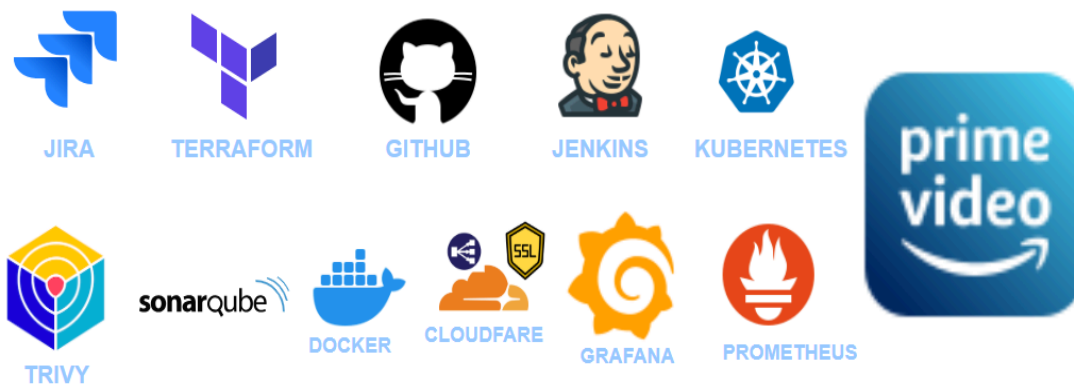


# prime video Clone App — End-to-End Secure &

## Scalable Deployment with DevSecOps & Jenkins !

Thrilled to showcase our latest project — a prime video **Clone Application** engineered with a **DevSecOps CI/CD pipeline** and powered by **Jenkins** Parameterise Build to deliver fully automated, secure, and scalable deployments.



**prime video Clone App — End-to-End Secure &  
Scalable Deployment with DevSecOps & Jenkins !**

### Tech Stack & Key Integration

- Kubernetes & Docker — for seamless, containerized, and scalable application deployments Jenkins — to enable reusable, standardized, and consistent CI/CD pipelines.
- SonarQube, Trivy, OWASP Dependency-Check — ensuring robust security, vulnerability scanning, and code quality automation.
- Prometheus & Grafana — for comprehensive real-time monitoring and alerting.
- Gmail Email alerts and collaboration-driven notifications
- Parameterized environment orchestration — for on-demand infrastructure setup and teardown.

## Key Highlights & Takeaways:

- Security-by-design with integrated DevSecOps practices
- Automated, reusable pipelines ensuring consistency and efficiency
- Production-grade scalability leveraging Kubernetes
- Rapid deployment lifecycle driven by modern CI/CD automation

This solution represents a blueprint for secure, scalable, and production-ready application delivery, embodying best practices in DevSecOps, automation, and cloud-native architecture.

## Now, let's get started and dig deeper into each of these steps:

### STEP 1 : Configure Infrastructure In AWS Cloud

- Launch an EC2 Instance Ubuntu (22.04) T3 X Large Instance
- Go to the AWS Management Console → EC2 → Instances. Click Launch Instance.
- Set the following configurations:
- Name: <Instance\_Name>
- AMI (Amazon Machine Image): Ubuntu Server 22.04 LTS (HVM), SSD Volume type.
- Instance Type: t3.xlarge (4 vCPUs, 16 GB RAM) o
- Key Pair: Select an existing key pair or create a new one.
- Storage: Default (e.g., 30GB GP3 SSD, adjust as needed)

### STEP 2 : Configure Security Group

Port	Protocol	Description
22	TCP	SSH (for remote access)
80	TCP	HTTP (Web traffic)
443	TCP	HTTPS (Secure web traffic)
8080	TCP	Web applications (Tomcat, etc.)
587	TCP	SMTP (Email sending)
465	TCP	SMTP over SSL
3000	TCP	Web apps (Grafana, Node.js, etc.)
9000	TCP	SonarQube/Web apps

Connect EC2 Instance through terminus, mobaxterm

## STEP 2 : Install Jenkins, Docker , awscli, terraform, kubectl , eksctl and Trivy

- Clone the GITHUB Project repositories  
`https://github.com/Shwetanarwade/amazon-prime-video-kubernetes.git`
- `cd hotstar-kubernetes/scripts/`
- Install the TOOLS in the VM machine via Scrtipts .
- add executable permission to shell script

```
chmod +x *.sh
```

- Access Jenkins in your browser:

[http://<public\\_TP>:8080](http://<public_TP>:8080)

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

- Unlock Jenkins using an administrative password and install the suggested plugins. Retrieve the initial admin password:

```
sudo cat /var/lib/jenkins/secrets/initialAdminPassword
```

- administrative password and install the suggested plugins.

## Customize Jenkins

Plugins extend Jenkins with additional features to support many different needs.

### Install suggested plugins

Install plugins the Jenkins community finds most useful.

### Select plugins to install

Select and install plugins most suitable for your needs.

- Create a user click on save and continue.
- Jenkins Getting Started Screen.
- Follow the setup wizard and install recommended plugins.

**Install Plugins like JDK, SonarQube Scanner, NodeJs, OWASP Dependency Check Goto Manage Jenkins → Plugins → Available Plugins → Install below plugins**

1. Eclipse Temurin Installer (Install without restart)
2. SonarQube Scanner (Install without restart)
3. NodeJs Plugin (Install Without restart) – 16.20.2
4. OWASP Dependency Check Plugins
5. Stage view
6. jdk Docker plugin
7. Docker 8. Docker Commons
9. Docker Pipeline
10. Docker API
11. Docker-build-step

The screenshot shows the Jenkins web interface. At the top, the Jenkins logo and navigation links are visible. The main section is titled 'Plugins' and contains a search bar with 'docker' entered. Below the search bar, there are three plugins listed, each with a checkmark in the 'Install' column:

Install	Name	Released
<input checked="" type="checkbox"/>	<b>Docker</b> 1274.vc0203fd2e74 Cloud Providers Cluster Management docker This plugin integrates Jenkins with Docker	3 mo 3 days ago
<input checked="" type="checkbox"/>	<b>Docker Commons</b> 457.v0f62a_94f11a_3 Library plugins (for use by other plugins) docker Provides the common shared functionality for various Docker-related plugins.	9 days 23 hr ago
<input checked="" type="checkbox"/>	<b>Docker Pipeline</b> 621.va_73f881d9232 pipeline DevOps Deployment docker Build and use Docker containers from pipelines.	9 days 22 hr ago Activate Windows Go to Settings to activate Windows.

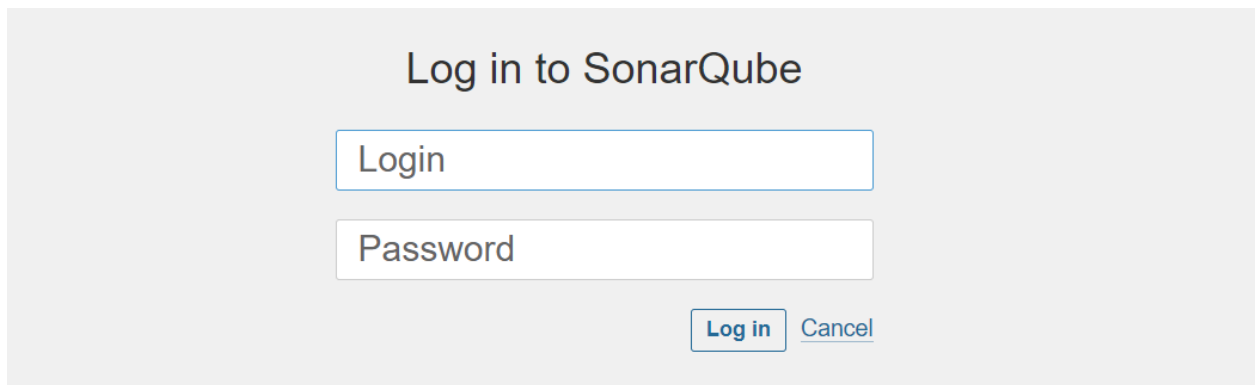
On the left sidebar, there are links for 'Updates', 'Available plugins', 'Installed plugins', 'Advanced settings', and 'Download progress'. A URL 'https://plugins.jenkins.io/docker-commons' is visible at the bottom left.

- Setup SonarQube Server
- we create a sonarqube container

`docker run -d --name sonar -p 9000:9000 sonarqube:lts-community`

```
ubuntu@ip-172-31-4-238:~/hotstar-kubernetes/scripts$ docker run -d --name sonar
:9000 sonarqube:lts-community
Unable to find image 'sonarqube:lts-community' locally
lts-community: Pulling from library/sonarqube
89dc6ea4eae2: Pull complete
31436012ac5b: Pull complete
2d16eb76e762: Pull complete
ac81863d97cb: Pull complete
26f6dfeccc10: Pull complete
a7343cdb8fb1: Pull complete
7c8e090ec954: Pull complete
4f4fb700ef54: Pull complete
Digest: sha256:1f0acc4139fc0bf09f39b962a63b56b10393cb0d07a4aa28346ea0af5b95f764
Status: Downloaded newer image for sonarqube:lts-community
cf1e37d1fe68d89a6ffe2e352a718c13b922e48aa69e6c887c87f2eae7e93d50
```

- Now SonarQube server is running you can access it from  
<publicIP of Instance>:9000
- Login to SonarQube ( default username and password is : admin-admin)



The image shows a login form for SonarQube. It has a title "Log in to SonarQube" at the top. Below the title are two input fields: "Login" and "Password". At the bottom right of the form are two buttons: "Log in" and "Cancel".

### Log in to SonarQube

- Enter username and password, click on login and change password username admin  
password admin

username admin

password admin

### Update your password

This account should not use the default password.

Enter a new password

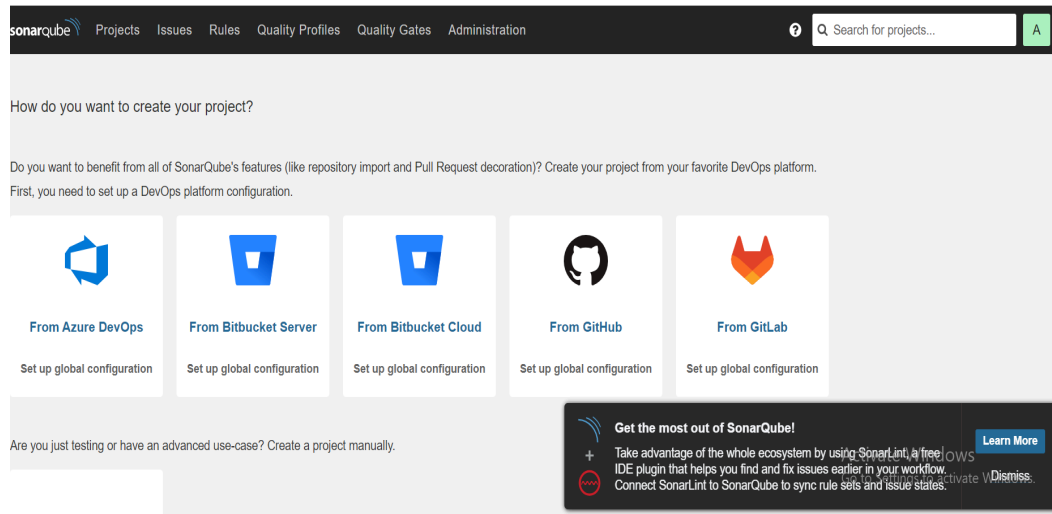
All fields marked with \* are required

Old Password \*

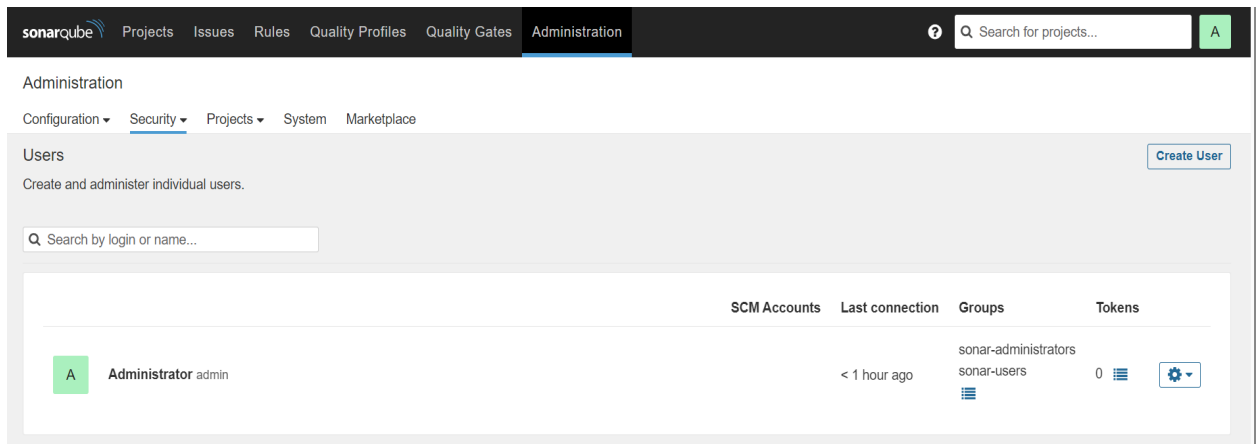
New Password \*

Confirm Password \*

Update New password, This is Sonar Dashboard



B. - Create Sonar token in order to connect with Jenkins Click on Administration → Security → Users → Click on Tokens and Update Token → Give it a name → and click on Generate Token




Create a token with a name and generate


**Tokens**

If you want to enforce security by not providing credentials of a real SonarQube user to run your code scan or to invoke web services, you can provide a User Token as a replacement of the user login. This will increase the security of your installation by not letting your analysis user's password going through your network.

**Generate Tokens**

**Name** **Type** **Expires in**

 New token "Jenkins" has been created. Make sure you copy it now, you won't be able to see it again!

 `squ_6a6e33aa85c5816c4190db7430a67414cb321191`

Name	Type	Project	Last use	Created	Expiration
Jenkins	User		Never	June 9, 2025	July 9, 2025

[Revoke](#)

Go to manage jenkins & add Credentials → Add Secret Text. It should look like this

**New credentials**

Kind

Scope ?

Secret

ID ?

Now, go to Dashboard → Manage Jenkins → System and Add like the below image



## SonarQube installations

List of SonarQube installations

Name

SonarQube

Server URL

Default is <http://localhost:9000>

<http://18.117.186.245:9000/>

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

Sonar-token

Save Apply

Activate Windows  
Go to Settings to activate

The Configure System option is used in Jenkins to configure different server Global Tool Configuration is used to configure different tools that

We install using Plugins We will install a sonar scanner in the tools.

Manage Jenkins -> Tools -> SonarQube Scanner

## 1. JDK Configuration

## JDK installations

Add JDK

≡ JDK

Name

jdk

☒ Install automatically ?

≡ Install from adoptium.net ?

Version ?

jdk-17.0.9+9

Add Installer ▾

Save Apply

Activate Windows  
Go to Settings to activate Wi

## 2. Configuration of sonar-scanner

☰ SonarQube Scanner

Name

sonar-scanner

☒ Install automatically ?

☰ Install from Maven Central

Version

SonarQube Scanner 7.0.0.4796

Add Installer ▾

Add SonarQube Scanner

Activate Windows  
Go to Settings to activate

Save

Apply

### 3.Configuration of nodejs

☰ NodeJS

Name

nodejs

☒ Install automatically ?

☰ Install from nodejs.org

Version

NodeJS 17.9.0

For the underlying architecture, if available, force the installation of the 32bit package. Otherwise the build will fail

☐ Force 32bit architecture

Global npm packages to install

Specify list of packages to install globally -- see npm install -g. Note that you can fix the packages version by using the syntax `packageName@version`

Global npm packages refresh hours

Duration, in hours, before 2 npm cache update. Note that 0 will always update npm cache

Save

Apply

### 4.Configure Dependency Check

Add Dependency-Check

**Dependency-Check**

Name

DC

☒ Install automatically ?

**Install from github.com**

Version

dependency-check 12.0.2

Add Installer

Add Dependency-Check

Save Apply

Activate Windows  
Go to Settings to activate Win

Create Job for Hotstar Let's add a pipeline :

**Jenkins** shweta log

Dashboard > Hotstar-Pipeline > Configuration

**Configure**

General Triggers Pipeline Advanced

**General** Enabled

Description

This the hotstar clone pipeline

Plain text Preview

☐ Discard old builds ?

☐ Do not allow concurrent builds

☐ Do not allow the pipeline to resume if the controller restarts

Save Apply

Activate Windows  
Go to Settings to activate Windows

## Setup Jenkins GitHub token inorder to connect with Private Registry

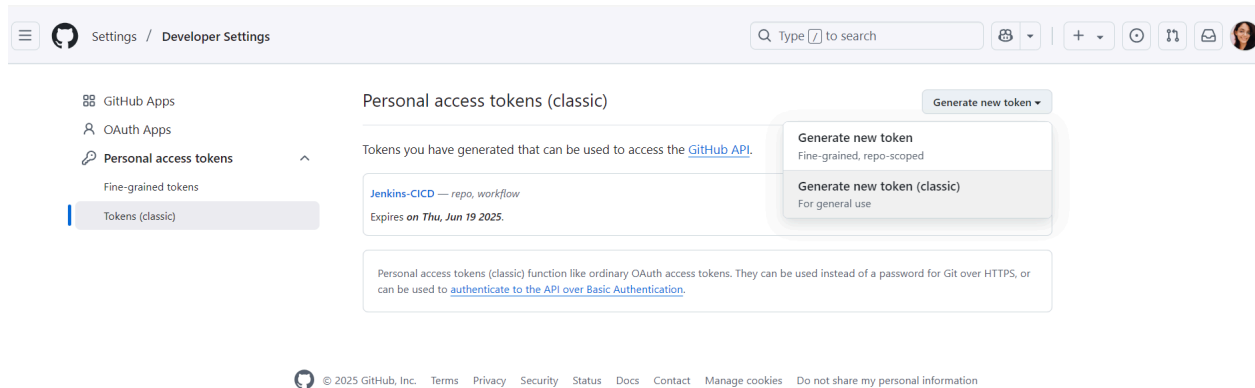
- Generate Classic GitHub Token

Go to GitHub → Settings → Developer Settings → Personal Access Tokens. Click Generate new token (classic). Set Expiration (or No Expiration if required).

Set Scopes:

- o repo → Full control of private repositories.
- o admin:repo\_hook → Manage repository webhooks.
- o workflow → Required for GitHub Actions (optional).

Click Generate token and copy the token



## Configure Jenkins with GitHub Token Add Credentials in Jenkins

1. Go to Jenkins Dashboard → Manage Jenkins → Manage Credentials.
2. Click Global Credentials (Unrestricted) → Add Credentials.
3. Select:
  - o Kind: Username and password
  - o Username: Your GitHub username
  - o Password: Paste your GitHub Token
  - o ID: github-token
  - o Description: GitHub Classic Token

Username with password

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

Username ?  
Shwetananarwade

☐ Treat username as secret ?

Password ?  
.....

ID ?  
github-token

Description ?  
github-token

Create

Activate Windows  
Go to Settings to activate Windows.

Create Job for Hotstar Let's add a pipeline , to test the Github Clone stage of Private Registry

Define your Pipeline using Groovy directly or pull it from source control.

Configure

- General
- Triggers
- Pipeline
- Advanced

Definition  
Pipeline script

Script ?

```

1 pipeline {
2   agent any
3
4   stages {
5     stage('git-clone') {
6       steps {
7         git branch: 'main', credentialsId: 'github-token', url: 'https://github.com/Shwetananarwade/hotstar-kubernetes
8       }
9     }
10  }
11 }
12

```

☒ Use Groovy Sandbox ?

Save Apply

Activate Windows  
Go to Settings to activate Windows.

Apply and Save and click on Build

Dashboard > Hotstar-Pipeline >

**Stage View**

Configure  
Delete Pipeline  
Full Stage View  
Favorite  
Open Blue Ocean  
Stages  
Rename  
Pipeline Syntax

Average stage times:  
(full run time: ~7s)

git-clone  
4s

#1 Jun 09 18:01 No Changes

4s

**Permalinks**

**Builds**

No builds

Today  
#1 12:31

## Add docker credentials

Username with password

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

Username ?  
shwetananwade2510

☐ Treat username as secret ?

Password ?  
.....

ID ?  
docker

Description ?  
docker

Create

Activate Windows  
Go to Settings to activate Windows.

Install Dockers scout in the app server Install Docker Scout: docker login Give Dockerhub credentials here curl -sSfL https://raw.githubusercontent.com/docker/scout-cli/main/install.sh | sh -s -- -b /usr/local/bin

```

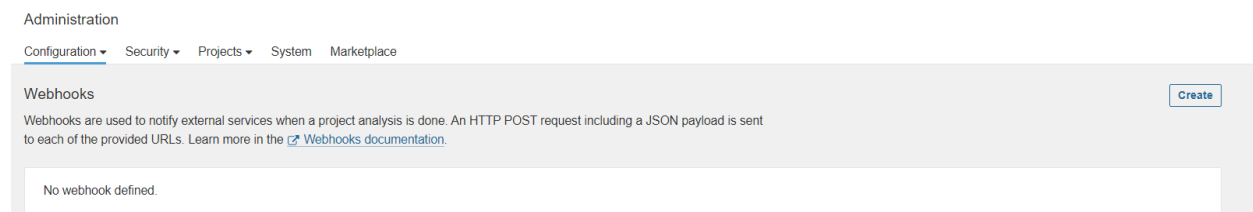
Username: aseemakram19
Password:
WARNING! Your password will be stored unencrypted in /home/ubuntu/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
ubuntu@ip-10-0-1-103:/var/lib/jenkins/workspace/amazon-prime-video$ curl -sSfL https://raw.githubusercontent.com/docker/scout-cli/main/install.sh | sh -s -- -b /usr/local/bin
[info] fetching release script for tag='v1.17.1'
[info] using release tag='v1.17.1' version='1.17.1' os='linux' arch='amd64'
install: cannot create regular file '/usr/local/bin/docker-scout': Permission denied
[error] failed to install docker-scout
ubuntu@ip-10-0-1-103:/var/lib/jenkins/workspace/amazon-prime-video$ sudo su
root@ip-10-0-1-103:/var/lib/jenkins/workspace/amazon-prime-video# curl -sSfL https://raw.githubusercontent.com/docker/scout-cli/main/install.sh | sh -s -- -b /usr/local/bin
[info] fetching release script for tag='v1.17.1'
[info] using release tag='v1.17.1' version='1.17.1' os='linux' arch='amd64'
[info] installed /usr/local/bin/docker-scout
root@ip-10-0-1-103:/var/lib/jenkins/workspace/amazon-prime-video#

```

## In the Sonarqube Dashboard add a quality gate also

Administration → Configuration → Webhooks



## Create a webhook

### Create Webhook

All fields marked with \* are required

**Name \***

**URL \***

Server endpoint that will receive the webhook payload, for example:  
 "http://my\_server/foo". If HTTP Basic authentication is used, HTTPS is recommended to avoid man in the middle attacks. Example:  
 "https://myLogin.myPassword@my\_server/foo"

**Secret**

If provided, secret will be used as the key to generate the HMAC hex (lowercase) digest value in the 'X-Sonar-Webhook-HMAC-SHA256' header.

## Create a Gmail SMTP App Password

An App Password is a 16-character password that allows third-party applications (like Jenkins) to send emails using Gmail SMTP securely.

**Step 1: Enable 2-Step Verification Before generating an App Password, you must enable 2-Step Verification in your Google Account.**

1. Go to Google Account Security: Google My Account
2. Scroll to "Signing in to Google".
3. Click "2-Step Verification" → Click "Get Started".
4. Follow the steps to set up 2-Step Verification (via SMS or Authenticator App).

**Step 2: Generate an App Password**

1. Go to App Passwords Page: Google App Passwords
2. Sign in with your Google Account.
3. Under "Select app", choose "Mail".
4. Under "Select device", choose "Other (Custom Name)" and enter "Jenkins SMTP".
5. Click "Generate".
6. Copy the 16-character App Password (e.g., abcd efgh ijkl mnop).

Google Account

← App passwords

that use modern security standards. Before you create an app password, you should check to see if your app needs this in order to sign in.  
[Learn more](#)

Your app passwords

primevideo

Created on Jun 7, last used on Jun 7



To create a new app specific password, type a name for it below...

App name

hotstar-clone

Create

Add credentials as Username and password in jenkins



Jenkins

Dashboard > Manage Jenkins > Credentials > System > Global credentials (unrestricted) >

### New credentials

Kind

Username with password

Scope ?

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

Username ?

shwetnarwade2510@gmail.com

☒ Treat username as secret ?

Password ?

.....

ID ?

Create

Activate Windows  
Go to Settings to activate Windows.

### Step 3: Configure Gmail SMTP in Jenkins

1. Go to Jenkins Dashboard → Manage Jenkins → Configure System.
2. Scroll to "E-mail Notification".
3. Set the following:
  - o SMTP Server: [smtp.gmail.com](mailto:smtp.gmail.com)
  - o Use SMTP Authentication: Checked
  - o User Name: Your Gmail ID (your-email@gmail.com)
  - o Password: Paste the App Password
  - o SMTP Port: 465
  - o Use TLS: Checked

E-mail Notification

SMTP server  
smtp.gmail.com

Default user e-mail suffix ?  
@gmail.com

Advanced ^ Edited

☒ Use SMTP Authentication ?

User Name  
shwetanarwade2510@gmail.com

For security when using authentication it is recommended to enable either TLS or SSL

Password  
.....

☒ Use SSL ?

Save Apply

Activate Windows  
Go to Settings to activate Windows.

#### 4. Click Save. Email Extension Plugin xsuc kxeb xcvk xqkf

##### 1. Basic Email Notification SMTP Server: [smtp.gmail.com](https://smtp.gmail.com)

- Email Suffix: @gmail.com (default user email domain)
- SMTP Authentication: Enabled
- Username: Your Gmail address
- Password: Your Gmail password or App Password (for 2-factor authentication)
- Use TLS: Checked
- SMTP Port: 587
- Reply-To Address: Your email address
- Charset: UTF-8

SMTP server  
smtp.gmail.com

SMTP Port  
587

Advanced ^ Edited

Credentials  
shwetanarwade2510@gmail.com/\*\*\*\*\* (smtp.gmail)

+ Add

☐ Use SSL

☒ Use TLS

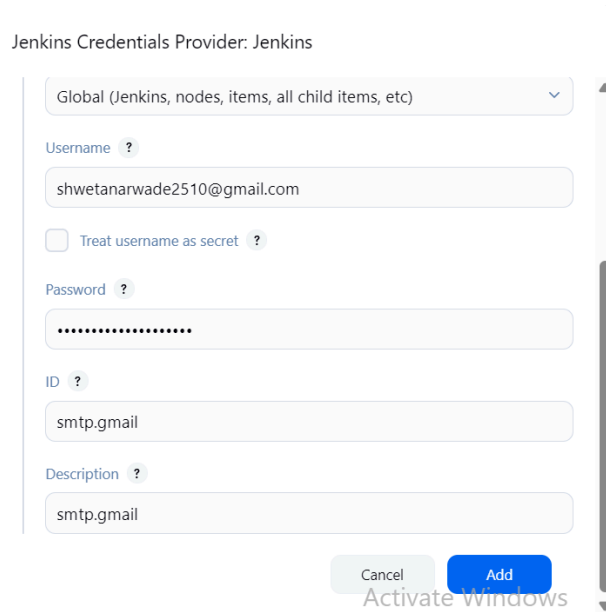
☐ Use OAuth 2.0

Advanced Email Properties

Save Apply

Activate Windows  
Go to Settings to activate Windows.

## Add your gmail credentials in credentials section of Jenkins



Jenkins Credentials Provider: Jenkins

Global (Jenkins, nodes, items, all child items, etc) ▾

Username ?  
shwetananarwade2510@gmail.com

☐ Treat username as secret ?

Password ?  
.....

ID ?  
smtp.gmail

Description ?  
smtp.gmail

Cancel Add

**And last Register for NVD API for Dependency Check The National Vulnerability Database (NVD) API provides access to security vulnerabilities (CVEs) and is often used with tools like OWASP Dependency-Check to identify security risks in software dependencies.**

### Step 1: Create an NVD API Key

1. Go to the NVD API Registration Page:
  - o Open: NVD API Registration
2. Sign In or Create an Account:
  - o Click Sign In (or create an account if you don't have one).
3. Request an API Key:
  - o Provide your details and agree to the terms.
  - o Click Submit.
4. Receive API Key via Email:
  - o Once approved, you'll receive an API key.

## Request an API Key

To request an NVD API Key, please provide your organization name and a valid email address, and indicate your organization type. You must scroll to end of the Terms of Use Agreement and check "I agree to the Terms of Use" to obtain an API Key. Upon submitting the request, you will receive an email containing a single-use hyperlink that is used to activate and view your API Key. If your key is not activated within seven days, a new request for an API Key must be submitted.

**Organization Name:**

**Email Address:**

**Organization Type:**

**Terms of Use**

The National Vulnerability Database (NVD) was created by the National Institute of Standards and Technology (NIST) and is being made available as a public service. The NVD offers some of its public data in machine-readable format via an Application Programming Interface ("API"). This service is offered subject to this Terms of Use and [NIST Website Policies](#) (collectively, the "Terms of Use" or "TOU").

**Use**

Activate Windows  
Go to Settings to activate Windows

## Then Configure the pipeline

```
pipeline{
  agent any
  tools{
    jdk 'jdk'
    nodejs 'node'
  }
  environment {
    SCANNER_HOME=tool 'sonar-scanner'
  }
  stages {
    stage('clean workspace'){
      steps{
        cleanWs()
      }
    }
  }
}
```

```

stage('Checkout from Git'){
    steps{
        git branch: 'main', credentialsId: 'github-token', url:
'https://github.com/Aseemakram19/hotstar-kubernetes.git'
    }
}
stage("Sonarqube Analysis "){
    steps{
        withSonarQubeEnv('SonarQube') {
            sh "' $SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=Hotstar \
-Dsonar.projectKey=Hotstar '"
        }
    }
}
stage("quality gate"){
    steps {
        script {
            waitForQualityGate abortPipeline: false, credentialsId: 'Sonar-token'
        }
    }
}
stage('Install Dependencies') {
    steps {
        sh "npm install"
    }
}
stage('OWASP FS SCAN') {

```

```

    steps {
        dependencyCheck additionalArguments: '--scan ./ --disableYarnAudit
--disableNodeAudit --nvdApiKey d7e8c629-7da9-4f96-8a4a-a45fd3f213ba', odcInstallation: 'DC'
        dependencyCheckPublisher pattern: '**/dependency-check-report.xml'
    }
}

stage('TRIVY FS SCAN') {
    steps {
        sh "trivy fs . > trivyfs.txt"
    }
}

stage("Docker Build & Push"){
    steps{
        script{
            withDockerRegistry(credentialsId: 'docker', toolName: 'docker'){
                sh "docker build -t hotstar ."
                sh "docker tag hotstar shwetanarwade2510/hotstar:latest "
                sh "docker push shwetanarwade2510/hotstar:latest "
            }
        }
    }
}

stage("TRIVY"){
    steps{
        sh "trivy image shwetanarwade2510/hotstar:latest > trivyimage.txt"
    }
}

```

```

stage('Deploy to container'){
    steps{
        sh 'docker run -d --name hotstar -p 3000:3000 shwetanarwade2510/hotstar:latest'
    }
}

}

post {
    always {
        script {
            def buildStatus = currentBuild.currentResult
            def buildUser =
currentBuild.getBuildCauses('hudson.model.Cause$UserIdCause')[0]?.userId ?: 'Github User'

            emailext (
                subject: "Pipeline ${buildStatus}: ${env.JOB_NAME} #${env.BUILD_NUMBER}",
                body: """
                <p>This is a Jenkins HOTSTAR CICD pipeline status.</p>
                <p>Project: ${env.JOB_NAME}</p>
                <p>Build Number: ${env.BUILD_NUMBER}</p>
                <p>Build Status: ${buildStatus}</p>
                <p>Started by: ${buildUser}</p>
                <p>Build URL: <a href="${env.BUILD_URL}">${env.BUILD_URL}</a></p>
                """,
                to: 'shwetanarwade2510@gmail.com',
                from: 'shwetanarwade2510@gmail.com',
                replyTo: 'shwetanarwade2510@gmail.com',

```

```

    mimeType: 'text/html',
    attachmentsPattern: 'trivyfs.txt,trivyimage.txt'
  )
}
}
}
}
}
}

```

## Stage View after building pipeline

The screenshot shows the Jenkins Stage View for a pipeline named 'Hotstar-Pipeline'. The interface includes a left sidebar with navigation options like 'Changes', 'Build Now', 'Configure', 'Delete Pipeline', 'Full Stage View', 'Favorite', 'Open Blue Ocean', 'Stages', 'Rename', and 'Pipeline Syntax'. The main area displays the 'Stage View' with a table of stages and their durations. Below the table, there are 'Permalinks' for the last build, last stable build, last successful build, and last completed build. The bottom right corner shows the 'Activate Windows' watermark and the Jenkins version 'Jenkins 2.504.2'.

Stage	Declarative: Tool Install	clean workspace	Checkout from Git	Sonarqube Analysis	quality gate	Install Dependencies	OWASP FS SCAN	TRIVY FS SCAN	Docker Build & Push	TRIVY
Average stage times:	25s	368ms	2s	30s	793ms	39s	26s	12s	4min 44s	1min 48s
Jun 09 19:41	25s	368ms	2s	30s	793ms (paused for 10)	39s	26s	12s	4min 44s	1min 48s

Permalinks

- Last build (#1), 1 hr 39 min ago
- Last stable build (#1), 1 hr 39 min ago
- Last successful build (#1), 1 hr 39 min ago
- Last completed build (#1), 1 hr 39 min ago

Activate Windows  
Go to Settings to activate Windows.  
REST API Jenkins 2.504.2

Our Application is live with this output

```
<publicip_of_instance>:3000
```

You can see the report has been generated and the status shows as passed. You can see that there are 943 lines it scanned. To see a detailed report, you can go to issues. You will see that in status, a graph will also be generated and Vulnerabilities.



There's a new version of SonarQube available. Upgrade to the latest active version to access new updates and features. [Learn More](#)

sonarqube Projects Issues Rules Quality Profiles Quality Gates Administration

Search for projects... [Create Project](#)

Search by project name or key

1 project(s) Perspective: Overall Status Sort by: Name

Hotstar **Passed** Last analysis: 15 minutes ago

Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications	Lines
1 <b>C</b>	1 <b>B</b>	Vulnerabilities <b>E</b>	2 <b>A</b>	0.0% <b>C</b>	0.0% <b>C</b>	943 <b>X</b> JavaScript...

1 of 1 shown

Embedded database should be used for evaluation purposes only  
The embedded database will not scale, it will not support upgrading to newer versions of SonarQube, and there is no support for migrating your data out of it into a different database engine.

SonarQube™ technology is powered by SonarSource SA  
Community Edition - v9.9.8 (build 100196) - [LGPL v3](#) - [Community](#) - [Documentation](#) - [Plugins](#) - [Web API](#)

Activate Windows  
Go to Settings to activate Windows.

Our Application is live with this output

← ↻ 🔒 Not secure | 13.216.214.1:3000

prime video Home Store Live TV Categories My Stuff Search Nikhil

**AVATAR**

18 Dec 2009 PG-13

Play + ⌵

Popular Movies and TV shows

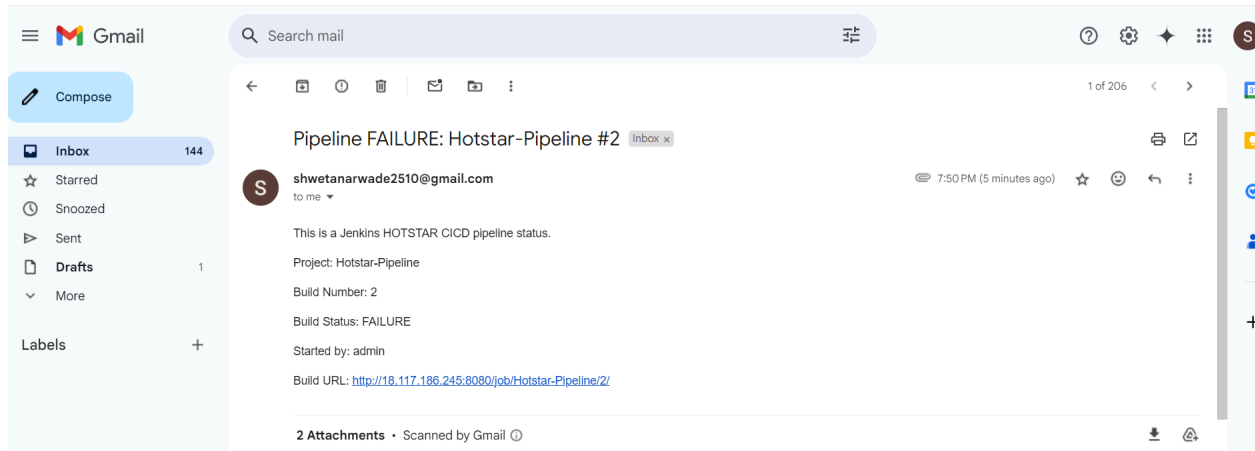
**APPLICATION**

DEPLOYMENT

MOVIE POSTER DB

THE W

## Email alert with Post build



## 2. Verify Docker Images in Docker Hub

