

TOOLING WEBSITE DEPLOYMENT AUTOMATION WITH CONTINUOUS INTEGRATION.

INTRODUCTION TO JENKINS

TASK

In this project, we will enhance the architecture prepared in Project 8 by adding a Jenkins server, configure a job to automatically deploy source codes changes from Git to NFS server.

1. Infrastructure: **AWS**
2. Webserver Linux: **Red hat Enterprise Linux 8**
3. Storage Server: **Red Hat Enterprise Linux + NFS Server**
4. Code Repository: **GitHub**
5. Automation Server: **Jenkins Server (Linux Ubuntu 20.04)**

STEP 1.

CREATE AND CONFIGURE JENKINS SERVER

We will create another AWS EC2 instance on ubuntu server 20.04 LTS, name it "Jenkins".

Connect to a terminal and update.

```
$ sudo apt update -y
```

We will install JDK (Java development kit) because Jenkins is a java-based application.

```
$ sudo apt install default-jdk-headless
```

The command above installs default OpenJDK package without any graphical user interface (headless version) on a Debian-based system.

Install Jenkins

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

This is used to download the Jenkins repository key and add it to the apt keyring on a Debian-based system.

```
$ sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > \  
/etc/apt/sources.list.d/jenkins.list'
```

```

done.
Processing triggers for libc-bin (2.31-0ubuntu9.9) ...
Processing triggers for man-db (2.9.1-1) ...
Processing triggers for ca-certificates (20211016ubuntu0.20.04.1) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
done.
ubuntu@ip-172-31-37-181:~$ sudo wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key | sudo apt-key add -
OK
ubuntu@ip-172-31-37-181:~$ sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > \
> /etc/apt/sources.list.d/jenkins.list'
ubuntu@ip-172-31-37-181:~$

```

```

$ curl -fsSL https://pkg.jenkins.io/debian-stable/jenkins.io-2023.key | sudo tee \
  /usr/share/keyrings/jenkins-keyring.asc > /dev/null
$ echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
  https://pkg.jenkins.io/debian-stable binary/ | sudo tee \
  /etc/apt/sources.list.d/jenkins.list > /dev/null
$ sudo apt-get update
$ sudo apt-get install fontconfig openjdk-11-jre
$ sudo apt-get install Jenkins

```

check that Jenkins is installed correctly and running.

```
$ 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 Wed 2023-06-21 19:52:00 UTC; 2min 41s ago
     Main PID: 18452 (java)
       Tasks: 36 (limit: 1141)
      Memory: 329.4M
      CGroup: /system.slice/jenkins.service
              └─18452 /usr/bin/java -Djava.net.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Jun 21 19:52:00 ip-172-31-37-181 jenkins[18452]: 7f102d7982a0446fae1b3be7693e51e2
Jun 21 19:52:00 ip-172-31-37-181 jenkins[18452]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Jun 21 19:52:00 ip-172-31-37-181 jenkins[18452]: *****
Jun 21 19:52:00 ip-172-31-37-181 jenkins[18452]: *****
Jun 21 19:52:00 ip-172-31-37-181 jenkins[18452]: 2023-06-21 19:52:00.400+0000 [id=20] INFO Jenkins.Init@SectorRunner$1onAttained: Completed initialization
Jun 21 19:52:00 ip-172-31-37-181 jenkins[18452]: 2023-06-21 19:52:00.405+0000 [id=22] INFO hudson.lifecycle.Lifecycle$onReady: Jenkins is fully up and running
Jun 21 19:52:00 ip-172-31-37-181 systemd[3]: Started Jenkins Continuous Integration Server.
Jun 21 19:52:00 ip-172-31-37-181 jenkins[18452]: 2023-06-21 19:52:00.947+0000 [id=44] INFO h.w.DownloadService$Downloadable$load: Obtained the updated data file for hudson.tasks.Maven.MavenIn
Jun 21 19:52:00 ip-172-31-37-181 jenkins[18452]: 2023-06-21 19:52:00.947+0000 [id=44] INFO hudson.util.Retrier$onStart: Performed the action check updates server successfully at the attempt #1

```

```

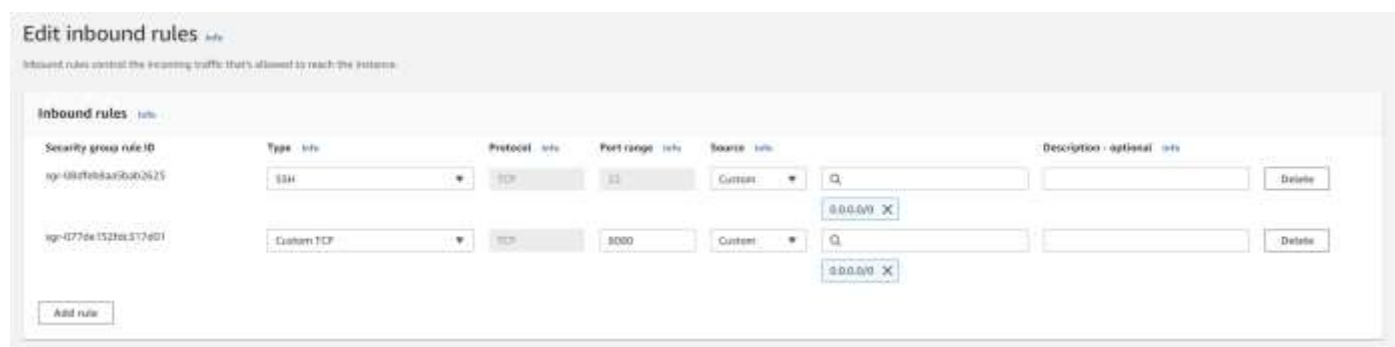
Tasks: 36 (limit: 1141)
Memory: 329.4M
CGroup: /system.slice/jenkins.service
        └─18452 /usr/bin/java -Djava.net.headless=true -jar /usr/share/java/jenkins.war --webroot=/var/cache/jenkins/war --httpPort=8080

Jun 21 19:52:00 ip-172-31-37-181 jenkins[18452]: 7f102d7982a0446fae1b3be7693e51e2
Jun 21 19:52:00 ip-172-31-37-181 jenkins[18452]: This may also be found at: /var/lib/jenkins/secrets/initialAdminPassword
Jun 21 19:52:00 ip-172-31-37-181 jenkins[18452]: *****
Jun 21 19:52:00 ip-172-31-37-181 jenkins[18452]: *****
Jun 21 19:52:00 ip-172-31-37-181 jenkins[18452]: 2023-06-21 19:52:00.400+0000 [id=20] INFO Jenkins.Init@SectorRunner$1onAttained: Completed initialization

```

Notice the Jenkins password highlighted which can also be found in the /var/lib/jenkins/secrets/initialAdminPassword directory

By default, Jenkins server listens on port 8080. Hence, we will need to open up port 8080 in our security group in-bound rules.



Perform initial Jenkins setup.

From your browser access <http://<Jenkins-Server-Public-IP-Address-or-Public-DNS-Name>:8080>

\$ **sudo cat /var/lib/jenkins/secrets/initialAdminPassword** to retrieve password.

Input default admin password, then you will be asked which plugins to install – choose suggested plugins.

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

1

input password that can be found in /var/lib/Jenkins/secrets/initialAdminPassword

Getting Started

Customize Jenkins

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

2

Install suggested plugins

Install plugins the Jenkins

Select plugins to install

Select and install plugins most

Getting Started

Create First Admin User

Username

#username

Password

Confirm password

Full name

#fullname

E-mail address

admin@jenkins.com

input your credentials, save and continue, and finish

3

Continue

Jenkins 2.421.1

Skip and continue as admin

Save and Continue

Jenkins is ready!

Your Jenkins setup is complete.

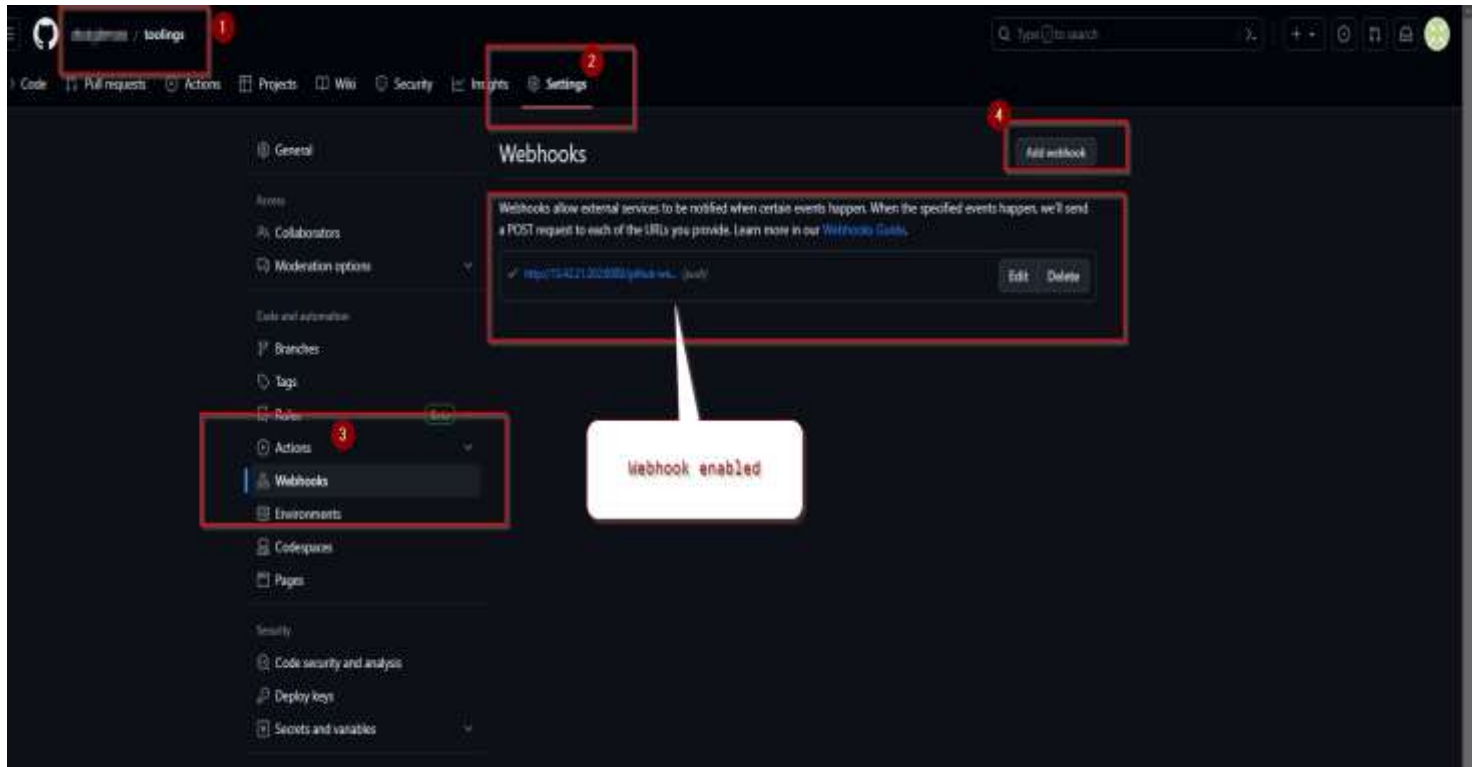
Start using Jenkins

STEP 2.

CONFIGURE JENKINS TO RETRIEVE SOURCE CODES FROM GITHUB USING WEBHOOKS

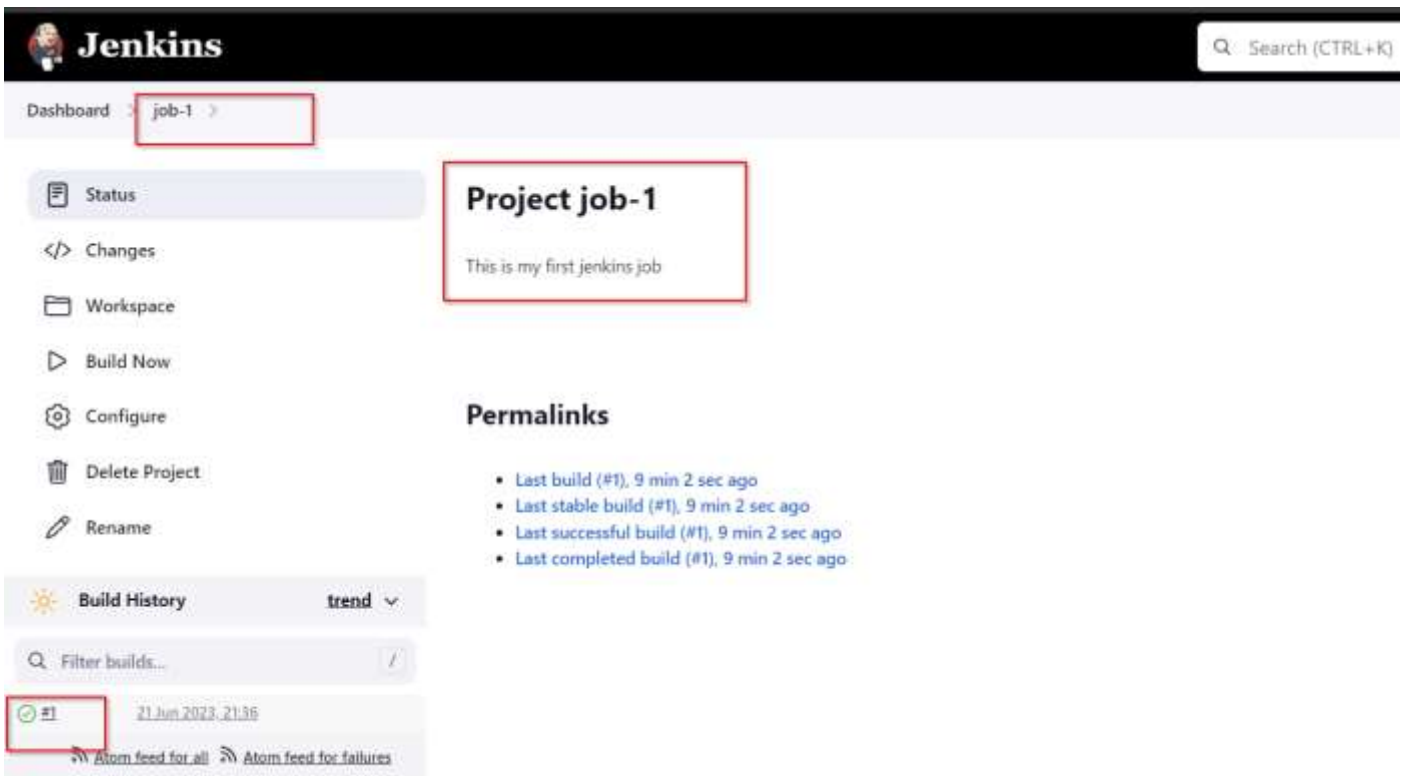
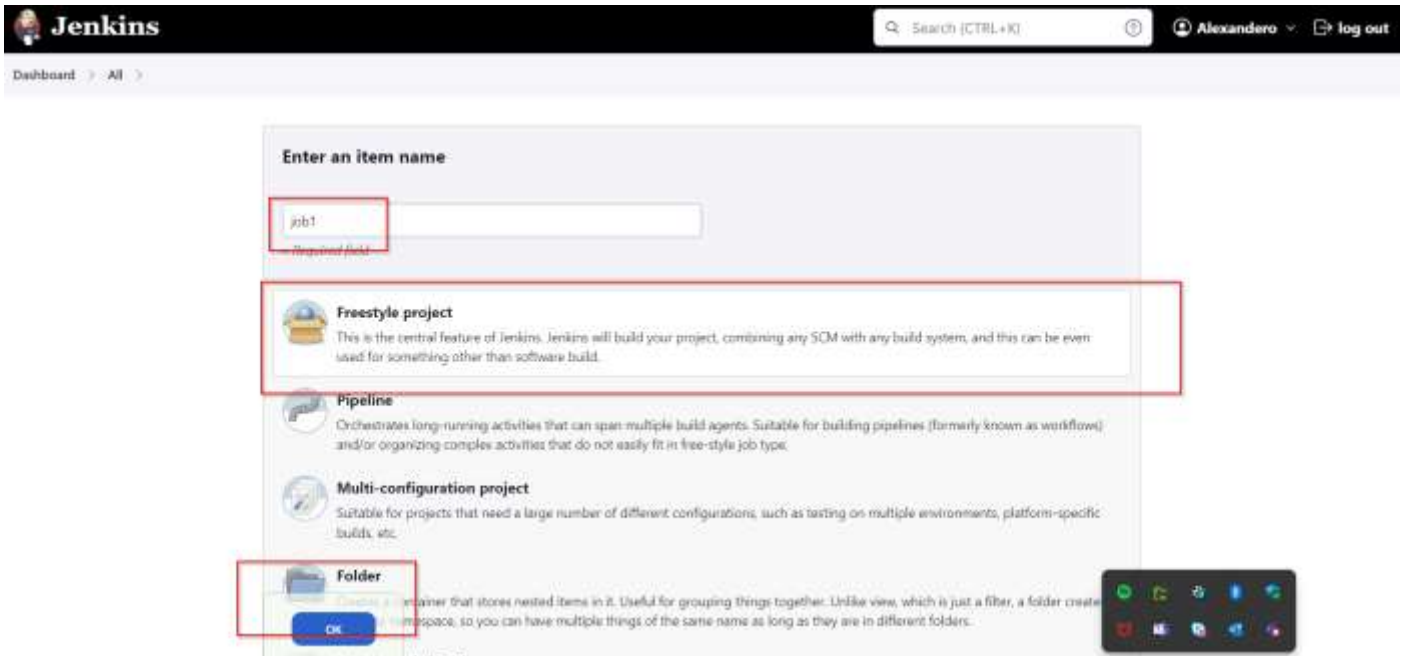
In this part, you will learn how to configure a simple Jenkins job/project. This job will be triggered by GitHub webhooks and will execute a 'build' task to retrieve codes from GitHub and store it locally on Jenkins server.

1. Enable webhooks in your GitHub repository settings.



Go to Jenkins web console, click "New Item" and create a "Freestyle project".

Connect your GitHub repository, you will need to provide its URL, you can copy from the repository itself.



Open the build and check in "Console Output" if it has run successfully.

Console Output

```

Started by user Alexandero
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/job-1
The recommended git tool is: NONE
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/obaigbenaa/toolings.git
> git init /var/lib/jenkins/workspace/job-1 # timeout=10
Fetching upstream changes from https://github.com/obaigbenaa/toolings.git
> git --version # timeout=10
> git --version # 'git version 2.25.1'
> git fetch --tags --force --progress -- https://github.com/obaigbenaa/toolings.git +refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/obaigbenaa/toolings.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/master^{commit} # timeout=10
Checking out Revision 2603d2f2261310c59ff6d029f43631a8e6b6dae (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 2603d2f2261310c59ff6d029f43631a8e6b6dae # timeout=10
Commit message: "new update"
First time build. Skipping changelog.
Finished: SUCCESS

```

Check for job-1 on the terminal in the directory below

\$ cd /var/lib/Jenkins/workspace

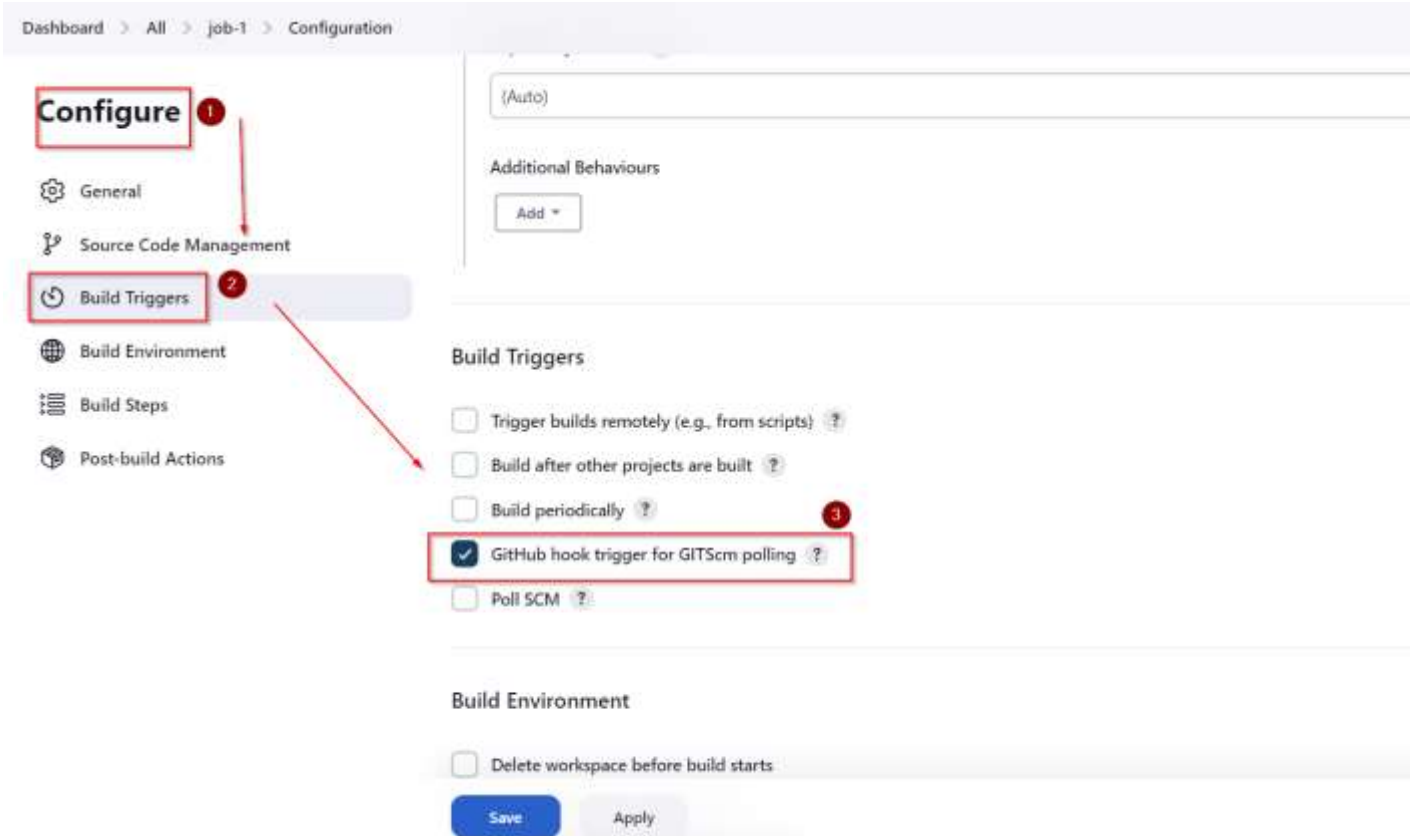
```

ubuntu@ip-172-31-37-181:~$ cd /var/lib/jenkins/workspace/job-1
ubuntu@ip-172-31-37-181:/var/lib/jenkins/workspace/job-1$ ls
Dockerfile Jenkinsfile README.md apache-config.conf html start-apache tooling-db.sql
ubuntu@ip-172-31-37-181:/var/lib/jenkins/workspace/job-1$ cat README.md
this is a readme file adjusted
ubuntu@ip-172-31-37-181:/var/lib/jenkins/workspace/job-1$ cd ..
ubuntu@ip-172-31-37-181:/var/lib/jenkins/workspace$ ls
job-1
ubuntu@ip-172-31-37-181:/var/lib/jenkins/workspace$

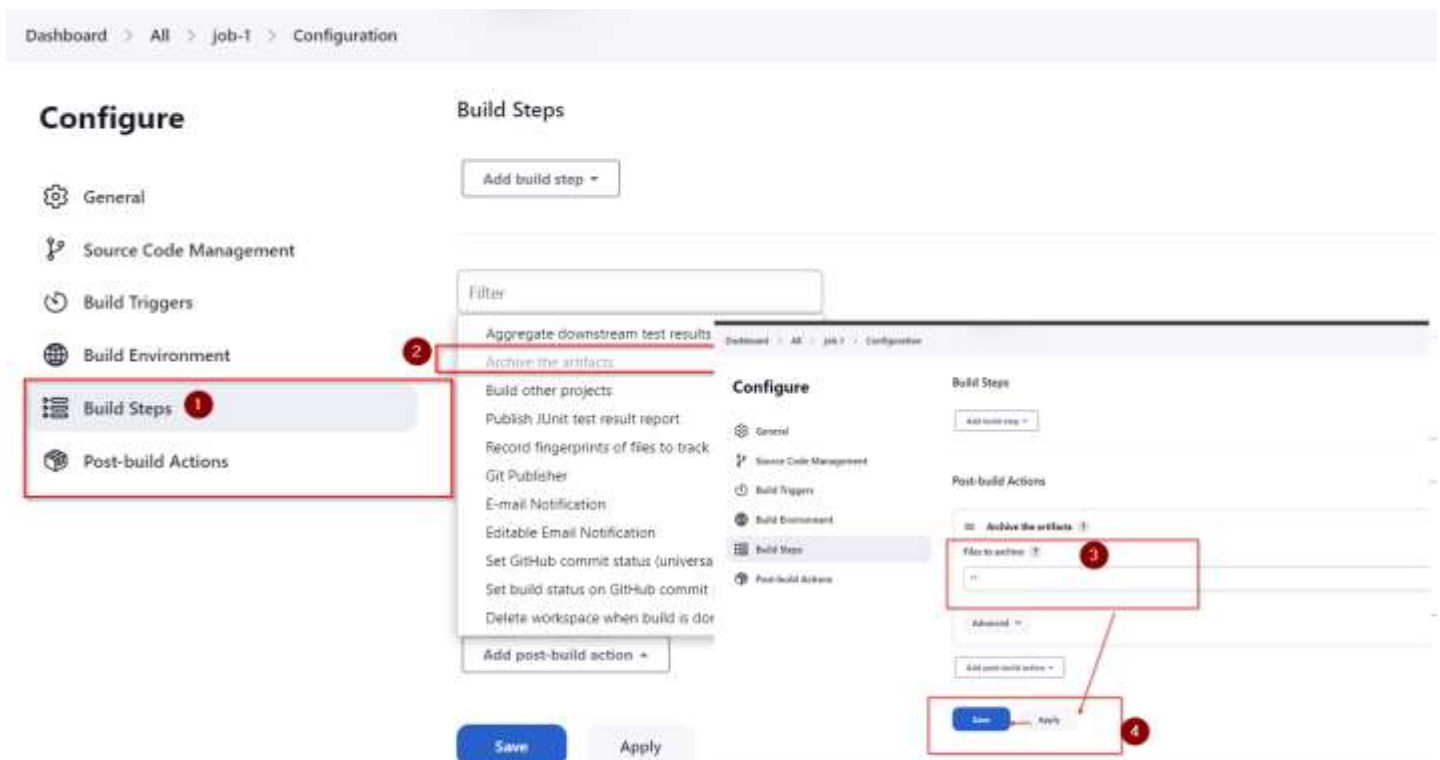
```

Click "Configure" your job/project and add these two configurations.

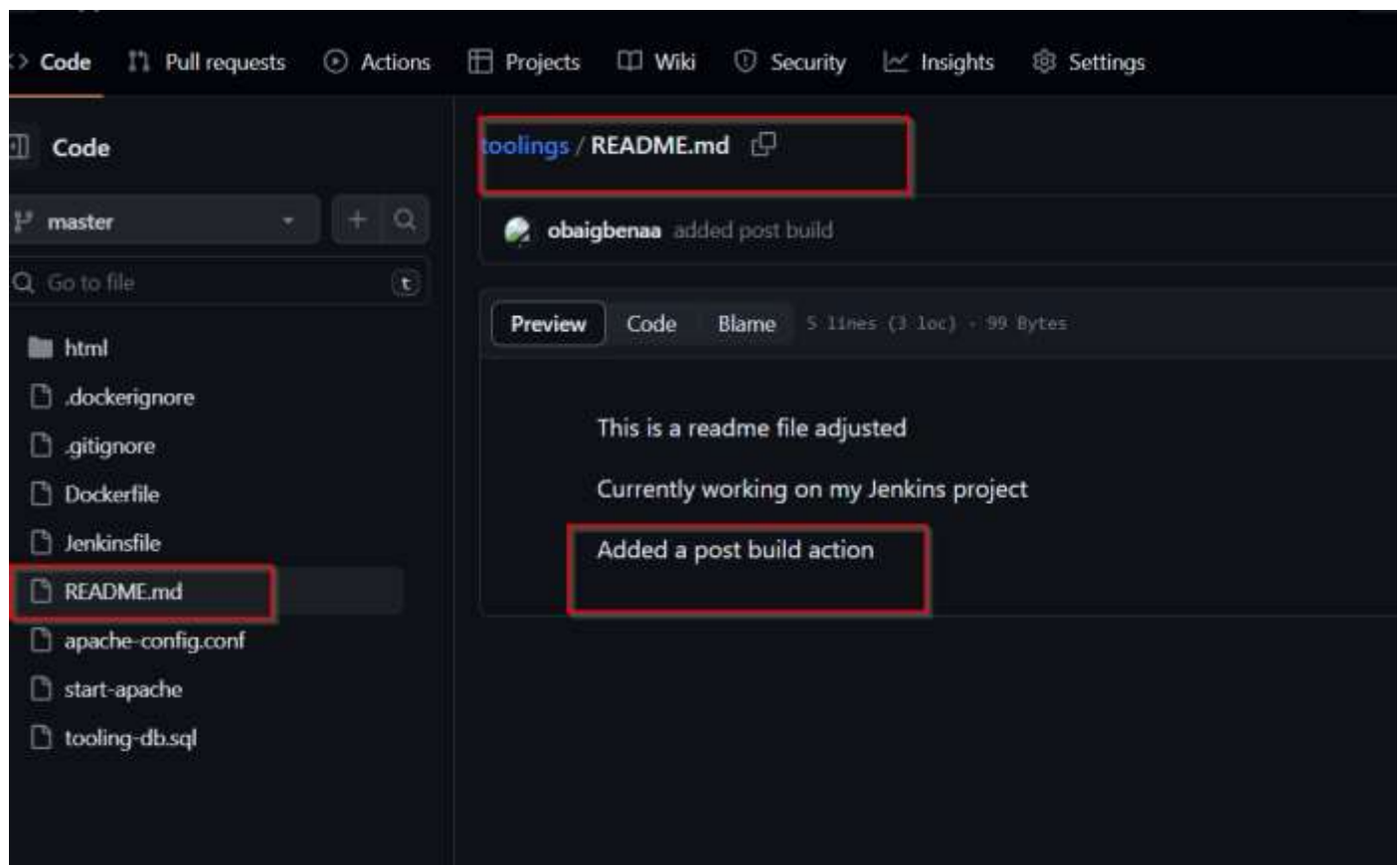
1. Configure triggering the job from GitHub webhook:



2. Configure "Post-build Actions" to archive all the files – files resulted from a build are called "artifacts".



Now, go ahead and make some changes in any file in your GitHub repository (e.g. README.MD file) and push the changes to the master branch.



You will see that a new build has been launched automatically (by webhook) and you can see its results - artifacts, saved on Jenkins server.

Status

</> Changes

Workspace

Build Now

Configure

Delete Project

GitHub Hook Log

Rename

Project job-1

This is my first jenkins job



Last Successful Artifacts

Expand all Collapse all

View

Permalinks

- Last build (#3), 1 min 8 sec ago
- Last stable build (#3), 1 min 8 sec ago
- Last
- Last

Build History trend ▾

Filter builds... /

✓ #3	21 Jun 2023, 23:24
✓ #2	21 Jun 2023, 23:00
✓ #1	21 Jun 2023, 21:36

Atom feed for all Atom feed for failures

Jenkins

Dashboard > All > job-1 > #3

Build #3 (21 Jun 2023, 23:24:35)

✓ Build Artifacts
Expand all Collapse all

</> Changes
1 added (not built (details / githublogs))

Started by GitHub push by abundance

Revision: 30c8dc47f844b3242b95827052c36
Repository: https://github.com/abundance/awesome-github

By default, the artifacts are stored on Jenkins server locally.

```
$ cd /var/lib/jenkins/jobs/job-1/builds/3
```

```
ubuntu@ip-172-31-37-181:~$ ls /var/lib/jenkins/jobs/job-1/
builds  config.xml  github-polling.log  nextBuildNumber
ubuntu@ip-172-31-37-181:~$ cd builds
-bash: cd: builds: No such file or directory
ubuntu@ip-172-31-37-181:~$ ls /var/lib/jenkins/jobs/job-1/builds/
1 2 3 legacyIds permalinks
ubuntu@ip-172-31-37-181:~$ cd 3
-bash: cd: 3: No such file or directory
ubuntu@ip-172-31-37-181:~$ cat 3
cat: 3: No such file or directory
ubuntu@ip-172-31-37-181:~$ ls /var/lib/jenkins/jobs/job-1/builds/3/
archive build.xml changelog.xml log polling.log
ubuntu@ip-172-31-37-181:~$ ls /var/lib/jenkins/jobs/job-1/builds/3/archive/
Dockerfile Jenkinsfile README.md apache-config.conf html start-apache tooling-db.sql
ubuntu@ip-172-31-37-181:~$ ls /var/lib/jenkins/jobs/job-1/builds/3/archive/README.md
/var/lib/jenkins/jobs/job-1/builds/3/archive/README.md
ubuntu@ip-172-31-37-181:~$ cd /var/lib/jenkins/jobs/job-1/builds/3/
ubuntu@ip-172-31-37-181:/var/lib/jenkins/jobs/job-1/builds/3$ ls
archive build.xml changelog.xml log polling.log
ubuntu@ip-172-31-37-181:/var/lib/jenkins/jobs/job-1/builds/3$ pwd
/var/lib/jenkins/jobs/job-1/builds/3
ubuntu@ip-172-31-37-181:/var/lib/jenkins/jobs/job-1/builds/3$
```

Step 3.

CONFIGURE JENKINS TO COPY FILES TO NFS SERVER VIA SSH

Now we have our artifacts saved locally on Jenkins server, the next step is to copy them to our NFS server to /mnt/apps directory

1. Install "Publish Over SSH" plugin.

On main dashboard select "Manage Jenkins" and choose "Manage Plugins" menu item.

On "Available" tab search for "Publish Over SSH" plugin and install it



2. Configure the job/project to copy artifacts over to NFS server.

On main dashboard select "Manage Jenkins" and choose "System" menu item.

Scroll down to Publish over SSH plugin configuration section and configure it to be able to connect to your NFS server:

Provide a private key (content of .pem file that you use to connect to NFS server via SSH)

Arbitrary name

Hostname - Here, input the private IP address of your NFS server

Username - ec2-user (since NFS server is based on EC2 with RHEL 8)

Remote directory - /mnt/apps since our Web Servers use it as a mounting point to retrieve files from the NFS server.

Test the configuration and make sure the connection returns Success. Remember, that TCP port 22 on NFS server must be open to receive SSH connections.

Dashboard > Manage Jenkins > System >

Publish over SSH

Jenkins SSH Key ?

Passphrase ?

Path to key ?

Key ?

```
1z0kkM5yNkT4VcySPmU5fb/zLrg36+P9oEw1pwLF5ji3wutSLNmoiodh+9taVb7  
n9WqNgUZNufoFs6SkVsx8ue9lgMplvDUr+llXFWP1vsugCGQYqeKT9AUqeKPUPWw  
6S8z8QKBgA/z+dw6xslvC7FWzJ1EcmnoB5kfevqw6Qc056GEOfcuioHudGkZWDC  
rXiqB8vveITKSOoB0IQPmmeXly//a8YbGdsoGU65rkHM3SIUplCy3zcD2RQ6za+  
fZSN+wyrdrAMiGUuqfU/YBsdJGlfurHIQ74cDn5jvhTtkCsz1Bov  
-----END RSA PRIVATE KEY-----
```

☐ Disable exec ?

Test the configuration and make sure the connection returns Success. Remember, that TCP port 22 on NFS server must be open to receive SSH connections.

Dashboard > Manage Jenkins > System >

☐ Disable exec ?

SSH Servers

SSH Server

Name ?

nfz-server

Hostname ?

172.31.32.27

Username ?

ac2-user

Remote Directory ?

/mnt/apps

Advanced ▾

Success

Test Configuration

MM

Save the configuration, open your Jenkins job/project configuration page and add another one "Post-build Action".

Configure it to send all files produced by the build into our previously defined remote directory. In our case we want to copy all files and directories – so we use ** to send all files. Save this configuration and go ahead.

Dashboard > job-1 > Configuration

Configure

- General
- Source Code Management
- Build Triggers
- Build Environment
- Build Steps
- Post-build Actions**

Send build artifacts over SSH ?

SSH Publishers

SSH Server Name ?
nfs-server

Advanced ▾

Transfers

Transfer Set
Source files ?
**

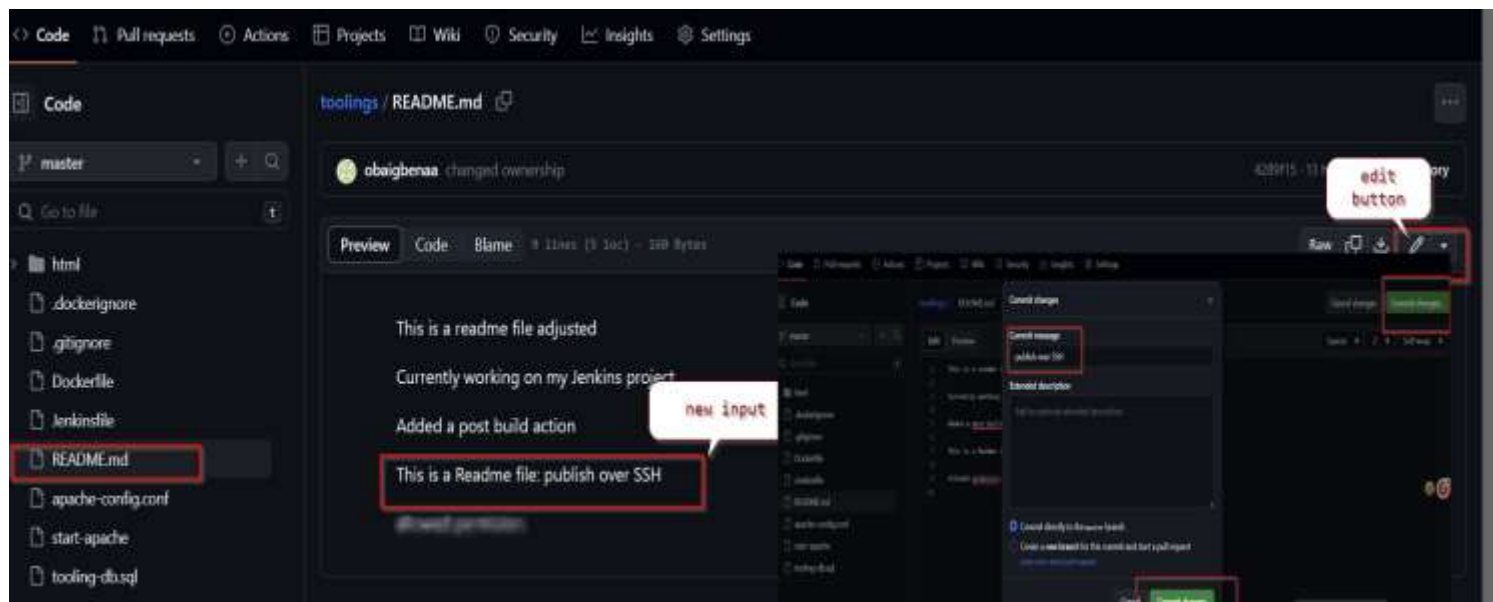
Remove prefix ?

Remote directory ?

Save Apply

Go to your github repository, edit the Readme.md file for job-1.

Webhook will trigger a new job and in the "Console Output".



Notice the error message when webhook tries to trigger a build in the console output, this is because the username inputed during the SSH server configuration was “ec2-user”. But the /mnt/apps directory is owned by “root” by default.

To change ownership to ec2-user for ‘user’ and ‘group’.

Use the command:

```
$ sudo chown -R ec2-user:ec2-user /mnt/apps/
```



```
[ec2-user@nfs-server ~]$ ls -ls /mnt/apps/
total 4
4 drwxr-xr-x. 3 root root 4096 Jun 21 12:06 html
[ec2-user@nfs-server ~]$ sudo chown -R ec2-user:ec2-user /mnt/apps/
[ec2-user@nfs-server ~]$ ls
[ec2-user@nfs-server ~]$ ls
[ec2-user@nfs-server ~]$ ls -latr /mnt/apps/
total 36
drwxr-xr-x. 5 root root 41 Jun 19 23:51 ..
drwxr-xr-x. 3 ec2-user ec2-user 4096 Jun 21 12:06 html
-rw-rw-r--. 1 ec2-user ec2-user 47 Jun 23 01:30 .dockerignore
-rw-rw-r--. 1 ec2-user ec2-user 313 Jun 23 01:30 Dockerfile
-rw-rw-r--. 1 ec2-user ec2-user 4202 Jun 23 01:30 Jenkinsfile
-rw-rw-r--. 1 ec2-user ec2-user 160 Jun 23 01:30 README.md
-rw-rw-r--. 1 ec2-user ec2-user 332 Jun 23 01:30 apache-config.conf
-rw-rw-r--. 1 ec2-user ec2-user 163 Jun 23 01:30 start-apache
drwxr-xr-x. 3 ec2-user ec2-user 161 Jun 23 01:30 .
-rw-rw-r--. 1 ec2-user ec2-user 1674 Jun 23 01:30 tooling-db.sql
[ec2-user@nfs-server ~]$ cd /mnt/apps/
[ec2-user@nfs-server apps]$ ls
apache-config.conf Dockerfile html Jenkinsfile README.md start-apache tooling-db.sql
```

Now we will trigger the last build on github and the result should look like below.

Dashboard > job-1 > #5 > Console Output

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete build '#5'

Polling Log

Git Build Data

Previous Build

Next Build

Console Output

Started by GitHub push by vishal@vishal

Running as SYSTEM

Building in workspace /var/lib/jenkins/workspace/job-1

The recommended git tool is: NONE

No credentials specified

> git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/job-1/.git # timeout=10

Fetching changes from the remote Git repository

> git config remote.origin.url https://github.com/vishal@vishal/toolings.git # timeout=10

Fetching upstream changes from https://github.com/vishal@vishal/toolings.git

> git --version # timeout=10

> git --version # 'git version 2.25.1'

> git fetch --tags --force --progress -- https://github.com/vishal@vishal/toolings.git +refs/heads/*:refs/remotes/origin/* # timeout=10

> git rev-parse refs/remotes/origin/master^{commit} # timeout=10

Checking out Revision 4289f155d7253c7d3c52089aef92c06ca7fd15d3 (refs/remotes/origin/master)

> git config core.sparsecheckout # timeout=10

> git checkout -f 4289f155d7253c7d3c52089aef92c06ca7fd15d3 # timeout=10

Commit message: "changed ownership"

> git rev-list --no-walk 1eb71856fd80f802572e174cd95e8aea57a1b3f0 # timeout=10

Archiving artifacts

SSH: Connecting from host [ip-172-31-37-181]

SSH: Connecting with configuration [nfs-server] ...

SSH: Disconnecting configuration [nfs-server] ...

SSH: Transferred 25 file(s)

Finished: SUCCESS

The screenshot shows the Jenkins web interface. At the top, the Jenkins logo is on the left, and a search bar with the text "Search (CTRL+K)" is on the right. Below the header, a breadcrumb trail shows "Dashboard" followed by "job-1" and "#5", with "#5" highlighted in a red box. On the left sidebar, there are several menu items: "Status" (highlighted), "Changes", "Console Output", "Edit Build Information", "Delete build '#5'", "Polling Log", "Git Build Data", and "Previous Build". The main content area displays "Build #5 (23 Jun 2023, 01:30:33)" with a green checkmark icon. Below this, there are sections for "Build Artifacts" (with a red box around the "Expand all Collapse all" and "View" links), "Changes" (with a red box around the change "1. changed ownership (details / githubweb)"), and "Started by GitHub push by c1c4g10m1a". At the bottom, the "git" icon is shown, followed by the "Revision" (4289f155d7353c7d3c52069aef92c06ca7fd15d3) and "Repository" (https://github.com/c1c4g10m1a/toolings.git) information, with a link to "refs/remotes/origin/master".

To make sure the files in /mnt/apps have been updated – connect via SSH to your NFS server and check README.MD file

```
$ cat /mnt/apps/README.md
```

```
[ec2-user@ip-172-31-32-27 ~]$ cat /mnt/apps/README.md
This is a readme file adjusted

Currently working on my Jenkins project

Added a post build action

This is a Readme file: publish over SSH

allowed permission.
```

If you see the changes you previously made in your GitHub – the job works as expected!