

CONTINUOUS INTEGRATION PIPELINE FOR TOOLING WEBSITE

A DevOps team utilizes various tooling solutions in order to help the team carry out their day-to-day activities in managing, developing, testing, deploying and monitoring different projects. This project also majors on how Jenkins contribute to the DevOps process by continually integrating developer codes in a timely manner and shows how it connects with the webserver to deliver a deployable software products.

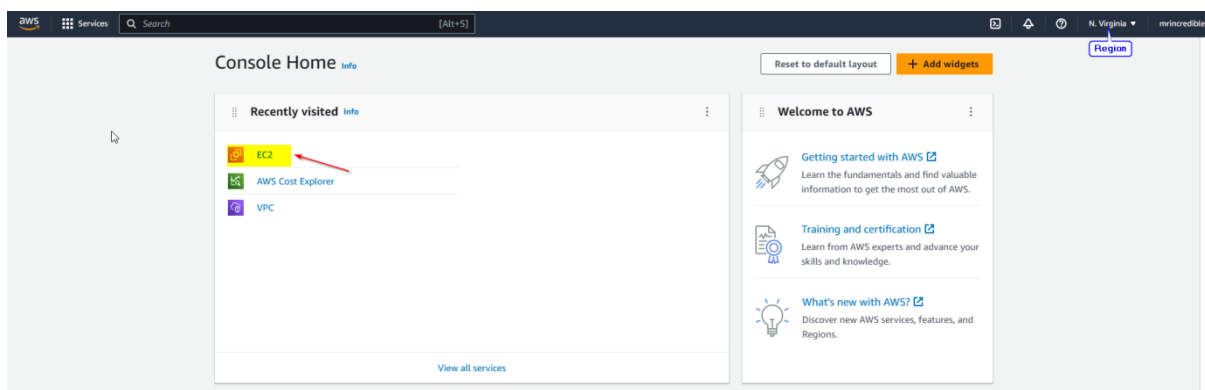
In this project we would be implementing a DevOps solution that consist of the following

Pre-requisite for the projects is the following.

- 1) Fundamental Knowledge of Installing and downloading software
- 2) Basic Understanding of Linux Commands
- 3) AWS account login with EC2 instances
- 4) Webserver Linux: Ubuntu 20.4 LTS
- 5) NFS Server: Red Hat Enterprise Linux 9
- 6) Code Repository: GitHub
- 7) Internet connection

IMPLEMENTATION STEPS:

- i) Ensure you login with your details to your AWS console via the <https://aws.amazon.com>
- ii) Click on the EC2 link to create instances.



- iii) Click on launch instance dropdown button and select launch

Launch instance

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Launch instance ▲

Migrate a server ↗

Launch instance

Launch instance from template

Virginia Region

Scheduled events

↻

US East (N. Virginia)

Service health

↻

AWS Health Dashboard ↗

Region

US East (N. Virginia)

Status

✔ This service is operating normally

Zones

Zone name	Zone ID
us-east-1a	use1-az2
us-east-1b	use1-az4

Select Ubuntu from the quick start option and note that amazon machine image selection varies from user to user .Select Ubuntu 20.4 LTS SSD Volume type .

jenkins

Add additional tags

▼ Application and OS Images (Amazon Machine Image) [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Q Search our full catalog including 1000s of application and OS images

Recents

Quick Start

Amazon Linux

aws

macOS

Mac

Ubuntu

ubuntu

Windows

Microsoft

Red Hat

Red Hat

S

➤

🔍

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Ubuntu Server 20.04 LTS (HVM), SSD Volume Type

Free tier eligible ▼

ami-0261755bbcb8c4a84 (64-bit (x86)) / ami-097d5b19d4f1a7d1b (64-bit (Arm))

Virtualization: hvm ENA enabled: true Root device type: ebs

Click on the “Create new key pair” link and ensure the Checkbox remains unchanged on the “Create security group”.

Key pair name - *required*

Select



Create new key pair

▼ Network settings [Info](#)

Edit

Network [Info](#)

vpc-0c3c371436c0dcd9d

Subnet [Info](#)

No preference (Default subnet in any availability zone)

Auto-assign public IP [Info](#)

Enable

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☒ Create security group

☐ Select existing security group

We'll create a new security group called 'launch-wizard-36' with the following rules:

☒ Allow SSH traffic from

Helps you connect to your instance

Anywhere

0.0.0.0/0

Key pair name

Key pairs allow you to connect to your instance securely.

jenkins

The name can include upto 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type



RSA

RSA encrypted private and public key pair



ED25519

ED25519 encrypted private and public key pair

Private key file format



.pem

For use with OpenSSH



.ppk

For use with PuTTY



When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** [Learn more](#)

Cancel

Create key pair

Once new key pair is created we select 1 and launch instance

Rules with source or 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

▼ Configure storage Info

Advanced

1x

8

GiB

gp2

Root volume (Not encrypted)

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage

Add new volume

The selected AMI contains more instance store volumes than the instance allows. Only the first 0 instance store volumes from the AMI will be accessible from the instance

0 x File systems

Edit

Number of instances Info

1

Software Image (AMI)

Canonical, Ubuntu, 20.04 LTS, ...[read more](#)
ami-0261755bbcb8c4a84

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier

Cancel

Launch instance

Click to connect to ssh

The screenshot shows the AWS Management Console 'Instances' page. A table lists two instances: 'NFSServer' and 'jenkins'. The 'jenkins' instance is selected, and its public IP address, 34.230.43.147, is highlighted in the 'Public IPv4 address' field. The 'Connect' button is also visible.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public
NFSServer	i-0a7991d28d2c2a498	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	ec2-34
jenkins	i-0fbaf7bb9588d59c0	Running	t2.micro	2/2 checks passed	No alarms	us-east-1a	ec2-34

Instance: i-0fbaf7bb9588d59c0 (jenkins)

Public IPv4 address: 34.230.43.147 Open address

Private IPv4 addresses: 172.31.82.151

JENKINS SERVER CONFIGURATION

To configure Jenkins, we can navigate to the documentation on their website and look at it to understand the platform of download that suits you. In this case we are using the Ubuntu Debian.

Open git bash on visual studio code or whichever console is convenient to use. We are using git bash here with Visual Studio Code

We rename the ip address as webserver as seen below and perform an update check and installing the default headless jdk as shown below.

```
oshor@Oshority MINGW64 ~/Downloads (master)
$ ssh -i "jenkins.pem" ubuntu@ec2-34-230-43-147.compute-1.amazonaws.com
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.15.0-1036-aws x86_64)

* Documentation:  https://help.ubuntu.com
* Management:    https://landscape.canonical.com
```

```
ubuntu@ip-172-31-82-151:~$ sudo hostname jenkins-server
ubuntu@ip-172-31-82-151:~$ bash
ubuntu@jenkins-server:~$ sudo apt update
sudo apt install default-jdk-headless
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Get:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease [114 kB]
Get:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease [108 kB]
```

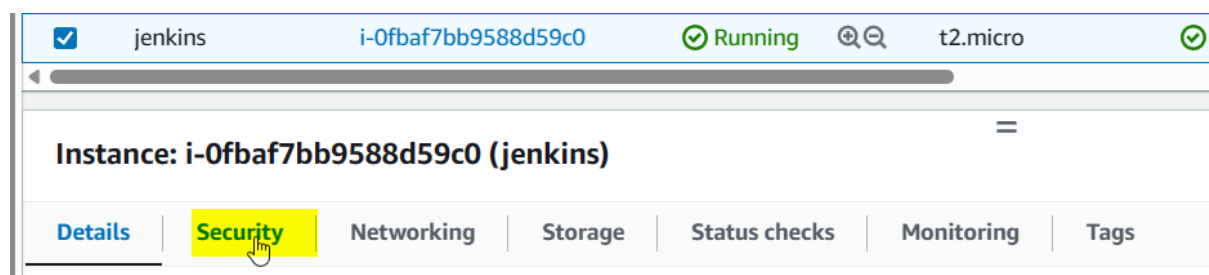
We then proceed to install Jenkins and to ensure easy installation add the key and perform the actions as seen below

```
ubuntu@jenkins-server:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease
Ign:4 https://pkg.jenkins.io/debian-stable binary/ InRelease
Get:5 https://pkg.jenkins.io/debian-stable binary/ Release [2044 B]
Get:6 https://pkg.jenkins.io/debian-stable binary/ Release.gpg [833 B]
Hit:7 http://security.ubuntu.com/ubuntu focal-security InRelease
Get:8 https://pkg.jenkins.io/debian-stable binary/ Packages [24.9 kB]
Fetched 27.8 kB in 1s (36.5 kB/s)
Reading package lists... Done
Building dependency tree
Reading state information... Done
76 packages can be upgraded. Run 'apt list --upgradable' to see them.
ubuntu@jenkins-server:~$ sudo apt-get install jenkins -y
Reading package lists... Done
Building dependency tree
ubuntu@jenkins-server:~$ wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key
add
OK
ubuntu@jenkins-server:~$ sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > \
> /etc/apt/sources.list.d/jenkins.list'
Command 'udo' not found, but can be installed with:
sudo apt install udo
ubuntu@jenkins-server:~$ sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > \
/etc/apt/sources.list.d/jenkins.list'
ubuntu@jenkins-server:~$ sudo apt update
Hit:1 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal InRelease
Hit:2 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:3 http://us-east-1.ec2.archive.ubuntu.com/ubuntu focal-backports InRelease
```

Once installed ,we can confirm if Jenkins was successfully installed .

```
ubuntu@jenkins-server:~$ sudo systemctl status jenkins
● jenkins.service - Jenkins Continuous Integration Server
   Loaded: loaded (/lib/systemd/system/jenkins.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2023-06-20 02:57:24 UTC; 3min 26s ago
     Main PID: 4573 (java)
       Tasks: 36 (limit: 1141)
      Memory: 301.5M
      CGroup: /system.slice/jenkins.service
              └─4573 /usr/bin/java -Djava.awt.headless=true -jar /usr/share/java/jenkins.war --webroot=
```

Please note that Jenkins runs on port 8080. Hence we would need to add the security group in the Jenkins server to ensure it successful launched on the web page

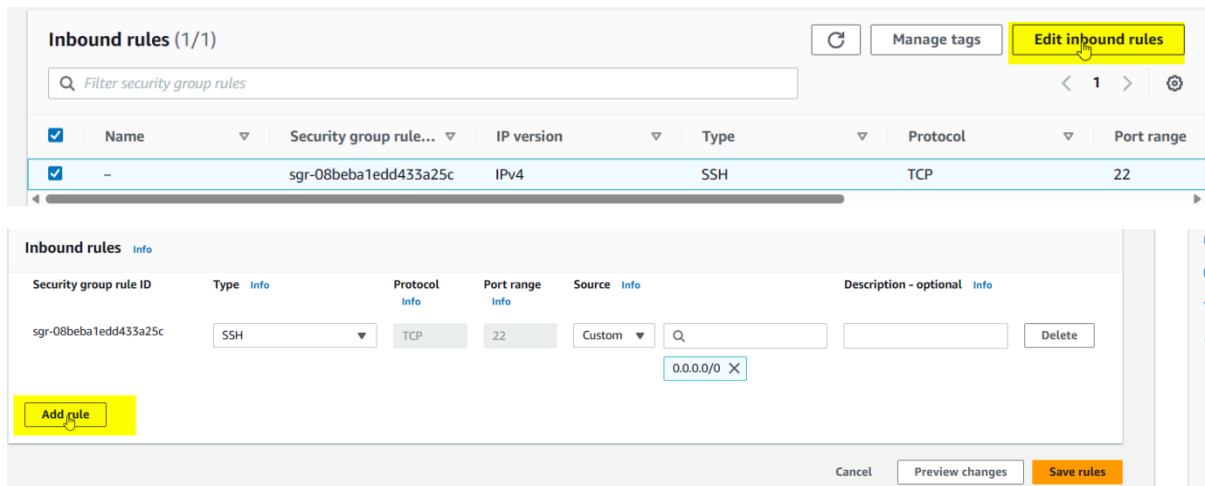


Security groups

 [sg-008eca2f5199c6c47 \(launch-wizard-36\)](#)

▼ Inbound rules

Edit the inbound rules and add the rule of port :8080 for Jenkins



Inbound rules (1/1)

Filter security group rules

Name	Security group rule...	IP version	Type	Protocol	Port range
-	sgr-08beba1edd433a25c	IPv4	SSH	TCP	22

Edit inbound rules

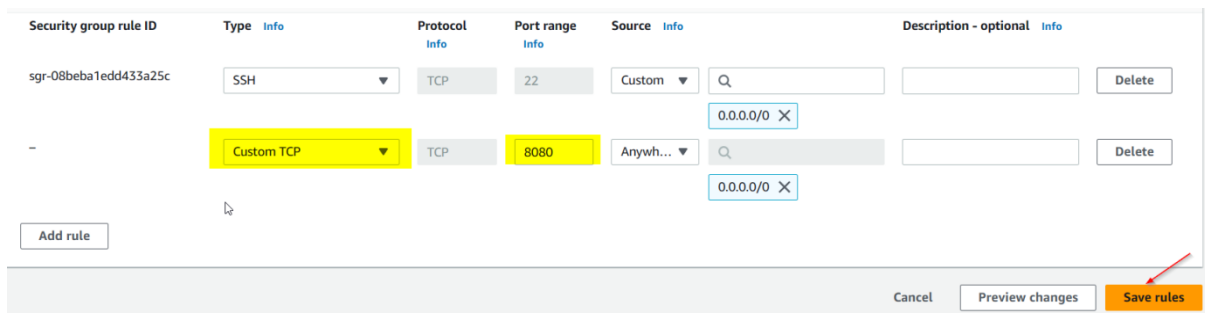
Inbound rules

Security group rule ID	Type	Protocol	Port range	Source	Description - optional
sgr-08beba1edd433a25c	SSH	TCP	22	Custom	

Add rule

Cancel Preview changes **Save rules**

Save the rules



Security group rule ID Type Protocol Port range Source Description - optional

sgr-08beba1edd433a25c	SSH	TCP	22	Custom	
-	Custom TCP	TCP	8080	Anywh...	

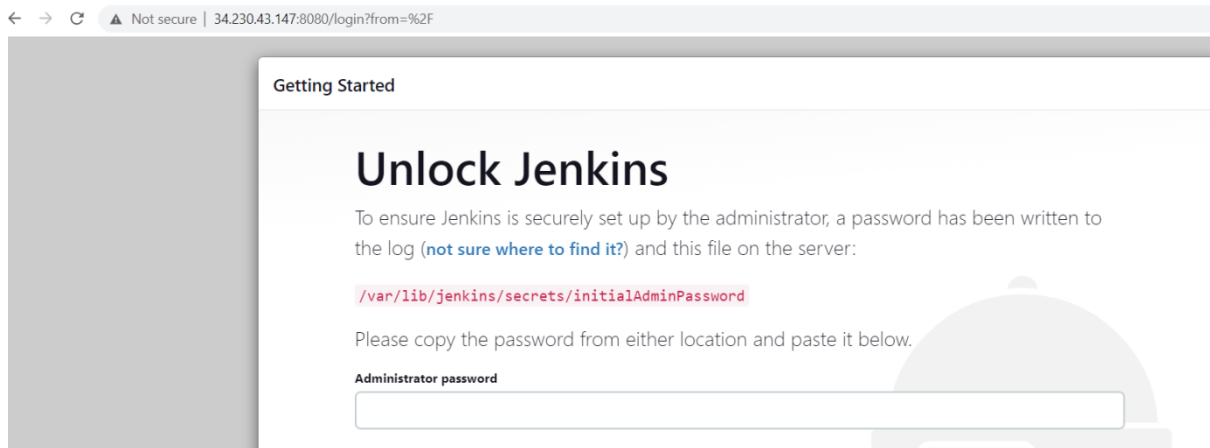
Add rule

Cancel Preview changes **Save rules**

✔ Inbound security group rules successfully modified on security group (sg-008eca2f5199c6c47 | launch-wizard-36)

► Details

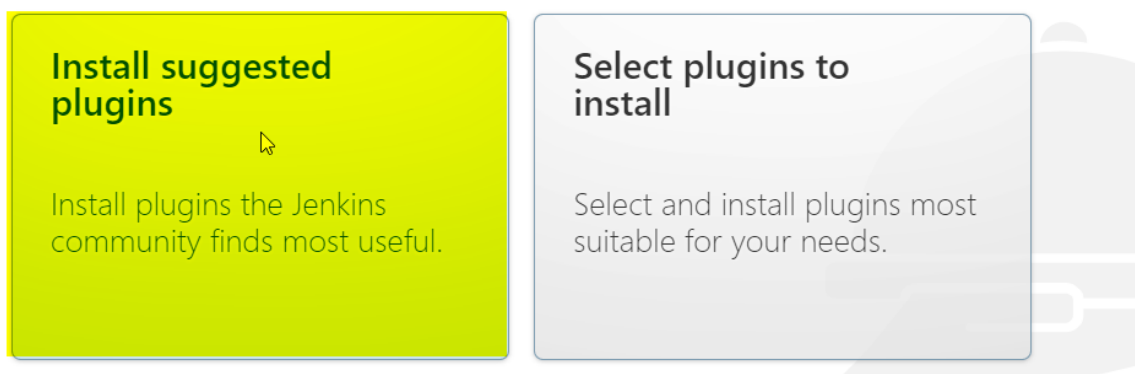
Proceed to launch the public ip address and your Jenkins would be displayed.



You can retrieve your admin password in your terminal and paste here to access as the admin. Once done you should see this page and click on Installed suggested plugins and have all suggested plugins installed as shown below.

Customize Jenkins

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



Getting Started

✓ Folders	✓ OWASP Markup Formatter	✓ Build Timeout	✓ Credentials Binding
✓ Timestampers	✓ Workspace Cleanup	✓ Ant	↻ Gradle
↻ Pipeline	↻ GitHub Branch Source	↻ Pipeline: GitHub Groovy Libraries	↻ Pipeline: Stage View
↻ Git	↻ SSH Build Agents	○ Matrix Authorization Strategy	↻ PAM Authentication
↻ LDAP	↻ Email Extension	✓ Mailer	

Once you fill in your details and have it successful, you get this page and Jenkins is ready to get used.

Instance Configuration

Jenkins URL:

http://34.230.43.147:8080/

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.

Jenkins is ready!

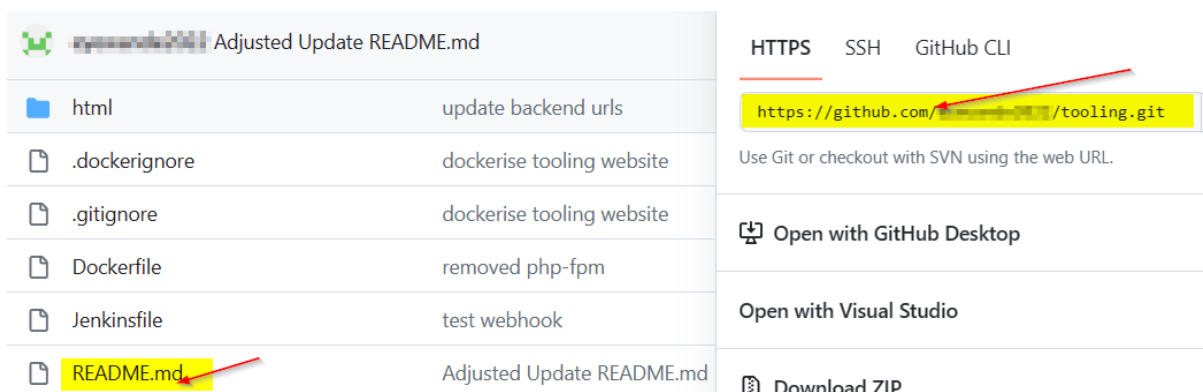
Your Jenkins setup is complete.

Start using Jenkins

JENKINS CONFIGURATION WITH GITHUB USING WEBHOOKS

Next steps would be to configure Jenkins to retrieve source codes from GitHub using Webhooks.

Firstly we create our first job on Jenkins and connect to a repository chosen to perform this actions .We are using a README.md file to perform this as well as the URL copied



The screenshot shows a GitHub repository interface. On the left, a file list includes 'html' (update backend urls), '.dockerignore' (dockerise tooling website), '.gitignore' (dockerise tooling website), 'Dockerfile' (removed php-fpm), 'Jenkinsfile' (test webhook), and 'README.md' (Adjusted Update README.md). The 'README.md' file is highlighted with a red arrow. On the right, a sidebar shows options: 'HTTPS' (selected), 'SSH', and 'GitHub CLI'. Below these, a yellow box contains the URL 'https://github.com/[redacted]/tooling.git', with a red arrow pointing to it. Other options include 'Use Git or checkout with SVN using the web URL.', 'Open with GitHub Desktop', 'Open with Visual Studio', and 'Download ZIP'.

File	Description
html	update backend urls
.dockerignore	dockerise tooling website
.gitignore	dockerise tooling website
Dockerfile	removed php-fpm
Jenkinsfile	test webhook
README.md	Adjusted Update README.md

HTTPS SSH GitHub CLI

[https://github.com/\[redacted\]/tooling.git](https://github.com/[redacted]/tooling.git)

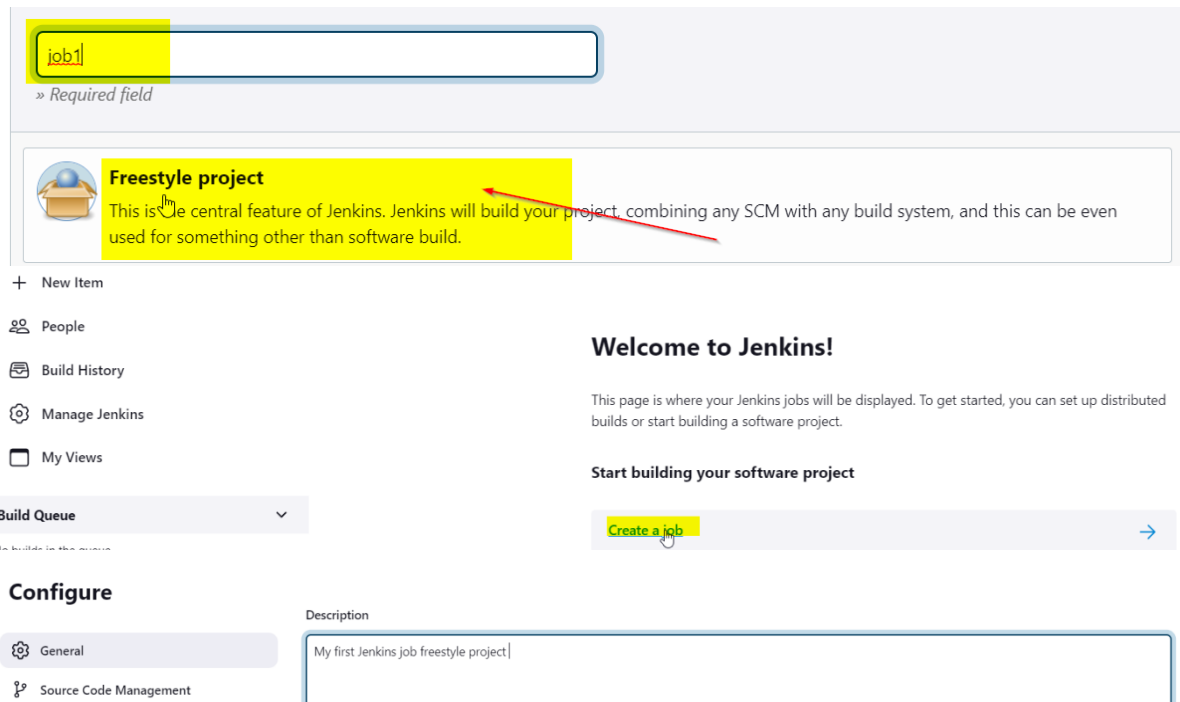
Use Git or checkout with SVN using the web URL.

Open with GitHub Desktop

Open with Visual Studio

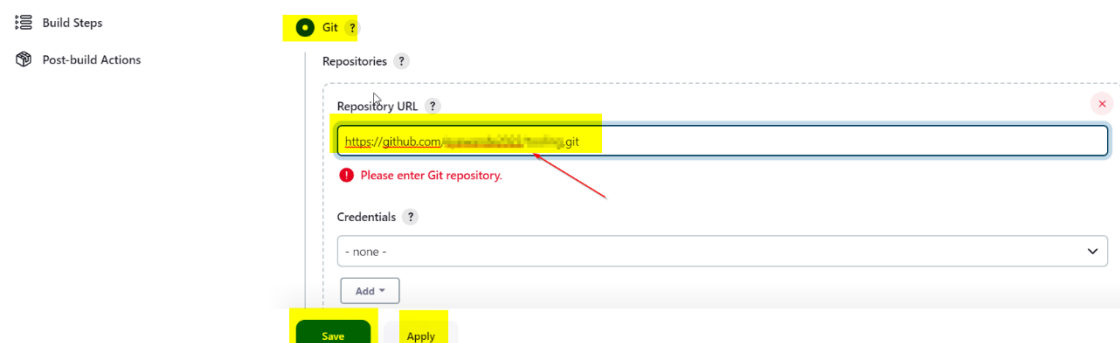
Download ZIP

Creating a job on Jenkins. You create a name and select “free project” and click ok



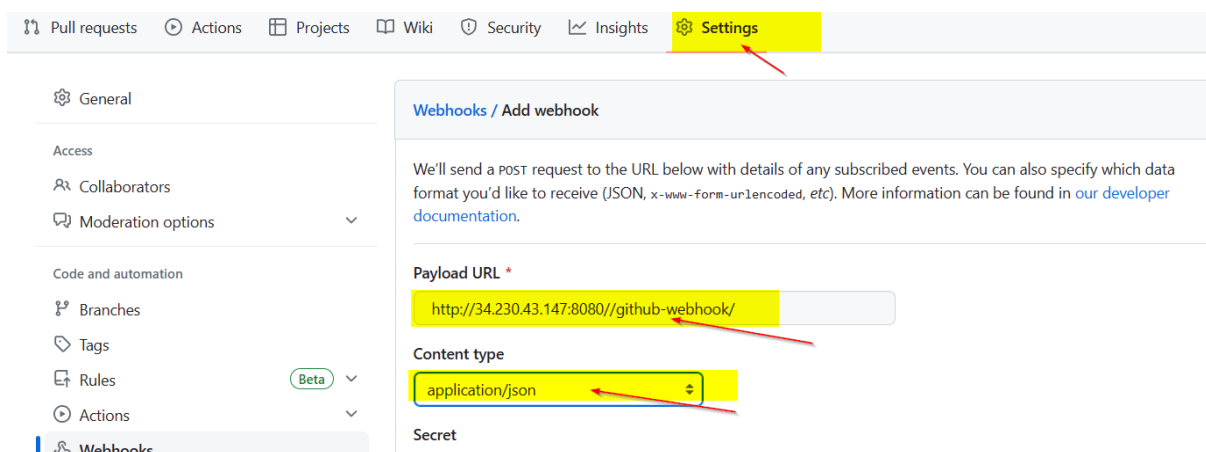
The image shows the Jenkins 'New Item' page. At the top, a text input field contains 'job1' and is highlighted with a yellow box. Below it, a description for the 'Freestyle project' is shown, also highlighted with a yellow box. A red arrow points from this description to the 'Create a job' button. The left sidebar contains links for 'New Item', 'People', 'Build History', 'Manage Jenkins', and 'My Views'. The main area has a 'Welcome to Jenkins!' message and a 'Start building your software project' section with a 'Create a job' button. Below this, the 'Configure' section is visible, with 'General' and 'Source Code Management' tabs. The 'Description' field contains 'My first Jenkins job freestyle project'.

You describe the job. Navigate to Source code management and click git and fill in the repository URL . Click on “Apply” and “Save”



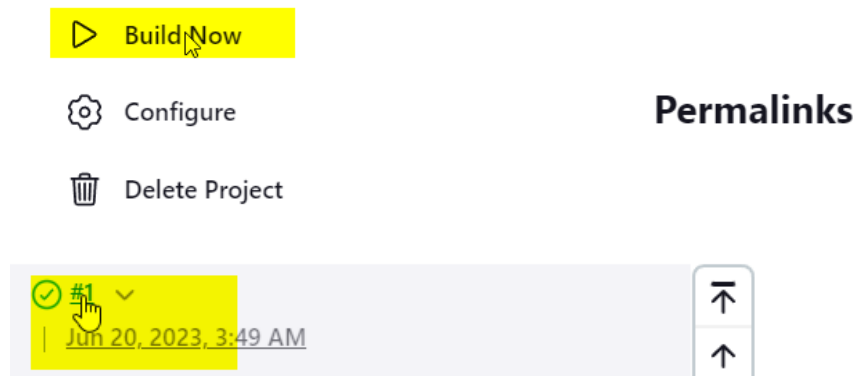
The image shows the 'Source Code Management' configuration page in Jenkins. The 'Git' option is selected and highlighted with a yellow box. Below it, the 'Repository URL' field is highlighted with a yellow box and contains the URL 'https://github.com:username:password@github.com:username/repo.git'. A red arrow points from this field to the 'Apply' button. The 'Credentials' dropdown is set to 'none'. At the bottom, there are 'Save' and 'Apply' buttons.

Navigate to your GitHub account and click on settings and add your webhook as shown below .

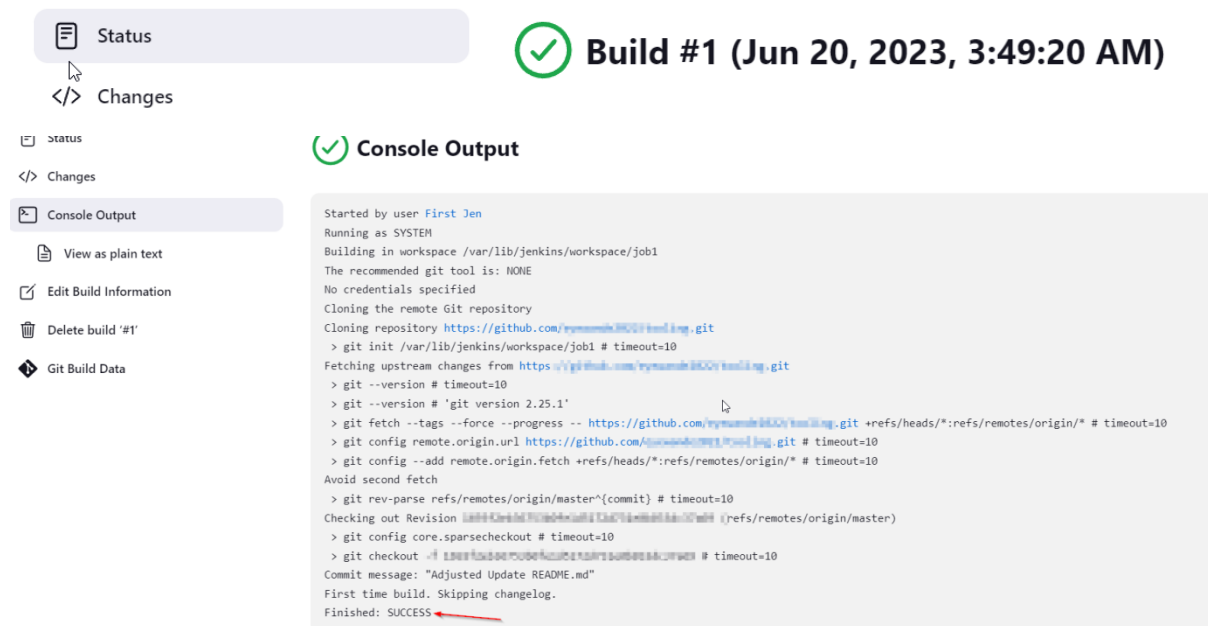


The image shows the GitHub 'Settings' page, specifically the 'Webhooks' section. The 'Settings' tab in the top navigation bar is highlighted with a yellow box. In the left sidebar, the 'Webhooks' option is selected. The main content area is titled 'Webhooks / Add webhook'. It contains a description of webhooks and a form to add a new one. The 'Payload URL' field is highlighted with a yellow box and contains the URL 'http://34.230.43.147:8080/github-webhook/'. The 'Content type' dropdown is also highlighted with a yellow box and set to 'application/json'. A red arrow points from the 'Payload URL' field to the 'Add webhook' button.

Once successfully added. We trigger the “Build Now” button and confirm if the build was successful



Navigate through the status and console output which gives you the details of the job and also states if it was a success as shown below



The console output illustrated how Jenkins created a workspace file with the help of the user admin called “job1”. If we navigate to this (/var/lib/jenkins/workspace) directory on our local machine and view the file content you would see the job1 created as shown below

```
ubuntu@jenkins-server:~$ cd /var/lib/jenkins/workspace
ubuntu@jenkins-server:/var/lib/jenkins/workspace$ ls
job1
ubuntu@jenkins-server:/var/lib/jenkins/workspace$
```

Please note that this is a manual trigger hence it doesn't run anything. To begin running ,we need to add 2 configurations in our project

To get this done we need to navigate to the “configure” and navigate to “Build Trigger” and select the GitHub webhook as shown below .

The image shows two screenshots of the Jenkins web interface. The top screenshot is the 'Console Output' view for a job named 'job1', showing a green checkmark and the text 'Console Output'. The bottom screenshot is the 'Configure' page for 'Project job1'. On the left sidebar, the 'Configure' button is highlighted in yellow with a red arrow pointing to it. In the main content area, under the 'Build Triggers' section, the 'GitHub hook trigger for GITScm polling' checkbox is checked and highlighted in yellow with a red arrow. Other options like 'Trigger builds remotely', 'Build after other projects are built', 'Build periodically', and 'Poll SCM' are unchecked. The 'Build Environment' section is partially visible at the bottom.

Dashboard > job1 > #1 > Console Output

Status

Console Output

Started by user First Jen

Dashboard > job1 >

Status

Project job1

My first Jenkins job freestyle project

Changes

Workspace

Build Now

Configure

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

Build Triggers

- ☐ Trigger builds remotely (e.g., from scripts) ?
- ☐ Build after other projects are built ?
- ☐ Build periodically ?
- ☒ GitHub hook trigger for GITScm polling ?
- ☐ Poll SCM ?

Build Environment

Apply and Save. Then proceed to try to automate using your Jenkins which is different from manually building it as we did previously .

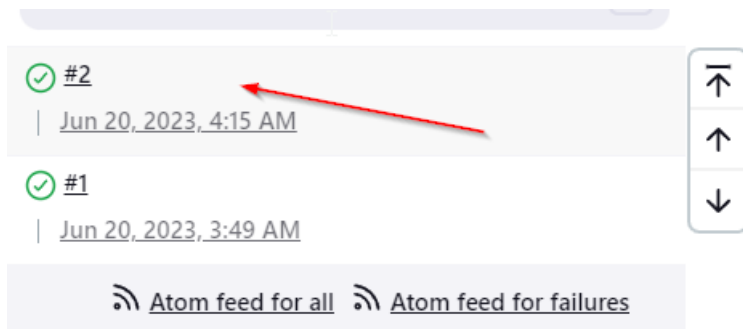
Navigate to the README.md file on GitHub and do the editing and click on commit changes



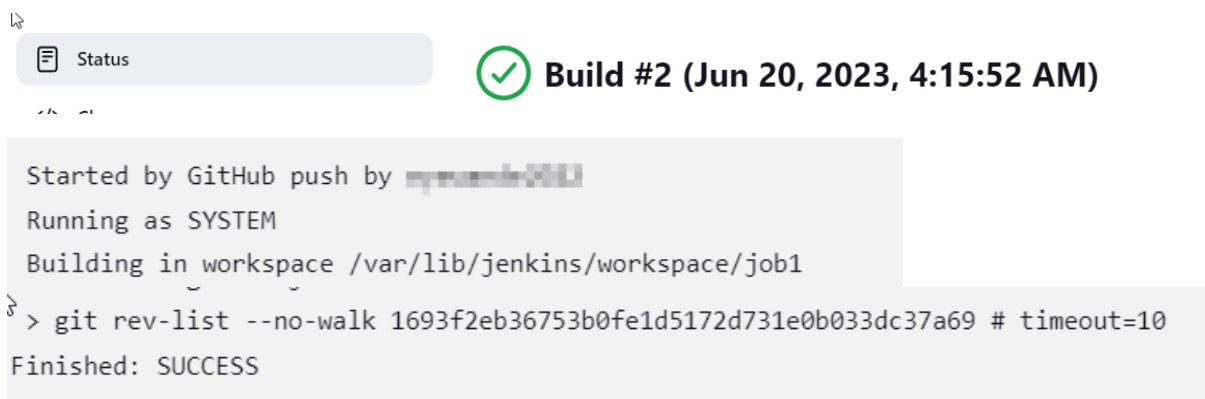
Ensure you also make changes on the commit message to always show clear difference between changes and commit on Jenkins.

A screenshot of the Jenkins "Commit changes" dialog box. The dialog has a title bar "Commit changes" with a close button. Below the title bar, there is a section "Commit message" with a text input field containing "Adjustment Notice README.md". A red arrow points to this input field. Below the "Commit message" section, there is a section "Extended description" with a text area containing the placeholder text "Add an optional extended description..". Below the "Extended description" section, there are two radio buttons: "Commit directly to the master branch" (which is selected) and "Create a new branch for this commit and start a pull request". Below the radio buttons, there is a link "Learn more about pull requests". At the bottom of the dialog, there are two buttons: "Cancel" and "Commit changes". The "Commit changes" button is highlighted in yellow, and a red arrow points to it.

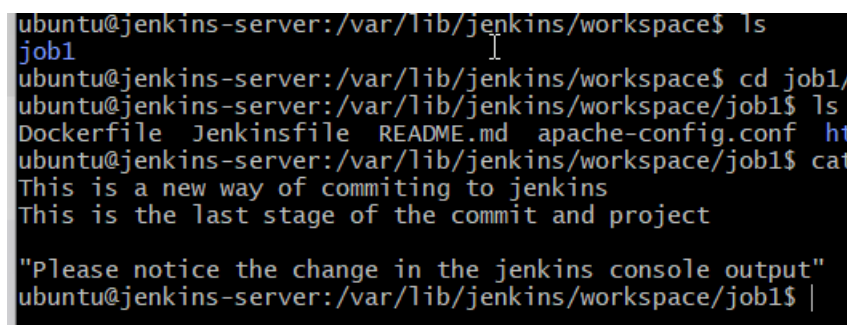
Once committed, Jenkins is triggered within seconds and hence we can see the second build



Click on the build2 to check the status and console output respectively.



On the terminal. Let us change directory to job1. It is evident that the README.md is located in the job1 folder and if we check its content, it is as expected and displayed below



As a DevOps engineer, we need to be able to set up a Jenkins server that would help in monitoring the changes we have on version control management software like GitHub. This would make it easy for monitoring the source codes the developers load on the git repository

POST-BUILD ACTIONS CONFIGURATION

Whenever Jenkins triggers source codes, it packages the files and they are called artefacts.

Navigate back to “configure” section in Jenkins and click on it and locate the post-Build actions as seen below

Configure

Delete Project

GitHub Hook Log

General

Source Code Management

Build Triggers

Build Environment

Build Steps

Post-build Actions

Permalinks

- [Last build \(#2\), 19 min ago](#)
- [Last stable build \(#2\), 19 min ago](#)
- [Last successful build \(#2\), 19 min ago](#)
- [Last completed build \(#2\), 19 min ago](#)

☐ Add timestamps to the Co

☐ Inspect build log for publi

☐ Terminate a build if it's stu

☐ With Ant ?

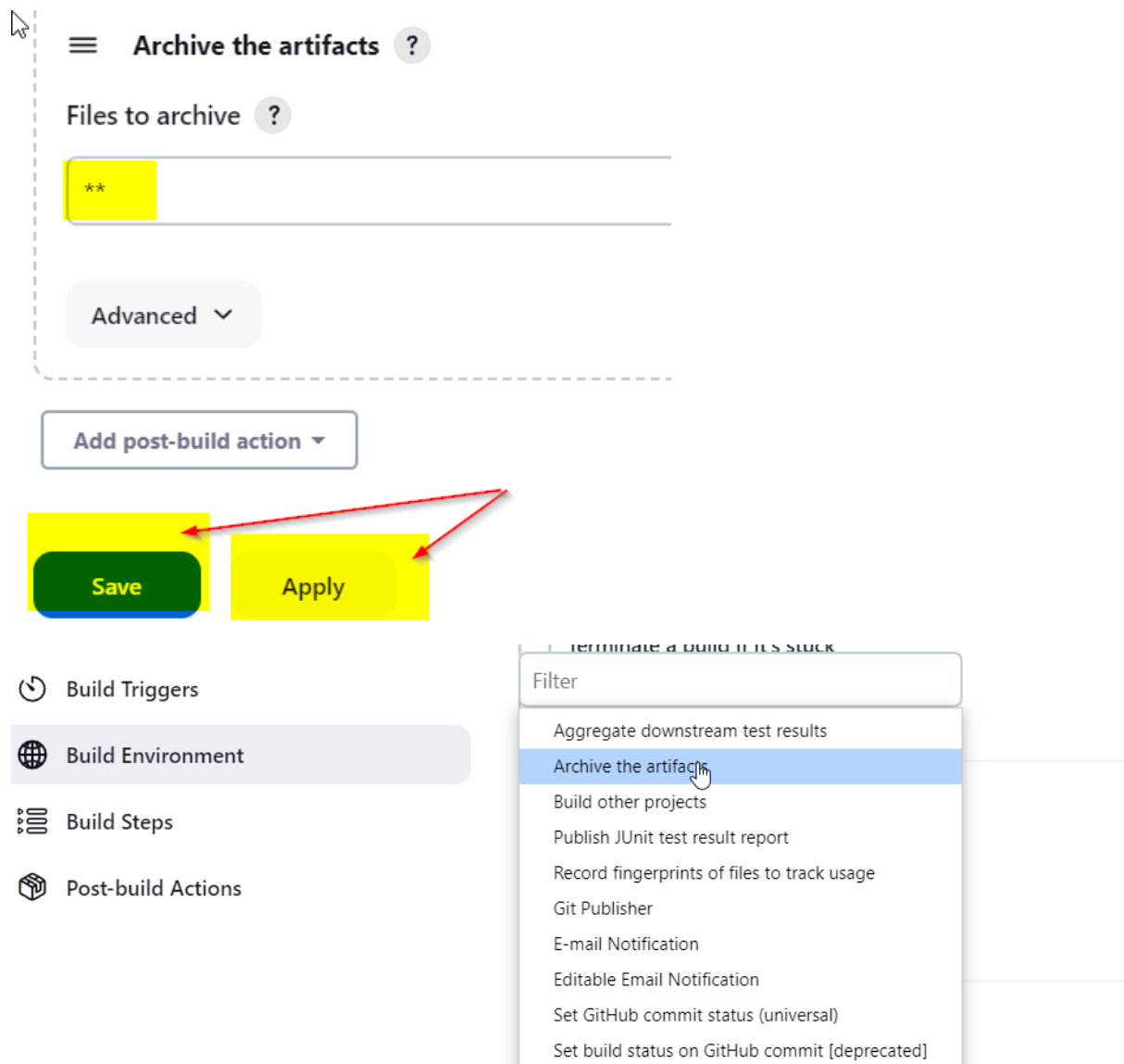
Build Steps

Add build step ▾

Post-build Actions

Add post-build action ▾

Select “archive the artifacts”, Type ** to make sure Jenkins archives every build artifacts. Click on “ Apply” and “save”.



Next step is to find the Jenkins build on the terminal and change directory into job1 and check its content, locate builds folder and access it

```
ubuntu@jenkins-server:/var/lib/jenkins/workspace/job1$ cd /var/lib/jenkins/jobs/job1
ubuntu@jenkins-server:/var/lib/jenkins/jobs/job1$ ls
builds config.xml github-polling.log nextBuildNumber
ubuntu@jenkins-server:/var/lib/jenkins/jobs/job1$ cd builds/
ubuntu@jenkins-server:/var/lib/jenkins/jobs/job1/builds$ ls
1 2 3 legacyIds permalinks
ubuntu@jenkins-server:/var/lib/jenkins/jobs/job1/builds$ ls -la
total 24
drwxr-xr-x 5 jenkins jenkins 4096 Jun 20 04:41 .
drwxr-xr-x 3 jenkins jenkins 4096 Jun 20 04:41 ..
drwxr-xr-x 2 jenkins jenkins 4096 Jun 20 03:49 1
drwxr-xr-x 2 jenkins jenkins 4096 Jun 20 04:15 2
drwxr-xr-x 3 jenkins jenkins 4096 Jun 20 04:41 3
-rw-r--r-- 1 jenkins jenkins  0 Jun 20 03:37 legacyIds
-rw-r--r-- 1 jenkins jenkins 126 Jun 20 04:41 permalinks
ubuntu@jenkins-server:/var/lib/jenkins/jobs/job1/builds$
```

Navigating to “build 3” folders, we can see the archived files we shown below

```
1 2 3 legacyIds  permalinks
ubuntu@jenkins-server:/var/lib/jenkins/jobs/job1/builds$ cd 3/
ubuntu@jenkins-server:/var/lib/jenkins/jobs/job1/builds/3$ ls
archive  build.xml  changelog.xml  log  polling.log
ubuntu@jenkins-server:/var/lib/jenkins/jobs/job1/builds/3$ cd archive
ubuntu@jenkins-server:/var/lib/jenkins/jobs/job1/builds/3/archive$ ls
Dockerfile  Jenkinsfile  README.md  apache-config.conf  html  start-apache  tooling-db.sql
ubuntu@jenkins-server:/var/lib/jenkins/jobs/job1/builds/3/archive$
```

CONFIGURE JENKINS TO COPY FILES TO THE NFS SERVER VIA SSH

We have the artifacts saved in our Jenkins server ,we have to copy them into our NFS server and have to save it to our /mnt/apps directory


Navigate to the nfs server created and

<input checked="" type="checkbox"/>	NFSServer	i-0a7991d28d2c2a498	✔ Running		t2.micro	✔ 2/2 checks passed	No alarms	+	us-east-1a	ec2
<input type="checkbox"/>	jenkins	i-0fbaf7bb9588d59c0	✔ Running		t2.micro	✔ 2/2 checks passed	No alarms	+	us-east-1a	ec2

Check the content of /mnt/apps directory.

```
[ec2-user@nfs-server ~]$ ls /mnt
apps  logs  opt
[ec2-user@nfs-server ~]$ ls /mnt/apps
html
[ec2-user@nfs-server ~]$ cd /mnt/apps
```

We should install the Publish Over SSH plugin on Jenkins.
Navigate to “Manage Jenkins”.

 Manage Jenkins

☐ My Views

Build Queue

No builds in the queue.

S	W	Name ↓	Last Success
✓	☀	job1	29 min #3

Icon: S M L

Icon legend

Select Plugins, Click on available plugins and type on the search box “Publish Over SSH” and make sure you select and install without restart

System Configuration

System

Configure global settings and paths.

Tools

Configure tools, their locations and automatic installers.

Plugins

Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

Available plugins

Installed plugins

Advanced settings

Search plugin updates

Name ↓

Released

Installed

publish over ssh

Install

Name ↓

Released

Publish Over SSH 1.24

Artifact Uploaders Build Tools

Send build artifacts over SSH

1 yr 3 mo ago

Install without restart

Download now and install after restart

Update information obtained: 2 hr 14 min ago

Check now

Download progress

Preparation

• Checking internet connectivity

• Checking update center connectivity

• Success

Ionicons API

Success

Folders

Success

bouncycastle API

Success

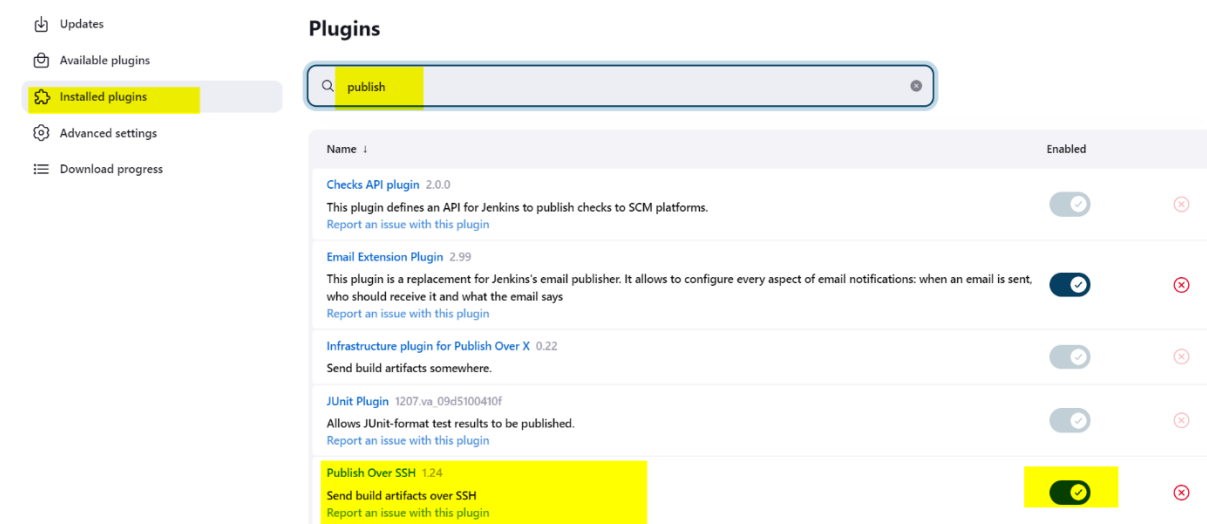
Instance Identity

Success

JavaBeans Activation Framework (JAF) API

Success

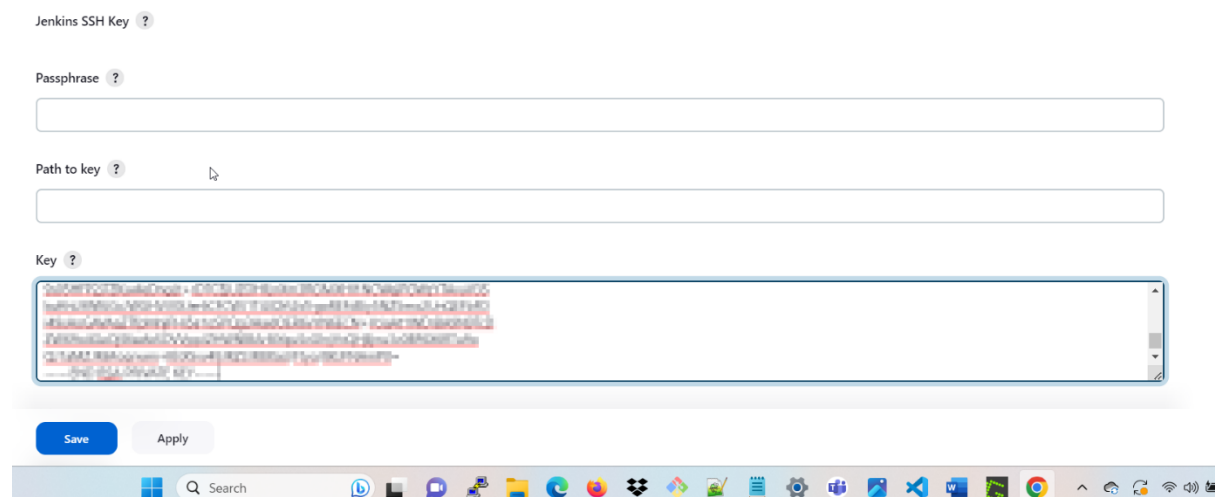
Successfully installed. Check Installed plugin and it can be seen it is correctly installed.



The screenshot shows the Jenkins 'Plugins' page. On the left, a sidebar contains links: 'Updates', 'Available plugins', 'Installed plugins' (highlighted), 'Advanced settings', and 'Download progress'. The main area is titled 'Plugins' and features a search bar with the text 'publish'. Below the search bar is a table of installed plugins. The table has columns for 'Name' and 'Enabled'. The 'Publish Over SSH' plugin is highlighted in yellow and is shown as enabled. Other plugins listed include 'Checks API plugin', 'Email Extension Plugin', 'Infrastructure plugin for Publish Over X', and 'JUnit Plugin'.

Name	Enabled
Checks API plugin 2.0.0 This plugin defines an API for Jenkins to publish checks to SCM platforms. Report an issue with this plugin	<input checked="" type="checkbox"/>
Email Extension Plugin 2.99 This plugin is a replacement for Jenkins's email publisher. It allows to configure every aspect of email notifications: when an email is sent, who should receive it and what the email says Report an issue with this plugin	<input checked="" type="checkbox"/>
Infrastructure plugin for Publish Over X 0.22 Send build artifacts somewhere.	<input checked="" type="checkbox"/>
JUnit Plugin 1207.va.09d5100410f Allows JUnit-format test results to be published. Report an issue with this plugin	<input checked="" type="checkbox"/>
Publish Over SSH 1.2.4 Send build artifacts over SSH Report an issue with this plugin	<input checked="" type="checkbox"/>

Navigate back to Manage Jenkins and click on Systems and navigate to Publish over SSH. Provide your private key



The screenshot shows the 'Publish over SSH' configuration page in Jenkins. It has a title 'Jenkins SSH Key' with a help icon. Below the title are four input fields: 'Passphrase', 'Path to key', and 'Key'. The 'Key' field contains a large block of redacted text. At the bottom of the form are 'Save' and 'Apply' buttons. The Windows taskbar is visible at the bottom of the screen.

Jenkins SSH Key ?

Passphrase ?

Path to key ?

Key ?

Save Apply

Click on Add SSH and fill in the server name, private ip address ,terminal username and remote directory

SSH Servers

Add

Advanced ▾

Save Apply

▼ Instance summary [Info](#)

Instance ID i-0a7991d28d2c2a498 (NFSServer)	Public IPv4 address 34.201.249.221 open address	Private IPv4 addresses 172.31.85.81
IPv6 address	Instance state -	Public IPv4 DNS -

Then we proceed to test configuration and it shows success

ec2-user

Remote Directory ?
/mnt/apps

Advanced ▾

Success **Test Configuration**

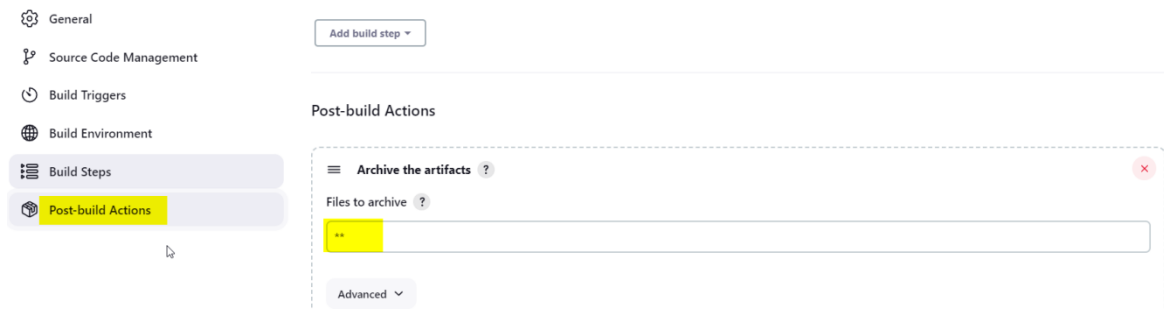
Add

Success **Test Configuration**

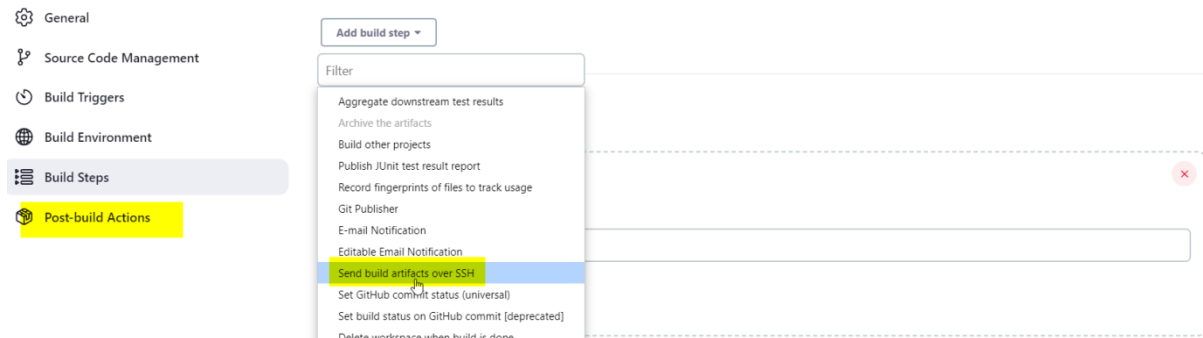
Click on “Apply” and ”Save ” and ensure that port :22 is open in the nfs server .

We are to configure it to send all files produced by the build into our previously defined remote directory .

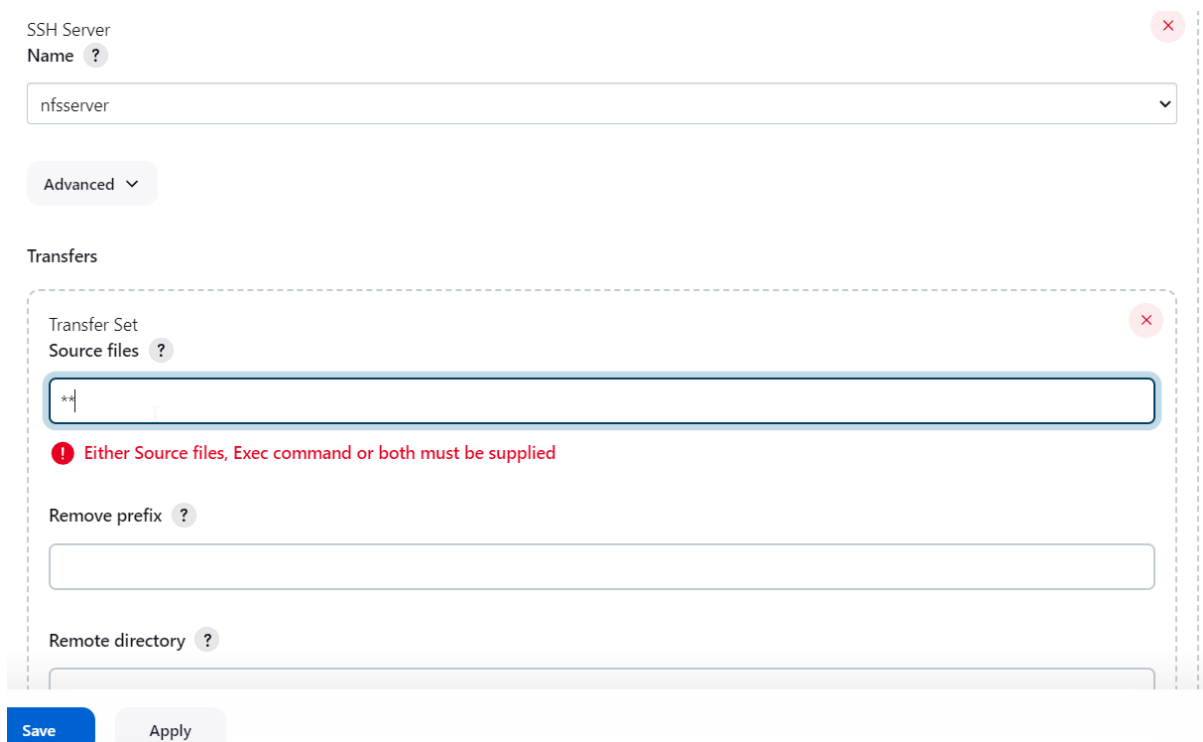
Navigate back to the Jenkins dashboard, go to configure and navigate to the post-build actions



Add another post build action. “send build actions over SSH



Type in ** in the source file edit box and “Apply” and “Save “



Now Jenkins is listening now. We navigate back to git hub and make a change.

Commit changes

Commit message

Conclusion | README.md

Extended description

Add an optional extended description..

☒ Commit directly to the master branch

☐ Create a new branch for this commit and start a pull request

[Learn more about pull requests](#)

Cancel

Commit changes

tooling / README.md in master

Cancel changes

Commit changes...

Edit Preview

Spaces 2 Soft wrap

1 This is a new way of committing to jenkins

2 This is the last stage of the commit and project

3

4 "Please notice the change in the jenkins console output"

5 Great jenkins job

6

7 FINAL CONCLUSION APPROACHING

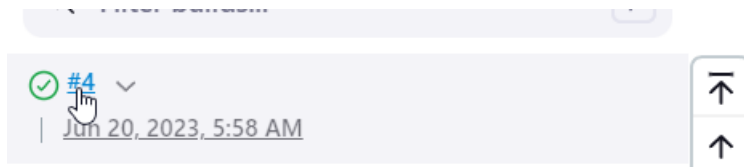
Once changes is committed and we check on Jenkins its pending and then successful .

#4

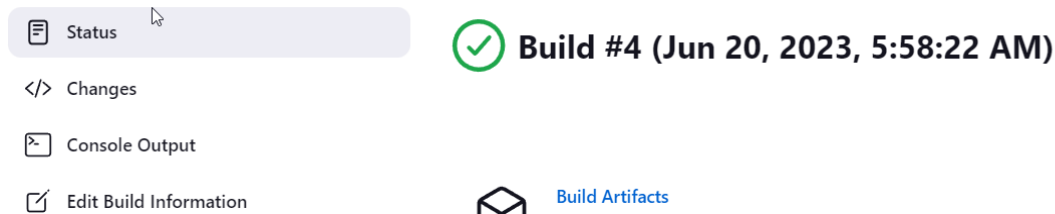
(pending—In the quiet period. Expires in 1.5 sec)

#3

Jun 20, 2023, 4:41 AM



Clicking the link, we can check the status as successful.



Navigate back to the terminal and check the files in /mnt/apps as shown below, we can see all the files are there .

```
[ec2-user@ip-172-31-85-81 apps]$ sudo hostname nfs-server
[ec2-user@ip-172-31-85-81 apps]$ bash
[ec2-user@nfs-server apps]$ ls
html mnt
[ec2-user@nfs-server apps]$ ls -la /mnt/apps
total 36
drwxr-xr-x. 4 ec2-user ec2-user 172 Jun 20 05:58 .
drwxr-xr-x. 5 root     root     41 Jun 12 16:56 ..
-rw-r--r--. 1 ec2-user ec2-user 332 Jun 20 05:58 apache-config.conf
-rw-r--r--. 1 ec2-user ec2-user 313 Jun 20 05:58 Dockerfile
-rw-r--r--. 1 ec2-user ec2-user  47 Jun 20 05:58 .dockerignore
drwxr-xr-x. 3 ec2-user ec2-user 4096 Jun 15 21:49 html
-rw-r--r--. 1 ec2-user ec2-user 4202 Jun 20 05:58 Jenkinsfile
drwxr-xr-x. 3 ec2-user ec2-user  18 Jun 19 20:25 mnt
-rw-r--r--. 1 ec2-user ec2-user  201 Jun 20 05:58 README.md
-rw-r--r--. 1 ec2-user ec2-user  163 Jun 20 05:58 start-apache
-rw-r--r--. 1 ec2-user ec2-user 1674 Jun 20 05:58 tooling-db.sql
[ec2-user@nfs-server apps]$
```

We have been able to move the files from Jenkins artifacts to the NFS server.

```
-rw-r--r--. 1 ec2-user ec2-user 1674 Jun 20 05:58 tooling-db.sql
[ec2-user@nfs-server apps]$ ls
apache-config.conf Dockerfile html Jenkinsfile mnt README.md start-apache tooling-db.sql
[ec2-user@nfs-server apps]$
```

This is exactly what we want to achieve.

Checking the README.md file we can see the files on there.

```
[ec2-user@nfs-server apps]$ cat README.md
This is a new way of committing to jenkins
This is the last stage of the commit and project

"Please notice the change in the jenkins console output"
Great jenkins job

FINAL CONCLUSION APPROACHING
[ec2-user@nfs-server apps]$
```

We have been able to successfully implement the Jenkins server.