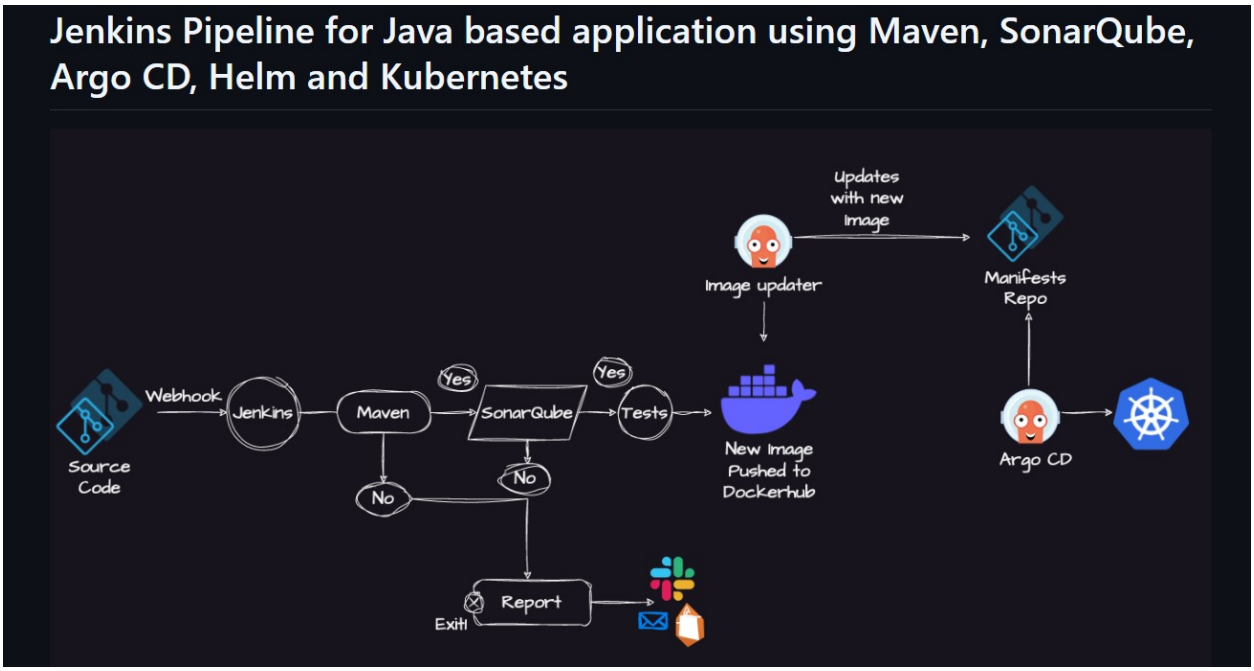
JENKINS END TO END CICD Implementation with Detailed Notes



Here are the step-by-step details to set up an end-to-end Jenkins pipeline for a Java application using SonarQube, Argo CD, Helm, and Kubernetes:

Prerequisites:

- Java application code hosted on a Git repository
- Jenkins server
- Kubernetes cluster
- Helm package manager
- Argo CD

```
1. Install the necessary Jenkins plugins:
   1.1 Git plugin
   1.2 Maven Integration plugin
   1.3 Pipeline plugin
   1.4 Kubernetes Continuous Deploy plugin

2. Create a new Jenkins pipeline:
   2.1 In Jenkins, create a new pipeline job and configure it with the Git repository URL for the Java application.
   2.2 Add a Jenkinsfile to the Git repository to define the pipeline stages.

3. Define the pipeline stages:
    Stage 1: Checkout the source code from Git.
    Stage 2: Build the Java application using Maven.
    Stage 3: Run unit tests using JUnit and Mockito.
    Stage 4: Run SonarQube analysis to check the code quality.
    Stage 5: Package the application into a JAR file.
    Stage 6: Deploy the application to a test environment using Helm.
    Stage 7: Run user acceptance tests on the deployed application.
    Stage 8: Promote the application to a production environment using Argo CD.

4. Configure Jenkins pipeline stages:
    Stage 1: Use the Git plugin to check out the source code from the Git repository.
    Stage 2: Use the Maven Integration plugin to build the Java application.
    Stage 3: Use the JUnit and Mockito plugins to run unit tests.
    Stage 4: Use the SonarQube plugin to analyze the code quality of the Java application.
    Stage 5: Use the Maven Integration plugin to package the application into a JAR file.
    Stage 6: Use the Kubernetes Continuous Deploy plugin to deploy the application to a test environment using Helm.
    Stage 7: Use a testing framework like Selenium to run user acceptance tests on the deployed application.
    Stage 8: Use Argo CD to promote the application to a production environment.
```

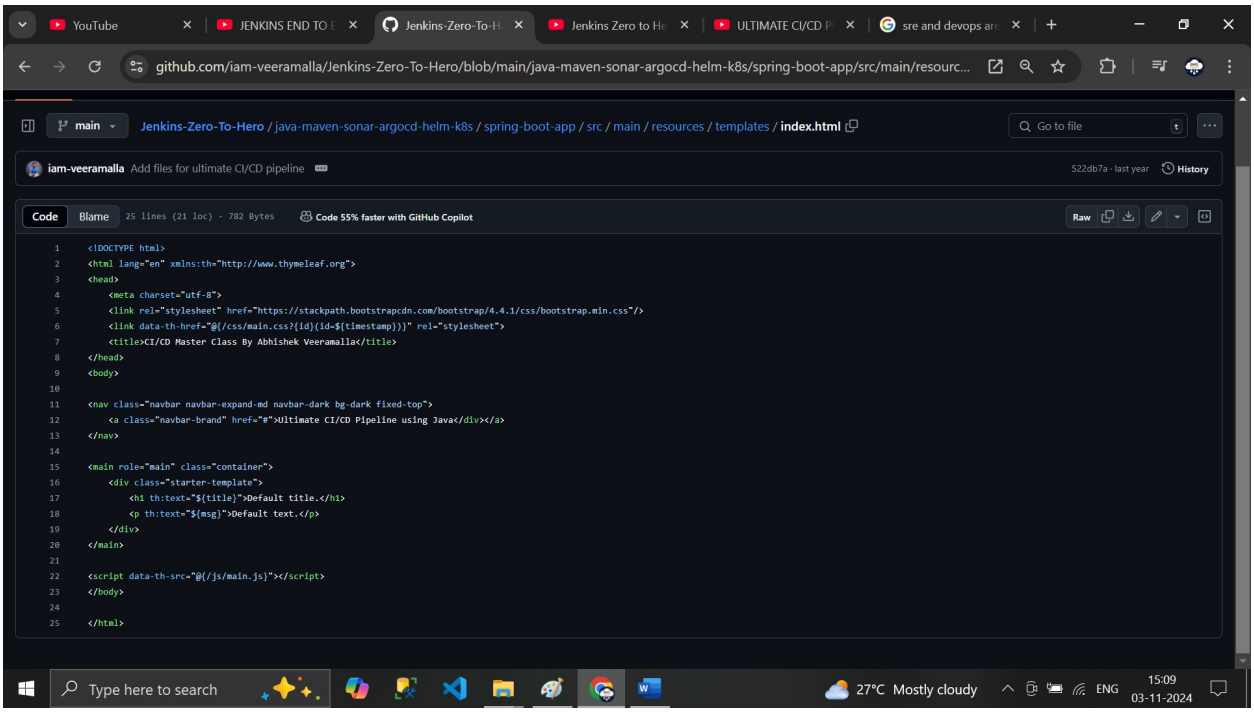
```
5. Set up Argo CD:
    Install Argo CD on the Kubernetes cluster.
    Set up a Git repository for Argo CD to track the changes in the Helm charts and Kubernetes manifests.
    Create a Helm chart for the Java application that includes the Kubernetes manifests and Helm values.
    Add the Helm chart to the Git repository that Argo CD is tracking.

6. Configure Jenkins pipeline to integrate with Argo CD:
   6.1 Add the Argo CD API token to Jenkins credentials.
   6.2 Update the Jenkins pipeline to include the Argo CD deployment stage.

7. Run the Jenkins pipeline:
   7.1 Trigger the Jenkins pipeline to start the CI/CD process for the Java application.
   7.2 Monitor the pipeline stages and fix any issues that arise.
```

This end-to-end Jenkins pipeline will automate the entire CI/CD process for a Java application, from code checkout to production deployment, using popular tools like SonarQube, Argo CD, Helm, and Kubernetes.

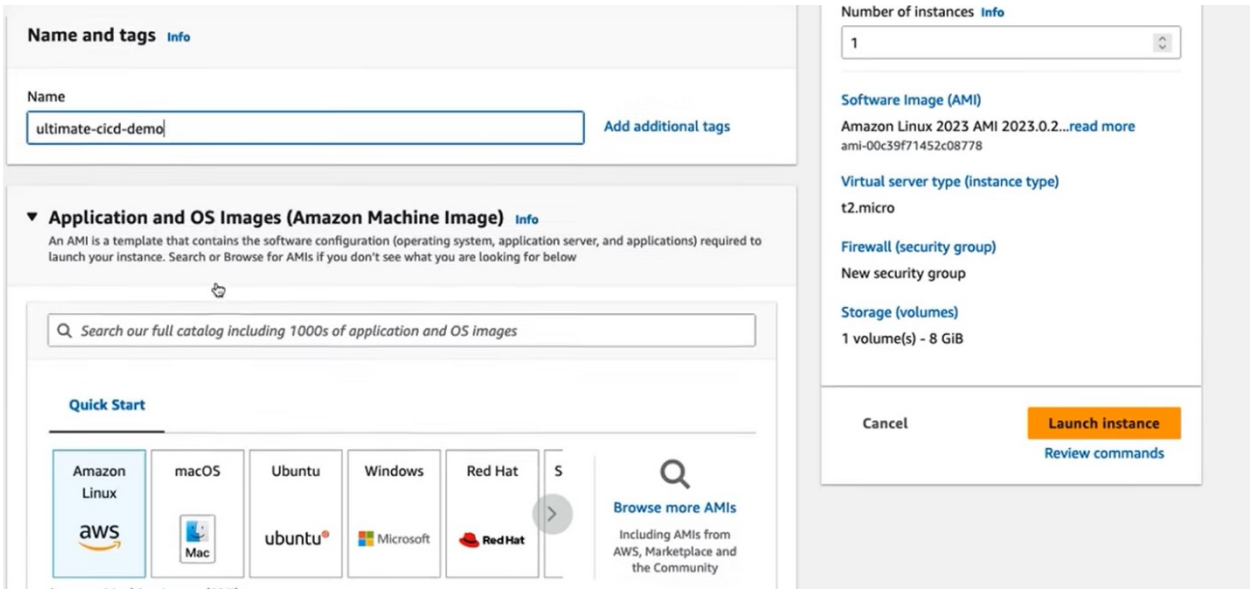
springboot applicaton

Code | Blame | 24 lines (19 loc) · 771 Bytes | Code 55% faster with GitHub Copilot

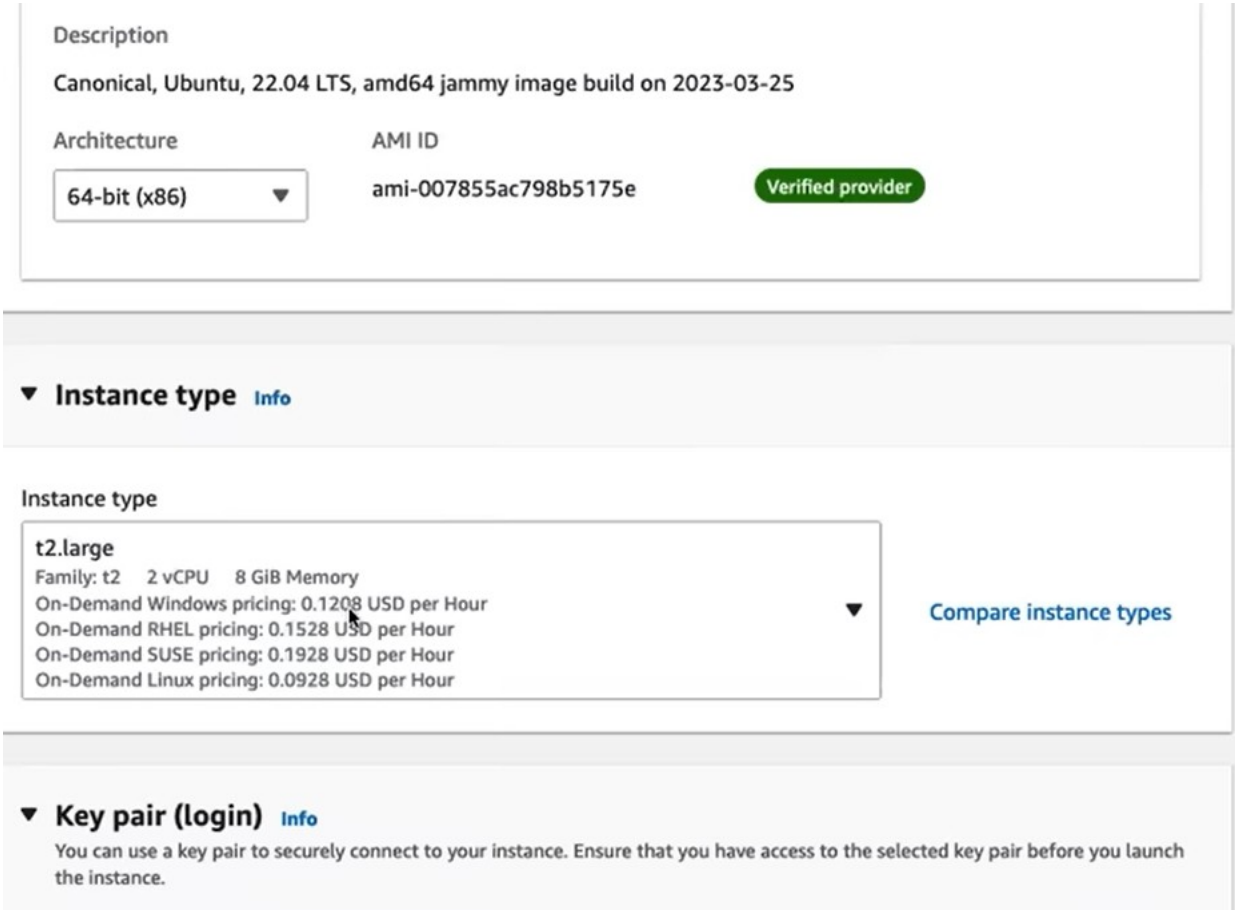
```java
package com.abhishek;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
import org.springframework.web.bind.annotation.GetMapping;

@SpringBootApplication
@Controller
public class StartApplication {

    @GetMapping("/")
    public String index(final Model model) {
        model.addAttribute("title", "I have successfuly built a sprint boot application using Maven");
        model.addAttribute("msg", "This application is deployed on to Kubernetes using Argo CD");
        return "index";
    }

    public static void main(String[] args) {
        SpringApplication.run(StartApplication.class, args);
    }

}
```

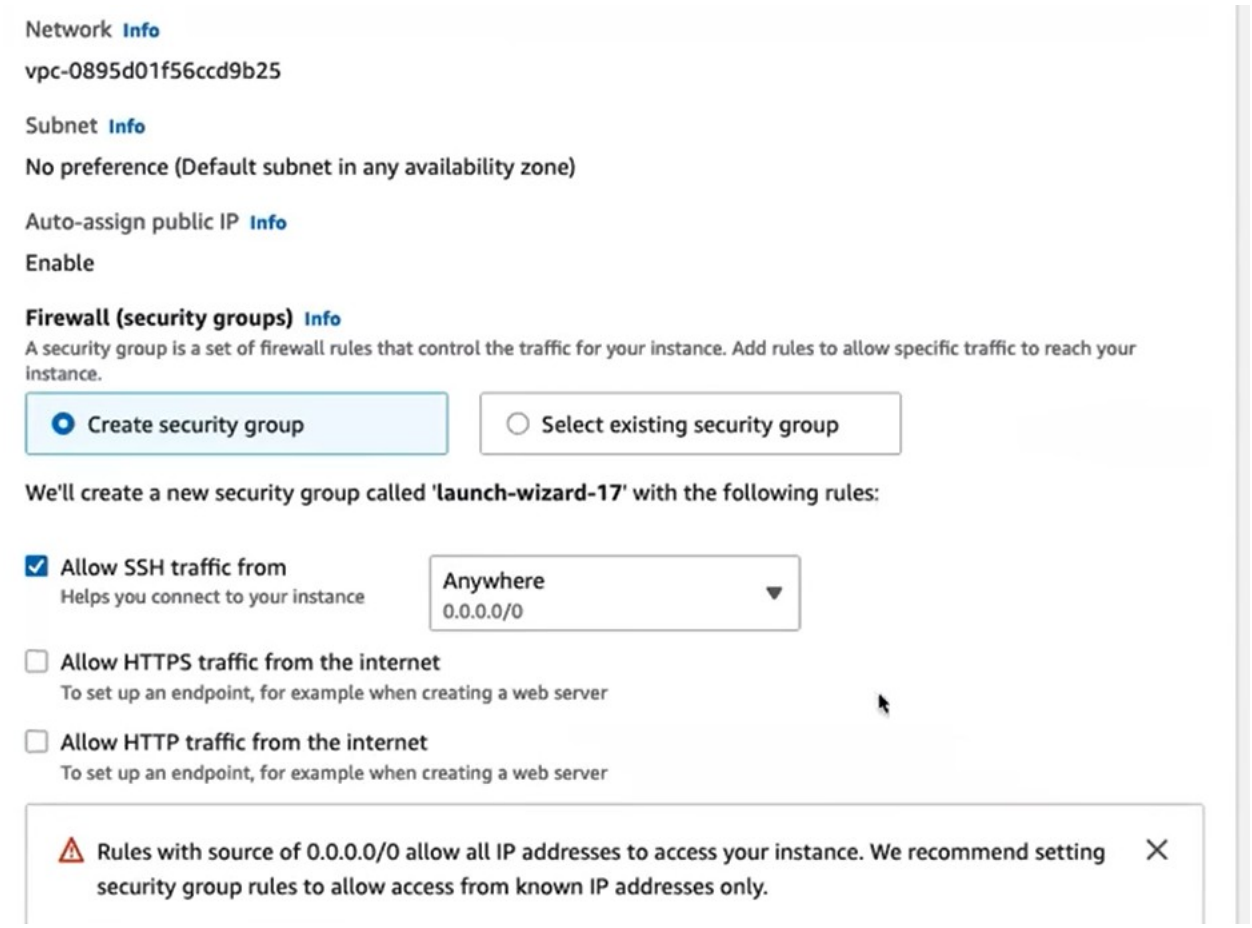


select AWS EC2 instance …..Paid one … Free tier will not Work with many microservices

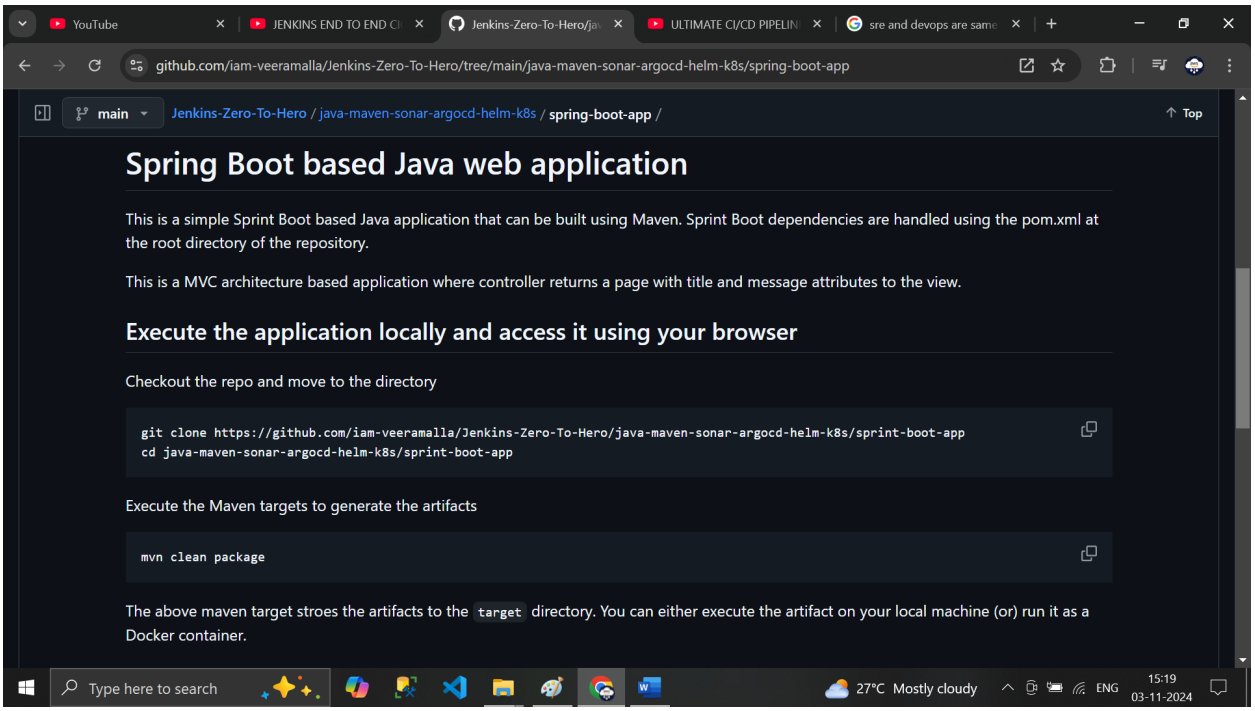


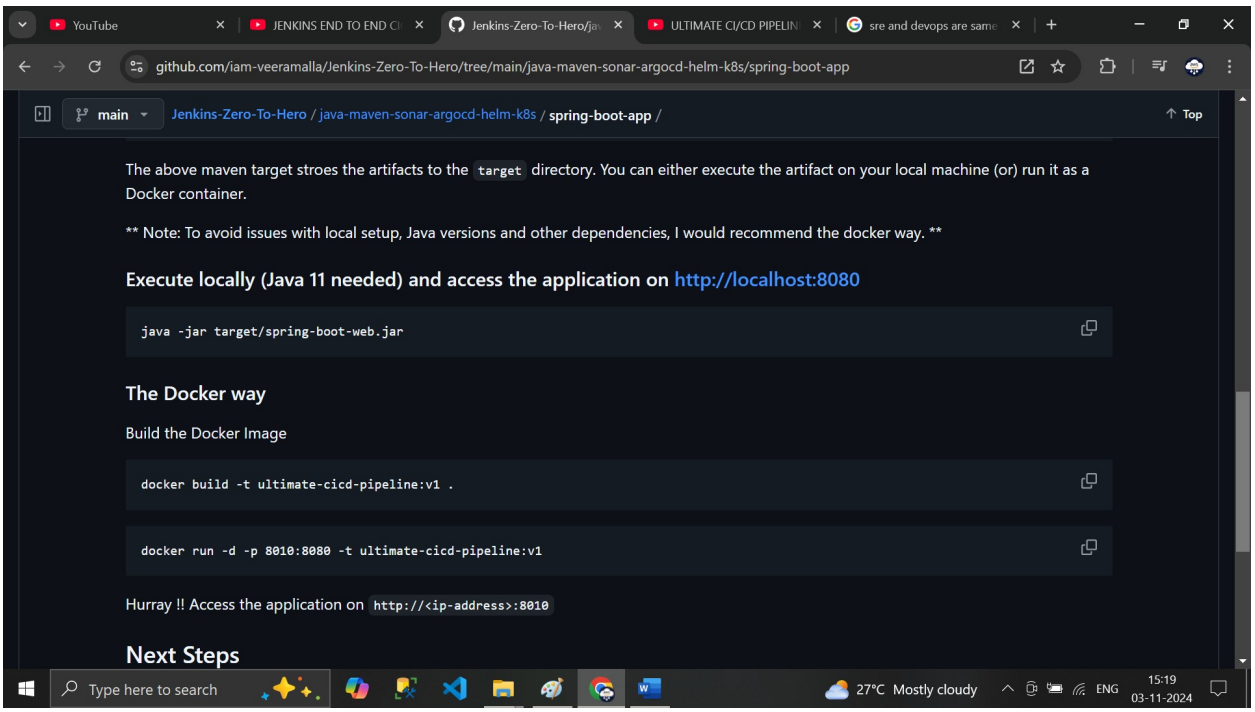
 # instance type t2.large 2 cpu

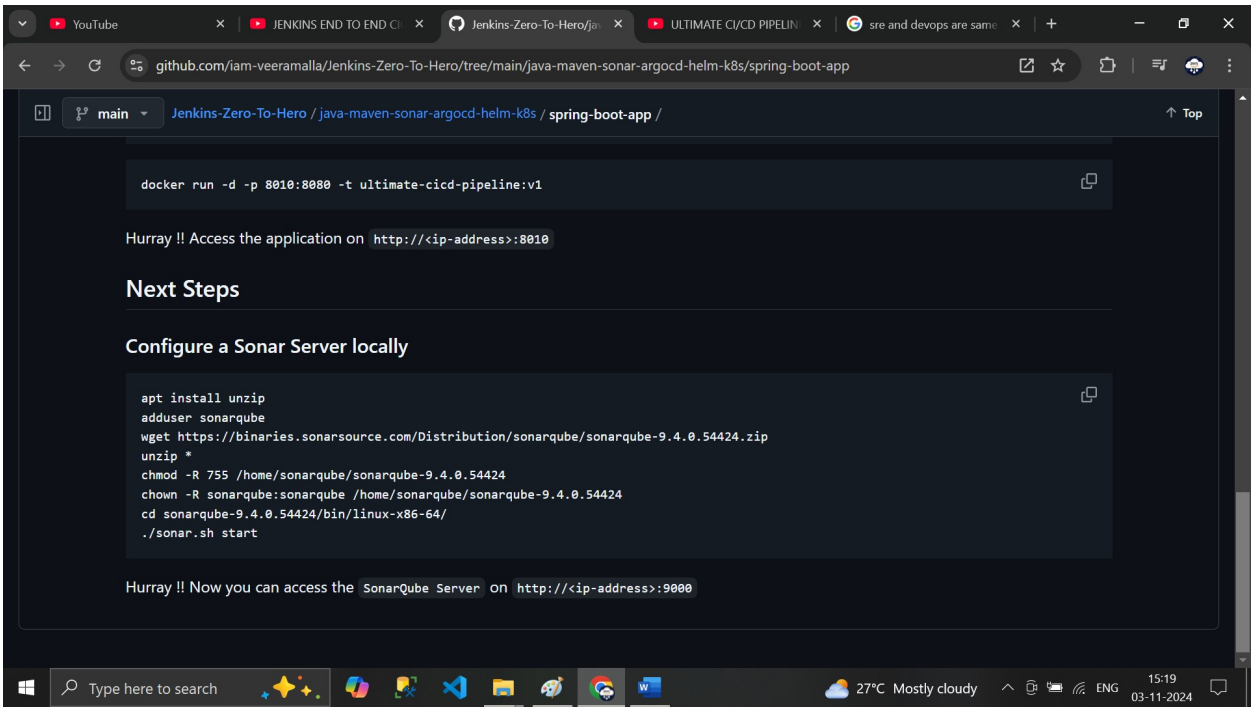





```
➜  Jenkins-Zero-To-Hero git:(main) cd java-maven-sonar-argocd-helm-k8s
➜  java-maven-sonar-argocd-helm-k8s git:(main) ls
README.md       spring-boot-app
➜  java-maven-sonar-argocd-helm-k8s git:(main) cd spring-boot-app
➜  spring-boot-app git:(main) ls
Dockerfile README.md  pom.xml    src
➜  spring-boot-app git:(main)
```


github.com/iam-veeramalla/Jenkins-Zero-To-Hero/tree/main/java-maven-sonar-argocd-helm-k8s/spring-boot-app
main
Jenkins-Zero-To-Hero / java-maven-sonar-argocd-helm-k8s / spring-boot-app /
Top
Spring Boot based Java web application
This is a simple Sprint Boot based Java application that can be built using Maven. Sprint Boot dependencies are handled using the pom.xml at the root directory of the repository.
This is a MVC architecture based application where controller returns a page with title and message attributes to the view.
Execute the application locally and access it using your browser
Checkout the repo and move to the directory
git clone https://github.com/iam-veeramalla/Jenkins-Zero-To-Hero/java-maven-sonar-argocd-helm-k8s/sprint-boot-app
cd java-maven-sonar-argocd-helm-k8s/sprint-boot-app
Execute the Maven targets to generate the artifacts
mvn clean package
The above maven target stroes the artifacts to the target directory. You can either execute the artifact on your local machine (or) run it as a Docker container.
Type here to search
27°C Mostly cloudy
ENG
15:19
03-11-2024


github.com/iam-veeramalla/Jenkins-Zero-To-Hero/tree/main/java-maven-sonar-argocd-helm-k8s/spring-boot-app
main
Jenkins-Zero-To-Hero / java-maven-sonar-argocd-helm-k8s / spring-boot-app /
Top
The above maven target stroes the artifacts to the target directory. You can either execute the artifact on your local machine (or) run it as a Docker container.
** Note: To avoid issues with local setup, Java versions and other dependencies, I would recommend the docker way. **
Execute locally (Java 11 needed) and access the application on http://localhost:8080
java -jar target/spring-boot-web.jar
The Docker way
Build the Docker Image
docker build -t ultimate-cicd-pipeline:v1 .
docker run -d -p 8010:8080 -t ultimate-cicd-pipeline:v1
Hurray !! Access the application on http://<ip-address>:8010
Next Steps
Type here to search
27°C Mostly cloudy
ENG
15:19
03-11-2024

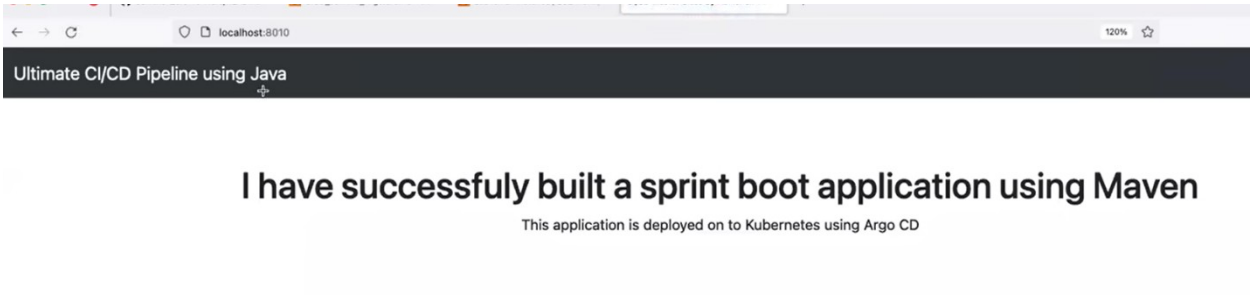



```
→  spring-boot-app git:(main) mvn clean package
[INFO] Scanning for projects...
[INFO]
[INFO] -------------------< com.abhishek:spring-boot-demo >--------------------
[INFO] Building spring-boot-demo 1.0
[INFO]   from pom.xml
[INFO] --------------------------------[ jar ]---------------------------------
[INFO]
[INFO] --- clean:3.1.0:clean (default-clean) @ spring-boot-demo ---
[INFO]
[INFO] --- resources:3.1.0:resources (default-resources) @ spring-boot-demo ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Copying 1 resource
[INFO] Copying 3 resources
[INFO]
[INFO] --- compiler:3.8.0:compile (default-compile) @ spring-boot-demo ---
[INFO] Changes detected - recompiling the module!
```

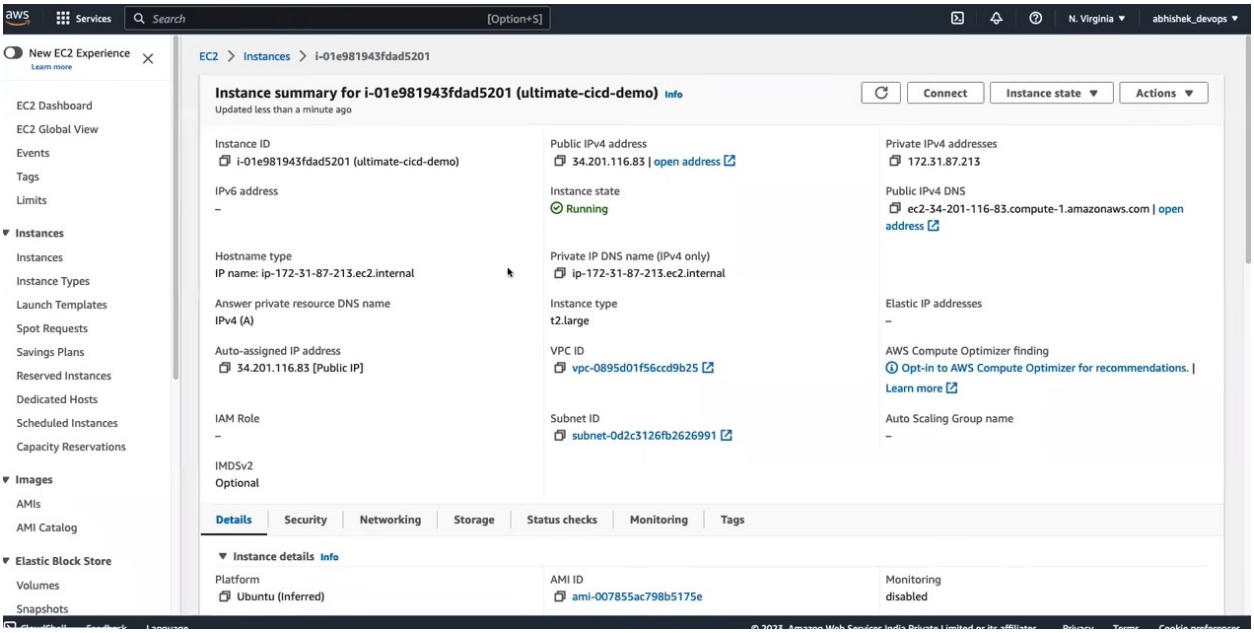
```
→  spring-boot-app git:(main) ✗ docker build -t ultimate-cicd-pipeline:v1 .

[+] Building 6.5s (9/9) FINISHED
 => [internal] load build definition from Dockerfile                                0.1s
 => => transferring dockerfile: 581B                                                0.0s
 => [internal] load .dockerignore                                                   0.0s
 => => transferring context: 2B                                                     0.0s
 => [internal] load metadata for docker.io/adoptopenjdk/openjdk11:alpine-jre        5.0s
 => [auth] adoptopenjdk/openjdk11:pull token for registry-1.docker.io               0.0s
 => [internal] load build context                                                   0.9s
 => => transferring context: 19.07MB                                                0.9s
 => [1/3] FROM docker.io/adoptopenjdk/openjdk11:alpine-jre@sha256:246e1fe322c6c5aca  0.0s
 => CACHED [2/3] WORKDIR /opt/app                                                   0.0s
 => [3/3] COPY target/spring-boot-web.jar app.jar                                   0.2s
 => exporting to image                                                              0.2s
 => => exporting layers                                                             0.2s
 => => writing image sha256:ce6c27fede6033db689dbfad8786890924b92abbcb761e27bd3c594  0.0s
 => => naming to docker.io/library/ultimate-cicd-pipeline:v1                        0.0s
```

```
→  spring-boot-app git:(main) ✗ docker run -d -p 8010:8080 -t ultimate-cicd-pipeline:v1
8ec65093671787350fbcc2c025a4828f692b47698452511eec0e03fcc2fee28a
→  spring-boot-app git:(main) ✗
```

Test @ port 8010



AWS EC2 instance ready .. cp the public ip address + pem file & connect to the server



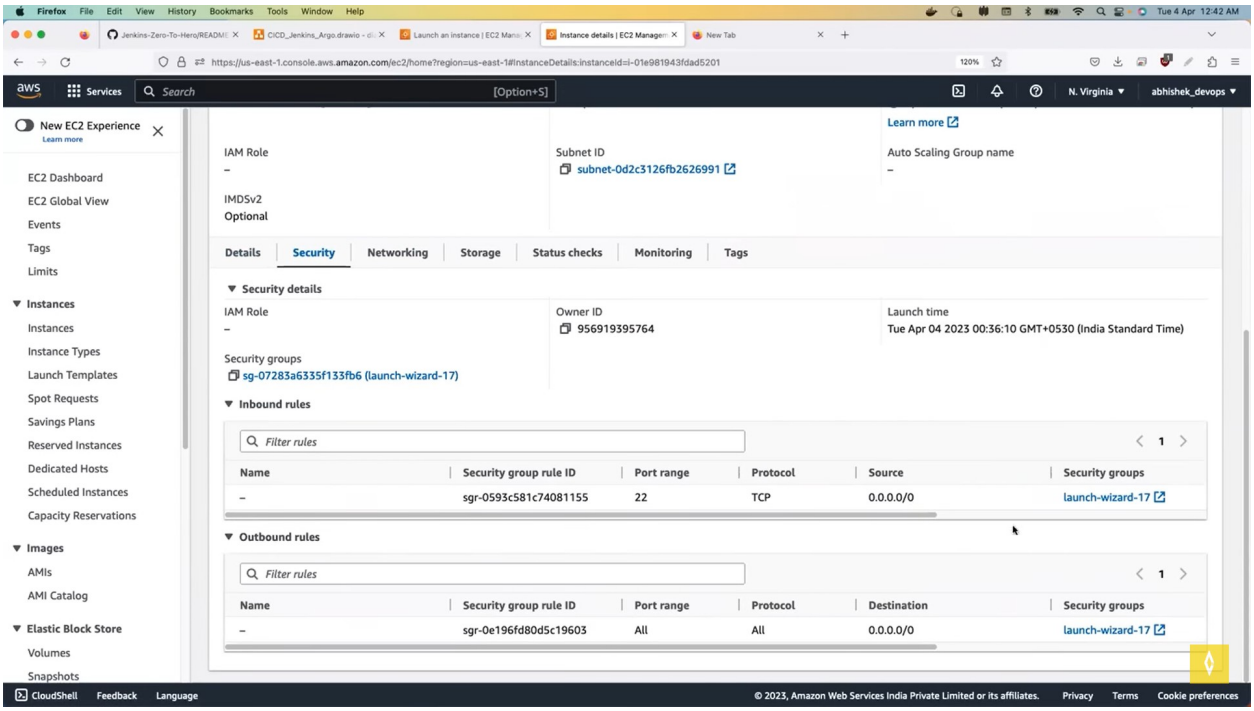
 # update packages + java in sudo

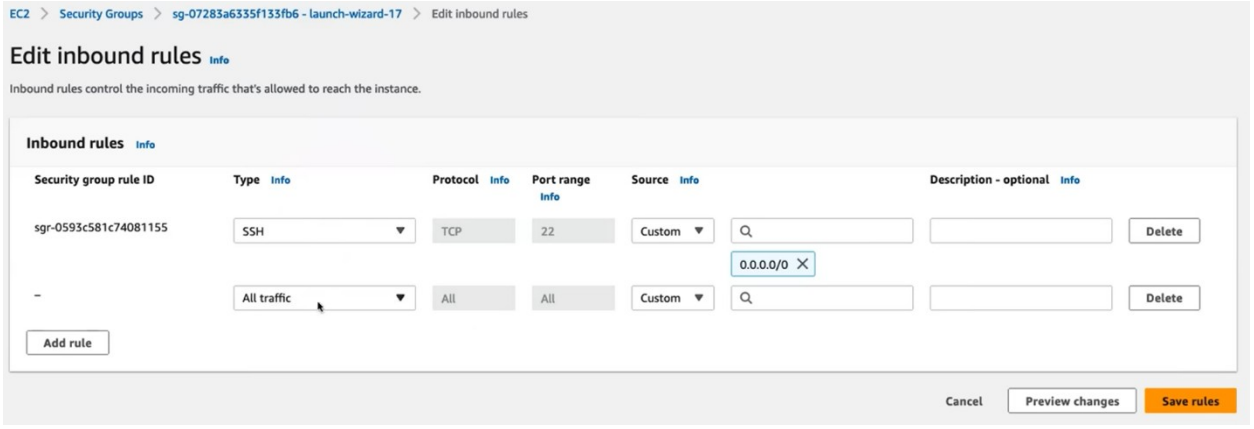
```
ubuntu@ip-172-31-87-213:~$ sudo apt update
sudo apt install openjdk-11-jre
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease [119 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease [108 kB]
Get:4 http://security.ubuntu.com/ubuntu jammy-security InRelease [110 kB]
Get:5 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe amd64 Packages [14.1 MB]
Get:6 http://security.ubuntu.com/ubuntu jammy-security/main amd64 Packages [728 kB]
Get:7 http://security.ubuntu.com/ubuntu jammy-security/main Translation-en [147 kB]
Get:8 http://security.ubuntu.com/ubuntu jammy-security/main amd64 c-n-f Metadata [9020 B]
Get:9 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 Packages [701 kB]
Get:10 http://security.ubuntu.com/ubuntu jammy-security/restricted Translation-en [109 kB]
Get:11 http://security.ubuntu.com/ubuntu jammy-security/restricted amd64 c-n-f Metadata [588 B]
Get:12 http://security.ubuntu.com/ubuntu jammy-security/universe amd64 Packages [715 kB]
Get:13 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy/universe Translation-en [5652 kB]
61% [6 Packages store 0 B] [13 Translation-en 66.9 kB/5652 kB 1%] [12 Packages 327 kB/715 kB 46%]
```

install Jenkins

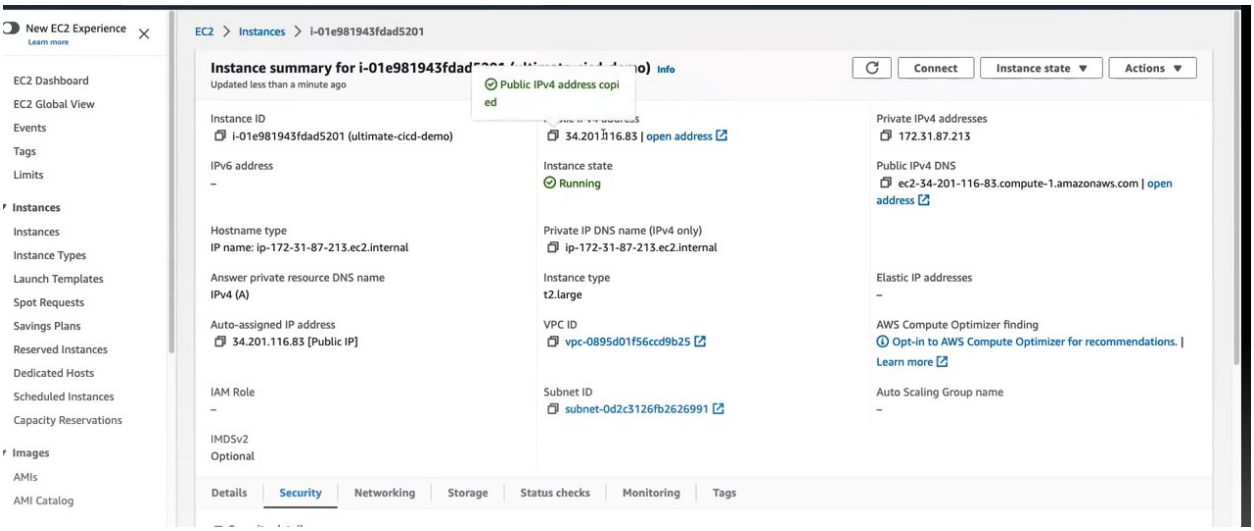
```
ubuntu@ip-172-31-87-213:~$ curl -fsSL https://pkg.jenkins.io/debian/jenkins.io-2023.key | sudo tee \
  /usr/share/keyrings/jenkins-keyring.asc > /dev/null
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
  https://pkg.jenkins.io/debian binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
sudo apt-get update
sudo apt-get install jenkins
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu jammy-security InRelease
Ign:5 https://pkg.jenkins.io/debian binary/ InRelease
Get:6 https://pkg.jenkins.io/debian binary/ Release [2044 B]
Get:7 https://pkg.jenkins.io/debian binary/ Release.gpg [833 B]
Get:8 https://pkg.jenkins.io/debian binary/ Packages [52.5 kB]
Fetched 55.3 kB in 1s (107 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree... 0%
```

Jenkins Server will start @ port 8080 -- @ EC2 instance configure inbound traffic rules

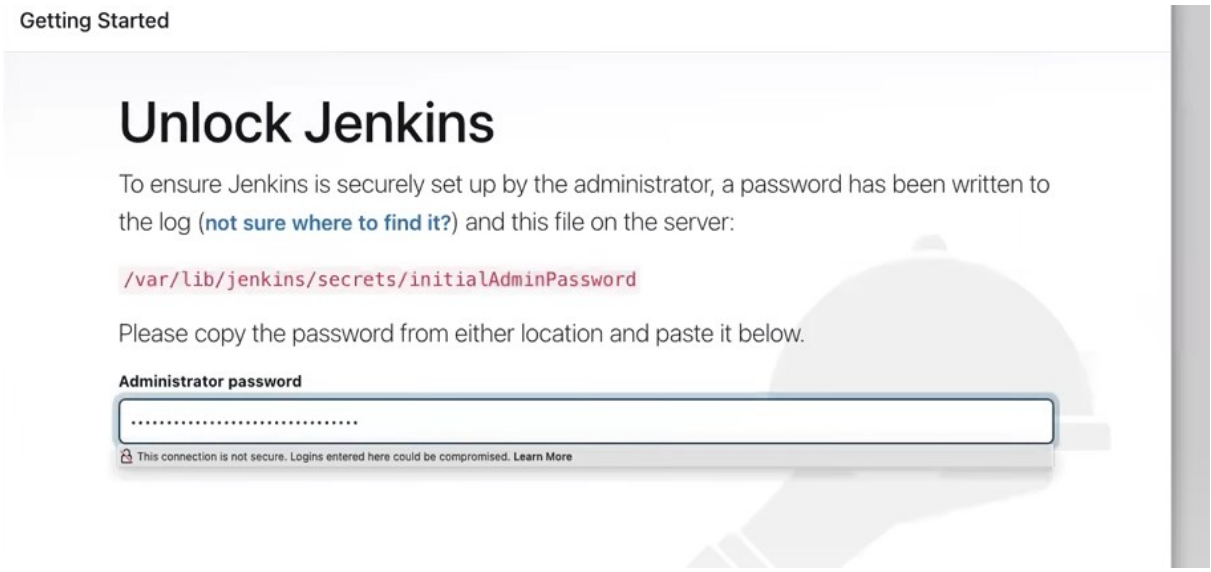


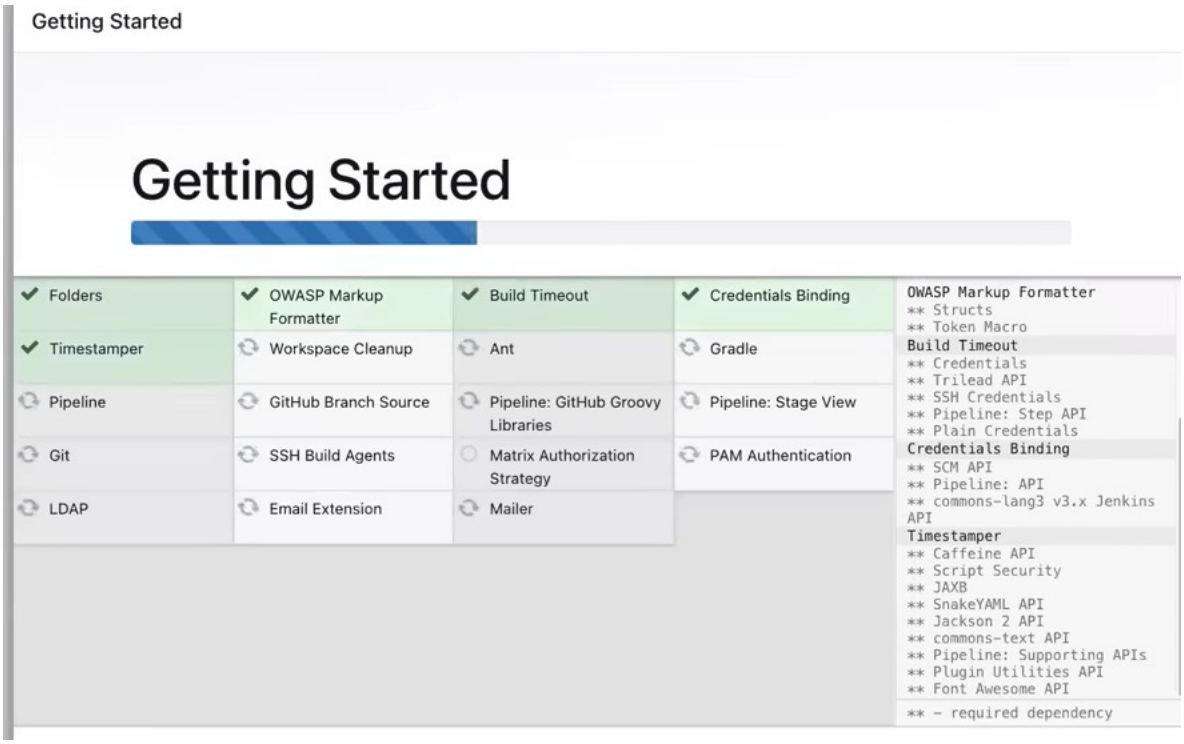


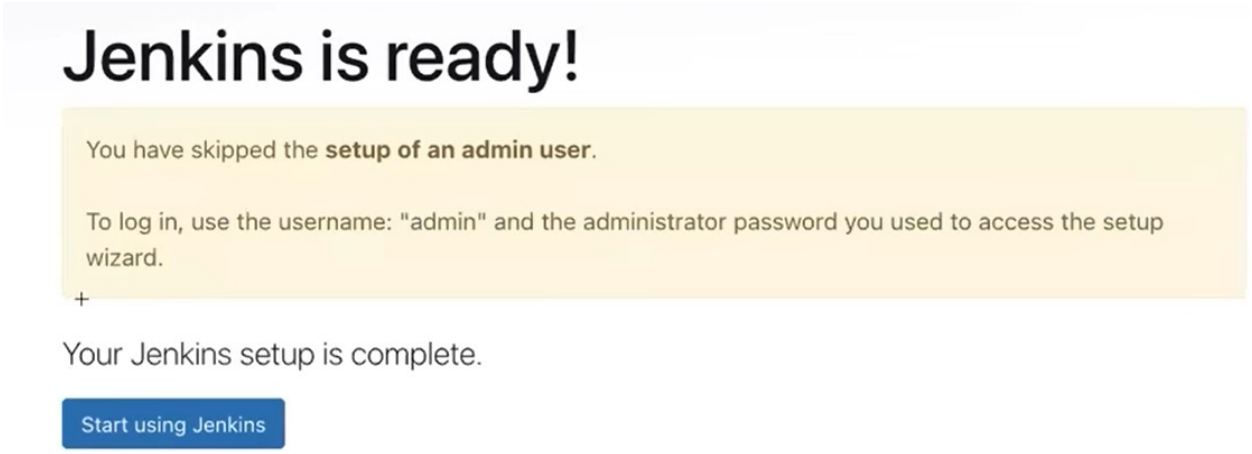
use ip address Public Ec2 instance & check the running Server

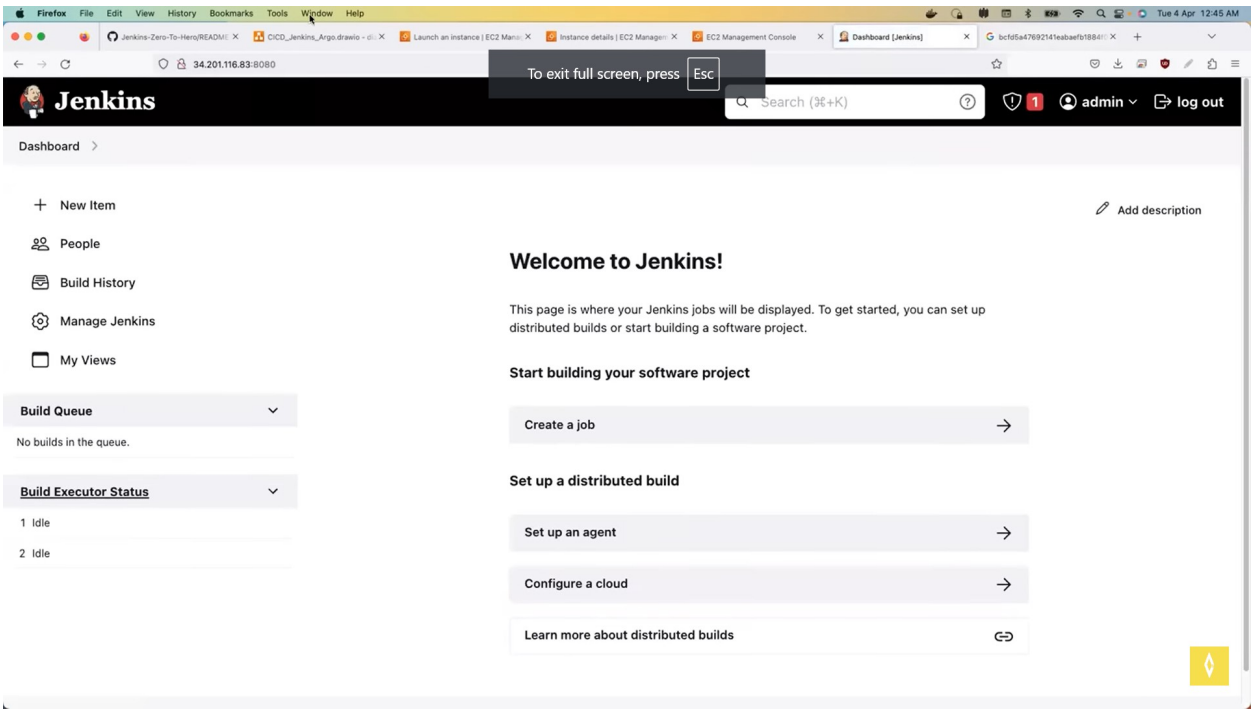


cp the iinitial password

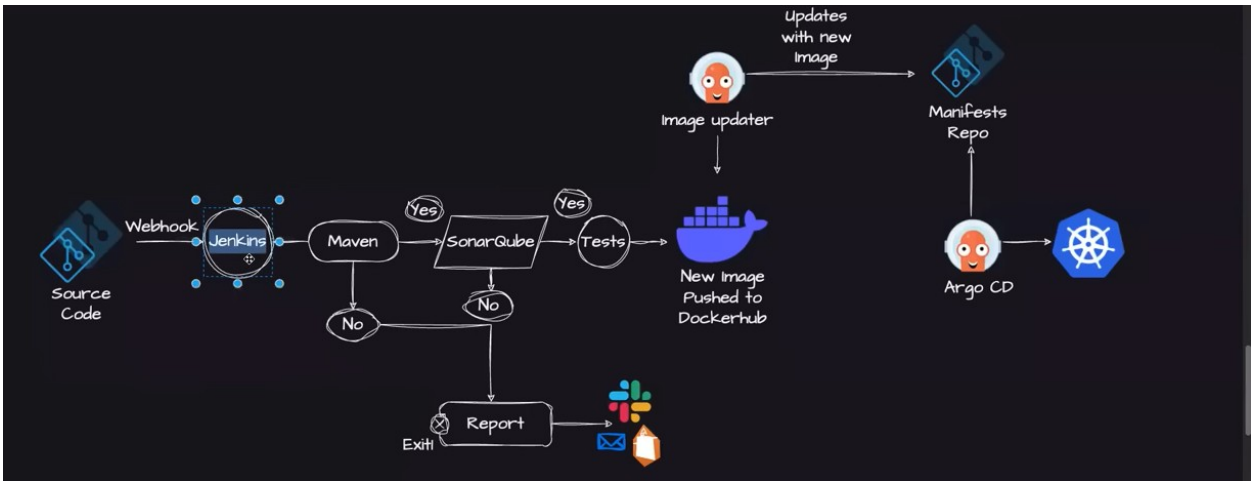

Getting Started
Unlock Jenkins
To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:
/var/lib/jenkins/secrets/initialAdminPassword
Please copy the password from either location and paste it below.
Administrator password
This connection is not secure. Logins entered here could be compromised. Learn More


Getting Started
Getting Started
Folders
OWASP Markup Formatter
Build Timeout
Credentials Binding
Timestamper
Workspace Cleanup
Ant
Gradle
Pipeline
GitHub Branch Source
Pipeline: GitHub Groovy Libraries
Pipeline: Stage View
Git
SSH Build Agents
Matrix Authorization Strategy
PAM Authentication
LDAP
Email Extension
Mailer
OWASP Markup Formatter
** Structs
** Token Macro
Build Timeout
** Credentials
** Trilead API
** SSH Credentials
** Pipeline: Step API
** Plain Credentials
Credentials Binding
** SCM API
** Pipeline: API
** commons-lang3 v3.x Jenkins API
Timestamper
** Caffeine API
** Script Security
** JAXB
** SnakeYAML API
** Jackson 2 API
** commons-text API
** Pipeline: Supporting APIs
** Plugin Utilities API
** Font Awesome API
** - required dependency

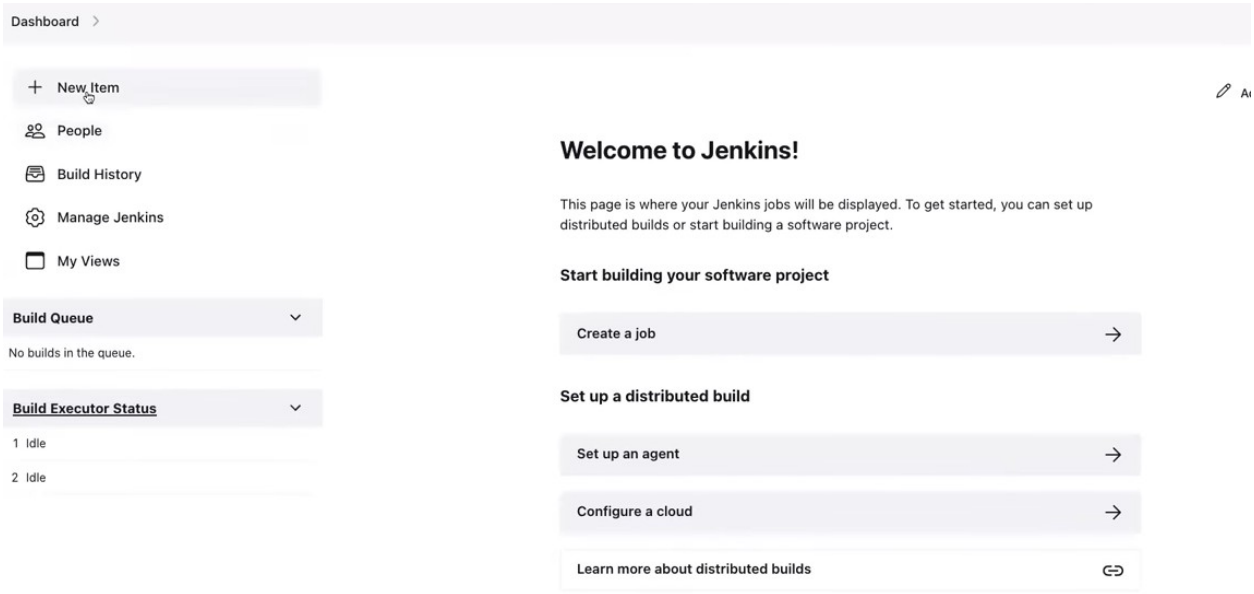

Jenkins is ready!
You have skipped the setup of an admin user.
To log in, use the username: "admin" and the administrator password you used to access the setup wizard.
Your Jenkins setup is complete.
Start using Jenkins

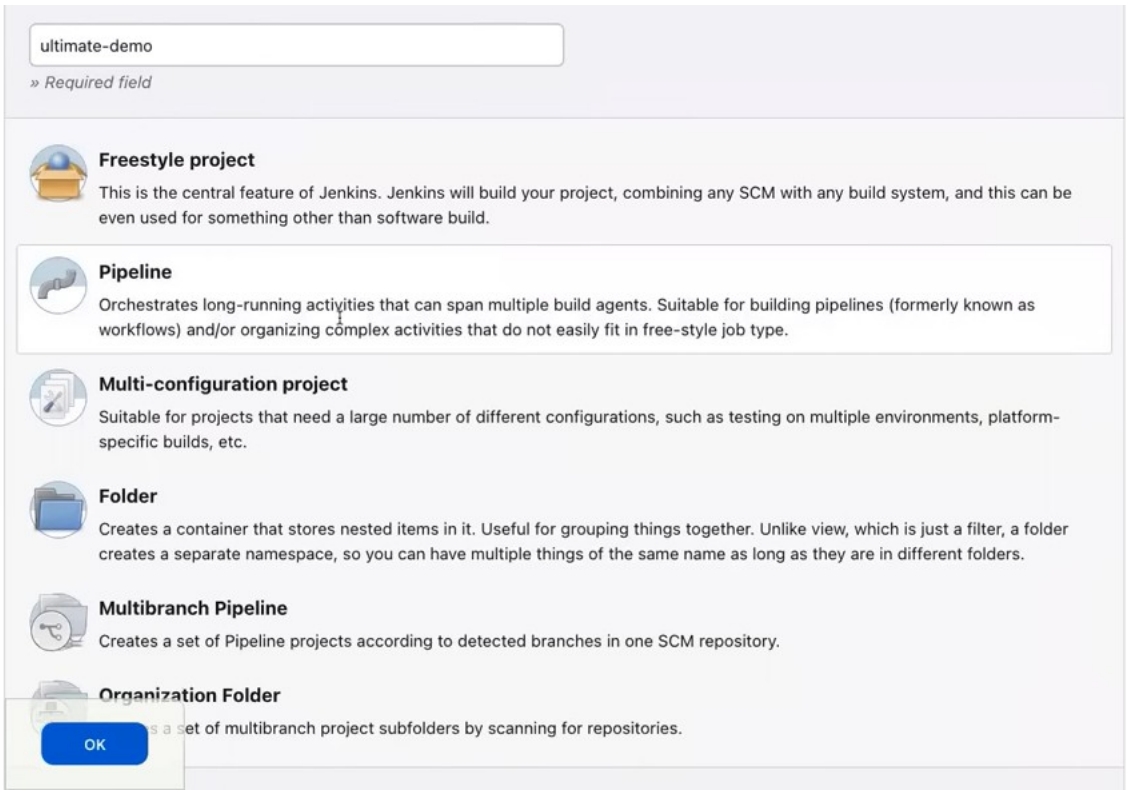

Jenkins
Search (⌘+K)
admin
log out
Dashboard
New Item
People
Build History
Manage Jenkins
My Views
Add description
Welcome to Jenkins!
This page is where your Jenkins jobs will be displayed. To get started, you can set up distributed builds or start building a software project.
Start building your software project
Create a job
Set up a distributed build
Set up an agent
Configure a cloud
Learn more about distributed builds
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle

next Step

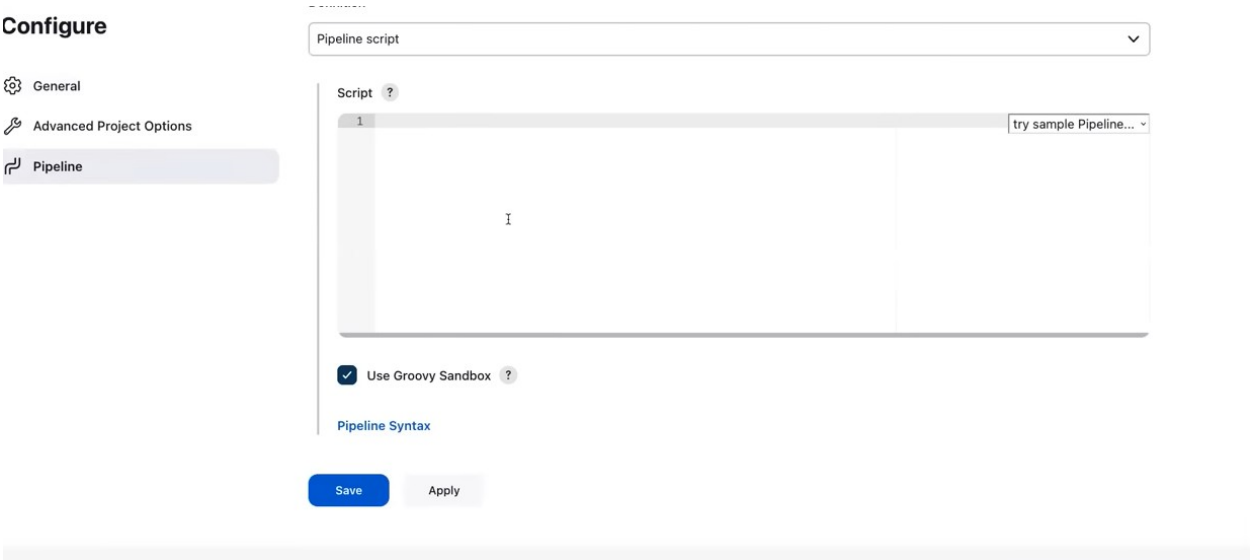


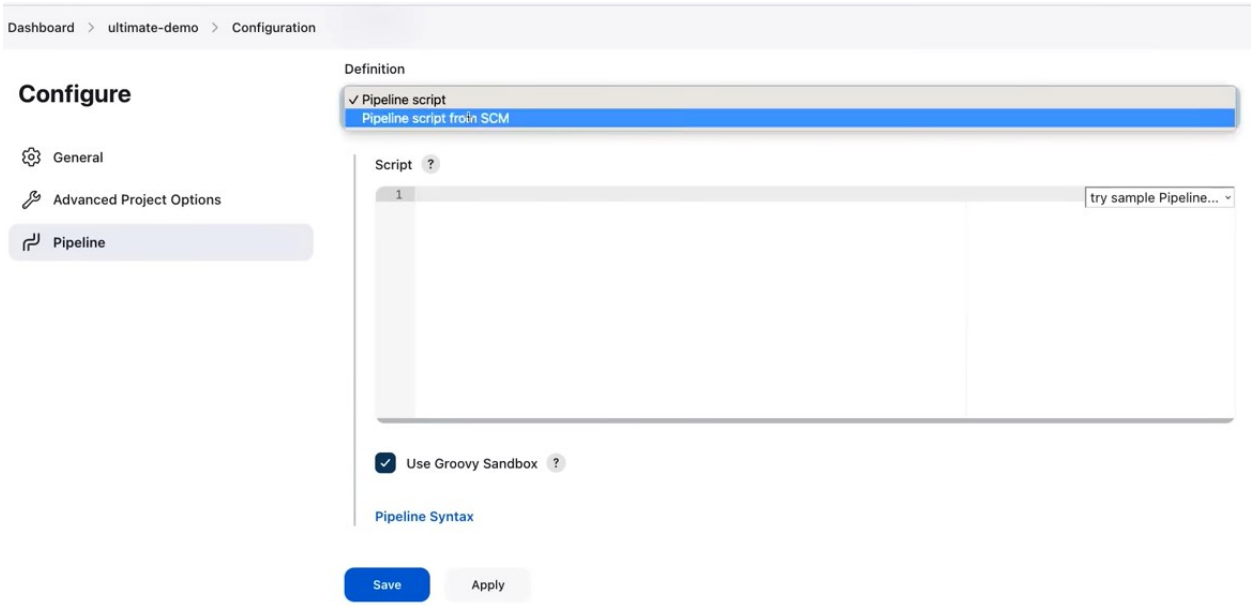
now write a Pipeline using jenkin ui - click on newitem –select - pipeline



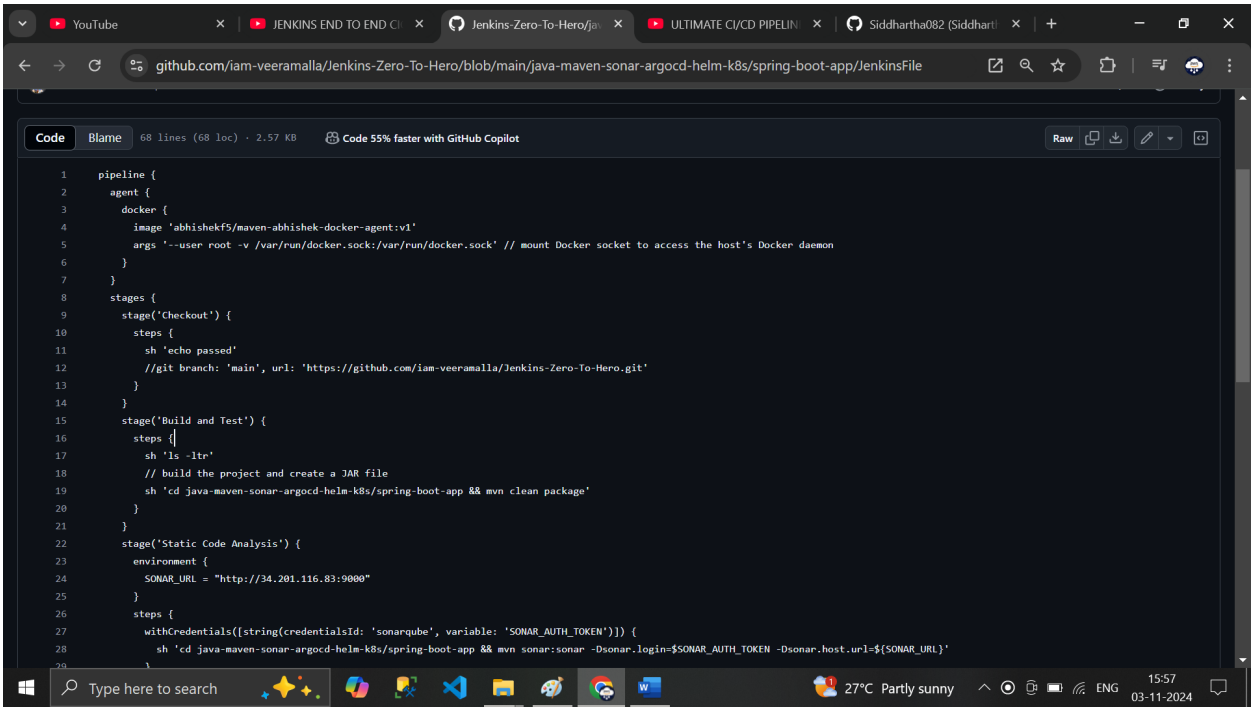


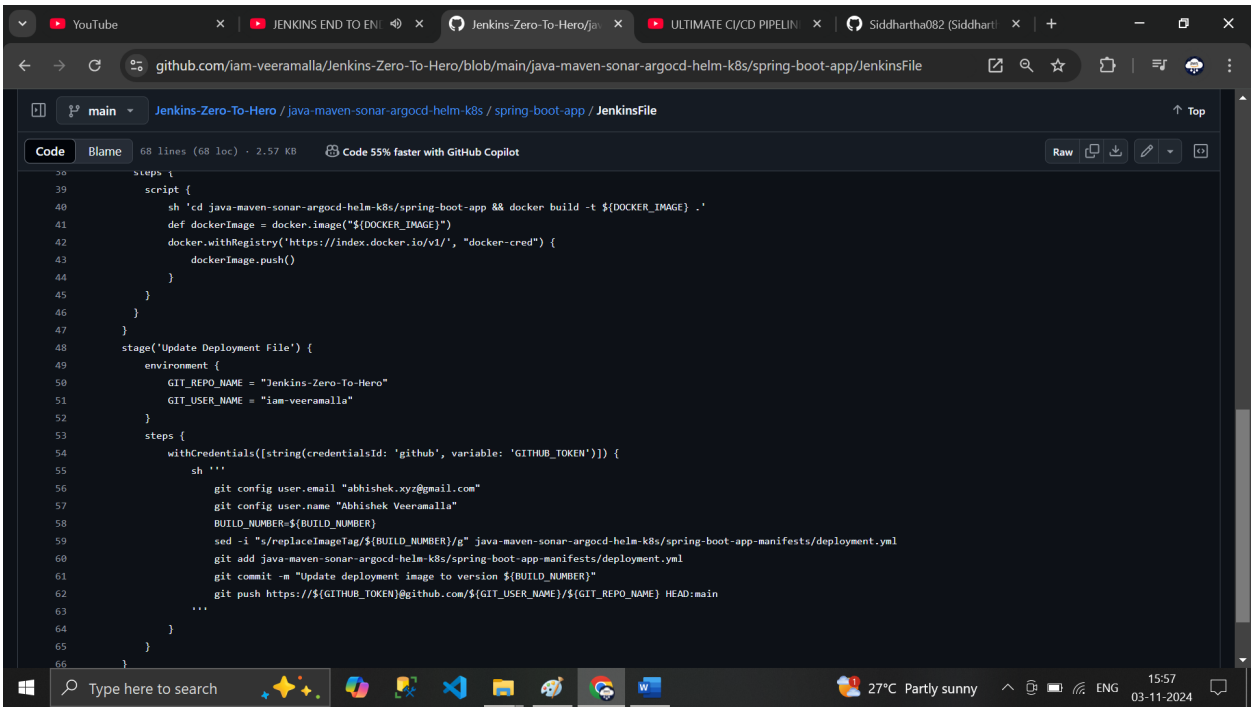
Groovy script or Go to Repository

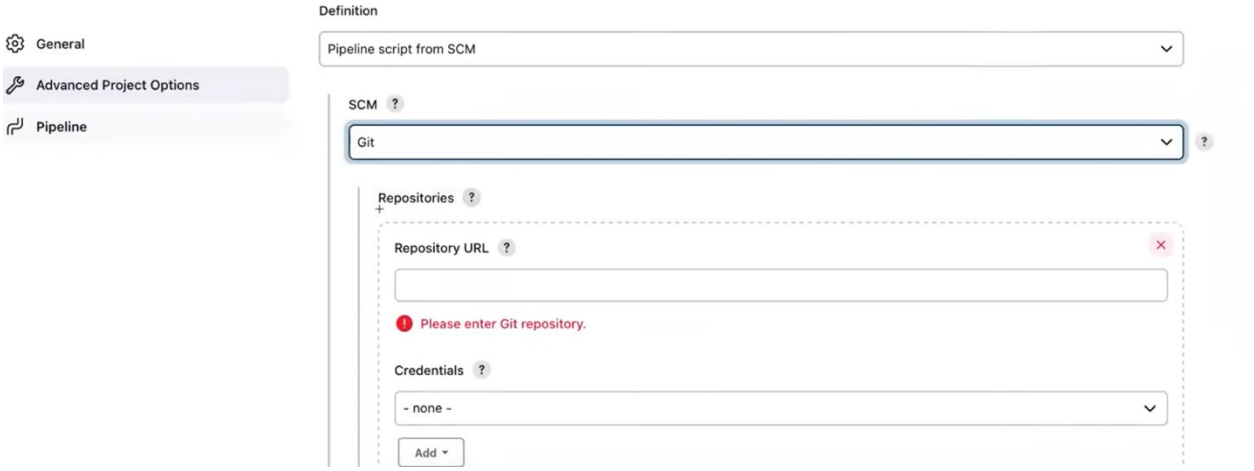


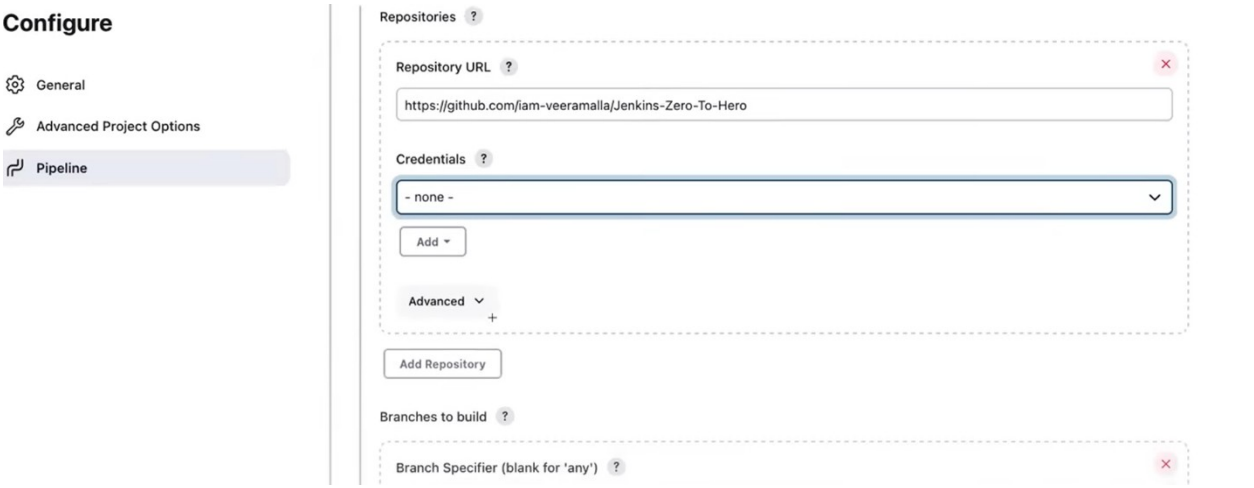


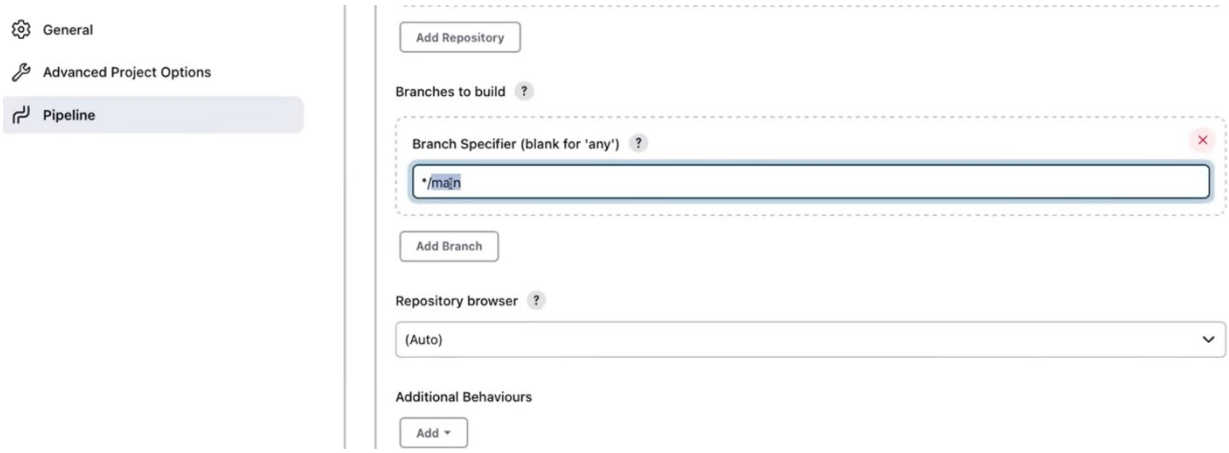
Jenkinsfile

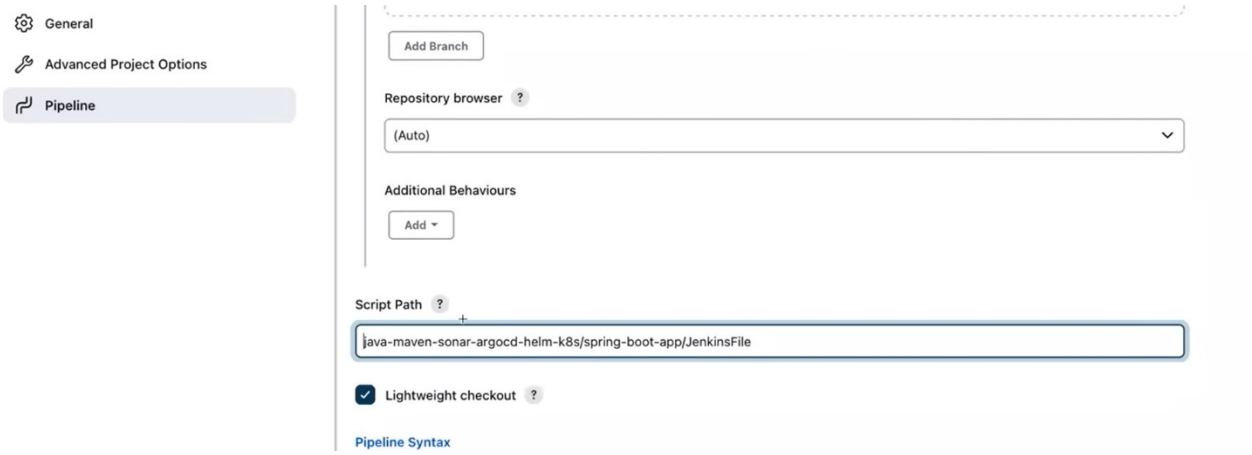


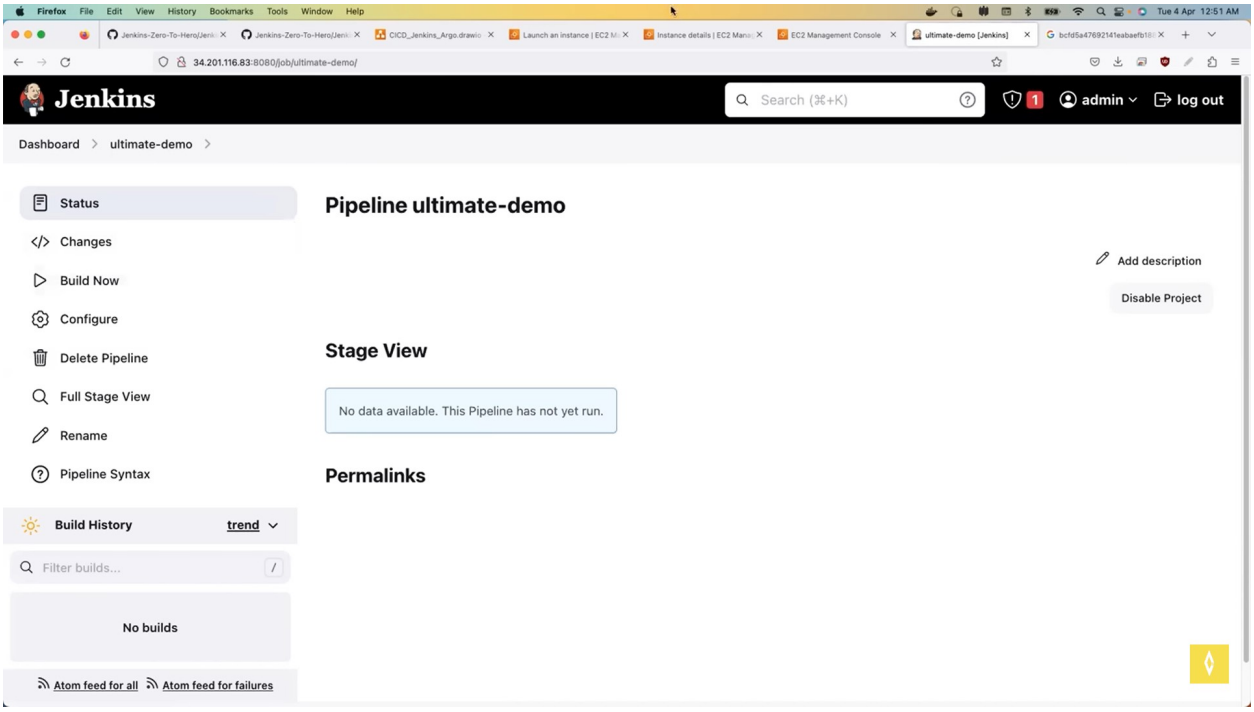

github.com/iam-veeramalla/Jenkins-Zero-To-Hero/blob/main/java-maven-sonar-argocd-helm-k8s/spring-boot-app/JenkinsFile
main
Jenkins-Zero-To-Hero / java-maven-sonar-argocd-helm-k8s / spring-boot-app / JenkinsFile
Code
Blame
68 lines (68 loc) · 2.57 KB
Code 55% faster with GitHub Copilot
Raw
Top
steps {
script {
sh 'cd java-maven-sonar-argocd-helm-k8s/spring-boot-app && docker build -t ${DOCKER_IMAGE} .'
def dockerImage = docker.image("${DOCKER_IMAGE}")
docker.withRegistry('https://index.docker.io/v1/', "docker-cred") {
dockerImage.push()
}
}
}
}
stage('Update Deployment File') {
environment {
GIT_REPO_NAME = "Jenkins-Zero-To-Hero"
GIT_USER_NAME = "iam-veeramalla"
}
steps {
withCredentials([string(credentialsId: 'github', variable: 'GITHUB_TOKEN')]) {
sh '''
git config user.email "abhishek.xyz@gmail.com"
git config user.name "Abhishek Veeramalla"
BUILD_NUMBER=${BUILD_NUMBER}
sed -i "s/replaceImageTag/${BUILD_NUMBER}/g" java-maven-sonar-argocd-helm-k8s/spring-boot-app-manifests/deployment.yml
git add java-maven-sonar-argocd-helm-k8s/spring-boot-app-manifests/deployment.yml
git commit -m "Update deployment image to version ${BUILD_NUMBER}"
git push https://${GITHUB_TOKEN}@github.com/${GIT_USER_NAME}/${GIT_REPO_NAME} HEAD:main
'''
}
}
}
}
Type here to search
27°C Partly sunny
ENG
15:57
03-11-2024


General
Advanced Project Options
Pipeline
Definition
Pipeline script from SCM
SCM
Git
Repositories
Repository URL
Please enter Git repository.
Credentials
- none -
Add


Configure
General
Advanced Project Options
Pipeline
Repositories
Repository URL
https://github.com/iam-veeramalla/Jenkins-Zero-To-Hero
Credentials
- none -
Add
Advanced
Add Repository
Branches to build
Branch Specifier (blank for 'any')


General
Advanced Project Options
Pipeline
Add Repository
Branches to build
Branch Specifier (blank for 'any')
*/main
Add Branch
Repository browser
(Auto)
Additional Behaviours
Add

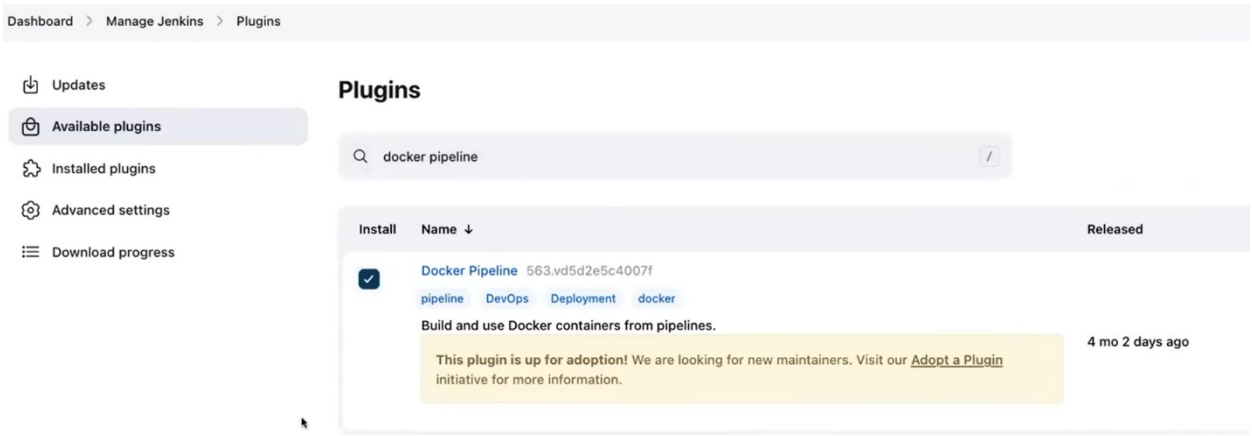

General
Advanced Project Options
Pipeline
Add Branch
Repository browser
(Auto)
Additional Behaviours
Add
Script Path
java-maven-sonar-argocd-helm-k8s/spring-boot-app/JenkinsFile
Lightweight checkout
Pipeline Syntax

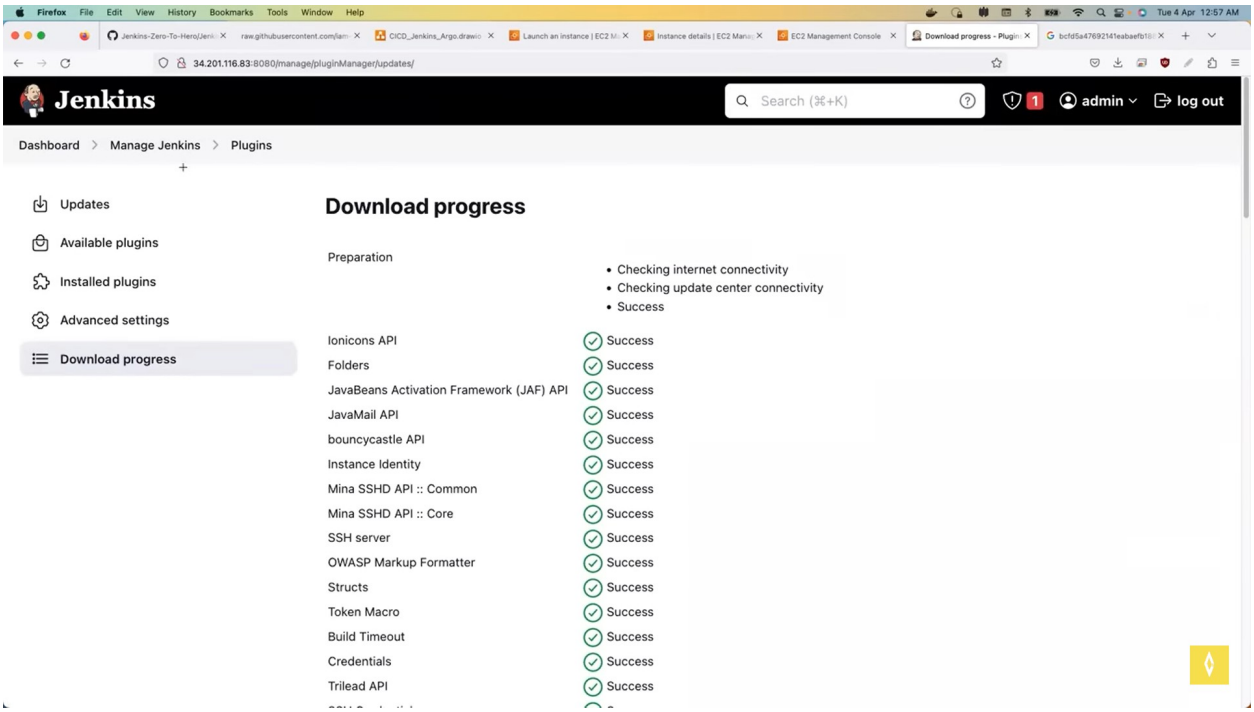

Jenkins
Search (⌘+K)
admin
log out
Dashboard
ultimate-demo
Status
Changes
Build Now
Configure
Delete Pipeline
Full Stage View
Rename
Pipeline Syntax
Build History
trend
Filter builds...
No builds
Atom feed for all
Atom feed for failures
Pipeline ultimate-demo
Add description
Disable Project
Stage View
No data available. This Pipeline has not yet run.
Permalinks

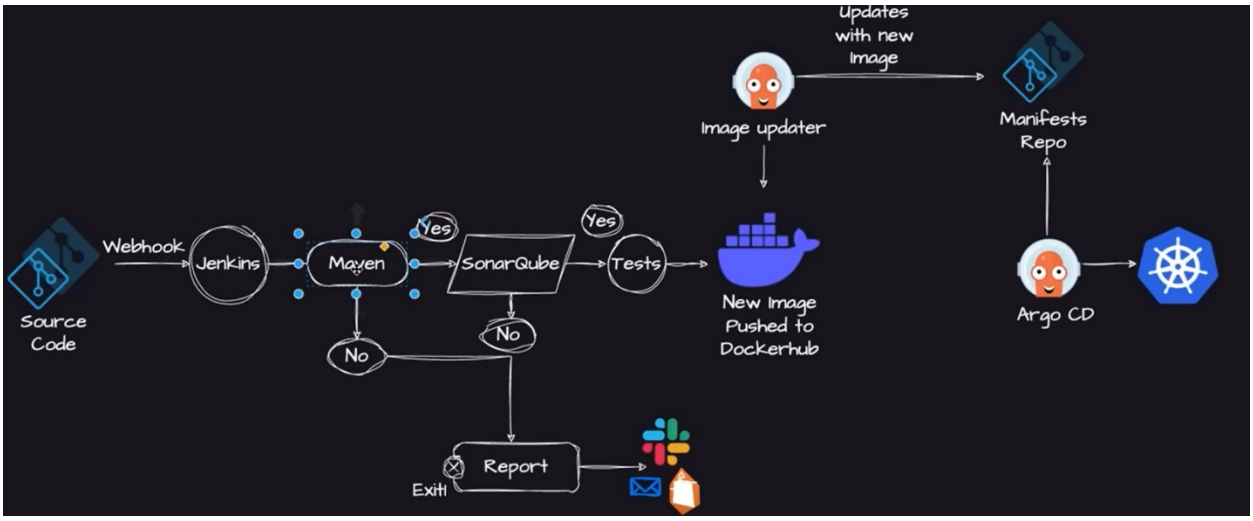
----- install Docker Container in the Jenkins – using Docker Plugins

-- note Docker container default has maven----

-

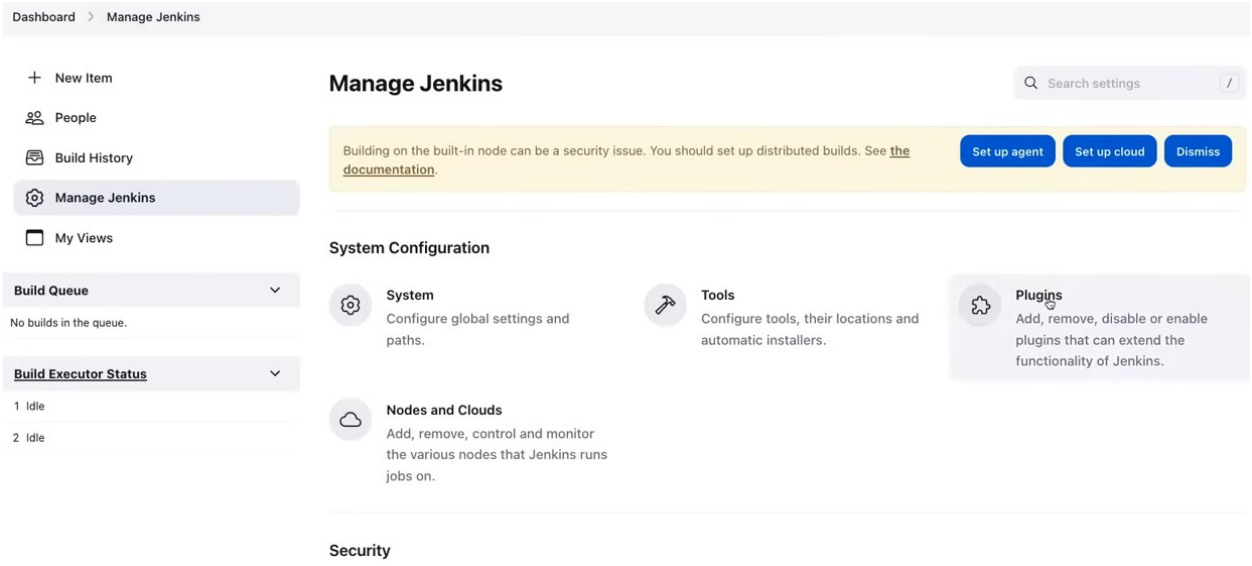


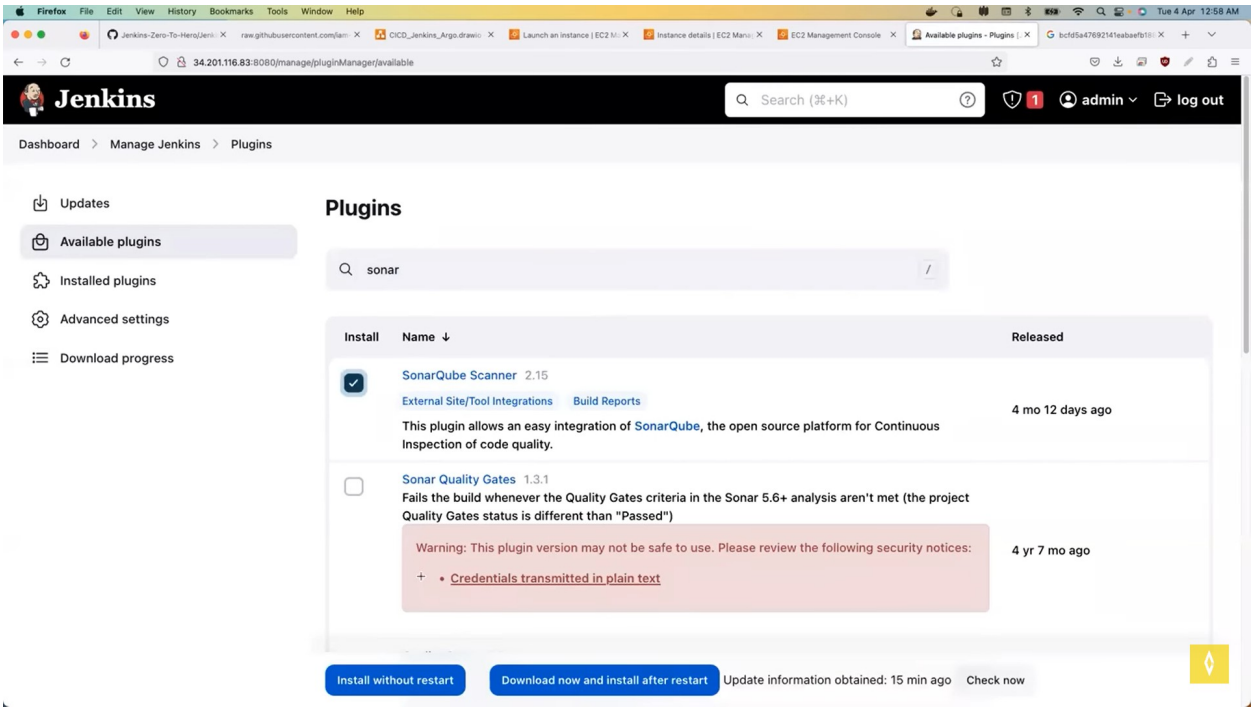


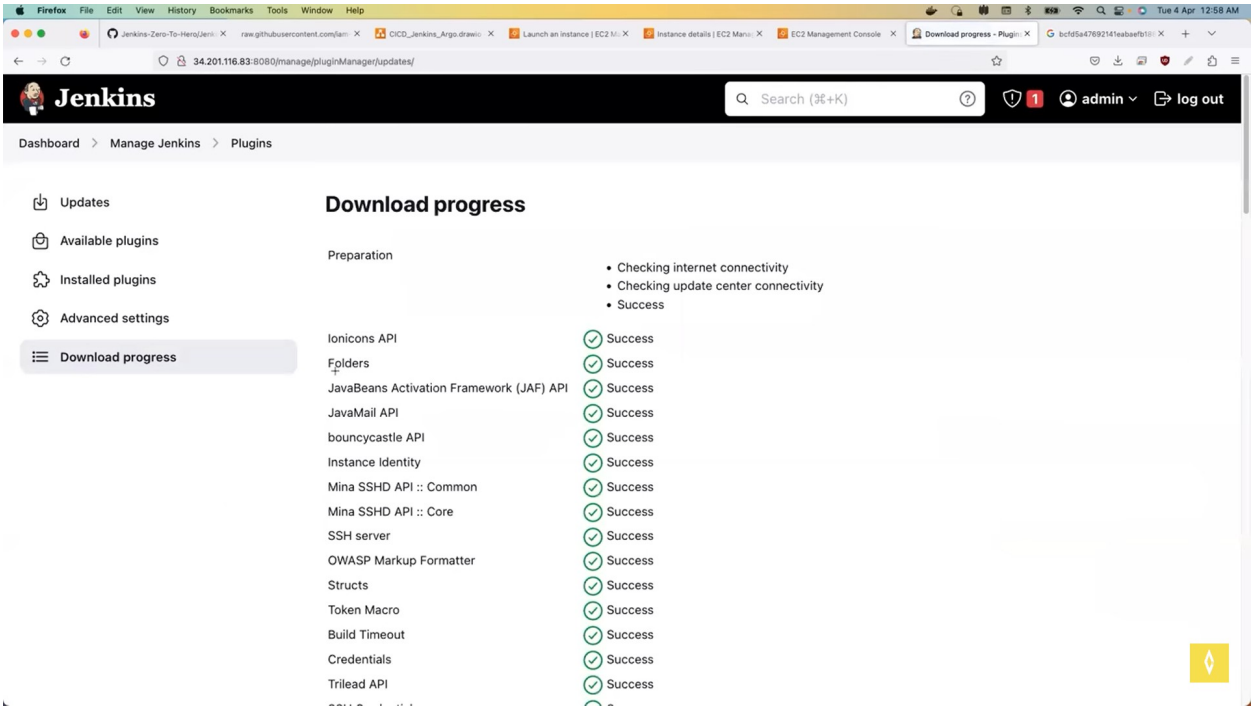


now install SonarQube

Go back to Jenkins – Plugins




Jenkins
Search (⌘+K)
admin
log out
Dashboard > Manage Jenkins > Plugins
Updates
Available plugins
Installed plugins
Advanced settings
Download progress
Plugins
sonar
Install
Name ↓
Released
SonarQube Scanner 2.15
External Site/Tool Integrations
Build Reports
This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.
4 mo 12 days ago
Sonar Quality Gates 1.3.1
Fails the build whenever the Quality Gates criteria in the Sonar 5.6+ analysis aren't met (the project Quality Gates status is different than "Passed")
Warning: This plugin version may not be safe to use. Please review the following security notices:
• Credentials transmitted in plain text
4 yr 7 mo ago
Install without restart
Download now and install after restart
Update information obtained: 15 min ago
Check now


Jenkins
Search (⌘+K)
admin
log out
Dashboard > Manage Jenkins > Plugins
Updates
Available plugins
Installed plugins
Advanced settings
Download progress
Download progress
Preparation
• Checking internet connectivity
• Checking update center connectivity
• Success
Ionicons API Success
Folders Success
JavaBeans Activation Framework (JAF) API Success
JavaMail API Success
bouncycastle API Success
Instance Identity Success
Mina SSHD API :: Common Success
Mina SSHD API :: Core Success
SSH server Success
OWASP Markup Formatter Success
Structs Success
Token Macro Success
Build Timeout Success
Credentials Success
Trilead API Success

Next Steps

Configure a Sonar Server locally

```
apt install unzip
adduser sonarqube
wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-9.4.0.54424.zip
unzip *
chmod -R 755 /home/sonarqube/sonarqube-9.4.0.54424
chown -R sonarqube:sonarqube /home/sonarqube/sonarqube-9.4.0.54424
cd sonarqube-9.4.0.54424/bin/linux-x86-64/
./sonar.sh start
```

Hurray !! Now you can access the `SonarQube Server` on `http://<ip-address>:9000`

```
ubuntu@ip-172-31-87-213:~$ adduser sonarqube

adduser: Only root may add a user or group to the system.
ubuntu@ip-172-31-87-213:~$ sudo su -
root@ip-172-31-87-213:~# adduser sonarqube
 ^C
root@ip-172-31-87-213:~# adduser sonarqube
Adding user `sonarqube' ...
Adding new group `sonarqube' (1001) ...
Adding new user `sonarqube' (1001) with group `sonarqube' ...
Creating home directory `/home/sonarqube' ...
Copying files from `/etc/skel' ...
New password:
Retype new password:
passwd: password updated successfully
Changing the user information for sonarqube
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] y
root@ip-172-31-87-213:~#
```

```
root@ip-172-31-87-213:~# sudo su - sonarqube
sonarqube@ip-172-31-87-213:~$
```

Download sonar Binary

```
root@ip-172-31-87-213:~# sudo su - sonarqube
sonarqube@ip-172-31-87-213:~$ wget https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-9.4.0.54424.zip
--2023-04-03 19:30:59--  https://binaries.sonarsource.com/Distribution/sonarqube/sonarqube-9.4.0.54424.zip
Resolving binaries.sonarsource.com (binaries.sonarsource.com)... 18.160.10.89, 18.160.10.93, 18.160.10.111, ...
Connecting to binaries.sonarsource.com (binaries.sonarsource.com)|18.160.10.89|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 287666040 (274M) [binary/octet-stream]
Saving to: 'sonarqube-9.4.0.54424.zip'

sonarqube-9.4.0.54424.zip   100%[=========================================>] 274.34M   102MB/s    in 2.7s

2023-04-03 19:31:02 (102 MB/s) - 'sonarqube-9.4.0.54424.zip' saved [287666040/287666040]

sonarqube@ip-172-31-87-213:~$
```

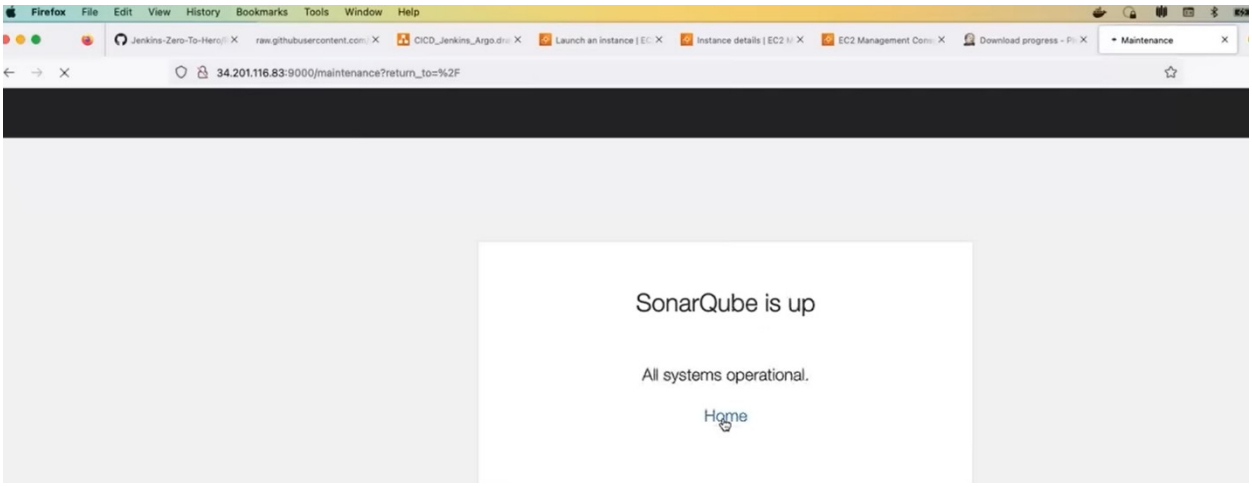
```
sonarqube@ip-172-31-87-213:~$ unzip *
Command 'unzip' not found, but can be installed with:
apt install unzip
Please ask your administrator.
sonarqube@ip-172-31-87-213:~$
logout
root@ip-172-31-87-213:~# apt install unzip
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
Suggested packages:
  zip
The following NEW packages will be installed:
  unzip
0 upgraded, 1 newly installed, 0 to remove and 6 not upgraded.
Need to get 174 kB of archives.
After this operation, 385 kB of additional disk space will be used.
Get:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu jammy-updates/main amd64 unzip amd64 6.0-26ubuntu3.1 [174 kB]
Fetched 174 kB in 0s (9201 kB/s)
```

```
No user sessions are running outdated binaries.

No VM guests are running outdated hypervisor (qemu) binaries on this host.
root@ip-172-31-87-213:~# sudo su -
root@ip-172-31-87-213:~# sudo su - sonarqube
sonarqube@ip-172-31-87-213:~$ ls
sonarqube-9.4.0.54424.zip
sonarqube@ip-172-31-87-213:~$ unzip *
```

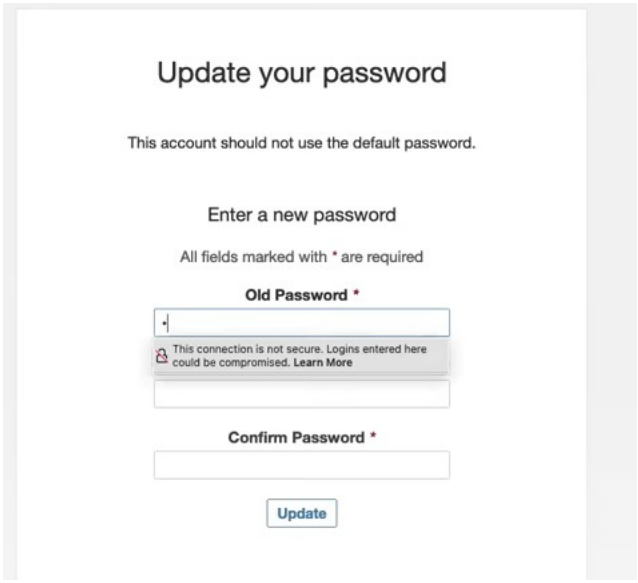
```
 creating: sonarqube-9.4.0.54424/elasticsearch/
 creating: sonarqube-9.4.0.54424/elasticsearch/lib/
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/elasticsearch-7.17.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/java-version-checker-7.17.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/elasticsearch-launchers-7.17.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/elasticsearch-x-content-7.17.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/elasticsearch-lz4-7.17.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/elasticsearch-cli-7.17.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/elasticsearch-core-7.17.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/elasticsearch-secure-sm-7.17.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/elasticsearch-geo-7.17.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/lucene-core-8.11.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/lucene-analyzers-common-8.11.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/lucene-backward-codecs-8.11.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/lucene-grouping-8.11.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/lucene-highlighter-8.11.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/lucene-join-8.11.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/lucene-memory-8.11.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/lucene-misc-8.11.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/lucene-queries-8.11.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/lucene-queryparser-8.11.1.jar
inflating: sonarqube-9.4.0.54424/elasticsearch/lib/lucene-sandbox-8.11.1.jar
```

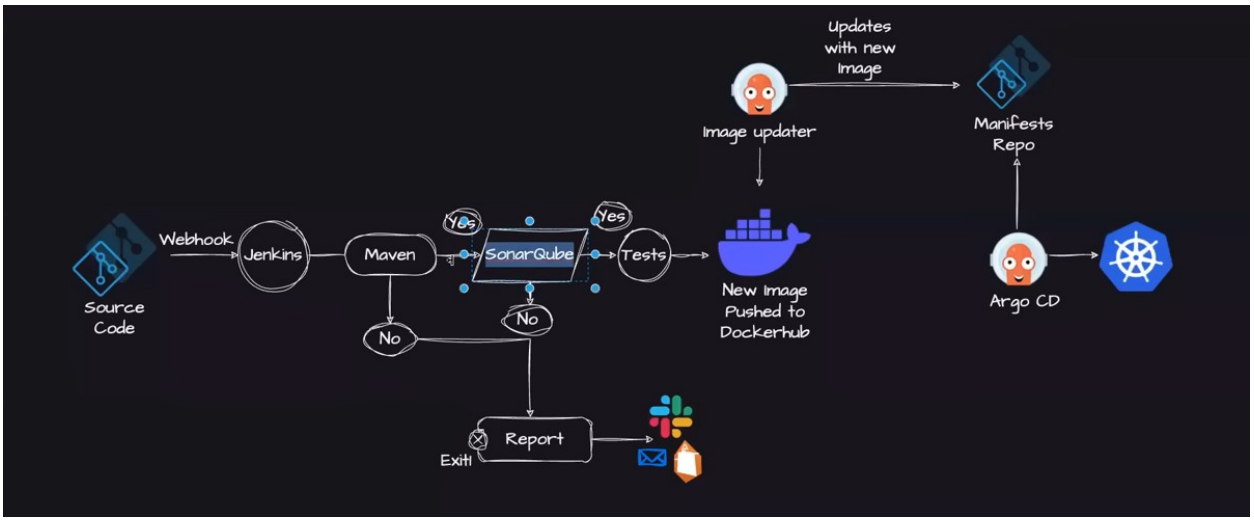
```
sonarqube@ip-172-31-87-213:~$ ls
sonarqube-9.4.0.54424  sonarqube-9.4.0.54424.zip
sonarqube@ip-172-31-87-213:~$ cd sonarqube-9.4.0.54424
sonarqube@ip-172-31-87-213:~/sonarqube-9.4.0.54424$ cd ..
sonarqube@ip-172-31-87-213:~$ chmod -R 755 /home/sonarqube/sonarqube-9.4.0.54424
sonarqube@ip-172-31-87-213:~$ chown -R sonarqube:sonarqube /home/sonarqube/sonarqube-9.4.0.54424
sonarqube@ip-172-31-87-213:~$ cd sonarqube-9.4.0.54424/bin/
jsw-license/         linux-x86-64/        macosx-universal-64/ windows-x86-64/
sonarqube@ip-172-31-87-213:~$ cd sonarqube-9.4.0.54424/bin/linux-x86-64/
sonarqube@ip-172-31-87-213:~/sonarqube-9.4.0.54424/bin/linux-x86-64$ ls
lib  sonar.sh  wrapper
sonarqube@ip-172-31-87-213:~/sonarqube-9.4.0.54424/bin/linux-x86-64$ ./sonar.sh start
Starting SonarQube...
Started SonarQube.
sonarqube@ip-172-31-87-213:~/sonarqube-9.4.0.54424/bin/linux-x86-64$
```

 # note Sonar Server Starts @ Port 9000

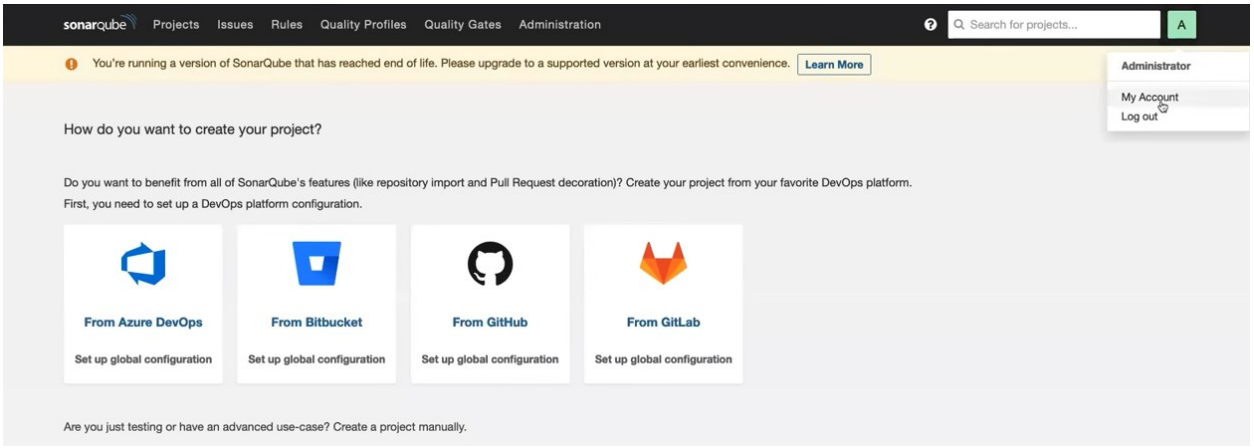


 # First time Login + password – is Admin.. later Change the password

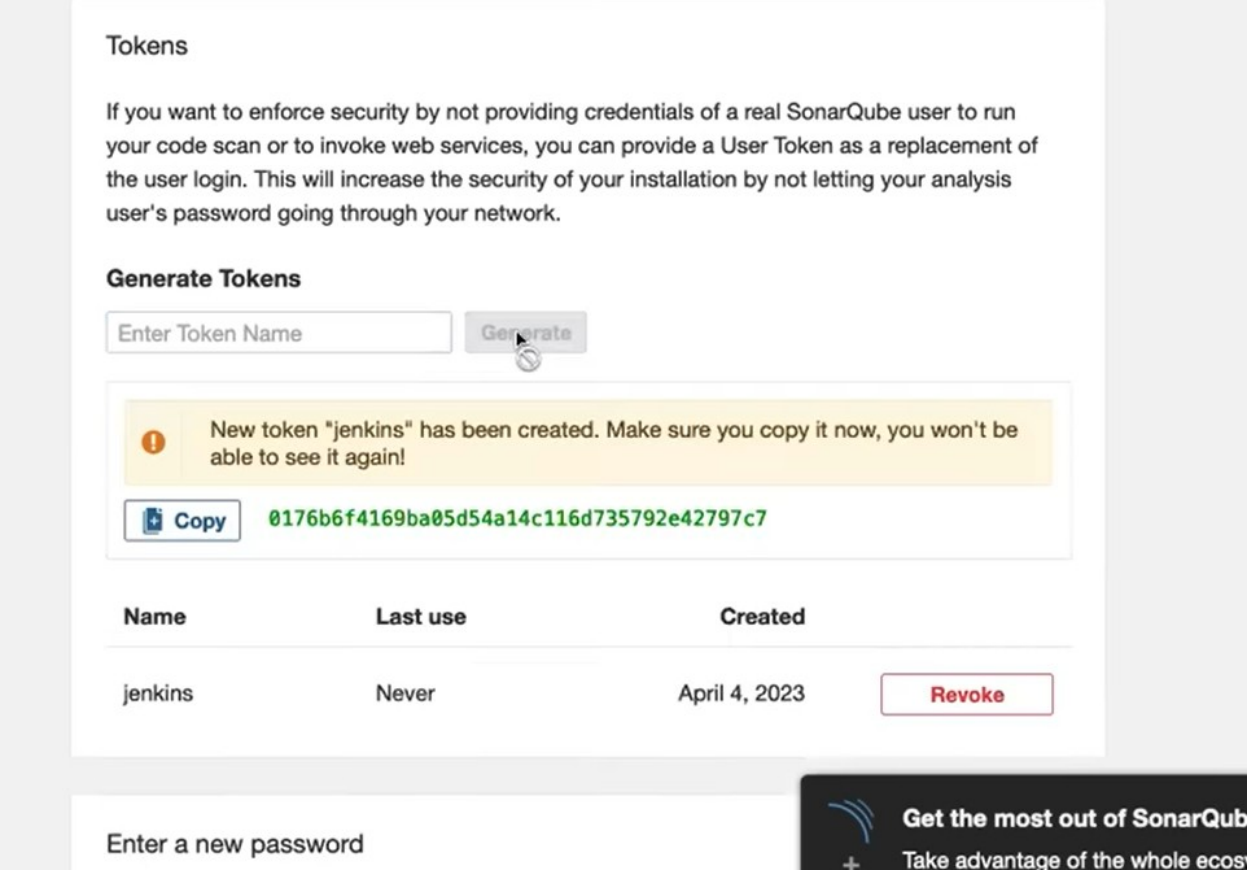




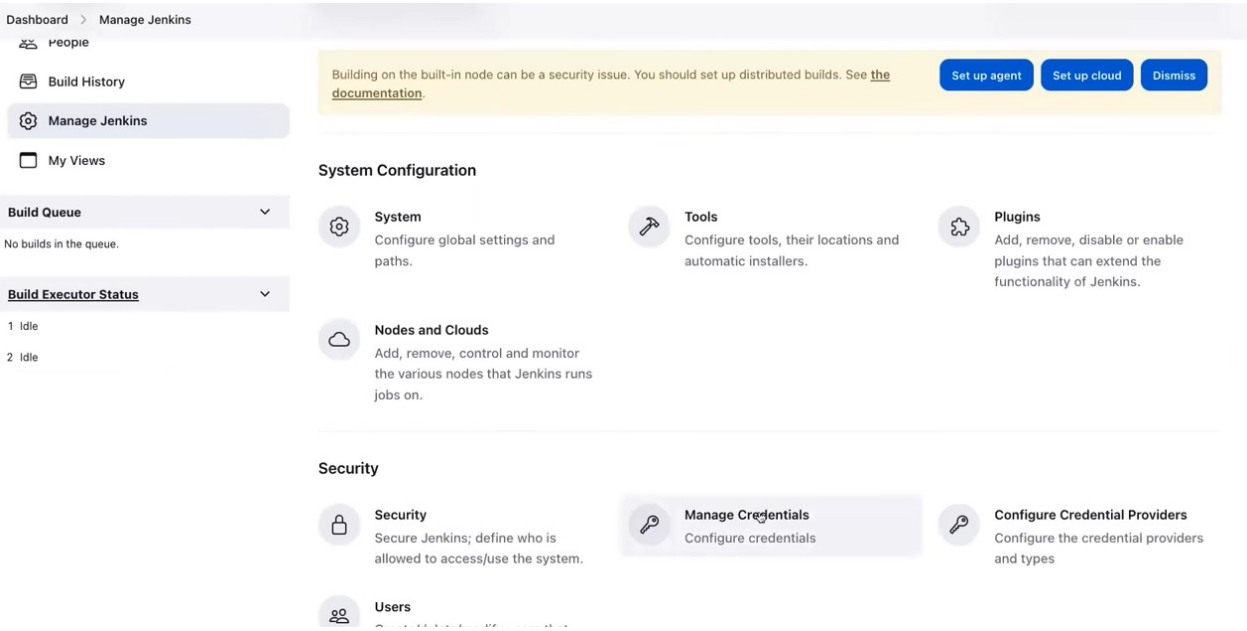
 # now Jenkins would talk / Connect to SonarCube.. – Go to my Account

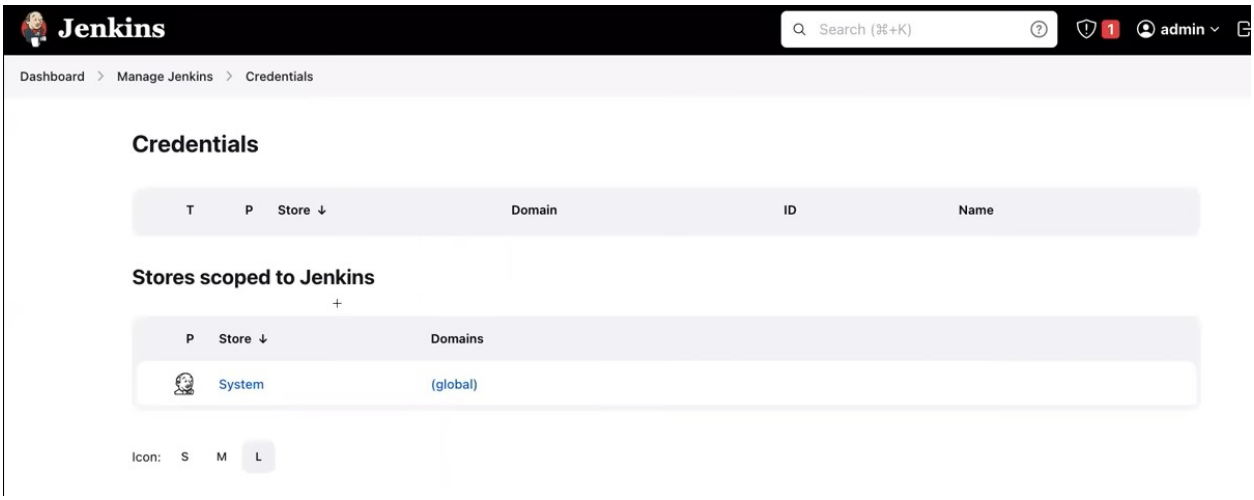


 # Click on Security – Generate New Tokens

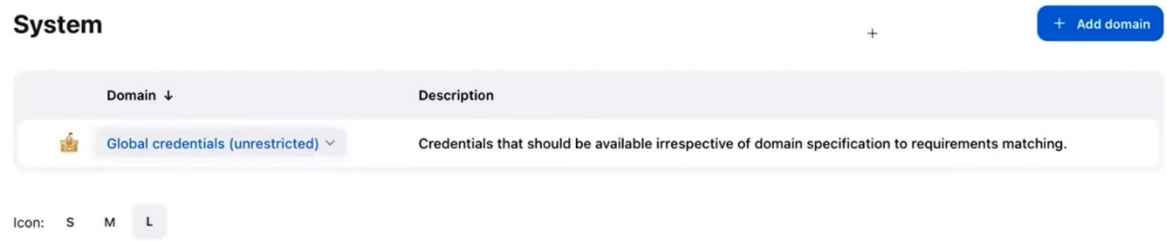


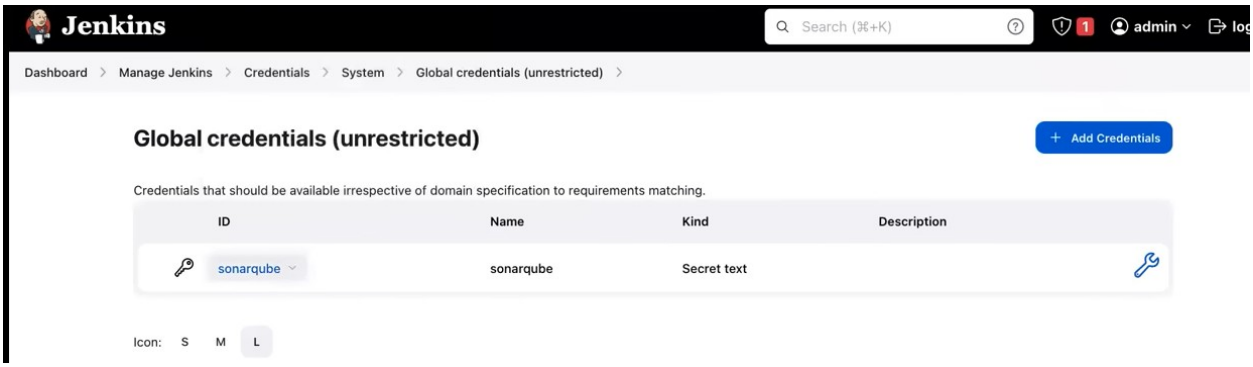
 # Cp the Token ID .. Go to Jenkins .. click on Manage Jenkins – Click on Manage Credientials



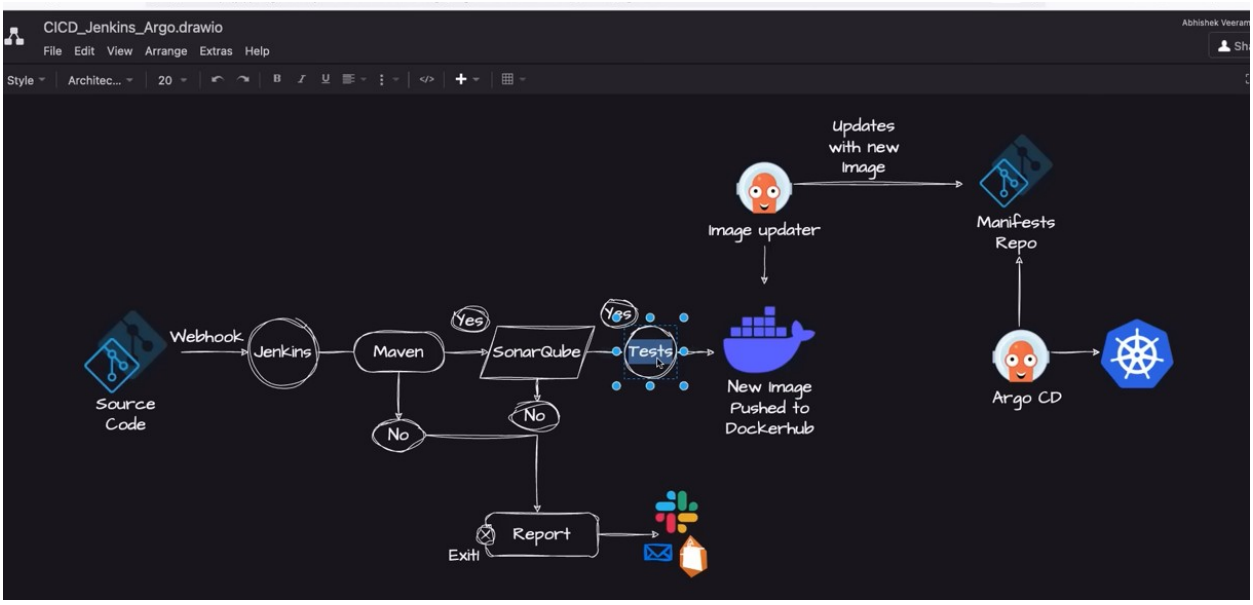


 # Click on System





 # Finally SonarQube Confirguration Completed



 # now install Docker in EC2 Instance – root user

```
sonarqube@ip-172-31-87-213:~/sonarqube-9.4.0.54424/bin/linux-x86-64$ sudo su -
[sudo] password for sonarqube:
sudo: a password is required
sonarqube@ip-172-31-87-213:~/sonarqube-9.4.0.54424/bin/linux-x86-64$
logout
root@ip-172-31-87-213:~# cl
```

Docker Slave Configuration

Run the below command to Install Docker

```
sudo apt update
sudo apt install docker.io
```

Grant Jenkins user and Ubuntu user permission to docker deamon.

```
sudo su -
usermod -aG docker jenkins
usermod -aG docker ubuntu
systemctl restart docker
```

Once you are done with the above steps, it is better to restart Jenkins.

```
http://<ec2-instance-public-ip>:8080/restart
```

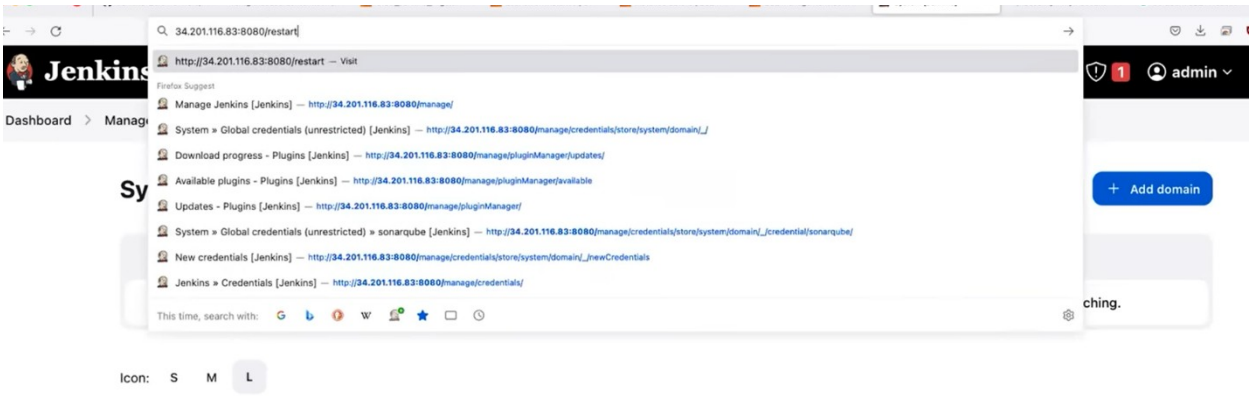
The docker agent configuration is now successful.

```
root@ip-172-31-87-213:~# sudo apt install docker.io
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-doc rinse zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io pigz runc ubuntu-fan
0 upgraded, 8 newly installed, 0 to remove and 6 not upgraded.
Need to get 72.4 MB of archives.
After this operation, 287 MB of additional disk space will be used.
Do you want to continue? [Y/n]
```

 # grant Permissions

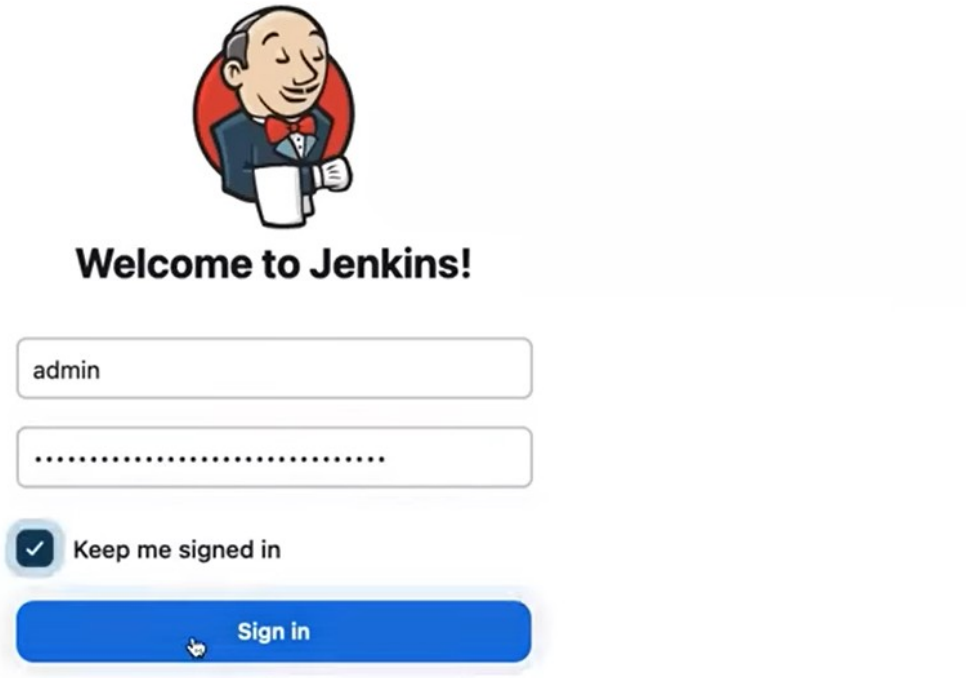
```
root@ip-172-31-87-213:~# usermod -aG docker jenkins
usermod -aG docker ubuntu
systemctl restart docker
```

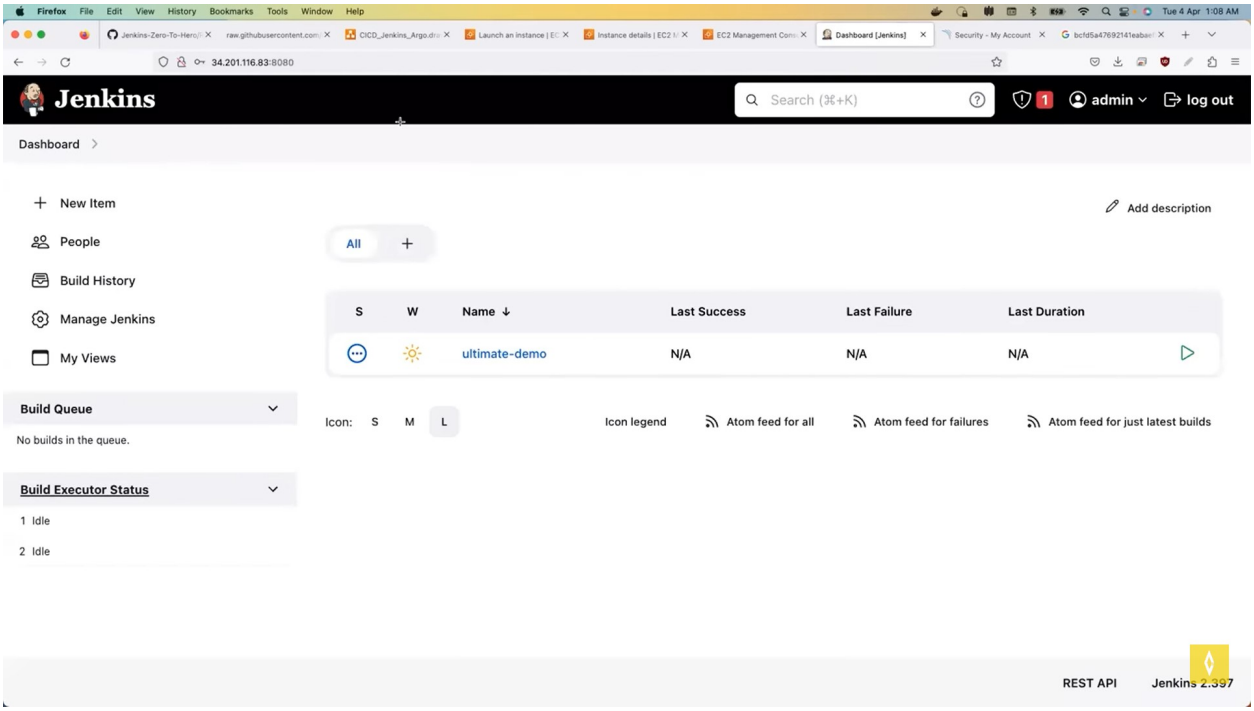
 # Restart - Jenkins



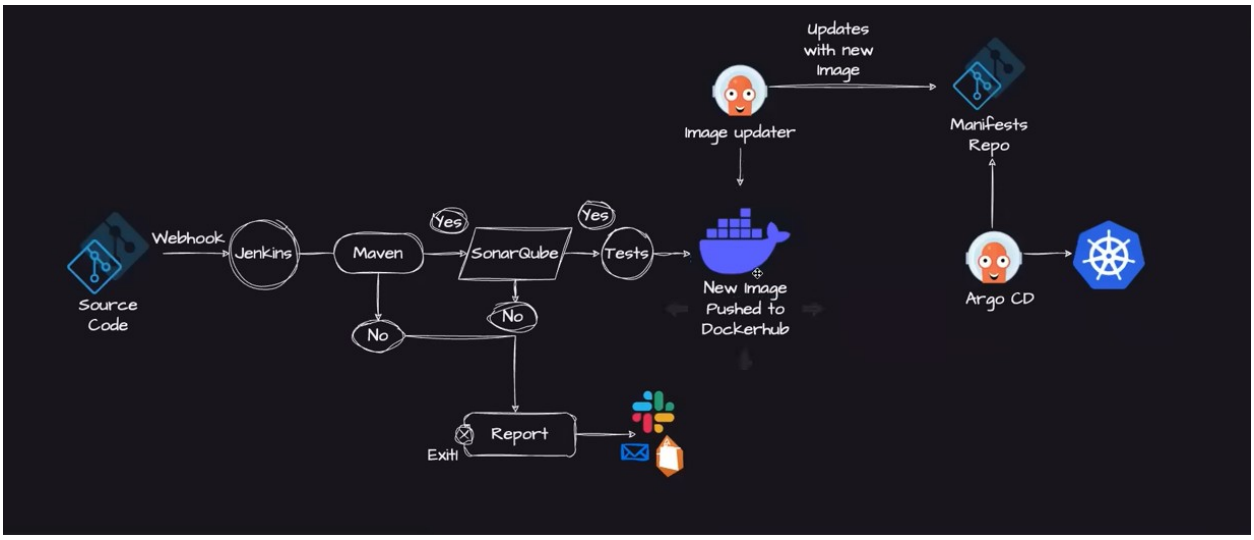
Please wait while Jenkins is restarting ...

Your browser will reload automatically when Jenkins is ready.





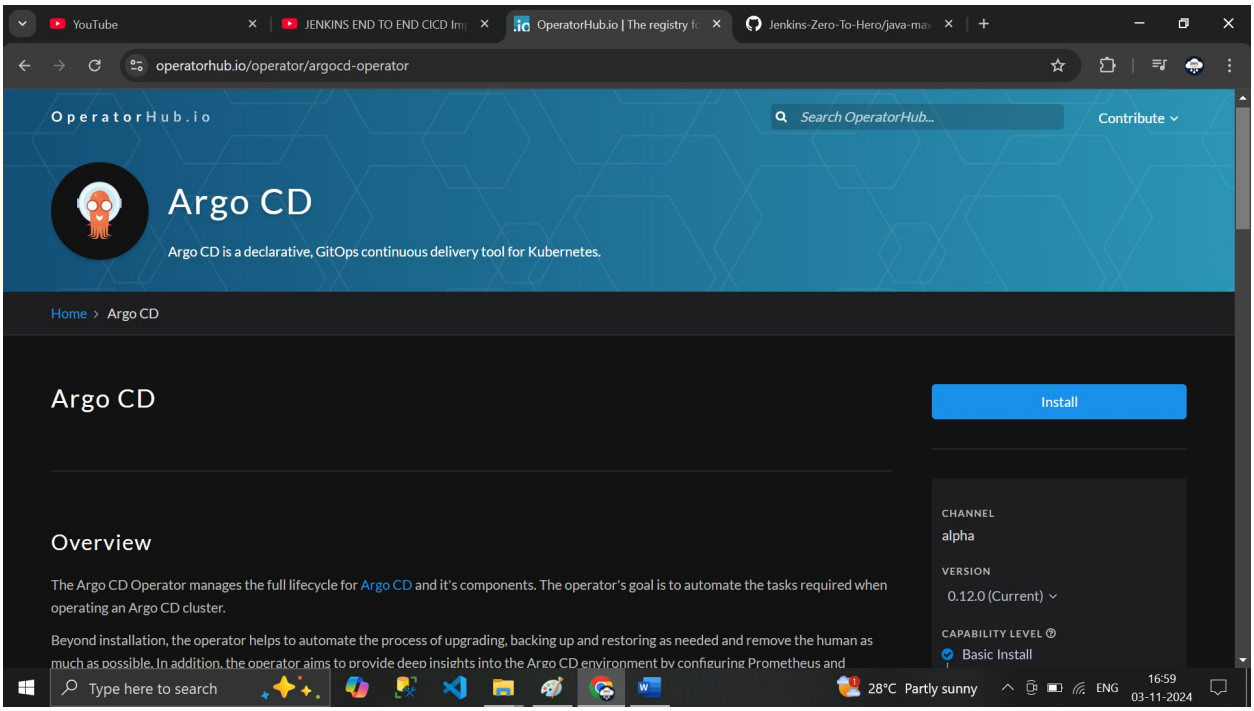
Docker step is Configured

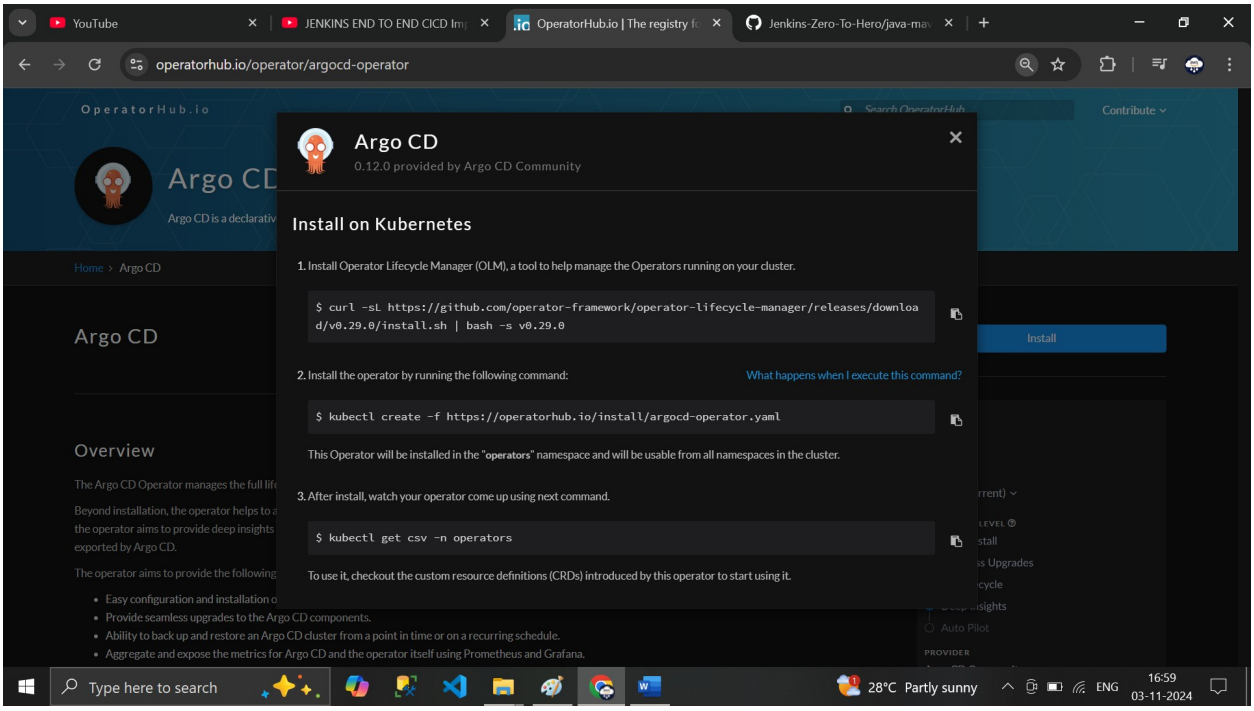


Final step .. install ArgoCD + Kubernetes Cluster …

```
➜  ~ git:(argo-all-namespaces) ✗ min
➜  ~ git:(argo-all-namespaces) ✗ minikube start --memory=4098 --driver=hyperkit
😄  minikube v1.25.2 on Darwin 13.2.1
✨  Using the hyperkit driver based on user configuration
👍  Starting control plane node minikube in cluster minikube
🔥  Creating hyperkit VM (CPUs=2, Memory=4098MB, Disk=20000MB) ...
🐳  Preparing Kubernetes v1.23.3 on Docker 20.10.12 ...
    ▪ kubelet.housekeeping-interval=5m
    ▪ Generating certificates and keys ...
    ▪ Booting up control plane ...
    ▪ Configuring RBAC rules ...
🔎  Verifying Kubernetes components...
    ▪ Using image gcr.io/k8s-minikube/storage-provisioner:v5
🌟  Enabled addons: storage-provisioner, default-storageclass

❗  /usr/local/bin/kubectl is version 1.21.3, which may have incompatibilites with Kubernetes 1.23.3.
    ▪ Want kubectl v1.23.3? Try 'minikube kubectl -- get pods -A'
🏄  Done! kubectl is now configured to use "minikube" cluster and "default" namespace by default
➜  ~ git:(argo-all-namespaces) ✗
```

https://operatorhub.io/operator/argocd-operator


YouTube
JENKINS END TO END CICD Im
OperatorHub.io | The registry
Jenkins-Zero-To-Hero/java-ma
operatorhub.io/operator/argocd-operator
OperatorHub.io
Search OperatorHub...
Contribute
Argo CD
Argo CD is a declarative, GitOps continuous delivery tool for Kubernetes.
Home > Argo CD
Argo CD
Install
Overview
The Argo CD Operator manages the full lifecycle for Argo CD and it's components. The operator's goal is to automate the tasks required when operating an Argo CD cluster.
Beyond installation, the operator helps to automate the process of upgrading, backing up and restoring as needed and remove the human as much as possible. In addition, the operator aims to provide deep insights into the Argo CD environment by configuring Prometheus and
CHANNEL
alpha
VERSION
0.12.0 (Current)
CAPABILITY LEVEL
Basic Install
Type here to search
28°C Partly sunny
ENG
16:59
03-11-2024


YouTube
JENKINS END TO END CICD Im
OperatorHub.io | The registry
Jenkins-Zero-To-Hero/java-ma
operatorhub.io/operator/argocd-operator
Argo CD
0.12.0 provided by Argo CD Community
Install on Kubernetes
1. Install Operator Lifecycle Manager (OLM), a tool to help manage the Operators running on your cluster.
$ curl -sL https://github.com/operator-framework/operator-lifecycle-manager/releases/download/v0.29.0/install.sh | bash -s v0.29.0
2. Install the operator by running the following command:
What happens when I execute this command?
$ kubectl create -f https://operatorhub.io/install/argocd-operator.yaml
This Operator will be installed in the "operators" namespace and will be usable from all namespaces in the cluster.
3. After install, watch your operator come up using next command.
$ kubectl get csv -n operators
To use it, checkout the custom resource definitions (CRDs) introduced by this operator to start using it.
Argo CD
Install
Overview
Type here to search
28°C Partly sunny
ENG
16:59
03-11-2024

Install on Kubernetes

1. Install Operator Lifecycle Manager (OLM), a tool to help manage the Operators running on your cluster.

```
$ curl -sL https://github.com/operator-framework/operator-lifecycle-manager/releases/download/v0.29.0/install.sh | bash -s v0.29.0
```

2. Install the operator by running the following command: What happens when I execute this command?

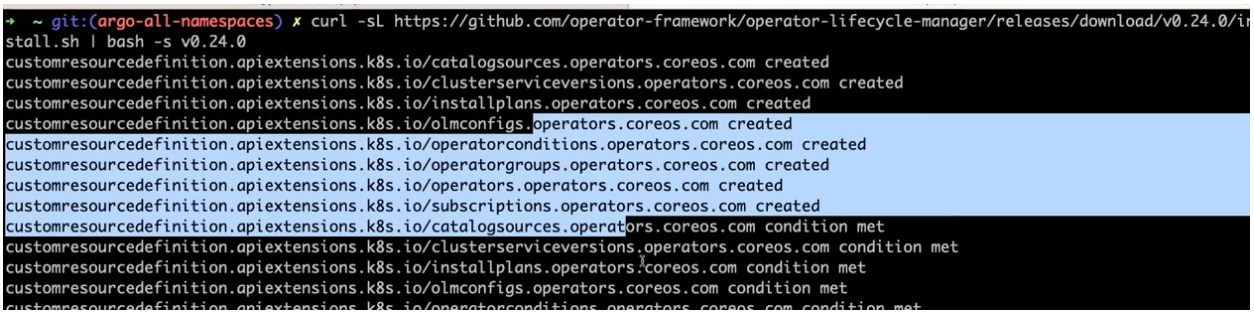
```
$ kubectl create -f https://operatorhub.io/install/argocd-operator.yaml
```

This Operator will be installed in the "**operators**" namespace and will be usable from all namespaces in the cluster.

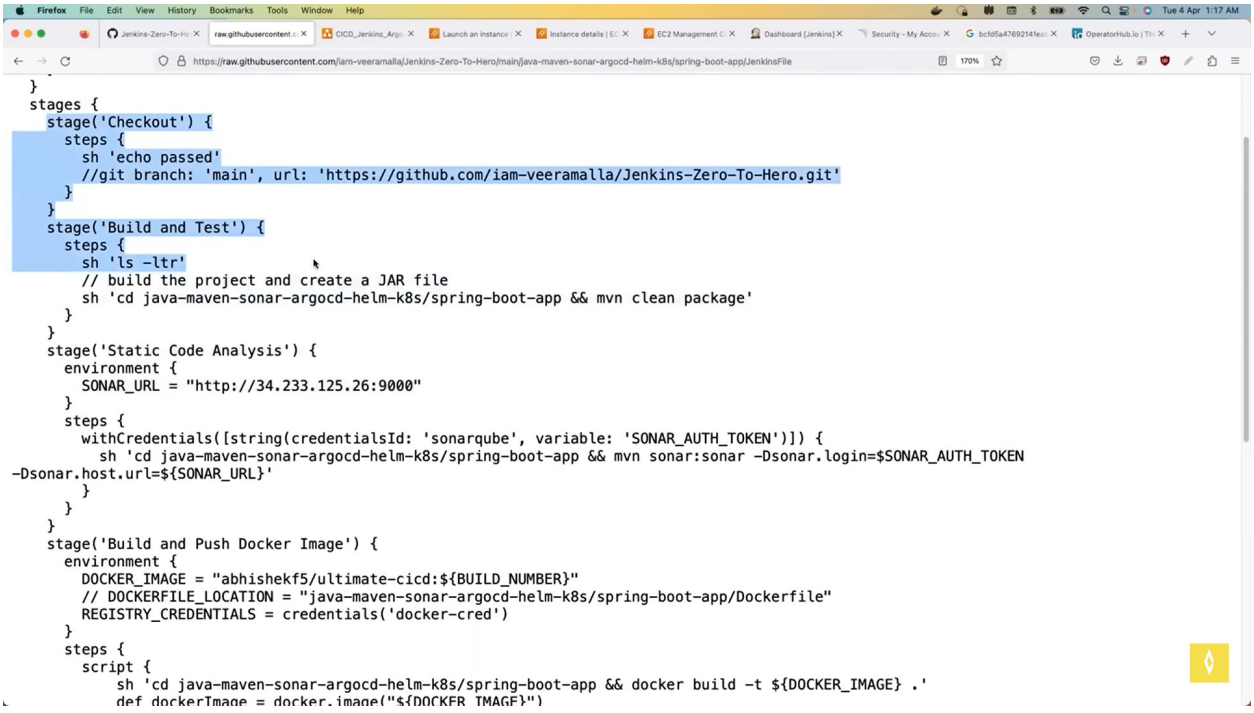
3. After install, watch your operator come up using next command.

```
$ kubectl get csv -n operators
```

To use it, checkout the custom resource definitions (CRDs) introduced by this operator to start using it.



```
customresourcedefinition.apiextensions.k8s.io/operators.operators.coreos.com condition met
customresourcedefinition.apiextensions.k8s.io/subscriptions.operators.coreos.com condition met
namespace/olm created
namespace/operators created
serviceaccount/olm-operator-serviceaccount created
clusterrole.rbac.authorization.k8s.io/system:controller:operator-lifecycle-manager created
clusterrolebinding.rbac.authorization.k8s.io/olm-operator-binding-olm created
olmconfig.operators.coreos.com/cluster created
deployment.apps/olm-operator created
deployment.apps/catalog-operator created
clusterrole.rbac.authorization.k8s.io/aggregate-olm-edit created
clusterrole.rbac.authorization.k8s.io/aggregate-olm-view created
operatorgroup.operators.coreos.com/global-operators created
operatorgroup.operators.coreos.com/olm-operators created
clusterserviceversion.operators.coreos.com/packageserver created
catalogsource.operators.coreos.com/operatorhubio-catalog created
Waiting for deployment "olm-operator" rollout to finish: 0 of 1 updated replicas are available...


deployment "olm-operator" successfully rolled out
deployment "catalog-operator" successfully rolled out




Package server phase: Installing
```

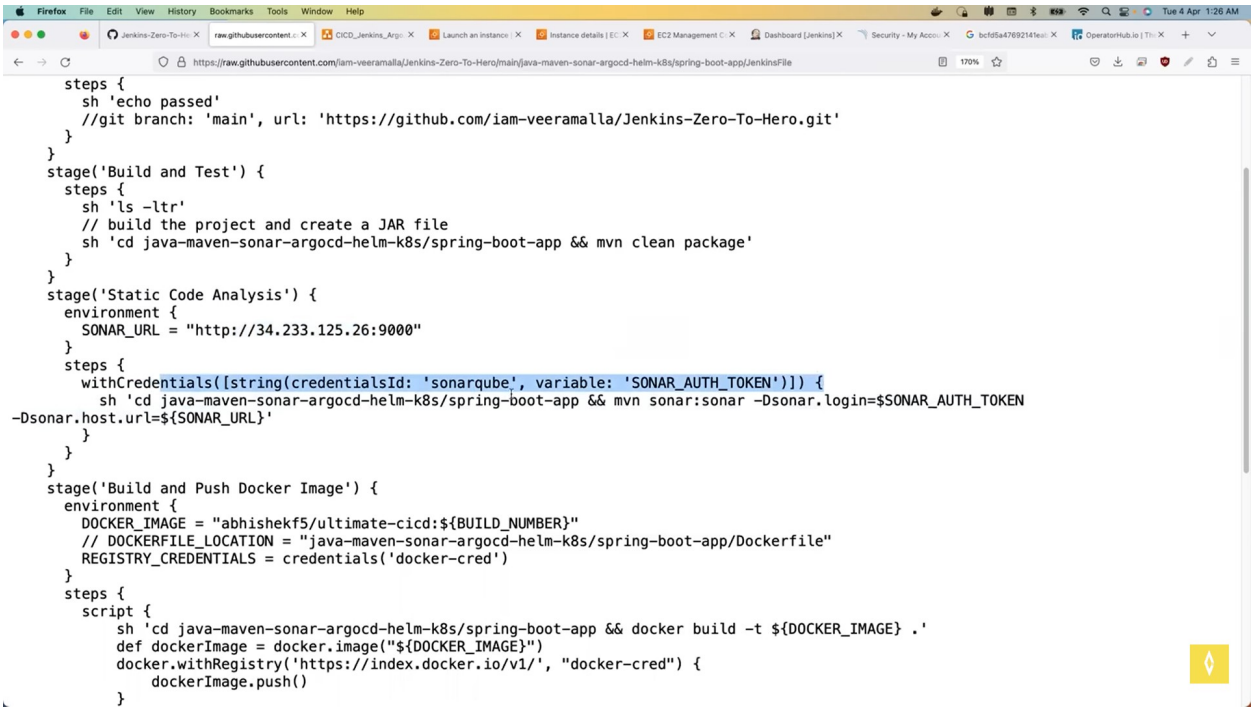


```
→  ~ git:(argo-all-namespaces) ✗ kubectl create -f https://operatorhub.io/install/argocd-operator.yaml
subscription.operators.coreos.com/my-argocd-operator created
→  ~ git:(argo-all-namespaces) ✗ kubectl get pods -n operators
No resources found in operators namespace.
→  ~ git:(argo-all-namespaces) ✗ kubectl get pods -n operators -w
^C
→  ~ git:(argo-all-namespaces) ✗ kubectl get pods -n operators
No resources found in operators namespace.
→  ~ git:(argo-all-namespaces) ✗ kubectl get pods -n operators
No resources found in operators namespace.
→  ~ git:(argo-all-namespaces) ✗ kubectl get pods -n operators
NAME                                                READY   STATUS              RESTARTS   AGE
argocd-operator-controller-manager-c775dbd54-vqmc4   0/2     ContainerCreating   0          35s
→  ~ git:(argo-all-namespaces) ✗
```

```
→  ~ git:(argo-all-namespaces) ✗ cd ~/go/src/github.com/
F5Networks/          example.com/         redhat-developer/    test/
argoproj/            iam-veeramalla/      secure-mid-stream/
```

```
→  spring-boot-app git:(main) ✗ ls
Dockerfile README.md  pom.xml    src        target
→  spring-boot-app git:(main) ✗ ls target
classes                 maven-archiver          spring-boot-web.jar
generated-sources       maven-status            spring-boot-web.jar.original
→  spring-boot-app git:(main) ✗
```

```
# For Java 8, try this
# FROM openjdk:8-jdk-alpine

# For Java 11, try this
FROM adoptopenjdk/openjdk11:alpine-jre

# Refer to Maven build -> finalName
ARG JAR_FILE=target/spring-boot-web.jar

# cd /opt/app
WORKDIR /opt/app

# cp target/spring-boot-web.jar /opt/app/app.jar
COPY ${JAR_FILE} app.jar

# java -jar /opt/app/app.jar
ENTRYPOINT ["java","-jar","app.jar"]

## sudo docker run -p 8080:8080 -t docker-spring-boot:1.0
## sudo docker run -p 80:8080 -t docker-spring-boot:1.0
## sudo docker run -p 443:8443 -t docker-spring-boot:1.0
~
~
```

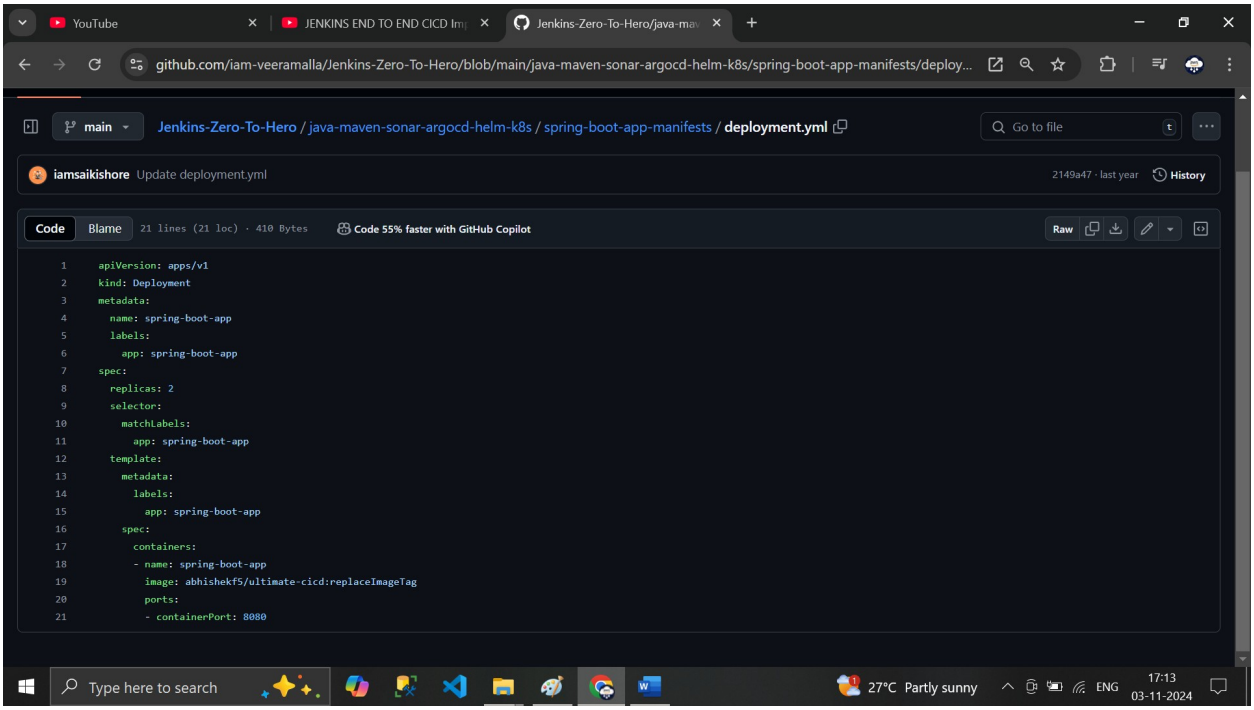


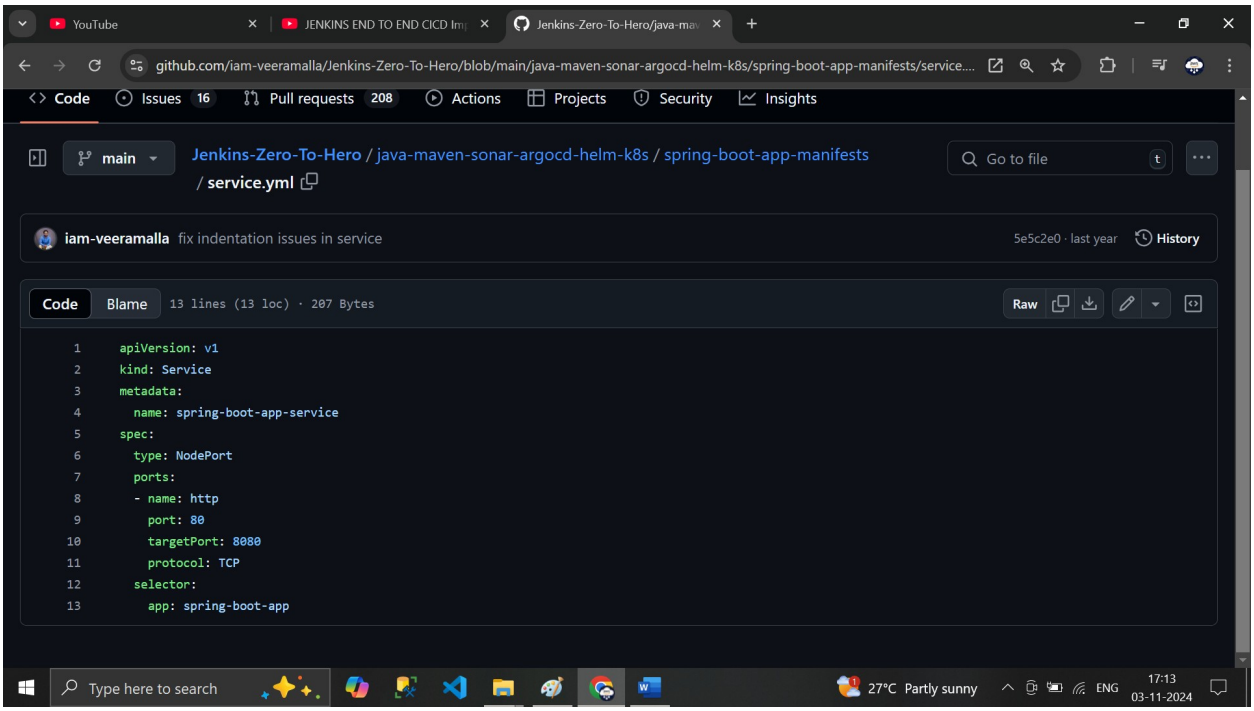
```
      steps {
        sh 'echo passed'
        //git branch: 'main', url: 'https://github.com/iam-veeramalla/Jenkins-Zero-To-Hero.git'
      }
    }
    stage('Build and Test') {
      steps {
        sh 'ls -ltr'
        // build the project and create a JAR file
        sh 'cd java-maven-sonar-argocd-helm-k8s/spring-boot-app && mvn clean package'
      }
    }
    stage('Static Code Analysis') {
      environment {
        SONAR_URL = "http://34.233.125.26:9000"
      }
      steps {
        withCredentials([string(credentialsId: 'sonarqube', variable: 'SONAR_AUTH_TOKEN')]) {
          sh 'cd java-maven-sonar-argocd-helm-k8s/spring-boot-app && mvn sonar:sonar -Dsonar.login=$SONAR_AUTH_TOKEN
-Dsonar.host.url=${SONAR_URL}'
        }
      }
    }
    stage('Build and Push Docker Image') {
      environment {
        DOCKER_IMAGE = "abhishekf5/ultimate-cicd:${BUILD_NUMBER}"
        // DOCKERFILE_LOCATION = "java-maven-sonar-argocd-helm-k8s/spring-boot-app/Dockerfile"
        REGISTRY_CREDENTIALS = credentials('docker-cred')
      }
      steps {
        script {
            sh 'cd java-maven-sonar-argocd-helm-k8s/spring-boot-app && docker build -t ${DOCKER_IMAGE} .'
            def dockerImage = docker.image("${DOCKER_IMAGE}")
            docker.withRegistry('https://index.docker.io/v1/', "docker-cred") {
                dockerImage.push()
            }
```

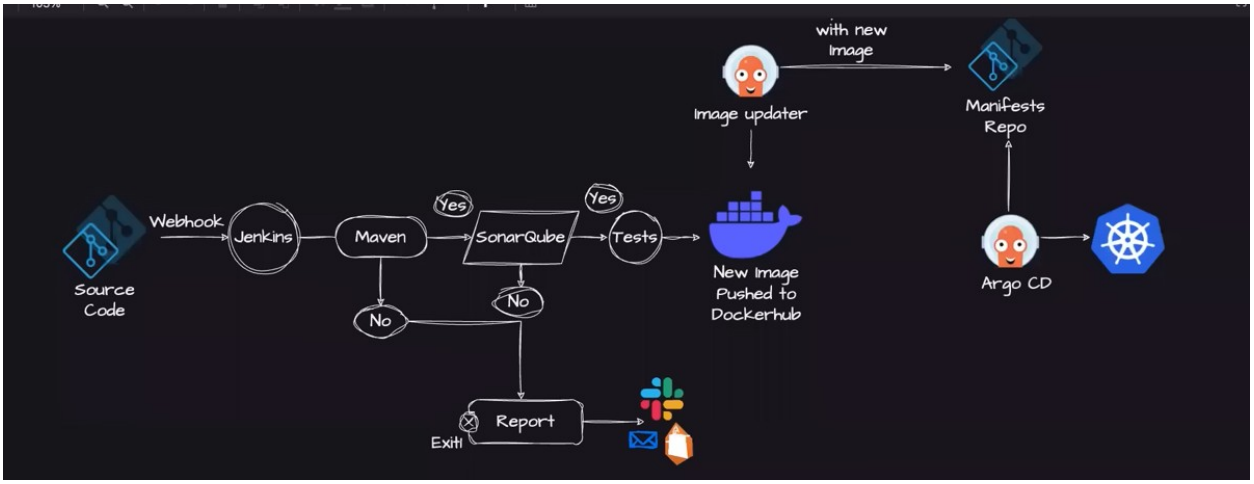
```
        SONAR_URL = "http://34.233.125.26:9000"
      }
      steps {
        withCredentials([string(credentialsId: 'sonarqube', variable: 'SONAR_AUTH_TOKEN')]) {
          sh 'cd java-maven-sonar-argocd-helm-k8s/spring-boot-app && mvn sonar:sonar -Dsonar.login=$SONAR_AUTH_TOKEN
-Dsonar.host.url=${SONAR_URL}'
        }
      }
    }
    stage('Build and Push Docker Image') {
      environment {
        DOCKER_IMAGE = "abhishekf5/ultimate-cicd:${BUILD_NUMBER}"
        // DOCKERFILE_LOCATION = "java-maven-sonar-argocd-helm-k8s/spring-boot-app/Dockerfile"
        REGISTRY_CREDENTIALS = credentials('docker-cred')
      }
      steps {
        script {
            sh 'cd java-maven-sonar-argocd-helm-k8s/spring-boot-app && docker build -t ${DOCKER_IMAGE} .'
            def dockerImage = docker.image("${DOCKER_IMAGE}")
            docker.withRegistry('https://index.docker.io/v1/', "docker-cred") {
                dockerImage.push()
            }
        }
      }
    }
    stage('Update Deployment File') {
        environment {
            GIT_REPO_NAME = "Jenkins-Zero-To-Hero"
            GIT_USER_NAME = "iam-veeramalla"
        }
        steps {
            withCredentials([string(credentialsId: 'github', variable: 'GITHUB_TOKEN')]) {
                sh '''
                    git config user.email "abhishek.xyz@gmail.com"
                    git config user.name "Abhishek Veeramalla"
                    BUILD_NUMBER=${BUILD_NUMBER}
```

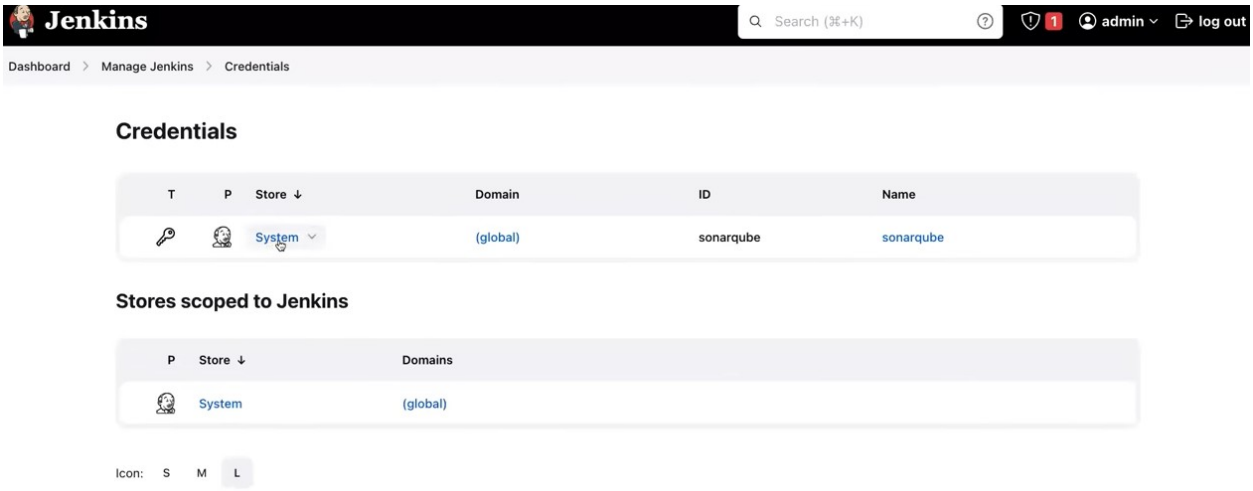
```
        }
    }
    stage('Update Deployment File') {
        environment {
            GIT_REPO_NAME = "Jenkins-Zero-To-Hero"
            GIT_USER_NAME = "iam-veeramalla"
        }
        steps {
            withCredentials([string(credentialsId: 'github', variable: 'GITHUB_TOKEN')]) {
                sh '''
                    git config user.email "abhishek.xyz@gmail.com"
                    git config user.name "Abhishek Veeramalla"
                    BUILD_NUMBER=${BUILD_NUMBER}
                    sed -i "s/replaceImageTag/${BUILD_NUMBER}/g" java-maven-sonar-argocd-helm-k8s/spring-boot-app-manifests/deployment.yml
                    git add java-maven-sonar-argocd-helm-k8s/spring-boot-app-manifests/deployment.yml
                    git commit -m "Update deployment image to version ${BUILD_NUMBER}"
                    git push https://${GITHUB_TOKEN}@github.com/${GIT_USER_NAME}/${GIT_REPO_NAME} HEAD:main
                '''
            }
        }
    }
  }
}
```

spring Boot Manifest

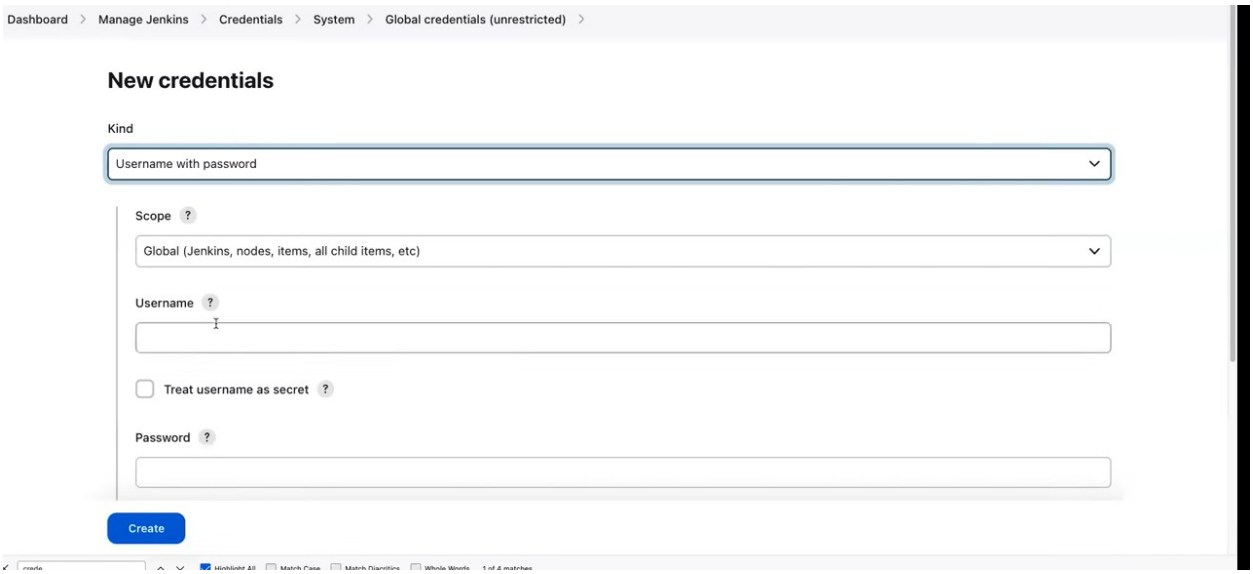



YouTube
JENKINS END TO END CICD Im
Jenkins-Zero-To-Hero/java-mav
github.com/iam-veeramalla/Jenkins-Zero-To-Hero/blob/main/java-maven-sonar-argocd-helm-k8s/spring-boot-app-manifests/service....
Code
Issues 16
Pull requests 208
Actions
Projects
Security
Insights
main
Jenkins-Zero-To-Hero / java-maven-sonar-argocd-helm-k8s / spring-boot-app-manifests / service.yml
Go to file
iam-veeramalla fix indentation issues in service
5e5c2e0 · last year
History
Code
Blame
13 lines (13 loc) · 207 Bytes
Raw
apiVersion: v1
kind: Service
metadata:
name: spring-boot-app-service
spec:
type: NodePort
ports:
- name: http
port: 80
targetPort: 8080
protocol: TCP
selector:
app: spring-boot-app
Type here to search
27°C Partly sunny
ENG
17:13
03-11-2024


with new Image
Image updater
Manifests Repo
Webhook
Jenkins
Maven
Yes
SonarQube
Yes
Tests
Source Code
No
No
New Image Pushed to Dockerhub
Argo CD
Report
Exit!

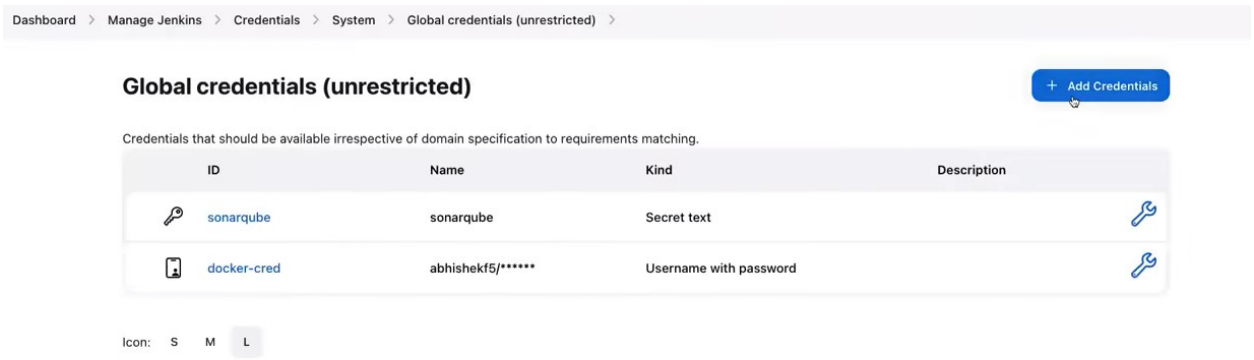


 # username + password – docker hub

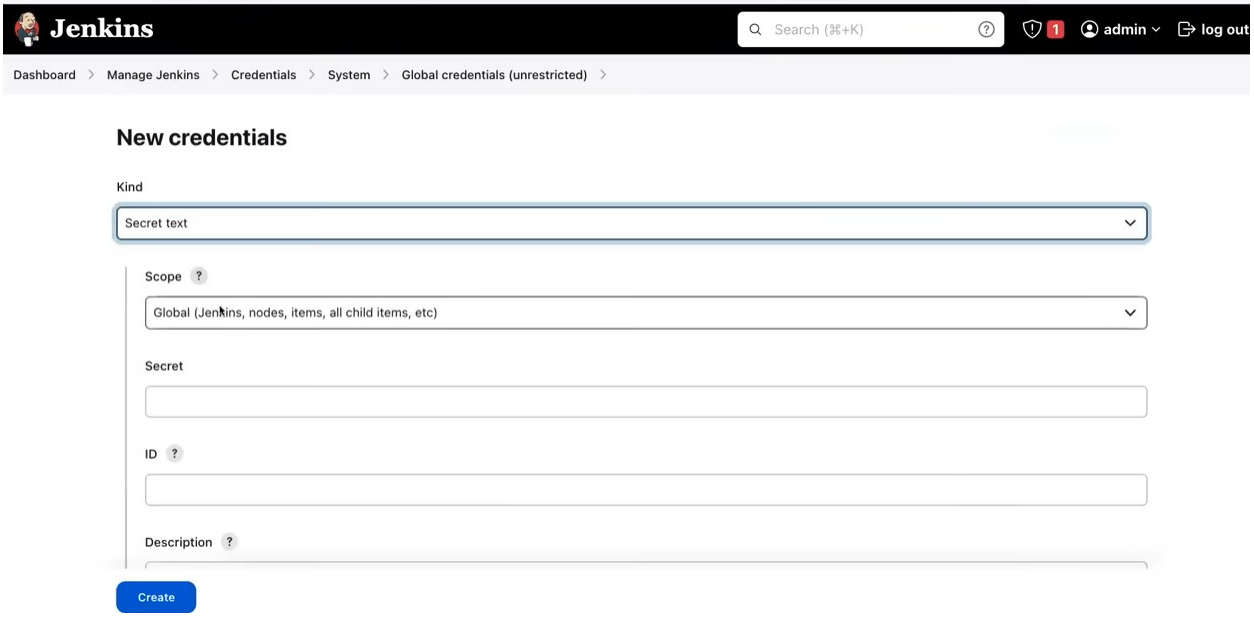


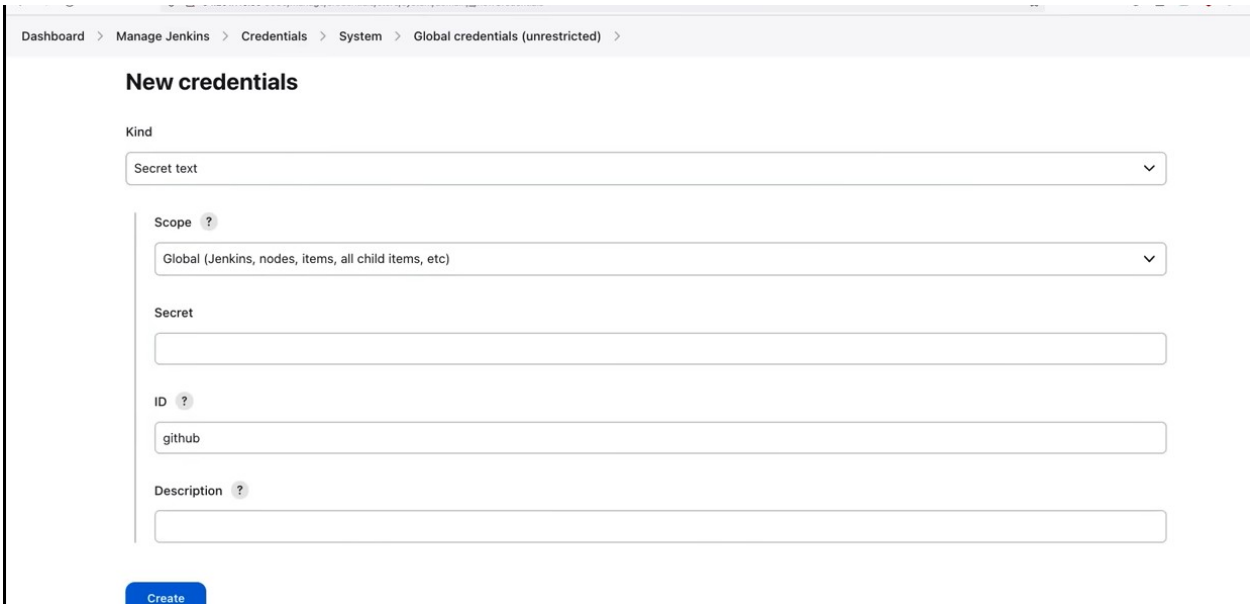
```
      steps {
        withCredentials([string(credentialsId: 'sonarqube', variable: 'SONAR_AUTH_TOKEN')]) {
          sh 'cd java-maven-sonar-argocd-helm-k8s/spring-boot-app && mvn sonar:sonar -Dsonar.login=$SONAR_AUTH_TOKEN
-Dsonar.host.url=${SONAR_URL}'
        }
      }
    }
    stage('Build and Push Docker Image') {
      environment {
        DOCKER_IMAGE = "abhishekf5/ultimate-cicd:${BUILD_NUMBER}"
        // DOCKERFILE_LOCATION = "java-maven-sonar-argocd-helm-k8s/spring-boot-app/Dockerfile"
        REGISTRY_CREDENTIALS = credentials('docker-cred')
      }
      steps {
        script {
            sh 'cd java-maven-sonar-argocd-helm-k8s/spring-boot-app && docker build -t ${DOCKER_IMAGE} .'
            def dockerImage = docker.image("${DOCKER_IMAGE}")
            docker.withRegistry('https://index.docker.io/v1/', "docker-cred") {
                dockerImage.push()
            }
        }
      }
    }
    stage('Update Deployment File') {
        environment {
            GIT_REPO_NAME = "Jenkins-Zero-To-Hero"
            GIT_USER_NAME = "iam-veeramalla"
        }
        steps {
            withCredentials([string(credentialsId: 'github', variable: 'GITHUB_TOKEN')]) {
                sh '''
                    git config user.email "abhishek.xyz@gmail.com"
                    git config user.name "Abhishek Veeramalla"
                    BUILD_NUMBER=${BUILD_NUMBER}
                    sed -i "s/replaceImageTag/${BUILD_NUMBER}/g" java-maven-sonar-argocd-helm-k8s/spring-boot-app-manifests/deployment.yml
```

 # note – password must be DockerHub only

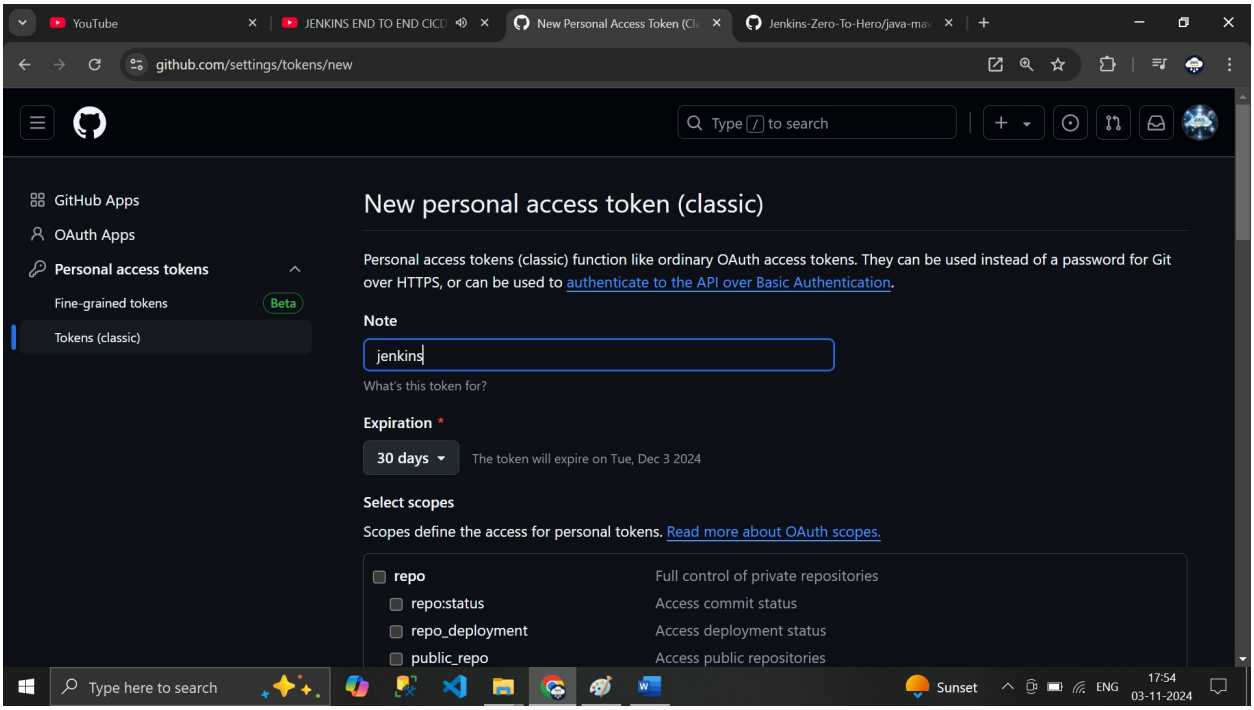


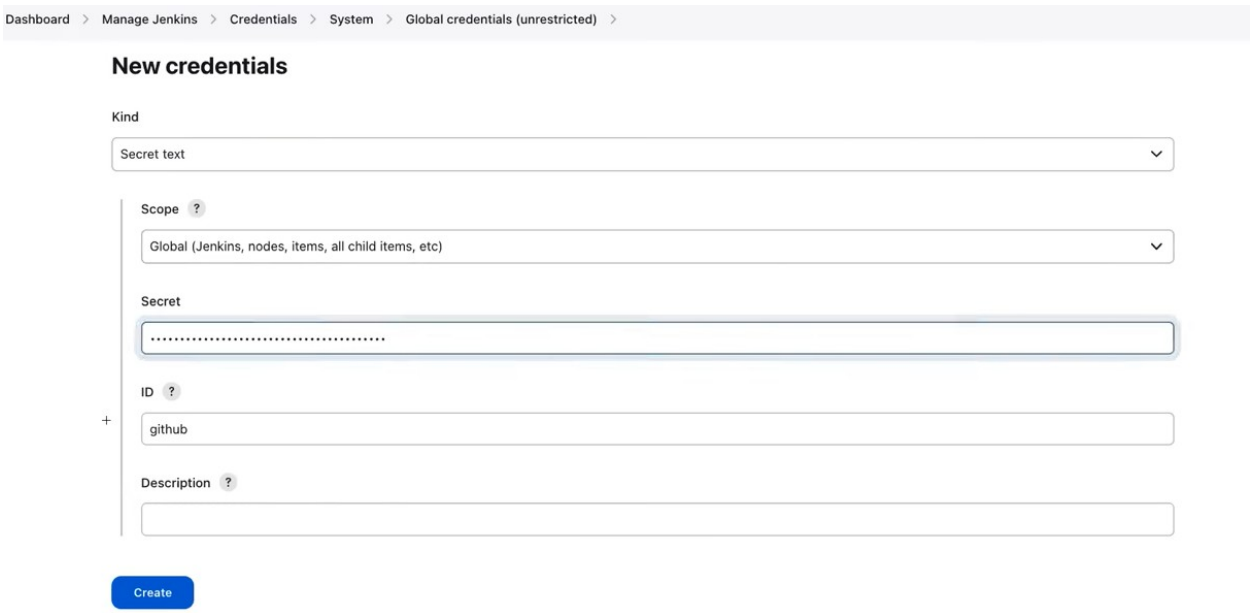
 # now its for GITHUb Credentials – Choose Secret text

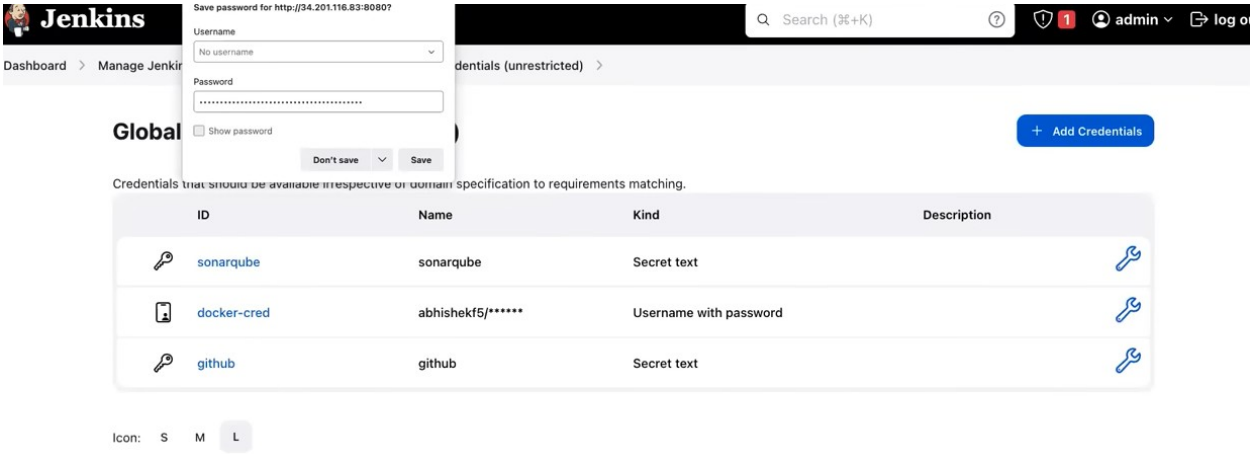




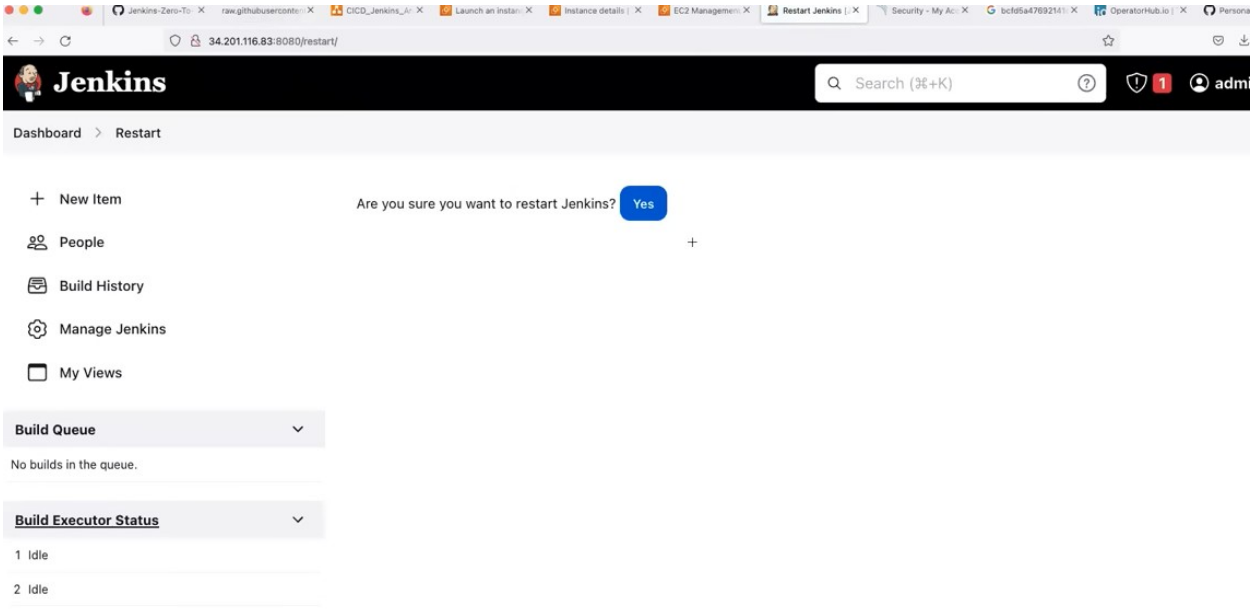
 # Access Key from Github – Settings- developer settings-personal- access-tokens – Generate new-token




Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >
New credentials
Kind
Secret text
Scope ?
Global (Jenkins, nodes, items, all child items, etc)
Secret
ID ?
github
Description ?
Create


Jenkins
Save password for http://34.201.116.83:8080?
Username
No username
Password
Show password
Don't save
Save
Search (⌘+K)
admin
Dashboard > Manage Jenkin dentials (unrestricted) >
Global
Add Credentials
Credentials that should be available irrespective of domain specification to requirements matching.
ID
Name
Kind
Description
sonarqube
sonarqube
Secret text
docker-cred
abhishekf5/******
Username with password
github
github
Secret text
Icon: S M L

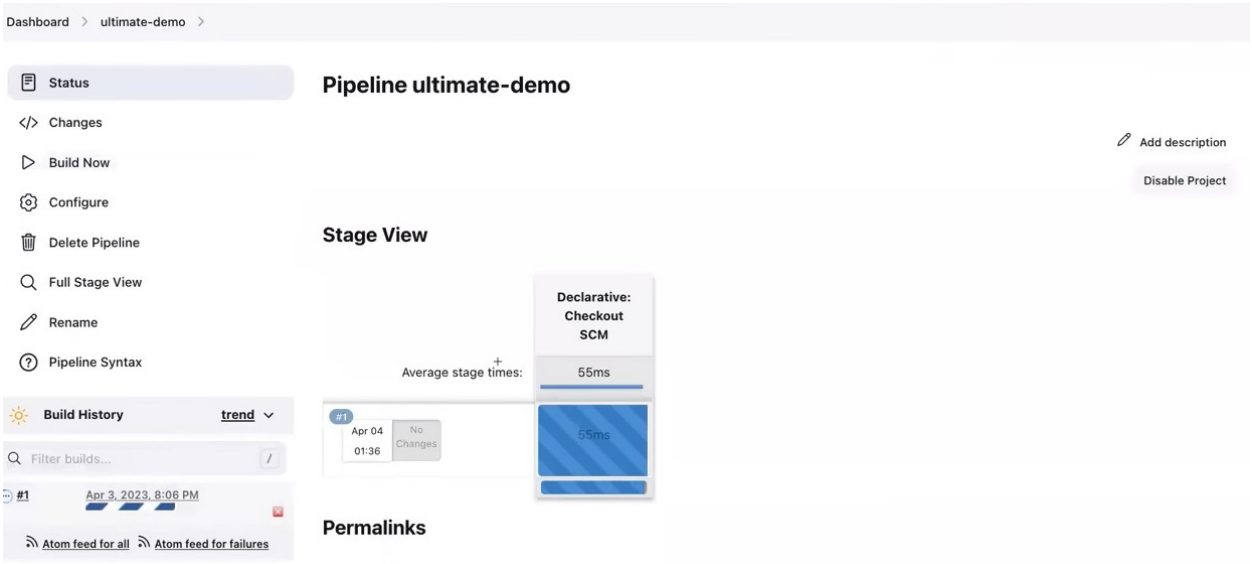
 # Restart

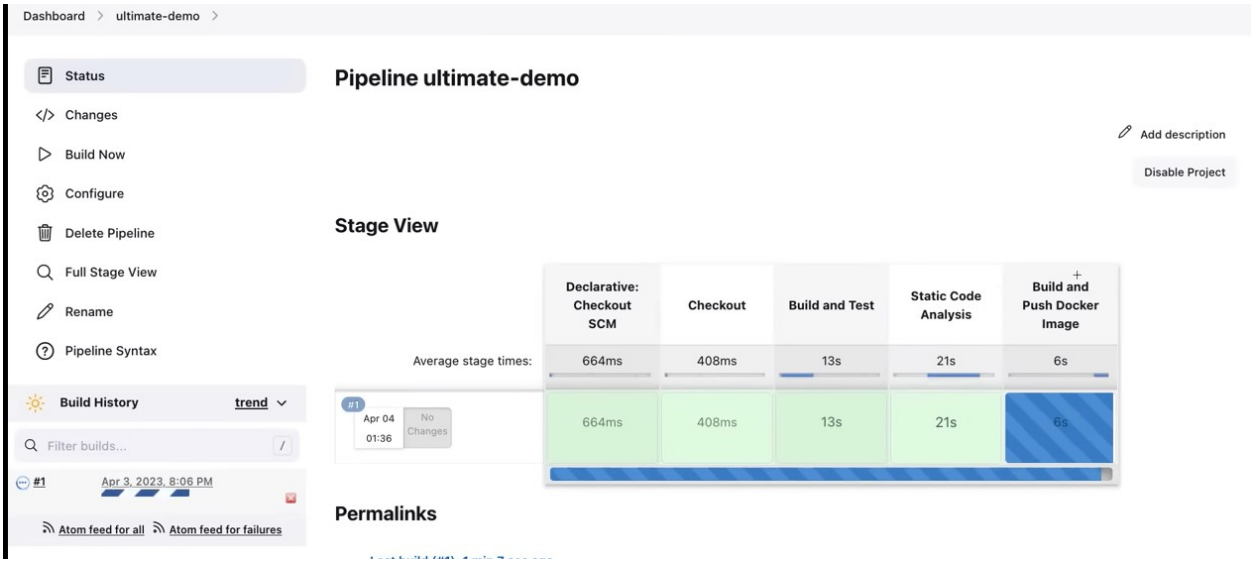


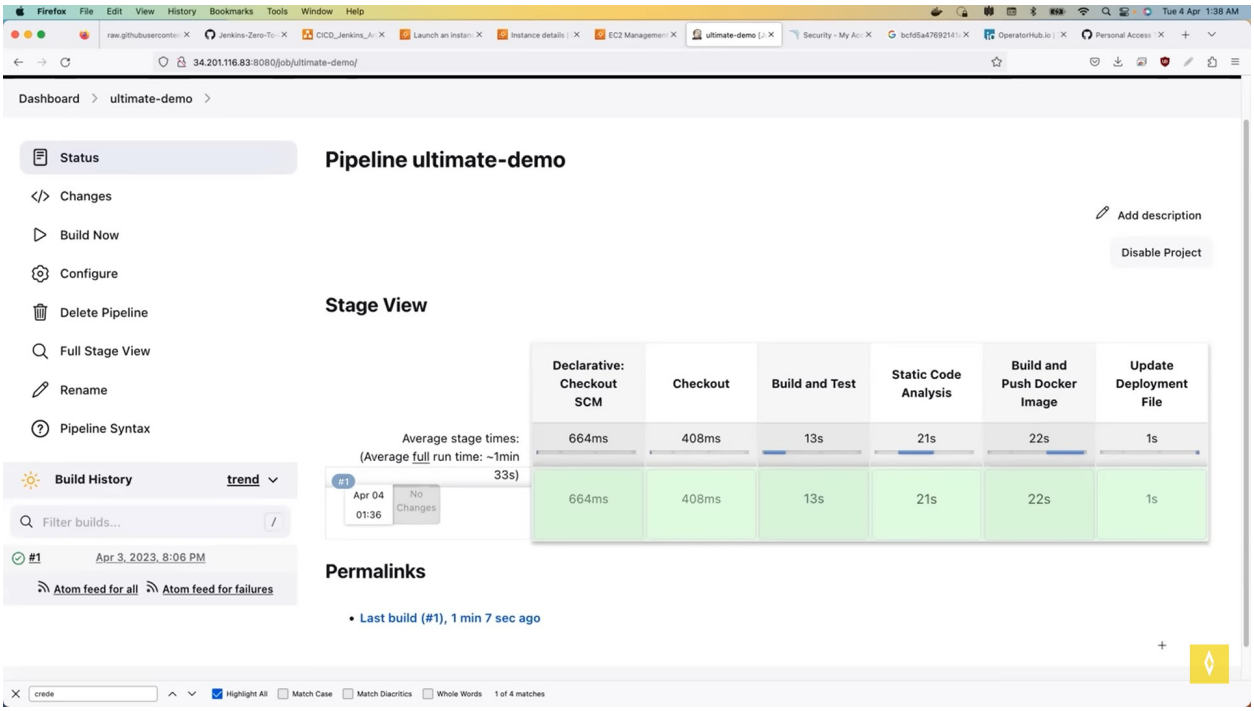

```
  }
  stage('Static Code Analysis') {
    environment {
      SONAR_URL = "http://34.201.116.83:9000"
    }
    steps {
      withCredentials([string(credentialsId: 'sonarqube', variable: 'SONAR_AUTH_TOKEN')]) {
        sh 'cd java-maven-sonar-argocd-helm-k8s/spring-boot-app && mvn sonar:sonar -Dsonar.login=$SONAR_AUTH_TOKEN -Dsonar.host.url=${SONAR_URL}'
      }
    }
  }
  stage('Build and Push Docker Image') {
    environment {
      DOCKER_IMAGE = "abhishekf5/ultimate-cicd:${BUILD_NUMBER}"
      // DOCKERFILE_LOCATION = "java-maven-sonar-argocd-helm-k8s/spring-boot-app/Dockerfile"
      REGISTRY_CREDENTIALS = credentials('docker-cred')
```

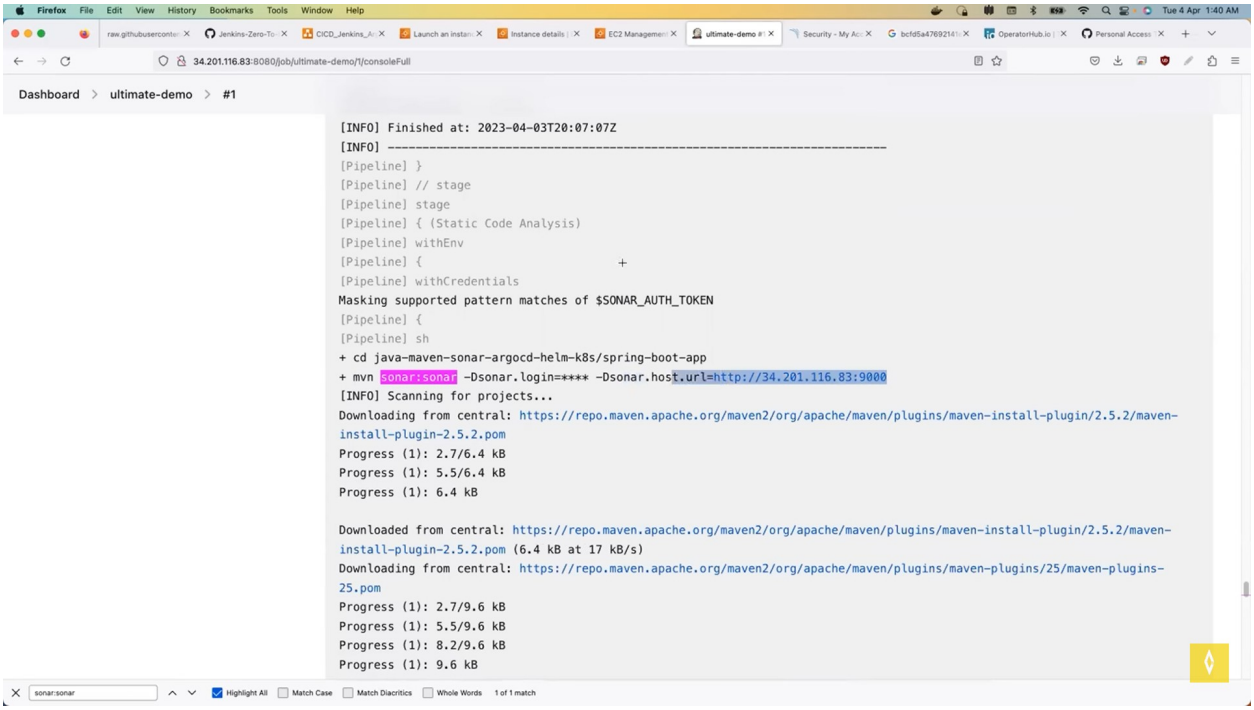
```
→  spring-boot-app git:(main) ✗ kubectl get pods -n operators
NAME                                              READY   STATUS    RESTARTS   AGE
argocd-operator-controller-manager-c775dbd54-vqmc4   2/2     Running   0          17m
→  spring-boot-app git:(main) ✗
```

 # now time for CD – PART - Click on Build now on Jenkins

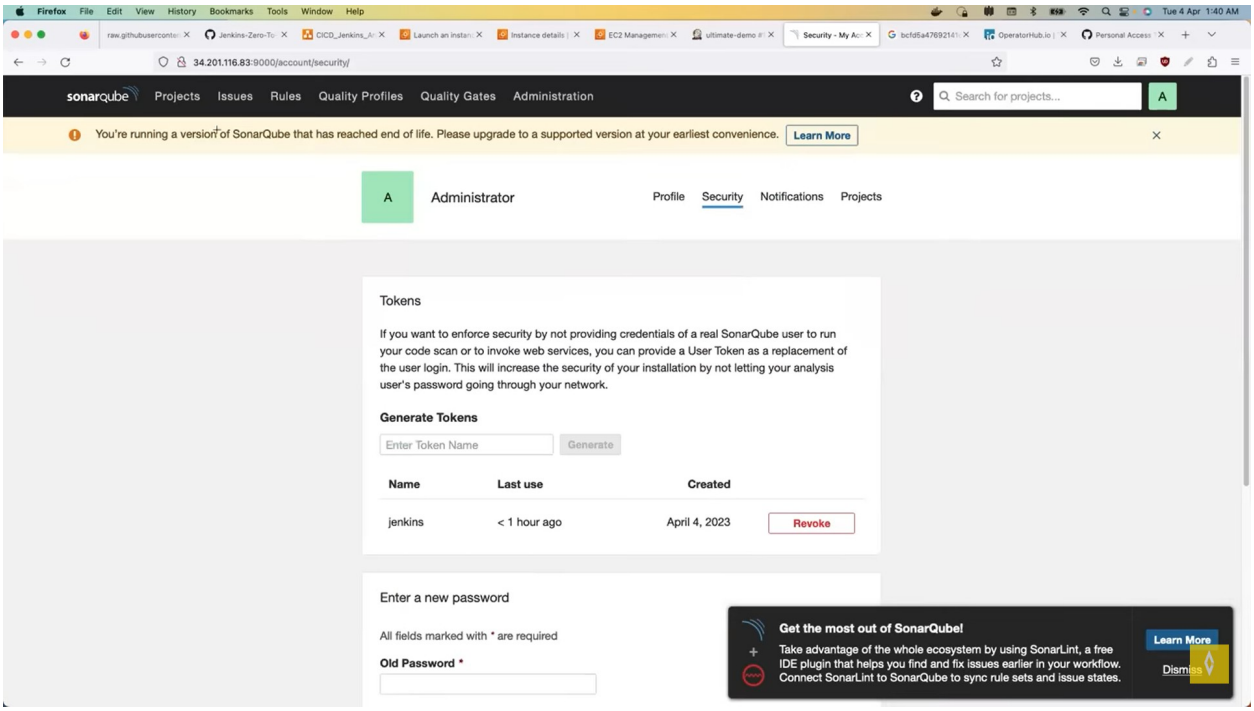

Dashboard
ultimate-demo
Status
Changes
Build Now
Configure
Delete Pipeline
Full Stage View
Rename
Pipeline Syntax
Build History
trend
Filter builds...
#1
Apr 3, 2023, 8:06 PM
Atom feed for all
Atom feed for failures
Pipeline ultimate-demo
Add description
Disable Project
Stage View
Declarative: Checkout SCM
Average stage times:
55ms
#1
Apr 04
01:36
No Changes
55ms
Permalinks


Dashboard
ultimate-demo
Status
Changes
Build Now
Configure
Delete Pipeline
Full Stage View
Rename
Pipeline Syntax
Build History
trend
Filter builds...
#1
Apr 3, 2023, 8:06 PM
Atom feed for all
Atom feed for failures
Pipeline ultimate-demo
Add description
Disable Project
Stage View
Declarative: Checkout SCM
Checkout
Build and Test
Static Code Analysis
Build and Push Docker Image
Average stage times:
664ms
408ms
13s
21s
6s
#1
Apr 04
01:36
No Changes
664ms
408ms
13s
21s
6s
Permalinks

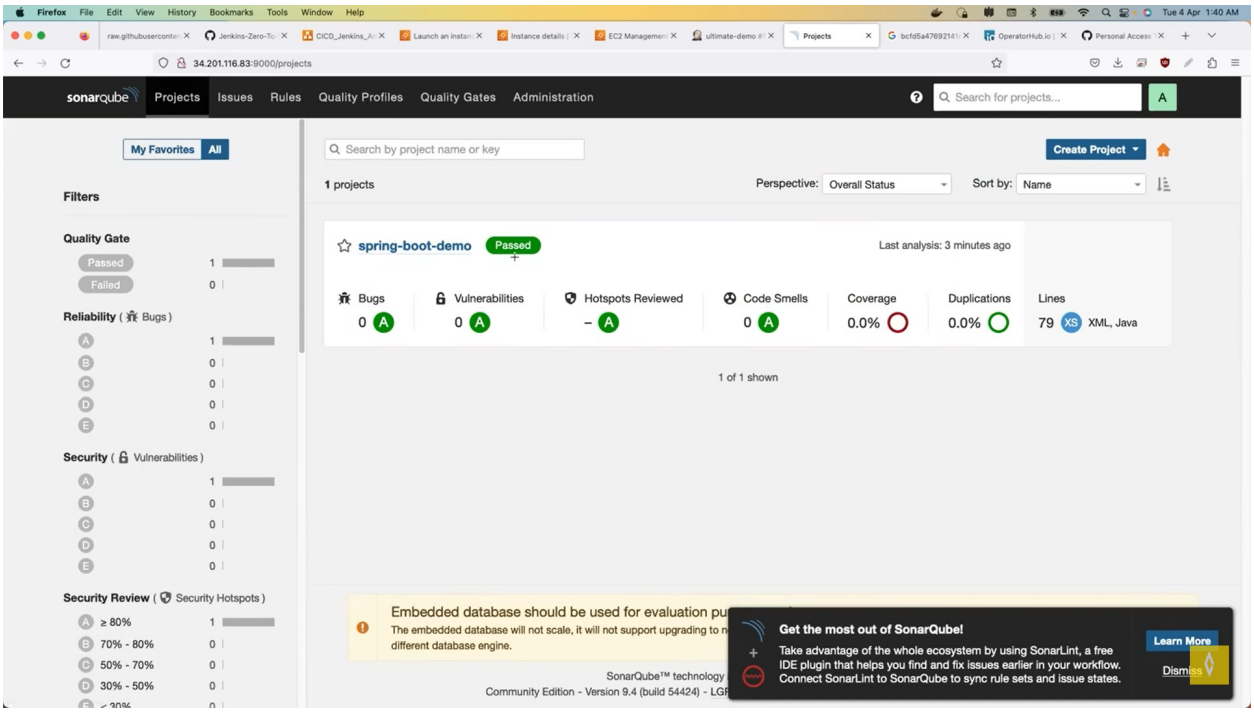




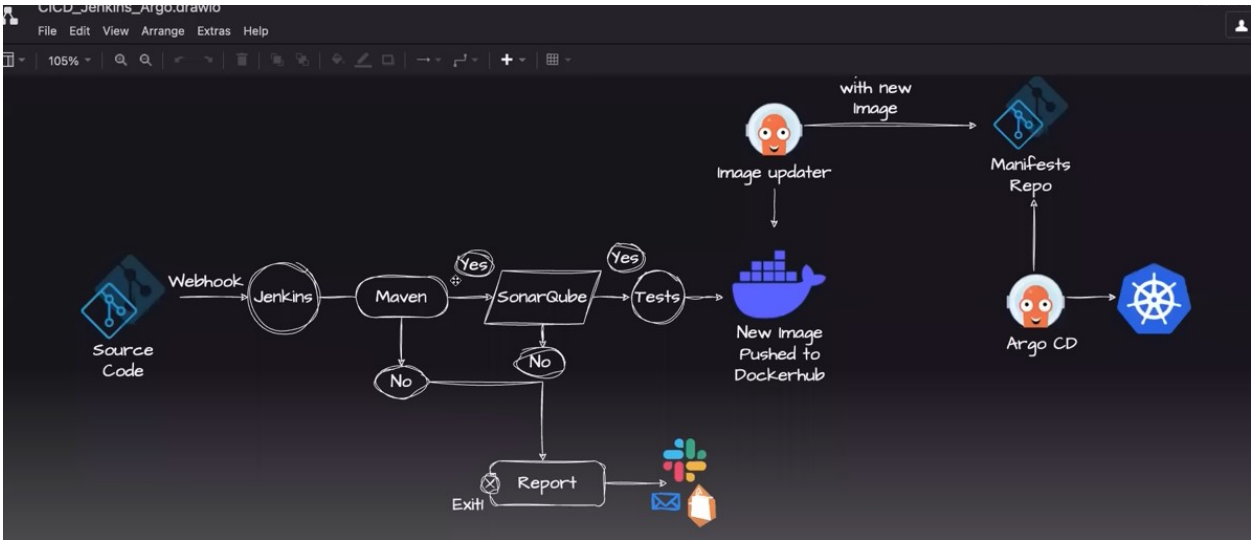
 # Go to SonarCube – Refresh it –



 # Go to Projects

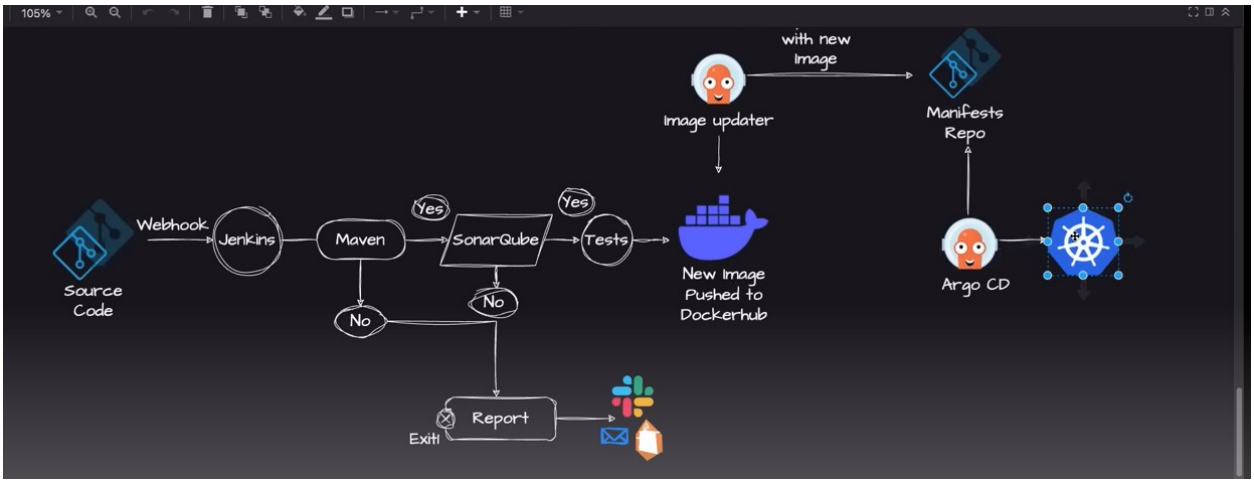


 # now check Docker image is Created or not ??



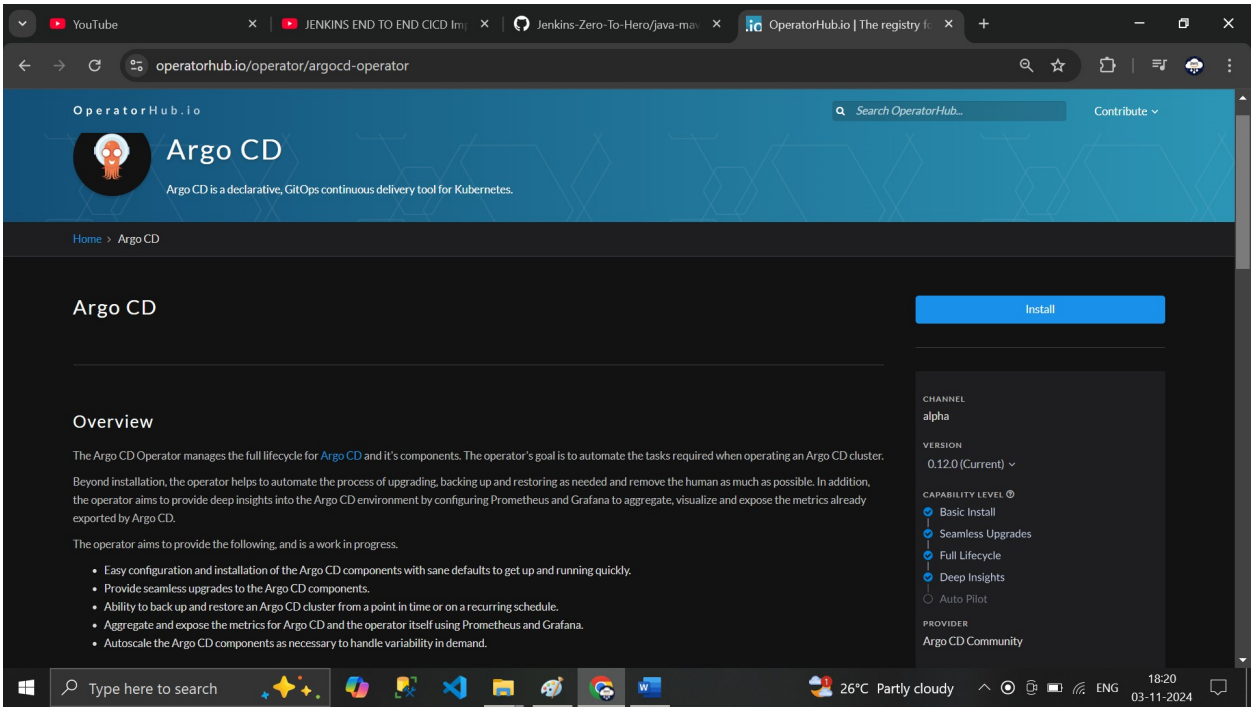

```
root@ip-172-31-87-213:~# docker images
REPOSITORY                          TAG          IMAGE ID       CREATED          SIZE
abhishekf5/ultimate-cicd            1            21eef10a77c5   4 minutes ago    169MB
abhishekf5/maven-abhishek-docker-agent   v1           3fb9145e2467   9 hours ago      913MB
adoptopenjdk/openjdk11              alpine-jre   41f668e6b55c   13 days ago      150MB
root@ip-172-31-87-213:~#
```

 # Final Stage on ArgoCD deploy on Kubernetes

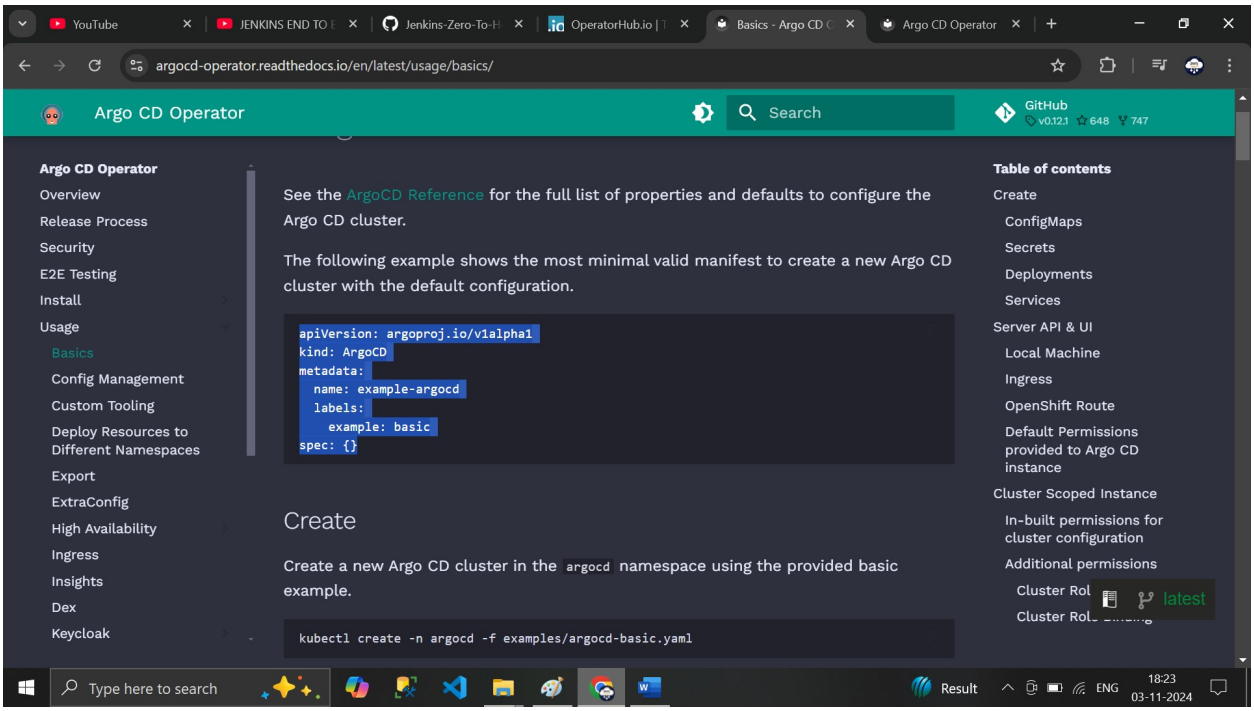


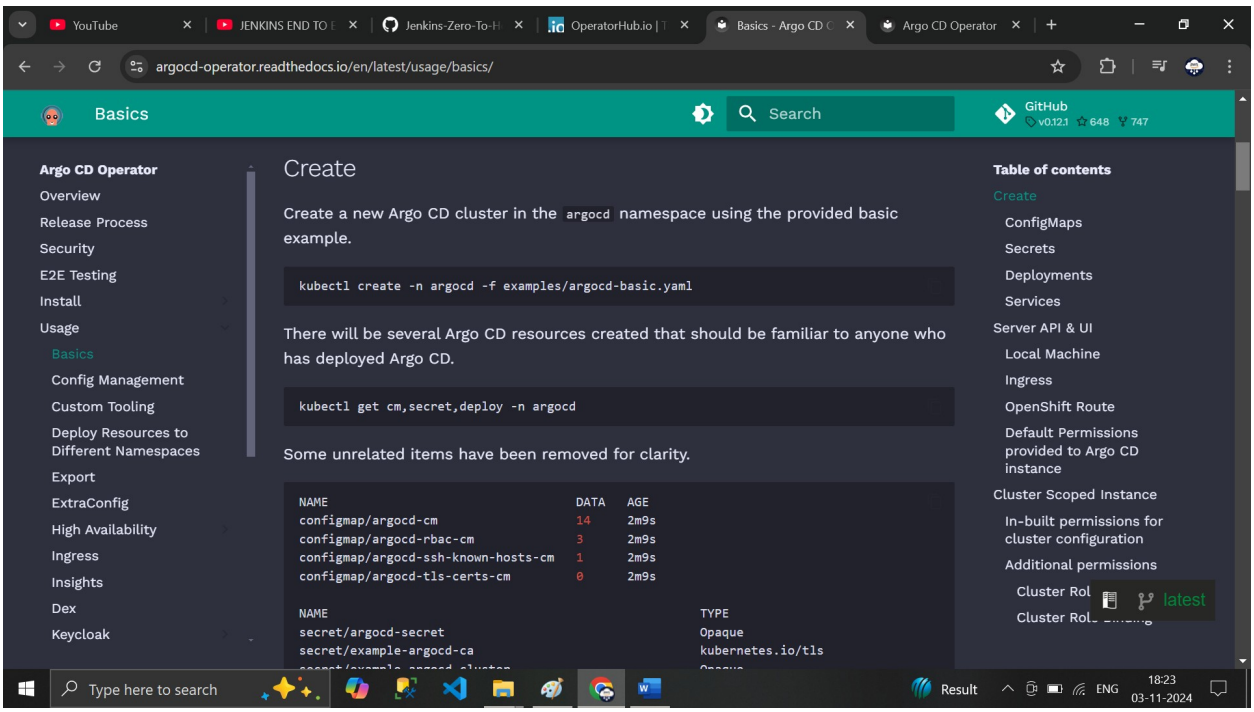
```
→  spring-boot-app git:(main) ✗ minikube status
minikube
type: Control Plane
host: Running
kubelet: Running
apiserver: Running
kubeconfig: Configured

→  spring-boot-app git:(main) ✗
```

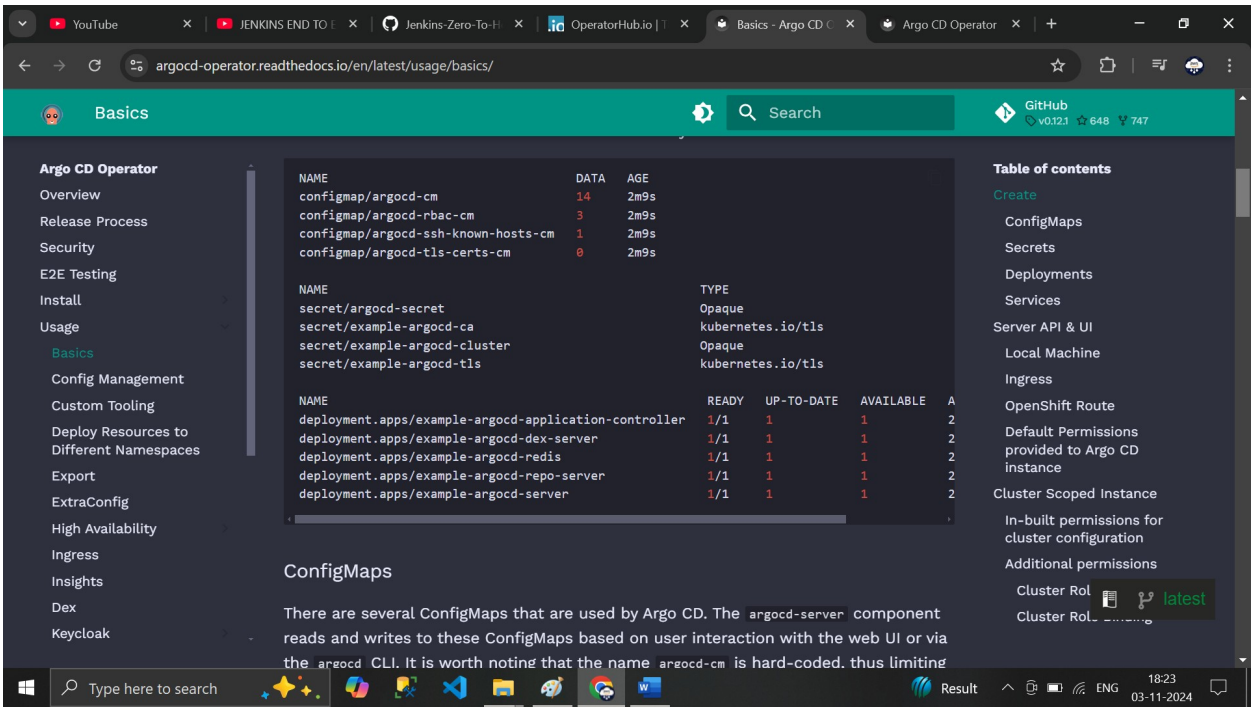
 # https://operatorhub.io/operator/argocd-operator

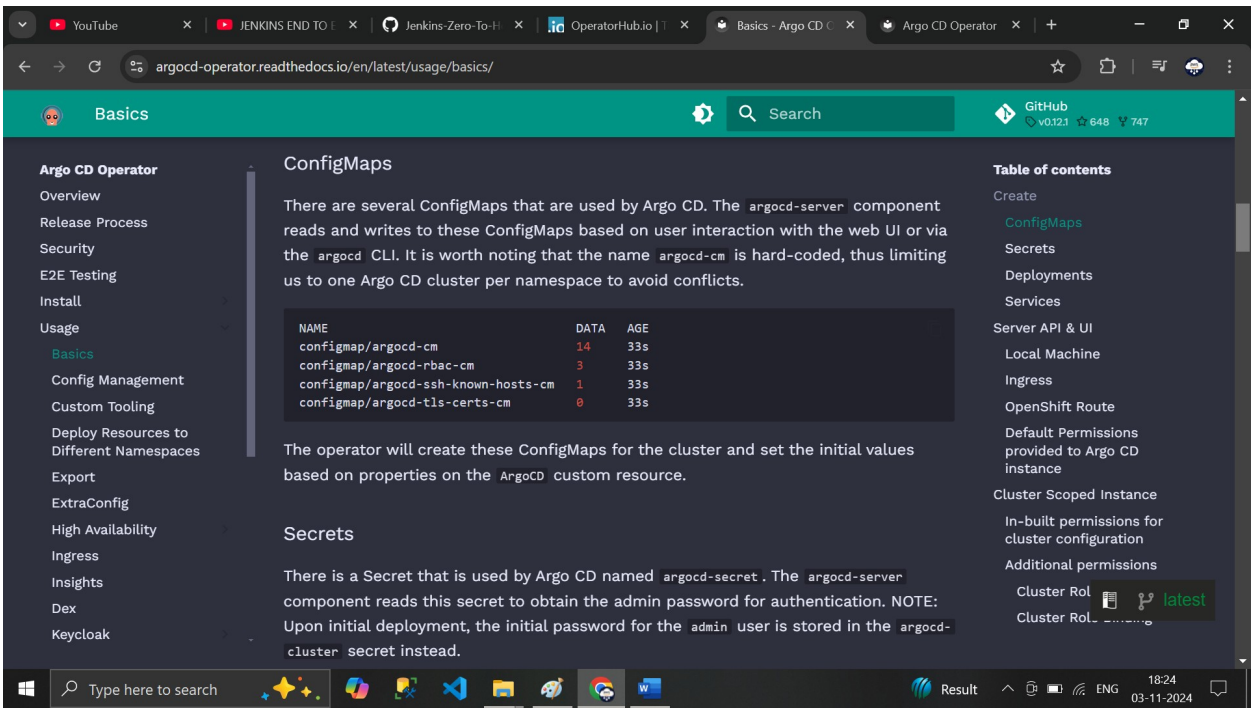


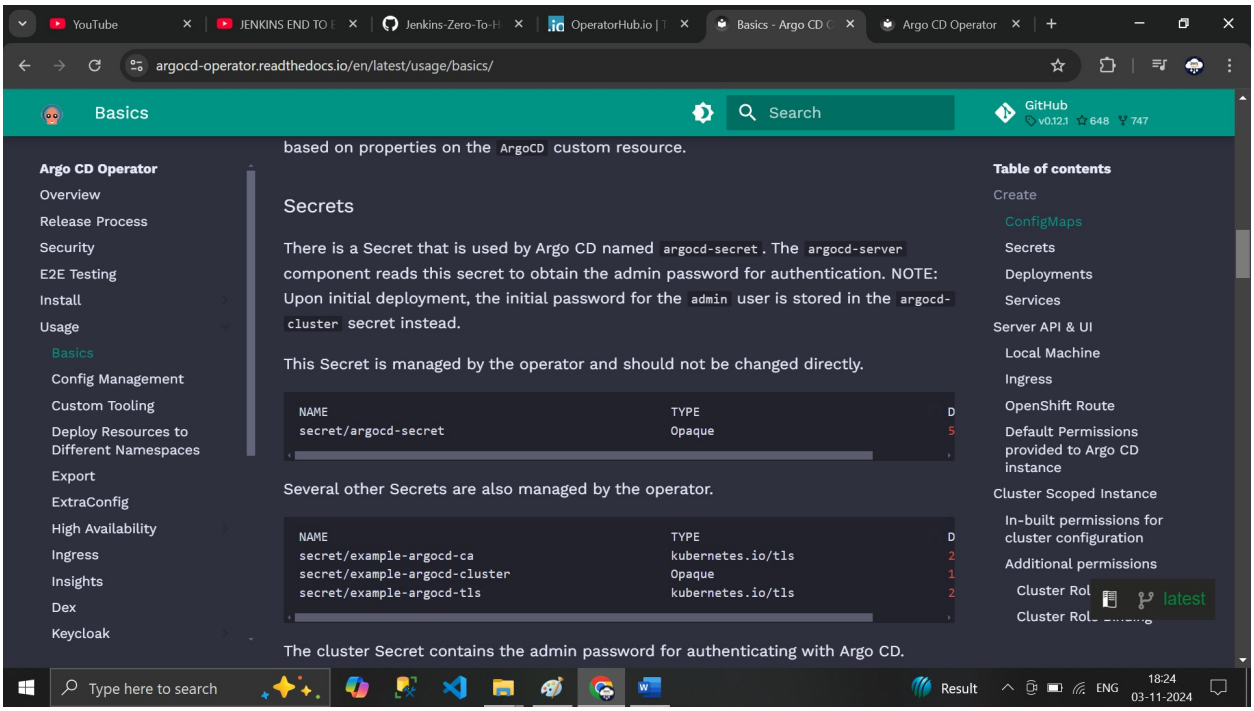
 # Click on operator documentation – usage- Basic ----# Create ArgoCD Controller

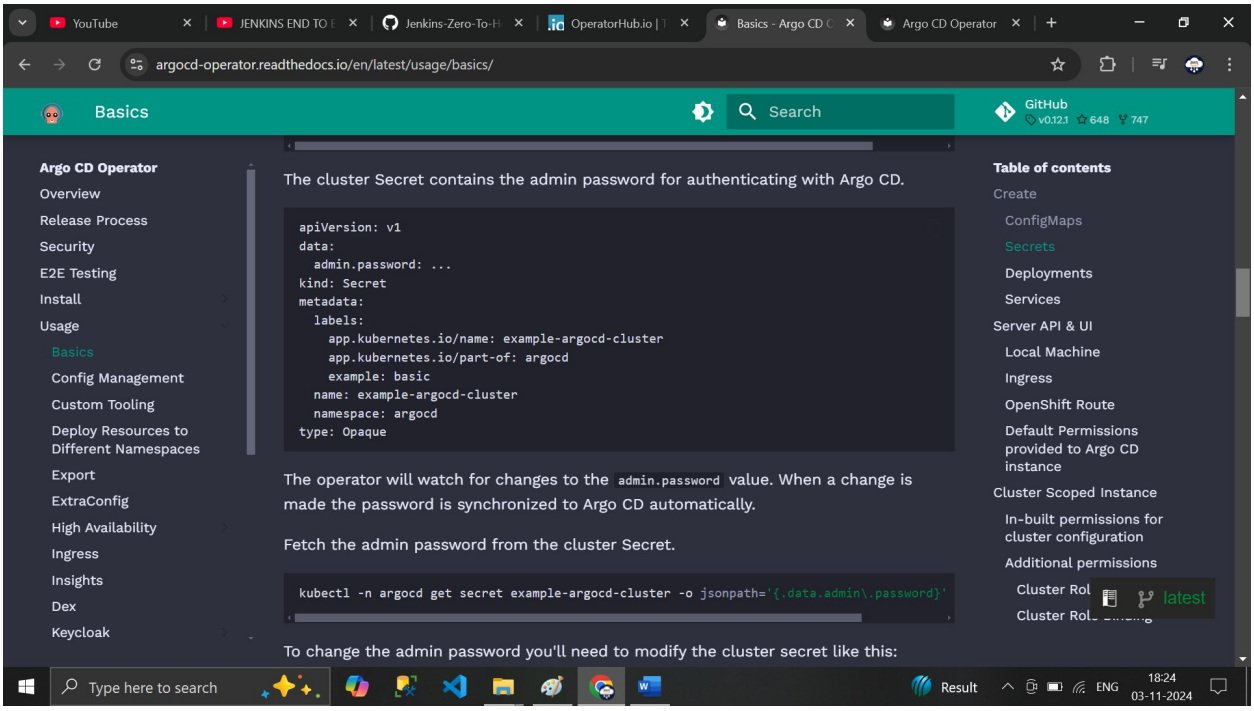

YouTube
JENKINS END TO
Jenkins-Zero-To-H
OperatorHub.io |
Basics - Argo CD
Argo CD Operator
argocd-operator.readthedocs.io/en/latest/usage/basics/
Argo CD Operator
Search
GitHub
v0.12.1 648 747
Argo CD Operator
Overview
Release Process
Security
E2E Testing
Install
Usage
Basics
Config Management
Custom Tooling
Deploy Resources to Different Namespaces
Export
ExtraConfig
High Availability
Ingress
Insights
Dex
Keycloak
See the ArgoCD Reference for the full list of properties and defaults to configure the Argo CD cluster.
The following example shows the most minimal valid manifest to create a new Argo CD cluster with the default configuration.
apiVersion: argoproj.io/v1alpha1
kind: ArgoCD
metadata:
name: example-argocd
labels:
example: basic
spec: {}
Create
Create a new Argo CD cluster in the argocd namespace using the provided basic example.
kubectl create -n argocd -f examples/argocd-basic.yaml
Table of contents
Create
ConfigMaps
Secrets
Deployments
Services
Server API & UI
Local Machine
Ingress
OpenShift Route
Default Permissions provided to Argo CD instance
Cluster Scoped Instance
In-built permissions for cluster configuration
Additional permissions
Cluster Rol
Cluster Rol
latest
Type here to search
Result
ENG
18:23
03-11-2024

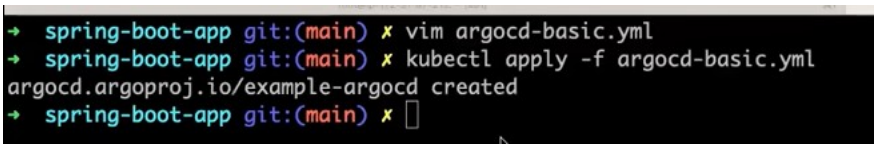

YouTube
JENKINS END TO
Jenkins-Zero-To-H
OperatorHub.io |
Basics - Argo CD
Argo CD Operator
argocd-operator.readthedocs.io/en/latest/usage/basics/
Basics
Search
GitHub
v0.12.1 648 747
Argo CD Operator
Overview
Release Process
Security
E2E Testing
Install
Usage
Basics
Config Management
Custom Tooling
Deploy Resources to Different Namespaces
Export
ExtraConfig
High Availability
Ingress
Insights
Dex
Keycloak
Create
Create a new Argo CD cluster in the argocd namespace using the provided basic example.
kubectl create -n argocd -f examples/argocd-basic.yaml
There will be several Argo CD resources created that should be familiar to anyone who has deployed Argo CD.
kubectl get cm,secret,deploy -n argocd
Some unrelated items have been removed for clarity.
NAME DATA AGE
configmap/argocd-cm 14 2m9s
configmap/argocd-rbac-cm 3 2m9s
configmap/argocd-ssh-known-hosts-cm 1 2m9s
configmap/argocd-tls-certs-cm 0 2m9s
NAME TYPE
secret/argocd-secret Opaque
secret/example-argocd-ca kubernetes.io/tls
Table of contents
Create
ConfigMaps
Secrets
Deployments
Services
Server API & UI
Local Machine
Ingress
OpenShift Route
Default Permissions provided to Argo CD instance
Cluster Scoped Instance
In-built permissions for cluster configuration
Additional permissions
Cluster Rol
Cluster Rol
latest
Type here to search
Result
ENG
18:23
03-11-2024


YouTube
JENKINS END TO E
Jenkins-Zero-To-H
OperatorHub.io | T
Basics - Argo CD
Argo CD Operator
argocd-operator.readthedocs.io/en/latest/usage/basics/
Basics
Search
GitHub
v0.12.1 648 747
Argo CD Operator
Overview
Release Process
Security
E2E Testing
Install
Usage
Basics
Config Management
Custom Tooling
Deploy Resources to Different Namespaces
Export
ExtraConfig
High Availability
Ingress
Insights
Dex
Keycloak
NAME DATA AGE
configmap/argocd-cm 14 2m9s
configmap/argocd-rbac-cm 3 2m9s
configmap/argocd-ssh-known-hosts-cm 1 2m9s
configmap/argocd-tls-certs-cm 0 2m9s
NAME TYPE
secret/argocd-secret Opaque
secret/example-argocd-ca kubernetes.io/tls
secret/example-argocd-cluster Opaque
secret/example-argocd-tls kubernetes.io/tls
NAME READY UP-TO-DATE AVAILABLE
deployment.apps/example-argocd-application-controller 1/1 1 1
deployment.apps/example-argocd-dex-server 1/1 1 1
deployment.apps/example-argocd-redis 1/1 1 1
deployment.apps/example-argocd-repo-server 1/1 1 1
deployment.apps/example-argocd-server 1/1 1 1
ConfigMaps
There are several ConfigMaps that are used by Argo CD. The argocd-server component reads and writes to these ConfigMaps based on user interaction with the web UI or via the argocd CLI. It is worth noting that the name argocd-cm is hard-coded, thus limiting
Table of contents
Create
ConfigMaps
Secrets
Deployments
Services
Server API & UI
Local Machine
Ingress
OpenShift Route
Default Permissions provided to Argo CD instance
Cluster Scoped Instance
In-built permissions for cluster configuration
Additional permissions
Cluster Rol
Cluster Rol
latest
Type here to search
Result
ENG
18:23
03-11-2024


YouTube
JENKINS END TO E
Jenkins-Zero-To-H
OperatorHub.io | T
Basics - Argo CD
Argo CD Operator
argocd-operator.readthedocs.io/en/latest/usage/basics/
Basics
Search
GitHub
v0.12.1 648 747
Argo CD Operator
Overview
Release Process
Security
E2E Testing
Install
Usage
Basics
Config Management
Custom Tooling
Deploy Resources to Different Namespaces
Export
ExtraConfig
High Availability
Ingress
Insights
Dex
Keycloak
ConfigMaps
There are several ConfigMaps that are used by Argo CD. The argocd-server component reads and writes to these ConfigMaps based on user interaction with the web UI or via the argocd CLI. It is worth noting that the name argocd-cm is hard-coded, thus limiting us to one Argo CD cluster per namespace to avoid conflicts.
NAME DATA AGE
configmap/argocd-cm 14 33s
configmap/argocd-rbac-cm 3 33s
configmap/argocd-ssh-known-hosts-cm 1 33s
configmap/argocd-tls-certs-cm 0 33s
The operator will create these ConfigMaps for the cluster and set the initial values based on properties on the ArgoCD custom resource.
Secrets
There is a Secret that is used by Argo CD named argocd-secret . The argocd-server component reads this secret to obtain the admin password for authentication. NOTE: Upon initial deployment, the initial password for the admin user is stored in the argocd-cluster secret instead.
Table of contents
Create
ConfigMaps
Secrets
Deployments
Services
Server API & UI
Local Machine
Ingress
OpenShift Route
Default Permissions provided to Argo CD instance
Cluster Scoped Instance
In-built permissions for cluster configuration
Additional permissions
Cluster Rol
Cluster Rol
latest
Type here to search
Result
ENG
18:24
03-11-2024


YouTube
JENKINS END TO E
Jenkins-Zero-To-H
OperatorHub.io | T
Basics - Argo CD
Argo CD Operator
argocd-operator.readthedocs.io/en/latest/usage/basics/
Basics
Search
GitHub
v0.12.1 648 747
Argo CD Operator
Overview
Release Process
Security
E2E Testing
Install
Usage
Basics
Config Management
Custom Tooling
Deploy Resources to Different Namespaces
Export
ExtraConfig
High Availability
Ingress
Insights
Dex
Keycloak
based on properties on the ArgoCD custom resource.
Secrets
There is a Secret that is used by Argo CD named argocd-secret. The argocd-server component reads this secret to obtain the admin password for authentication. NOTE: Upon initial deployment, the initial password for the admin user is stored in the argocd-cluster secret instead.
This Secret is managed by the operator and should not be changed directly.
NAME TYPE D
secret/argocd-secret Opaque 5
Several other Secrets are also managed by the operator.
NAME TYPE D
secret/example-argocd-ca kubernetes.io/tls 2
secret/example-argocd-cluster Opaque 1
secret/example-argocd-tls kubernetes.io/tls 2
The cluster Secret contains the admin password for authenticating with Argo CD.
Table of contents
Create
ConfigMaps
Secrets
Deployments
Services
Server API & UI
Local Machine
Ingress
OpenShift Route
Default Permissions provided to Argo CD instance
Cluster Scoped Instance
In-built permissions for cluster configuration
Additional permissions
Cluster Rol
Cluster Rol
latest
Type here to search
Result
ENG
18:24
03-11-2024

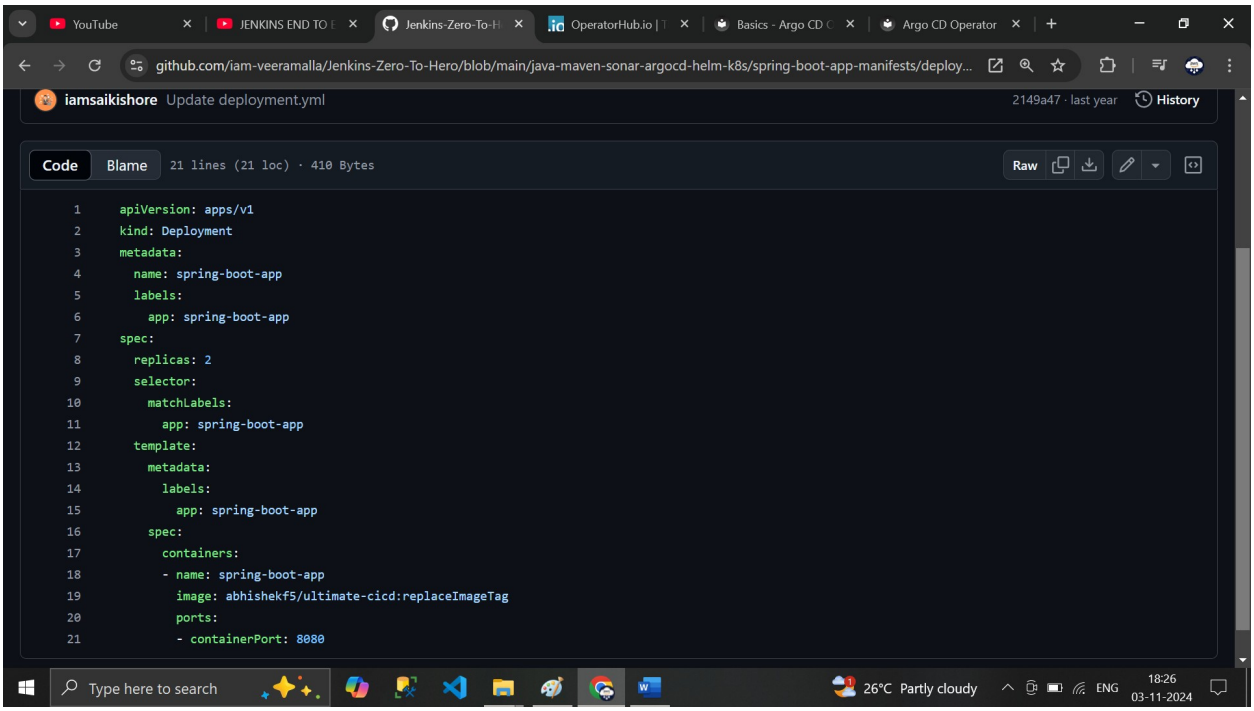

YouTube
JENKINS END TO E
Jenkins-Zero-To-H
OperatorHub.io | T
Basics - Argo CD
Argo CD Operator
argocd-operator.readthedocs.io/en/latest/usage/basics/
Basics
Search
GitHub
v0.12.1 648 747
Argo CD Operator
Overview
Release Process
Security
E2E Testing
Install
Usage
Basics
Config Management
Custom Tooling
Deploy Resources to Different Namespaces
Export
ExtraConfig
High Availability
Ingress
Insights
Dex
Keycloak
The cluster Secret contains the admin password for authenticating with Argo CD.
apiVersion: v1
data:
admin.password: ...
kind: Secret
metadata:
labels:
app.kubernetes.io/name: example-argocd-cluster
app.kubernetes.io/part-of: argocd
example: basic
name: example-argocd-cluster
namespace: argocd
type: Opaque
The operator will watch for changes to the admin.password value. When a change is made the password is synchronized to Argo CD automatically.
Fetch the admin password from the cluster Secret.
kubectl -n argocd get secret example-argocd-cluster -o jsonpath='{.data.admin\.password}'
To change the admin password you'll need to modify the cluster secret like this:
Table of contents
Create
ConfigMaps
Secrets
Deployments
Services
Server API & UI
Local Machine
Ingress
OpenShift Route
Default Permissions provided to Argo CD instance
Cluster Scoped Instance
In-built permissions for cluster configuration
Additional permissions
Cluster Rol
Cluster Rol
latest
Type here to search
Result
ENG
18:24
03-11-2024


spring-boot-app git:(main) x vim argocd-basic.yml
spring-boot-app git:(main) x kubectl apply -f argocd-basic.yml
argocd.argoproj.io/example-argocd created
spring-boot-app git:(main) x

```
apiVersion: argoproj.io/v1alpha1
kind: ArgoCD
metadata:
  name: example-argocd
  labels:
    example: basic
spec: {}
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
-- INSERT --
```

```
➜  spring-boot-app git:(main) ✗ kubectl get pods
NAME                                          READY   STATUS              RESTARTS   AGE
example-argocd-application-controller-0       0/1     ContainerCreating   0          17s
example-argocd-redis-6db56fc79f-fps8w         0/1     ContainerCreating   0          18s
example-argocd-repo-server-6b78f58dc4-vq9t8   0/1     Init:0/1            0          18s
example-argocd-server-846cdb99cc-p4jdc        0/1     ContainerCreating   0          18s
➜  spring-boot-app git:(main) ✗
```

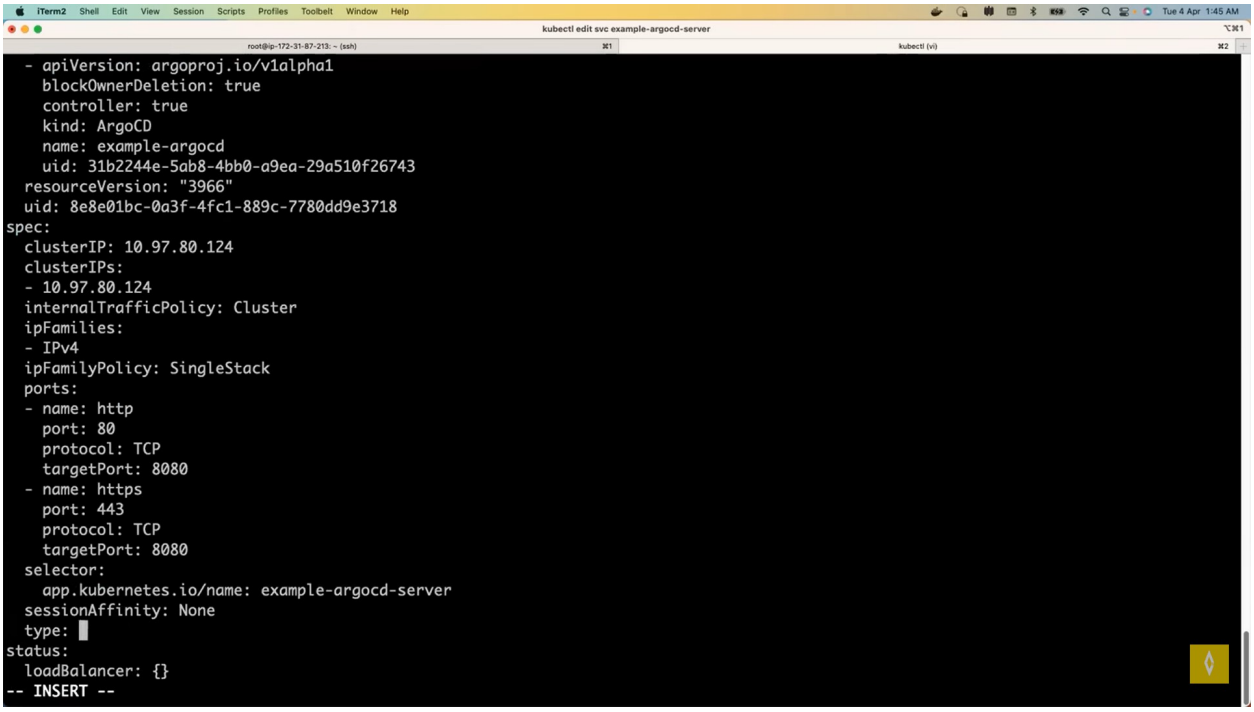
 # now Final Step .. argocd pull latest image from the git Deployment.yml file




```
→  spring-boot-app git:(main) ✗ kubectl get pods
NAME                                      READY   STATUS              RESTARTS   AGE
example-argocd-application-controller-0   0/1     ContainerCreating   0          17s
example-argocd-redis-6db56fc79f-fps8w     0/1     ContainerCreating   0          18s
example-argocd-repo-server-6b78f58dc4-vq9t8   0/1     Init:0/1            0          18s
example-argocd-server-846cdb99cc-p4jdc    0/1     ContainerCreating   0          18s
→  spring-boot-app git:(main) ✗ kubectl get pods  -w
NAME                                      READY   STATUS              RESTARTS   AGE
example-argocd-application-controller-0   0/1     ContainerCreating   0          50s
example-argocd-redis-6db56fc79f-fps8w     0/1     ContainerCreating   0          51s
example-argocd-repo-server-6b78f58dc4-vq9t8   0/1     Init:0/1            0          51s
example-argocd-server-846cdb99cc-p4jdc    0/1     ContainerCreating   0          51s
^C
→  spring-boot-app git:(main) ✗ kubectl get svc
NAME                              TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)             AGE
example-argocd-metrics            ClusterIP   10.104.135.129   <none>        8082/TCP            65s
example-argocd-redis              ClusterIP   10.109.78.67     <none>        6379/TCP            64s
example-argocd-repo-server        ClusterIP   10.109.56.71     <none>        8081/TCP,8084/TCP   64s
example-argocd-server             ClusterIP   10.97.80.124     <none>        80/TCP,443/TCP      64s
example-argocd-server-metrics     ClusterIP   10.100.182.196   <none>        8083/TCP            64s
kubernetes                        ClusterIP   10.96.0.1        <none>        443/TCP             33m
```

 # Change type: from Clusterip to nodeport --mode

```
➜  spring-boot-app git:(main) ✗ kubectl edit svc example-argocd-server
service/example-argocd-server edited
➜  spring-boot-app git:(main) ✗
```



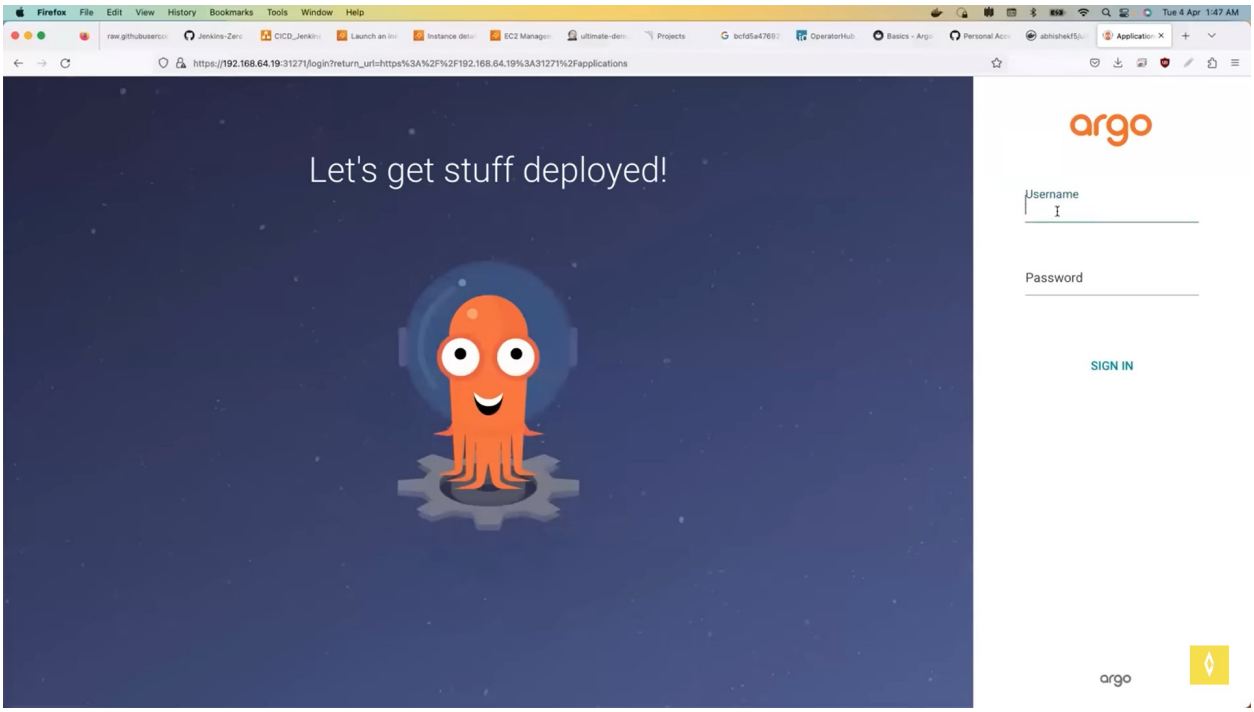

```
➜  spring-boot-app git:(main) ✗ kubectl get svc
NAME                          TYPE        CLUSTER-IP       EXTERNAL-IP   PORT(S)                      AGE
example-argocd-metrics        ClusterIP   10.104.135.129   <none>        8082/TCP                     100s
example-argocd-redis          ClusterIP   10.109.78.67     <none>        6379/TCP                     99s
example-argocd-repo-server    ClusterIP   10.109.56.71     <none>        8081/TCP,8084/TCP            99s
example-argocd-server         NodePort    10.97.80.124     <none>        80:31271/TCP,443:32471/TCP   99s
example-argocd-server-metrics ClusterIP   10.100.182.196   <none>        8083/TCP                     99s
kubernetes                    ClusterIP   10.96.0.1        <none>        443/TCP                      33m
➜  spring-boot-app git:(main) ✗
```

 # now Execute it on Browser - by using minikube port forwarding

```
→  spring-boot-app git:(main) ✗ minikube service argocd-server
→  spring-boot-app git:(main) ✗ minikube service list
|-------------|-----------------------------------------------|--------------|---------------------------|
|  NAMESPACE  |                     NAME                      | TARGET PORT  |            URL            |
|-------------|-----------------------------------------------|--------------|---------------------------|
| default     | example-argocd-metrics                        | No node port |                           |
| default     | example-argocd-redis                          | No node port |                           |
| default     | example-argocd-repo-server                    | No node port |                           |
| default     | example-argocd-server                         | http/80      | http://192.168.64.19:31271 |
|             |                                               | https/443    | http://192.168.64.19:32471 |
| default     | example-argocd-server-metrics                 | No node port |                           |
| default     | kubernetes                                    | No node port |                           |
| kube-system | kube-dns                                      | No node port |                           |
| olm         | operatorhubio-catalog                         | No node port |                           |
| olm         | packageserver-service                         | No node port |                           |
| operators   | argocd-operator-controller-manager-metrics-service | No node port |                           |
|-------------|-----------------------------------------------|--------------|---------------------------|
→  spring-boot-app git:(main) ✗
```

 # Check pods are up

```
→  spring-boot-app git:(main) ✗ kubectl get pods
NAME                                  READY   STATUS    RESTARTS   AGE
example-argocd-application-controller-0   1/1     Running   0          2m26s
example-argocd-redis-6db56fc79f-fps8w     1/1     Running   0          2m27s
example-argocd-repo-server-6b78f58dc4-vq9t8   1/1     Running   0          2m27s
example-argocd-server-846cdb99cc-p4jdc    1/1     Running   0          2m27s
→  spring-boot-app git:(main) ✗
```



 # password

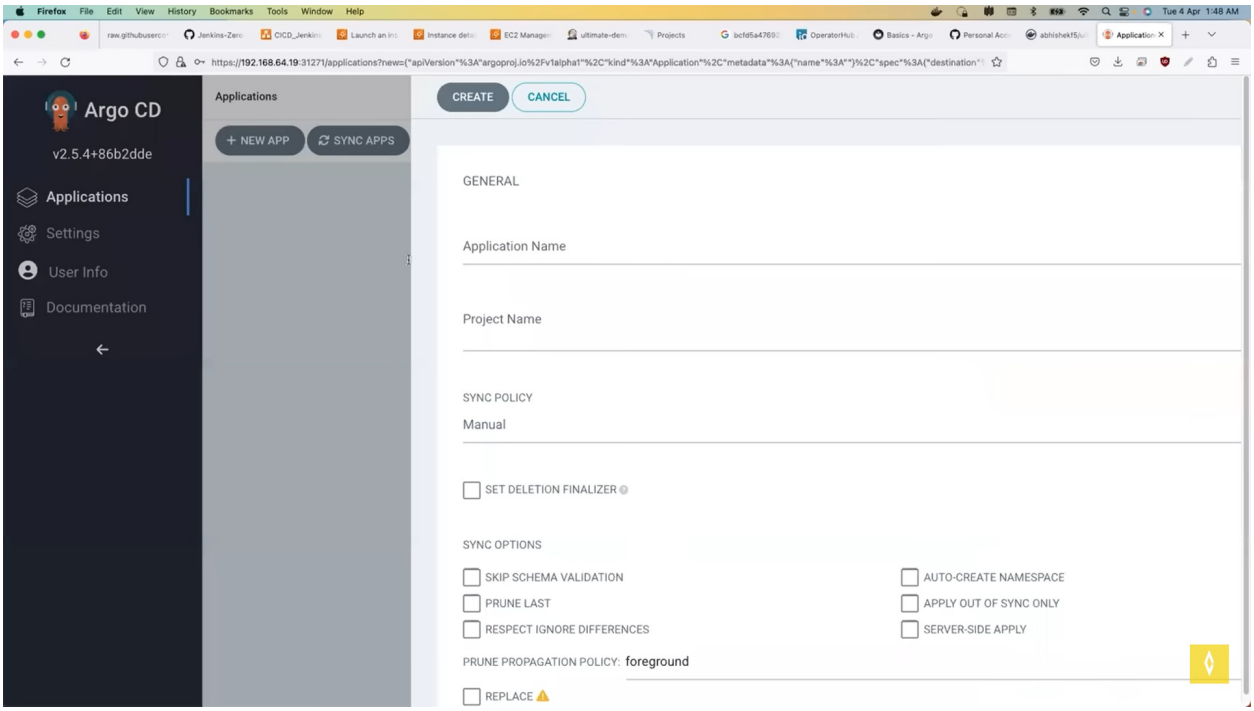
Argocd stores password in cluster

```
→  spring-boot-app git:(main) ✗ kubectl get secret
NAME                                                  TYPE                                  DATA   AGE
argocd-secret                                         Opaque                                5      3m6s
default-token-s9zf5                                   kubernetes.io/service-account-token   3      34m
example-argocd-argocd-application-controller-token-k9q4v   kubernetes.io/service-account-token   3      3m7s
example-argocd-argocd-grafana-token-gl9qp             kubernetes.io/service-account-token   3      3m6s
example-argocd-argocd-redis-ha-token-59fwd            kubernetes.io/service-account-token   3      3m7s
example-argocd-argocd-redis-token-jkd8q               kubernetes.io/service-account-token   3      3m6s
example-argocd-argocd-server-token-jf87l              kubernetes.io/service-account-token   3      3m7s
example-argocd-ca                                     kubernetes.io/tls                     3      3m6s
example-argocd-cluster                                Opaque                                1      3m6s
example-argocd-default-cluster-config                 Opaque                                4      3m6s
example-argocd-tls                                    kubernetes.io/tls                     2      3m6s
→  spring-boot-app git:(main) ✗
```

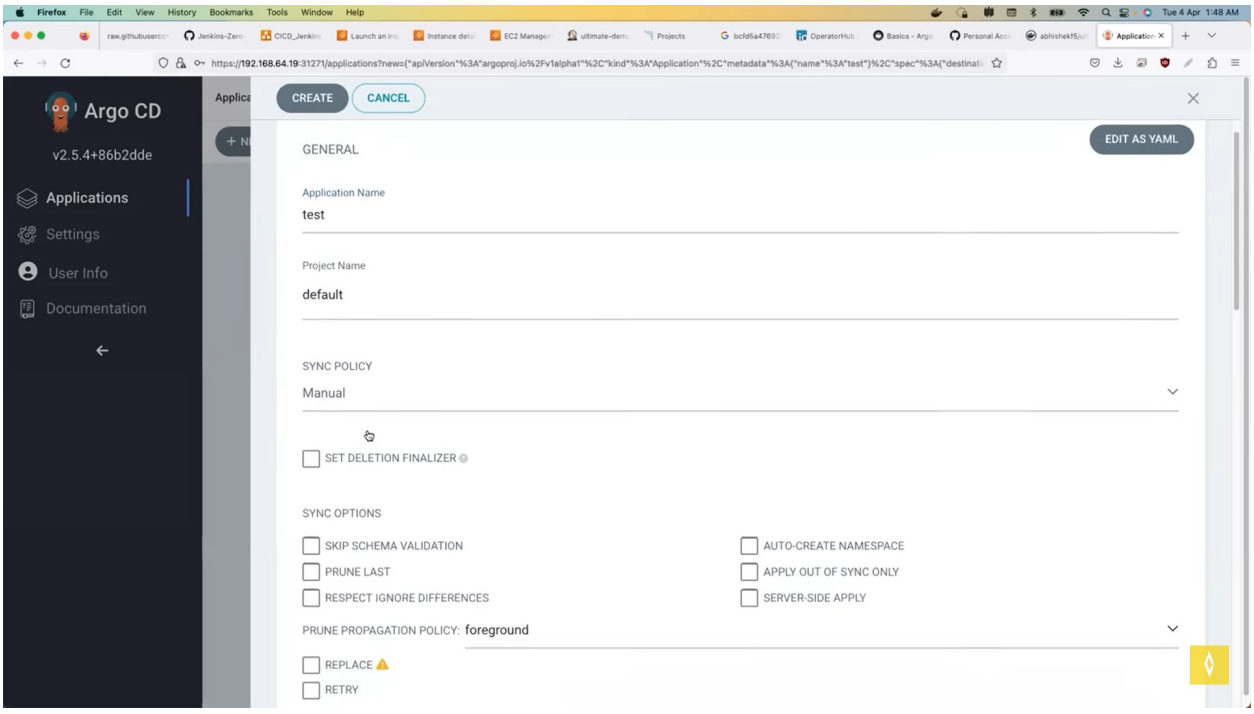
```
spring-boot-app git:(main) ✗ kubectl edit secret example-argocd-cluster
```

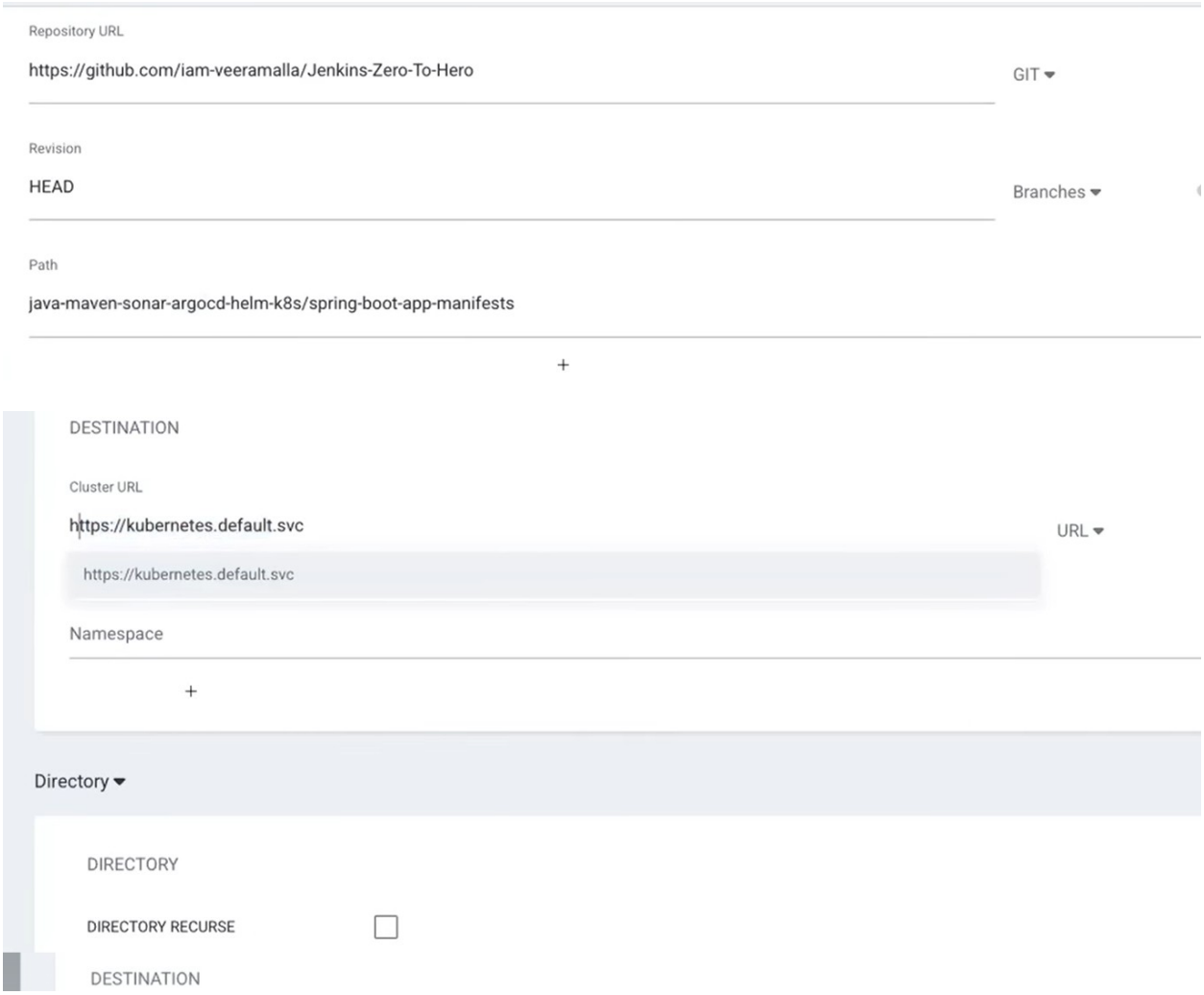
```
# Please edit the object below. Lines beginning with a '#' will be ignored,
# and an empty file will abort the edit. If an error occurs while saving this file will be
# reopened with the relevant failures.
#
apiVersion: v1
data:
  admin.password: cFlyQ2ZneXM4WjdUT2o2RWRBTGxKdFVRU240aTJES2E=
kind: Secret
metadata:
  creationTimestamp: "2023-04-03T20:14:14Z"
  labels:
    app.kubernetes.io/managed-by: example-argocd
    app.kubernetes.io/name: example-argocd-cluster
    app.kubernetes.io/part-of: argocd
  name: example-argocd-cluster
  namespace: default
  ownerReferences:
  - apiVersion: argoproj.io/v1alpha1
    blockOwnerDeletion: true
    controller: true
    kind: ArgoCD
    name: example-argocd
    uid: 31b2244e-5ab8-4bb0-a9ea-29a510f26743
  resourceVersion: "3940"
  uid: 6dcad65f-44ba-43c8-b351-474d594608ab
type: Opaque
~
~
~
~
~
"/var/folders/v6/8dw9y3q57x1247_vdg4812hm0000gn/T/kubectl-edit-vwpcu.yaml" 26L, 836B
```

```
→  spring-boot-app git:(main) ✗ echo cFlyQ2ZneXM4WjdUT2o2RWRBTGxKdFVRU240aTJES2E= | base64 -d
pYrCfgys8Z7TOj6EdALlJtUQSn4i2DKa%
→  spring-boot-app git:(main) ✗ echo -n cFlyQ2ZneXM4WjdUT2o2RWRBTGxKdFVRU240aTJES2E= | base64 -d
pYrCfgys8Z7TOj6EdALlJtUQSn4i2DKa%
→  spring-boot-app git:(main) ✗
```

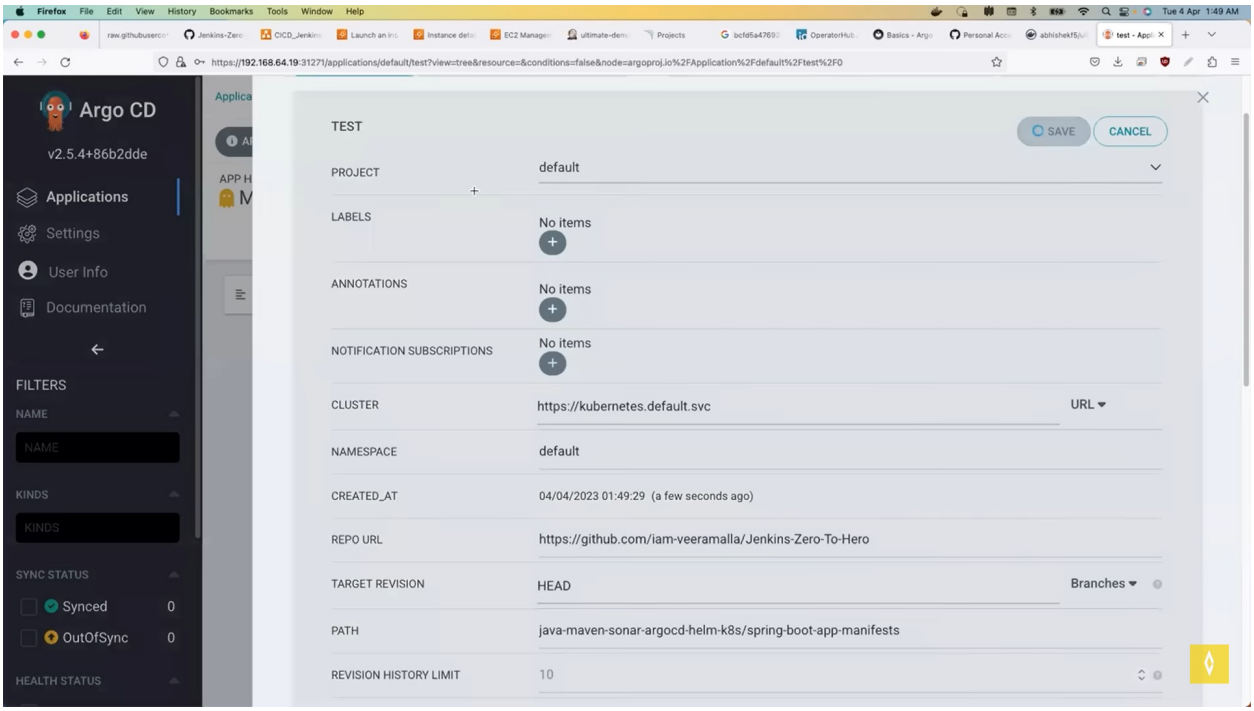

Argo CD
v2.5.4+86b2dde
Applications
Settings
User Info
Documentation
Applications
+ NEW APP
SYNC APPS
CREATE
CANCEL
GENERAL
Application Name
Project Name
SYNC POLICY
Manual
SET DELETION FINALIZER
SYNC OPTIONS
SKIP SCHEMA VALIDATION
AUTO-CREATE NAMESPACE
PRUNE LAST
APPLY OUT OF SYNC ONLY
RESPECT IGNORE DIFFERENCES
SERVER-SIDE APPLY
PRUNE PROPAGATION POLICY: foreground
REPLACE

 # Create app

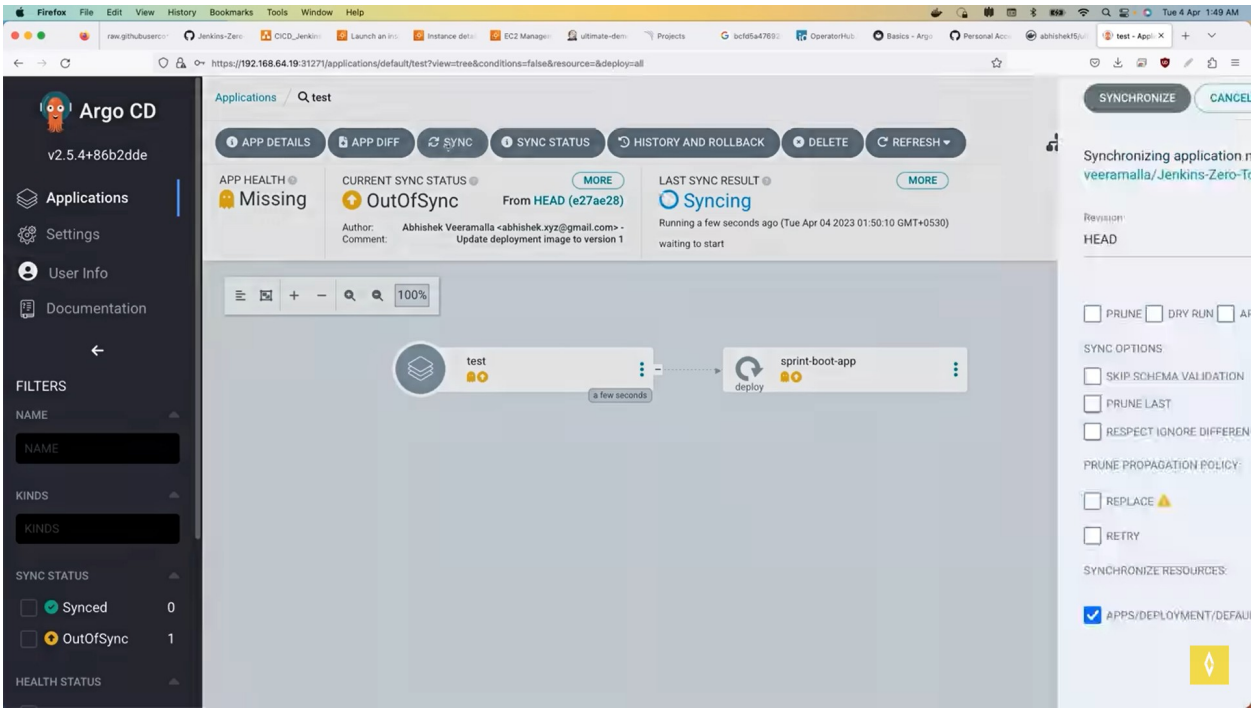

Argo CD
v2.5.4+86b2dde
Applications
Settings
User Info
Documentation
CREATE
CANCEL
EDIT AS YAML
GENERAL
Application Name
test
Project Name
default
SYNC POLICY
Manual
SET DELETION FINALIZER
SYNC OPTIONS
SKIP SCHEMA VALIDATION
AUTO-CREATE NAMESPACE
PRUNE LAST
APPLY OUT OF SYNC ONLY
RESPECT IGNORE DIFFERENCES
SERVER-SIDE APPLY
PRUNE PROPAGATION POLICY: foreground
REPLACE
RETRY

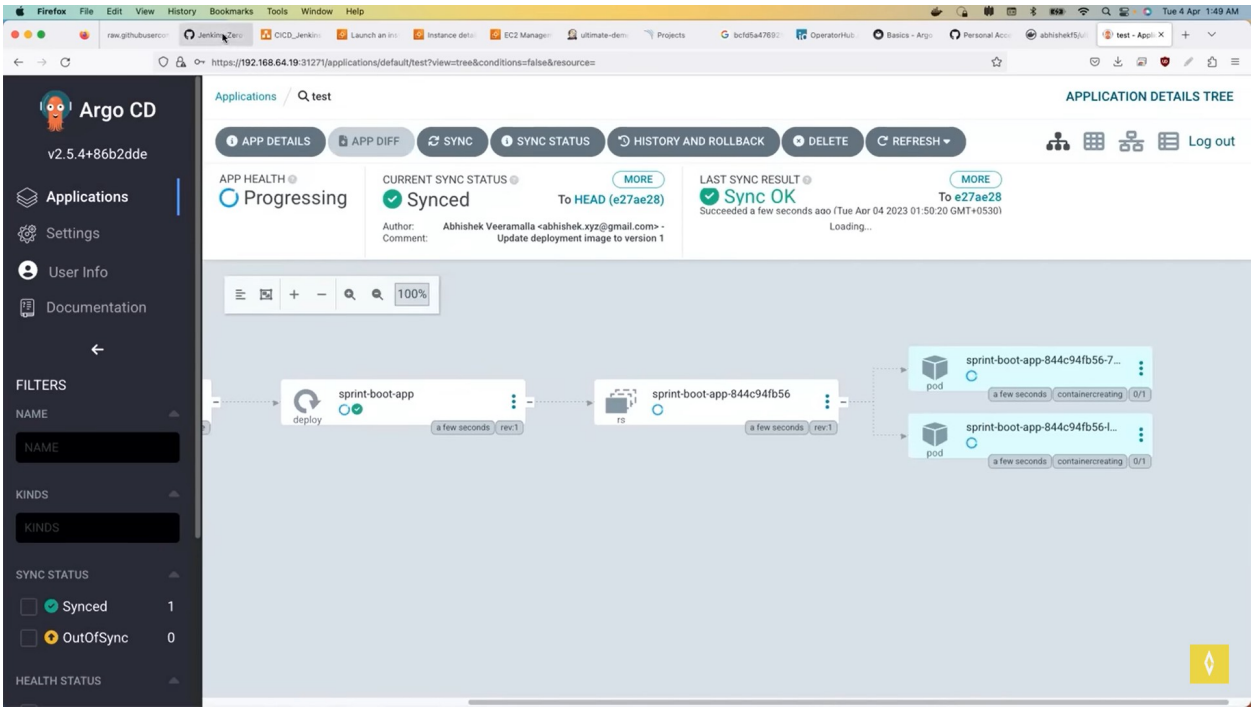


 # Click on Create



 # Click on Sync






```
→  spring-boot-app git:(main) ✗ kubectl get deploy
NAME                          READY   UP-TO-DATE   AVAILABLE   AGE
example-argocd-redis          1/1     1            1           6m55s
example-argocd-repo-server    1/1     1            1           6m55s
example-argocd-server         1/1     1            1           6m55s
sprint-boot-app               2/2     2            2           59s
→  spring-boot-app git:(main) ✗
```

```
→  spring-boot-app git:(main) ✗ kubectl get pods
NAME                                          READY   STATUS    RESTARTS   AGE
example-argocd-application-controller-0       1/1     Running   0          7m
example-argocd-redis-6db56fc79f-fps8w         1/1     Running   0          7m1s
example-argocd-repo-server-6b78f58dc4-vq9t8   1/1     Running   0          7m1s
example-argocd-server-846cdb99cc-p4jdc        1/1     Running   0          7m1s
sprint-boot-app-844c94fb56-72t67              1/1     Running   0          65s
sprint-boot-app-844c94fb56-lhmvb              1/1     Running   0          65s
→  spring-boot-app git:(main) ✗
```

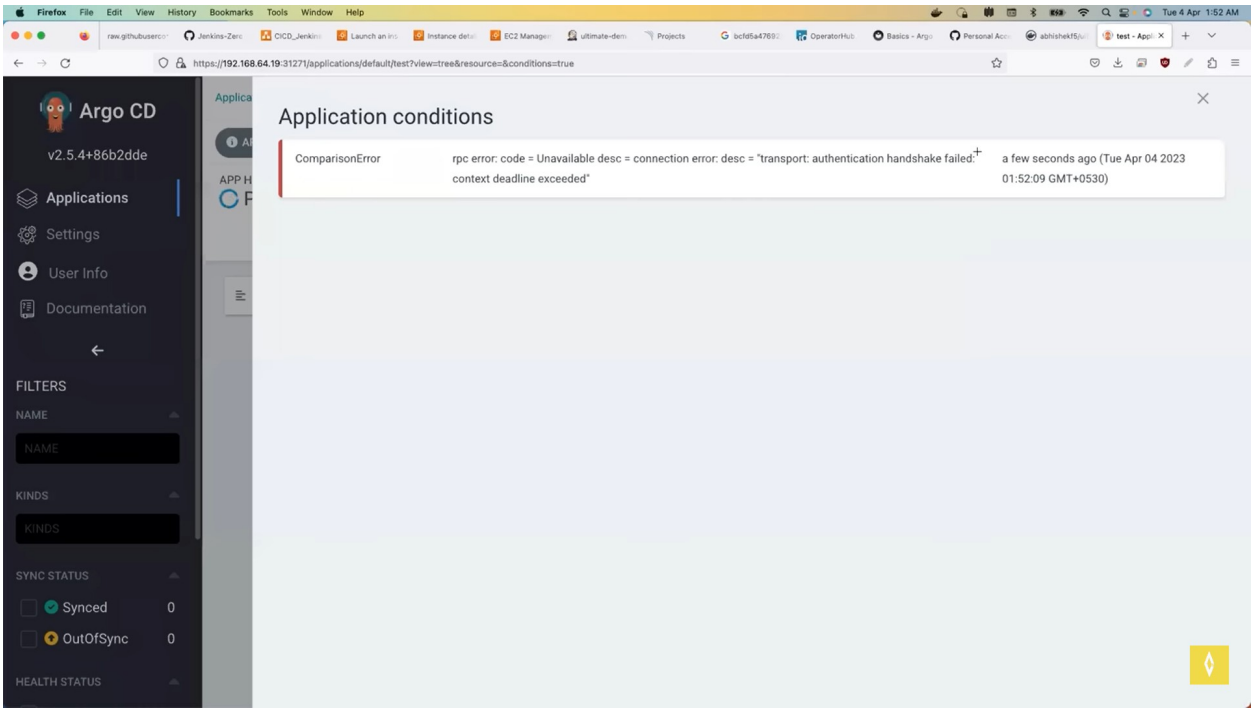
 # test ..

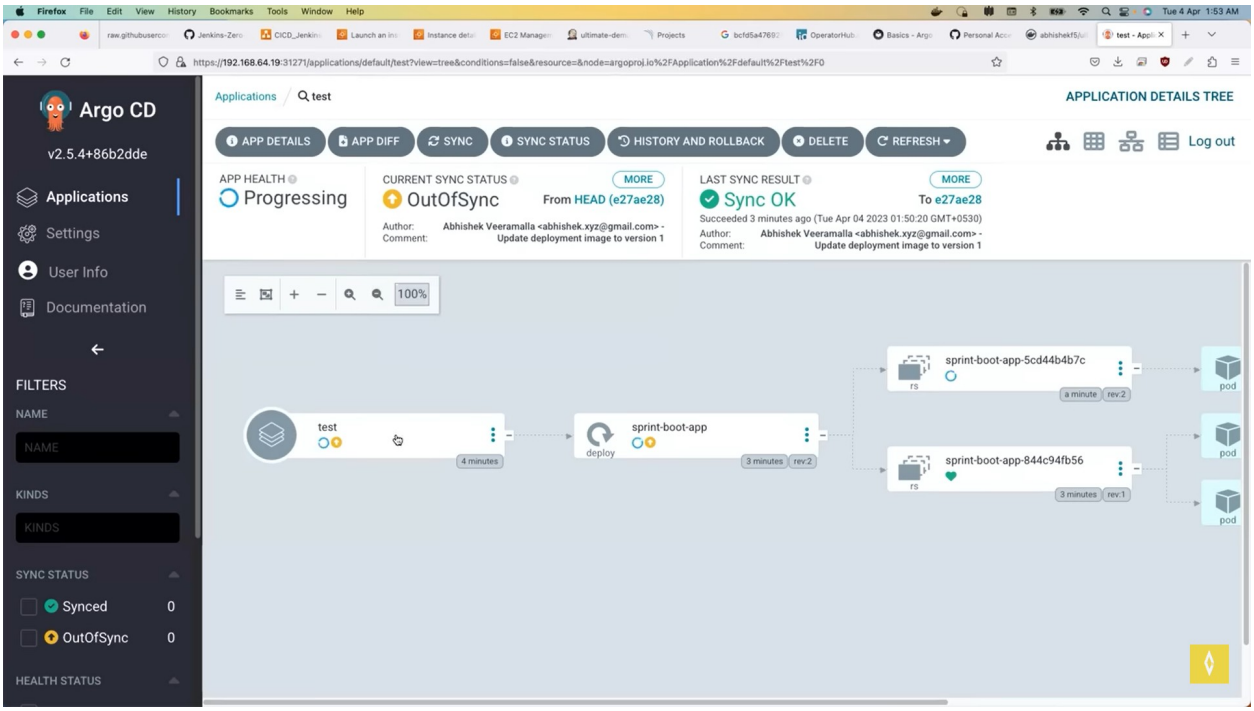
```
→  spring-boot-app git:(main) ✗ ls
Dockerfile      README.md       argocd-basic.yml pom.xml        src             target
→  spring-boot-app git:(main) ✗ kubectl edit deploy sprint-boot-app
```

 # in spring boot app .. will change some data .. see the reflection in ArgoCD

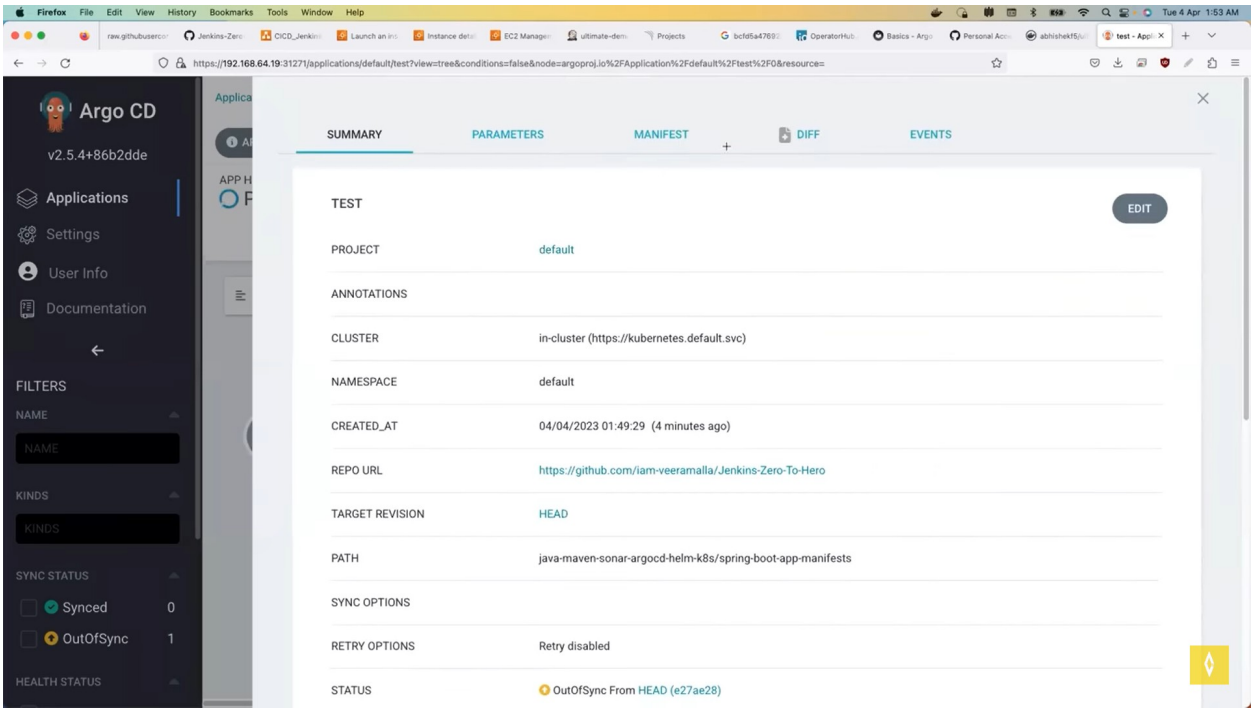
```
  metadata:
    creationTimestamp: null
    labels:
      app: sprint-boot-app
  spec:
    containers:
    - image: abhishekf5/ultimate-cicd:2
      imagePullPolicy: IfNotPresent
      name: sprint-boot-app
      ports:
      - containerPort: 8080
        protocol: TCP
      resources: {}
      terminationMessagePath: /dev/termination-log
      terminationMessagePolicy: File
    dnsPolicy: ClusterFirst
    restartPolicy: Always
    schedulerName: default-scheduler
    securityContext: {}
    terminationGracePeriodSeconds: 30
status:
  availableReplicas: 2
  conditions:
  - lastTransitionTime: "2023-04-03T20:21:01Z"
    lastUpdateTime: "2023-04-03T20:21:01Z"
    message: Deployment has minimum availability.
    reason: MinimumReplicasAvailable
    status: "True"
    type: Available
  - lastTransitionTime: "2023-04-03T20:20:11Z"
    lastUpdateTime: "2023-04-03T20:21:01Z"
```

 # Refresh ArgoCD – Error

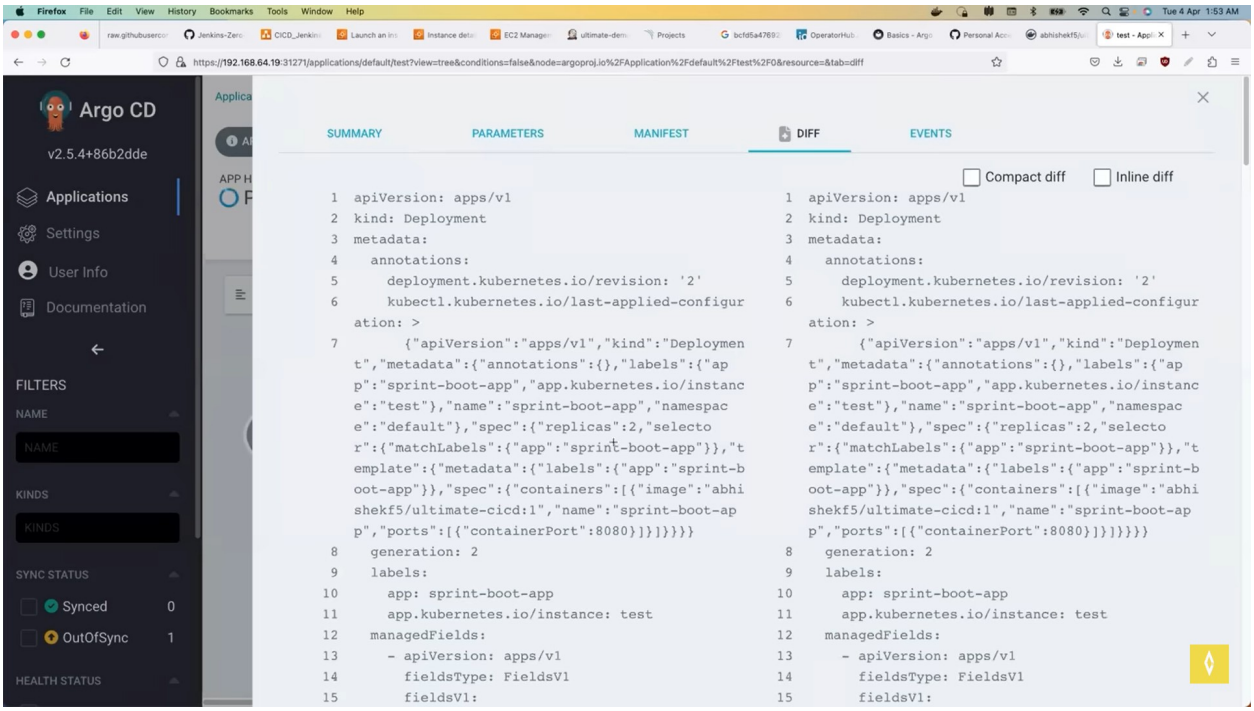


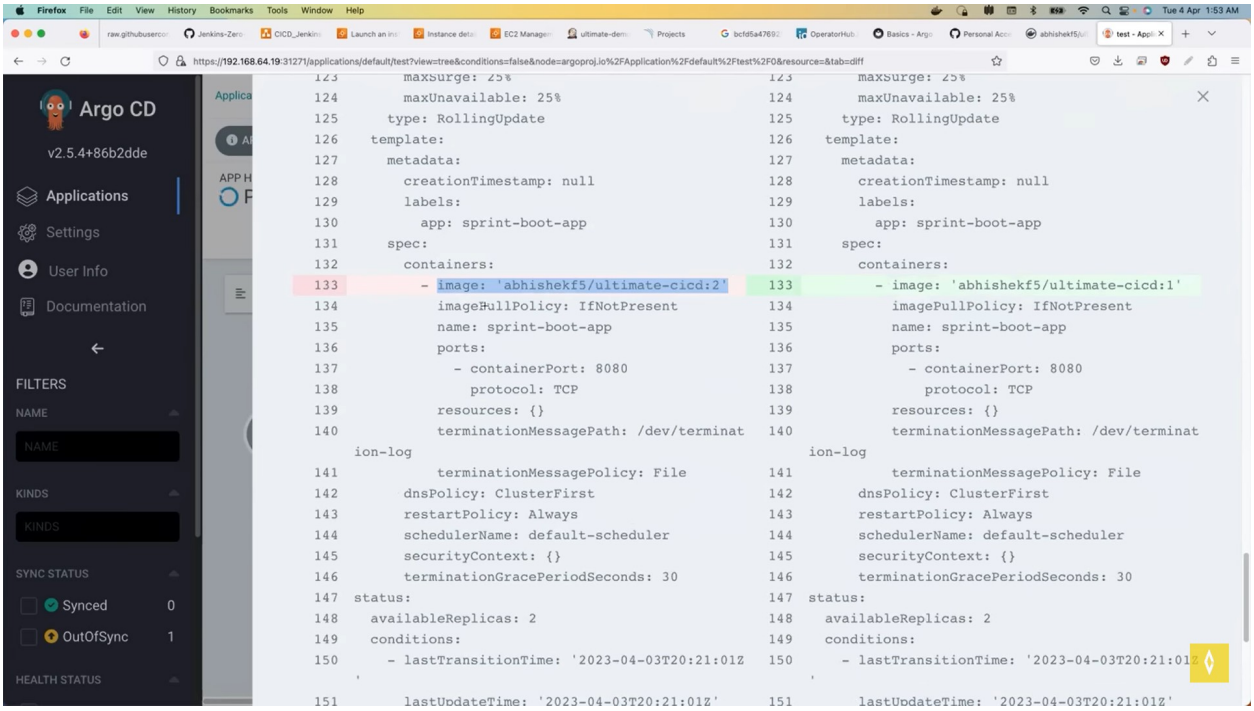


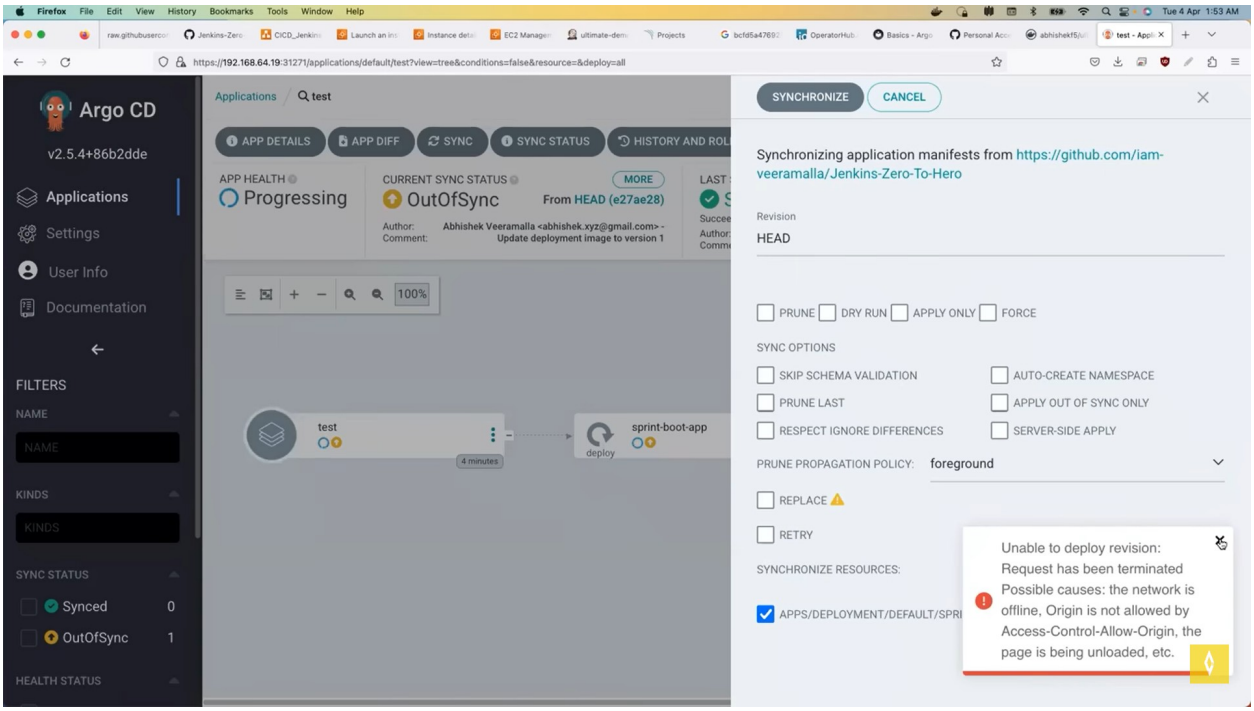
 # Click on test Dig deep



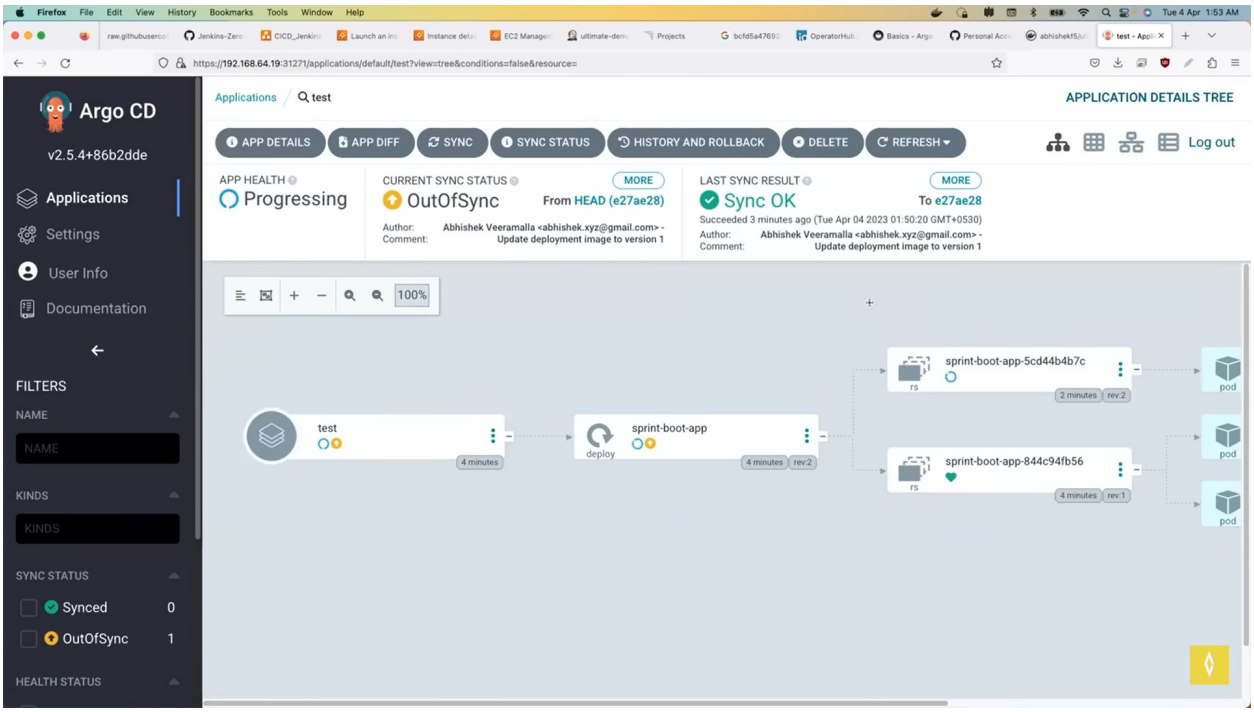
#- Click Diff

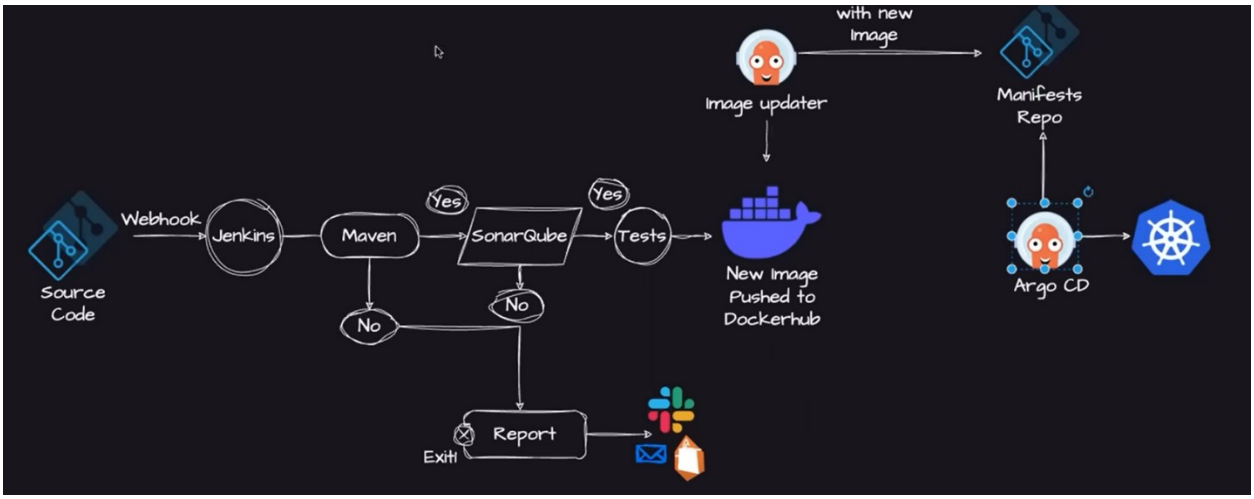

Argo CD
v2.5.4+86b2dde
Applications
Settings
User Info
Documentation
FILTERS
NAME
KINDS
SYNC STATUS
Synced 0
OutOfSync 1
HEALTH STATUS
SUMMARY
PARAMETERS
MANIFEST
DIFF
EVENTS
Compact diff
Inline diff


Argo CD
v2.5.4+86b2dde
Applications
Settings
User Info
Documentation
FILTERS
NAME
KINDS
SYNC STATUS
Synced 0
OutOfSync 1
HEALTH STATUS



 # Argocd Processing will revoke any changes made before – Automatically




with new Image
Image updater
Manifests Repo
Webhook
Jenkins
Maven
Yes
SonarQube
Yes
Tests
Source Code
New Image Pushed to Dockerhub
Argo CD
No
No
Report
Exit!