EXPERIMENT NO. 07

Aim: **To understand Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.**

Steps:

1) **Install and configure a Jenkins and SonarQube CICD environment using Docker containers.**
2) **Configure Jenkins with the SonarQube Scanner plugin for automated static code analysis.**

1) Install and configure a Jenkins and SonarQube CICD environment using Docker containers.

Installation of Jenkins

The version of Jenkins included with the default Ubuntu packages is often behind the latest available version from the project itself. To take advantage of the latest fixes and features, you can use the project-maintained packages to install Jenkins.

```
manjusha@apsit:~$ wget -q -O -
https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -
```

When the key is added, the system will return OK. Next, append the Debian package repository address to the server’s sources.list:

```
manjusha@apsit:~$ sudo sh -c 'echo deb http://pkg.jenkins.io/debian-stable
binary/ > /etc/apt/sources.list.d/jenkins.list'
```

When both of these are in place, run `update` so that `apt` will use the new repository:

```
manjusha@apsit:~$ sudo apt update
```

Finally, install Jenkins and its dependencies:

```
manjusha@apsit:~$sudo apt install jenkins
```

Let’s start Jenkins using systemctl:

```
manjusha@apsit:~$sudo systemctl start jenkins
```

Since systemctl doesn’t display output, you can use its status command to verify that Jenkins started successfully:

```
manjusha@apsit:~$sudo systemctl status jenkins
```

If everything went well, the beginning of the output should show that the service is active and configured to start at boot:

Now that Jenkins is running, let’s adjust our firewall rules so that we can reach it from a web browser to complete the initial setup.

Opening the Firewall

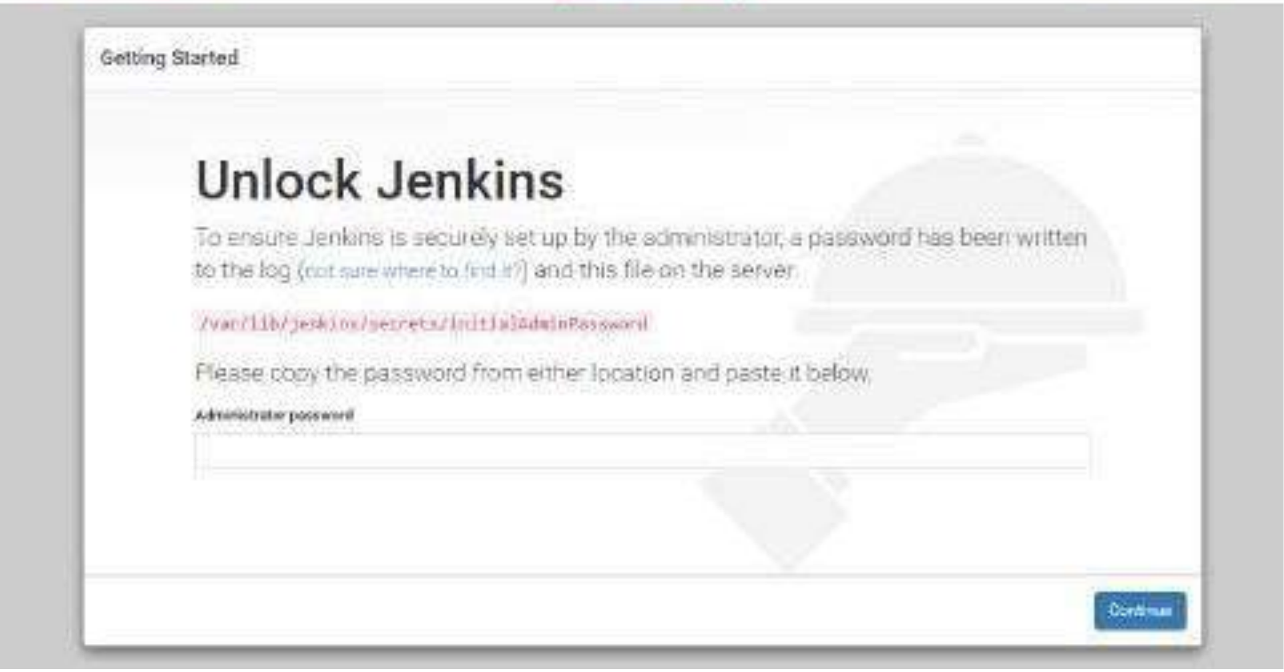
By default, Jenkins runs on port 8080, so let’s open that port using ufw:

```
manjusha@apsit:~$sudo ufw allow 8080
```

Setting Up Jenkins

To set up your installation, visit Jenkins on its default port, 8080, using your server domain name or IP address: **http://your_server_ip_or_domain:8080**

You should see the Unlock Jenkins screen, which displays the location of the initial 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.

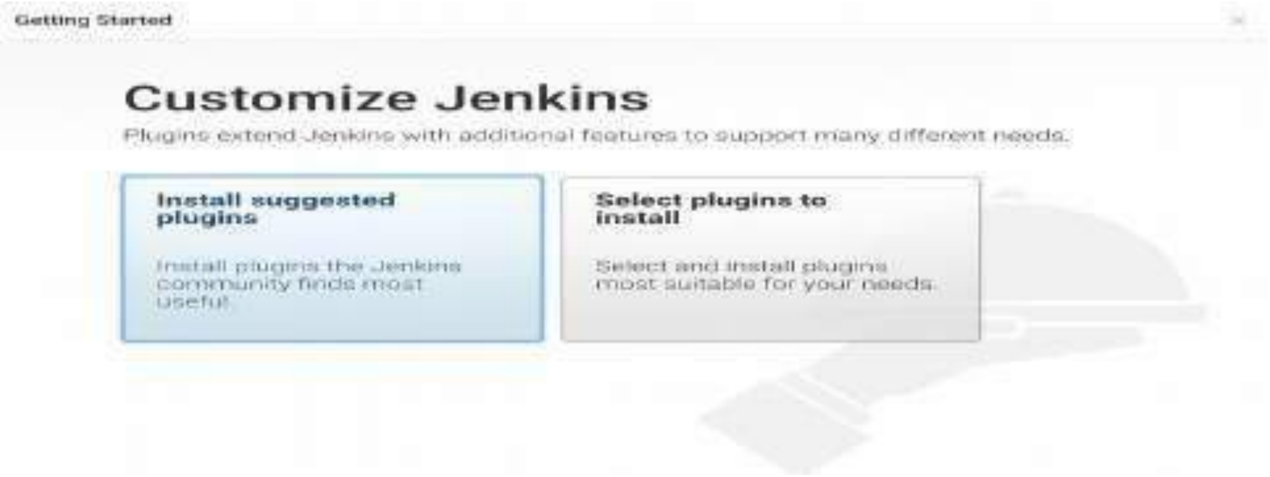
Administrator password

In the terminal window, use the cat command to display the password:

manjusha@apsit:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword

Copy the 32-character alphanumeric password from the terminal and paste it into the Administrator password field, then click Continue.

The next screen presents the option of installing suggested plugins or selecting specific plugins:



Getting Started

Customize Jenkins

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

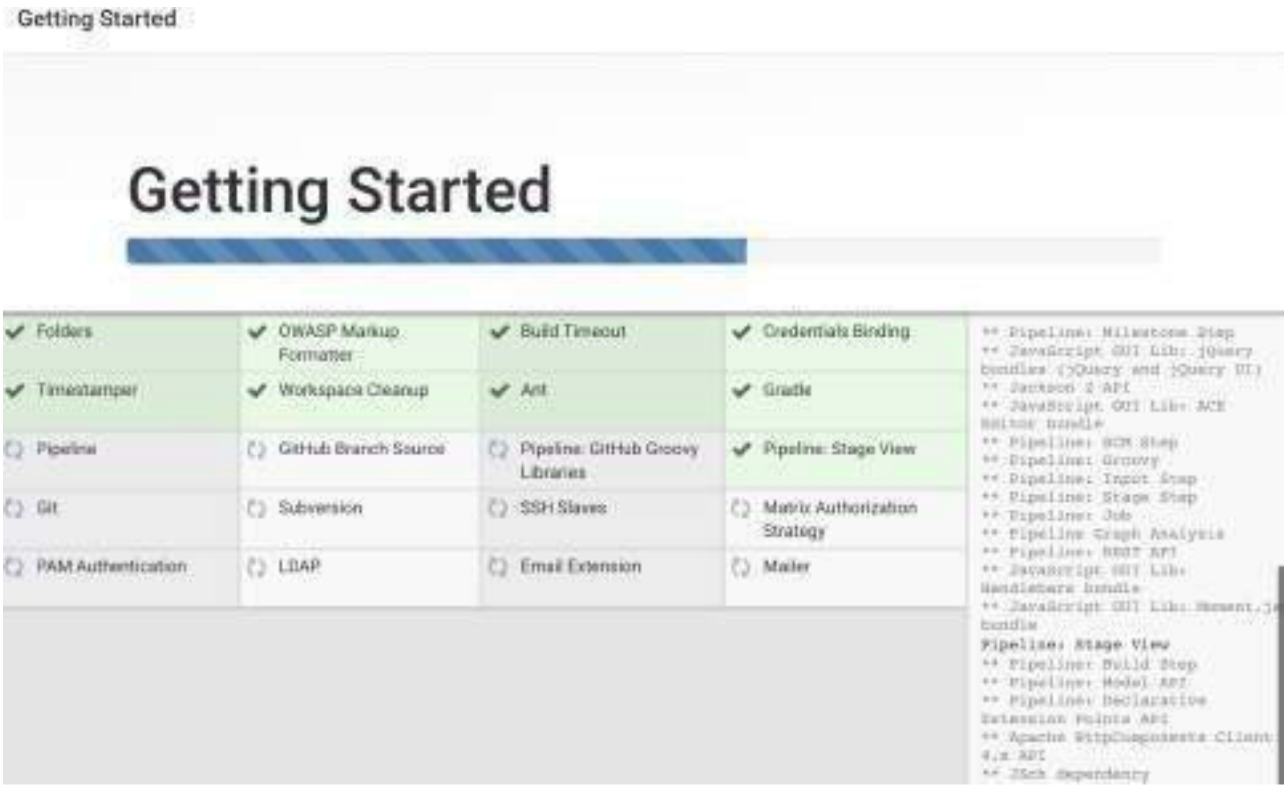
Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

We'll click the Install suggested plugins option, which will immediately begin the installation process:



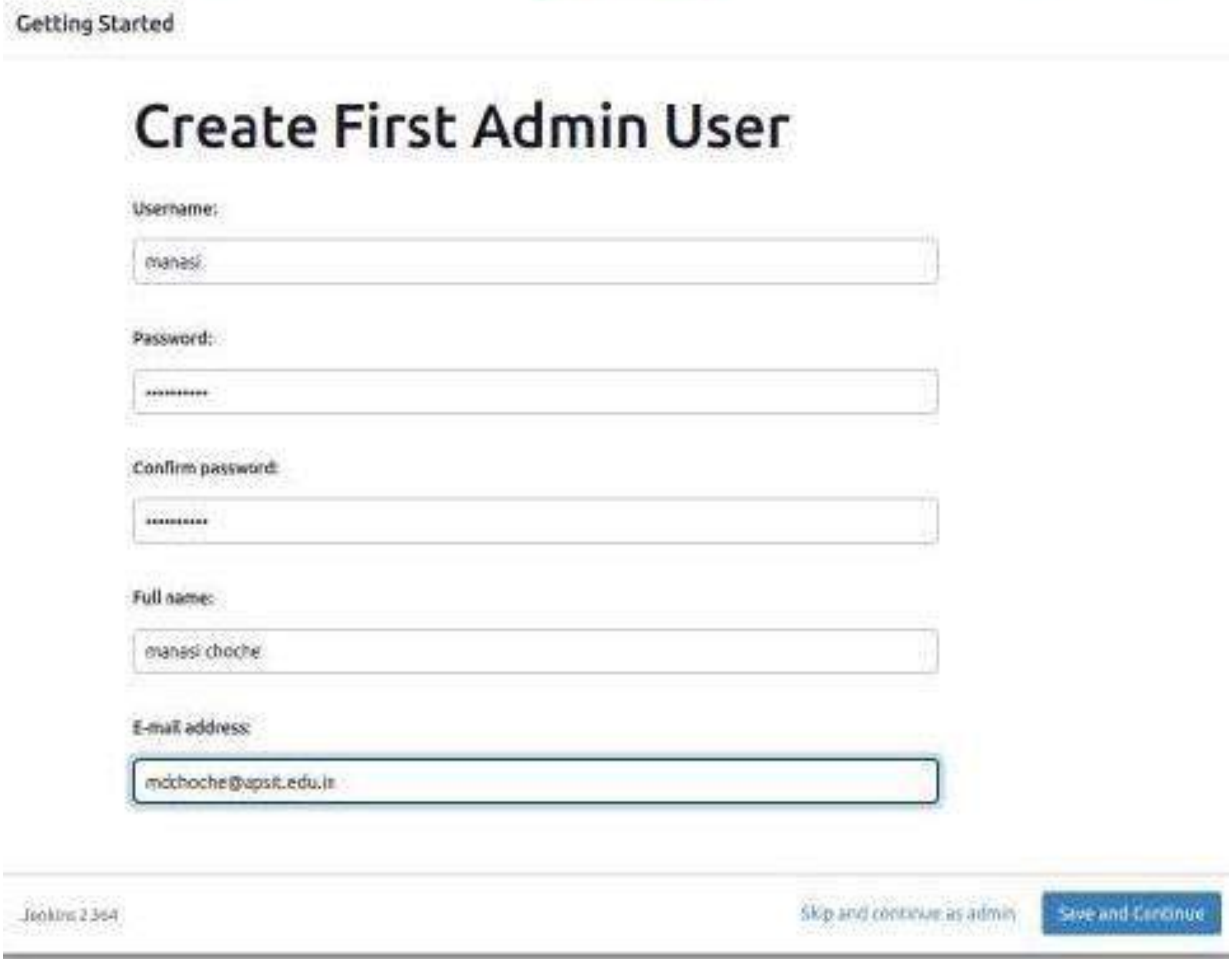
Getting Started

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	○ Subversion	○ SSH Slaves	○ Matrix Authorization Strategy
○ PAM Authentication	○ LDAP	○ Email Extension	○ Mailer

```
** Pipeline: Milestone Step
** JavaScript GUI Lib: jQuery
bundles (jQuery and jQuery UI)
** Jackson 2 API
** JavaScript GUI Lib: ACE
Editor bundle
** Pipeline: SCM Step
** Pipeline: Groovy
** Pipeline: Input Step
** Pipeline: Stage Step
** Pipeline: Job
** Pipeline Graph Analysis
** Pipeline: REST API
** JavaScript GUI Lib:
Handlebars bundle
** JavaScript GUI Lib: Moment.js
bundle
Pipeline: Stage View
** Pipeline: Build Step
** Pipeline: Model API
** Pipeline: Declarative
Extension Points API
** Apache HttpComponents Client
4.x API
** JSch dependency
```

When the installation is complete, you will be prompted to set up the first administrative user. It's possible to skip this step and continue as admin using the initial password we used above, but we'll take a moment to create the user.



Getting Started

Create First Admin User

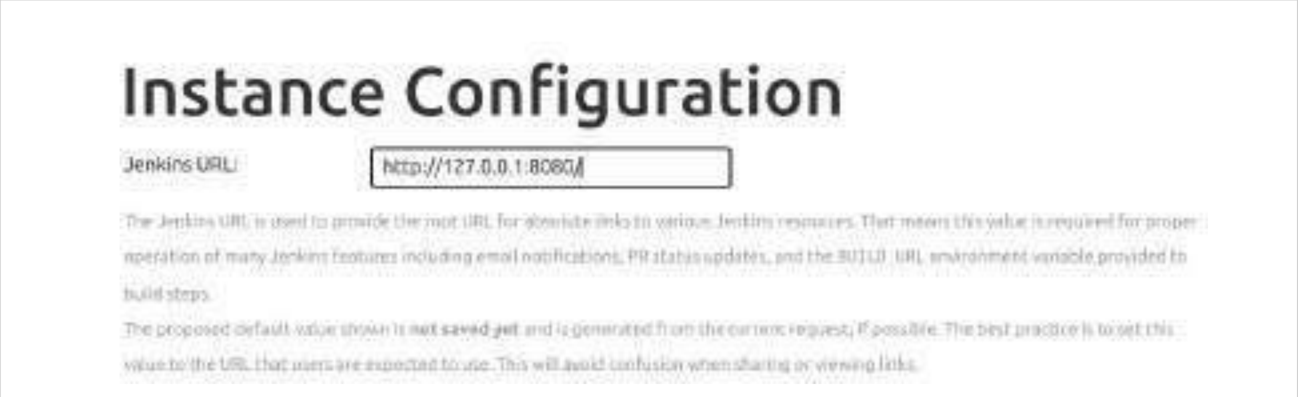
Username:
manasi

Password:
••••••••••

Confirm password:
••••••••••

Full name:
manasi choche

E-mail address:
mdchoche@apsit.edu.in



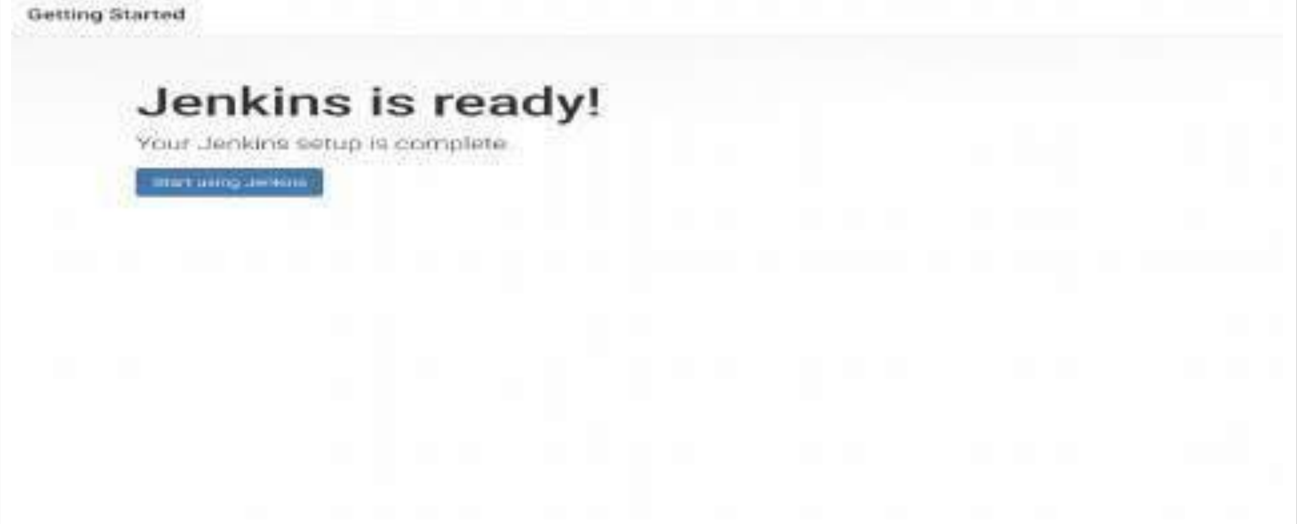
Instance Configuration

Jenkins URL: http://127.0.0.1:8080/

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

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

After confirming the appropriate information, click Save and Finish. You will see a confirmation page confirming that “Jenkins is Ready!”:

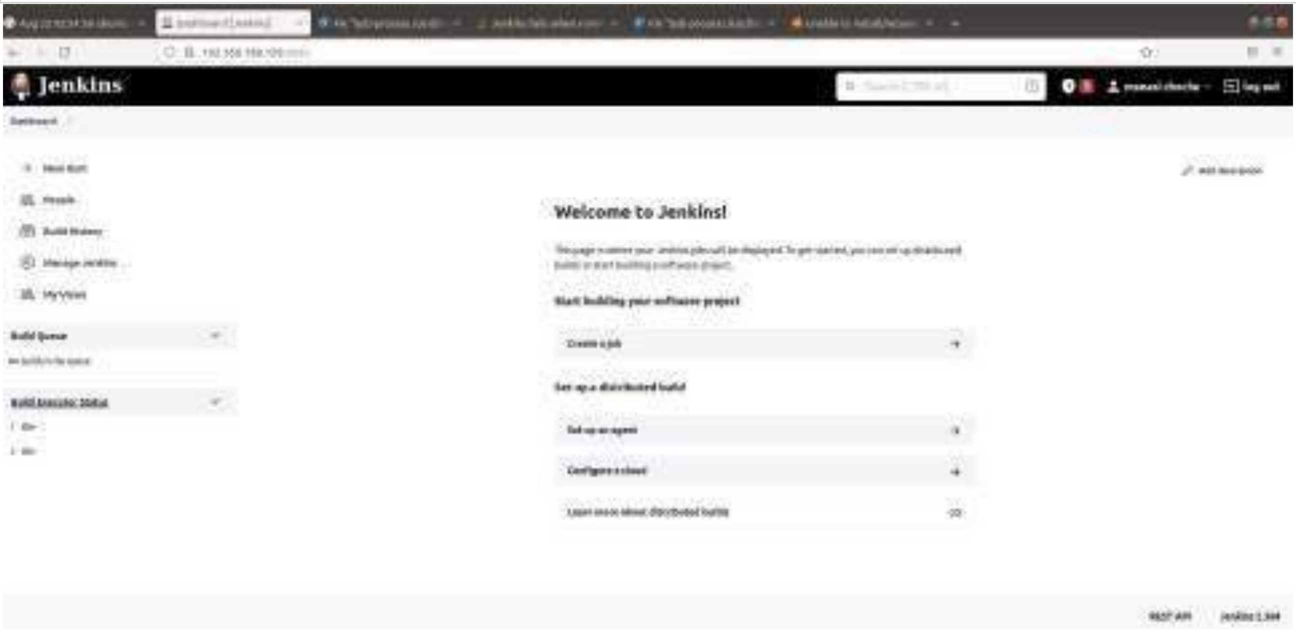


Getting Started

Jenkins is ready!

Your Jenkins setup is complete.

Click Start using Jenkins to visit the main Jenkins dashboard:



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

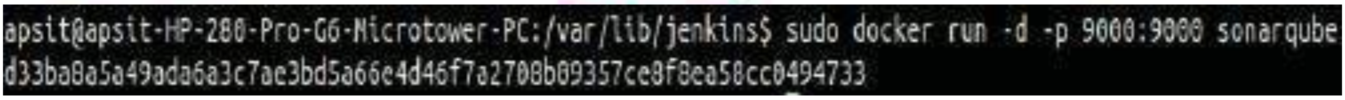
Configure a cloud

Learn more about distributed builds

SonarQube Setup

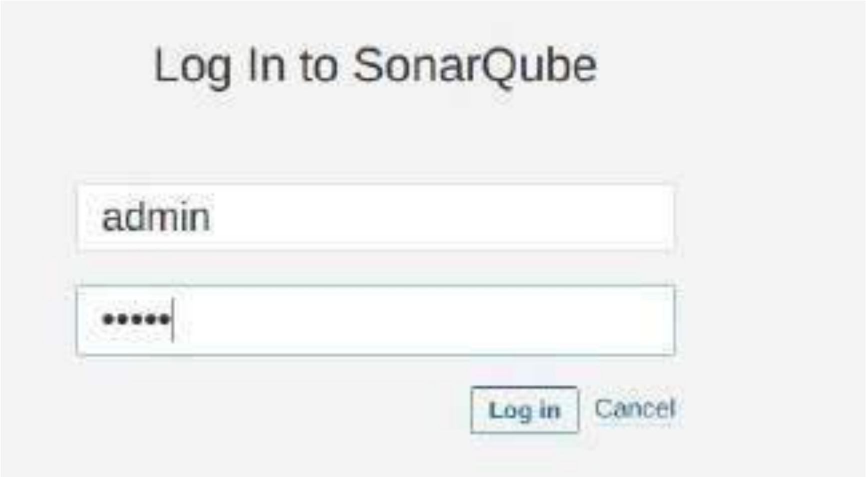
Before proceeding with the integration, we will setup SonarQube Instance. we are using SonarQube Docker Container.

```
manjusha@apsit:~$docker run -d -p 9000:9000 sonarqube
```



```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:/var/lib/jenkins$ sudo docker run -d -p 9000:9000 sonarqube
d33ba8a5a49ada6a3c7ae3bd5a66e4d46f7a2708b09357ce8f8ea58cc0494733
```

In the above command, we are forwarding port 9000 of the container to the port 9000 of the host machine as SonarQube is will run on port 9000. Then, from the browser, enter http://localhost:9000. After That, you will see the SonarQube is running. Then, login using default credentials (admin:admin).



Log In to SonarQube

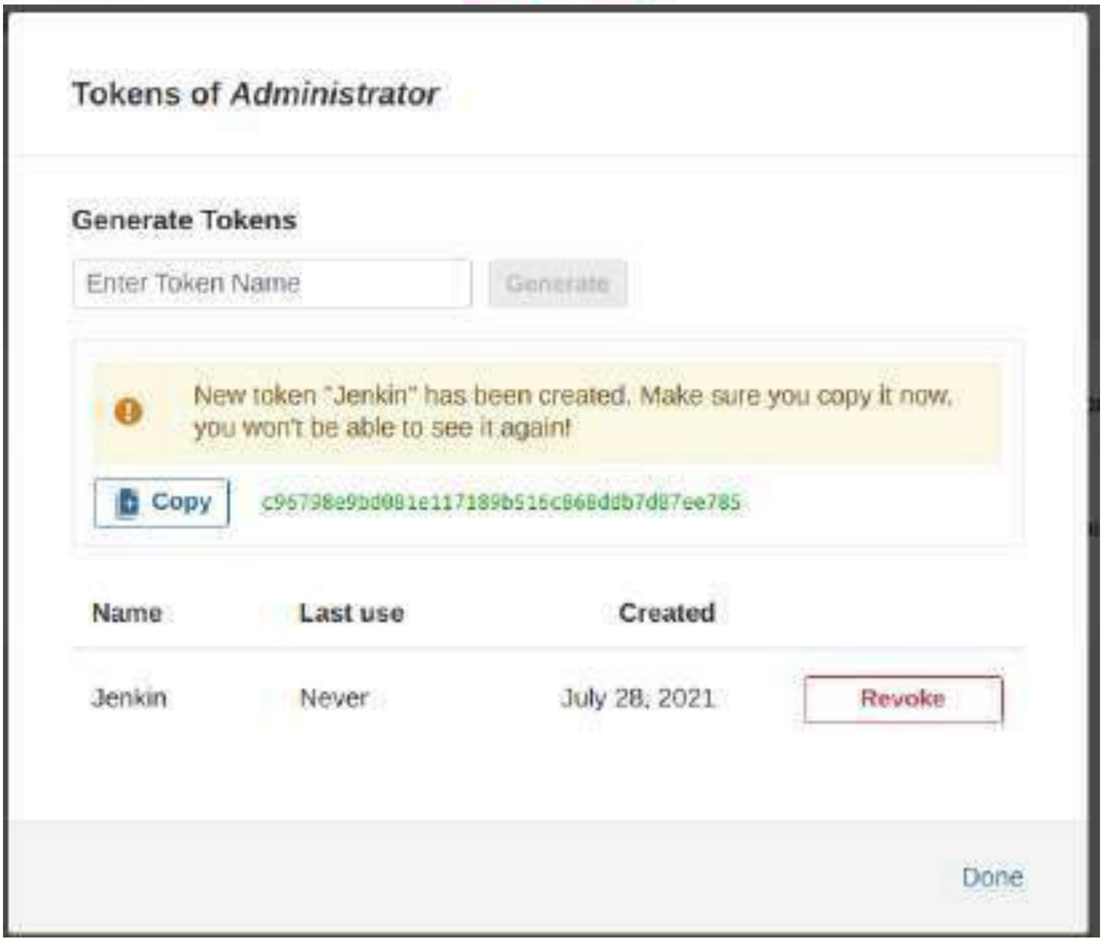
admin

•••••

Generate User Token

Now, we need to get the SonarQube user token to make connection between Jenkins and SonarQube. For the same, go to **Administration> User > My Account > Security** and then, from the bottom of the page you can create new tokens by clicking the Generate Button. Copy the Token and keep it safe.

C96798e9bd081e117189b516c868ddb7d87ee785 SonarQube



Tokens of *Administrator*

Generate Tokens

Enter Token Name

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

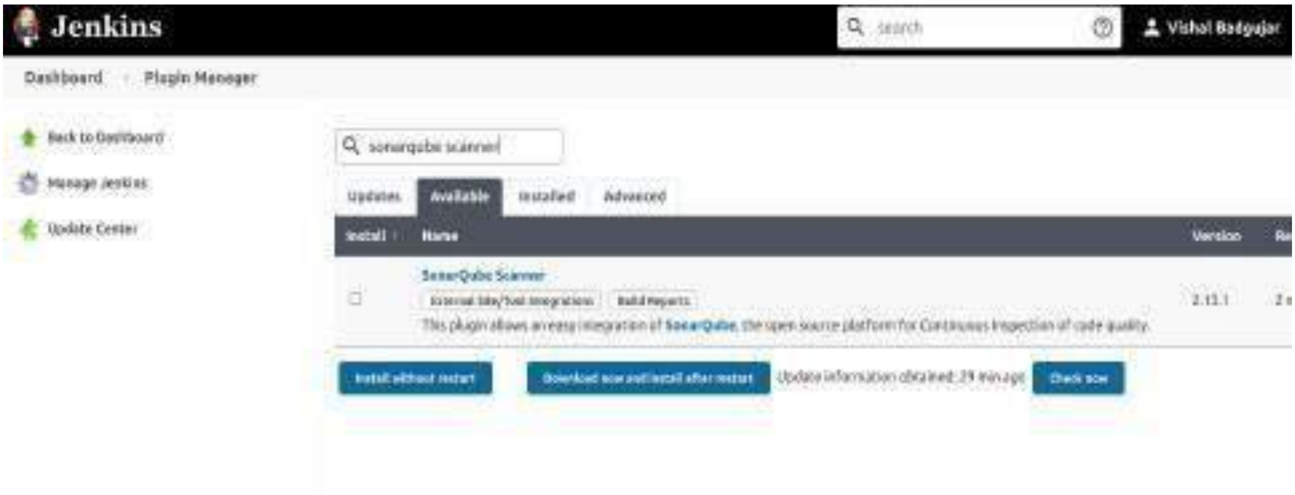
```
c96798e9bd081e117189b516c868ddb7d87ee785
```

Name	Last use	Created	
Jenkin	Never	July 28, 2021	Revoke

2) Configure Jenkins with the SonarQube Scanner plugin for automated static code analysis.

Jenkins Setup for SonarQube

Before all, we need to install the SonarQube Scanner plugin in Jenkins. For the same, go to **Manage Jenkins > Plugin Manager > Available.** From here, type SonarQube Scanner then select and install.



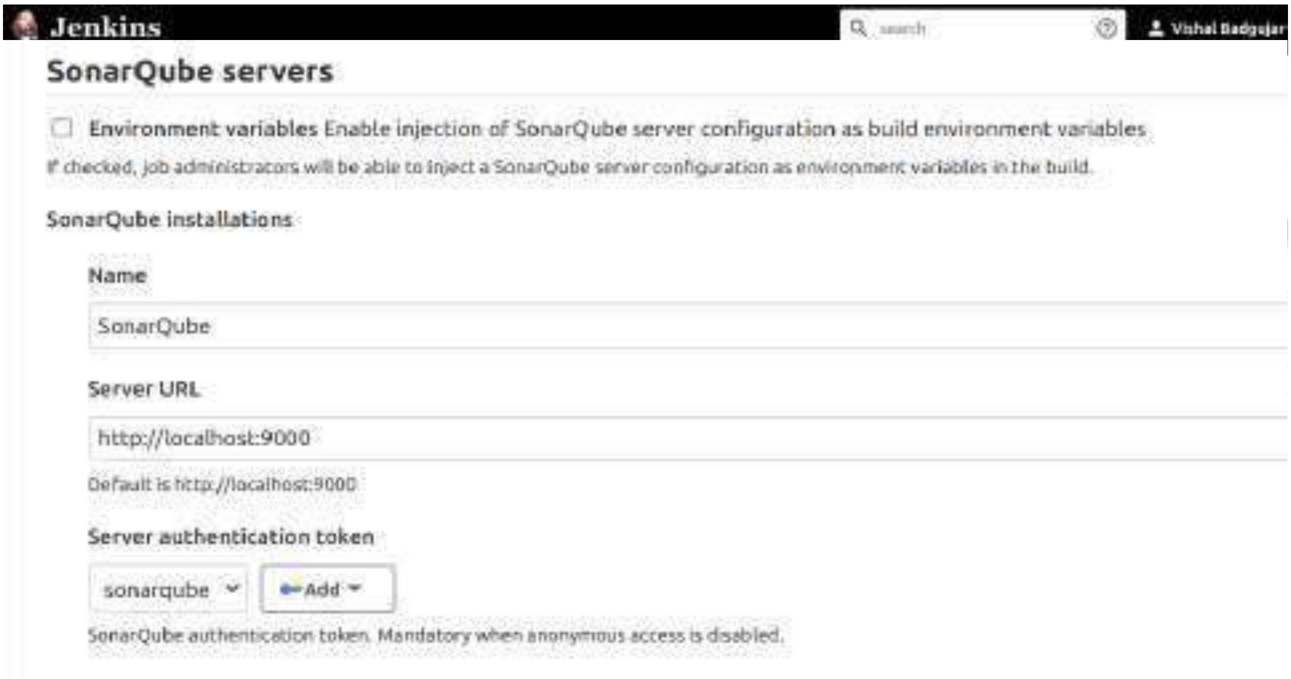
sonarqube scanner

Install	Name	Version
□	SonarQube Scanner External Site/Tool Integrations Build Reports This plugin allows an easy integration of SonarQube, the open source platform for Continuous Inspection of code quality.	2.13.1

Update information obtained: 29 min ago

Tool Configuration SonarQube Scanner

Now, we need to configure the Jenkins plugin for SonarQube Scanner to make a connection with the SonarQube Instance. For that, got to **Manage Jenkins > Configure System > SonarQube Server.** Then, Add SonarQube. In this, give the Installation Name, Server URL then Add the Authentication token in the Jenkins Credential Manager and select the same in the configuration.



SonarQube servers

□ Environment variables Enable injection of SonarQube server configuration as build environment variables

If checked, job administrators will be able to inject a SonarQube server configuration as environment variables in the build.

SonarQube installations

Name: SonarQube

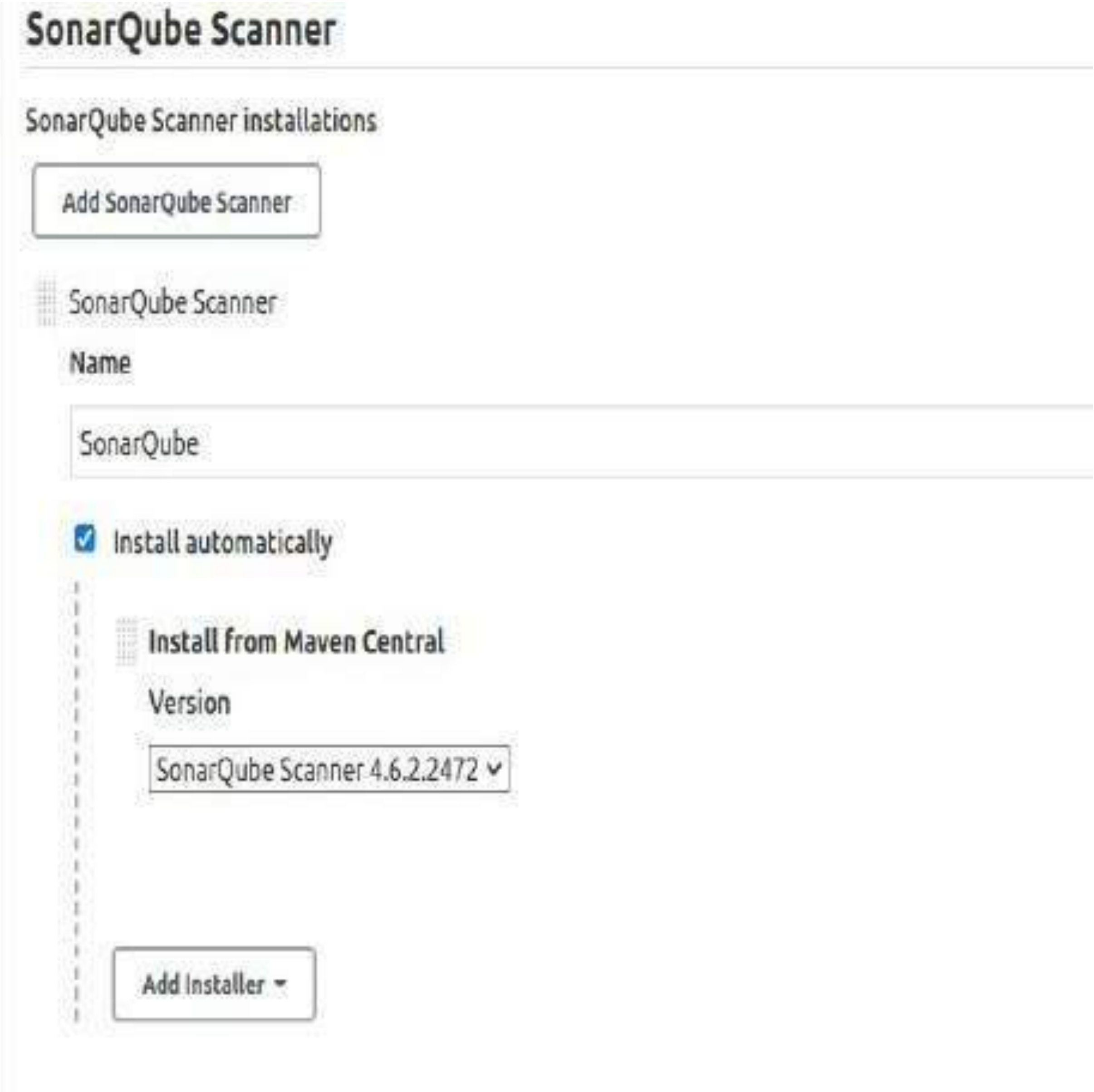
Server URL: http://localhost:9000

Default is http://localhost:9000

Server authentication token: sonarqube

SonarQube authentication token. Mandatory when anonymous access is disabled.

Then, we need to set-up the SonarQube Scanner to scan the source code in the various stage. For the same, go to **Manage Jenkins > Global Tool Configuration > SonarQube Scanner**. Then, Click **Add SonarQube Scanner Button**. From there, give some name of the scanner type and **Add Installer** of your choice. In this case, I have selected SonarQube Scanner from Maven Central.



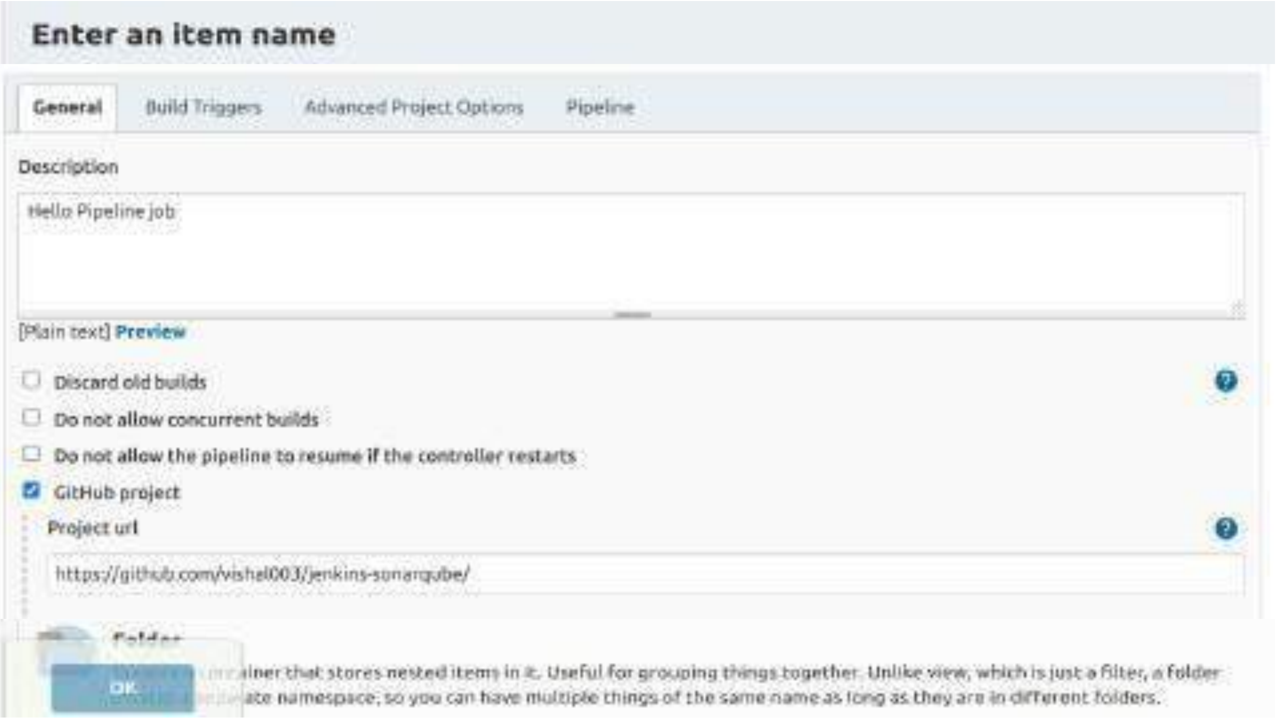
Name: SonarQube

☑ Install automatically

Version: SonarQube Scanner 4.6.2.2472

SonarQube Scanner in Jenkins Pipeline

Now, It's time to integrate the SonarQube Scanner in the Jenkins Pipeline. For the same, we are going to add one more stage in the Jenkinsfile called SonarQube and inside that, I am adding the following settings and code.



Enter an item name

Description

Hello Pipeline job

[Plain text] Preview

□ Discard old builds

□ Do not allow concurrent builds

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

☑ GitHub project

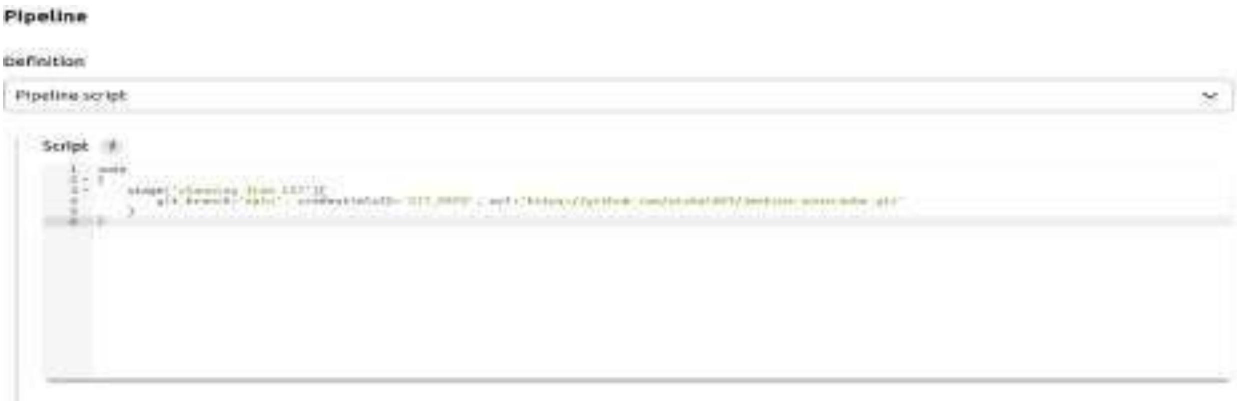
Project url

https://github.com/vishal003/jenkins-sonarqube/

Folder

...ainer that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder ...ate namespace, so you can have multiple things of the same name as long as they are in different folders.

Github Configuration in Jenkins Pipeline



Pipeline

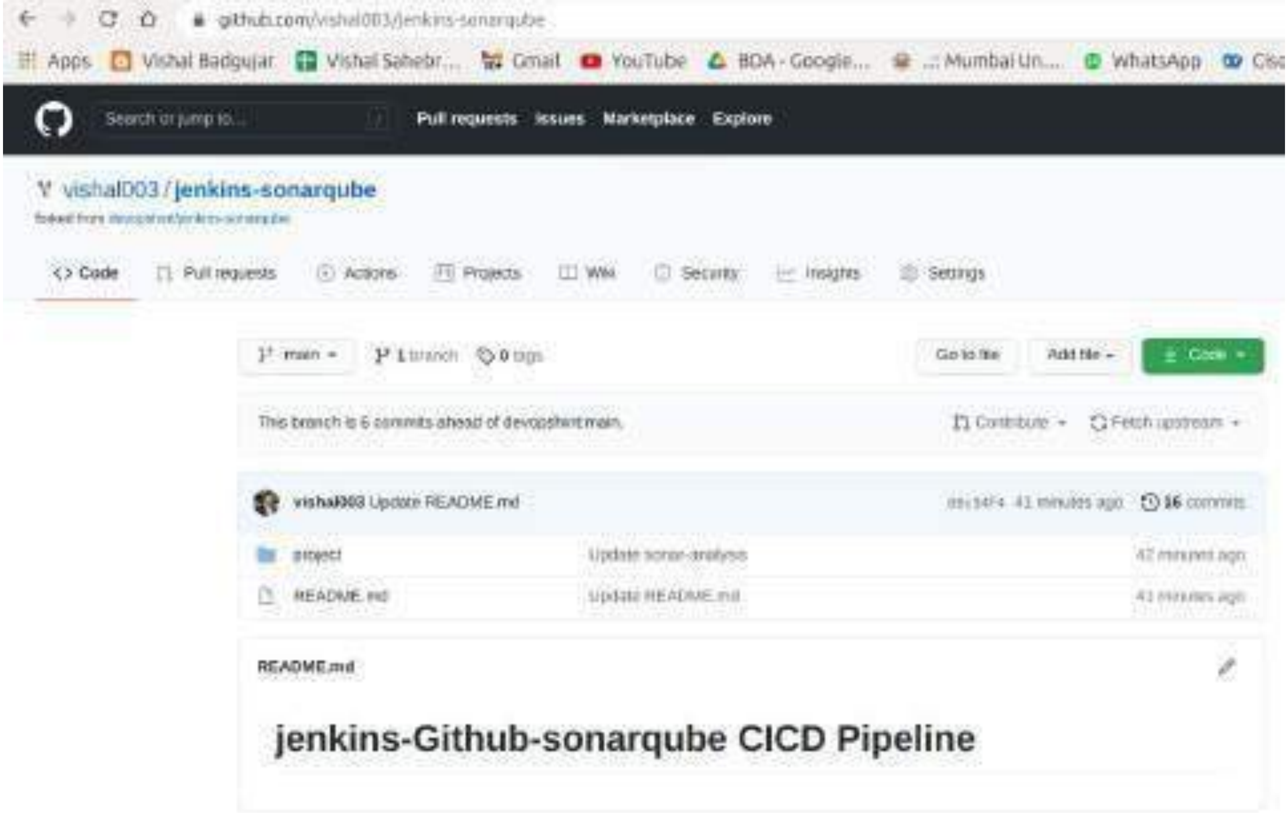
Definition

Pipeline script

Script

```
node
{
    stage('Cloning from GIT'){
        git branch:'main', credentialsId:'GIT_PASS', url:'https://github.com/vishal003/jenkins-sonarqube.git'
    }
}
```

Git Clonning into Jenkins



This branch is 6 commits ahead of devopshint:main.

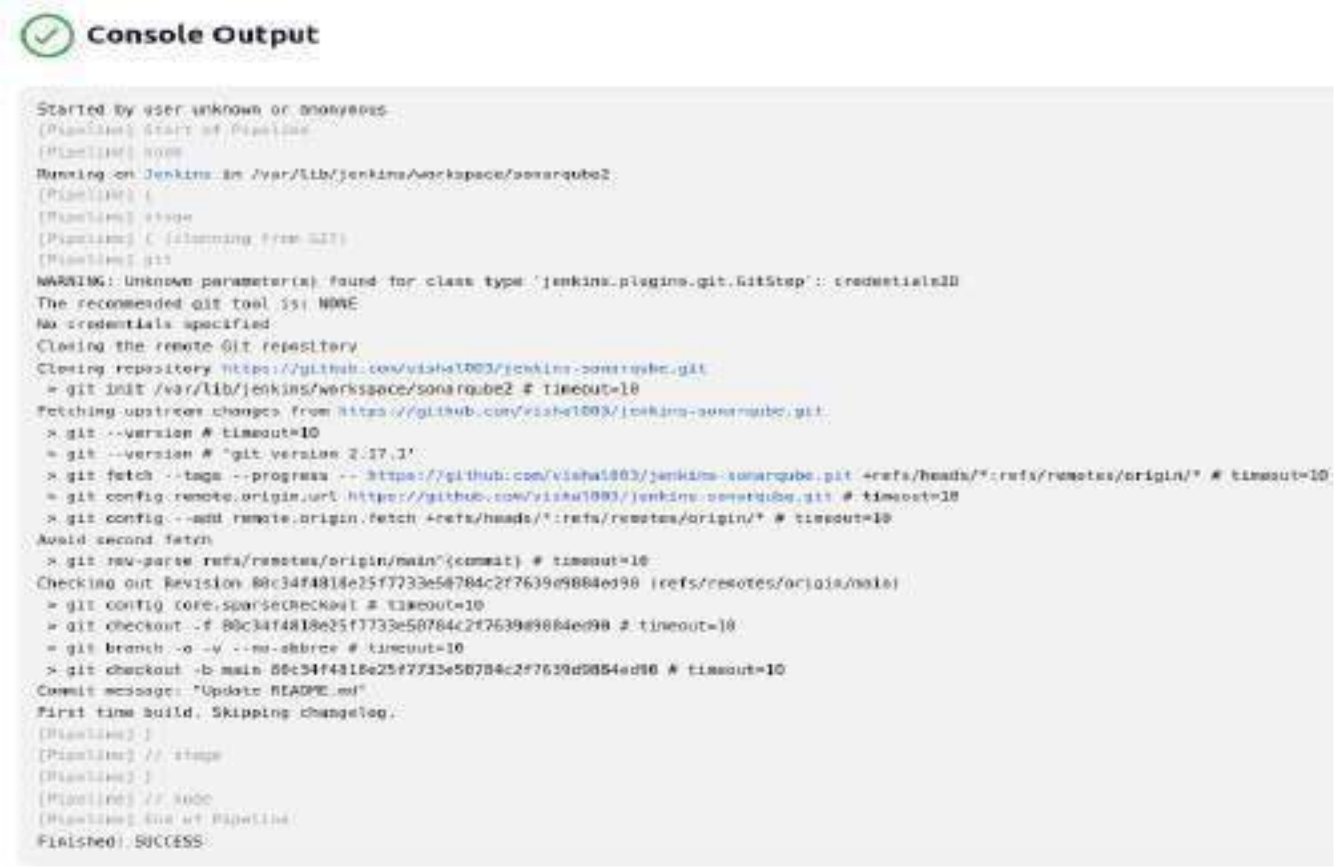
vishal003 Update README.md … 41 minutes ago 16 commits

project	Update sonar-analysis	42 minutes ago
README.md	Update README.md	41 minutes ago

README.md

jenkins-Github-sonarqube CICD Pipeline

Github Repository Contents



Console Output

```
Started by user unknown or anonymous
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/lib/jenkins/workspace/sonarqube2
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Cloning from GIT)
[Pipeline] git
WARNING: Unknown parameter(s) found for class type 'jenkins.plugins.git.GitStep': credentialsID
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/vishal003/jenkins-sonarqube.git
 > git init /var/lib/jenkins/workspace/sonarqube2 # timeout=10
Fetching upstream changes from https://github.com/vishal003/jenkins-sonarqube.git
 > git --version # timeout=10
 > git --version # 'git version 2.17.1'
 > git fetch --tags --progress -- https://github.com/vishal003/jenkins-sonarqube.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git config remote.origin.url https://github.com/vishal003/jenkins-sonarqube.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 80c34f4818e25f7733e50784c2f7639d9884ed98 (refs/remotes/origin/main)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f 80c34f4818e25f7733e50784c2f7639d9884ed98 # timeout=10
 > git branch -a -v --no-abbrev # timeout=10
 > git checkout -b main 80c34f4818e25f7733e50784c2f7639d9884ed98 # timeout=10
Commit message: "Update README.md"
First time build. Skipping changelog.
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Successfully Build Github Repository in Jenkins

Pre-requiste required for Integration settings of Jenkins SAST with SonarQube we have done here successfully, now in order to Integrate of Jenkins CICD with SonarQube with the help of sample JAVA program we will implement in next experiment.

Conclusion: Thus, we understood Static Analysis SAST process and learn to integrate Jenkins SAST to SonarQube/GitLab.

EXPERIMENT NO. 09

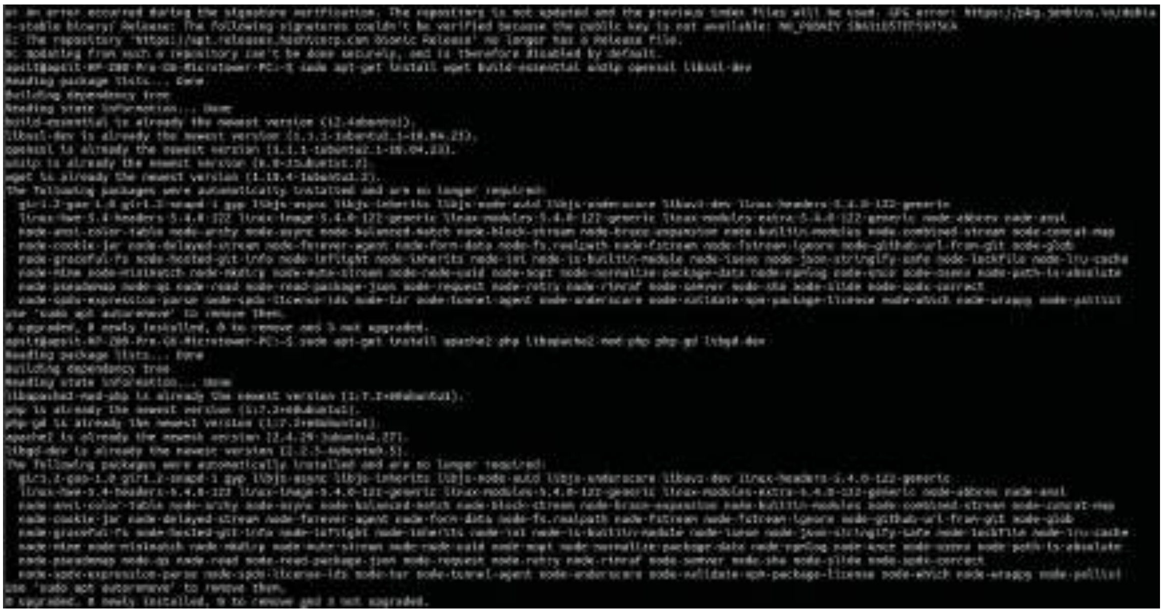
Aim: To Understand Continuous monitoring and Installation and configuration of Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin Executor) on Linux Machine.

Theory:

1 - Pre-requisite

First requirement is to install Apache and PHP first. Use the following commands to complete it. And use commands to install required packages for Nagios.

```
manjusha@apsit:~$ sudo apt-get update
manjusha@apsit:~$ sudo apt-get install wget build-essential unzip openssl
libssl-dev
manjusha@apsit:~$ sudo apt-get install apache2 php libapache2-mod-php php-gd
libgd-dev
```



```
W: An error occurred during the signature verification. The repository is not updated and the previous index files will be used. GPG error: https://pkg.jenkins.io/debian-stable binary/ Release: The following signatures couldn't be verified because the public key is not available: NO_PUBKEY [illegible]
E: The repository 'https://apt.releases.hashicorp.com bionic Release' no longer has a Release file.
N: Updating from such a repository can't be done securely, and is therefore disabled by default.
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo apt-get install wget build-essential unzip openssl libssl-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
build-essential is already the newest version (12.4ubuntu1).
libssl-dev is already the newest version (1.1.1-1ubuntu2.1~18.04.21).
openssl is already the newest version (1.1.1-1ubuntu2.1~18.04.21).
unzip is already the newest version (6.0-21ubuntu1.2).
wget is already the newest version (1.19.4-1ubuntu2.2).
The following packages were automatically installed and are no longer required:
  gir1.2-goa-1.0 gir1.2-snapd-1 gyp libjs-async libjs-inherits libjs-node-uuid libjs-underscore libuv1-dev linux-headers-5.4.0-122-generic
  linux-hwe-5.4-headers-5.4.0-122 linux-image-5.4.0-122-generic linux-modules-5.4.0-122-generic linux-modules-extra-5.4.0-122-generic node-abbrev node-ansi
  node-ansi-color-table node-archy node-async node-balanced-match node-block-stream node-brace-expansion node-builtin-modules node-combined-stream node-concat-map
  node-cookie-jar node-delayed-stream node-forever-agent node-form-data node-fs.realpath node-fstream node-fstream-ignore node-github-url-from-git node-glob
  node-graceful-fs node-hosted-git-info node-inflight node-inherits node-ini node-is-builtin-module node-isexe node-json-stringify-safe node-lockfile node-lru-cache
  node-mime node-minimatch node-mkdirp node-mute-stream node-node-uuid node-nopt node-normalize-package-data node-npmlog node-once node-osenv node-path-is-absolute
  node-pseudomap node-qs node-read node-read-package-json node-request node-retry node-rimraf node-semver node-sha node-slide node-spdx-correct
  node-spdx-expression-parse node-spdx-license-ids node-tar node-tunnel-agent node-underscore node-validate-npm-package-license node-which node-wrappy node-yallist
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
apsit@apsit-HP-280-Pro-G6-Microtower-PC:~$ sudo apt-get install apache2 php libapache2-mod-php php-gd libgd-dev
Reading package lists... Done
Building dependency tree
Reading state information... Done
libapache2-mod-php is already the newest version (1:7.2+60ubuntu1).
php is already the newest version (1:7.2+60ubuntu1).
php-gd is already the newest version (1:7.2+60ubuntu1).
apache2 is already the newest version (2.4.29-1ubuntu4.27).
libgd-dev is already the newest version (2.2.5-4ubuntu0.5).
The following packages were automatically installed and are no longer required:
  gir1.2-goa-1.0 gir1.2-snapd-1 gyp libjs-async libjs-inherits libjs-node-uuid libjs-underscore libuv1-dev linux-headers-5.4.0-122-generic
  linux-hwe-5.4-headers-5.4.0-122 linux-image-5.4.0-122-generic linux-modules-5.4.0-122-generic linux-modules-extra-5.4.0-122-generic node-abbrev node-ansi
  node-ansi-color-table node-archy node-async node-balanced-match node-block-stream node-brace-expansion node-builtin-modules node-combined-stream node-concat-map
  node-cookie-jar node-delayed-stream node-forever-agent node-form-data node-fs.realpath node-fstream node-fstream-ignore node-github-url-from-git node-glob
  node-graceful-fs node-hosted-git-info node-inflight node-inherits node-ini node-is-builtin-module node-isexe node-json-stringify-safe node-lockfile node-lru-cache
  node-mime node-minimatch node-mkdirp node-mute-stream node-node-uuid node-nopt node-normalize-package-data node-npmlog node-once node-osenv node-path-is-absolute
  node-pseudomap node-qs node-read node-read-package-json node-request node-retry node-rimraf node-semver node-sha node-slide node-spdx-correct
  node-spdx-expression-parse node-spdx-license-ids node-tar node-tunnel-agent node-underscore node-validate-npm-package-license node-which node-wrappy node-yallist
Use 'sudo apt autoremove' to remove them.
0 upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
```

2 – Create Nagios User

Create a new user account for Nagios in your system and assign a password.

```
manjusha@apsit:~$ sudo adduser nagios
```

Now create a group for Nagios setup “nagcmd” and add nagios user to this group. Also, add nagios user in the Apache group.

```
manjusha@apsit:~$sudo groupadd nagcmd
manjusha@apsit:~$sudo usermod -a -G nagcmd nagios
manjusha@apsit:~$sudo usermod -a -G nagcmd www-data
```

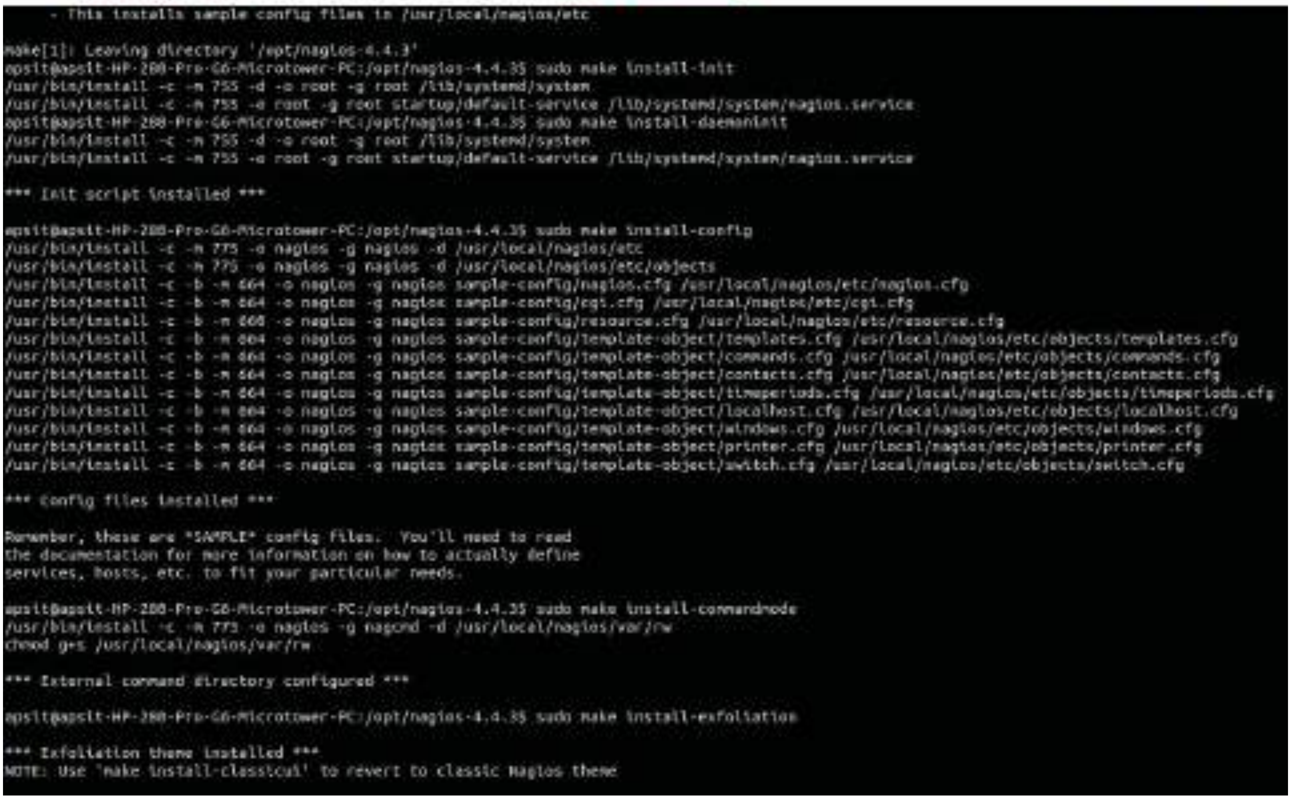
Step 3 – Install Nagios Core Service

After installing required dependencies and adding user accounts and Nagios core installation. Download latest Nagios core service from the official site.

```
manjusha@apsit:~$cd /opt/

manjusha@apsit:~$sudo wget
https://assets.nagios.com/downloads/nagioscore/releases/nagios-4.4.3.tar.gz
manjusha@apsit:~$sudo tar xzf nagios-4.4.3.tar.gz
```

After extracting naviate to nagios source directory and install using make command.

```
manjusha@apsit:~$cd  nagios-4.4.3

manjusha@apsit:~$sudo ./configure --with-command-group=nagcmd
manjusha@apsit:~$sudo make all
manjusha@apsit:~$sudo make install


manjusha@apsit:~$sudo make install-init
manjusha@apsit:~$sudo make install-daemoninit
manjusha@apsit:~$sudo make install-config
manjusha@apsit:~$sudo make install-commandmode
manjusha@apsit:~$sudo make install-exfoliation
```



```
    - This installs sample config files in /usr/local/nagios/etc

make[1]: Leaving directory '/opt/nagios-4.4.3'
apsit@apsit-HP-288-Pro-G6-Microtower-PC:/opt/nagios-4.4.3$ sudo make install-init
/usr/bin/install -c -m 755 -d -o root -g root /lib/systemd/system
/usr/bin/install -c -m 755 -o root -g root startup/default-service /lib/systemd/system/nagios.service
apsit@apsit-HP-288-Pro-G6-Microtower-PC:/opt/nagios-4.4.3$ sudo make install-daemoninit
/usr/bin/install -c -m 755 -d -o root -g root /lib/systemd/system
/usr/bin/install -c -m 755 -o root -g root startup/default-service /lib/systemd/system/nagios.service

*** Init script installed ***

apsit@apsit-HP-288-Pro-G6-Microtower-PC:/opt/nagios-4.4.3$ sudo make install-config
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc
/usr/bin/install -c -m 775 -o nagios -g nagios -d /usr/local/nagios/etc/objects
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/nagios.cfg /usr/local/nagios/etc/nagios.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/cgi.cfg /usr/local/nagios/etc/cgi.cfg
/usr/bin/install -c -b -m 660 -o nagios -g nagios sample-config/resource.cfg /usr/local/nagios/etc/resource.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/templates.cfg /usr/local/nagios/etc/objects/templates.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/commands.cfg /usr/local/nagios/etc/objects/commands.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/contacts.cfg /usr/local/nagios/etc/objects/contacts.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/timeperiods.cfg /usr/local/nagios/etc/objects/timeperiods.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/localhost.cfg /usr/local/nagios/etc/objects/localhost.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/windows.cfg /usr/local/nagios/etc/objects/windows.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/printer.cfg /usr/local/nagios/etc/objects/printer.cfg
/usr/bin/install -c -b -m 664 -o nagios -g nagios sample-config/template-object/switch.cfg /usr/local/nagios/etc/objects/switch.cfg

*** Config files installed ***

Remember, these are *SAMPLE* config files.  You'll need to read
the documentation for more information on how to actually define
services, hosts, etc. to fit your particular needs.

apsit@apsit-HP-288-Pro-G6-Microtower-PC:/opt/nagios-4.4.3$ sudo make install-commandmode
/usr/bin/install -c -m 775 -o nagios -g nagcmd -d /usr/local/nagios/var/rw
chmod g+s /usr/local/nagios/var/rw

*** External command directory configured ***

apsit@apsit-HP-288-Pro-G6-Microtower-PC:/opt/nagios-4.4.3$ sudo make install-exfoliation

*** Exfoliation theme installed ***
NOTE: Use 'make install-classicui' to revert to classic Nagios theme
```

Now copy event handlers scripts under libexec directory. These binaries provides multiple events triggers for your Nagios web interface.

```
manjusha@apsit:~$sudo cp -R contrib/eventhandlers/ /usr/local/nagios/libexec/
```

```
manjusha@apsit:~$sudo chown -R nagios:nagios
/usr/local/nagios/libexec/eventhandlers
```

Step 4 – Setup Apache with Authentication

Now create an Apache configuration file for your Nagios server as below:

```
manjusha@apsit:~$sudo  nano  /etc/apache2/conf-available/nagios.conf
```

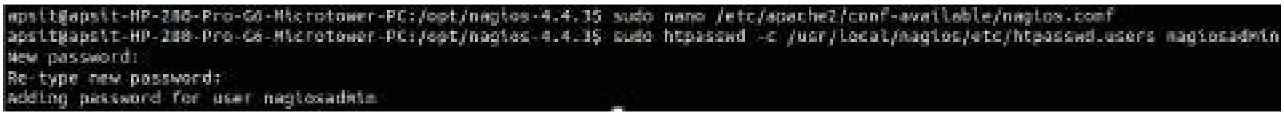
Add below lines to nagios.conf file.

ScriptAlias /nagios/cgi-bin "/usr/local/nagios/sbin"

```
<Directory "/usr/local/nagios/sbin">

   Options ExecCGI
   AllowOverride  None
   Order allow,deny
   Allow from all
   AuthName "Restricted Area"
   AuthType Basic
   AuthUserFile /usr/local/nagios/etc/htpasswd.users
   Require valid-user
</Directory>

Alias /nagios "/usr/local/nagios/share"

<Directory "/usr/local/nagios/share">
   Options None
   AllowOverride  None
   Order allow,deny
   Allow from all
   AuthName "Restricted Area"
   AuthType Basic
   AuthUserFile /usr/local/nagios/etc/htpasswd.users
   Require valid-user
</Directory>
```

To setup apache authentication for user **nagiosadmin**

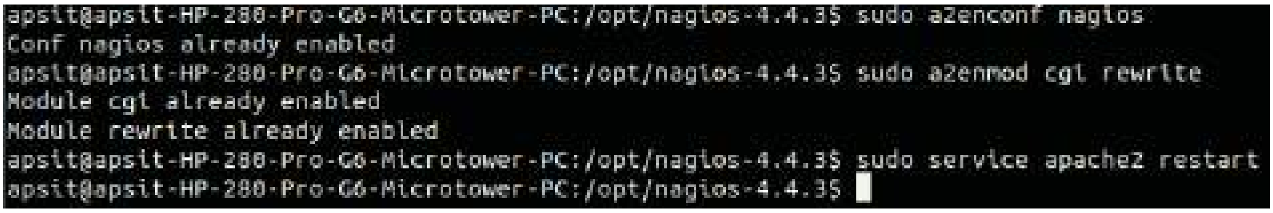
```
manjusha@apsit:~$sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users
nagiosadmin
```



```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:/opt/nagios-4.4.3$ sudo nano /etc/apache2/conf-available/nagios.conf
apsit@apsit-HP-280-Pro-G6-Microtower-PC:/opt/nagios-4.4.3$ sudo htpasswd -c /usr/local/nagios/etc/htpasswd.users nagiosadmin
New password:
Re-type new password:
Adding password for user nagiosadmin
```

Enable Apache configuration and restart Apache service to make the new settings take effect.cd

```
manjusha@apsit:~$sudo a2enconf nagios
manjusha@apsit:~$sudo a2enmod cgi rewrite
manjusha@apsit:~$sudo service apache2 restart
```

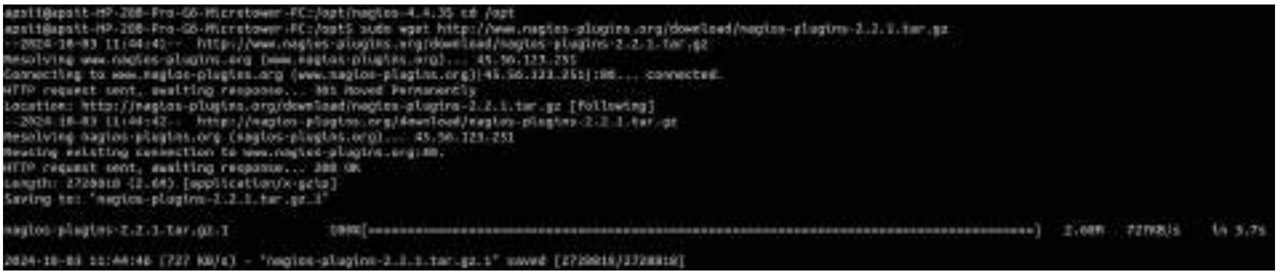


```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:/opt/nagios-4.4.3$ sudo a2enconf nagios
Conf nagios already enabled
apsit@apsit-HP-280-Pro-G6-Microtower-PC:/opt/nagios-4.4.3$ sudo a2enmod cgi rewrite
Module cgi already enabled
Module rewrite already enabled
apsit@apsit-HP-280-Pro-G6-Microtower-PC:/opt/nagios-4.4.3$ sudo service apache2 restart
apsit@apsit-HP-280-Pro-G6-Microtower-PC:/opt/nagios-4.4.3$
```

Step 5 – Installing Nagios Plugins

After installing and configuring Nagios core service, Download latest nagios-plugins source and install using follocdwing commands.

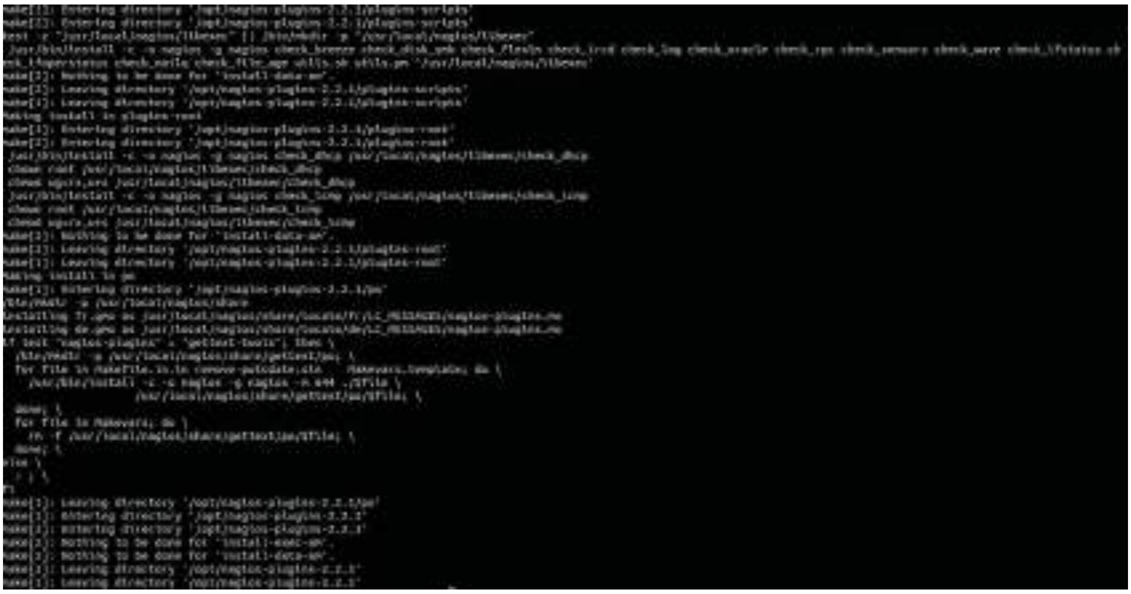
```
manjusha@apsit:~$cd /opt
manjusha@apsit:~$sudo wget http://www.nagios-plugins.org/download/nagios-
plugins-2.2.1.tar.gz
manjusha@apsit:~$sudo tar xzf nagios-plugins-2.2.1.tar.gznagios
manjusha@apsit:~$cd nagios-plugins-2.2.1
```



```
apsit@apsit-HP-280-Pro-G6-Microtower-PC:/opt/nagios-4.4.3$ cd /opt
apsit@apsit-HP-280-Pro-G6-Microtower-PC:/opt$ sudo wget http://www.nagios-plugins.org/download/nagios-plugins-2.2.1.tar.gz
--2024-10-03 11:44:41--  http://www.nagios-plugins.org/download/nagios-plugins-2.2.1.tar.gz
Resolving www.nagios-plugins.org (www.nagios-plugins.org)... 45.56.123.251
Connecting to www.nagios-plugins.org (www.nagios-plugins.org)|45.56.123.251|:80... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: http://nagios-plugins.org/download/nagios-plugins-2.2.1.tar.gz [following]
--2024-10-03 11:44:42--  http://nagios-plugins.org/download/nagios-plugins-2.2.1.tar.gz
Resolving nagios-plugins.org (nagios-plugins.org)... 45.56.123.251
Reusing existing connection to www.nagios-plugins.org:80.
HTTP request sent, awaiting response... 200 OK
Length: 2728818 (2.6M) [application/x-gzip]
Saving to: 'nagios-plugins-2.2.1.tar.gz.1'

nagios-plugins-2.2.1.tar.gz.1          100%[===================================================================================>]   2.60M   727KB/s    in 3.7s

2024-10-03 11:44:46 (727 KB/s) - 'nagios-plugins-2.2.1.tar.gz.1' saved [2728818/2728818]
```

Now compile and install Nagios plugins

```
manjusha@apsit:~$sudo ./configure --with-nagios-user=nagios --with-nagios-
group=nagios --with-openssl
manjusha@apsit:~$sudo make
manjusha@apsit:~$sudo make install
```

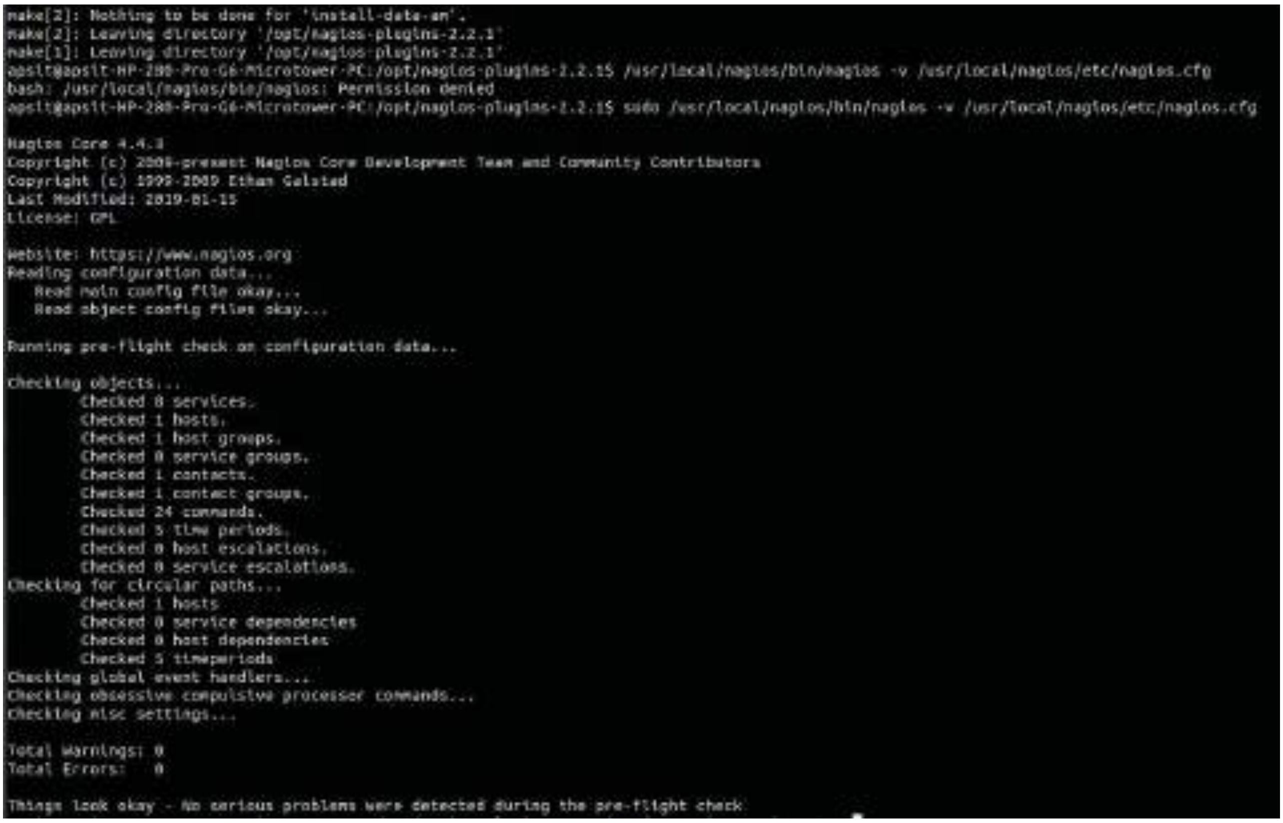


```
make[1]: Entering directory '/opt/nagios-plugins-2.2.1/plugins-scripts'
make[2]: Entering directory '/opt/nagios-plugins-2.2.1/plugins-scripts'
test -z "/usr/local/nagios/libexec" || /bin/mkdir -p "/usr/local/nagios/libexec"
 /usr/bin/install -c -o nagios -g nagios check_breeze check_disk_smb check_flexlm check_ircd check_log check_oracle check_rpc check_sensors check_wave check_ifstatus ch
eck_ifoperstatus check_mailq check_file_age utils.sh utils.pm '/usr/local/nagios/libexec'
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/opt/nagios-plugins-2.2.1/plugins-scripts'
make[1]: Leaving directory '/opt/nagios-plugins-2.2.1/plugins-scripts'
Making install in plugins-root
make[1]: Entering directory '/opt/nagios-plugins-2.2.1/plugins-root'
make[2]: Entering directory '/opt/nagios-plugins-2.2.1/plugins-root'
 /usr/bin/install -c -o nagios -g nagios check_dhcp /usr/local/nagios/libexec/check_dhcp
 chown root /usr/local/nagios/libexec/check_dhcp
 chmod ug=rx,u+s /usr/local/nagios/libexec/check_dhcp
 /usr/bin/install -c -o nagios -g nagios check_icmp /usr/local/nagios/libexec/check_icmp
 chown root /usr/local/nagios/libexec/check_icmp
 chmod ug=rx,u+s /usr/local/nagios/libexec/check_icmp
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/opt/nagios-plugins-2.2.1/plugins-root'
make[1]: Leaving directory '/opt/nagios-plugins-2.2.1/plugins-root'
Making install in po
make[1]: Entering directory '/opt/nagios-plugins-2.2.1/po'
/bin/mkdir -p /usr/local/nagios/share
installing fr.gmo as /usr/local/nagios/share/locale/fr/LC_MESSAGES/nagios-plugins.mo
installing de.gmo as /usr/local/nagios/share/locale/de/LC_MESSAGES/nagios-plugins.mo
if test "nagios-plugins" = "gettext-tools"; then \
  /bin/mkdir -p /usr/local/nagios/share/gettext/po; \
  for file in Makefile.in.in remove-potcdate.sin    Makevars.template; do \
    /usr/bin/install -c -o nagios -g nagios -m 644 ./$file \
                        /usr/local/nagios/share/gettext/po/$file; \
  done; \
  for file in Makevars; do \
    rm -f /usr/local/nagios/share/gettext/po/$file; \
  done; \
else \
  : ; \
fi
make[1]: Leaving directory '/opt/nagios-plugins-2.2.1/po'
make[1]: Entering directory '/opt/nagios-plugins-2.2.1'
make[2]: Entering directory '/opt/nagios-plugins-2.2.1'
make[2]: Nothing to be done for 'install-exec-am'.
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/opt/nagios-plugins-2.2.1'
make[1]: Leaving directory '/opt/nagios-plugins-2.2.1'
```

Step 6 – Verify Settings

Use the Nagios commands to verify the Nagios installation and configuration file. After successfully verify start the Nagios core service.

```
manjusha@apsit:~$/usr/local/nagios/bin/nagios                                    -v
/usr/local/nagios/etc/nagios.cfg
manjusha@apsit:~$ sudo service nagios start
```



```
make[2]: Nothing to be done for 'install-data-am'.
make[2]: Leaving directory '/opt/nagios-plugins-2.2.1'
make[1]: Leaving directory '/opt/nagios-plugins-2.2.1'
apsit@apsit-HP-280-Pro-G6-Microtower-PC:/opt/nagios-plugins-2.2.1$ /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg
bash: /usr/local/nagios/bin/nagios: Permission denied
apsit@apsit-HP-280-Pro-G6-Microtower-PC:/opt/nagios-plugins-2.2.1$ sudo /usr/local/nagios/bin/nagios -v /usr/local/nagios/etc/nagios.cfg

Nagios Core 4.4.3
Copyright (c) 2009-present Nagios Core Development Team and Community Contributors
Copyright (c) 1999-2009 Ethan Galstad
Last Modified: 2019-01-15
License: GPL

Website: https://www.nagios.org
Reading configuration data...
   Read main config file okay...
   Read object config files okay...

Running pre-flight check on configuration data...

Checking objects...
        Checked 8 services.
        Checked 1 hosts.
        Checked 1 host groups.
        Checked 0 service groups.
        Checked 1 contacts.
        Checked 1 contact groups.
        Checked 24 commands.
        Checked 5 time periods.
        Checked 0 host escalations.
        Checked 0 service escalations.
Checking for circular paths...
        Checked 1 hosts
        Checked 0 service dependencies
        Checked 0 host dependencies
        Checked 5 timeperiods
Checking global event handlers...
Checking obsessive compulsive processor commands...
Checking misc settings...

Total Warnings: 0
Total Errors:   0

Things look okay - No serious problems were detected during the pre-flight check
```

Also configure Nagios to auto start on system boot.

Step 7 – Access Nagios Web Interface

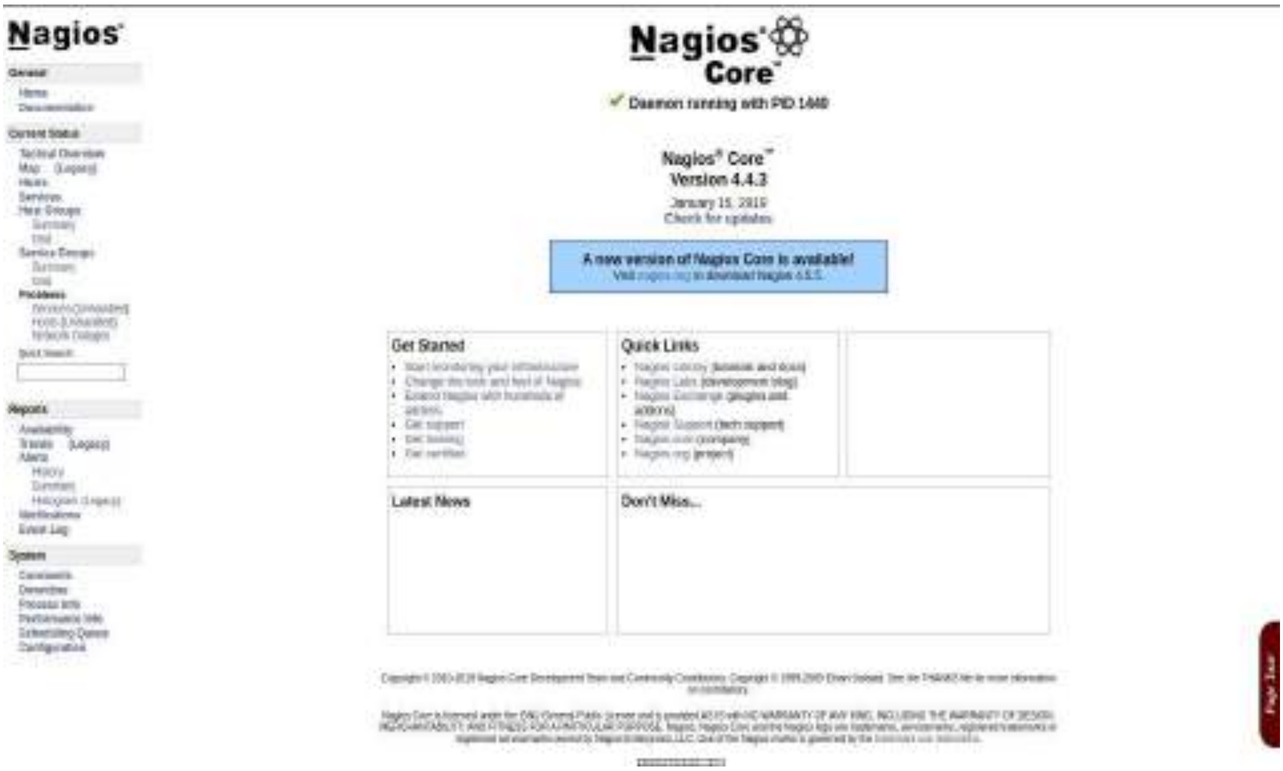
Access your nagios setup by access nagios server using hostname or ip address followed by /nagios.

http://127.0.0.1/nagios/

Prompting for Apache Authentication Password –

username: nagiosadmin
Password : 123456 (which you enter while configuration)

Nagios After login screen –



✓ Daemon running with PID 1440

Nagios® Core™
Version 4.4.3
January 15, 2019
Check for updates

A new version of Nagios Core is available!
Visit nagios.org to download Nagios 4.5.5.

We have successfully installed and configured Nagios Monitoring Server core service in our system now we need to install NRPE on all remote Linux systems to monitor with Nagios.

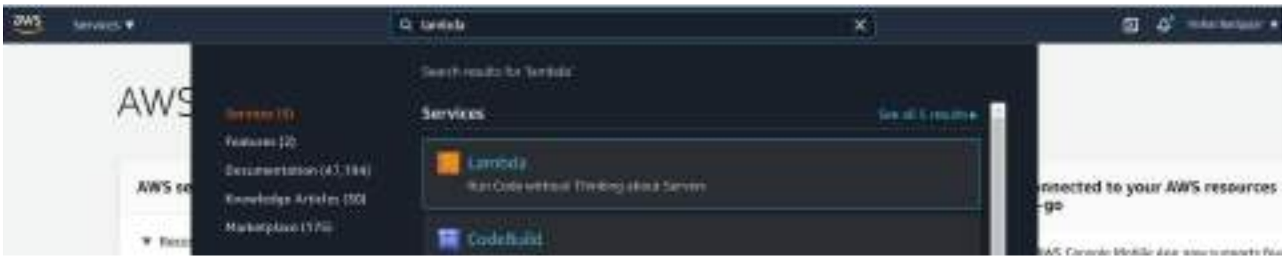
Conclusion: Hence, we successfully understood Continuous monitoring and Installation and configuration of Nagios Core, Nagios Plugins and NRPE (Nagios Remote Plugin Executor) on Linux Machine.

EXPERIMENT NO. 11

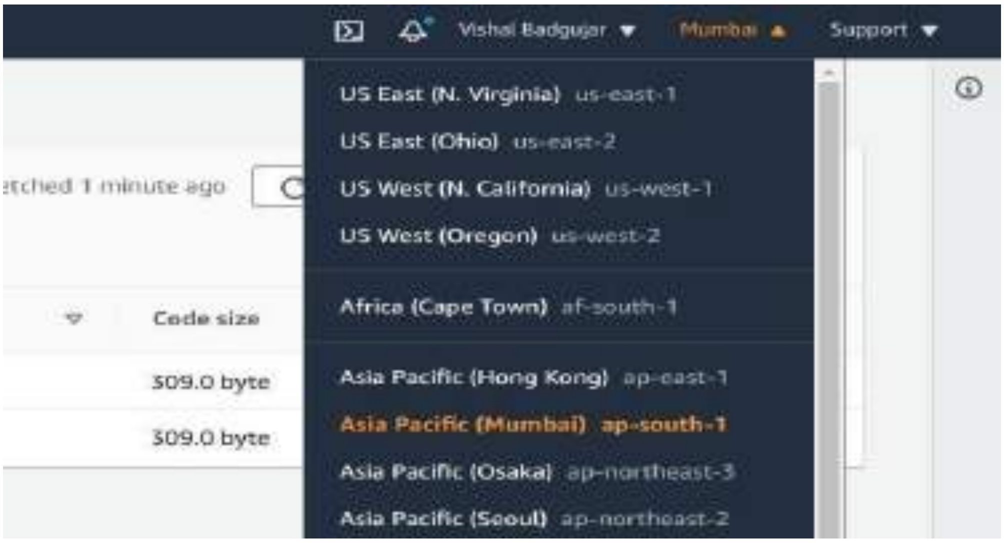
Aim: To understand AWS Lambda, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

Steps: First Lambda functions using Python

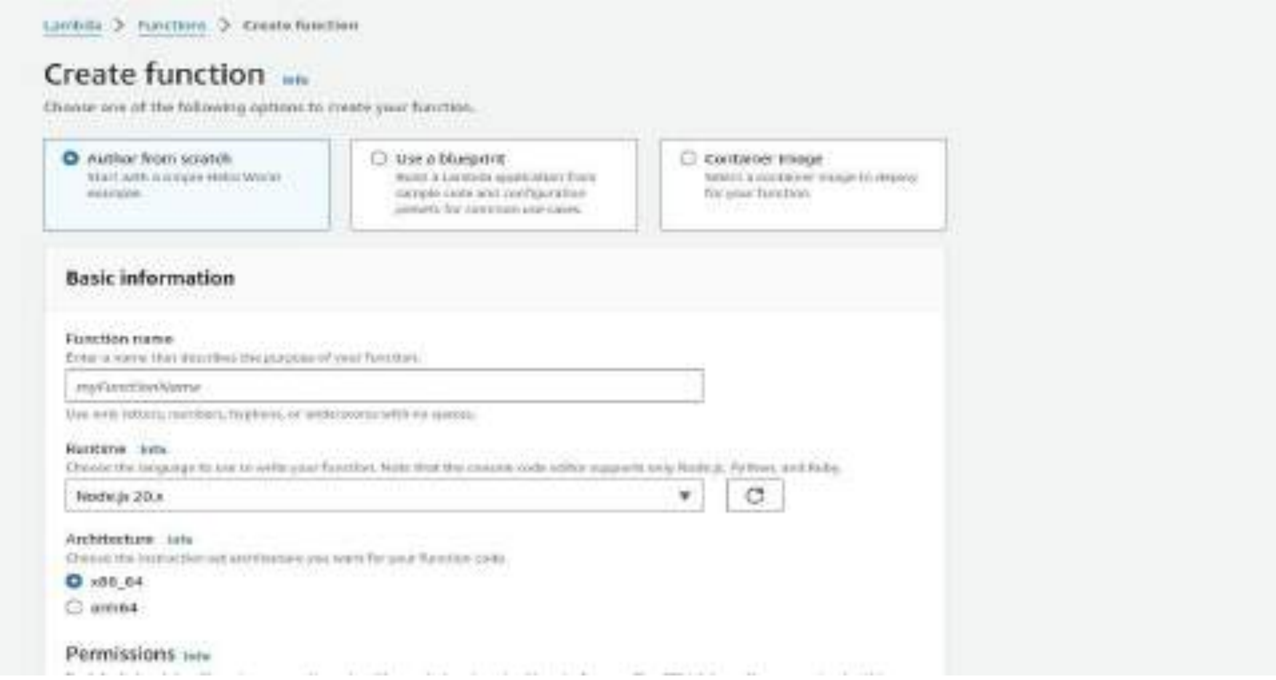
1. Open Aws Console and search for Lambda Service and open home screen of Lambda.



2. Choose region in which you need to create Lambda function as it is region specific.


Vishal Badgujar ▼
Mumbai ▲
Support ▼
US East (N. Virginia) us-east-1
US East (Ohio) us-east-2
US West (N. California) us-west-1
US West (Oregon) us-west-2
Africa (Cape Town) af-south-1
Asia Pacific (Hong Kong) ap-east-1
Asia Pacific (Mumbai) ap-south-1
Asia Pacific (Osaka) ap-northeast-3
Asia Pacific (Seoul) ap-northeast-2
tched 1 minute ago
Code size
309.0 byte
309.0 byte

3. Create sum as a Lambda Function in Python Language so select latest version of Python and choose role with basic Lambda Permission to allow cloudwatch for monitoring.



Lambda > Functions > Create function

Create function Info

Choose one of the following options to create your function.

● Author from scratch
Start with a simple Hello World example.

○ Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.

○ Container image
Select a container image to deploy for your function.

Basic information

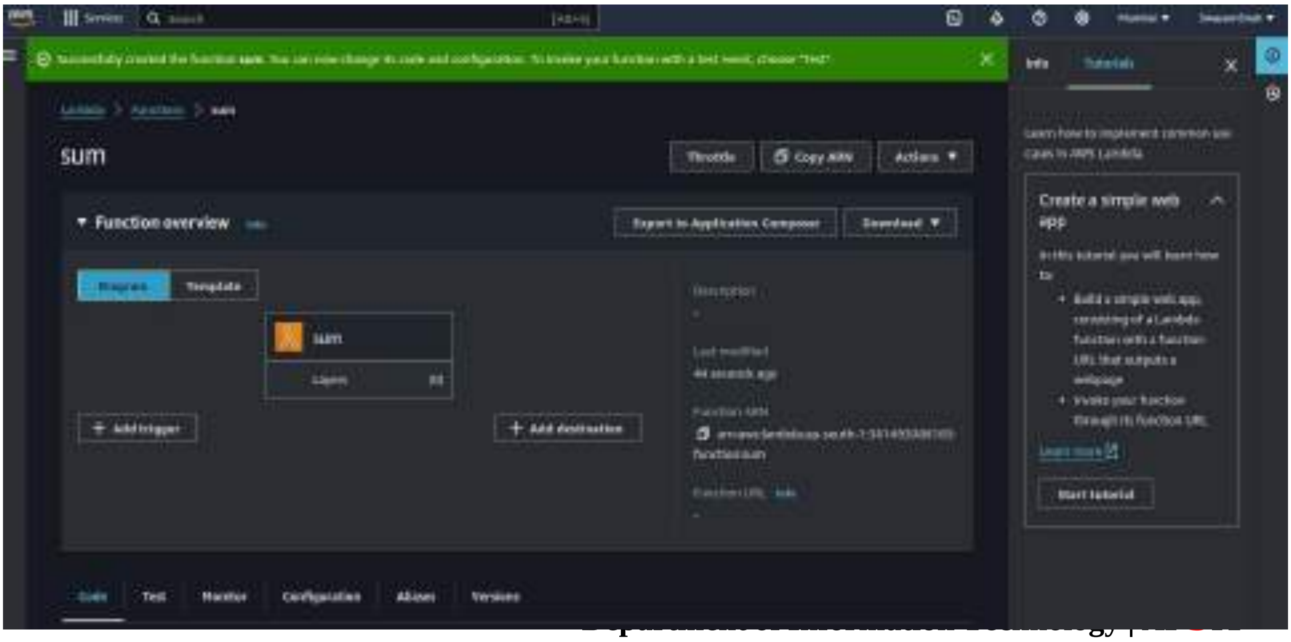
Function name
Enter a name that describes the purpose of your function.
myFunctionName
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Node.js 20.x

Architecture Info
Choose the instruction set architecture you want for your function code.
● x86_64
○ arm64

Permissions Info

4. Lambda sum function is created successfully



Successfully created the function sum. You can now change its code and configuration. To invoke your function with a test event, choose "Test".

Lambda > Functions > sum

sum

▼ Function overview Info

Description
-

Last modified
44 seconds ago

Function ARN
arn:aws:lambda:ap-south-1:[illegible]:function:sum

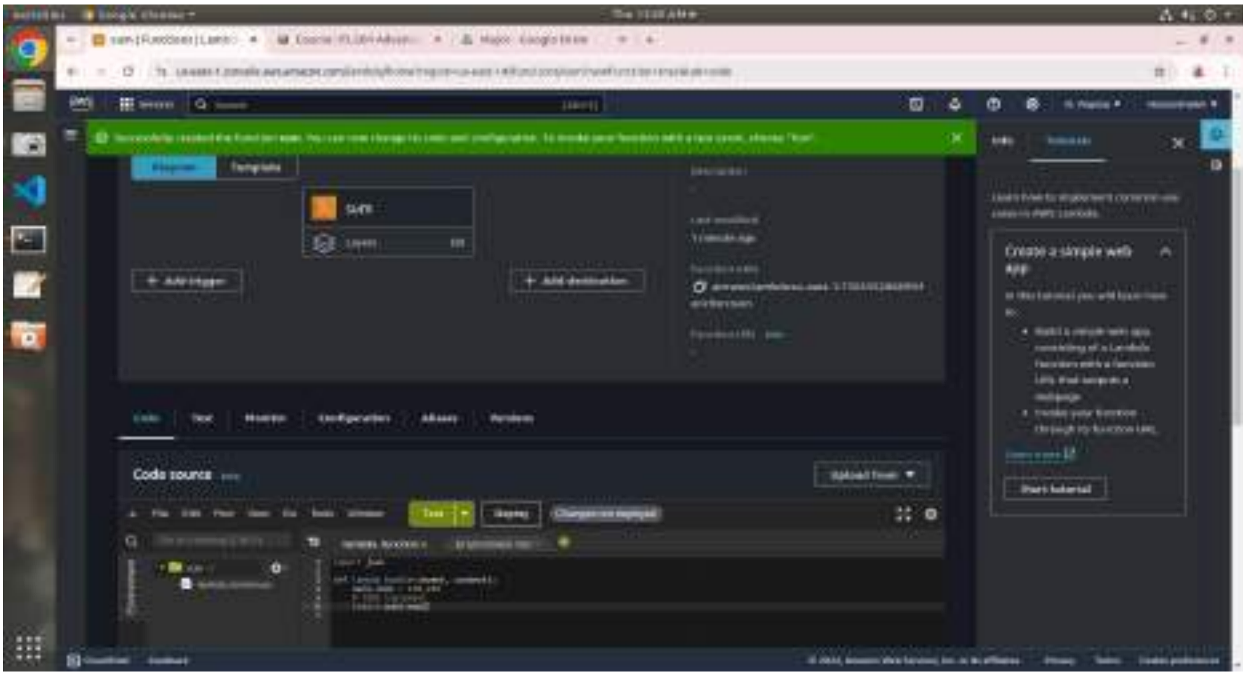
Function URL Info
-

Learn how to implement common use cases in AWS Lambda.

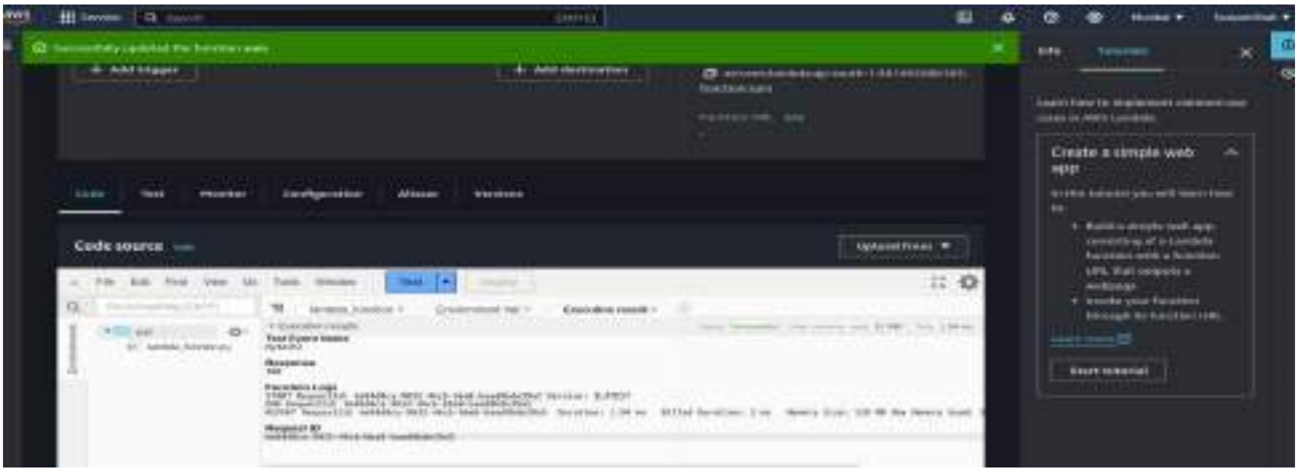
Create a simple web app

In this tutorial you will learn how to:
- Build a simple web app, consisting of a Lambda function with a function URL that outputs a webpage
- Invoke your function through its function URL

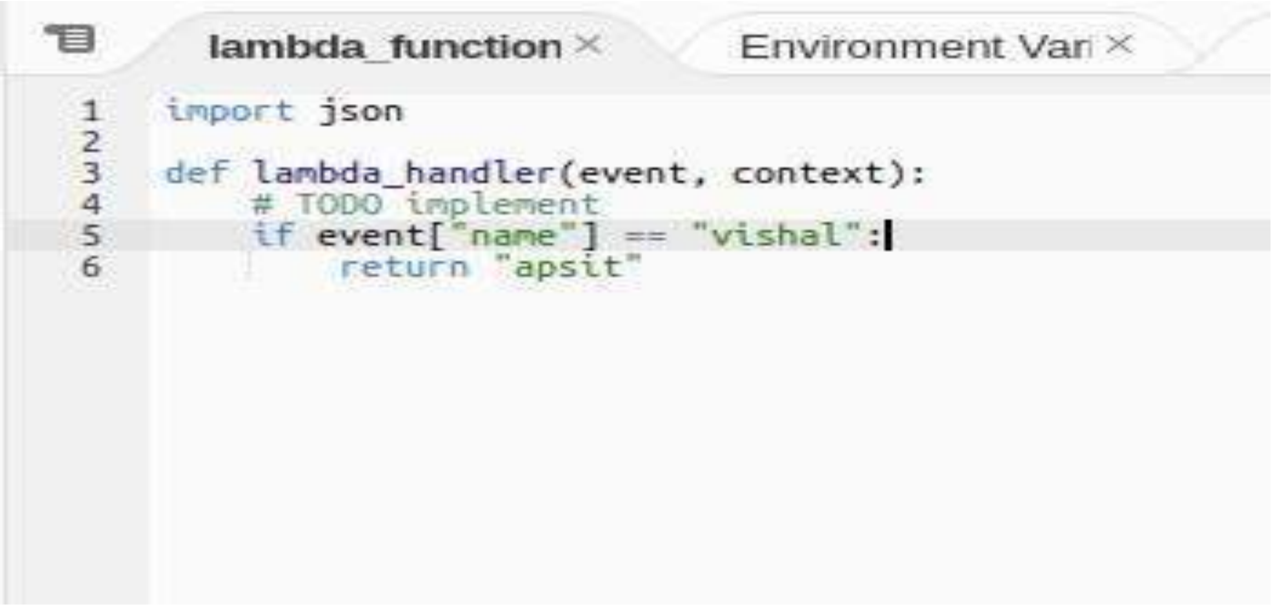
5. Write a sample python code for sum of two numbers:



6. Configure Test Event in Json Format

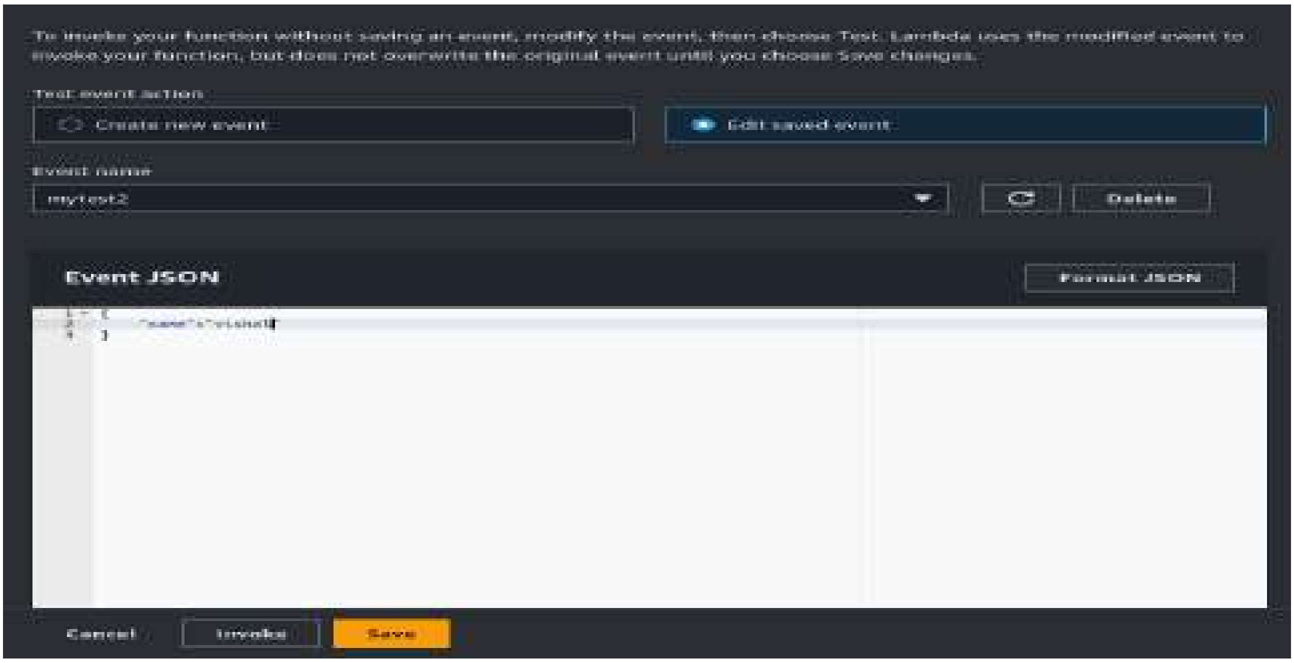


Write a sample Second sample python Code:



```python
import json

def lambda_handler(event, context):
    # TODO implement
    if event["name"] == "vishal":
        return "apsit"
```

Configure Test Event



To invoke your function without saving an event, modify the event, then choose Test. Lambda uses the modified event to invoke your function, but does not overwrite the original event until you choose Save changes.

Test event action

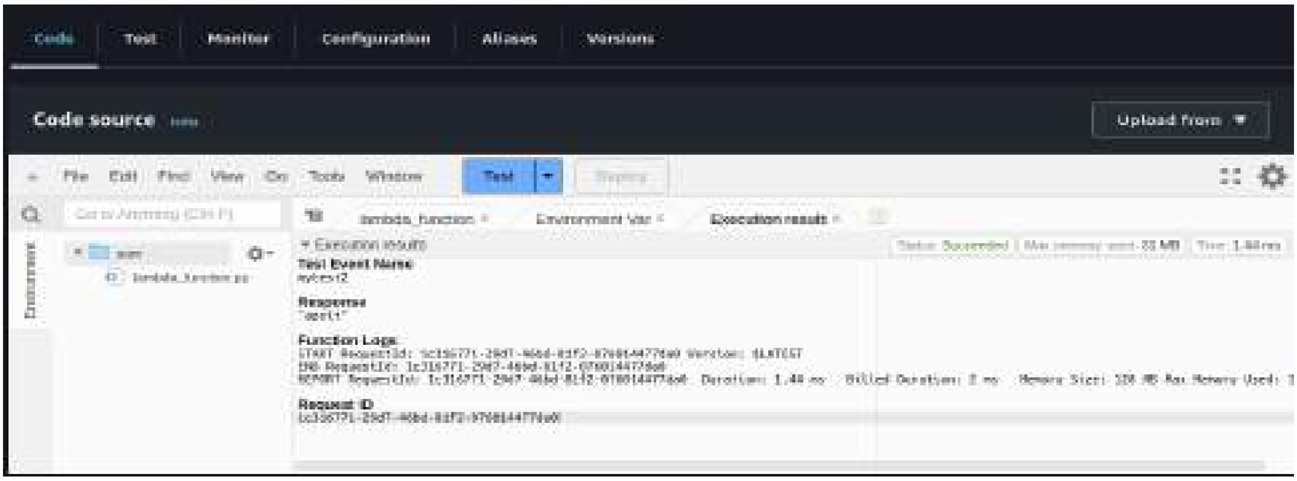
○ Create new event ● Edit saved event

Event name

mytest2

```
{
    "name":"vishal"
}
```

If condition met returns a value as apsit



▾ Execution results

Status: Succeeded | Max memory used: 33 MB | Time: 1.44 ms

Test Event Name
mytest2

Response
"apsit"

Function Logs

```
START RequestId: 1c316771-29d7-46bd-81f2-07601447...a0 Version: $LATEST
END RequestId: 1c316771-29d7-46bd-81f2-07601447...a0
REPORT RequestId: 1c316771-29d7-46bd-81f2-07601447...a0  Duration: 1.44 ms  Billed Duration: 2 ms  Memory Size: 128 MB Max Memory Used: 3
```

Request ID
1c316771-29d7-46bd-81f2-07601447...a0

Conclusion: Hence, we have understood the use of lambda service of amazon console, its workflow, various functions and create your first Lambda functions using Python / Java / Nodejs.

EXPERIMENT NO. 12

Aim: To create a Lambda function which will log "An Image has been added" once you add an object to a specific bucket in S3

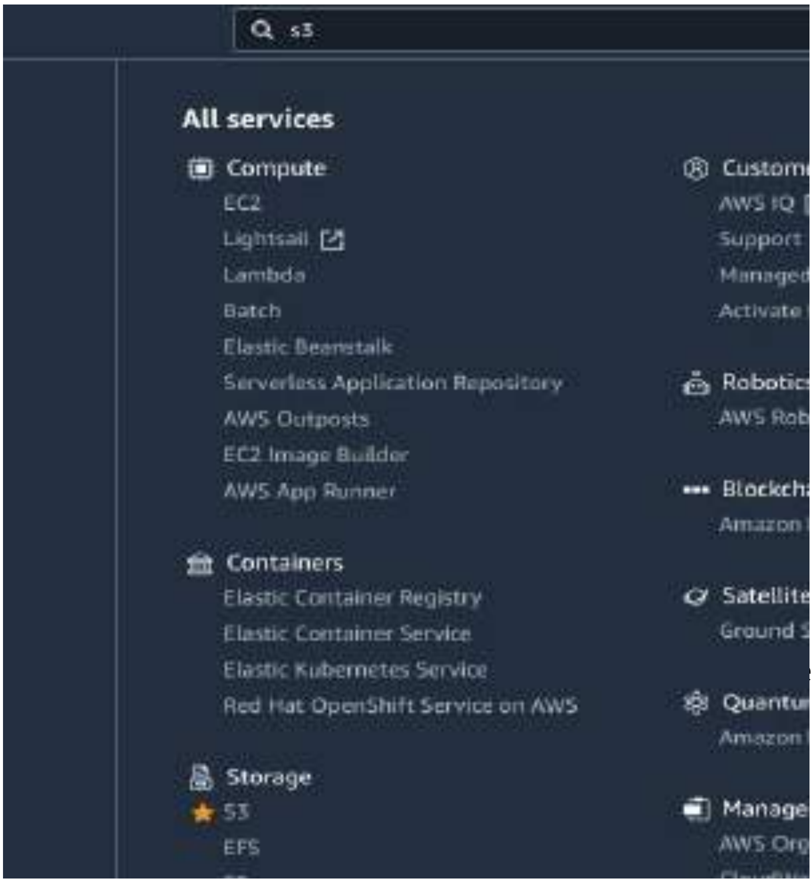
Theory:

Creating S3 Bucket

Let us start first by creating a s3 bucket in AWS console using the steps given below —

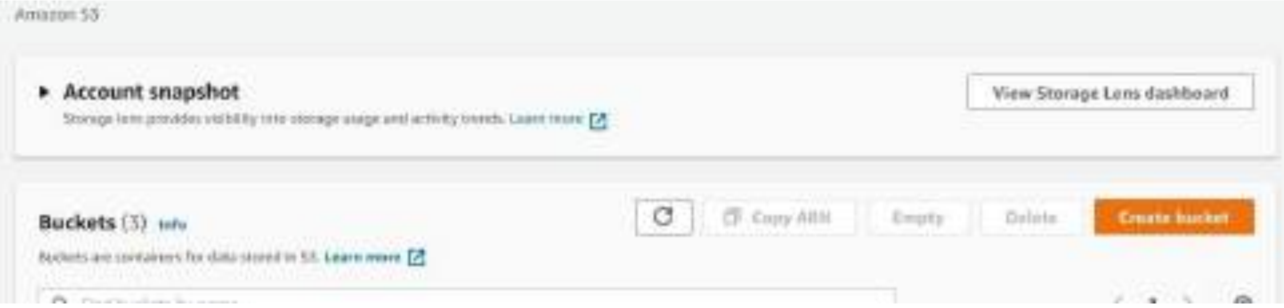
Step 1

Go to Amazon services and click **S3** in storage section as highlighted in the image given below −



Step 2

Click **S3** storage and **Create bucket** which will store the files uploaded.



Amazon S3

▸ **Account snapshot**

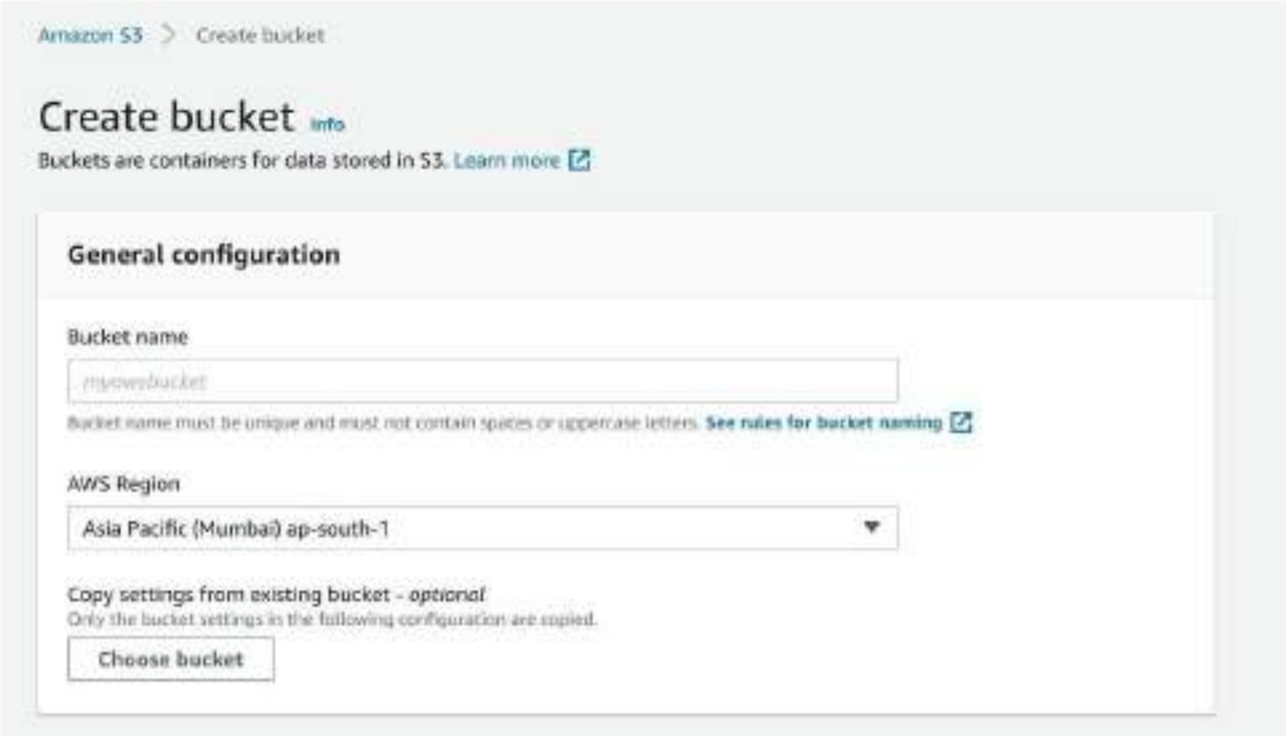
Storage lens provides visibility into storage usage and activity trends. Learn more

Buckets (3) Info

Buckets are containers for data stored in S3. Learn more

Step 3

Once you click Create bucket button, you can see a screen as follows −



Amazon S3 > Create bucket

Create bucket Info

Buckets are containers for data stored in S3. Learn more

General configuration

Bucket name: myawsbucket

Bucket name must be unique and must not contain spaces or uppercase letters. See rules for bucket naming

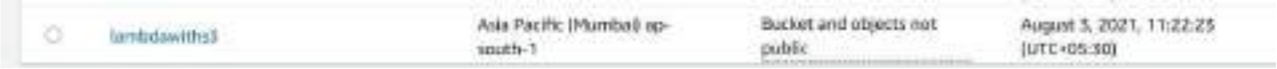
AWS Region: Asia Pacific (Mumbai) ap-south-1

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

Step 4

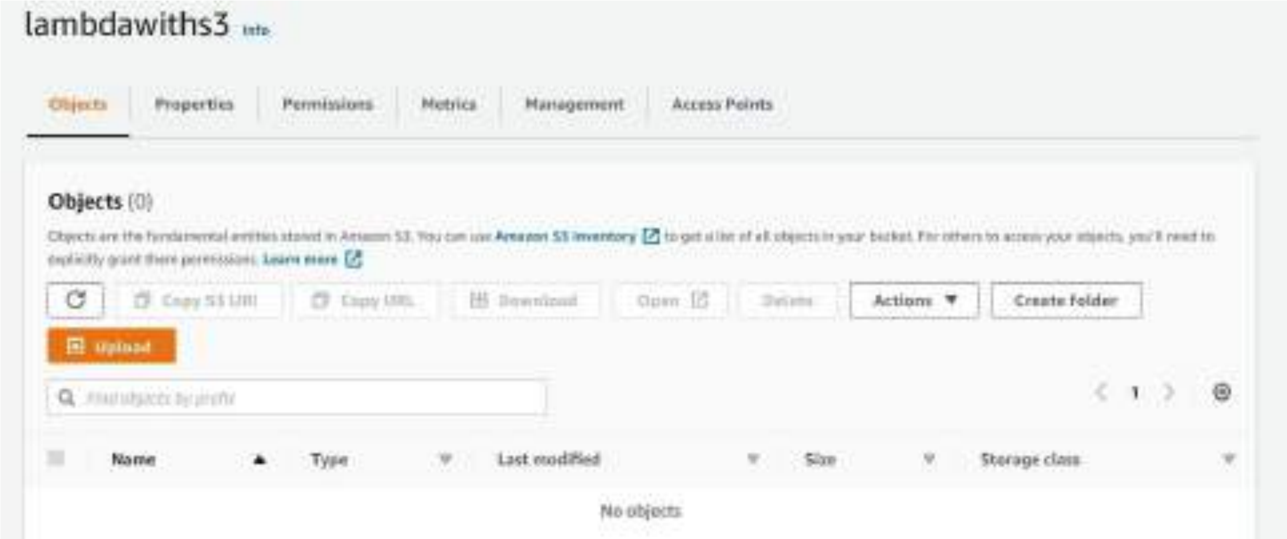
Enter the details Bucket name, Select the Region and click Create button at the bottom left side. Thus, we have created bucket with name :



○ lambdawiths3	Asia Pacific (Mumbai) ap-south-1	Bucket and objects not public	August 3, 2021, 11:22:25 (UTC+05:30)

Step 5

Now, click the bucket name and it will ask you to upload files as shown below —



lambdawiths3

Objects (0)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
No objects				

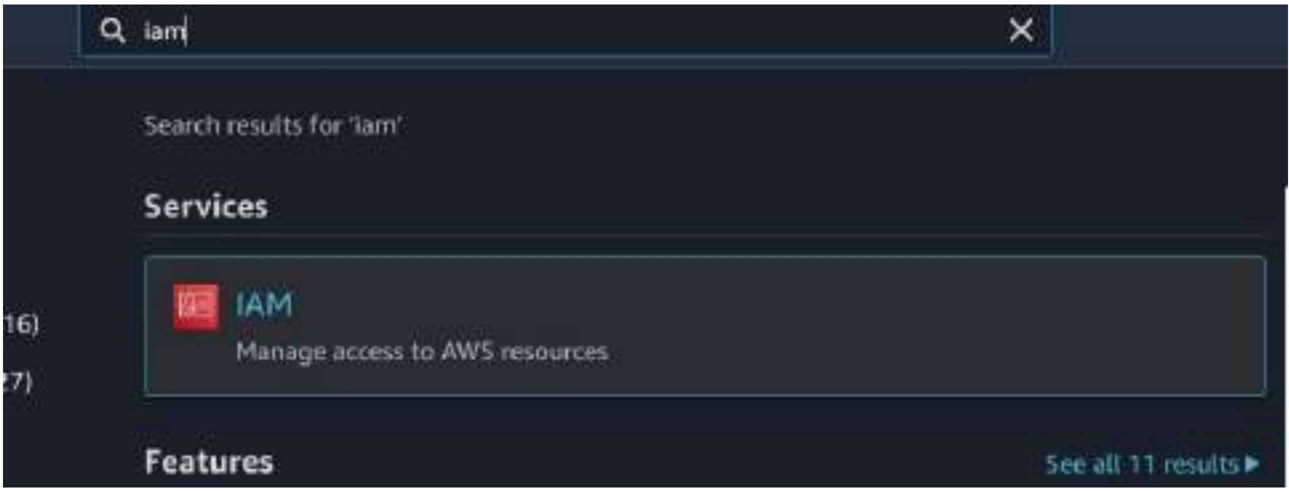
Thus, we are done with bucket creation in S3.

Create Role that Works with S3 and Lambda

To create role that works with S3 and Lambda, please follow the Steps given below —

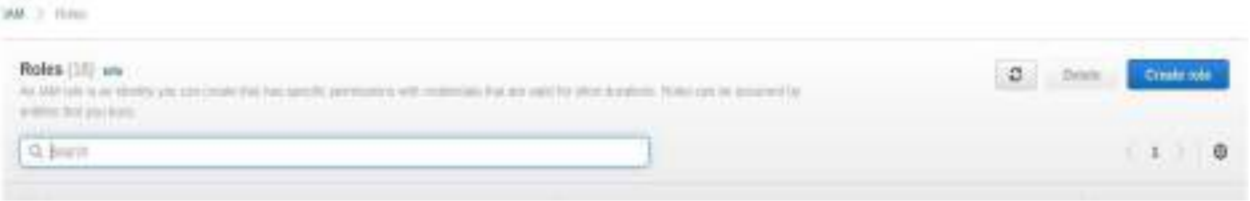
Step 1

Go to AWS services and select IAM as shown below —



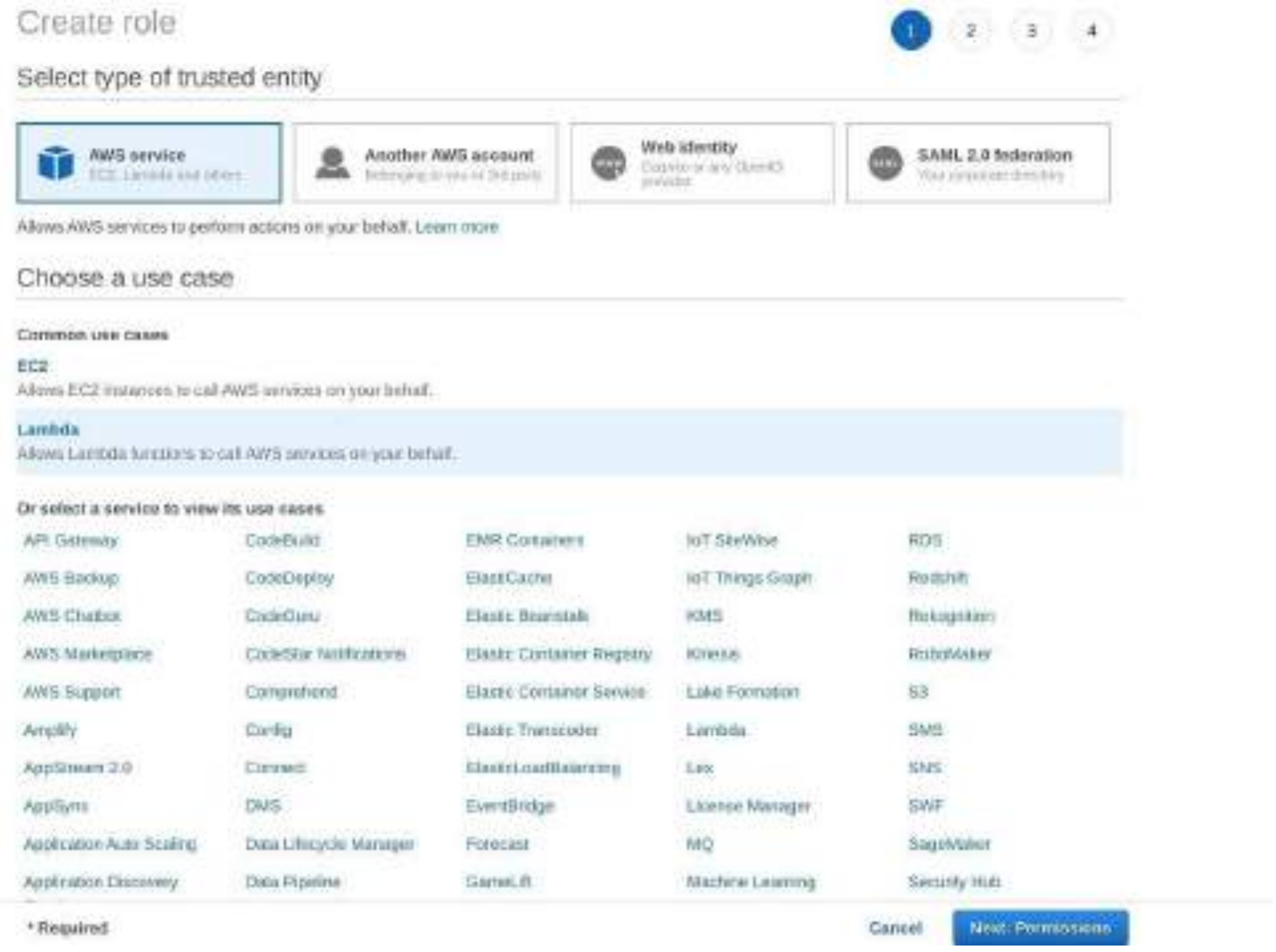
Step 2

Now, click **IAM -> Roles** as shown below —



Step 3

Now, click **Create role** and choose the services that will use this role. Select Lambda and click **Permission** button.



Create role

1 2 3 4

Select type of trusted entity

AWS service
EC2, Lambda and others

Another AWS account
Belonging to you or 3rd party

Web identity
Cognito or any OpenID provider

SAML 2.0 federation
Your corporate directory

Allows AWS services to perform actions on your behalf. Learn more

Choose a use case

Common use cases

EC2
Allows EC2 instances to call AWS services on your behalf.

Lambda
Allows Lambda functions to call AWS services on your behalf.

Or select a service to view its use cases

API Gateway
AWS Backup
AWS Chatbot
AWS Marketplace
AWS Support
Amplify
AppStream 2.0
AppSync
Application Auto Scaling
Application Discovery

CodeBuild
CodeDeploy
CodeGuru
CodeStar Notifications
Comprehend
Config
Connect
DMS
Data Lifecycle Manager
Data Pipeline

EMR Containers
ElastiCache
Elastic Beanstalk
Elastic Container Registry
Elastic Container Service
Elastic Transcoder
ElasticLoadBalancing
EventBridge
Forecast
GameLift

IoT SiteWise
IoT Things Graph
KMS
Kinesis
Lake Formation
Lambda
Lex
License Manager
MQ
Machine Learning

RDS
Redshift
Rekognition
RoboMaker
S3
SMS
SNS
SWF
SageMaker
Security Hub

* Required

Cancel
Next: Permissions

S

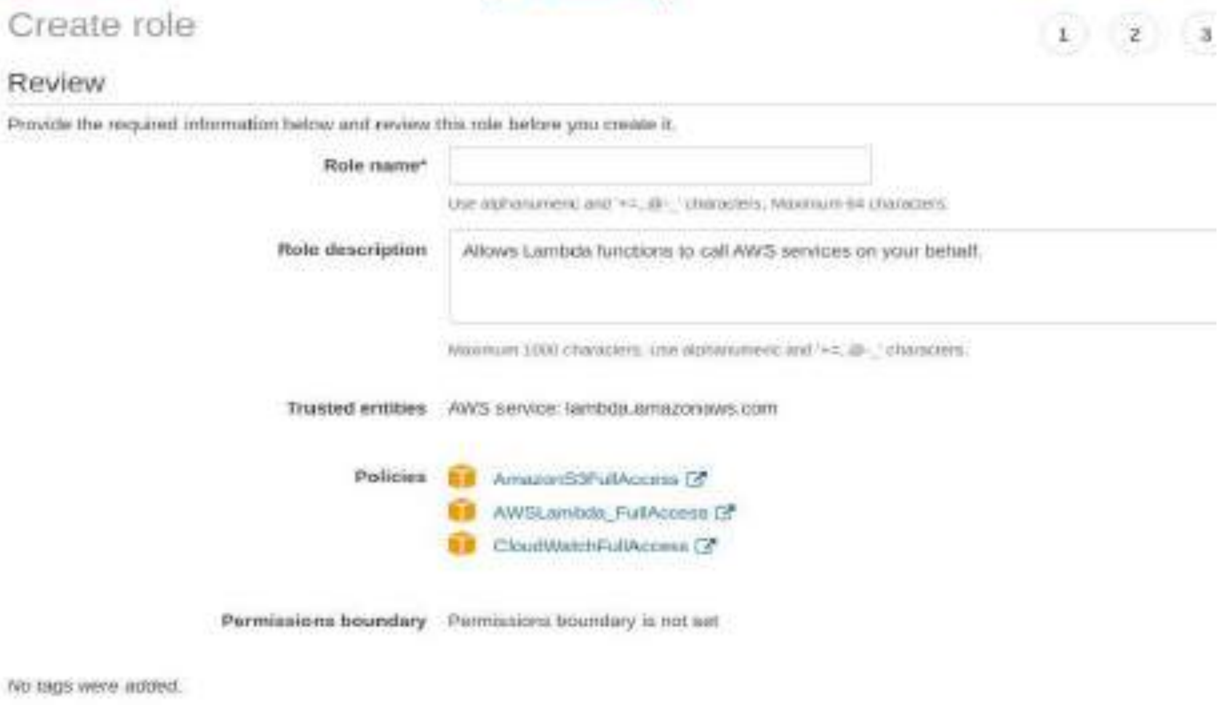
4

Add the permission from below and click Review.

AmazonS3FullAccess, AWSLambdaFullAccess and CloudWatchFullAccess.

Step 5

Observe that we have chosen the following permissions −



Create role

Review

Provide the required information below and review this role before you create it.

Role name*

Use alphanumeric and '+=,.@-_' characters. Maximum 64 characters.

Role description: Allows Lambda functions to call AWS services on your behalf.

Maximum 1000 characters. Use alphanumeric and '+=,.@-_' characters.

Trusted entities: AWS service: lambda.amazonaws.com

Policies:
- AmazonS3FullAccess
- AWSLambda_FullAccess
- CloudWatchFullAccess

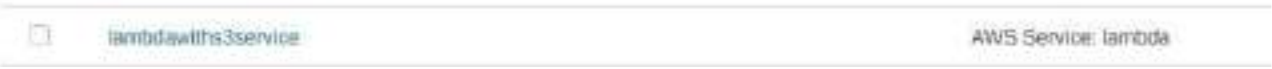
Permissions boundary: Permissions boundary is not set

No tags were added.

Observe that the Policies that we have selected are **AmazonS3FullAccess, AWSLambdaFullAccess and CloudWatchFullAccess.**

Step 6

Now, enter the Role name, Role description and click Create Role button at the bottom.



☐ lambdawiths3service AWS Service: lambda

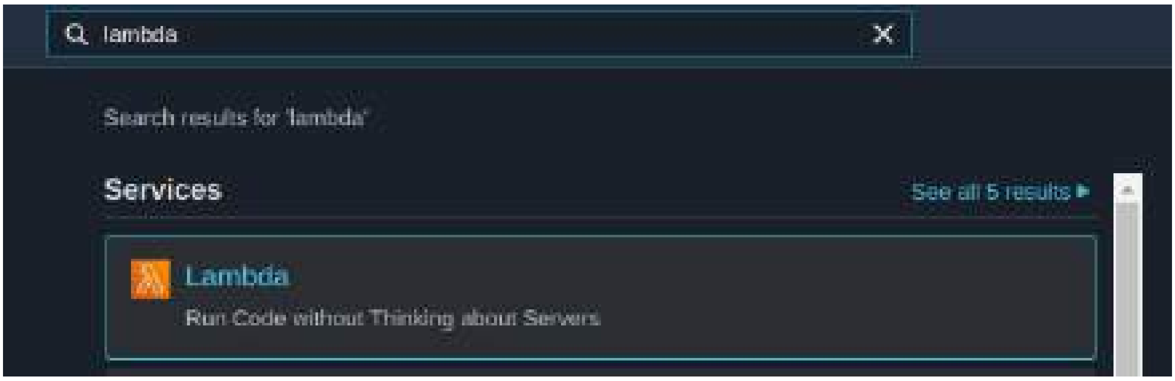
Thus, our role named lambdawiths3service is created.

Create Lambda function and Add S3 Trigger

In this section, let us see how to create a Lambda function and add a S3 trigger to it. For this purpose, you will have to follow th Steps given below —

Step 1

Go to AWS Services and select Lambda as shown below −

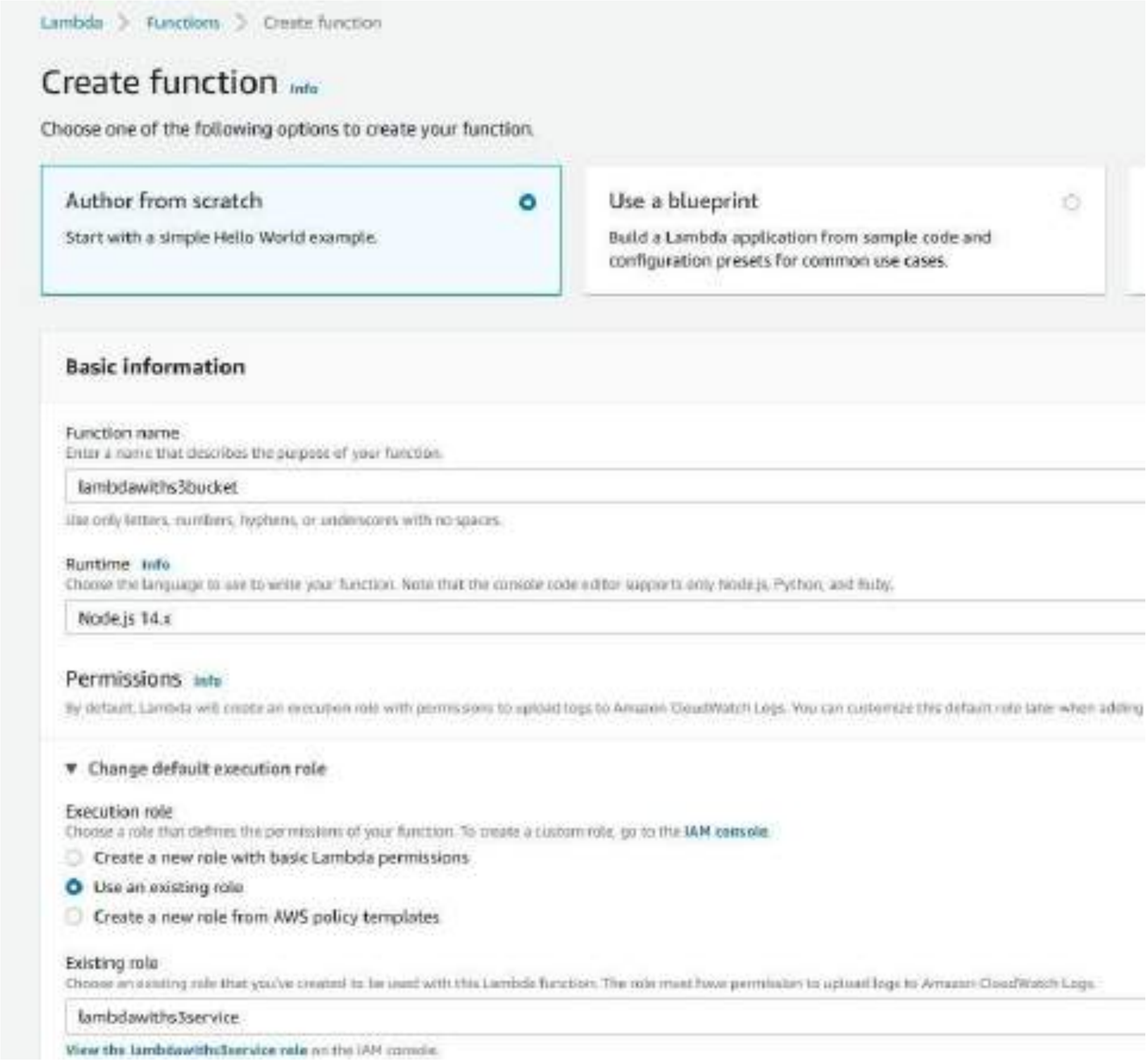


lambda

Search results for 'lambda'

Step 2

Click **Lambda** and follow the process for adding **Name**. Choose the **Runtime, Role** etc. and create the function. The Lambda function that we have created is shown in the screenshot below −



Lambda > Functions > Create function

Create function Info

Choose one of the following options to create your function.

● Author from scratch
Start with a simple Hello World example.

○ Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.

Basic information

Function name
Enter a name that describes the purpose of your function.
lambdawiths3bucket
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Node.js 14.x

Permissions Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding

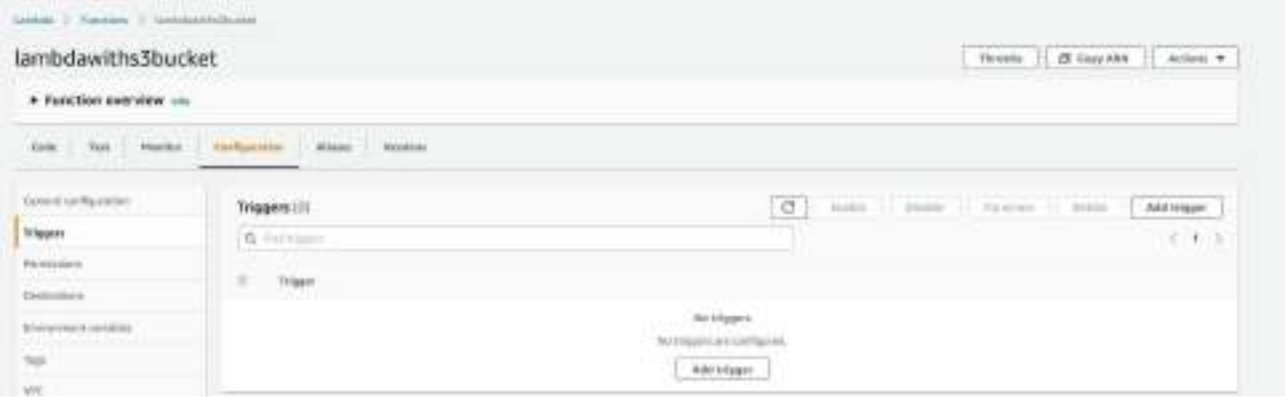
▼ Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
○ Create a new role with basic Lambda permissions
● Use an existing role
○ Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
lambdawiths3service
View the lambdawiths3service role on the IAM console.

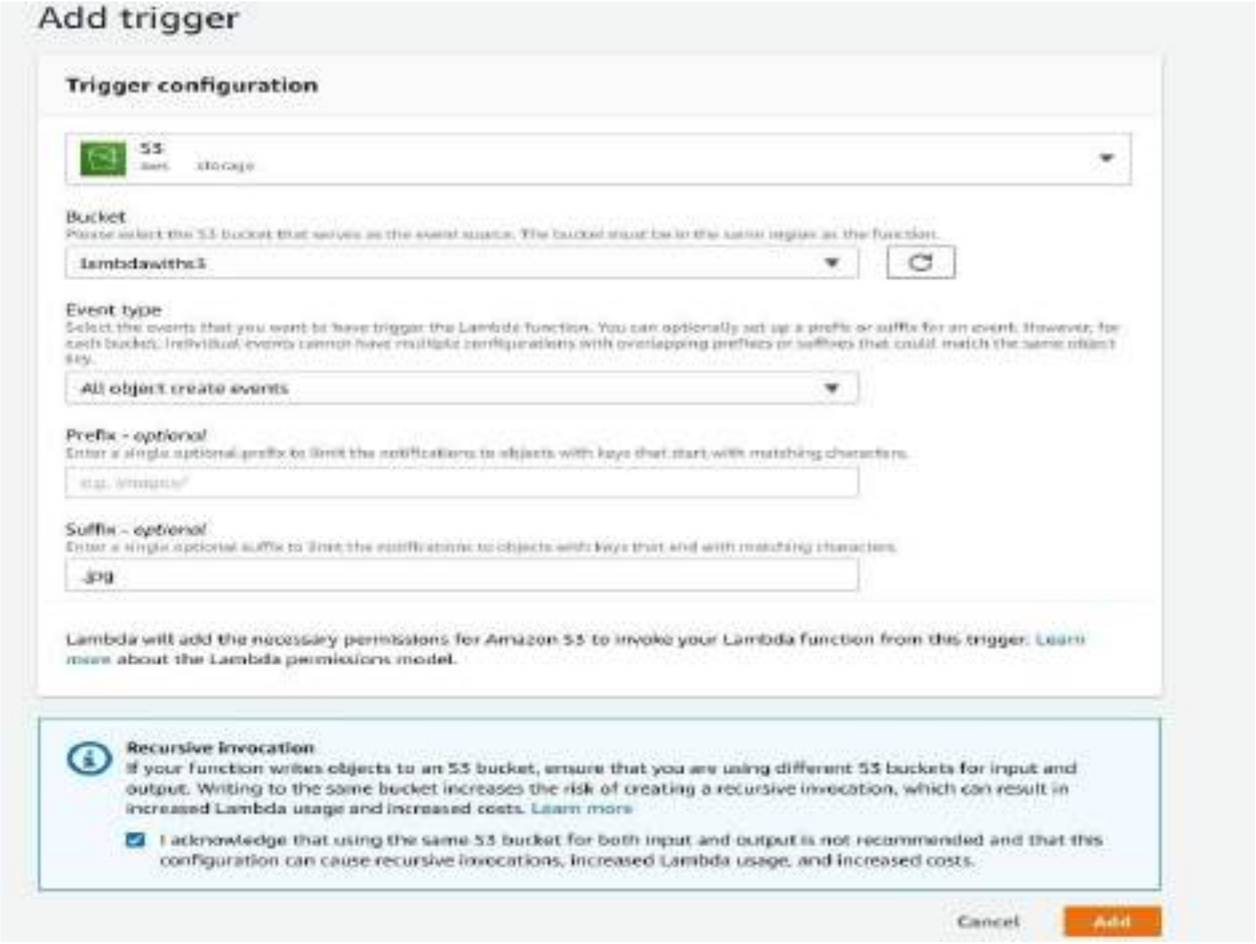
Step 3

Now let us add the S3 trigger.



Step 4

Choose the trigger from above and add the details as shown below —



Add trigger

Trigger configuration

S3
aws storage

Bucket
Please select the S3 bucket that serves as the event source. The bucket must be in the same region as the function.

lambdawiths3

Event type
Select the events that you want to have trigger the Lambda function. You can optionally set up a prefix or suffix for an event. However, for each bucket, individual events cannot have multiple configurations with overlapping prefixes or suffixes that could match the same object key.

All object create events

Prefix - *optional*
Enter a single optional prefix to limit the notifications to objects with keys that start with matching characters.

e.g. images/

Suffix - *optional*
Enter a single optional suffix to limit the notifications to objects with keys that end with matching characters.

.jpg

Lambda will add the necessary permissions for Amazon S3 to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.

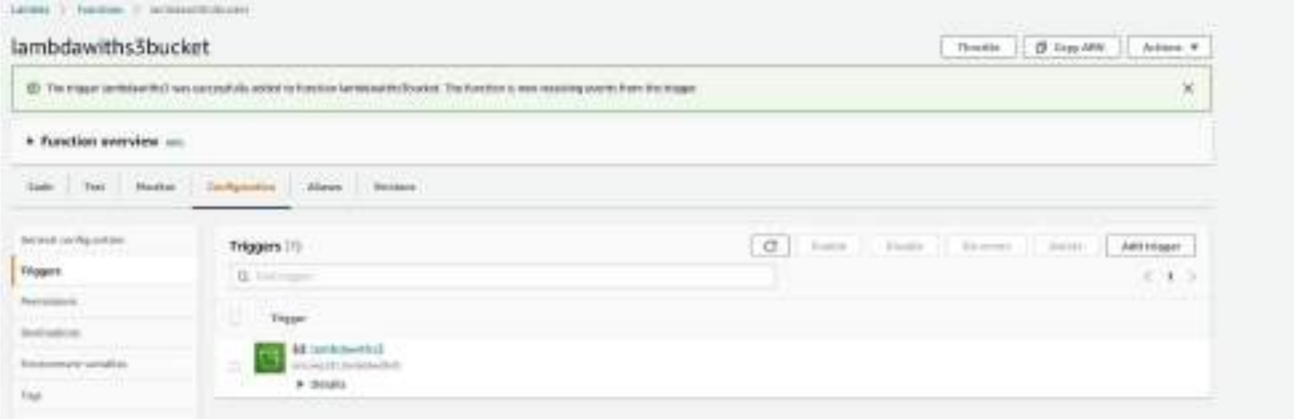
Recursive invocation
If your function writes objects to an S3 bucket, ensure that you are using different S3 buckets for input and output. Writing to the same bucket increases the risk of creating a recursive invocation, which can result in increased Lambda usage and increased costs. Learn more

☑ I acknowledge that using the same S3 bucket for both input and output is not recommended and that this configuration can cause recursive invocations, increased Lambda usage, and increased costs.

You can add Prefix and File pattern which are used to filter the files added. For Example, to trigger lambda only for .jpg images. as we need to trigger Lambda for all jpg image files uploaded. Click Add button to add the trigger.

Step 5

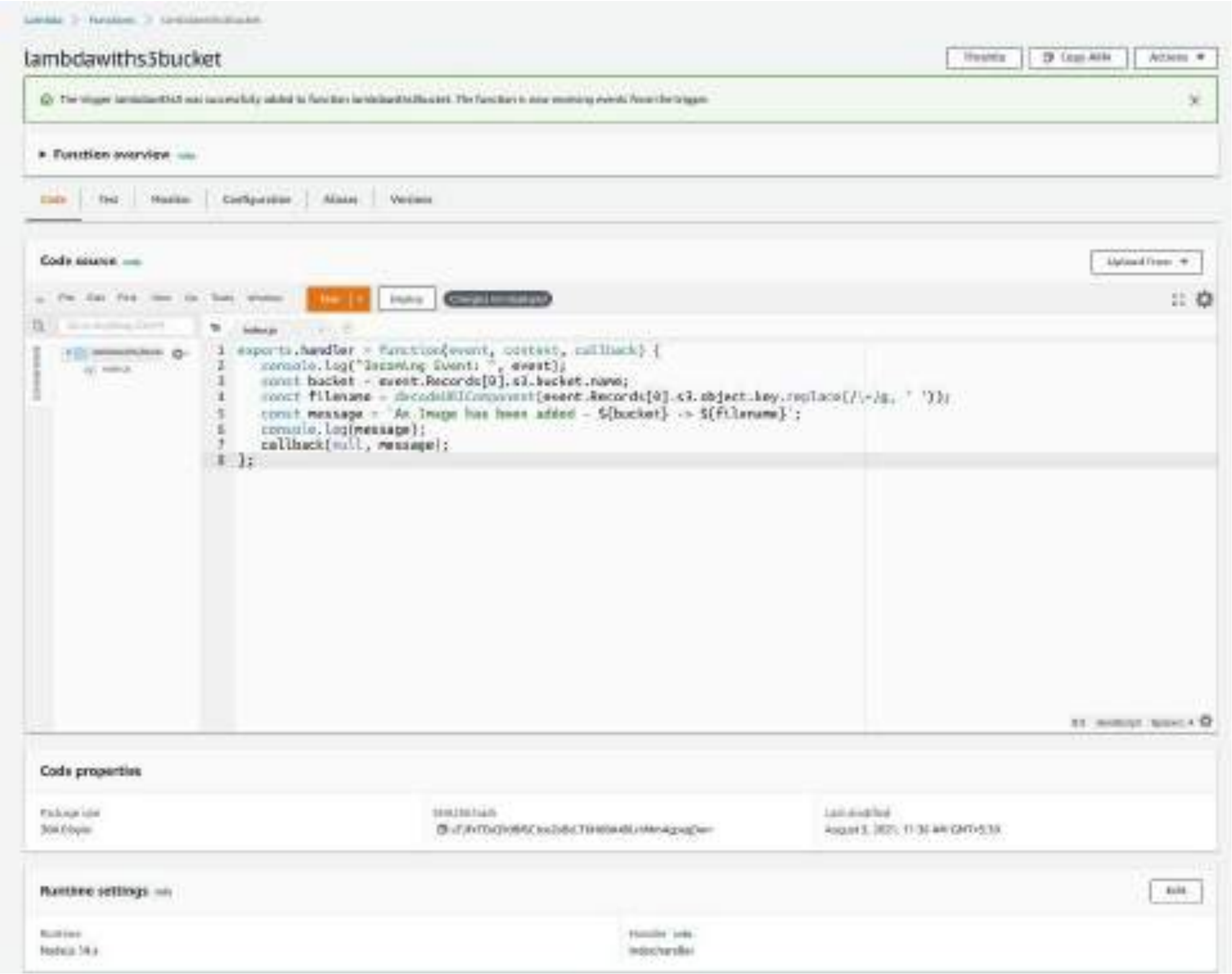
You can find the the trigger display for the Lambda function as shown below —



Step 6

Let’s add the details for the aws lambda function. Here, we will use the online editor to add our code and use nodejs as the runtime environment.

To trigger S3 with AWS Lambda, we will have to use S3 event in the code as shown below —



```
exports.handler = function(event, context, callback) {
  console.log("Incoming Event: ", event);
  const bucket = event.Records[0].s3.bucket.name;
  const filename = decodeURIComponent(event.Records[0].s3.object.key.replace(/\+/g, ' '));
  const message = `An Image has been added - ${bucket} -> ${filename}`;
  console.log(message);
  callback(null, message);
};
```

Step 7:

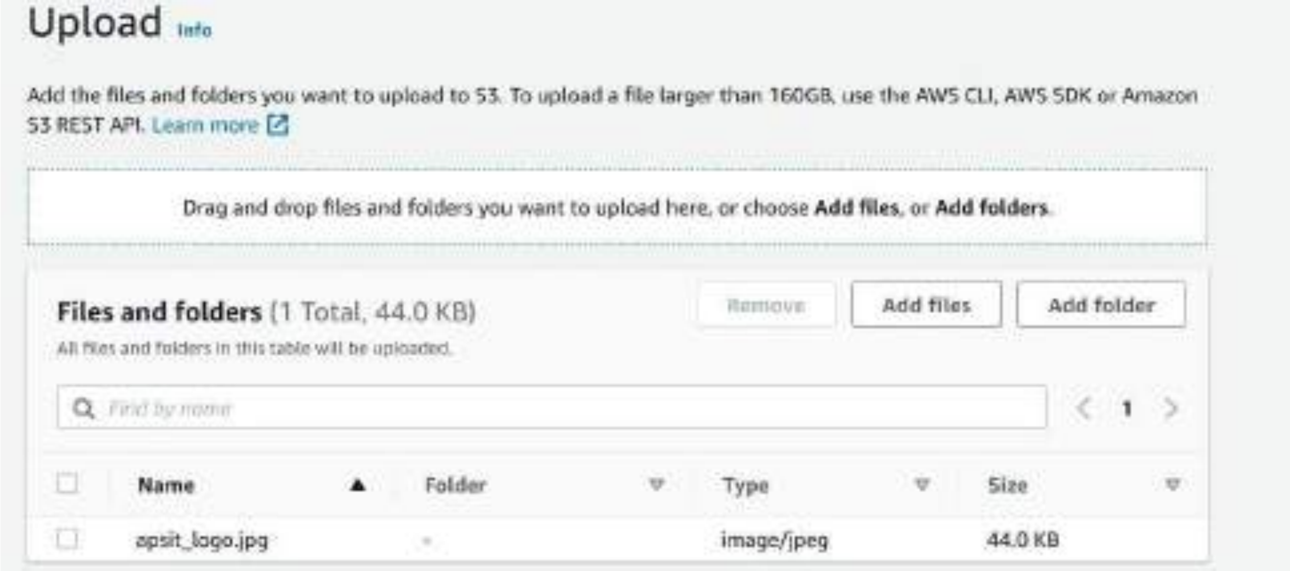
let us save the changes and test the lambda function with S3upload.

Step 8:

Now, save the Lambda function. Open S3 from Amazon services and open the bucket we created earlier namely lambdawiths3.

Upload the image in it as shown below —

Click **Add files** to add files. You can also drag and drop the files. Now, click **Upload** button.



Upload Info

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. Learn more

Drag and drop files and folders you want to upload here, or choose **Add files**, or **Add folders**.

Files and folders (1 Total, 44.0 KB)

All files and folders in this table will be uploaded.

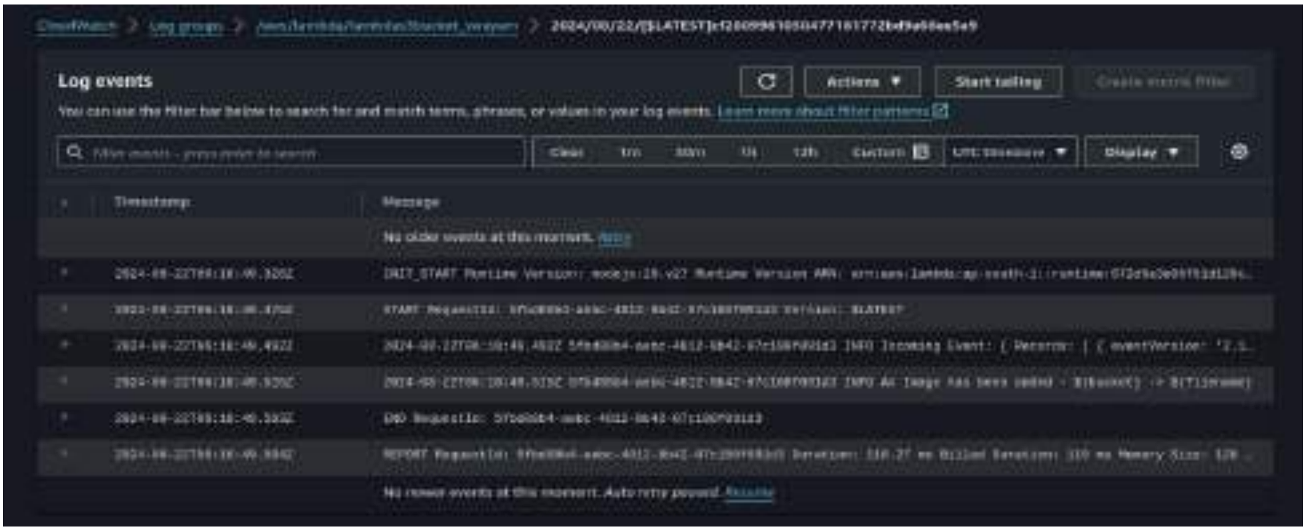
Name	Folder	Type	Size
apsit_logo.jpg	-	image/jpeg	44.0 KB

Thus, we have uploaded one image in our S3 bucket.

Step 9

To see the trigger details, go to AWS service and select CloudWatch. Open the logs for the Lambda

AWS Lambda function gets triggered when file is uploaded in S3 bucket and the details are logged in Cloudwatch as shown below —



CloudWatch > Log groups > /aws/lambda/[illegible] > 2024/08/22/[$LATEST][illegible]

Log events

You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns

```
Timestamp                  Message
                           No older events at this moment. Retry
2024-08-22T09:18:40.326Z   INIT_START Runtime Version: nodejs:20.v27 Runtime Version ARN: arn:aws:lambda:ap-south-1::runtime:[illegible]...
2024-08-22T09:18:40.470Z   START RequestId: [illegible] Version: $LATEST
2024-08-22T09:18:40.492Z   2024-08-22T09:18:40.492Z [illegible] INFO Incoming Event: { Records: [ { eventVersion: '2.1...
2024-08-22T09:18:40.535Z   2024-08-22T09:18:40.535Z [illegible] INFO An image has been added - ${bucket} -> ${filename}
2024-08-22T09:18:40.553Z   END RequestId: [illegible]
2024-08-22T09:18:40.553Z   REPORT RequestId: [illegible] Duration: 116.27 ms Billed Duration: 117 ms Memory Size: 128 ...
                           No newer events at this moment. Auto retry paused. Resume
```

An image has been Added -> apsit_logo.jpg you can see in cloudwatch logs.

Conclusion: Hence, we created a Lambda function which will log “An Image has been added” once you add an object to a specific bucket in S3.