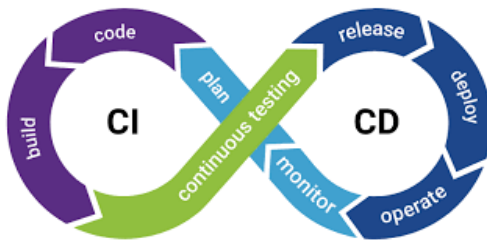




PROJECT  
ON

## CI/CD PIPELINE USING JENKINS



## CI/CD PIPELINE WITH JENKINS

\*\*\*\*\***QUESTIONNAIRE**\*\*\*\*\*

### **Q.What is continuous integration?**

Continuous Integration is a software development practice that involves frequently integrating code changes to the shared code repository where automatic builds and tests are performed.

Main goal of this continuous integration is to catch and fix the issues early in the development process, reducing the risk of integration issues and ensuring that the software is always in a releasable state.

#### **Real time example:**

A real-time example for continuous integration could be a software development team working on an e-commerce website.

The team uses a continuous integration tool, such as Jenkins to automatically build and test the website every time a code change is committed to the code repository.

For instance, when a developer submits new code to the website, the continuous integration tool automatically pulls the code from the repository, builds the application, and runs a suite of automated tests to verify that the changes have not introduced any issues.

If any issues are found, the team is notified immediately, and they can fix the issues before they become more significant problems.

#### **MAJOR BENEFIT:**

By using continuous integration, the development team can catch issues early in the development process, ensure that the website is always in a working state, and improve the overall quality of the software.

=====

### **Q.What is Continuous Deployment/Delivery?**

Continuous Deployment is a software development practice where code changes are automatically deployed to **production environments** after passing through automated testing and validation stages, ensuring that software updates can be released to end-users quickly and reliably.

To implement Cd, organizations typically use a combination of automation tools, infrastructure-as-code, and version control systems. This enables them to automate the entire deployment process and achieve high levels of reliability, scalability, and efficiency.

#### **Real time example:**

A real-time example for continuous delivery and deployment could be a software development team working on a mobile application for a bank.

## CI/CD PIPELINE WITH JENKINS

**The team uses a continuous delivery and deployment tool, such as Circle CI or GitLab CI/CD, to automatically build, test, and deploy the mobile application every time a code change is committed to the code repository.**

For instance, when a developer submits new code to the mobile application, the continuous delivery tool automatically pulls the code from the repository, builds the application, runs a suite of automated tests, and packages the application for release to production.

**Once the application is built, tested, and packaged, the continuous deployment tool deploys the application to the production environment, making it available for use by bank customers.**

By using continuous delivery and deployment, the development team can quickly and reliably release new features and updates to the mobile application, ensuring that the bank's customers have access to the latest and most secure version of the application.

Jenkins is a popular open-source automation server that can be used for various software development tasks, including building, testing, and deploying software.

=====

### **Q.What is Jenkins?**

Jenkins is a tool that is used for automation and it's a open source server that allows the developers to build, test and deploy software. By using Jenkins we can make continuous integration of (projects) or end -to-endpoint automation. Jenkins is entirely developed using Java.

=====

### **Q.Why only continuous Integration?**

If we did not use CI , we have to check all the things manually, such as pulling code from the repo, building the code, run the code, test the code, finding the bugs.

Simply by using the CI we can perform all the things using automation.

=====

### **Q.Why we have to use only Jenkins?**

Jenkins is a popular **open-source automation server** that provides a wide range of features and plugins for continuous integration and continuous delivery (CI/CD) pipelines. It allows you to automate the build, test, and deployment processes, and can be easily integrated with various tools and platforms.

By using Jenkins we can reduce our time to 25-30% of time,

One of the **main advantages of Jenkins is its flexibility and customizability**. With its extensive plugin ecosystem and support for various programming languages and technologies, Jenkins can be tailored to meet the specific needs of your project. Additionally, **Jenkins is well-documented, has a large user community, and provides robust security features.**

## CI/CD PIPELINE WITH JENKINS

That being said, there are other automation tools and platforms available that you may want to consider, depending on your specific requirements and preferences. Some examples include Travis CI, Circle CI, GitLab CI/CD, and Azure DevOps.

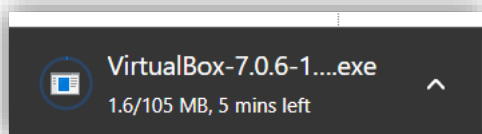
\*\*\*\*\*PROCEDURE\*\*\*\*\*

- To do this project first we need to install VirtualBox

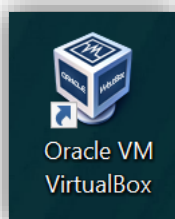
<https://www.virtualbox.org/wiki/Downloads>



- click on windows hosts and it automatically downloads installer.exe file



- After installation we can see



- For the ubuntu server we need vdi file for that

## CI/CD PIPELINE WITH JENKINS

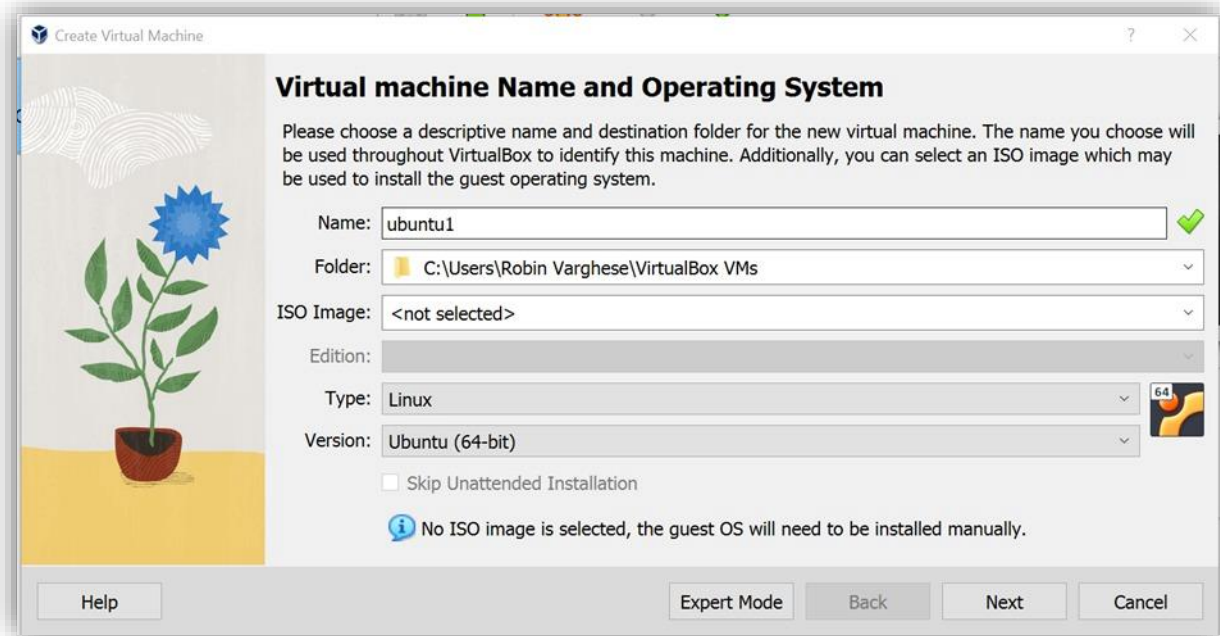
<https://www.osboxes.org/ubuntu/>



- Download it ubuntu 20.04.4 Focal Fossa
- Extract the vdi file
- Tap new option visible as shown in figure below



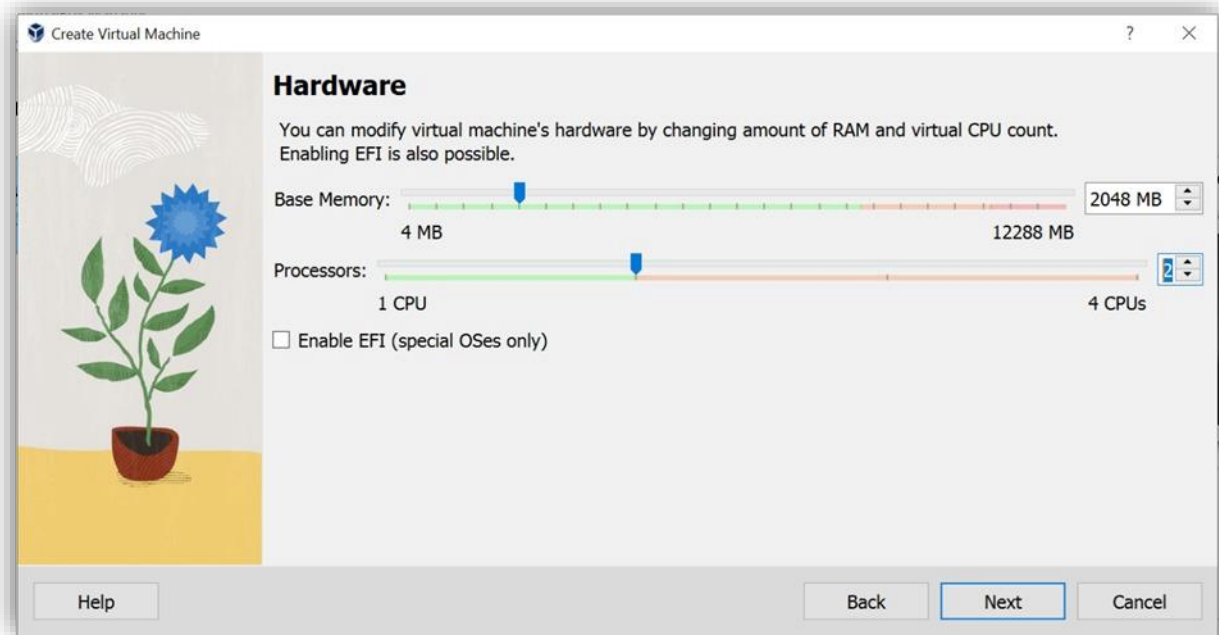
Give a name to your server, type must be linux and Version must be ubuntu(64-bit)



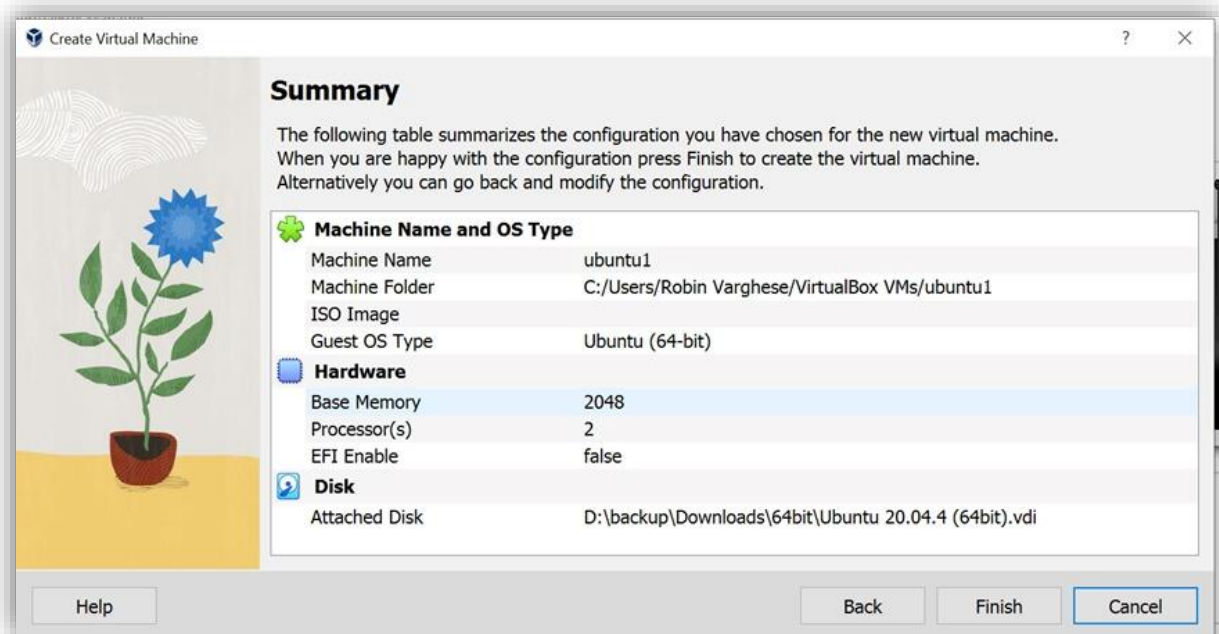
- Change the processors to "2" cpu's

## CI/CD PIPELINE WITH JENKINS

And then next

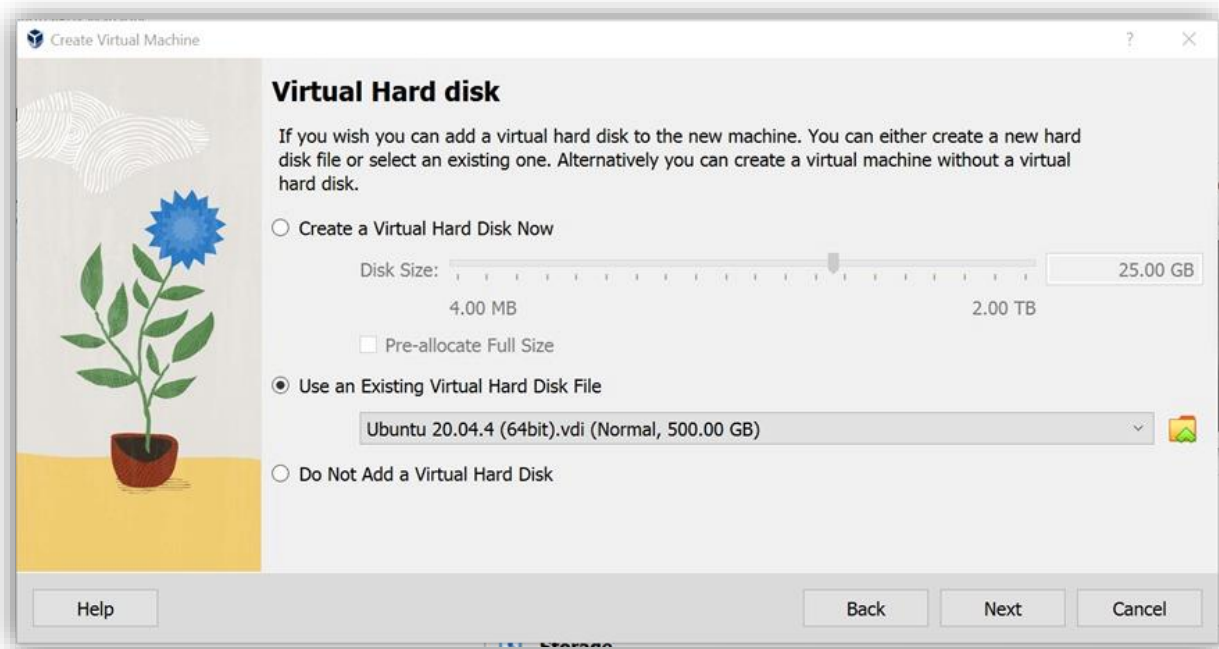


- Cross-verify the details in the summary shown below



- Select use as an Existing Virtual Hard Disk File
- Browse it to the extracted vdi file and select it

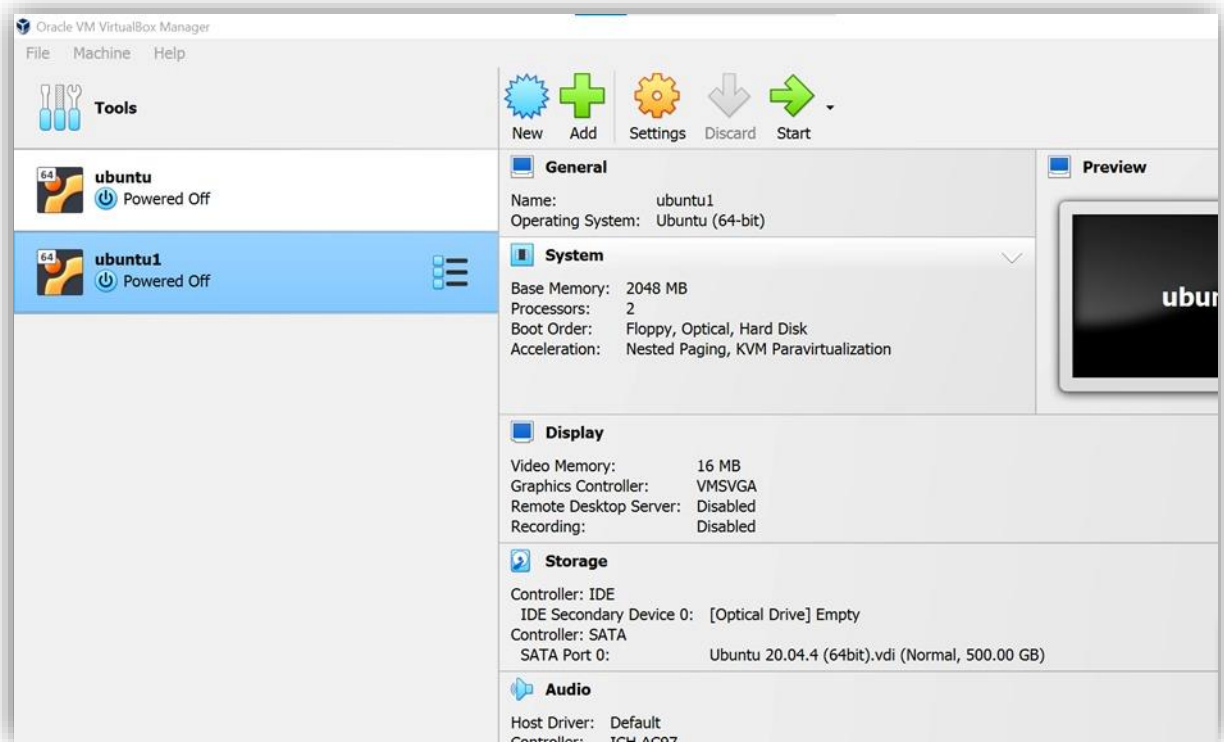
## CI/CD PIPELINE WITH JENKINS



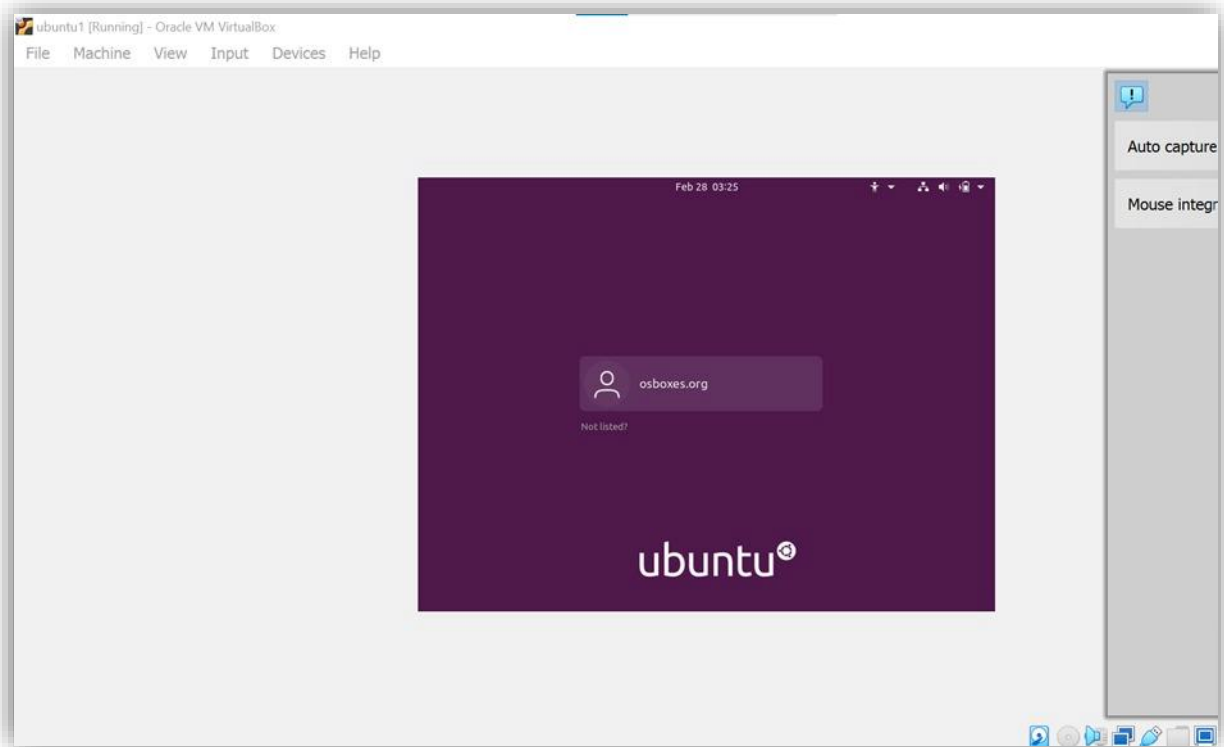
- Press next and finish it
- We can observe that ubuntu1 is created below
- Select it and press start



## CI/CD PIPELINE WITH JENKINS



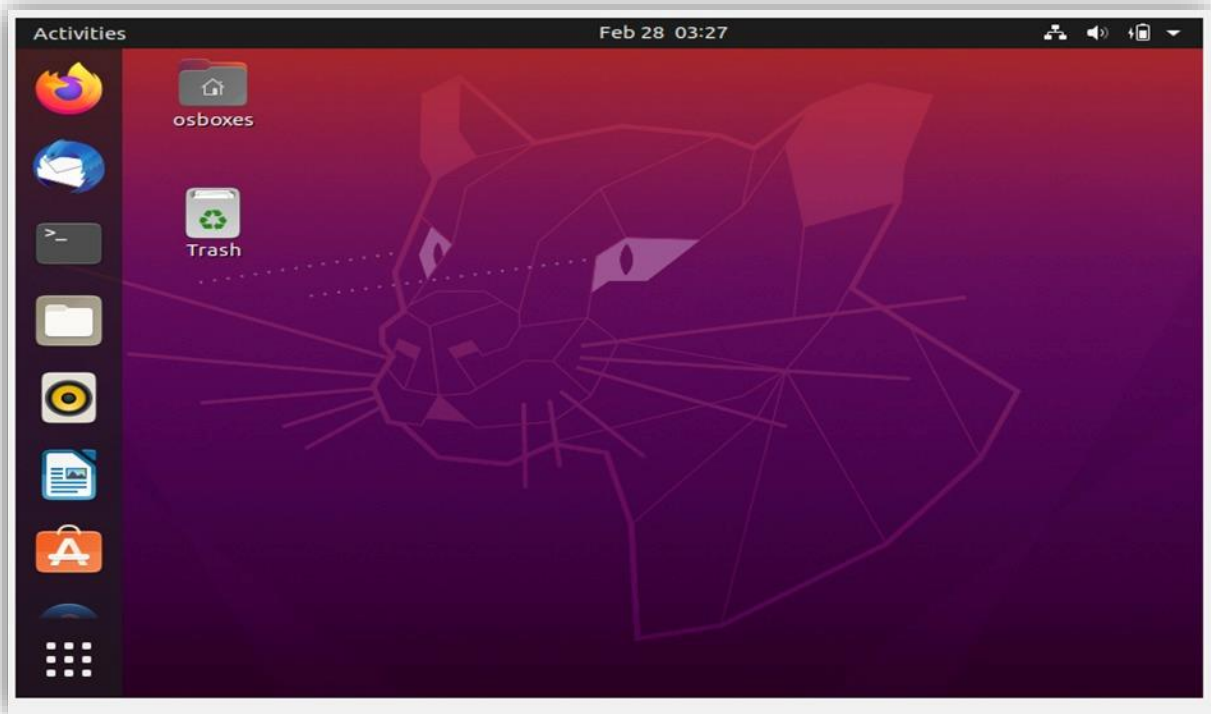
- Enter into osboxes with password: “osboxes.org”



- We will enter into Desktop page of ubuntu as shown below



## CI/CD PIPELINE WITH JENKINS



- Get into terminal shown above
- Now follow the commands below
- We need to update the ubuntu server

```
$ sudo apt update
```

```
$ sudo apt upgrade
```

## CI/CD PIPELINE WITH JENKINS

```
vivek@nixcraft-asus:~$ sudo apt-get update
[sudo] password for vivek:
Hit:1 https://deb.nodesource.com/node_10.x bionic InRelease
Get:2 http://security.ubuntu.com/ubuntu bionic-security InRelease [83.2 kB]
Hit:3 http://archive.ubuntu.com/ubuntu bionic InRelease
Hit:4 http://ppa.launchpad.net/ansible/ansible/ubuntu bionic InRelease
Hit:5 http://ppa.launchpad.net/prerelease.keybase.io/deb stable InRelease
Ign:6 http://dl.google.com/linux/chrome/deb stable InRelease
Get:7 http://archive.ubuntu.com/ubuntu bionic-updates InRelease [88.7 kB]
Hit:8 http://ppa.launchpad.net/gezakovacs/ppa/ubuntu bionic InRelease
Hit:9 http://dl.google.com/linux/chrome/deb stable Release
Hit:10 http://ppa.launchpad.net/openshot.developers/ppa/ubuntu bionic InRelease
Get:12 http://security.ubuntu.com/ubuntu bionic-security/universe i386 Packages [89.2 kB]
Hit:13 http://ppa.launchpad.net/peek-developers/stable/ubuntu bionic InRelease
Hit:14 http://repo.pritunl.com/stable/apt bionic InRelease
Get:15 http://security.ubuntu.com/ubuntu bionic-security/universe amd64 Packages [89.2 kB]
Get:16 http://archive.ubuntu.com/ubuntu bionic-updates/main amd64 Packages [413 kB]
Get:17 http://archive.ubuntu.com/ubuntu bionic-updates/main i386 Packages [369 kB]
Get:18 http://archive.ubuntu.com/ubuntu bionic-updates/main Translation-en [153 kB]
Fetched 1,286 kB in 3s (377 kB/s)
Reading package lists... Done
vivek@nixcraft-asus:~$
```

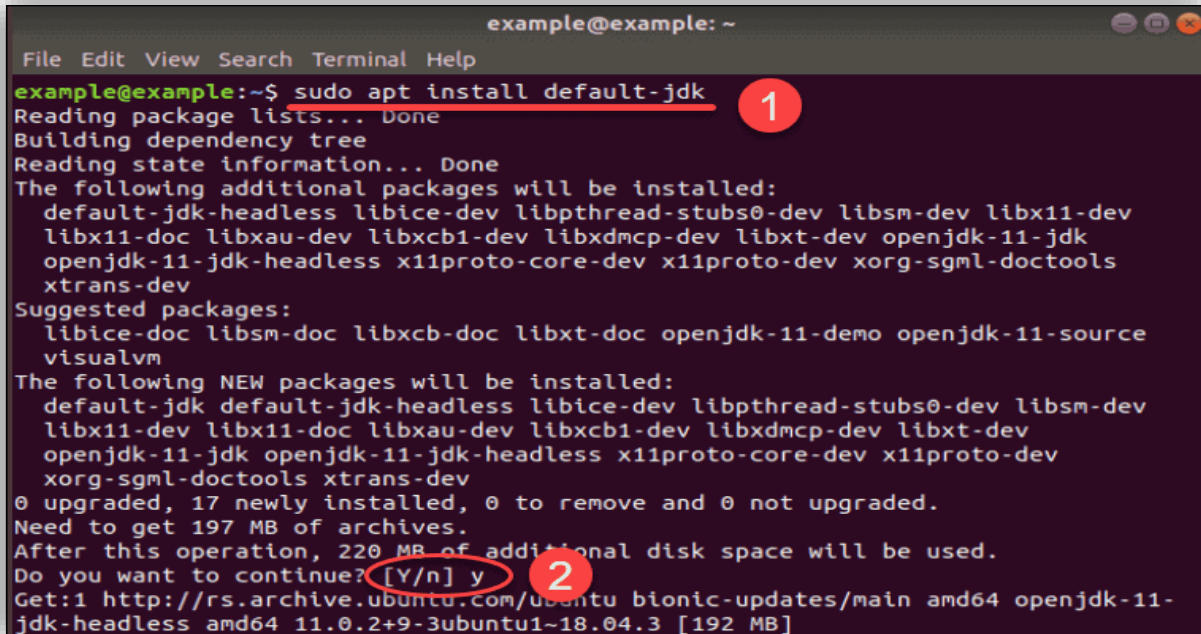
```
omer@ubuntu: ~
File Edit View Search Terminal Help
omer@ubuntu:~$ sudo apt-get upgrade
Reading package lists... Done
Building dependency tree
Reading state information... Done
Calculating upgrade... Done
The following packages were automatically installed and are no longer required:
  efibootmgr libfwupd1 linux-headers-5.3.0-28 linux-headers-5.3.0-28-generic
  linux-image-5.3.0-28-generic linux-modules-5.3.0-28-generic
  linux-modules-extra-5.3.0-28-generic
Use 'sudo apt autoremove' to remove them.
The following packages have been kept back:
  libgl1-mesa-dri libxatracker2 linux-generic-hwe-18.04
  linux-headers-generic-hwe-18.04 linux-image-generic-hwe-18.04 netplan.io
The following packages will be upgraded:
  base-files dmidecode dmsetup fwupdate fwupdate-signed gdb gdbserver
  gir1.2-json-1.0 gir1.2-nm-1.0 gnome-software gnome-software-common
  gnome-software-plugin-snap grub-common grub-pc grub-pc-bin grub2-common
  initramfs-tools initramfs-tools-bin initramfs-tools-core iproute2 kmod
  libasound2 libasound2-data libc-bin libc6 libc6-dbg libcryptsetup12
  libdevmapper1.02.1 libdjvulibre-text libdjvulibre21 libdrm-amdgpu1
  libdrm-common libdrm-intel1 libdrm-nouveau2 libdrm-radeon1 libdrm2
  libegl-mesa0 libgbm1 libgl1-mesa-glx libglapi-mesa libglx-mesa0 libgnutls30
  libjson-glib-1.0-0 libjson-glib-1.0-common libkmod2 libnm0
  libnss-myhostname libnss-systemd libpam-modules libpam-modules-bin
  libpam-runtime libpam-systemd libpam0g libpcap0.8 libsane-common libsane1
  libsgutils2-2 libsystemd0 libudev1 libxau6 linux-firmware locales
  multiarch-support network-manager
  network-manager-config-connectivity-ubuntu nplan open-vm-tools
  open-vm-tools-desktop python-apt-common python3-apt python3-distupgrade
```

- Here get is not an issue we may or may not use it
- For the installation of Jenkins in the ubuntu we require java
- So we install jdk file

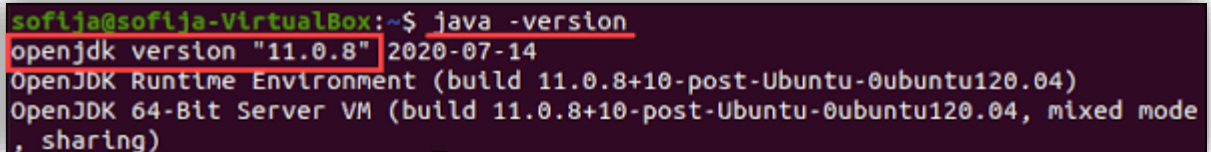
## CI/CD PIPELINE WITH JENKINS

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

```
$ java -version
```



```
example@example: ~
File Edit View Search Terminal Help
example@example:~$ sudo apt install default-jdk
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  default-jdk-headless libice-dev libpthread-stubs0-dev libsm-dev libx11-dev
  libx11-doc libxau-dev libxcb1-dev libxdmcp-dev libxt-dev openjdk-11-jdk
  openjdk-11-jdk-headless x11proto-core-dev x11proto-dev xorg-sgml-doctools
  xtrans-dev
Suggested packages:
  libice-doc libsm-doc libxcb-doc libxt-doc openjdk-11-demo openjdk-11-source
  visualvm
The following NEW packages will be installed:
  default-jdk default-jdk-headless libice-dev libpthread-stubs0-dev libsm-dev
  libx11-dev libx11-doc libxau-dev libxcb1-dev libxdmcp-dev libxt-dev
  openjdk-11-jdk openjdk-11-jdk-headless x11proto-core-dev x11proto-dev
  xorg-sgml-doctools xtrans-dev
0 upgraded, 17 newly installed, 0 to remove and 0 not upgraded.
Need to get 197 MB of archives.
After this operation, 220 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
Get:1 http://rs.archive.ubuntu.com/ubuntu bionic-updates/main amd64 openjdk-11-
jdk-headless amd64 11.0.2+9-3ubuntu1~18.04.3 [192 MB]
```



```
soflja@soflja-VirtualBox:~$ java -version
openjdk version "11.0.8" 2020-07-14
OpenJDK Runtime Environment (build 11.0.8+10-post-Ubuntu-0ubuntu120.04)
OpenJDK 64-Bit Server VM (build 11.0.8+10-post-Ubuntu-0ubuntu120.04, mixed mode
, sharing)
```

- Before installing the Jenkins we must get the key from official file for that we use

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

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

- Now updating again

```
$ sudo apt update
```

## CI/CD PIPELINE WITH JENKINS

```
linuxconf@linuxconf-01: /Desktop$ sudo apt update
Get:1 http://ppa.launchpad.net/peek-developers/daily/ubuntu eoan InRelease [15.9 kB]
Err:1 http://ppa.launchpad.net/peek-developers/daily/ubuntu eoan InRelease
  The following signatures couldn't be verified because the public key is not available: NO_PUBKEY 9578539176BAFBC6
Hit:2 http://au.archive.ubuntu.com/ubuntu focal InRelease
Hit:3 http://au.archive.ubuntu.com/ubuntu focal-updates InRelease
Hit:4 http://au.archive.ubuntu.com/ubuntu focal-backports InRelease
Hit:5 http://au.archive.ubuntu.com/ubuntu focal-security InRelease
Reading package lists... Done
W: GPG error: http://ppa.launchpad.net/peek-developers/daily/ubuntu eoan InRelease: The following signatures couldn't
be verified because the public key is not available: NO_PUBKEY 9578539176BAFBC6
E: The repository 'http://ppa.launchpad.net/peek-developers/daily/ubuntu eoan InRelease' is not signed.
N: Updating from such a repository can't be done securely, and is therefore disabled by default.
N: See apt-secure(8) manpage for repository creation and user configuration details.
```

- We are replacing their public key with our's

```
$ sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys <pubic key>
```

```
itslinuxfoss@ubuntu:~$
itslinuxfoss@ubuntu:~$ sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-
keys 7721F63BD38B4796
Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d instead (s
ee apt-key(8)).
Executing: /tmp/apt-key-gpghome.qystW0x4Gb/gpg.1.sh --keyserver keyserver.ubuntu
.com --recv-keys 7721F63BD38B4796
gpg: key 7721F63BD38B4796: public key "Google Inc. (Linux Packages Signing Autho
rity) <linux-packages-keymaster@google.com>" imported
gpg: Total number processed: 1
gpg:      imported: 1
itslinuxfoss@ubuntu:~$
itslinuxfoss@ubuntu:~$
```

- In the place of <public key> paste the key from the above error you get after the update and again update

```
$ sudo apt update
```

- Now atleast installing jenkins

```
$ sudo apt install jenkins
```

```
$ sudo systemctl start jenkins
```

```
$ sudo systemctl status Jenkins
```

```
$ jenkins --version
```



## CI/CD PIPELINE WITH JENKINS

```
linuxuser@newUbuntuHost:~$ sudo apt install jenkins
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  libfprint-2-tod1 libllvm10
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  daemon net-tools
The following NEW packages will be installed:
  daemon jenkins net-tools
0 upgraded, 3 newly installed, 0 to remove and 0 not upgraded.
Need to get 67.2 MB of archives.
After this operation, 68.5 MB of additional disk space will be used.
Do you want to continue? [Y/n] y
```

- Using firewall commands and assigning port for jenkins

```
$ sudo ufw allow 8080/tcp
```

```
$ sudo ufw enable
```

```
$ sudo ufw status
```

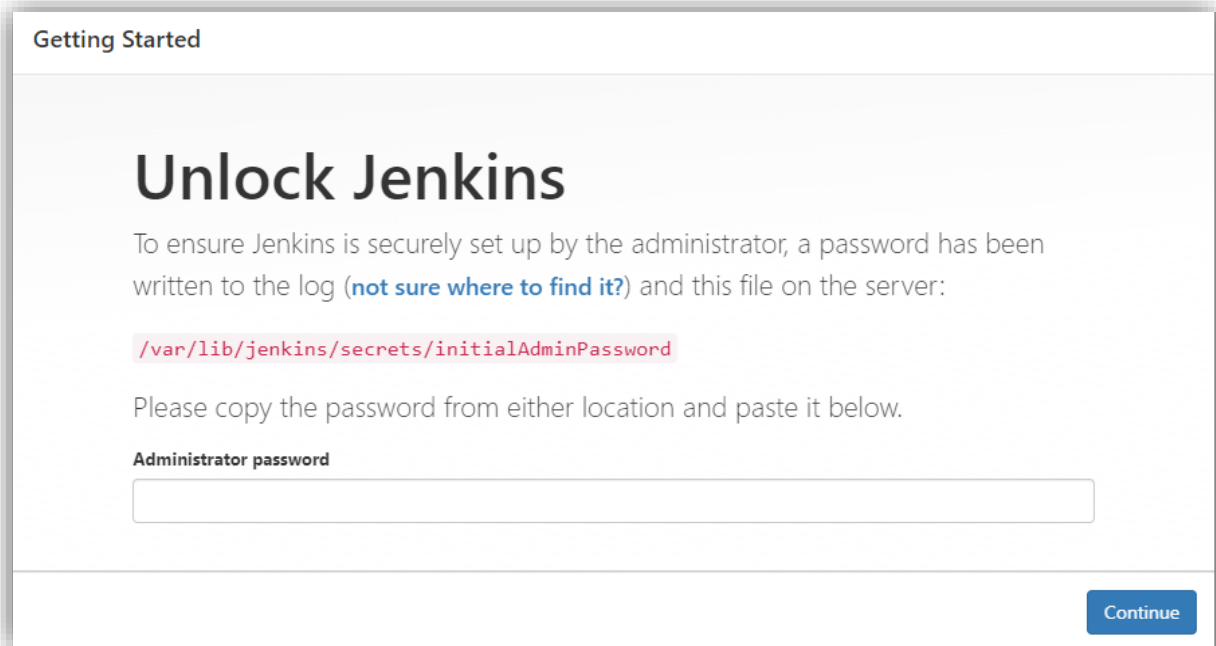
```
itslinuxfoss@itslinuxfoss-VirtualBox: ~$ sudo ufw allow 8080/tcp
Skipping adding existing rule
Skipping adding existing rule (v6)
itslinuxfoss@itslinuxfoss-VirtualBox:~$ sudo ufw enable
Firewall is active and enabled on system startup
itslinuxfoss@itslinuxfoss-VirtualBox:~$
```

<https://localhost:8080>

or just type "localhost:8080" in firefox browser

- Jenkins interface gets displayed.

## CI/CD PIPELINE WITH JENKINS



- In below path it will be visible. so go into that path and find the password and paste it .

```
$ cd /var/lib/jenkins/secrets  
$ cat initialAdminPassword
```

(password copy-paste)

- now we can access jenkins.
- Click on Select plugins to install.
- Create username and password.

(Its not mandatory we can get requied plugins from “Manage plugin” itself).

## CI/CD PIPELINE WITH JENKINS

Getting Started

Customize Jenkins

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

Install suggested plugins

Install plugins the Jenkins community finds most useful.

Select plugins to install

Select and install plugins most suitable for your needs.

Jenkins 2.138.2

Getting Started

Create First Admin User

Username

RisingPhoenix

Password

\*\*\*\*\*

Confirm password

\*\*\*\*\*

Full name

CI/CD PIPELINE WITH JENKINS

Jenkins 2.387.2

Skip and continue as admin

Save and Continue

15 | Page

## CI/CD PIPELINE WITH JENKINS

Getting Started

# Instance Configuration

Jenkins URL:

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 2.387.2

Not now

Save and Finish

Getting Started

# Jenkins is ready!

Your Jenkins setup is complete.

Start using Jenkins

Jenkins 2.387.2

- For the containerization we require docker so install.
- Get back to ubuntu terminal and follow the commands.

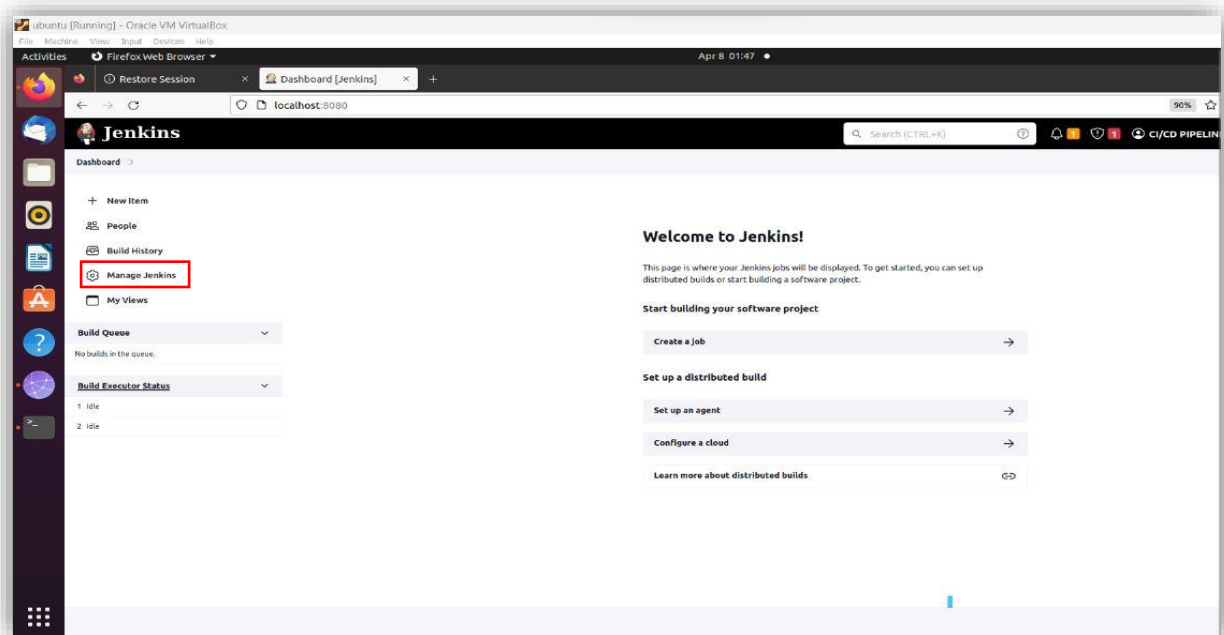


## CI/CD PIPELINE WITH JENKINS

```
$ sudo apt install docker.io
$ docker --version
$ docker login
<username of dockerhub>
<password of dockerhub>
$ sudo apt install git
$ git clone <git repository url>
$ ls -al
```

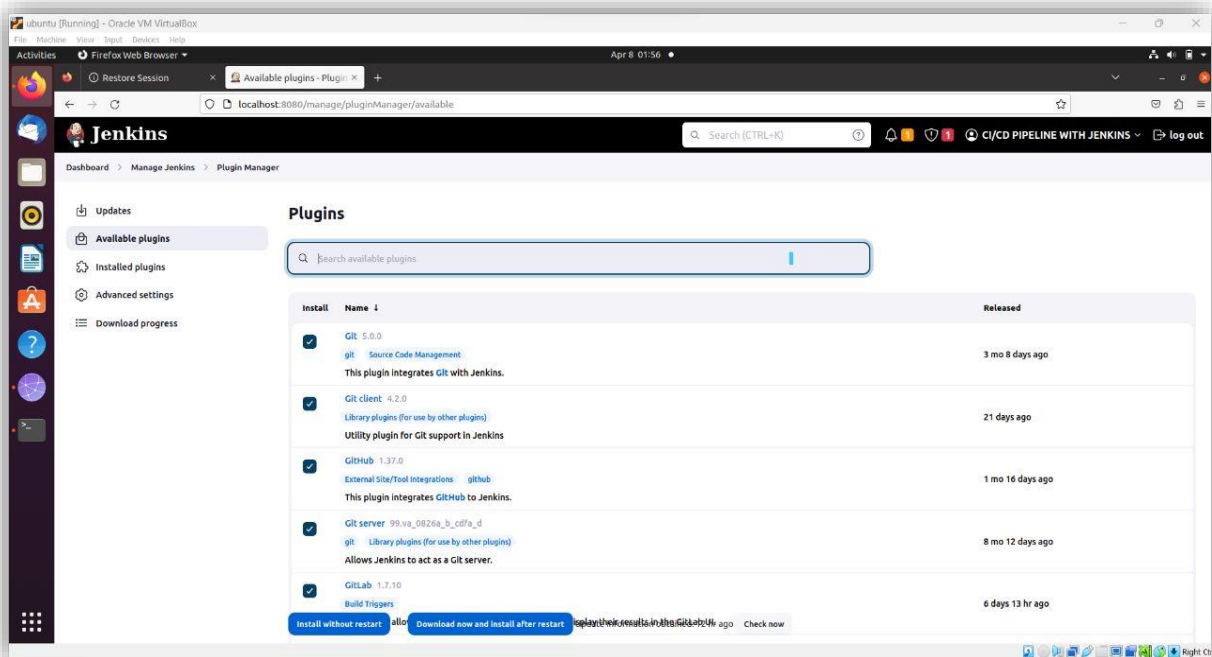
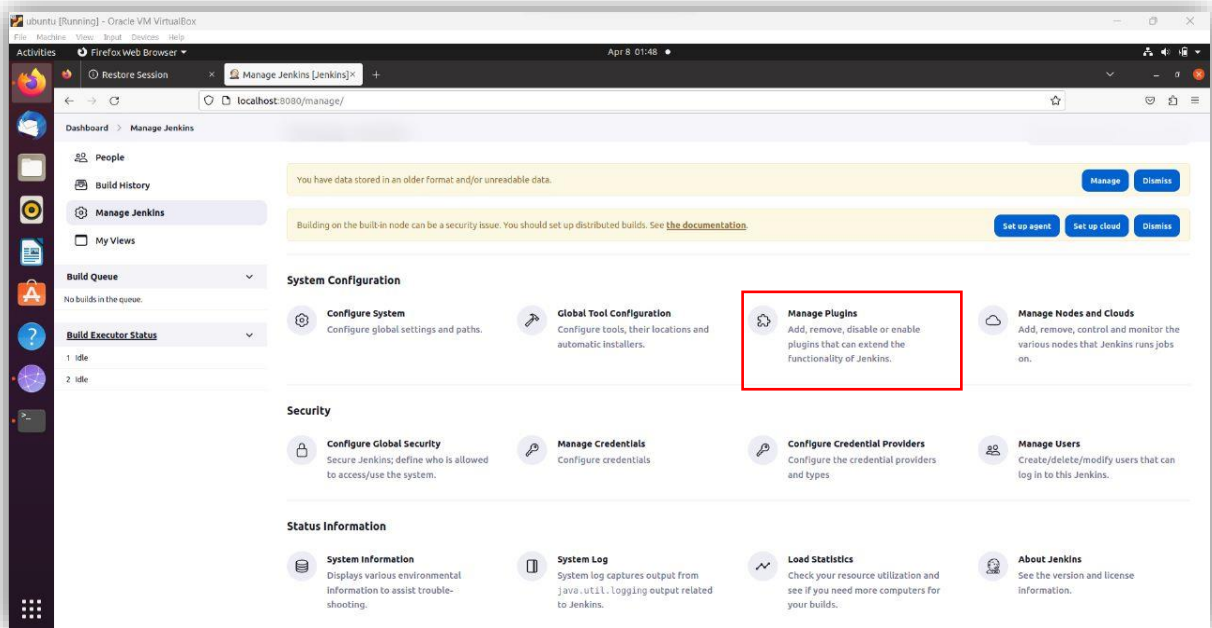
- change access permission of “.git” and every file in “.git” (even internally must change the permission or according to error getting ,we can change it).

```
$ chmod 777 <filename>
```

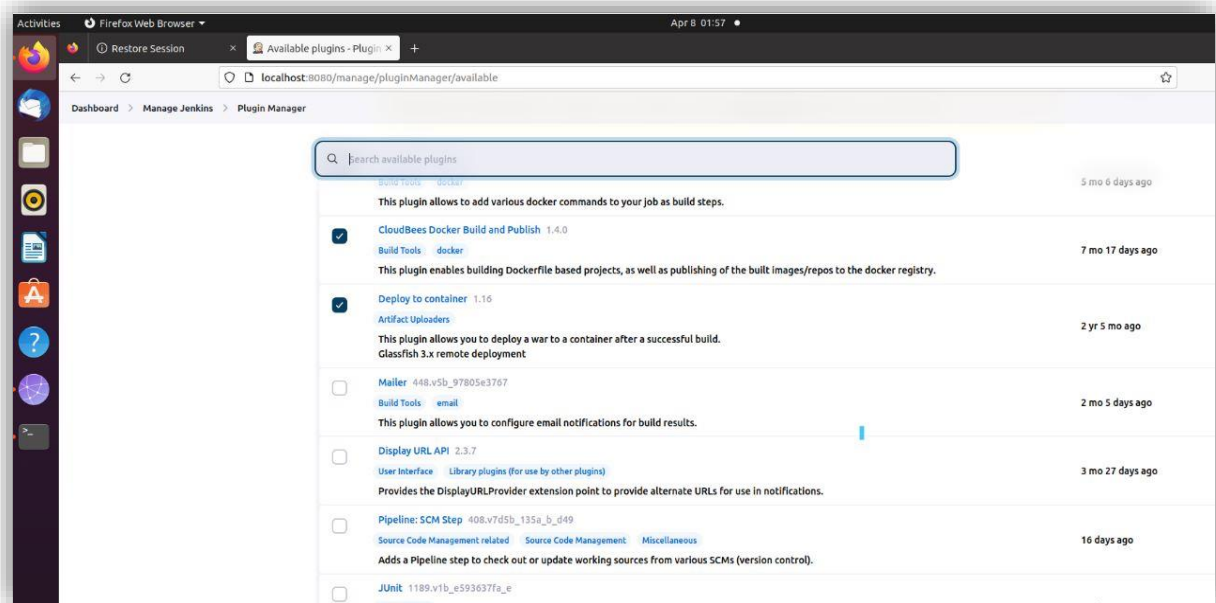
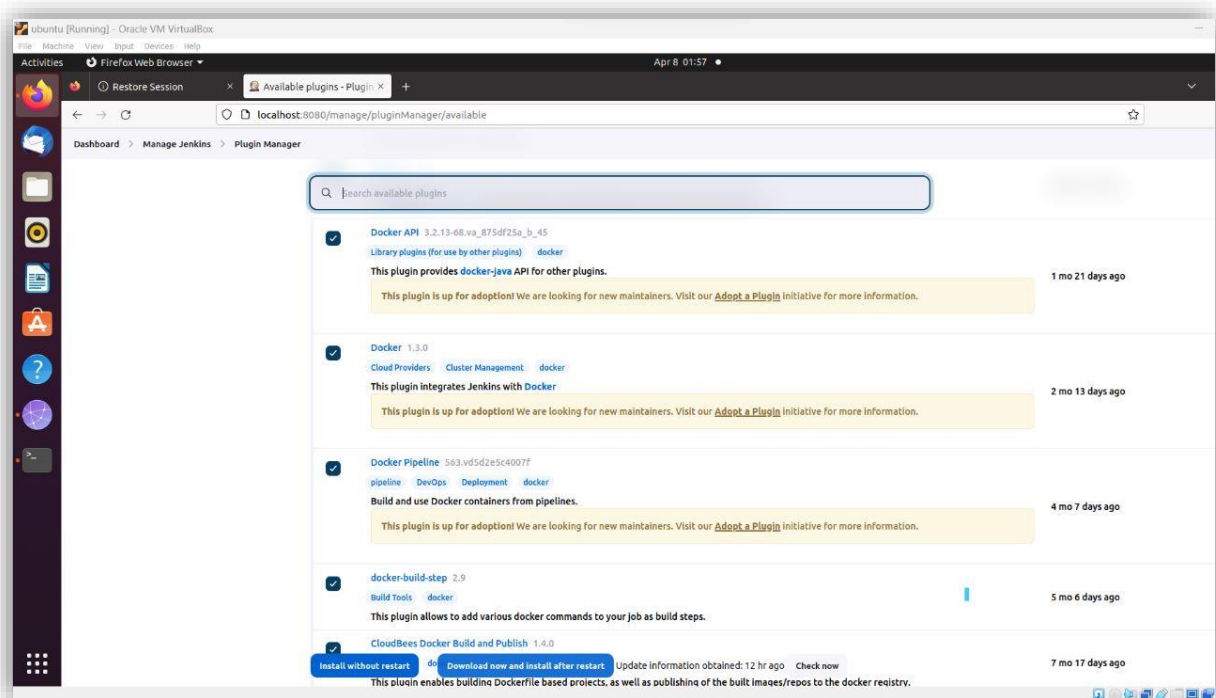


- Now press manage Jenkins inorder to install required pulgins for this project.
- And search for the below mentioned pulgins.

# CI/CD PIPELINE WITH JENKINS



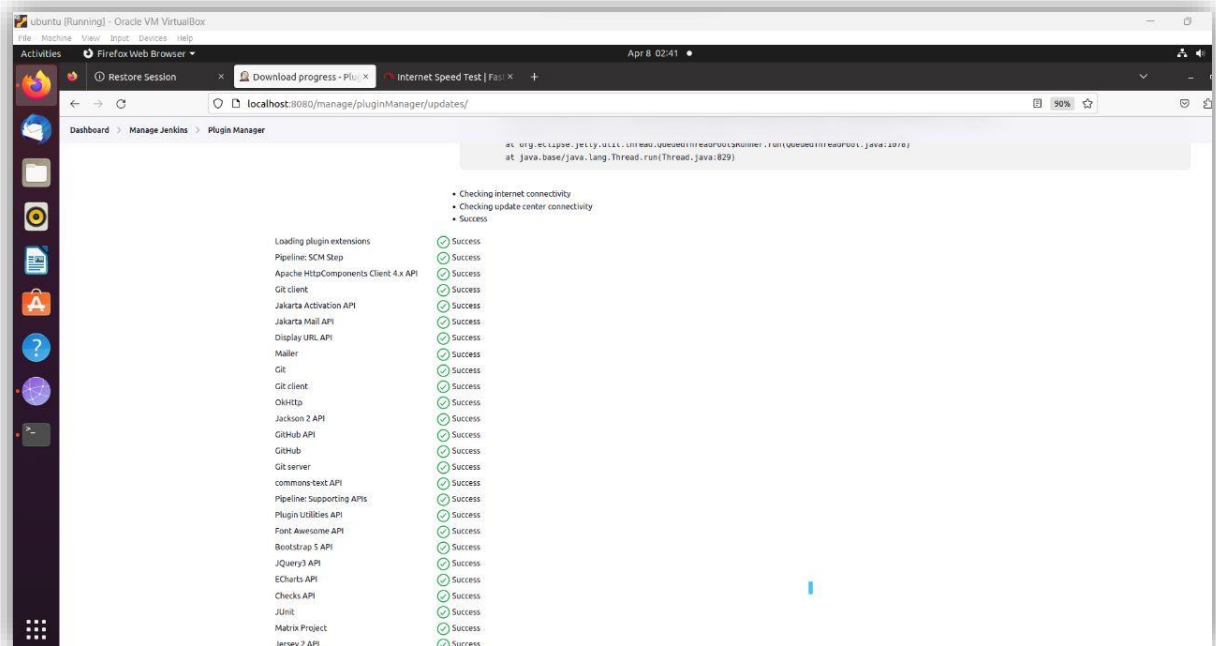
# CI/CD PIPELINE WITH JENKINS



Install without restart

- Now install without restarting the Jenkins.

## CI/CD PIPELINE WITH JENKINS



- After successful installation press the below option mentioned.

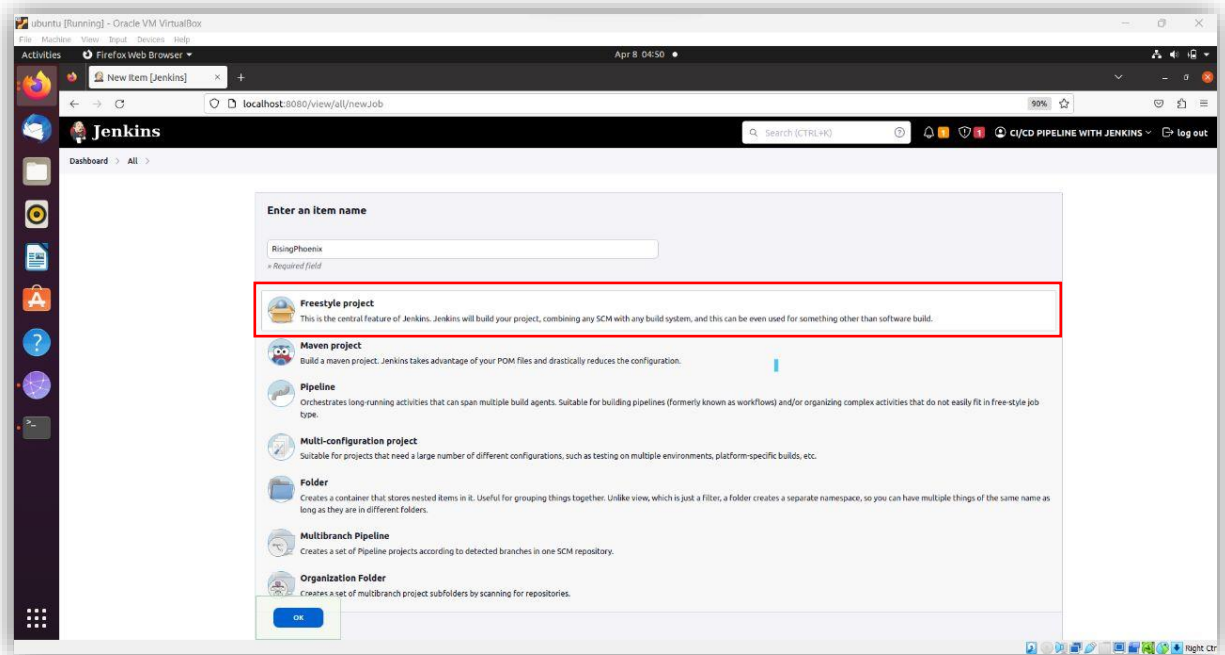
[Go back to the top page](#)  
(you can start using the installed plugins right away)

- Now in dashboard .click new item for the project.

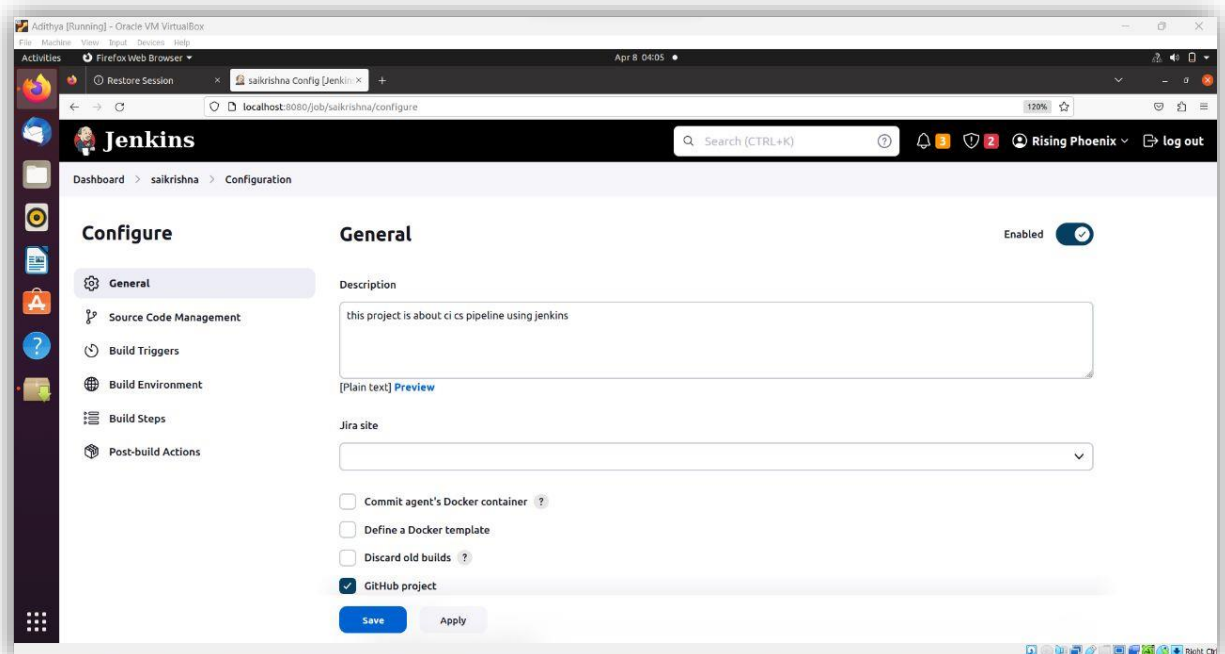
Dashboard >  
  
+ New Item

- In that give the name you like.
- Select the free style project.
- Press ok as shown below.

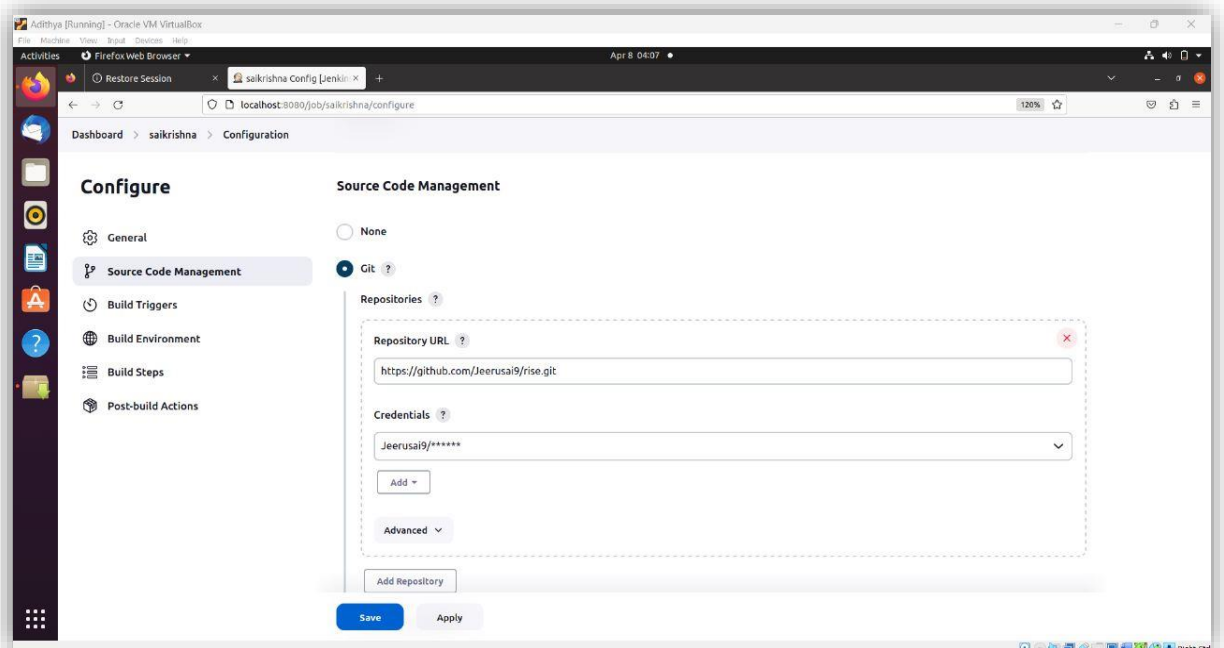
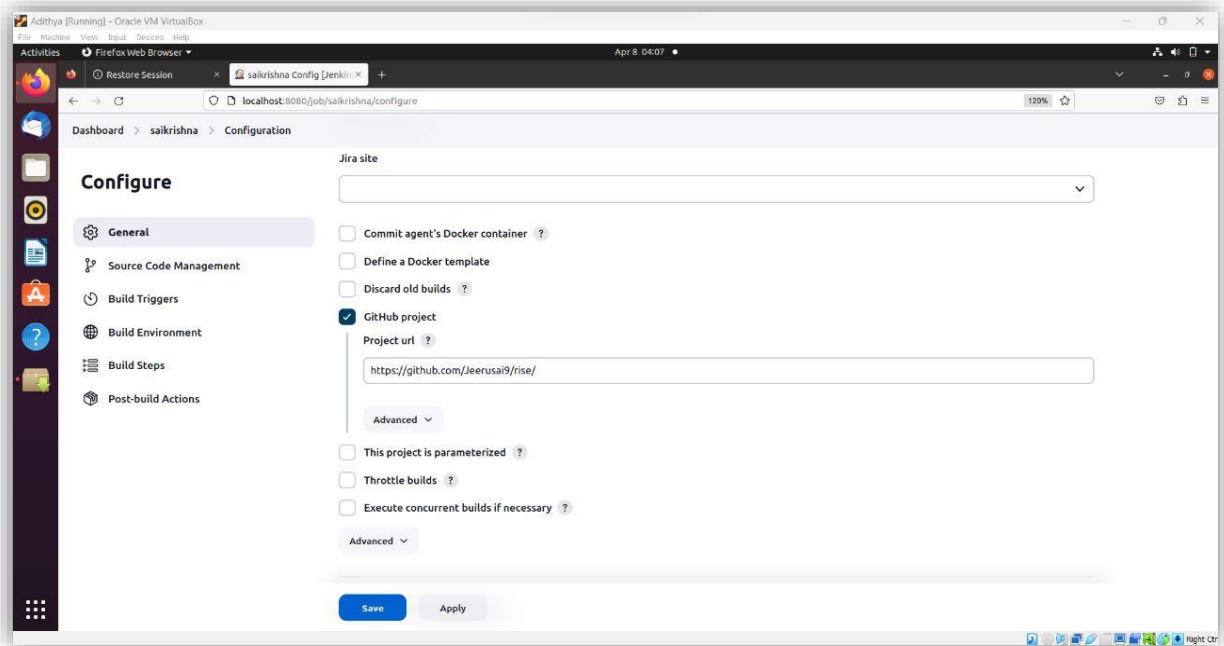
## CI/CD PIPELINE WITH JENKINS



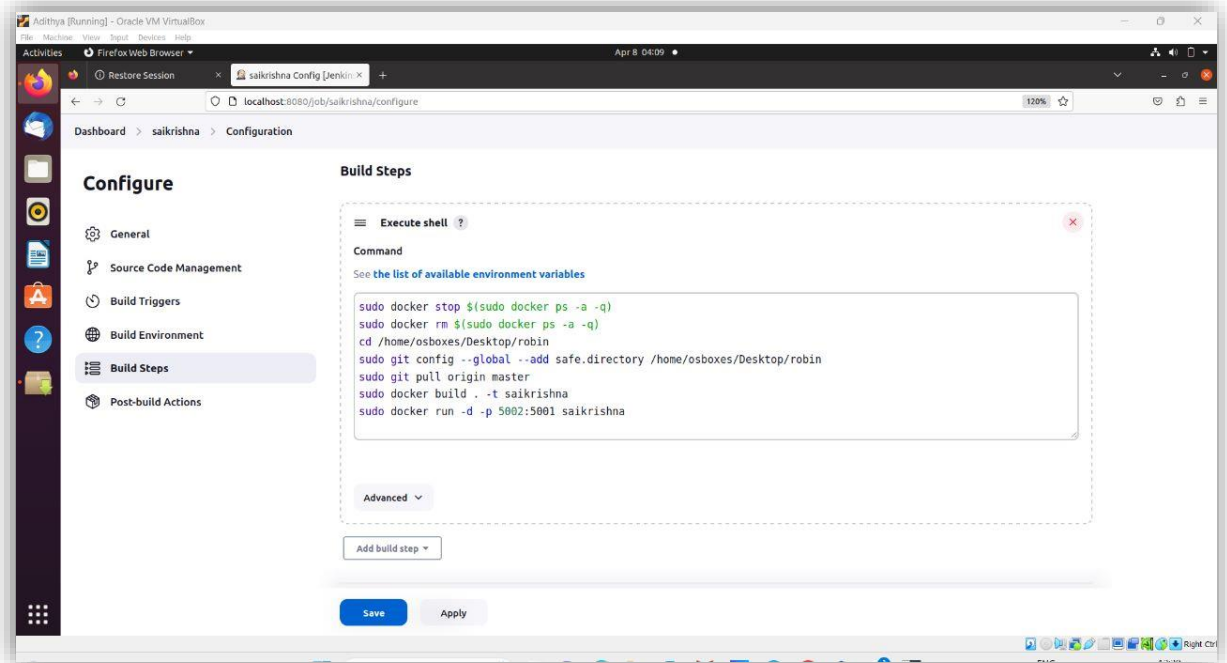
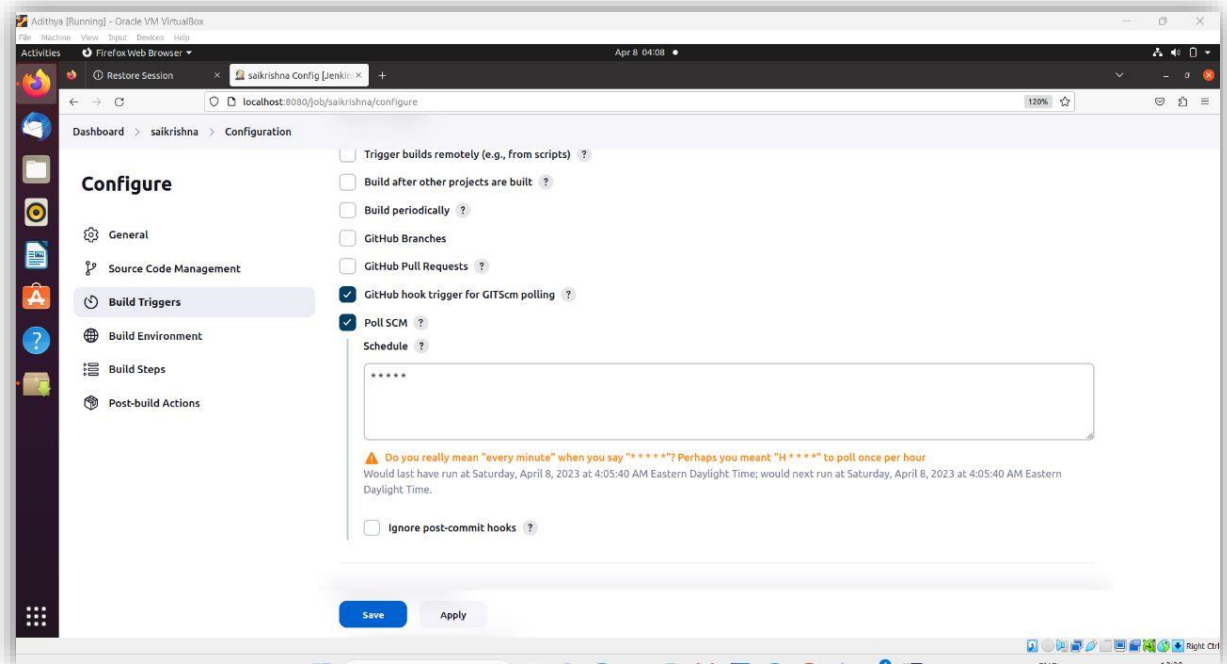
From there we arrive to configuration, there give the below mentioned options and commands.



## CI/CD PIPELINE WITH JENKINS



## CI/CD PIPELINE WITH JENKINS

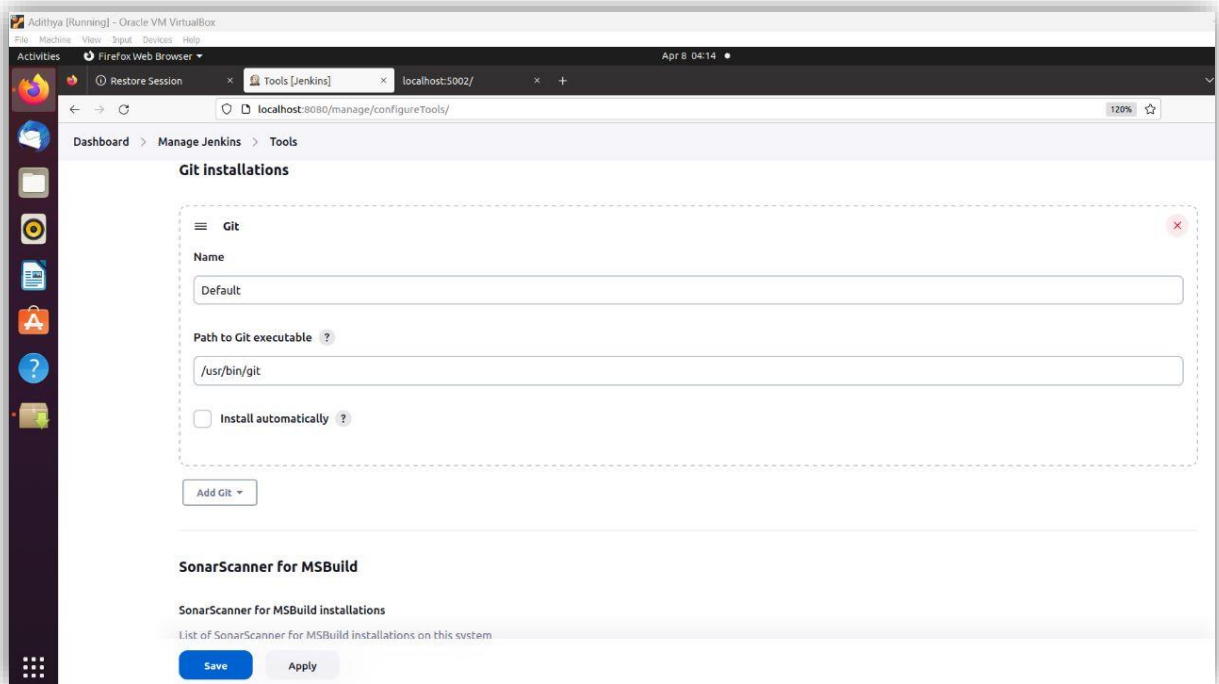


- Apply it and save it.
- Now add git executable path.
- We can find the path using below command.

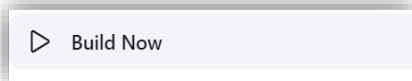
\$ whereis git



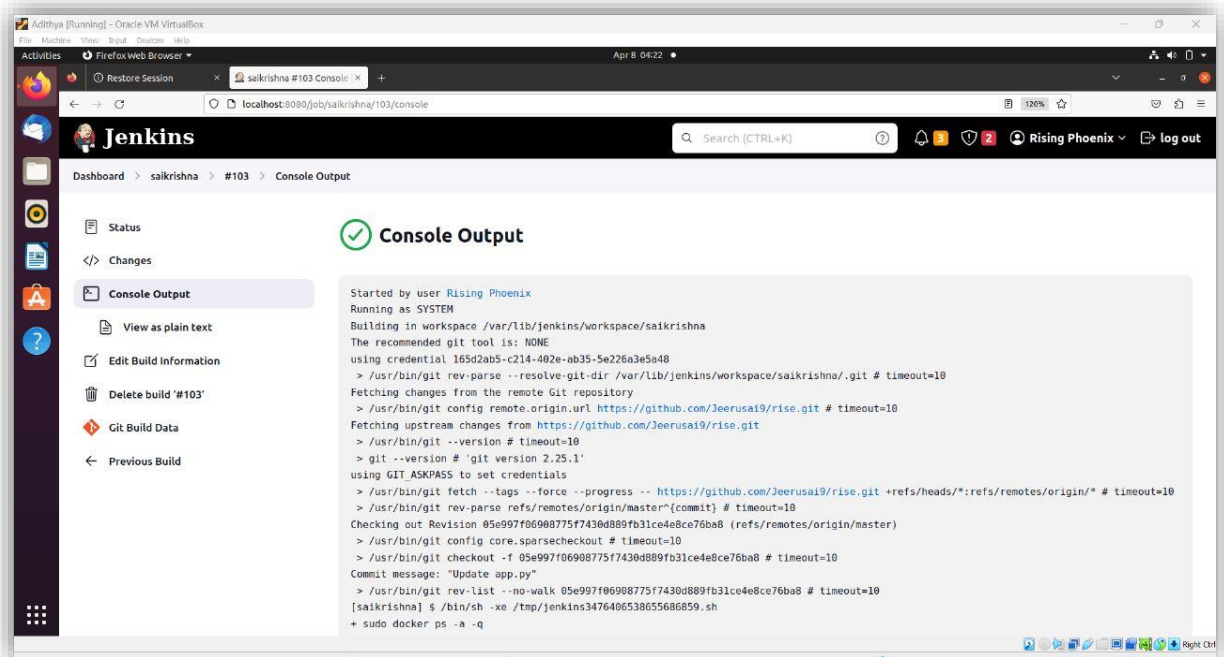
## CI/CD PIPELINE WITH JENKINS



- Apply and save.
- After completing this build it. you will see a successful console output as shown

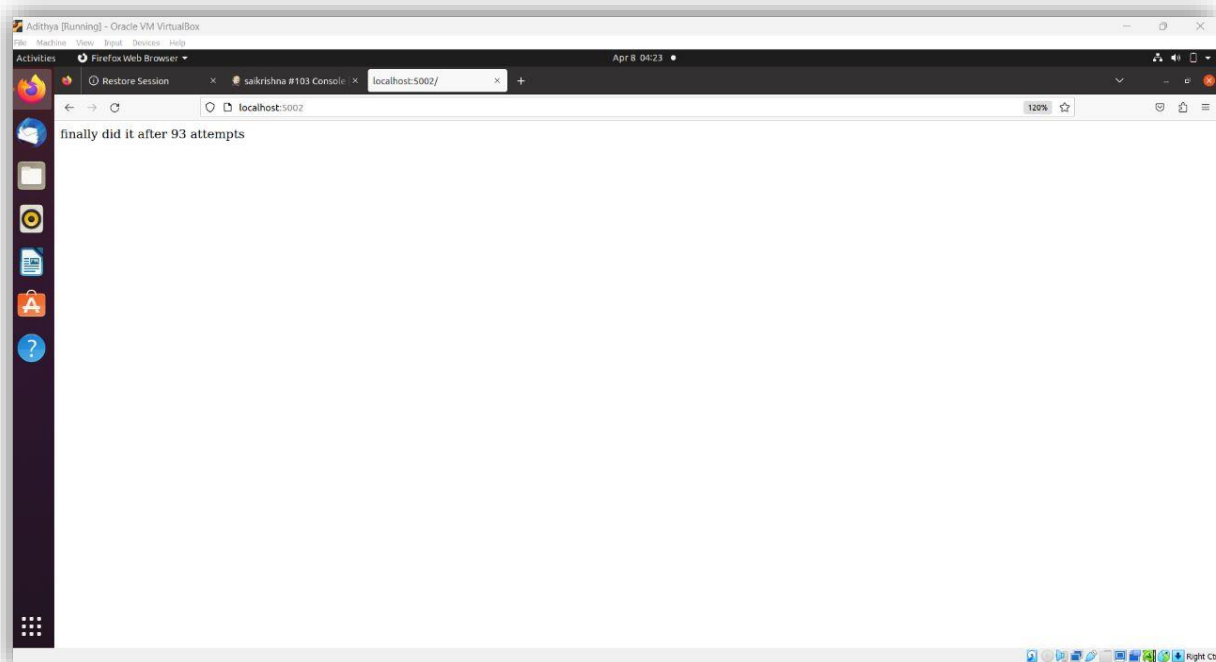
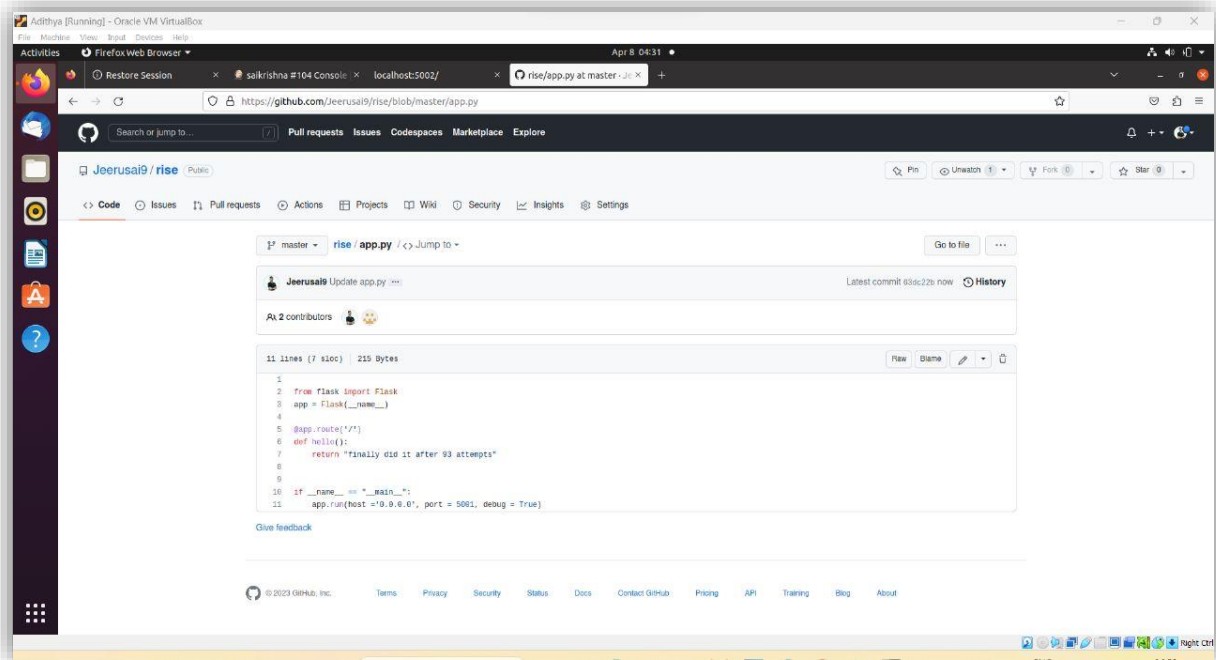


\*\*\*\*\***OUTPUT**\*\*\*\*\*



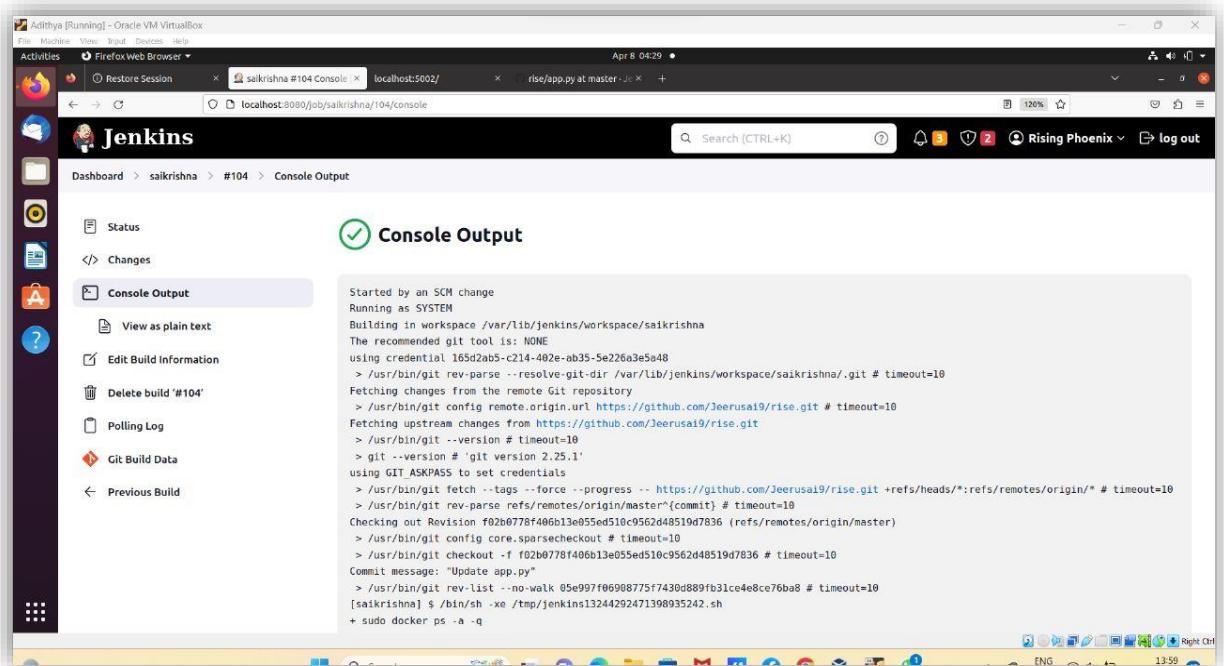
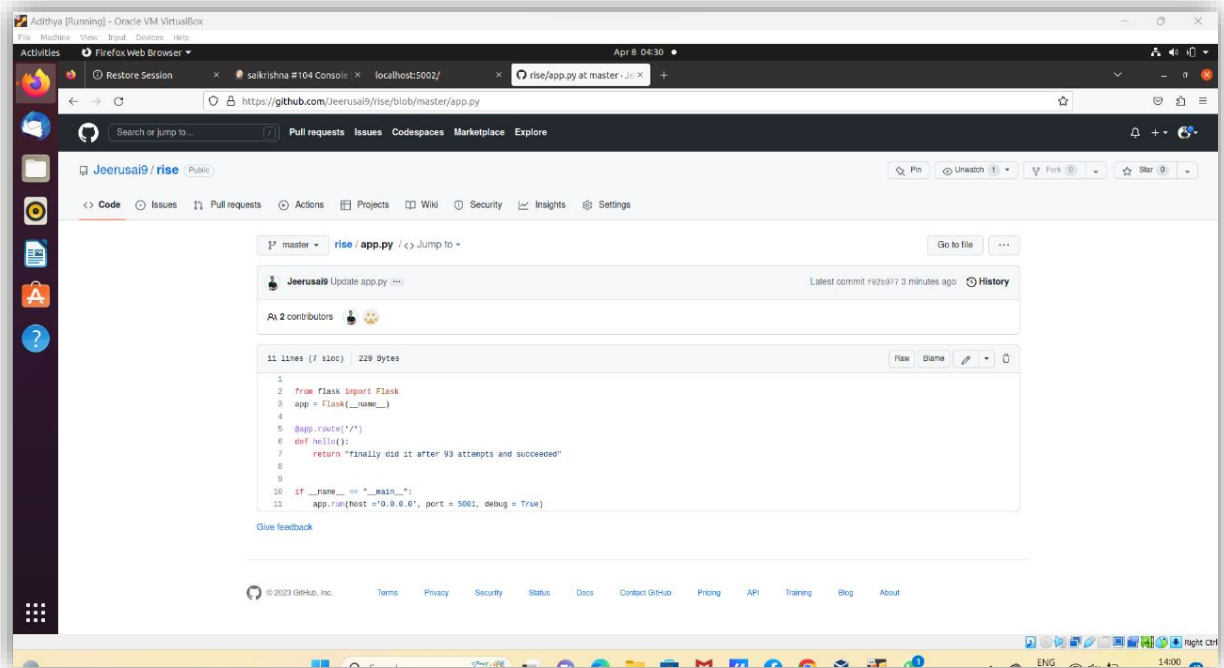


## CI/CD PIPELINE WITH JENKINS



- The output will be shown in the localhost as per the git repository app.py file .
- Now make changes in the code.We will observe the build getting automated and Parallely the localhost webpage getting changed as shown below.

# CI/CD PIPELINE WITH JENKINS



## CI/CD PIPELINE WITH JENKINS

