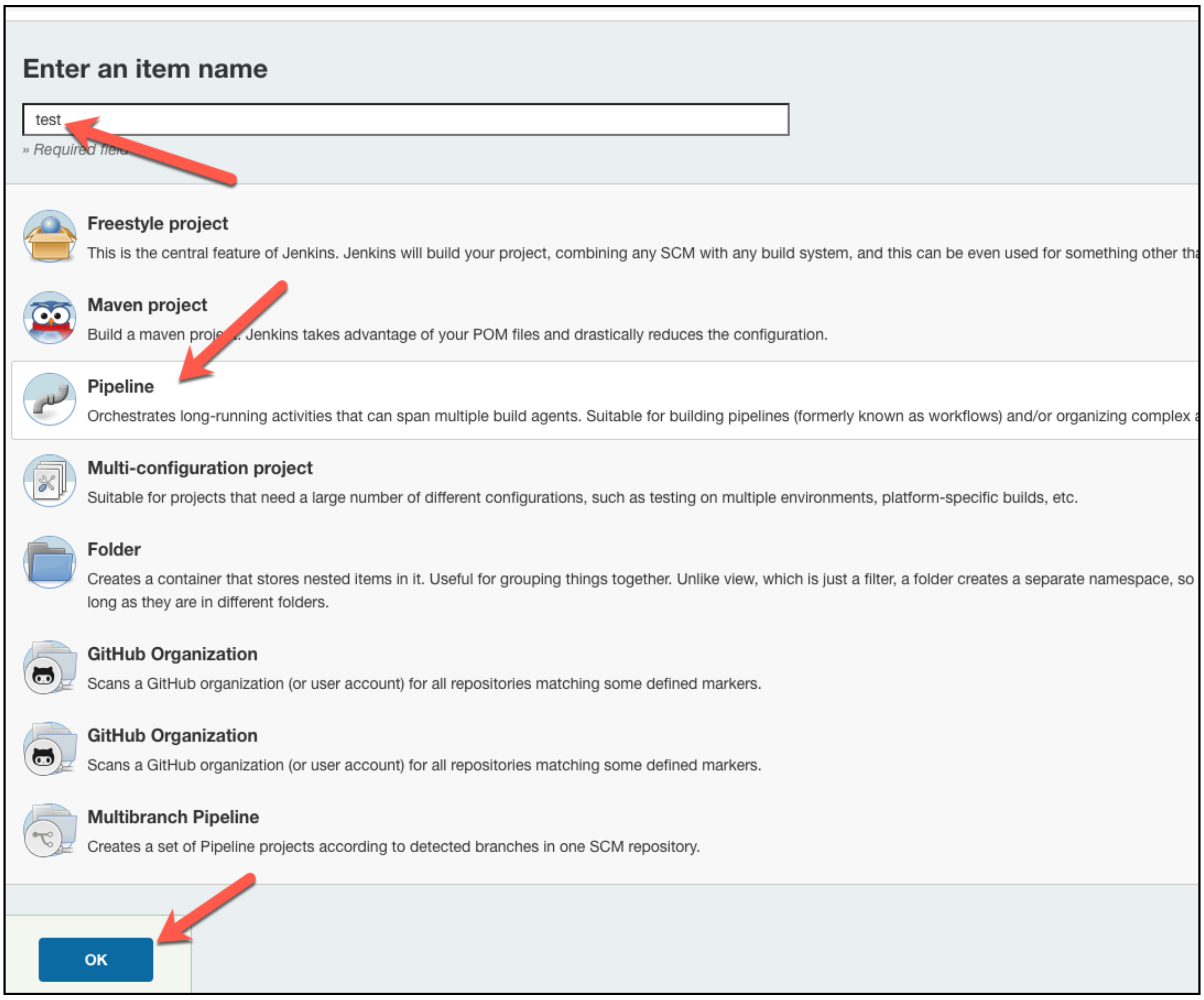
This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.

Start building your software project

Create a job

Set up a distributed build

Set up an agent · Configure a cloud · Learn more about distributed builds

3. 

Enter an item name

test

» Required field

Freestyle project

This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other th

Maven project

Build a maven project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

Pipeline

Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex

Multi-configuration project

Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder

Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so long as they are in different folders.

GitHub Organization

Scans a GitHub organization (or user account) for all repositories matching some defined markers.

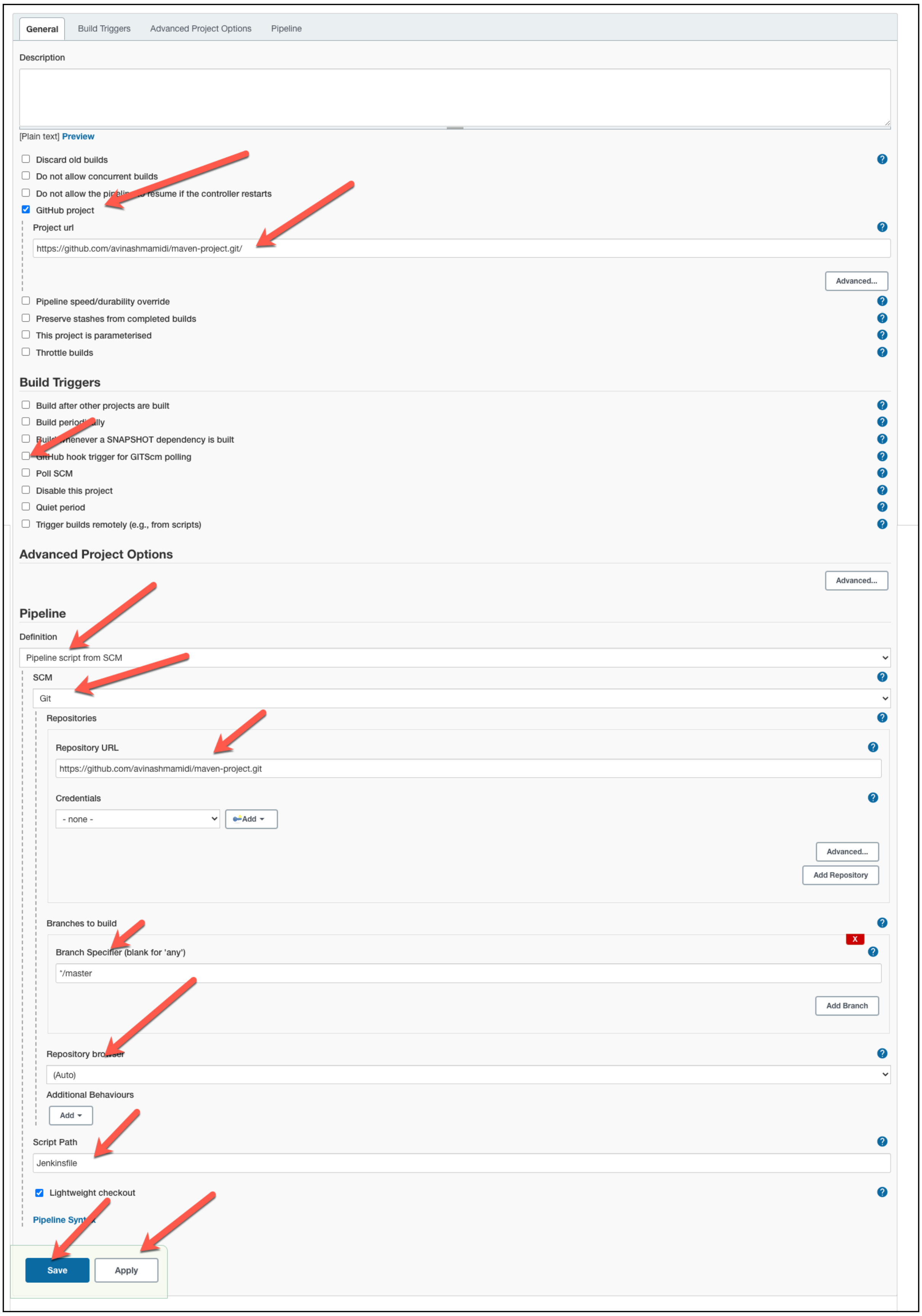
GitHub Organization

Scans a GitHub organization (or user account) for all repositories matching some defined markers.

Multibranch Pipeline

Creates a set of Pipeline projects according to detected branches in one SCM repository.

4.



General | Build Triggers | Advanced Project Options | Pipeline

Description

[Plain text] Preview

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

Project url: https://github.com/avinashmamidi/maven-project.git/

- [ ] Pipeline speed/durability override
- [ ] Preserve stashes from completed builds
- [ ] This project is parameterised
- [ ] Throttle builds

Build Triggers

- [ ] Build after other projects are built
- [ ] Build periodically
- [ ] Build whenever a SNAPSHOT dependency is built
- [ ] GitHub hook trigger for GITScm polling
- [ ] Poll SCM
- [ ] Disable this project
- [ ] Quiet period
- [ ] Trigger builds remotely (e.g., from scripts)

Advanced Project Options

Pipeline

Definition: Pipeline script from SCM

SCM: Git

Repositories

Repository URL: https://github.com/avinashmamidi/maven-project.git

Credentials: - none -

Branches to build

Branch Specifier (blank for 'any'): */master

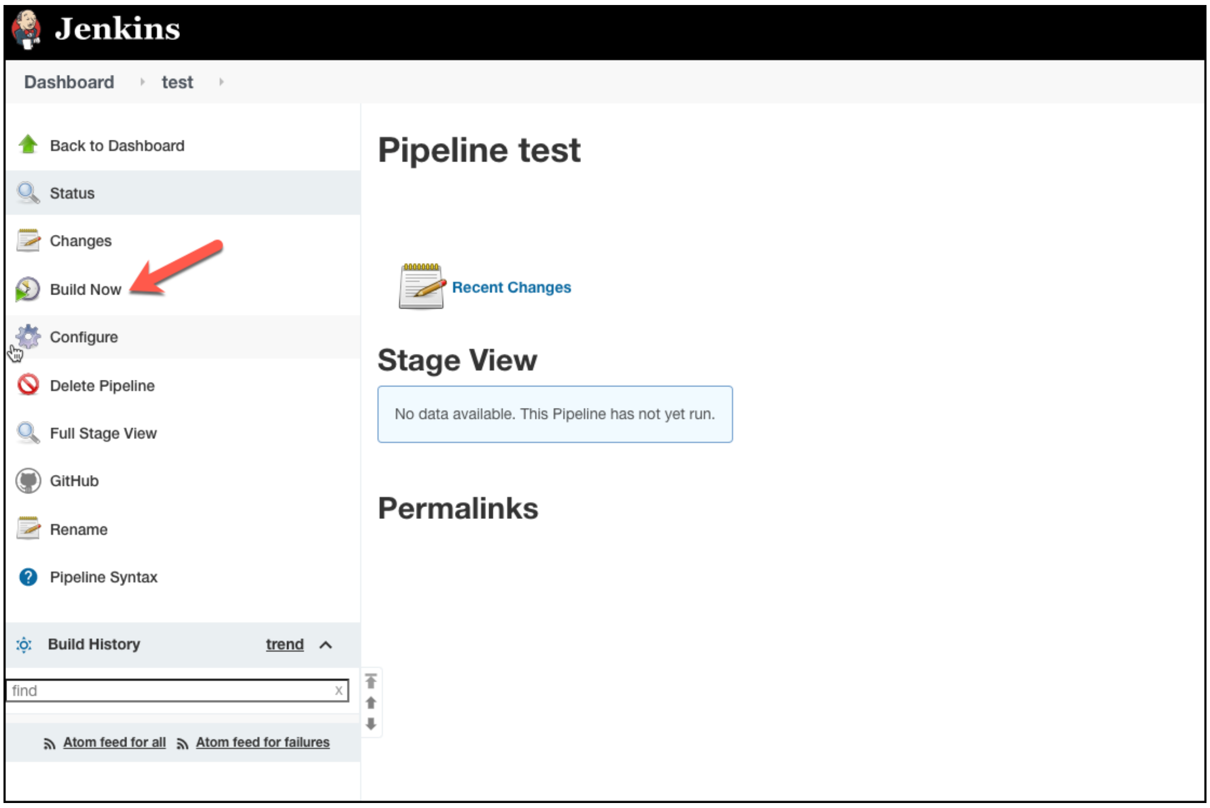
Repository browser: (Auto)

Additional Behaviours

Script Path: Jenkinsfile

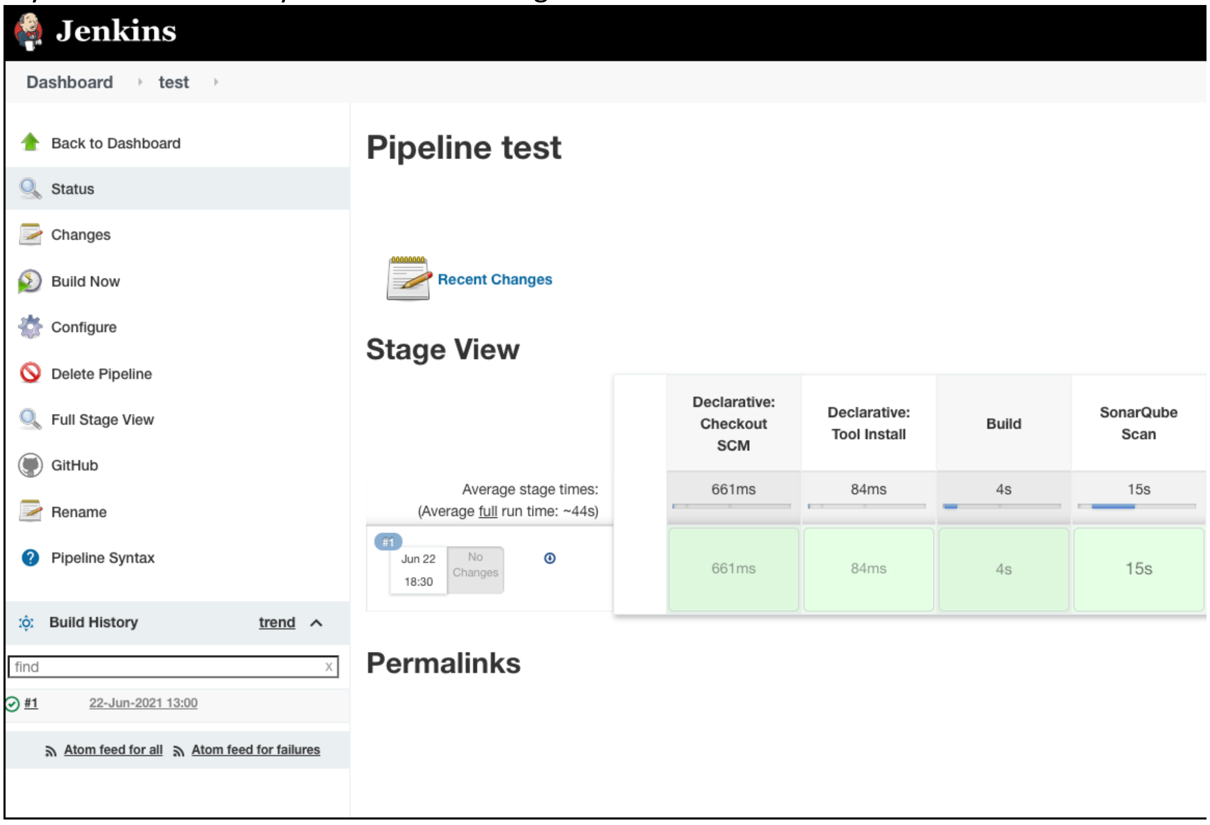
- [x] Lightweight checkout

Pipeline Syntax



No data available. This Pipeline has not yet run.

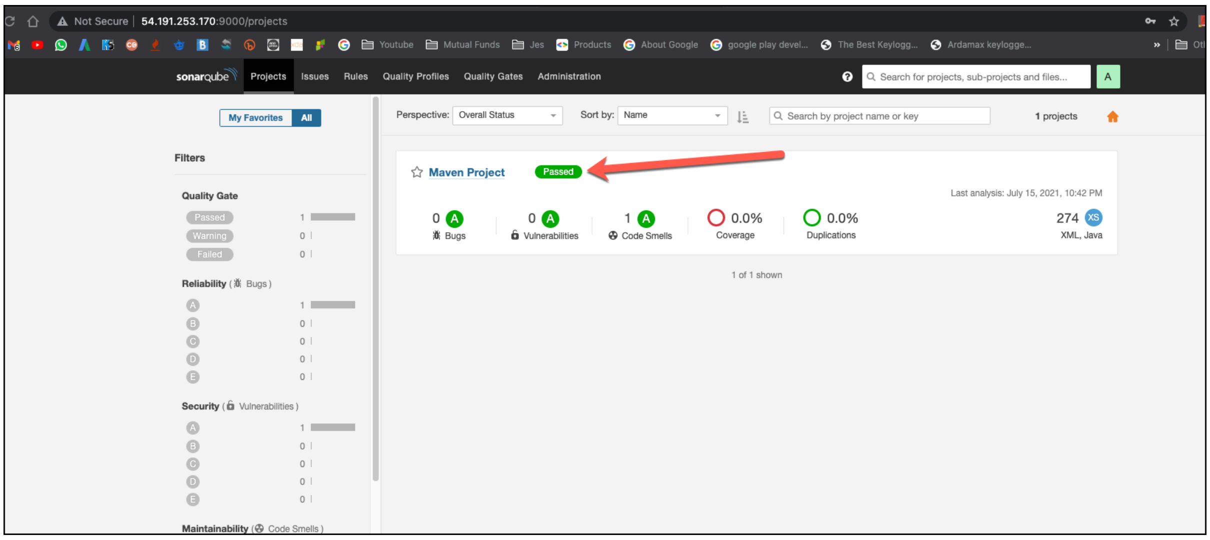
5.
6. If you follow the doc you should see this green



#1 22-Jun-2021 13:00

	Declarative: Checkout SCM	Declarative: Tool Install	Build	SonarQube Scan
Average stage times: (Average full run time: ~44s)	661ms	84ms	4s	15s
#1 Jun 22 18:30 No Changes	661ms	84ms	4s	15s

7.
8. Now sonarqube is configured with Jenkins. To check we can login to sonarqube from web browser

9. 

10. From below section we configure Jenkins with Nexus

Section-6: Login to Jenkins EC2 (Instance 1) → To configure Jenkins with Nexus

Install Maven:

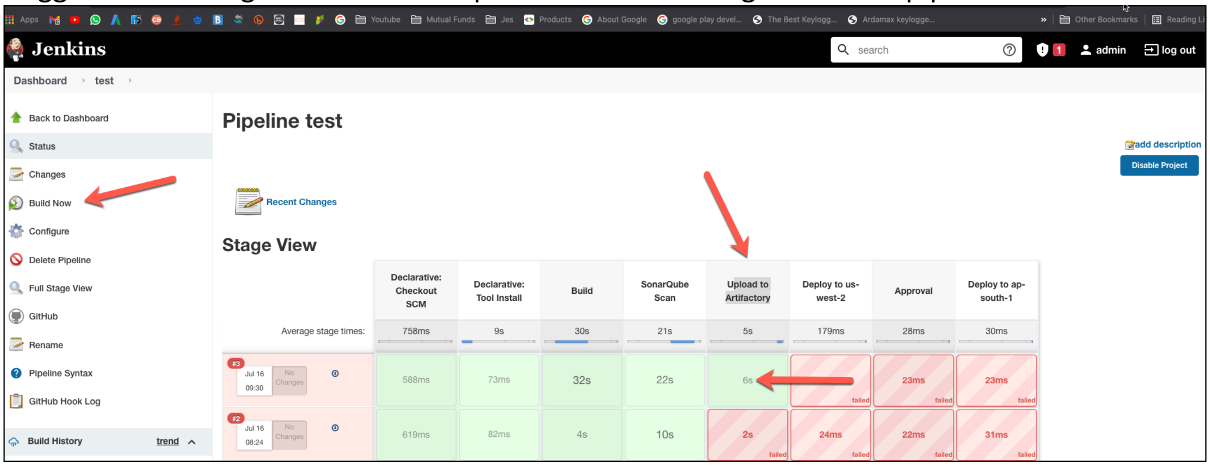
https://docs.aws.amazon.com/neptune/latest/userguide/iam-auth-connect-prerq.html

1. We have to encrypt nexus password and update it in the file below
   mvn -emp admin
2. Create folder in the root folder
   a. cd /root
   b. mkdir .m2
3. You will get encrypted password from above command, you need to change in below file.
   a. cd .m2
4. create settings-security.xml file and
   a. vi settings-security.xml
5. add the content to above file after change above password from line 6
   ```
   <?xml version="1.0"?>
   <settingsSecurity>
     <master>{admin}</master>
   </settingsSecurity>
   ```
6. We have to encrypt nexus password and update it in the below file
   mvn -ep admin
7. create settings.xml file
   a. vi settings.xml
8. above file below content after change above password from line 6
9.
   ```
   <?xml version="1.0" encoding="UTF-8"?>

   <settings 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/xsd/settings-1.0.0.xsd">
   ```

```
 <localRepository>/var/lib/jenkins/.m2/repository</localRepository>

<servers>
  <server>
     <id>nexus</id>
     <username>admin</username>
     <password>{admin}</password>
   </server>
</servers>

<mirrors>
   <mirror>
    <id>nexus</id>
    <name>nexus</name>
    <url>http://13.235.132.119:8081/repository/maven_project/</url>
    <mirrorOf>*</mirrorOf>
   </mirror>
 </mirrors>



</settings>
```

10. move above two files to /var/lib/jenkins/.m2
11. Change ownership and access of the 2 files
12. chown jenkins:jenkins settings.xml settings-security.xml
13. chmdo 755 settings.xml settings-security.xml

Section-7: Navigate to Jenkins Dashboard

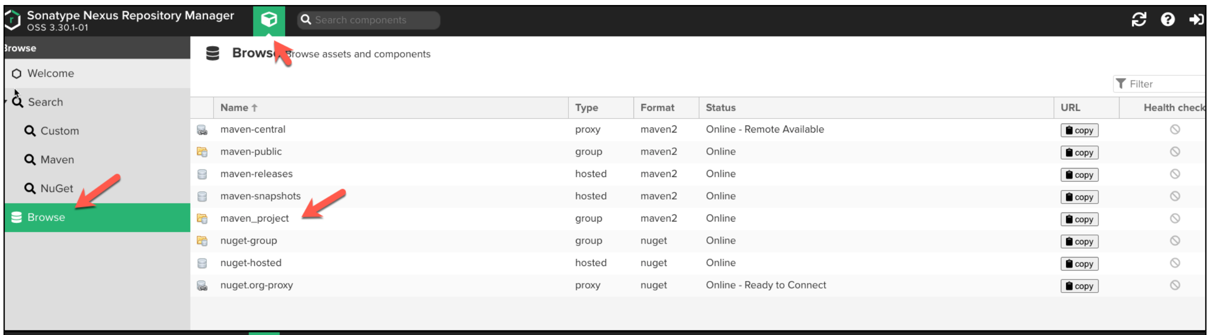
1. We can access Jenkins using the url below
2. http://public-ip:8080
3. Trigger the build again to see the “Upload to Artifact” to green in the pipeline
4. 

Pipeline test

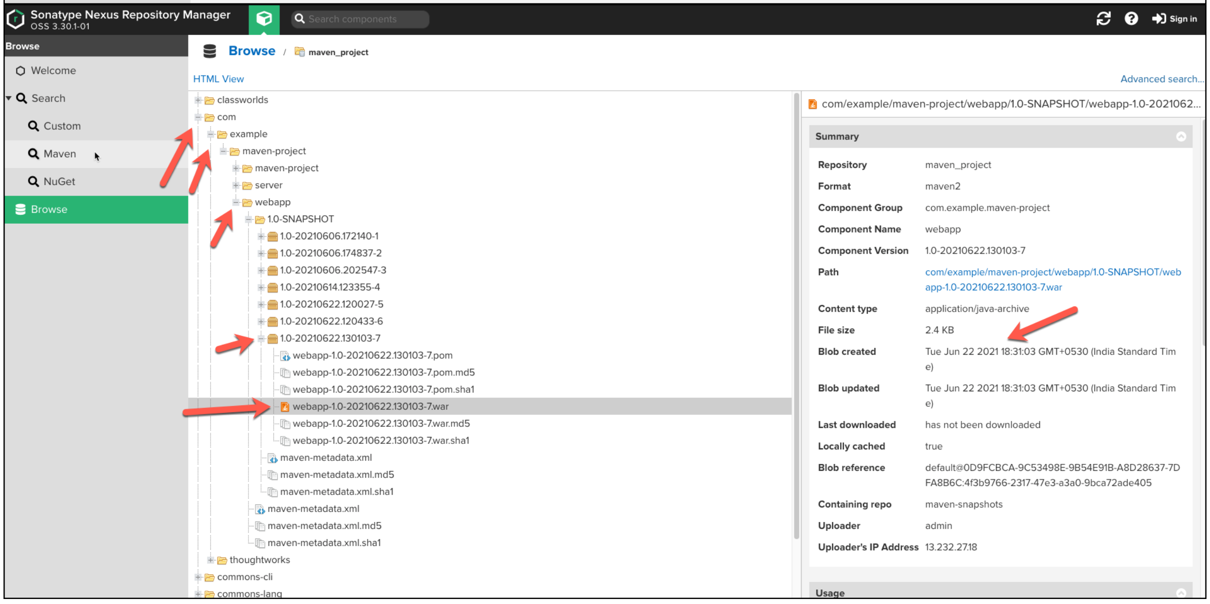
Stage View

	Declarative: Checkout SCM	Declarative: Tool Install	Build	SonarQube Scan	Upload to Artifactory	Deploy to us-west-2	Approval	Deploy to ap-south-1
Average stage times:	758ms	9s	30s	21s	5s	179ms	28ms	30ms
#3 Jul 16 09:30 No Changes	588ms	73ms	32s	22s	6s	failed	23ms failed	23ms failed
#2 Jul 16 08:24 No Changes	619ms	82ms	4s	10s	2s failed	24ms failed	22ms failed	31ms failed

5. Login to nexus on web browser to check whether artifact is uploaded

6. 

Name	Type	Format	Status	URL	Health check
maven-central	proxy	maven2	Online - Remote Available	copy	
maven-public	group	maven2	Online	copy	
maven-releases	hosted	maven2	Online	copy	
maven-snapshots	hosted	maven2	Online	copy	
maven_project	group	maven2	Online	copy	
nuget-group	group	nuget	Online	copy	
nuget-hosted	hosted	nuget	Online	copy	
nuget.org-proxy	proxy	nuget	Online - Ready to Connect	copy	

7. 

```
classworlds
com
  example
    maven-project
      maven-project
      server
      webapp
        1.0-SNAPSHOT
          1.0-20210606.172140-1
          1.0-20210606.174837-2
          1.0-20210606.202547-3
          1.0-20210614.123355-4
          1.0-20210622.120027-5
          1.0-20210622.120433-6
          1.0-20210622.130103-7
            webapp-1.0-20210622.130103-7.pom
            webapp-1.0-20210622.130103-7.pom.md5
            webapp-1.0-20210622.130103-7.pom.sha1
            webapp-1.0-20210622.130103-7.war
            webapp-1.0-20210622.130103-7.war.md5
            webapp-1.0-20210622.130103-7.war.sha1
          maven-metadata.xml
          maven-metadata.xml.md5
          maven-metadata.xml.sha1
        maven-metadata.xml
        maven-metadata.xml.md5
        maven-metadata.xml.sha1
  thoughtworks
commons-cli
commons-lang
```

com/example/maven-project/webapp/1.0-SNAPSHOT/webapp-1.0-2021062...

Summary

Repository	maven_project
Format	maven2
Component Group	com.example.maven-project
Component Name	webapp
Component Version	1.0-20210622.130103-7
Path	com/example/maven-project/webapp/1.0-SNAPSHOT/webapp-1.0-20210622.130103-7.war
Content type	application/java-archive
File size	2.4 KB
Blob created	Tue Jun 22 2021 18:31:03 GMT+0530 (India Standard Time)
Blob updated	Tue Jun 22 2021 18:31:03 GMT+0530 (India Standard Time)
Last downloaded	has not been downloaded
Locally cached	true
Blob reference	default@0D9FCBCA-9C53498E-9B54E91B-A8D28637-7DFA8B6C:4f3b9766-2317-47e3-a3a0-9bca72ade405
Containing repo	maven-snapshots
Uploader	admin
Uploader's IP Address	13.232.27.18

Usage

8. Since I ran the job 7 times we have 7 repos. You should see only one for the first time.

Section-8: Setup Ansible in Jenkins Instance (Instance 1) → For deployment

1 Install Ansible
sudo amazon-linux-extras install ansible2
2 Sudo su
3 Create user and password
useradd ansadmin
passwd ansadmin (enter password when prompted)
5. cd /etc/ansible/
6. vi ansible.cfg (uncomment host_key_checking = False)
7. vi hosts (enter below content)
[dev]
3.108.227.139 ansible_user=ansadmin ansible_password=ansadmin
(deploymentserverip username password)
[prod]
3.108.227.139 ansible_user=ansadmin ansible_password=ansadmin
(deploymentserverip username password)

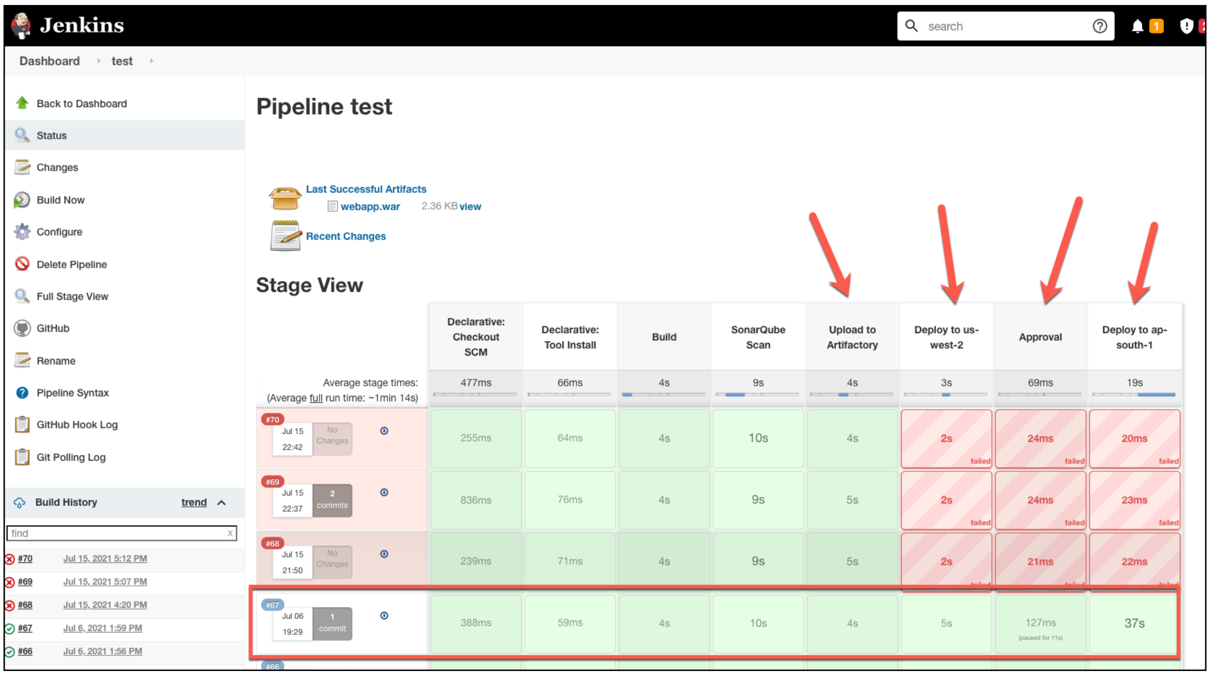
Section-9: Setup Deployment Instances

1. login to EC2 instance

2. Sudo su
3. Create user and password
   useradd ansadmin
   passwd ansadmin (enter password when prompted)
4. vi /etc/ssh/sshd_config (uncomment PasswordAuthentication yes and comment PasswordAuthentication yes)
5. service sshd restart
6. Make sure below is uncommented in visudo
   %wheel ALL=(ALL)   NOPASSWD: ALL
7. usermod -aG wheel ansadmin
8. Install tomcat webserver
   sudo amazon-linux-extras install tomcat8.5
9. Start tomcat service:
10. sudo systemctl start tomcat

    http://3.108.227.139:8080/webapp/

Section-10: Check the pipeline till deployment

1. We can access Jenkins using the url below
2. http://yourpublic-ip:8080
3. Once you trigger the build . You can find the whole pipeline is green and deployment succeded.
4. 

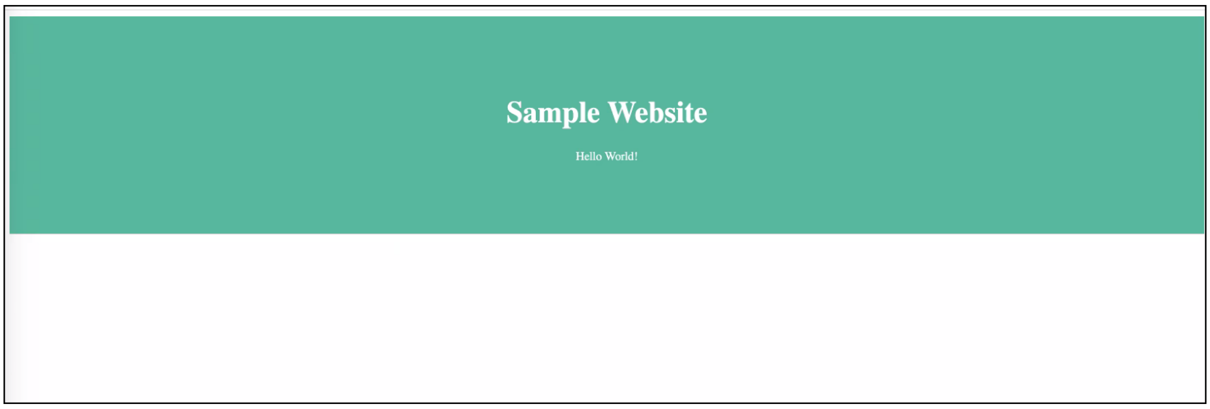
Pipeline test

Stage View

	Declarative: Checkout SCM	Declarative: Tool Install	Build	SonarQube Scan	Upload to Artifactory	Deploy to us-west-2	Approval	Deploy to ap-south-1
Average stage times: (Average full run time: ~1min 14s)	477ms	66ms	4s	9s	4s	3s	69ms	19s
#70 Jul 15 22:42 No Changes	255ms	64ms	4s	10s	4s	2s failed	24ms failed	20ms failed
#69 Jul 15 22:37 2 commits	836ms	76ms	4s	9s	5s	2s failed	24ms failed	23ms failed
#68 Jul 15 21:50 No Changes	239ms	71ms	4s	9s	5s	2s	21ms	22ms
#67 Jul 06 19:29 1 commit	388ms	59ms	4s	10s	4s	5s	127ms (paused for 11s)	37s

#70	Jul 15, 2021 5:12 PM
#69	Jul 15, 2021 5:07 PM
#68	Jul 15, 2021 4:20 PM
#67	Jul 6, 2021 1:59 PM
#66	Jul 6, 2021 1:56 PM

5. You can access application from following url
6. http://your-deployment-server-publicIP:8080/webapp/ which look like below

7. 

Note: Now our Jenkins pipeline is completed Including deployment. From below section you can see how to test of this Jenkins pipeline end to end.

Testing:

- Path to change content on application:
- maven-project-2/webapp/src/main/webapp/index.jsp
- Once you change content on above file and push it to Github.
- Jenkins job will automatically get triggered build, scan code, push the artifact and finally deploy
- You can repeat the testing multiple times