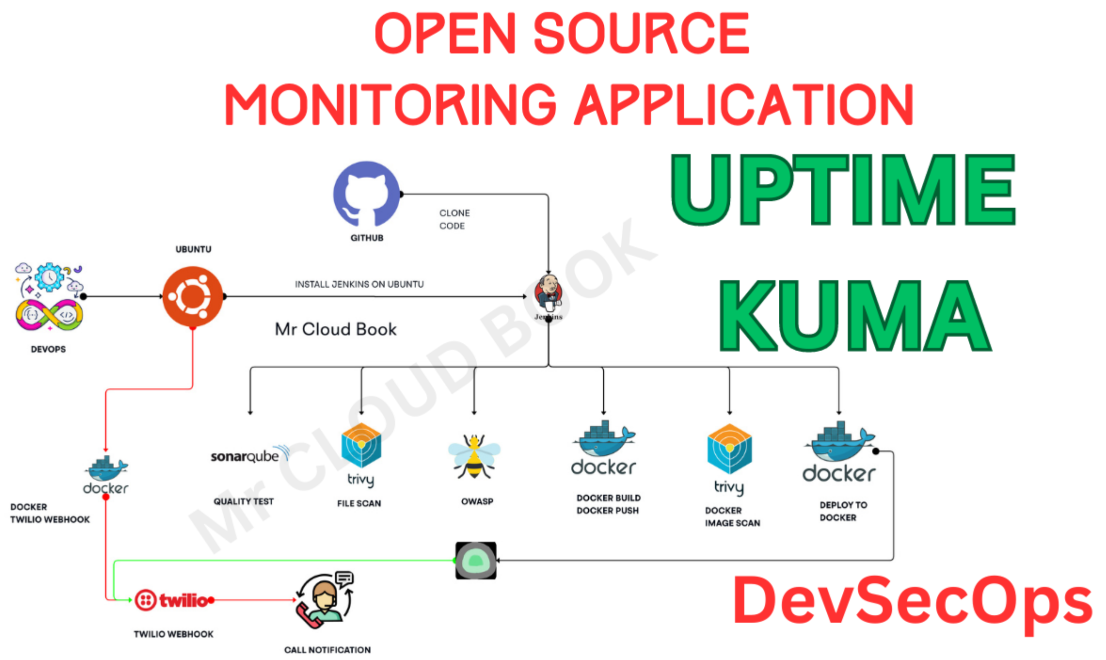
CI/CD Pipeline for Uptime-Kuma | Get Real-Time Call Alerts for Server Downtime | Open Source Monitoring



Uptime-Kuma is an open-source self-hosted monitoring tool that provides a sleek interface to track the uptime of websites, services, and APIs. When integrated into a CI/CD pipeline for a blog, it helps ensure continuous availability and performance monitoring. Here's a detailed explanation of how Uptime-Kuma can be utilized effectively in this context:

What is Uptime-Kuma?

- **Overview:** Uptime-Kuma is a self-hosted monitoring tool designed to track the uptime and performance of websites, services, and APIs.
- **Features:**
  - Real-time monitoring with customizable intervals.
  - Alerts via multiple channels (email, Slack, Telegram, etc.).
  - Historical data and uptime statistics.
  - Sleek, user-friendly interface.

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

1. AWS Account and Login

- If you don't have an AWS account, sign up at AWS Sign Up.
- Log in to the AWS Management Console.

2. Launch an EC2 Instance

A. Navigate to EC2 Dashboard:

- Go to the EC2 Dashboard.

B. Choose an Amazon Machine Image (AMI):

- Click “Launch Instance.”
- In the AMI selection screen, search for “Ubuntu 24.04”.

C. Choose an Instance Type:

- Select the t2.large instance type.
- Click “Next: Configure Instance Details.”

D. Configure Instance Details:

- Use the default settings.
- Ensure you have selected the correct VPC and subnet.
- Click “Next: Add Storage.”

E. Add Storage:

- Modify the root volume size to 30 GB.
- Click “Next: Add Tags.”

F. Add Tags:

- (Optional) Add tags for better management.
- Click “Next: Configure Security Group.”

G. Configure Security Group:

- Create a new security group.
- Set the security group name and description.
- Add a rule to allow all traffic (not recommended for production environments):
  - Type: All Traffic
  - Protocol: All
  - Port Range: All
  - Port Range : 22, 80, 443, 8080, 9000, 3001, 5000
  - Source: 0.0.0.0/0 (for IPv4) and ::/0 (for IPv6)
- **Warning:** Opening all ports is insecure and should only be done for learning purposes in a controlled environment.
- Click “Review and Launch.”

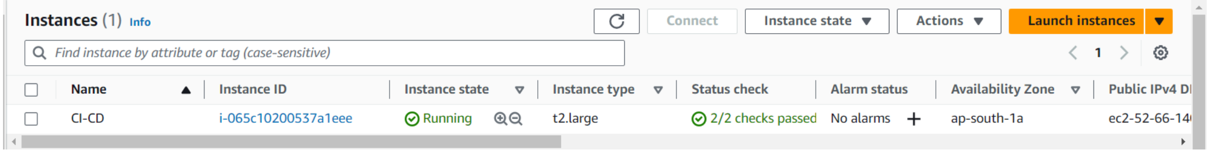
H. Review Instance Launch:

- Review all configurations.

- Click “Launch.”

I. Select/Create a Key Pair:

- Select “Create a new key pair.”
- Name the key pair.
- Download the key pair file (.pem) and store it securely.
- Click “Launch Instances.”



□	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DI
□	CI-CD	i-065c10200537a1eee	✓ Running	t2.large	✓ 2/2 checks passed	No alarms +	ap-south-1a	ec2-52-66-14

Install Jenkins, Docker and Trivy

To Install Jenkins

Connect to your console, and enter these commands to Install Jenkins

```
vi jenkins.sh

#!/bin/bash

sudo apt update -y

wget -O - https://packages.adoptium.net/artifactory/api/gpg/key/public | sudo tee /etc/apt/keyrings/adoptium.asc

echo "deb [signed-by=/etc/apt/keyrings/adoptium.asc] https://packages.adoptium.net/artifactory/deb $(awk -F= '/^VERSION_CODENAME/{print$2}' /etc/os-release) main" | sudo tee /etc/apt/sources.list.d/adoptium.list

sudo apt update -y

sudo apt install temurin-17-jdk -y

/usr/bin/java --version

curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee /usr/share/keyrings/jenkins-keyring.asc &gt; /dev/null

echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] https://pkg.jenkins.io/debian-stable binary/ | sudo tee /etc/apt/sources.list.d/jenkins.list &gt; /dev/null

sudo apt-get update -y

sudo apt-get install jenkins -y

sudo systemctl start jenkins

sudo systemctl status jenkins

sudo chmod 777 jenkins.sh

sudo su   #move into root and run
```

./jenkins.sh   # this will installl jenkins

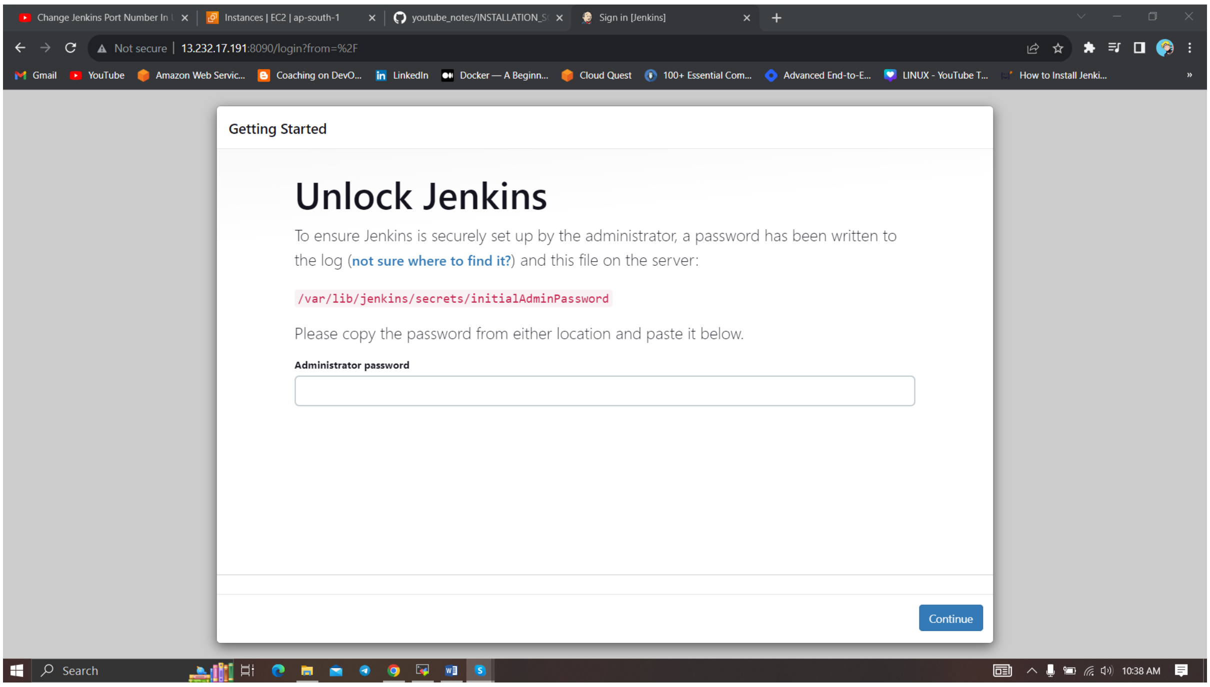
Once Jenkins is installed, you will need to go to your AWS EC2 Security Group and open Inbound Port 8080, since Jenkins works on Port 8080.

Now, grab your Public IP Address

EC2 Public IP Address:8080

# In Jenkins server

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



Getting Started

Unlock Jenkins

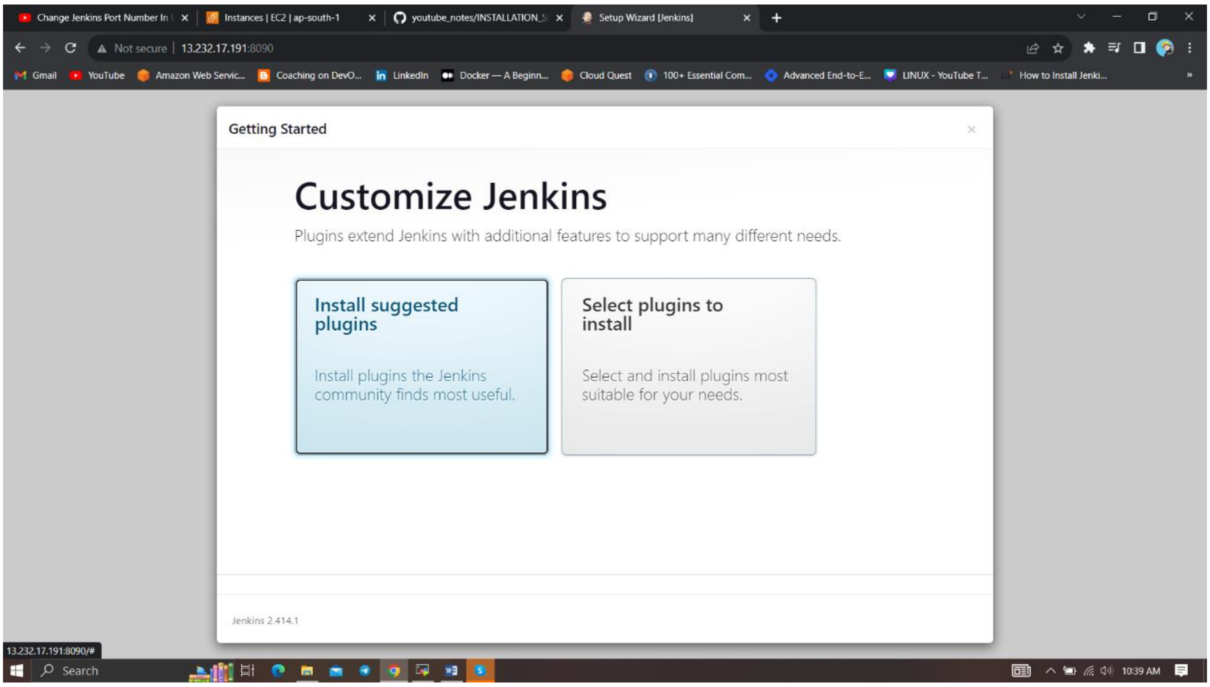
To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

/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.



Getting Started

Customize Jenkins

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

Install suggested plugins

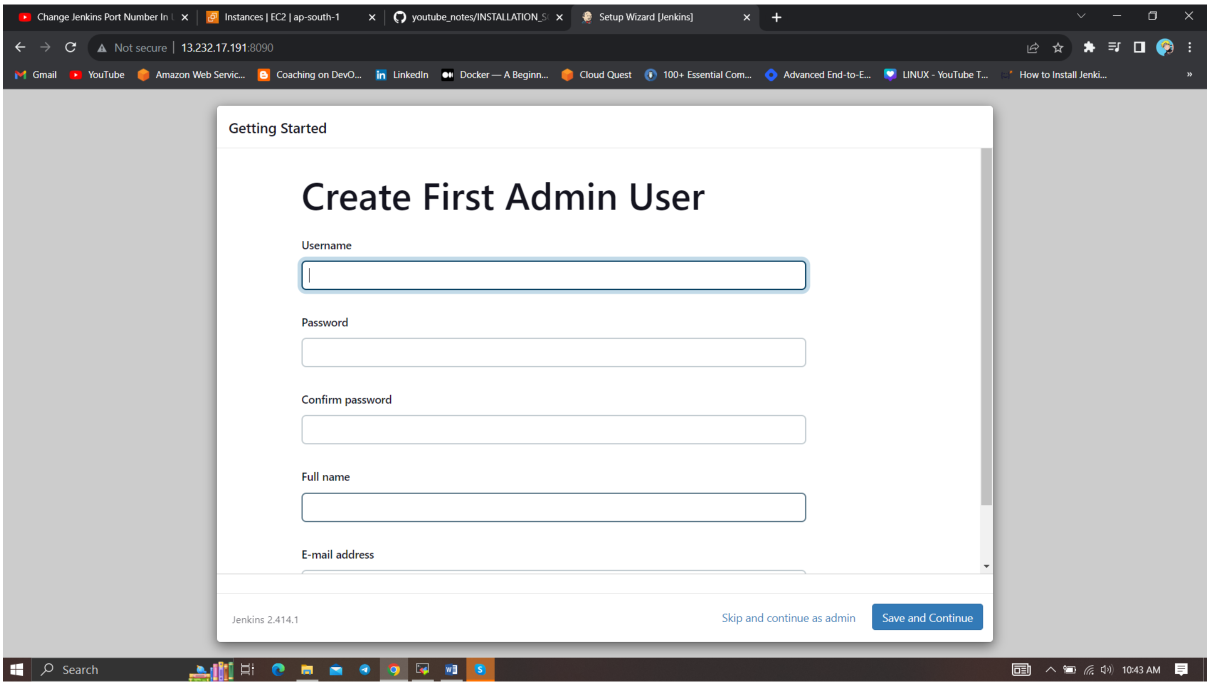
Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

Jenkins 2.414.1

Jenkins will now get installed and install all the libraries.



Getting Started

Create First Admin User

Username

Password

Confirm password

Full name

E-mail address

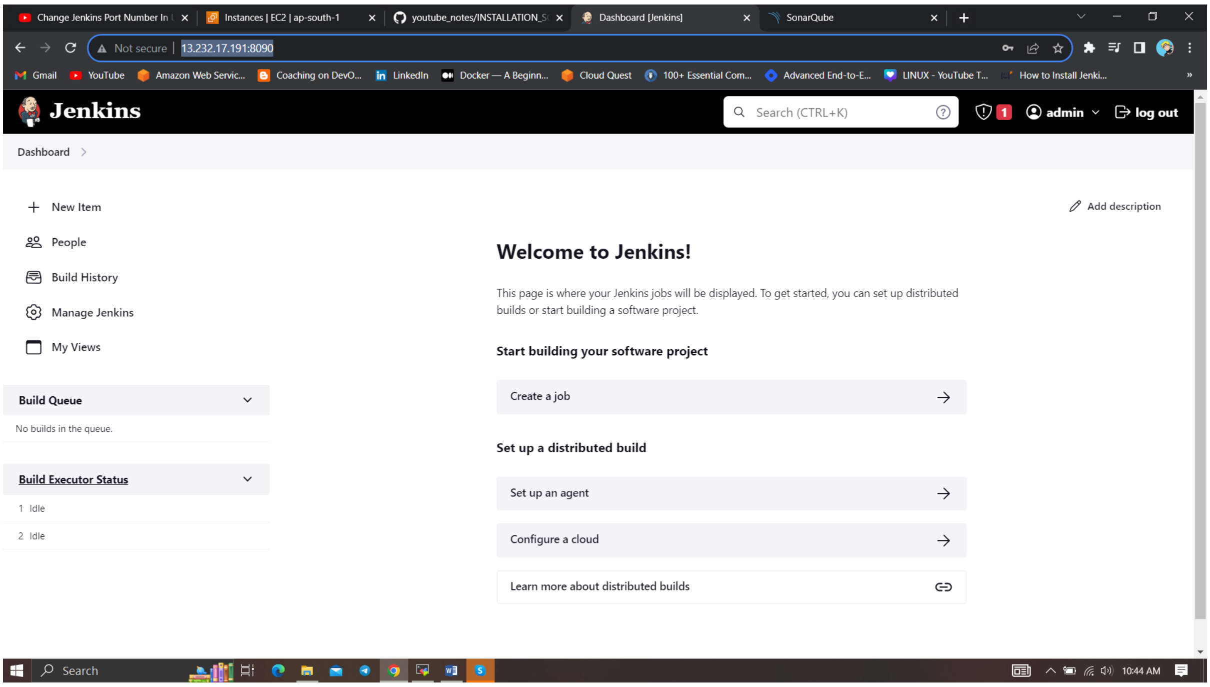
Jenkins 2.414.1

Skip and continue as admin

Save and Continue

Create a user click on save and continue.

Jenkins Getting Started Screen.



Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Welcome to Jenkins!

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

Start building your software project

Create a job

Set up a distributed build

Set up an agent

Configure a cloud

Learn more about distributed builds

Install Docker

# Add Docker's official GPG key:

sudo apt-get update

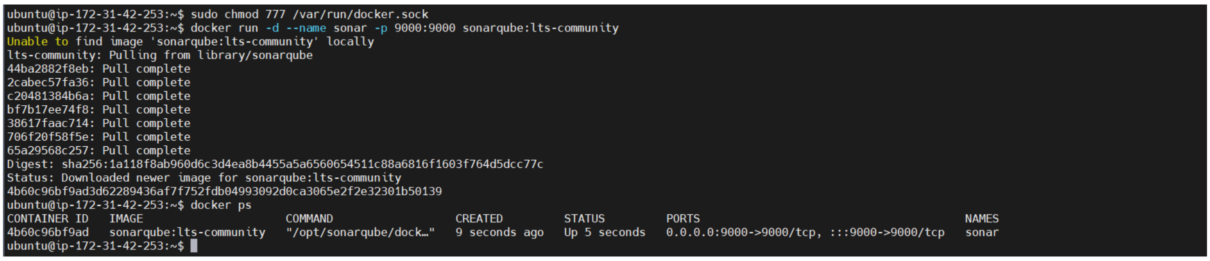
sudo apt-get install ca-certificates curl

sudo install -m 0755 -d /etc/apt/keyrings

```
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc

sudo chmod a+r /etc/apt/keyrings/docker.asc

# Add the repository to Apt sources:

echo \

  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \

  $(. /etc/os-release &amp;&amp; echo "$VERSION_CODENAME") stable" | \

  sudo tee /etc/apt/sources.list.d/docker.list &gt; /dev/null

sudo apt-get update

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin -y

sudo usermod -aG docker ubuntu

newgrp docker

sudo chmod 777 /var/run/docker.sock
```

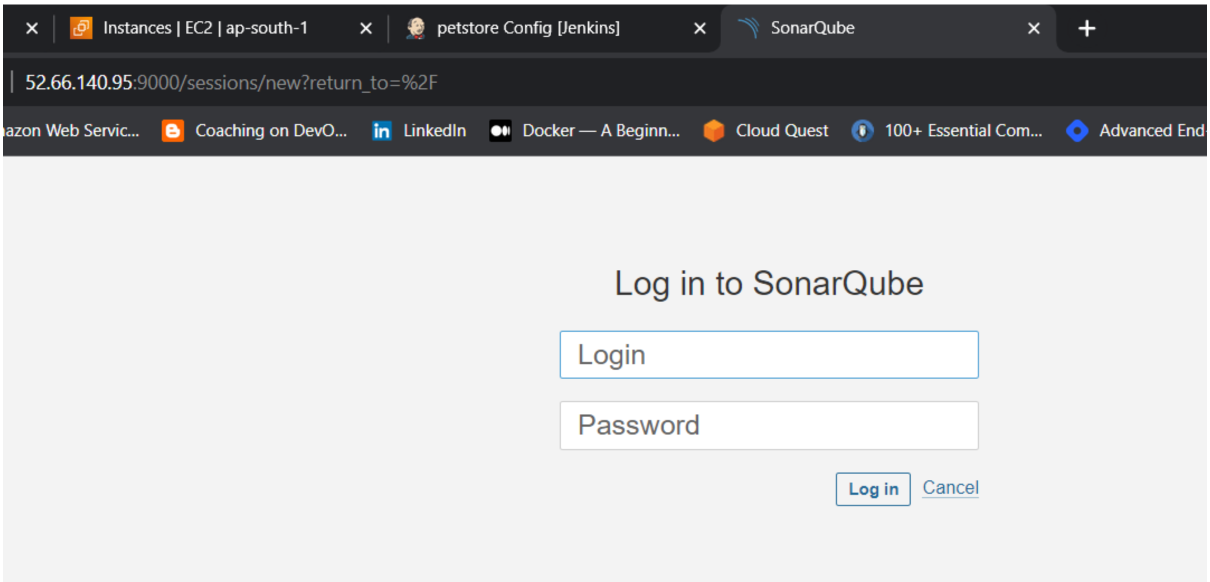
After the docker installation, we create a sonarqube container (Remember to add 9000 ports in the security group).

```
sudo chmod 777 /var/run/docker.sock

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



```
ubuntu@ip-172-31-42-253:~$ sudo chmod 777 /var/run/docker.sock
ubuntu@ip-172-31-42-253:~$ docker run -d --name sonar -p 9000:9000 sonarqube:lts-community
Unable to find image 'sonarqube:lts-community' locally
lts-community: Pulling from library/sonarqube
44ba2882f8eb: Pull complete
2cabec57fa36: Pull complete
c20481384b6a: Pull complete
bf7b17ee74f8: Pull complete
38617faac714: Pull complete
706f20f58f5e: Pull complete
65a29568c257: Pull complete
Digest: sha256:1a118f8ab960d6c3d4ea8b4455a5a6560654511c88a6816f1603f764d5dcc77c
Status: Downloaded newer image for sonarqube:lts-community
4b60c96bf9ad3d62289436af7f752fdb04993092d0ca3065e2f2e32301b50139
ubuntu@ip-172-31-42-253:~$ docker ps
CONTAINER ID   IMAGE                     COMMAND                  CREATED         STATUS         PORTS                                       NAMES
4b60c96bf9ad   sonarqube:lts-community   "/opt/sonarqube/dock…"   9 seconds ago   Up 5 seconds   0.0.0.0:9000->9000/tcp, :::9000->9000/tcp   sonar
ubuntu@ip-172-31-42-253:~$
```

Now our Sonarqube is up and running

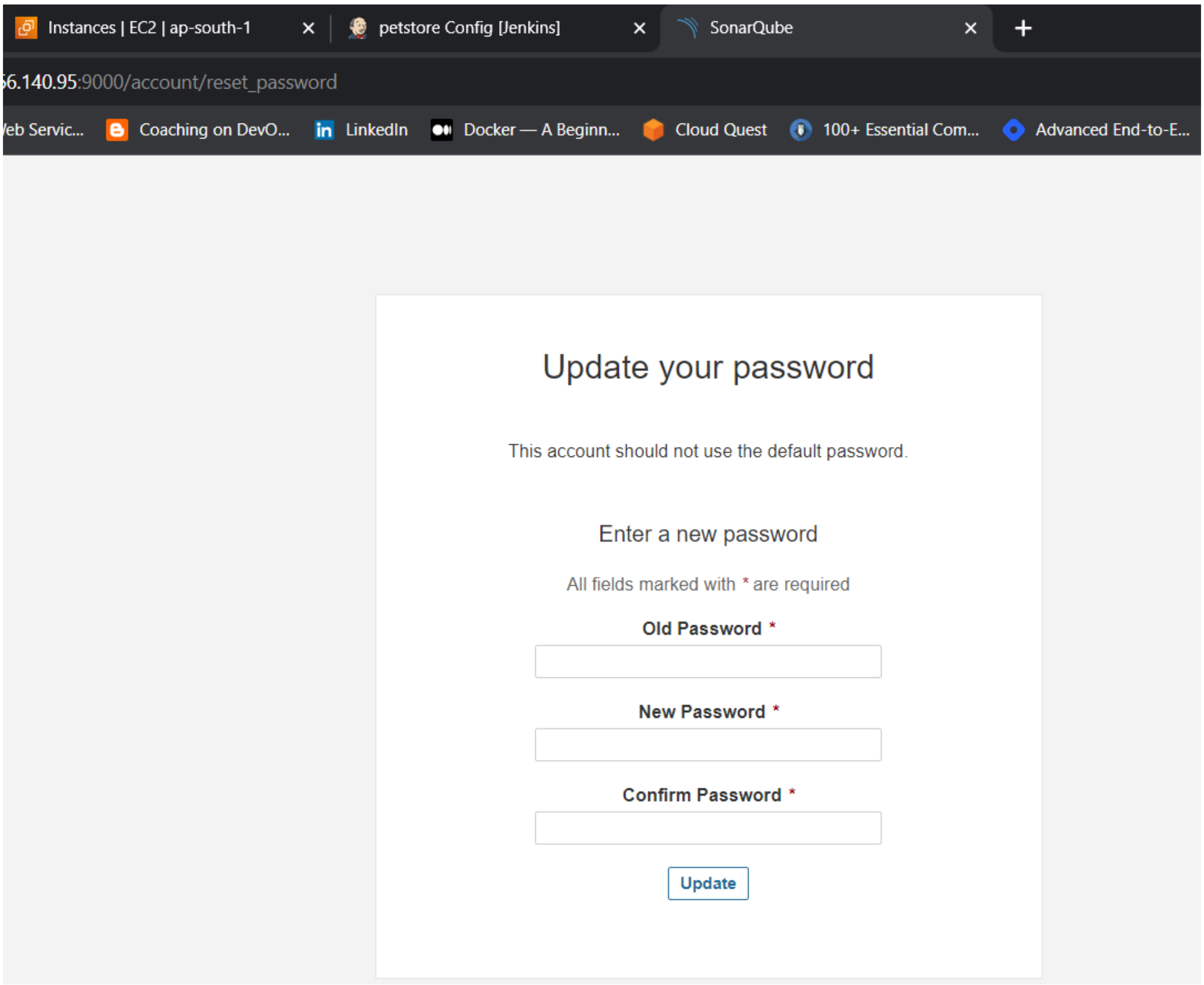


Log in to SonarQube

Enter username and password, click on login and change password

username admin

password admin



Update your password

This account should not use the default password.

Enter a new password

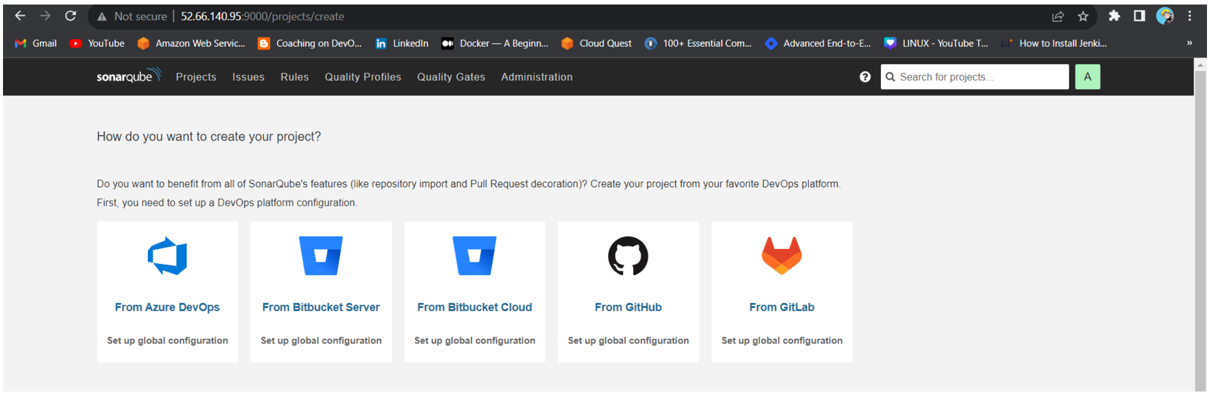
All fields marked with * are required

Old Password *

New Password *

Confirm Password *

Update New password, This is Sonar Dashboard.



How do you want to create your project?

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

Install Trivy

vi script.sh

sudo apt-get install wget apt-transport-https gnupg lsb-release -y

wget -qO - https://aquasecurity.github.io/trivy-repo/deb/public.key | gpg --dearmor | sudo tee /usr/share/keyrings/trivy.gpg &gt; /dev/null

echo "deb [signed-by=/usr/share/keyrings/trivy.gpg] https://aquasecurity.github.io/trivy-repo/deb $(lsb_release -sc) main" | sudo tee -a /etc/apt/sources.list.d/trivy.list

sudo apt-get update

sudo apt-get install trivy -y

Give permissions and run script

sudo chmod 777 script.sh

./script.sh

Next, we will log in to Jenkins and start to configure our Pipeline in Jenkins

Install Plugins like JDK, Sonarqube Scanner, NodeJs, OWASP Dependency Check

Install Plugin

Goto Manage Jenkins →Plugins → Available Plugins →

Install below plugins

Blue ocean

1 → Eclipse Temurin Installer

2 → SonarQube Scanner

3 → NodeJs Plugin

4 → Docker

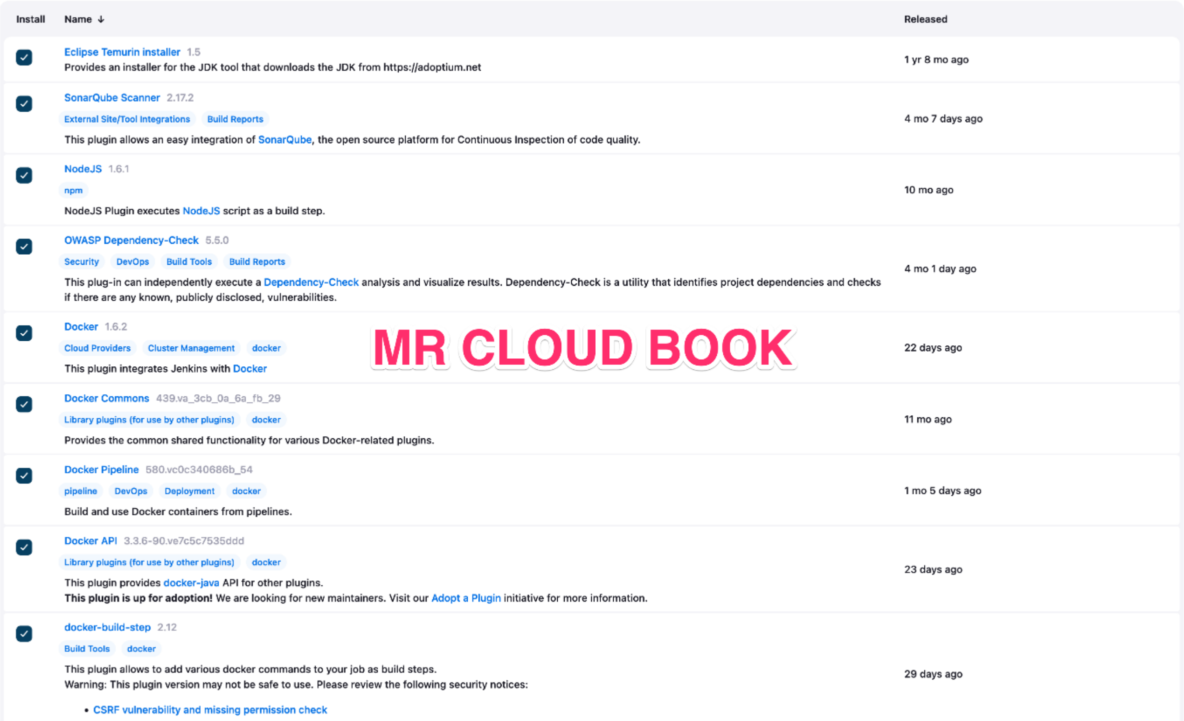
5 → Docker commons

6 → Docker pipeline

7 → Docker API

8 → Docker Build step

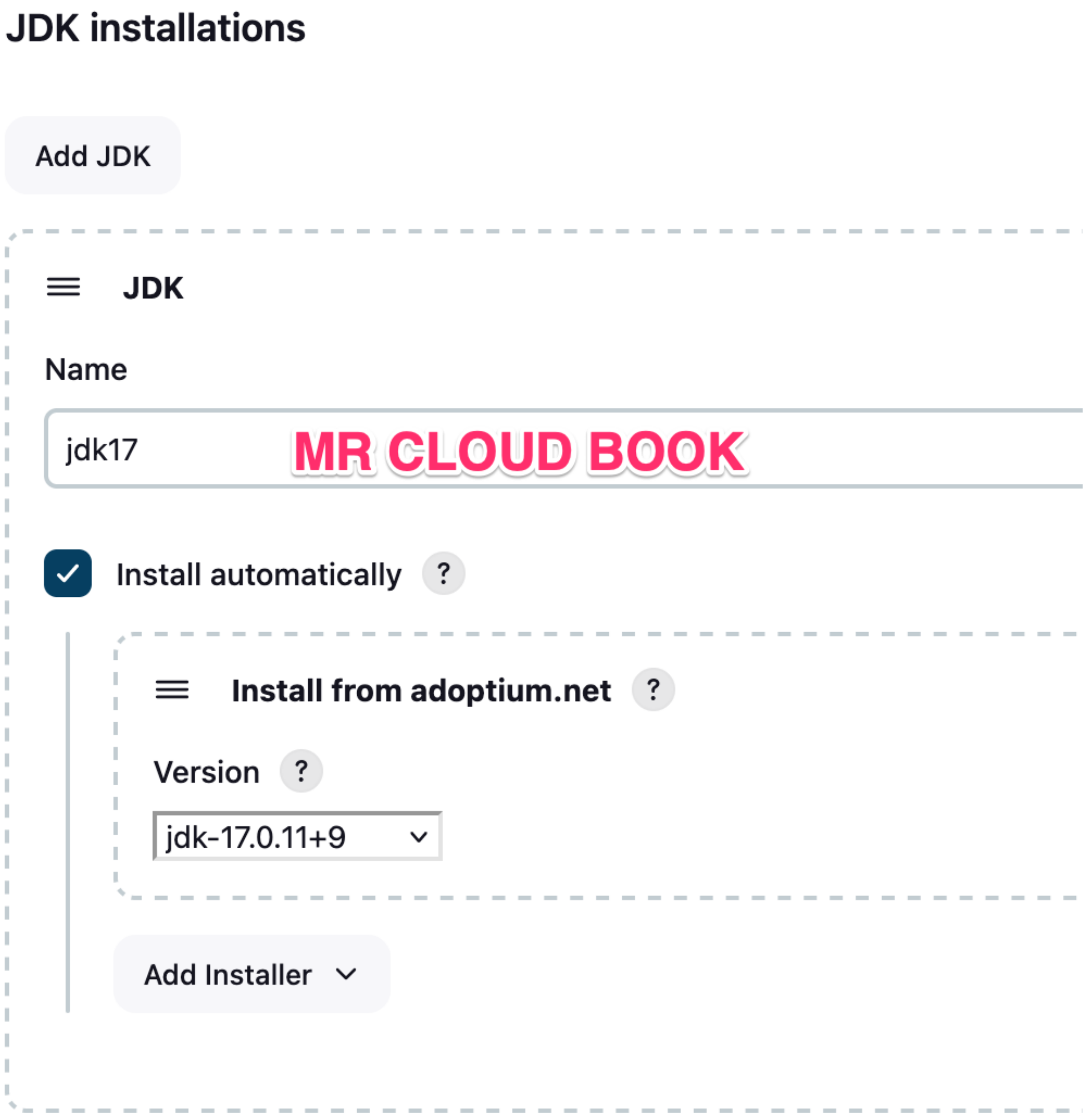
9 → Owasp Dependency Check



Install	Name ↓	Released
☑	Eclipse Temurin installer 1.5 Provides an installer for the JDK tool that downloads the JDK from https://adoptium.net	1 yr 8 mo ago
☑	SonarQube Scanner 2.17.2 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 7 days ago
☑	NodeJS 1.6.1 npm NodeJS Plugin executes NodeJS script as a build step.	10 mo ago
☑	OWASP Dependency-Check 5.5.0 Security · DevOps · Build Tools · Build Reports This plug-in can independently execute a Dependency-Check analysis and visualize results. Dependency-Check is a utility that identifies project dependencies and checks if there are any known, publicly disclosed, vulnerabilities.	4 mo 1 day ago
☑	Docker 1.6.2 Cloud Providers · Cluster Management · docker This plugin integrates Jenkins with Docker	22 days ago
☑	Docker Commons 439.va_3cb_0a_6a_fb_29 Library plugins (for use by other plugins) · docker Provides the common shared functionality for various Docker-related plugins.	11 mo ago
☑	Docker Pipeline 580.vc0c340686b_54 pipeline · DevOps · Deployment · docker Build and use Docker containers from pipelines.	1 mo 5 days ago
☑	Docker API 3.3.6-90.ve7c5c7535ddd Library plugins (for use by other plugins) · docker This plugin provides docker-java API for other plugins. **This plugin is up for adoption!** We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information.	23 days ago
☑	docker-build-step 2.12 Build Tools · docker This plugin allows to add various docker commands to your job as build steps. Warning: This plugin version may not be safe to use. Please review the following security notices: • CSRF vulnerability and missing permission check	29 days ago

Configure Java and Nodejs in Global Tool Configuration

Goto Manage Jenkins → Tools → Install JDK(17) and NodeJs(18)→ Click on Apply and Save



JDK installations

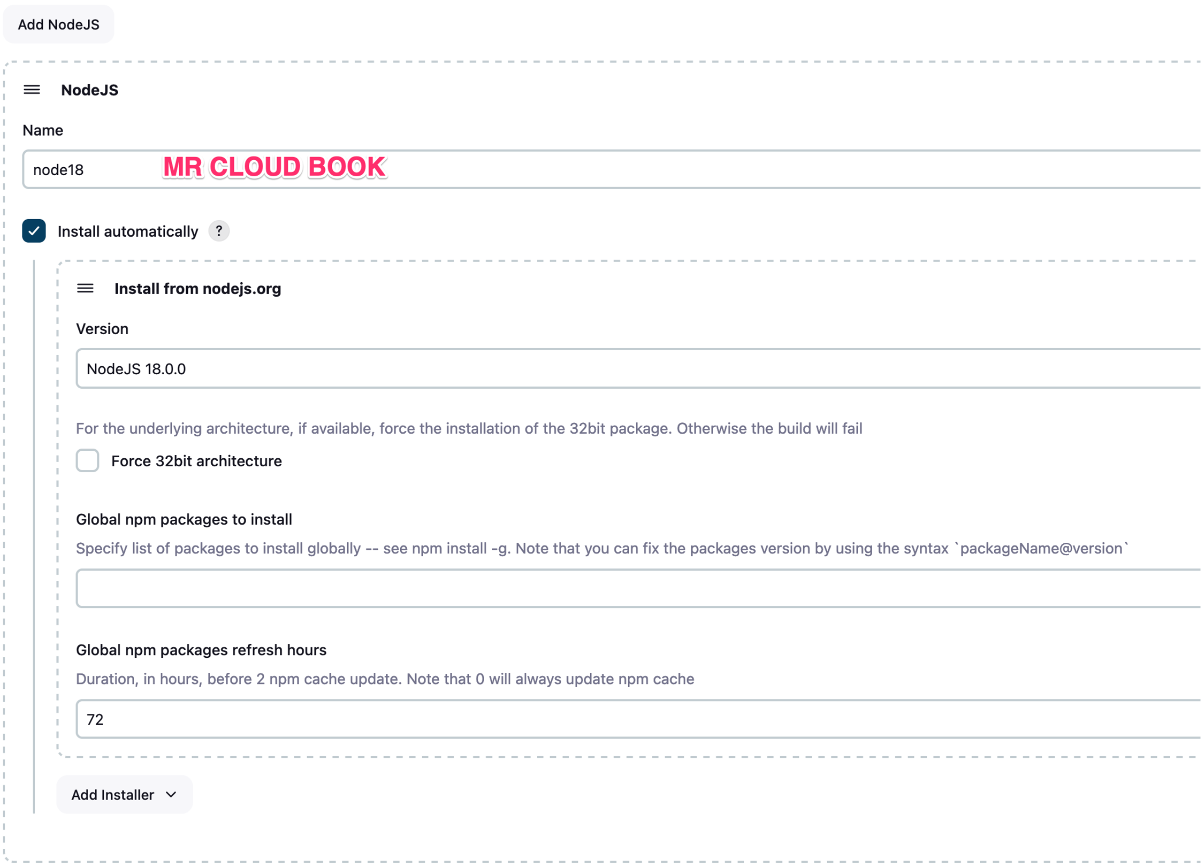
JDK

Name: jdk17

☑ Install automatically

Install from adoptium.net

Version: jdk-17.0.11+9



Add NodeJS

NodeJS

Name

node18 MR CLOUD BOOK

☑ Install automatically ?

Install from nodejs.org

Version

NodeJS 18.0.0

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

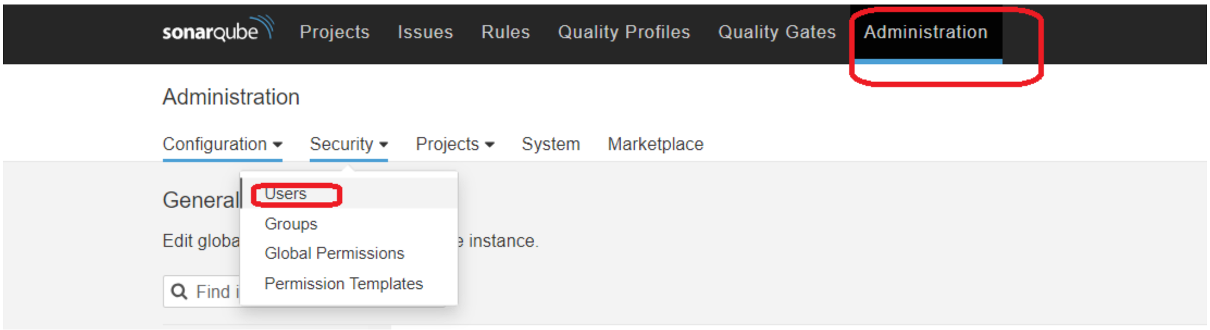
Duration, in hours, before 2 npm cache update. Note that 0 will always update npm cache

72

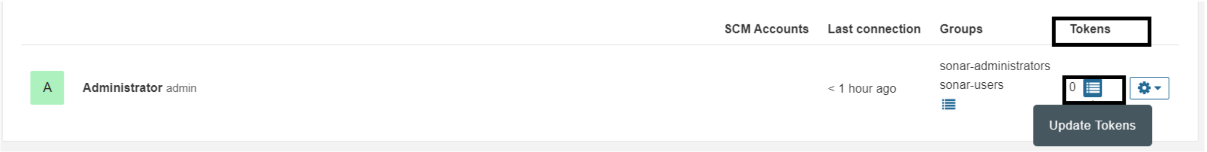
Add Installer

Grab the Public IP Address of your EC2 Instance, Sonarqube works on Port 9000, so <Public IP>:9000. Goto your Sonarqube Server.

Click on Administration → Security → Users → Click on Tokens and Update Token → Give it a name → and click on Generate Token



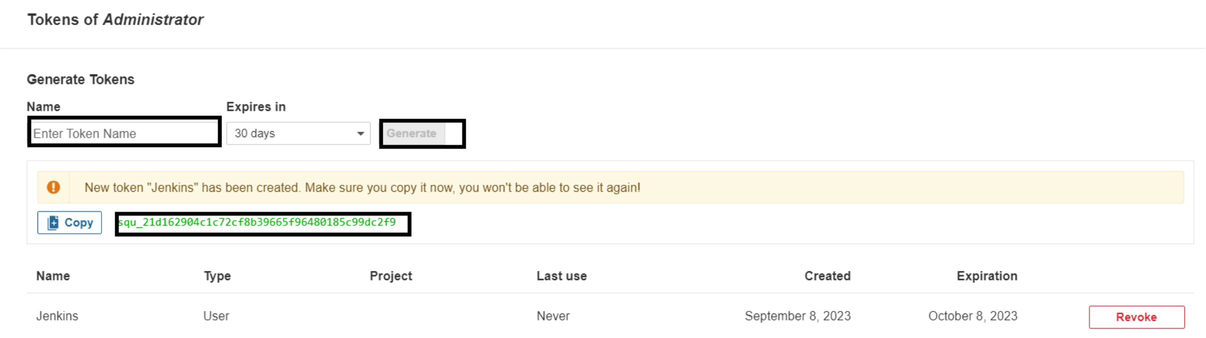
click on update Token



	SCM Accounts	Last connection	Groups	Tokens
A Administrator admin		< 1 hour ago	sonar-administrators sonar-users	0

Update Tokens

Create a token with a name and generate



Tokens of *Administrator*

Generate Tokens

Name: Enter Token Name

Expires in: 30 days

Generate

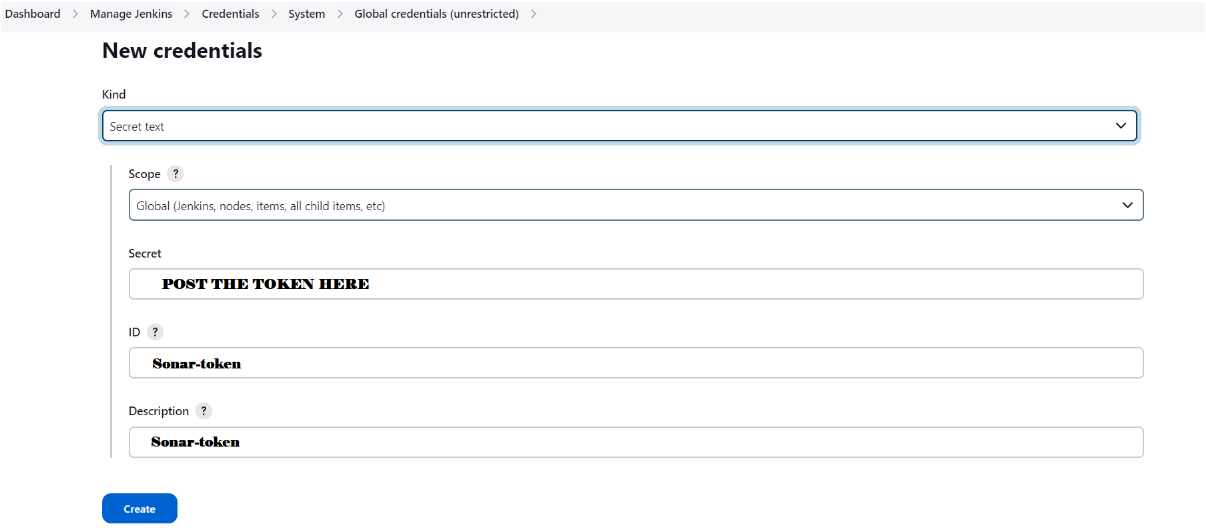
New token "Jenkins" has been created. Make sure you copy it now, you won't be able to see it again!

Copy `squ_21d162904c1c72cf8b39665f96480185c99dc2f9`

Name	Type	Project	Last use	Created	Expiration	
Jenkins	User		Never	September 8, 2023	October 8, 2023	Revoke

copy Token

Go to Jenkins Dashboard → Manage Jenkins → Credentials → Add Secret Text. It should look like this



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

New credentials

Kind: Secret text

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

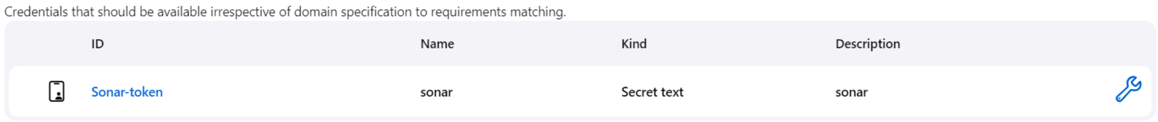
Secret: **POST THE TOKEN HERE**

ID ?: **Sonar-token**

Description ?: **Sonar-token**

Create

You will this page once you click on create

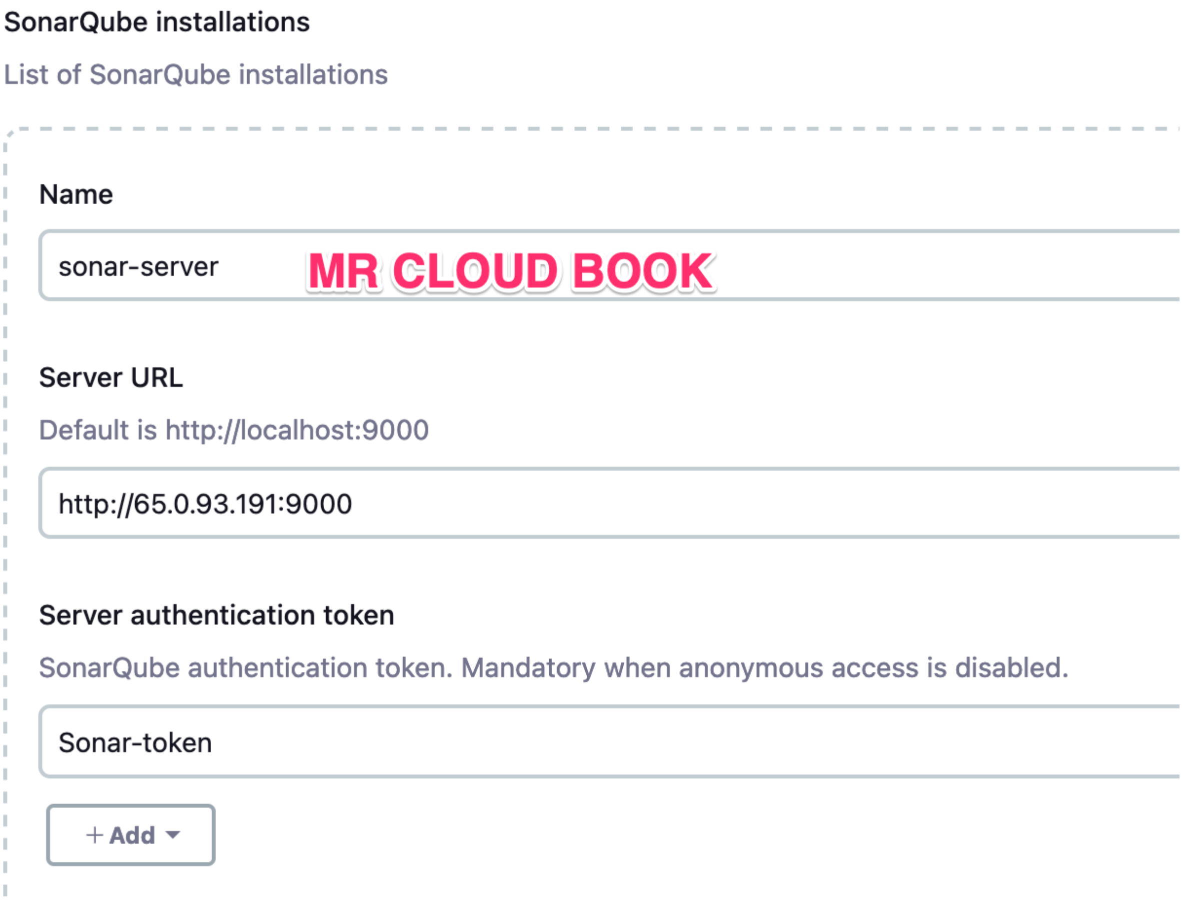


Credentials that should be available irrespective of domain specification to requirements matching.

	ID	Name	Kind	Description
	Sonar-token	sonar	Secret text	sonar

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

Search for SonarQube Installation



SonarQube installations

List of SonarQube installations

Name

sonar-server

Server URL

Default is http://localhost:9000

http://65.0.93.191:9000

Server authentication token

SonarQube authentication token. Mandatory when anonymous access is disabled.

Sonar-token

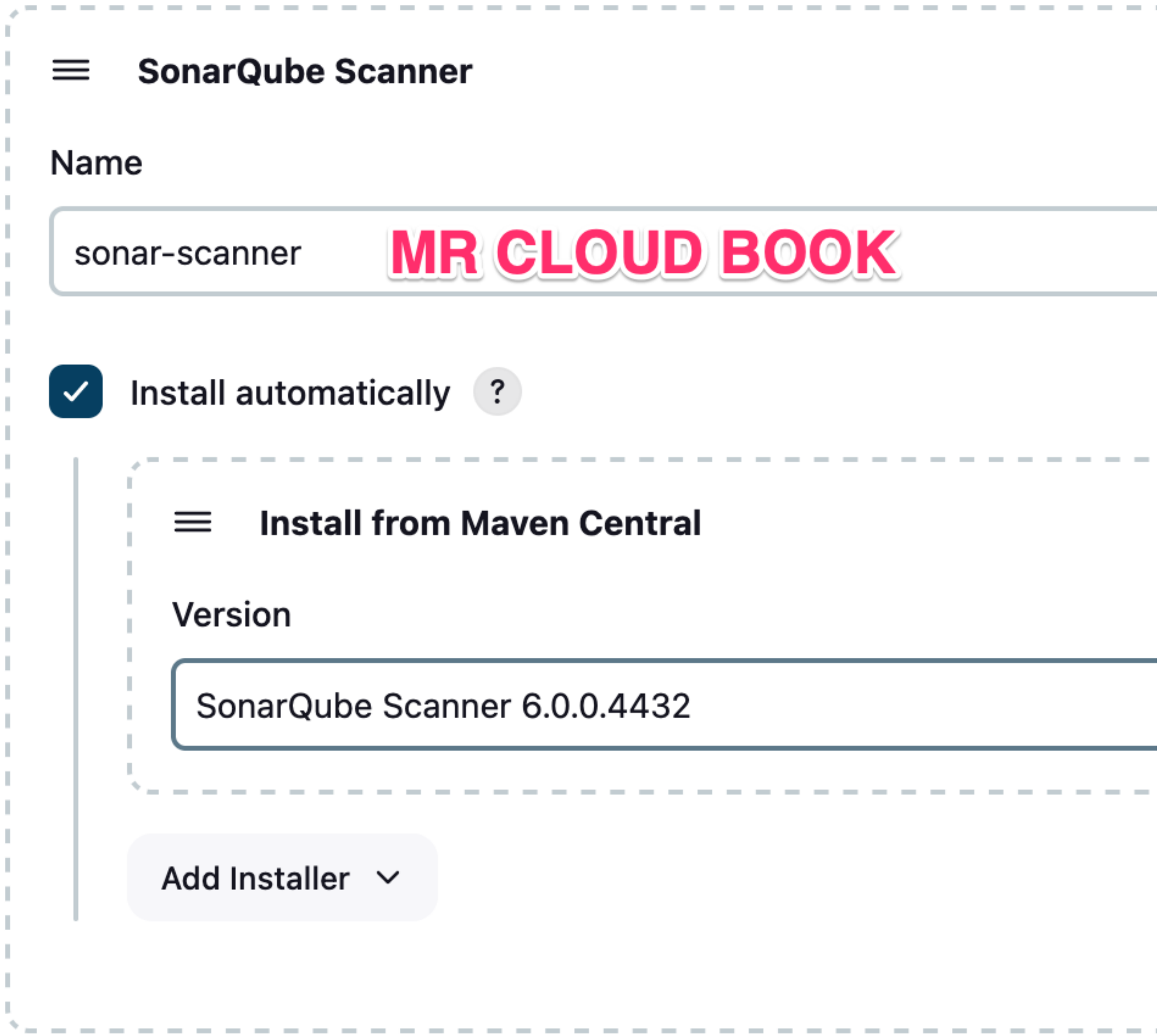
Click on Apply and Save

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



SonarQube Scanner

Name: sonar-scanner

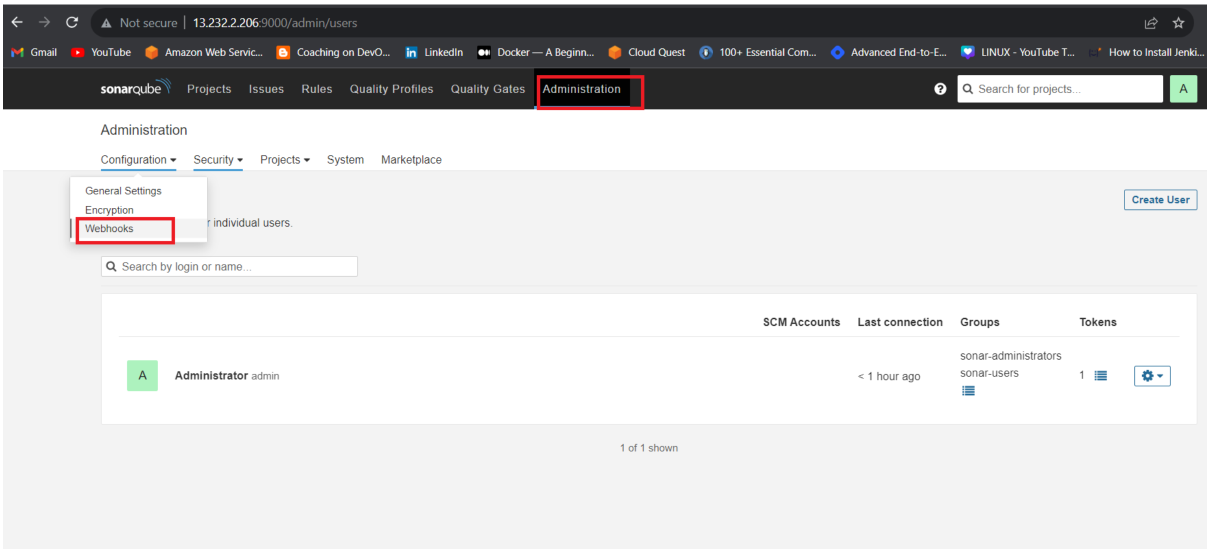
☑ Install automatically ?

Install from Maven Central

Version: SonarQube Scanner 6.0.0.4432

In the Sonarqube Dashboard add a quality gate also

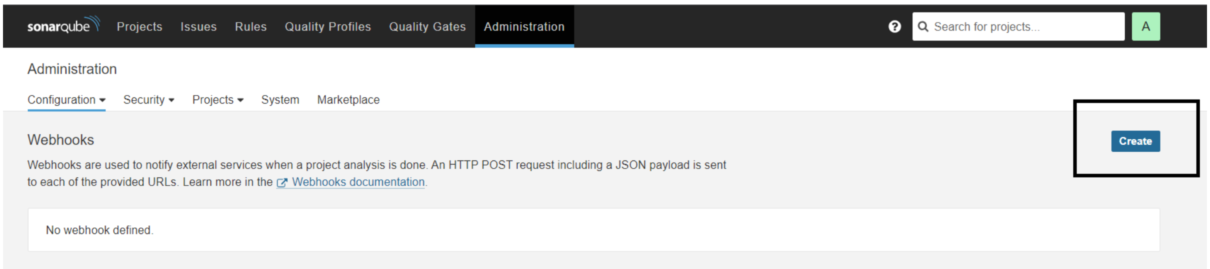
Administration–> Configuration–>Webhooks



	SCM Accounts	Last connection	Groups	Tokens
A Administrator admin		< 1 hour ago	sonar-administrators sonar-users	1

1 of 1 shown

Click on Create



Administration

Webhooks

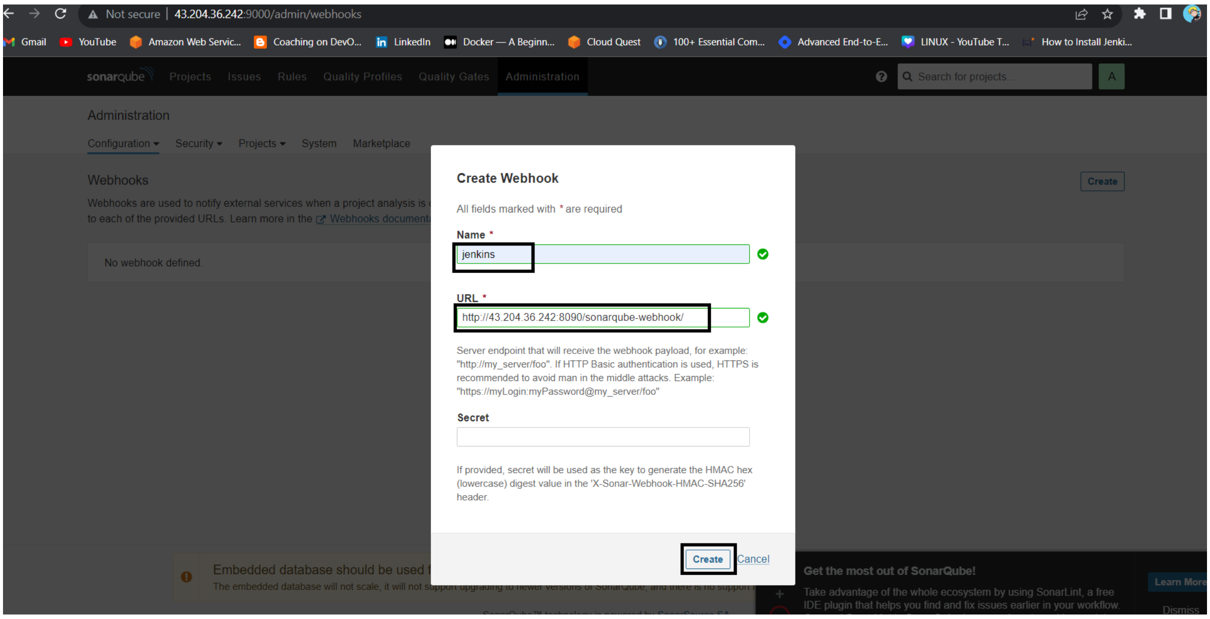
Webhooks are used to notify external services when a project analysis is done. An HTTP POST request including a JSON payload is sent to each of the provided URLs. Learn more in the Webhooks documentation.

No webhook defined.

Add details

#in url section of quality gate

http://jenkins-public-ip:8080/sonarqube-webhook/



Create Webhook

All fields marked with * are required

Name *
jenkins

URL *
http://43.204.36.242:8090/sonarqube-webhook/

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

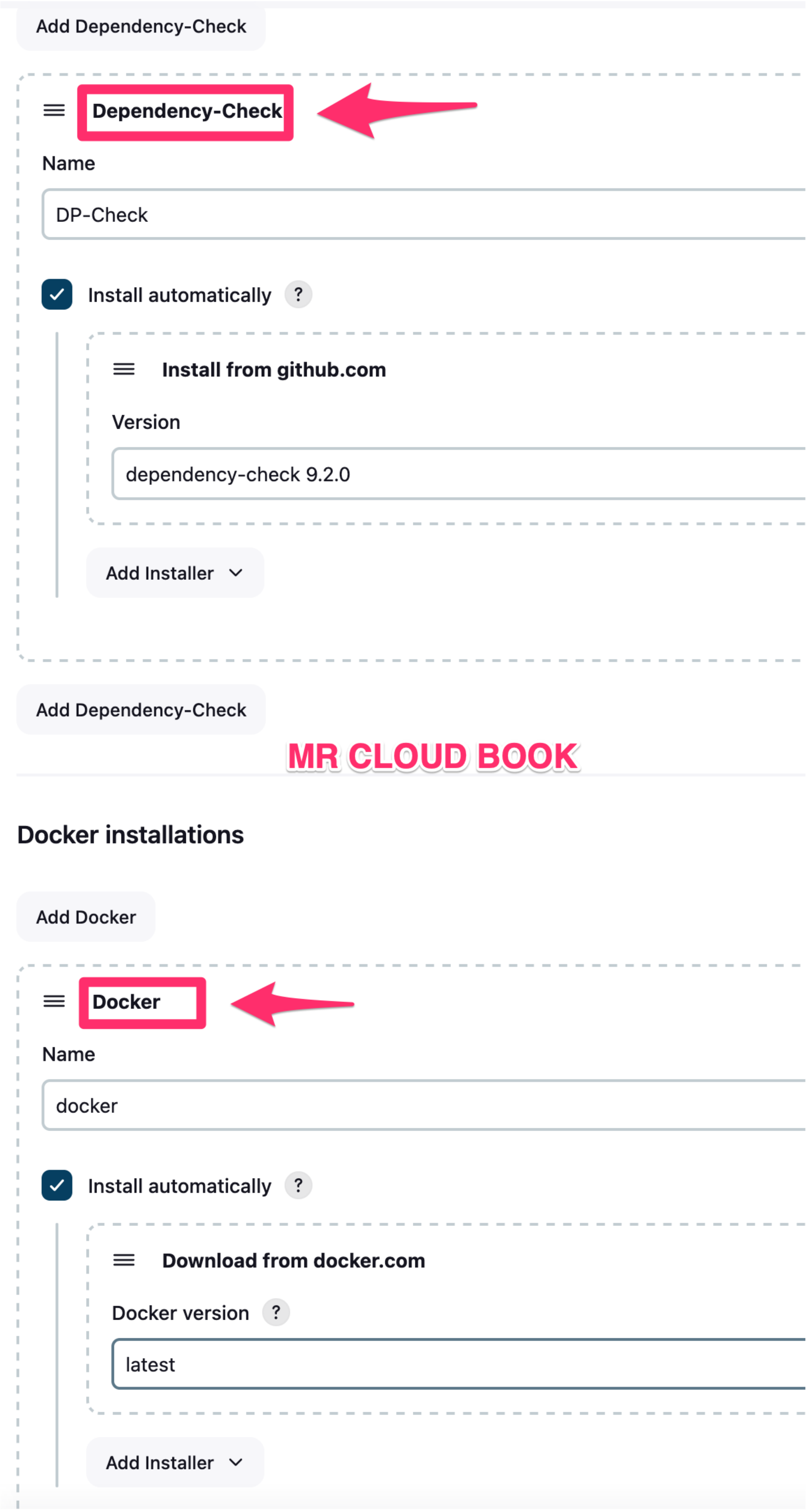
If provided, secret will be used as the key to generate the HMAC hex (lowercase) digest value in the 'X-Sonar-Webhook-HMAC-SHA256' header.

Get the most out of SonarQube!

Take advantage of the whole ecosystem by using SonarLint, a free IDE plugin that helps you find and fix issues earlier in your workflow.

First, we configured the Plugin and next, we had to configure the Tool

Goto Dashboard → Manage Jenkins → Tools →Dependency-Check & Docker



Dependency-Check

Name

DP-Check

☑ Install automatically ?

Install from github.com

Version

dependency-check 9.2.0

Docker installations

Docker

Name

docker

☑ Install automatically ?

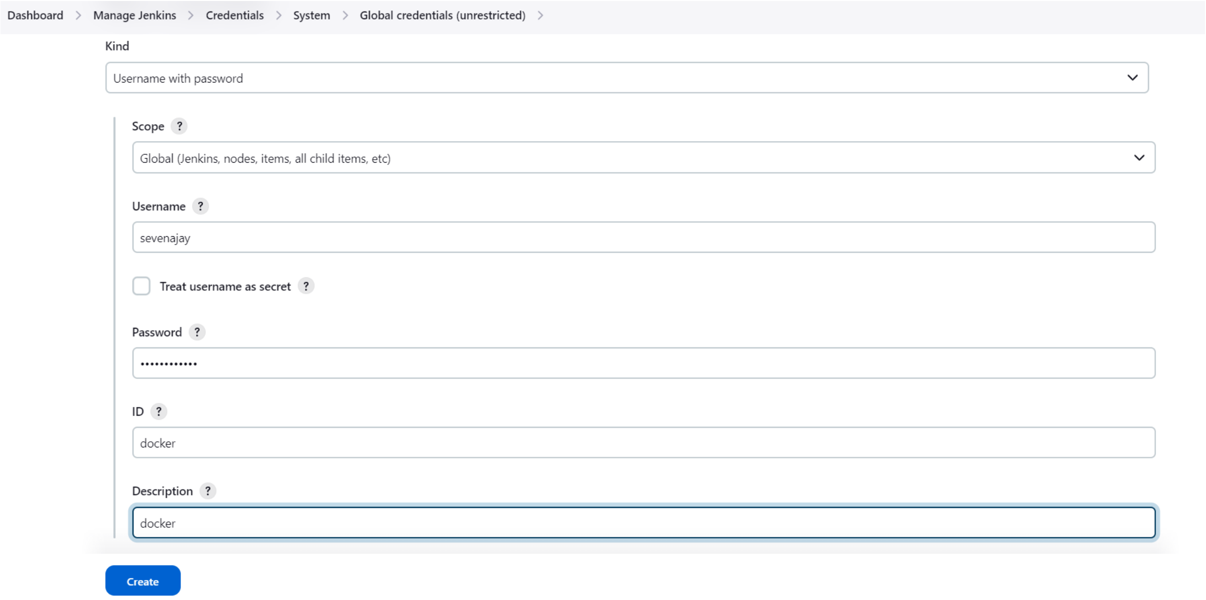
Download from docker.com

Docker version ?

latest

Go to manage Jenkins –> Credentials

Add DockerHub Username and Password under Global Credentials



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

Kind: Username with password

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

Username: sevenajay

□ Treat username as secret

Password: ••••••••••••

ID: docker

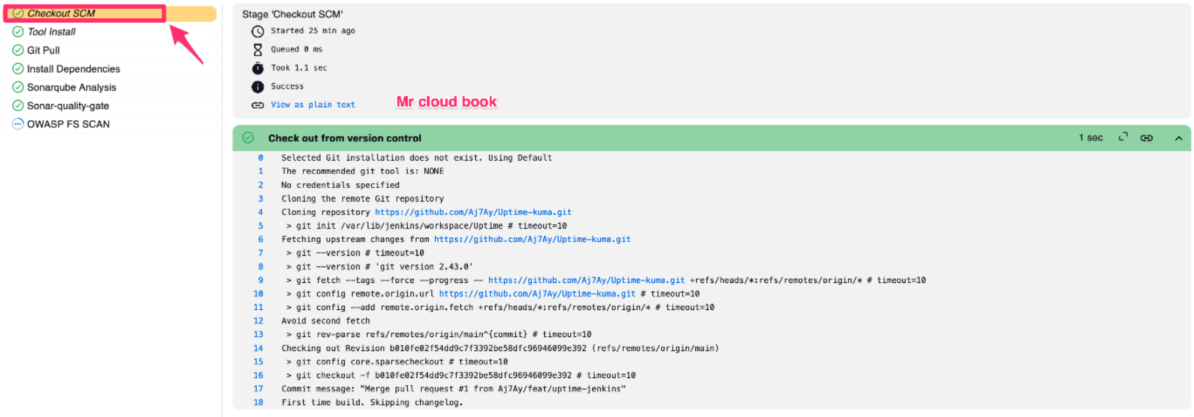
Description: docker

Create Job for Uptime-kuma

Add this stage to Pipeline Script

```
pipeline{
  agent any
  tools{
    jdk 'jdk17'
    nodejs 'node18'
  }
  environment {
    SCANNER_HOME=tool 'sonar-scanner'
  }
  stages {
    stage('Checkout from Git'){
      steps{
        git branch: 'main', url: 'https://github.com/Aj7Ay/uptime.git'
      }
    }
    stage('Install Dependencies') {
      steps {
        sh "npm install"
      }
    }
    stage("Sonarqube Analysis "){
      steps{
        withSonarQubeEnv('sonar-server') {
          sh ''' $SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=uptime \
          -Dsonar.projectKey=uptime '''
        }
      }
    }
    stage("quality gate"){
     steps {
        script {
          waitForQualityGate abortPipeline: false, credentialsId: 'Sonar-token'
        }
      }
    }
    stage('OWASP FS SCAN') {
      steps {
        dependencyCheck additionalArguments: '--scan ./ --disableYarnAudit --disableNodeAudit', odcInstallation: 'DP-Check'
        dependencyCheckPublisher pattern: '**/dependency-check-report.xml'
      }
    }
    stage('TRIVY FS SCAN') {
      steps {
        sh "trivy fs . &gt; trivyfs.json"
      }
    }
    stage("Docker Build &amp; Push"){
        steps{
            script{
                withDockerRegistry(credentialsId: 'docker', toolName: 'docker'){
                    sh "docker build -t uptime ."
                    sh "docker tag uptime sevenajay/uptime:latest "
                    sh "docker push sevenajay/uptime:latest "
                }
            }
        }
    }
    stage("TRIVY"){
        steps{
            sh "trivy image sevenajay/uptime:latest &gt; trivy.json"
        }
    }
    stage ("Remove container") {
        steps{
            sh "docker stop uptime | true"
            sh "docker rm uptime | true"
         }
    }
    stage('Deploy to container'){
        steps{
            sh 'docker run -d --name chatbot -v /var/run/docker.sock:/var/run/docker.sock -p 3001:3001 sevenajay/uptime:latest'
        }
    }
}
```

Git checkout

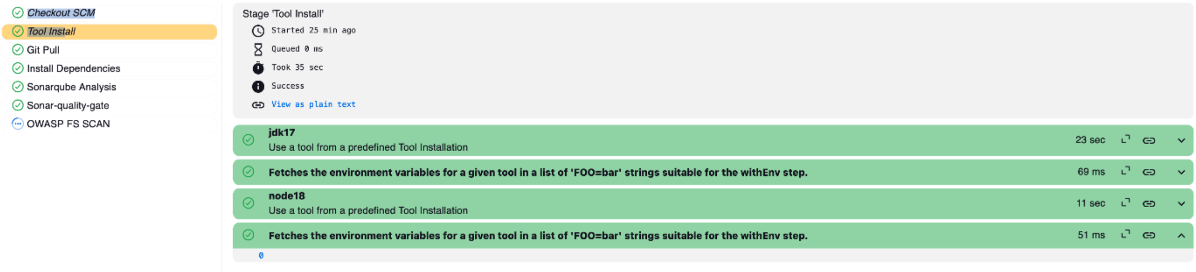


Stage 'Checkout SCM'
Started 25 min ago
Queued 0 ms
Took 1.1 sec
Success
View as plain text

Check out from version control

```
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/Aj7Ay/Uptime-kuma.git
 > git init /var/lib/jenkins/workspace/Uptime # timeout=10
Fetching upstream changes from https://github.com/Aj7Ay/Uptime-kuma.git
 > git --version # timeout=10
 > git --version # 'git version 2.43.0'
 > git fetch --tags --force --progress -- https://github.com/Aj7Ay/Uptime-kuma.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/Aj7Ay/Uptime-kuma.git # timeout=10
 > git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
 > git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision b010fe02f54dd9c7f3392be58dfc96946099e392 (refs/remotes/origin/main)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f b010fe02f54dd9c7f3392be58dfc96946099e392 # timeout=10
Commit message: "Merge pull request #1 from Aj7Ay/feat/uptime-jenkins"
First time build. Skipping changelog.
```

Tool install



Stage 'Tool Install'
Started 25 min ago
Queued 0 ms
Took 35 sec
Success
View as plain text

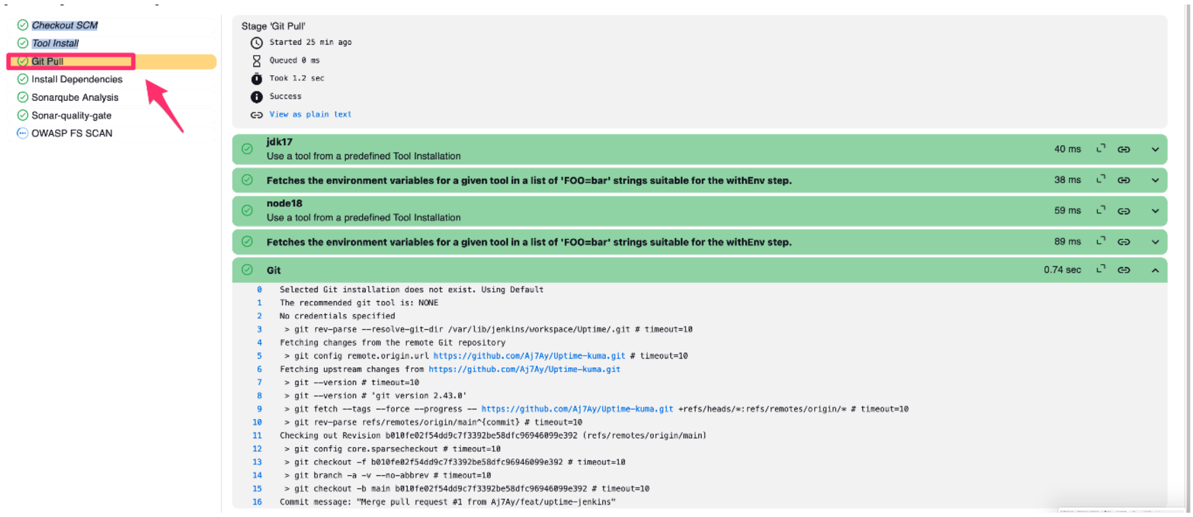
jdk17
Use a tool from a predefined Tool Installation

Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step.

node18
Use a tool from a predefined Tool Installation

Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step.

Git pull



Stage 'Git Pull'
Started 25 min ago
Queued 0 ms
Took 1.2 sec
Success
View as plain text

jdk17
Use a tool from a predefined Tool Installation

Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step.

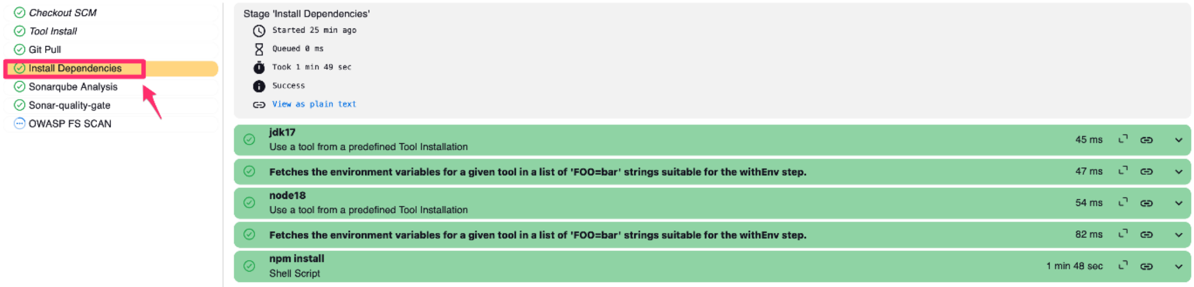
node18
Use a tool from a predefined Tool Installation

Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step.

Git

```
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
 > git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/Uptime/.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/Aj7Ay/Uptime-kuma.git # timeout=10
Fetching upstream changes from https://github.com/Aj7Ay/Uptime-kuma.git
 > git --version # timeout=10
 > git --version # 'git version 2.43.0'
 > git fetch --tags --force --progress -- https://github.com/Aj7Ay/Uptime-kuma.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision b010fe02f54dd9c7f3392be58dfc96946099e392 (refs/remotes/origin/main)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f b010fe02f54dd9c7f3392be58dfc96946099e392 # timeout=10
 > git branch -a -v --no-abbrev # timeout=10
 > git checkout -b main b010fe02f54dd9c7f3392be58dfc96946099e392 # timeout=10
Commit message: "Merge pull request #1 from Aj7Ay/feat/uptime-jenkins"
```

NPM install



Stage 'Install Dependencies'
Started 25 min ago
Queued 0 ms
Took 1 min 49 sec
Success
View as plain text

jdk17
Use a tool from a predefined Tool Installation

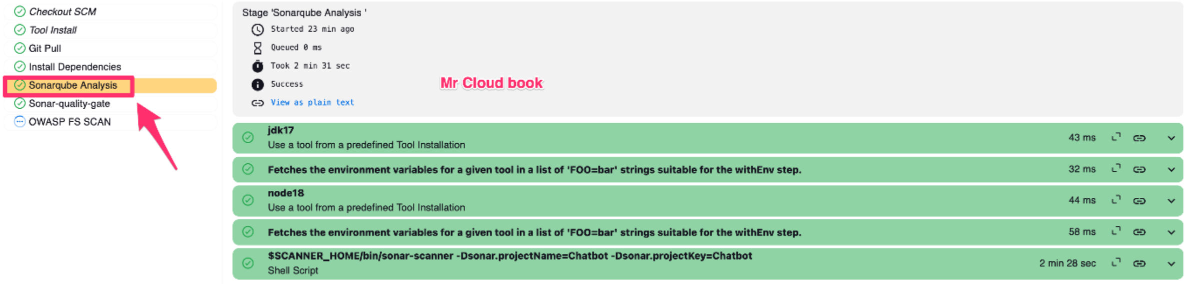
Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step.

node18
Use a tool from a predefined Tool Installation

Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step.

npm install
Shell Script

Sonarqube analysis

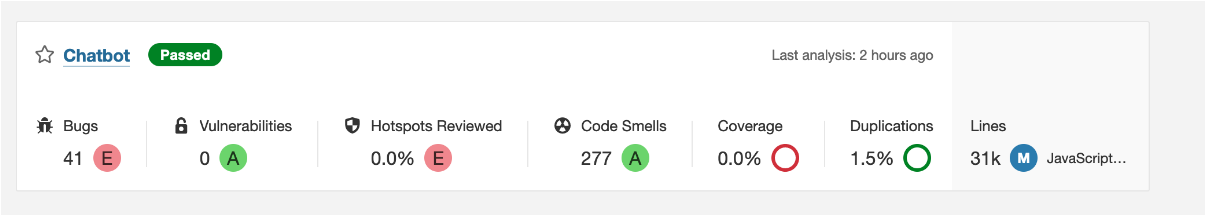


Checkout SCM
Tool Install
Git Pull
Install Dependencies
Sonarqube Analysis
Sonar-quality-gate
OWASP FS SCAN

Stage 'Sonarqube Analysis '
Started 23 min ago
Queued 0 ms
Took 2 min 31 sec
Success
View as plain text

Mr Cloud book

jdk17 — Use a tool from a predefined Tool Installation — 43 ms
Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step. — 32 ms
node18 — Use a tool from a predefined Tool Installation — 44 ms
Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step. — 58 ms
$SCANNER_HOME/bin/sonar-scanner -Dsonar.projectName=Chatbot -Dsonar.projectKey=Chatbot — Shell Script — 2 min 28 sec

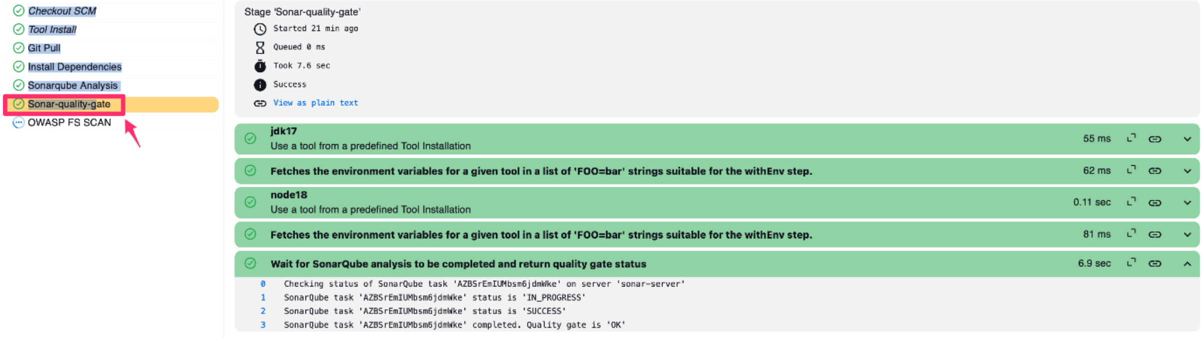


Chatbot Passed

Last analysis: 2 hours ago

Bugs	Vulnerabilities	Hotspots Reviewed	Code Smells	Coverage	Duplications	Lines
41 E	0 A	0.0% E	277 A	0.0%	1.5%	31k M JavaScript...

Quality Gate



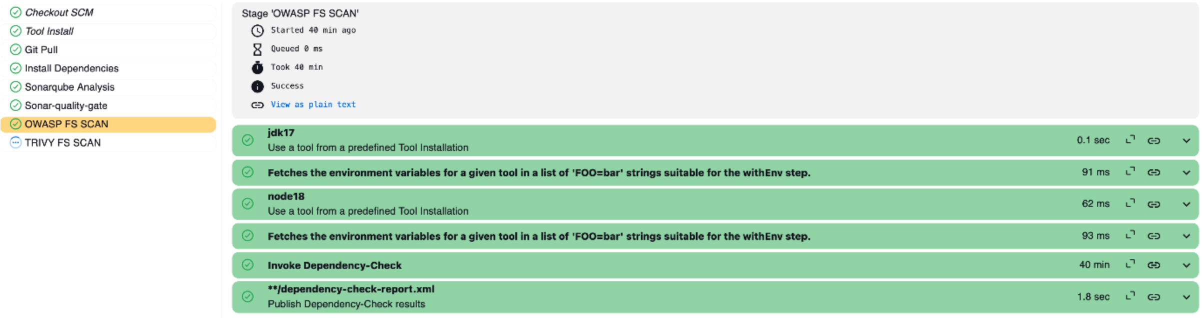
Checkout SCM
Tool Install
Git Pull
Install Dependencies
Sonarqube Analysis
Sonar-quality-gate
OWASP FS SCAN

Stage 'Sonar-quality-gate'
Started 21 min ago
Queued 0 ms
Took 7.6 sec
Success
View as plain text

jdk17 — Use a tool from a predefined Tool Installation — 55 ms
Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step. — 62 ms
node18 — Use a tool from a predefined Tool Installation — 0.11 sec
Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step. — 81 ms
Wait for SonarQube analysis to be completed and return quality gate status — 6.9 sec

```
Checking status of SonarQube task 'AZBSrEmIUMbsm6jdmWke' on server 'sonar-server'
SonarQube task 'AZBSrEmIUMbsm6jdmWke' status is 'IN_PROGRESS'
SonarQube task 'AZBSrEmIUMbsm6jdmWke' status is 'SUCCESS'
SonarQube task 'AZBSrEmIUMbsm6jdmWke' completed. Quality gate is 'OK'
```

Owasp

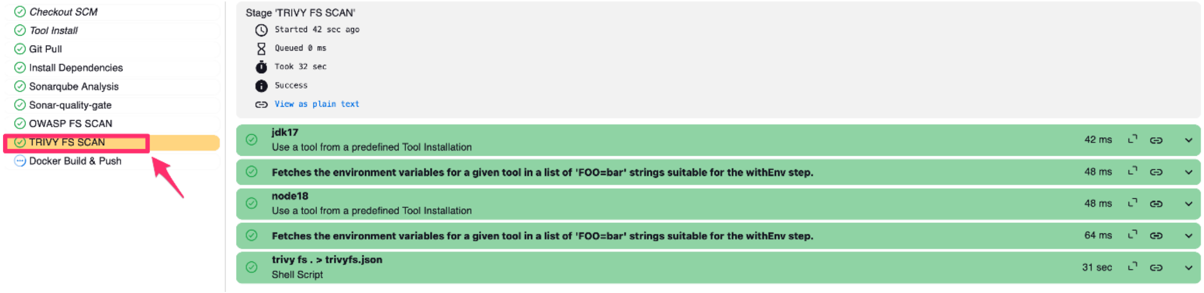


Checkout SCM
Tool Install
Git Pull
Install Dependencies
Sonarqube Analysis
Sonar-quality-gate
OWASP FS SCAN
TRIVY FS SCAN

Stage 'OWASP FS SCAN'
Started 40 min ago
Queued 0 ms
Took 40 min
Success
View as plain text

jdk17 — Use a tool from a predefined Tool Installation — 0.1 sec
Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step. — 91 ms
node18 — Use a tool from a predefined Tool Installation — 62 ms
Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step. — 93 ms
Invoke Dependency-Check — 40 min
****/dependency-check-report.xml** — Publish Dependency-Check results — 1.8 sec

Trivy FS

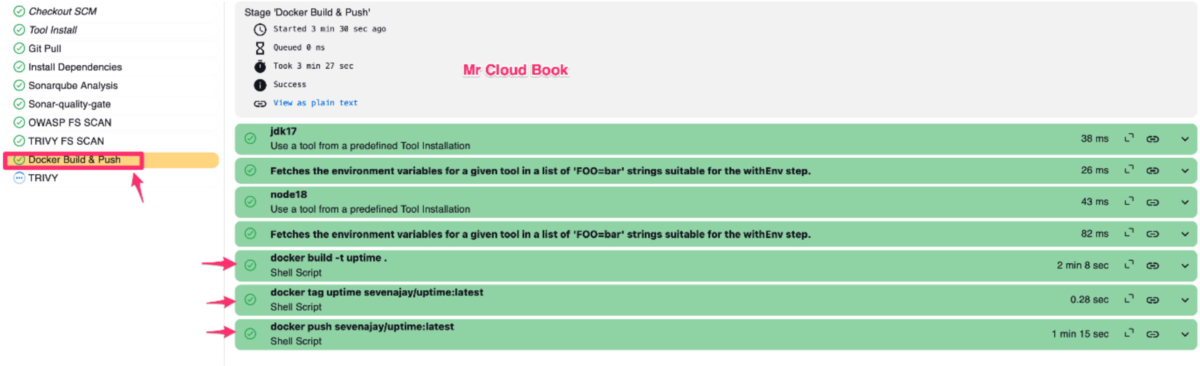


Checkout SCM
Tool Install
Git Pull
Install Dependencies
Sonarqube Analysis
Sonar-quality-gate
OWASP FS SCAN
TRIVY FS SCAN
Docker Build & Push

Stage 'TRIVY FS SCAN'
Started 42 sec ago
Queued 0 ms
Took 32 sec
Success
View as plain text

jdk17 — Use a tool from a predefined Tool Installation — 42 ms
Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step. — 48 ms
node18 — Use a tool from a predefined Tool Installation — 48 ms
Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step. — 64 ms
trivy fs . > trivyfs.json — Shell Script — 31 sec

Docker Build and push



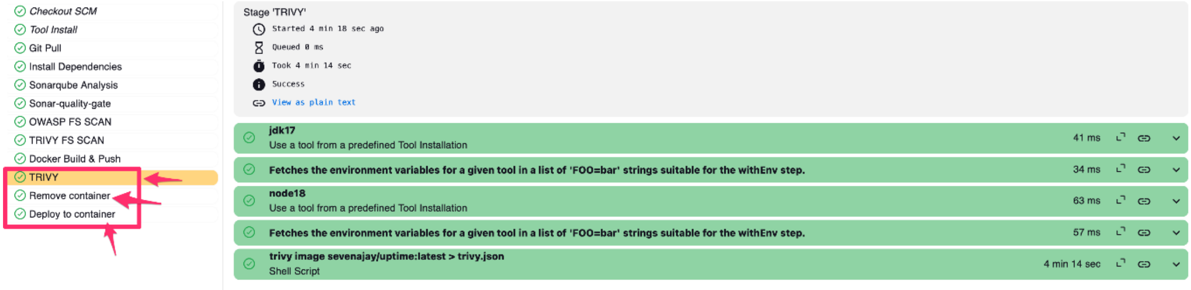
Checkout SCM
Tool Install
Git Pull
Install Dependencies
Sonarqube Analysis
Sonar-quality-gate
OWASP FS SCAN
TRIVY FS SCAN
Docker Build & Push
TRIVY

```
Stage 'Docker Build & Push'
Started 3 min 30 sec ago
Queued 0 ms
Took 3 min 27 sec
Success
View as plain text
```

Mr Cloud Book

```
jdk17
Use a tool from a predefined Tool Installation                                                      38 ms
Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step.   26 ms
node18
Use a tool from a predefined Tool Installation                                                      43 ms
Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step.   82 ms
docker build -t uptime .
Shell Script                                                                                        2 min 8 sec
docker tag uptime sevenajay/uptime:latest
Shell Script                                                                                        0.28 sec
docker push sevenajay/uptime:latest
Shell Script                                                                                        1 min 15 sec
```

Trivy Image scan and deploy to Docker



Checkout SCM
Tool Install
Git Pull
Install Dependencies
Sonarqube Analysis
Sonar-quality-gate
OWASP FS SCAN
TRIVY FS SCAN
Docker Build & Push
TRIVY
Remove container
Deploy to container

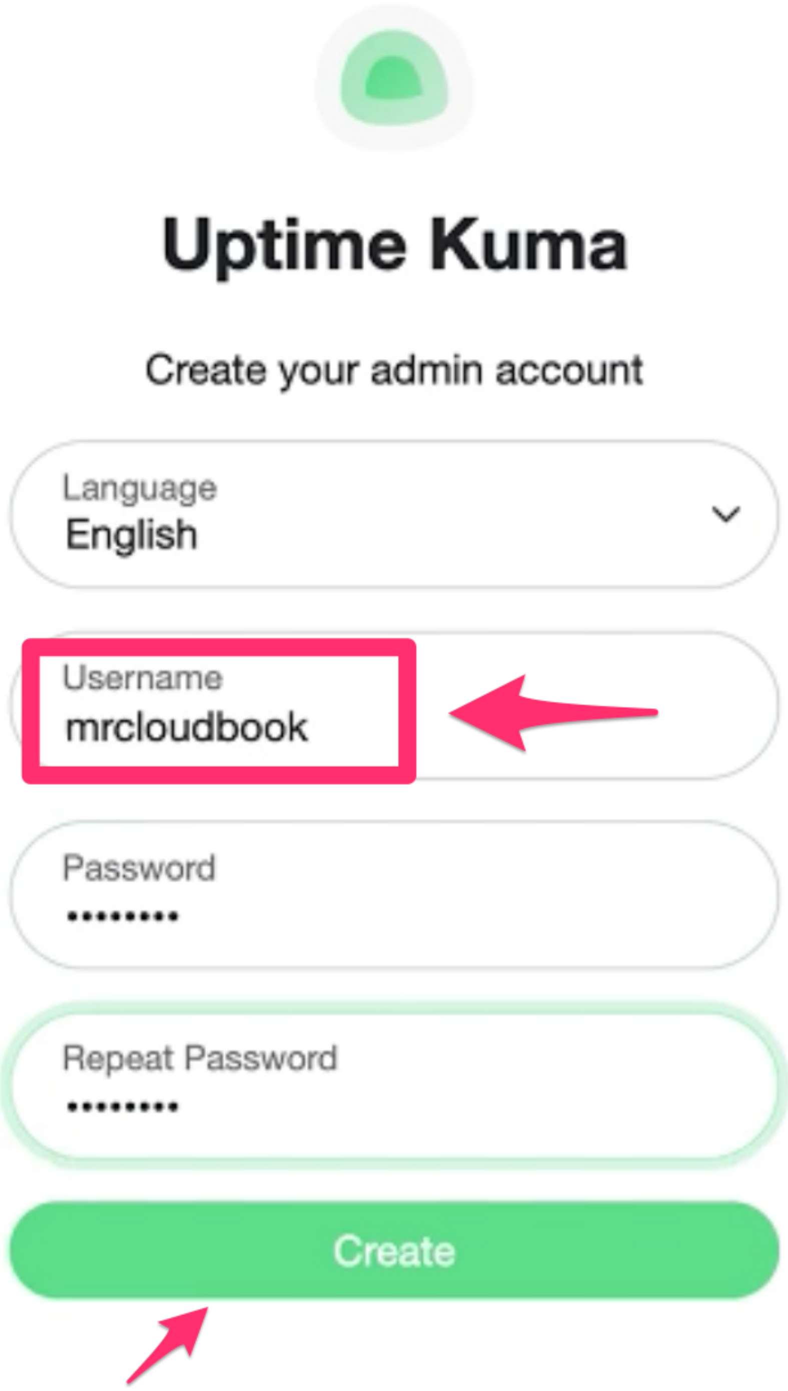
```
Stage 'TRIVY'
Started 4 min 18 sec ago
Queued 0 ms
Took 4 min 14 sec
Success
View as plain text
```

```
jdk17
Use a tool from a predefined Tool Installation                                                      41 ms
Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step.   34 ms
node18
Use a tool from a predefined Tool Installation                                                      63 ms
Fetches the environment variables for a given tool in a list of 'FOO=bar' strings suitable for the withEnv step.   57 ms
trivy image sevenajay/uptime:latest > trivy.json
Shell Script                                                                                        4 min 14 sec
```

Copy the Public of Server

public-ip&gt;:30001

Output Login Page



Uptime Kuma

Create your admin account

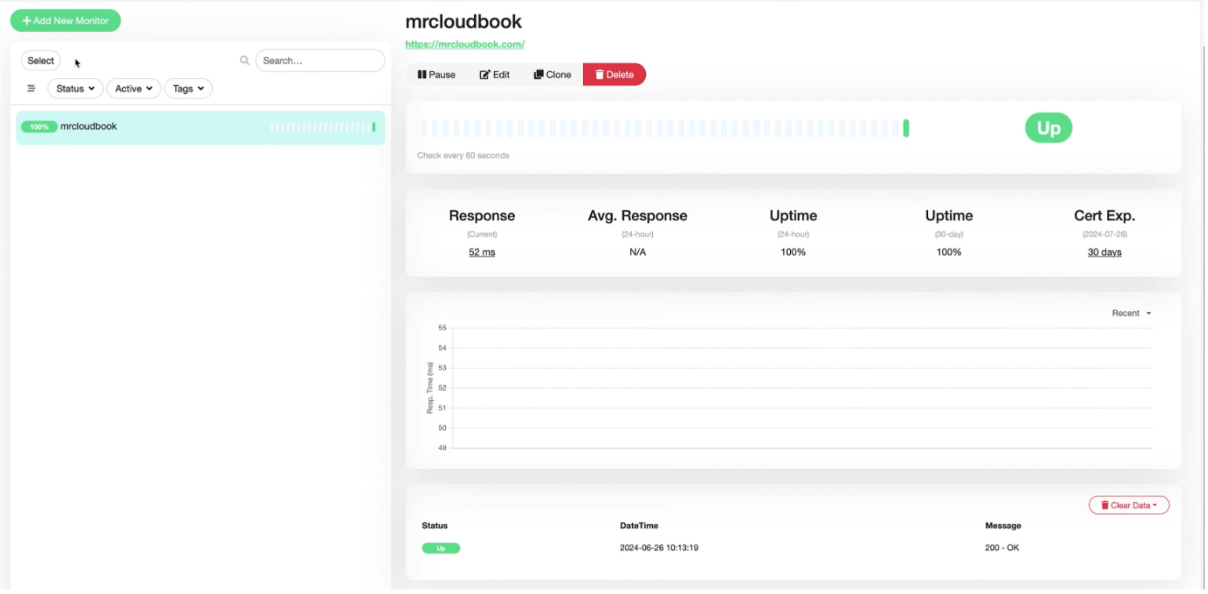
Language
English

Username
mrcloudbook

Password
•••••••••

Repeat Password
•••••••••

Output



Response (Current)	Avg. Response (24-hour)	Uptime (24-hour)	Uptime (30-day)	Cert Exp. (2024-07-26)
52 ms	N/A	100%	100%	30 days

Status	DateTime	Message
Up	2024-06-26 10:13:19	200 - OK

Twilio webhook setup

Clone the Github Repo

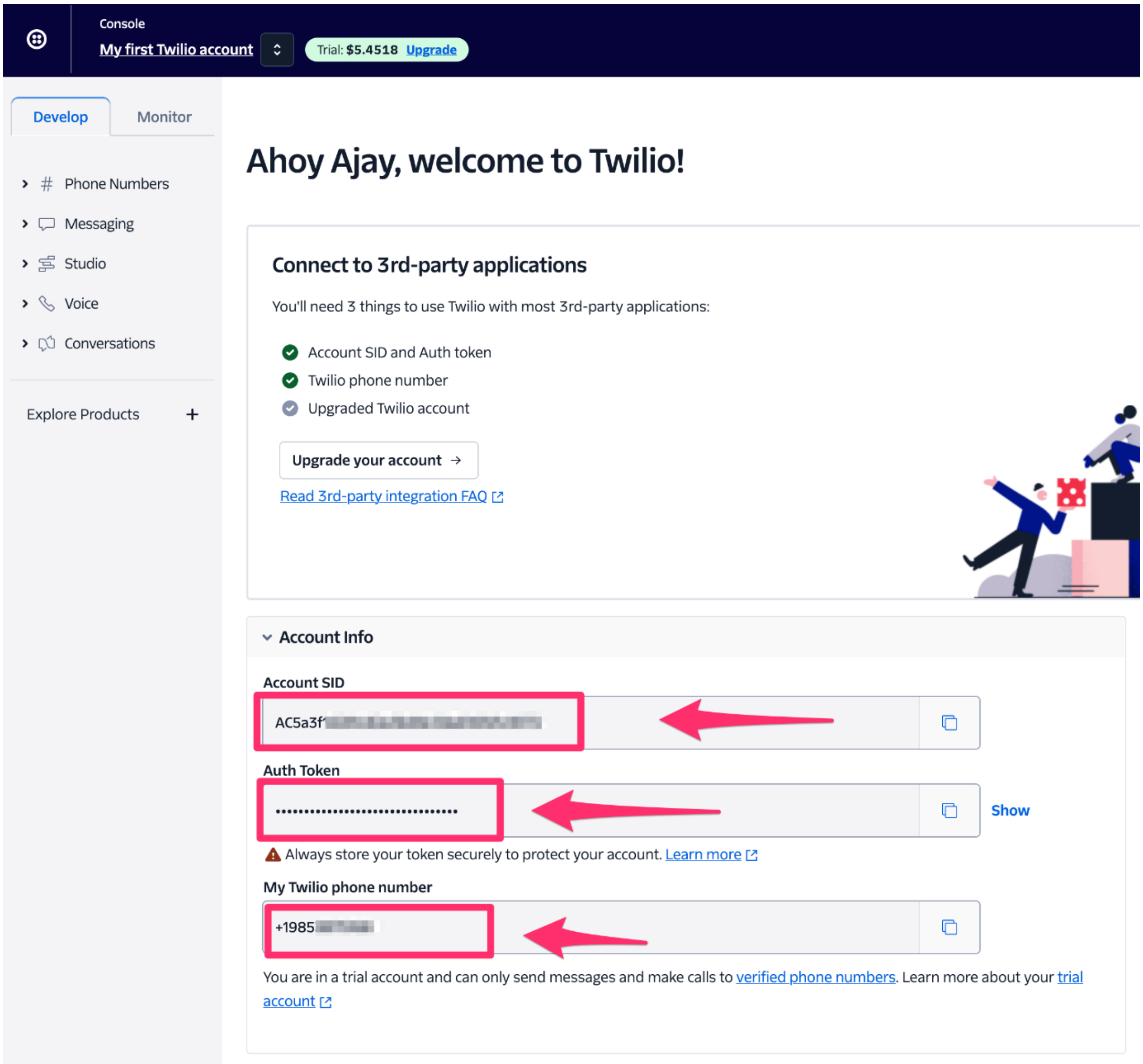
git clone https://github.com/Aj7Ay/Uptime-kuma.git

cd Uptime-kuma

cd Webhook

Create an Twilio Account From Here

Go to Console
copy Account SID & Account Auth Token and Mobile Number



Ahoy Ajay, welcome to Twilio!

Connect to 3rd-party applications

You'll need 3 things to use Twilio with most 3rd-party applications:

- ✓ Account SID and Auth token
- ✓ Twilio phone number
- ✓ Upgraded Twilio account

Account Info

Account SID: AC5a3f[redacted]

Auth Token: ••••••••••••••••••••••••••••••••• Show

Always store your token securely to protect your account. Learn more

My Twilio phone number: +1985[redacted]

You are in a trial account and can only send messages and make calls to verified phone numbers. Learn more about your trial account

Update the app.py

```
from flask import Flask

from twilio.rest import Client


app = Flask(__name__)


@app.route('/trigger-calls', methods=['GET', 'POST'])

def trigger_call():

    account_sid = '&lt;acc-sid&gt;'

    auth_token = '&lt;acc-auth&gt;'

    client = Client(account_sid, auth_token)
    # List of numbers to call
    numbers_to_call = ['&lt;country-code-number&gt;', '&lt;country-code-number&gt;']
    call_sid = []  # To store Call SIDs

    for number in numbers_to_call:
        call = client.calls.create(
                        twim='&lt;Response&gt;&lt;Say&gt;"change Alert message 
here"&lt;/Say&gt;&lt;/Response&gt;',
                        to=number,
                        from_='&lt;twilio-number&gt;'
                    )
        call_sids.append(call.sid)

    # Joining all the Call SIDs to return in response
    call_sids_str = ', 'join(call_sids)
    return f"Call initiated with SIDs: {call_sids_str}", 200


if __name__ == '__main__':
    app.run(host'0.0.0.0', port=5000)
```

Now install These on Server

```
sudo apt install python3-pip
sudo apt install python3.12-venv
python3 -m venv venv
source venv/bin/activate
```

Now install Requirements

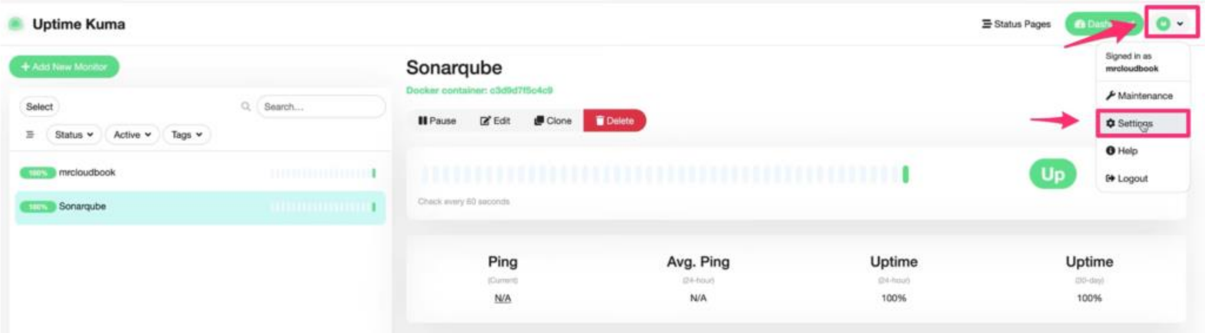
```
pip install -r requirements.txt
```

Create the Docker webhook

```
cd Uptime-kuma/webhook
docker build -t webhook .
docker run -d --name webhook -p 5000:5000 webhook:latest
```

webhook runs on 5000 port

Go to Uptime Kuma select Profile –> Settings

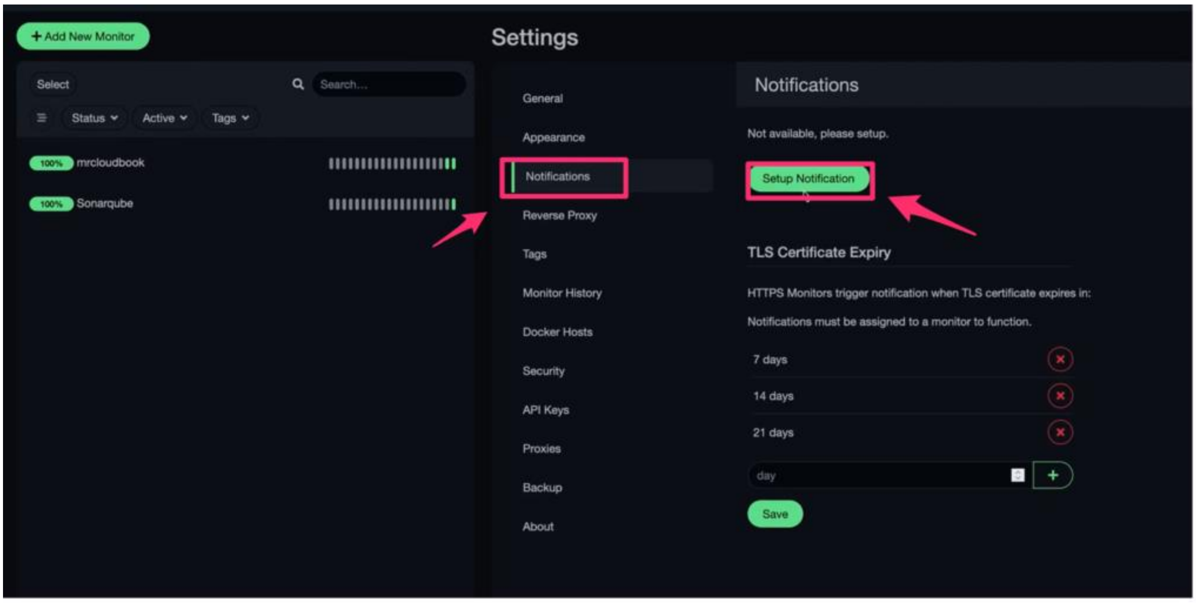


Sonarqube

Docker container: c3d9d7f5c4c9

Check every 60 seconds

Select Notification –> setup Notification



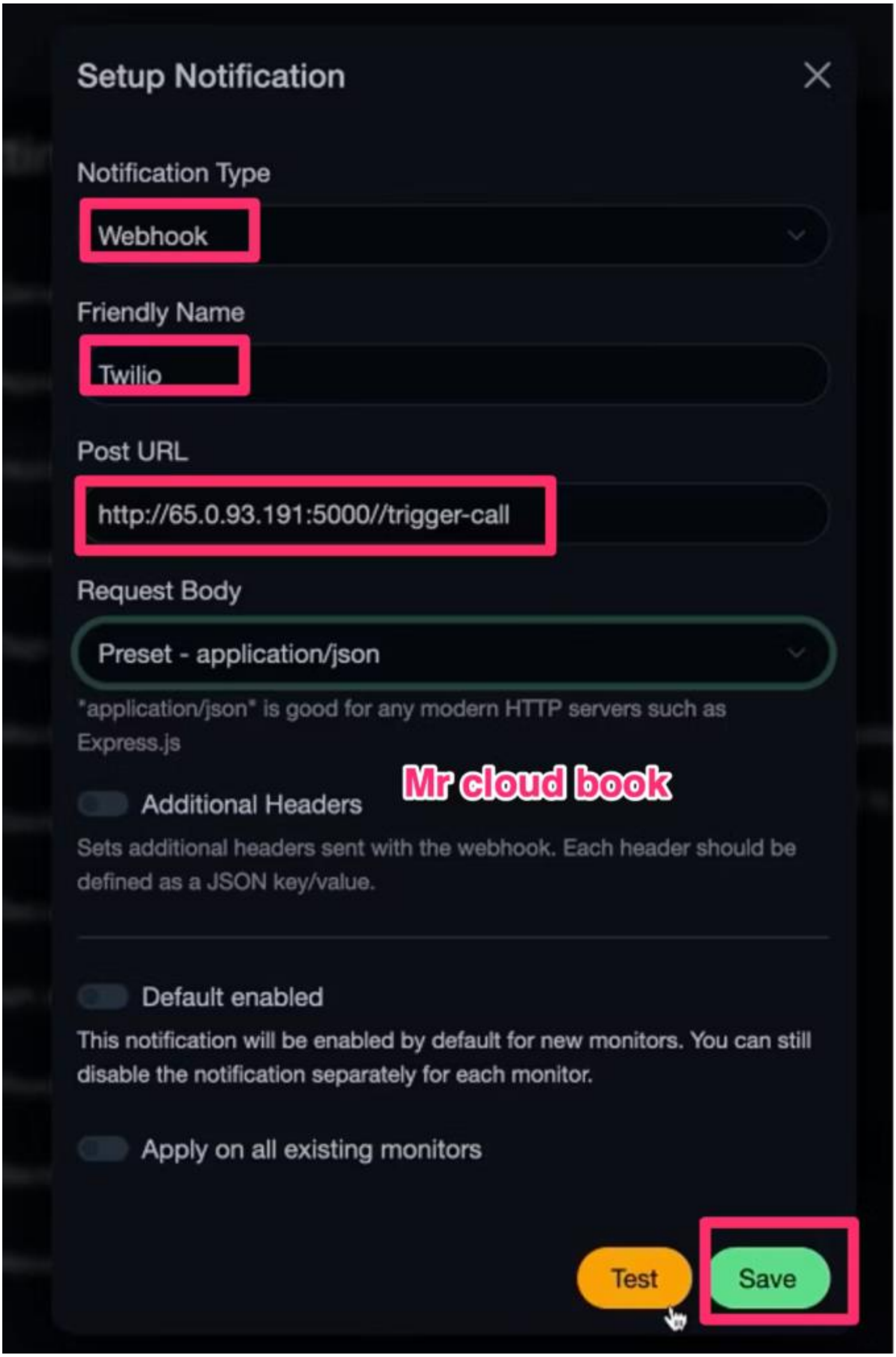
Not available, please setup.

HTTPS Monitors trigger notification when TLS certificate expires in:

Notifications must be assigned to a monitor to function.

Select Notification Type as webhook

Give Friendly name –> URL of webhook –> save



Setup Notification

Notification Type

Webhook

Friendly Name

Twilio

Post URL

http://65.0.93.191:5000//trigger-call

Request Body

Preset - application/json

"application/json" is good for any modern HTTP servers such as Express.js

Additional Headers

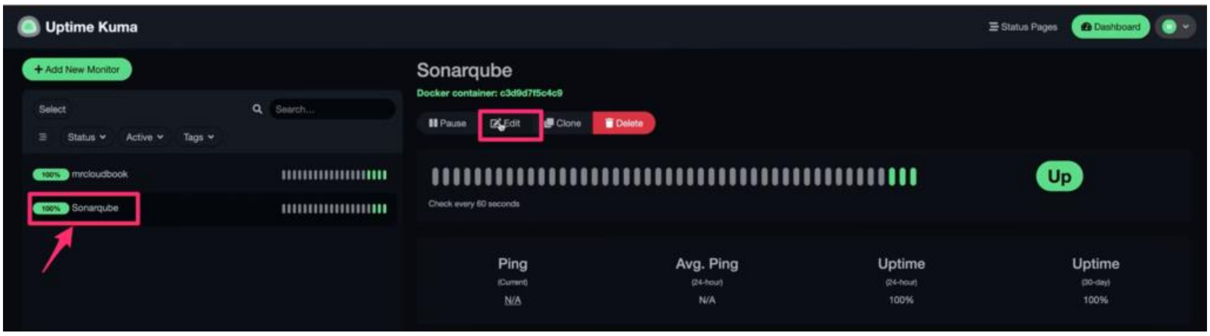
Sets additional headers sent with the webhook. Each header should be defined as a JSON key/value.

Default enabled

This notification will be enabled by default for new monitors. You can still disable the notification separately for each monitor.

Apply on all existing monitors

Select One service from Dashboard –> Edit



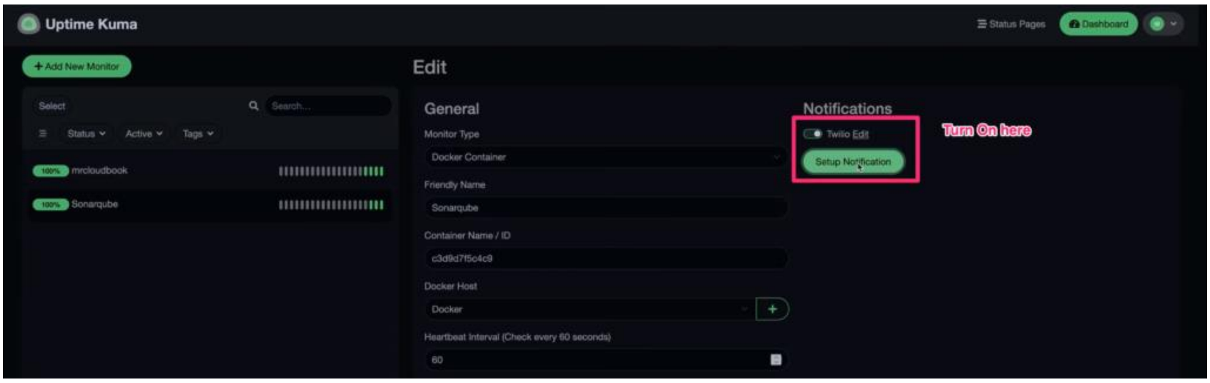
Sonarqube

Docker container: c3d9d7f5c4c9

Check every 60 seconds

Ping (Current)	Avg. Ping (24-hour)	Uptime (24-hour)	Uptime (30-day)
N/A	N/A	100%	100%

Turn on Notification to get call



Edit

General

Monitor Type: Docker Container

Friendly Name: Sonarqube

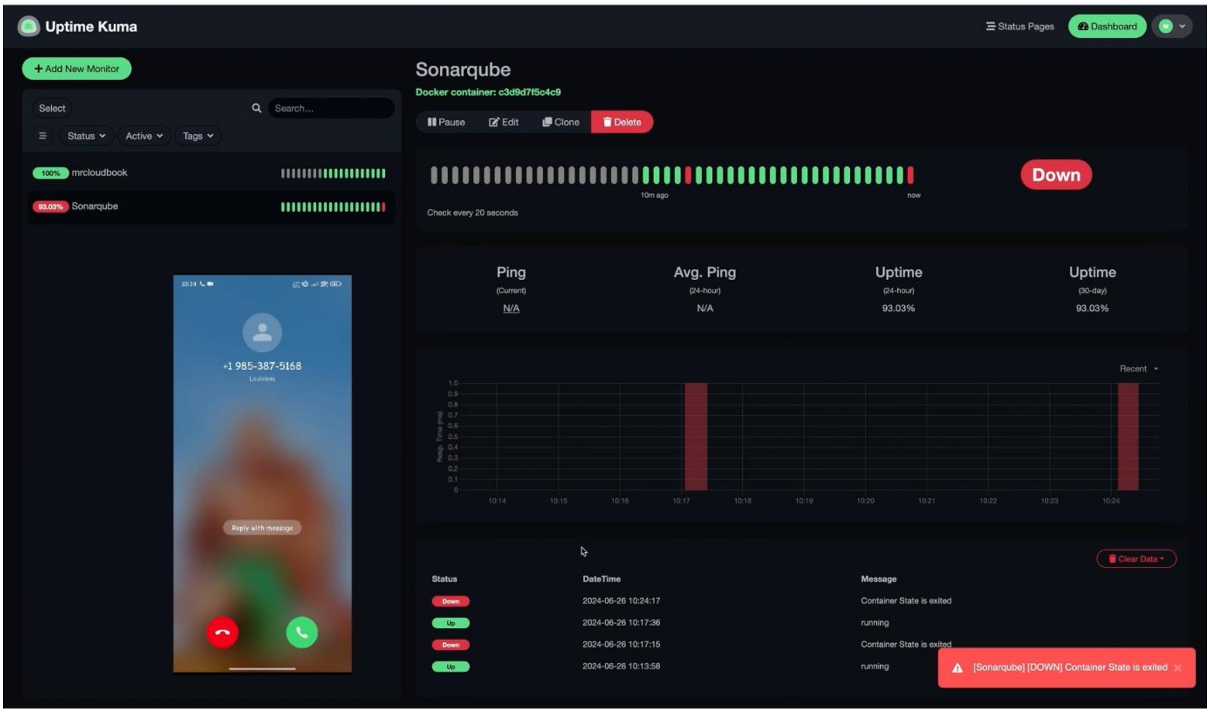
Container Name / ID: c3d9d7f5c4c9

Docker Host: Docker

Heartbeat Interval (Check every 60 seconds): 60

Notifications

You will get call when server or service Down



Sonarqube

Docker container: c3d9d7f5c4c9

Check every 20 seconds

Ping (Current)	Avg. Ping (24-hour)	Uptime (24-hour)	Uptime (30-day)
N/A	N/A	93.03%	93.03%

Status	DateTime	Message
Down	2024-06-26 10:24:17	Container State is exited
Up	2024-06-26 10:17:36	running
Down	2024-06-26 10:17:15	Container State is exited
Up	2024-06-26 10:13:58	running

[Sonarqube] [DOWN] Container State is exited

Remove Server at end.