SI report configuring jenkins+nexus

BY:

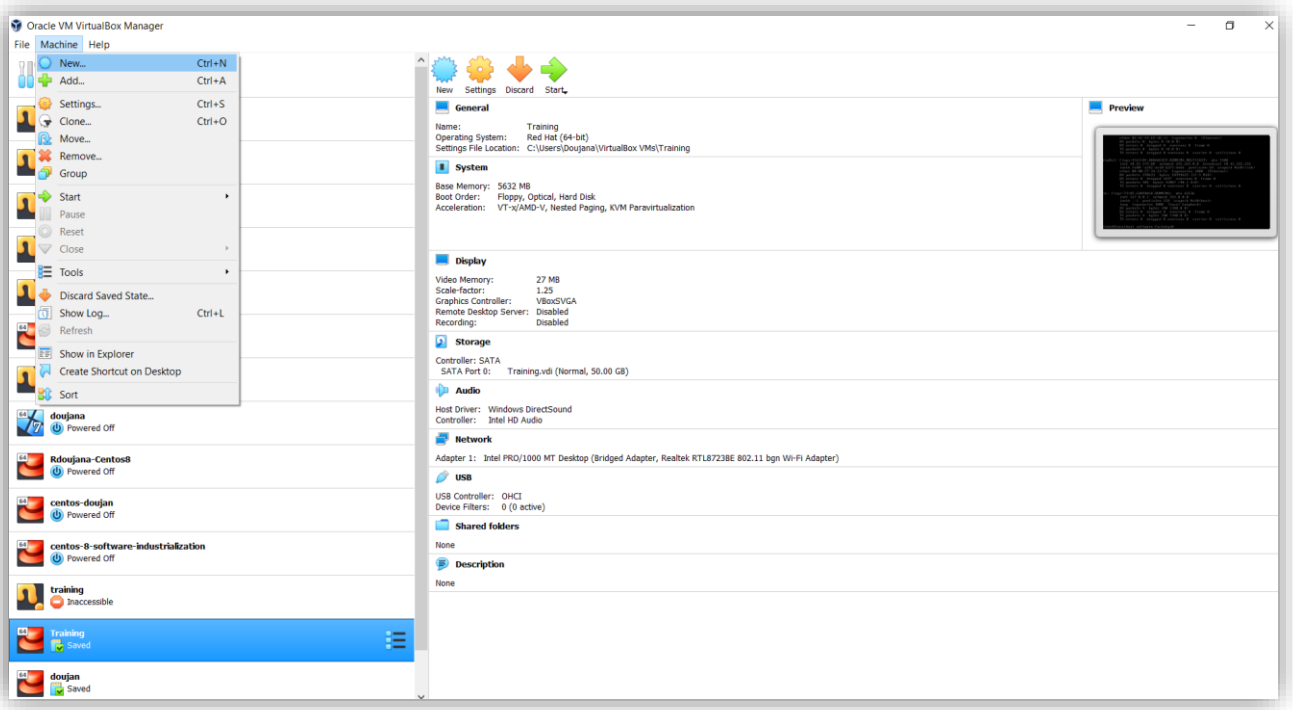
DOUJANA ALHABIB

Introduction:

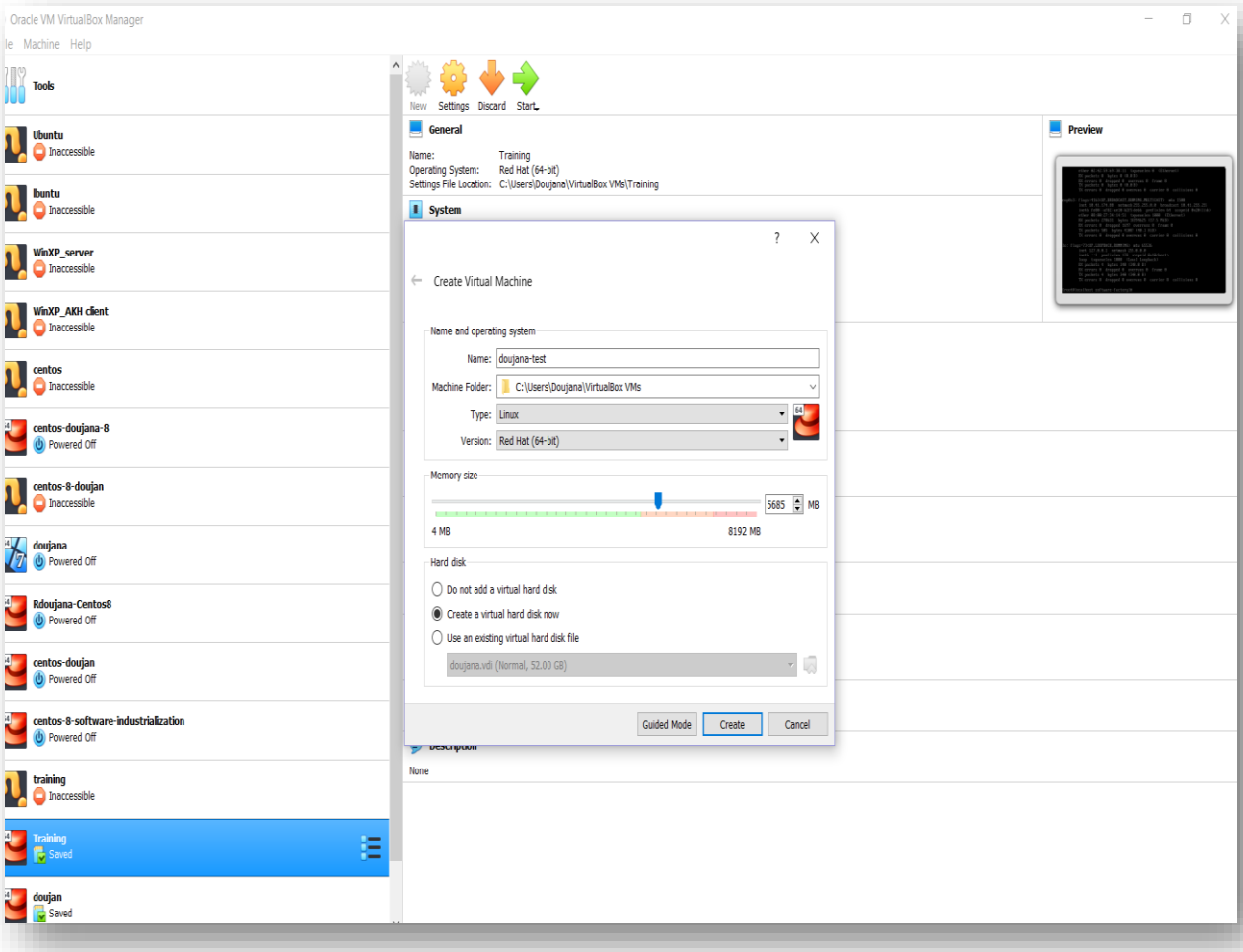
This project has part of the configuration we did in software Industrialization course in order to set up a software factory, and This configuration includes the setup of a CentOS Operating System and a set of docker containers for now jenkins and nexus, each containing a brick of the software factory.

-Setting up (centos8):

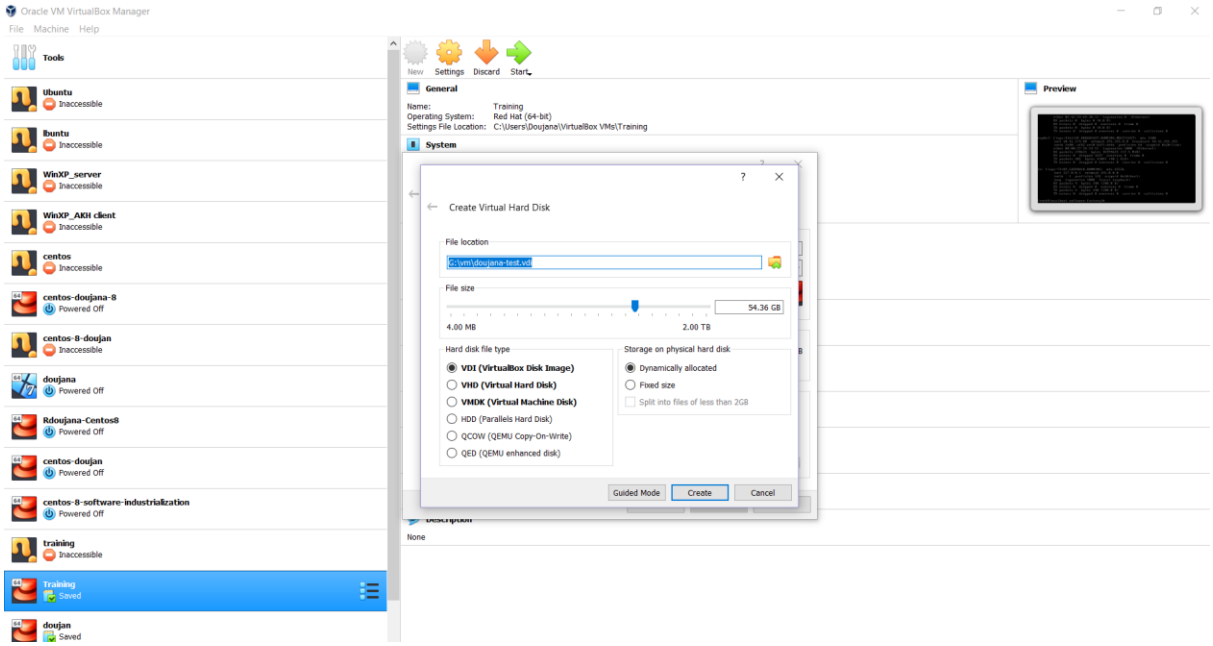
Now, we will create a virtual machine so to create a new virtual machine you go to machine and then you choose new for creating a virtual machine.



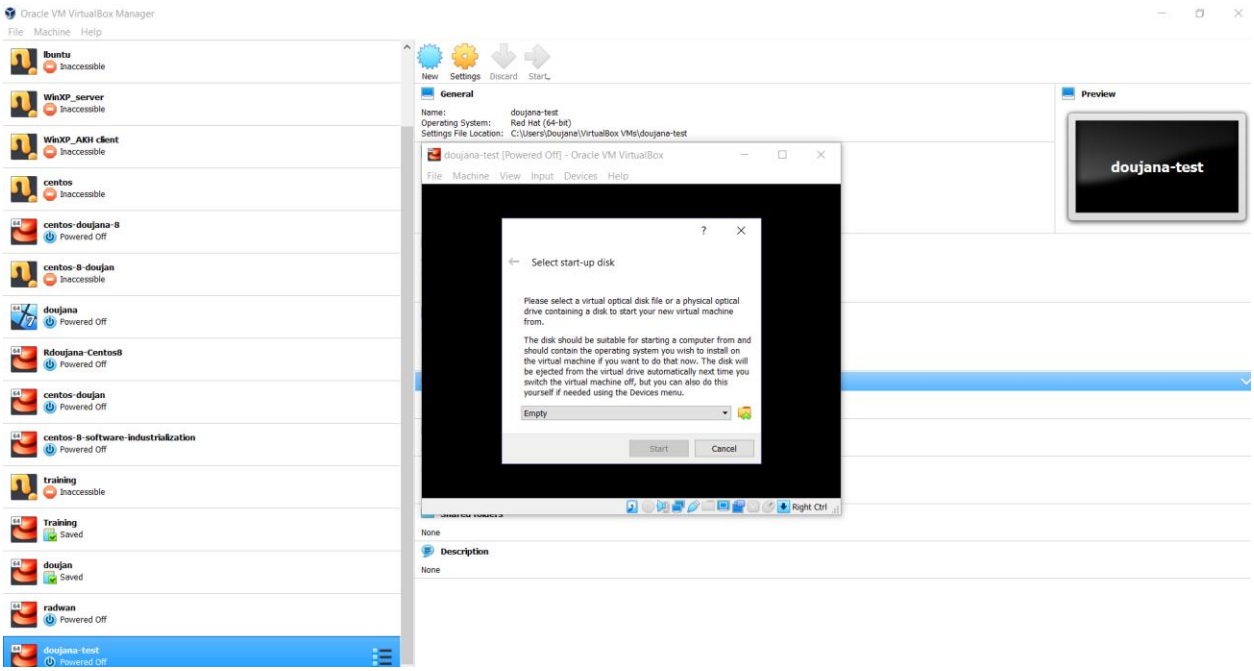
And then as we can see we have many things to fill so for the name now as an example I will name it **doujana-test**, of course you can write the name you want, you should specify where you will store your virtual machine it is better to choose a permanent place in which you will save your machine, for the memory size make the slider cross the green part, for the type we will for this tutorial choose Linux and them for the version I will choose red hat (64-bit) as it is shown in the picture below.



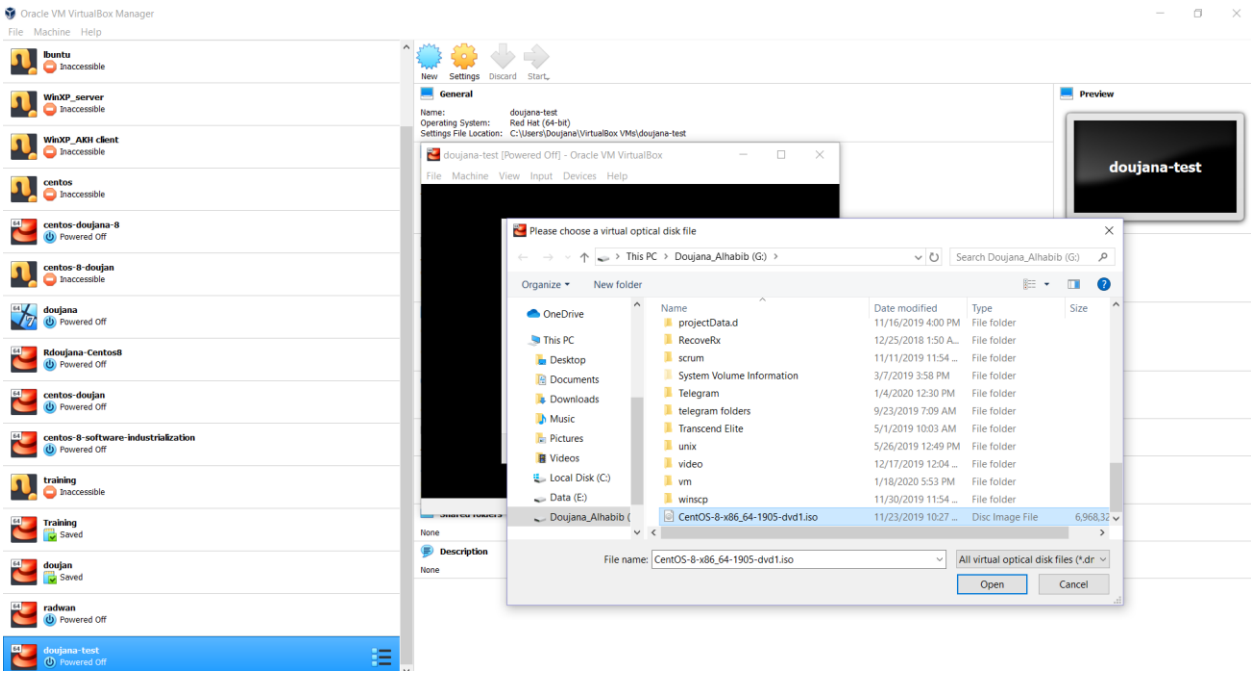
And then we need to choose the hard disk type for now we will choose the image we will create the image from the ISO, for the storage we will choose the dynamic allocation and the other choose dynamic option, for the file size we will put it as **50 GB** it will be better.



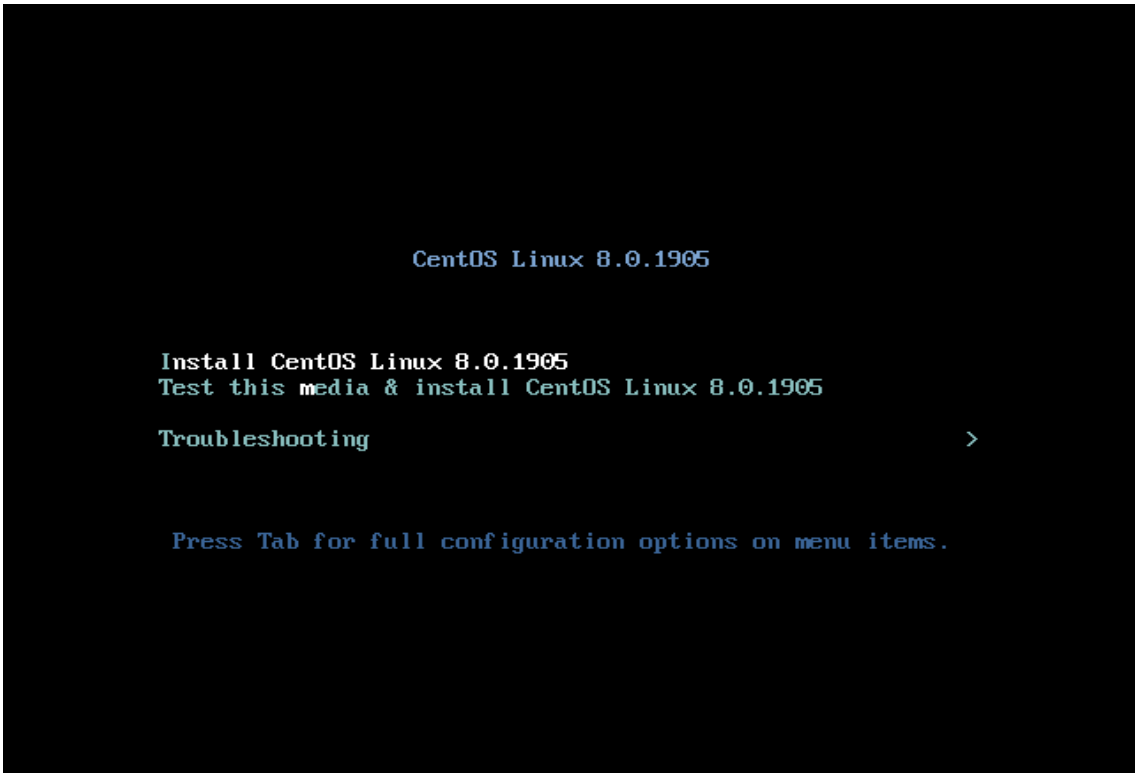
Click next then you will have a window asking you to browse the iso in which you will install your operating system centos8.



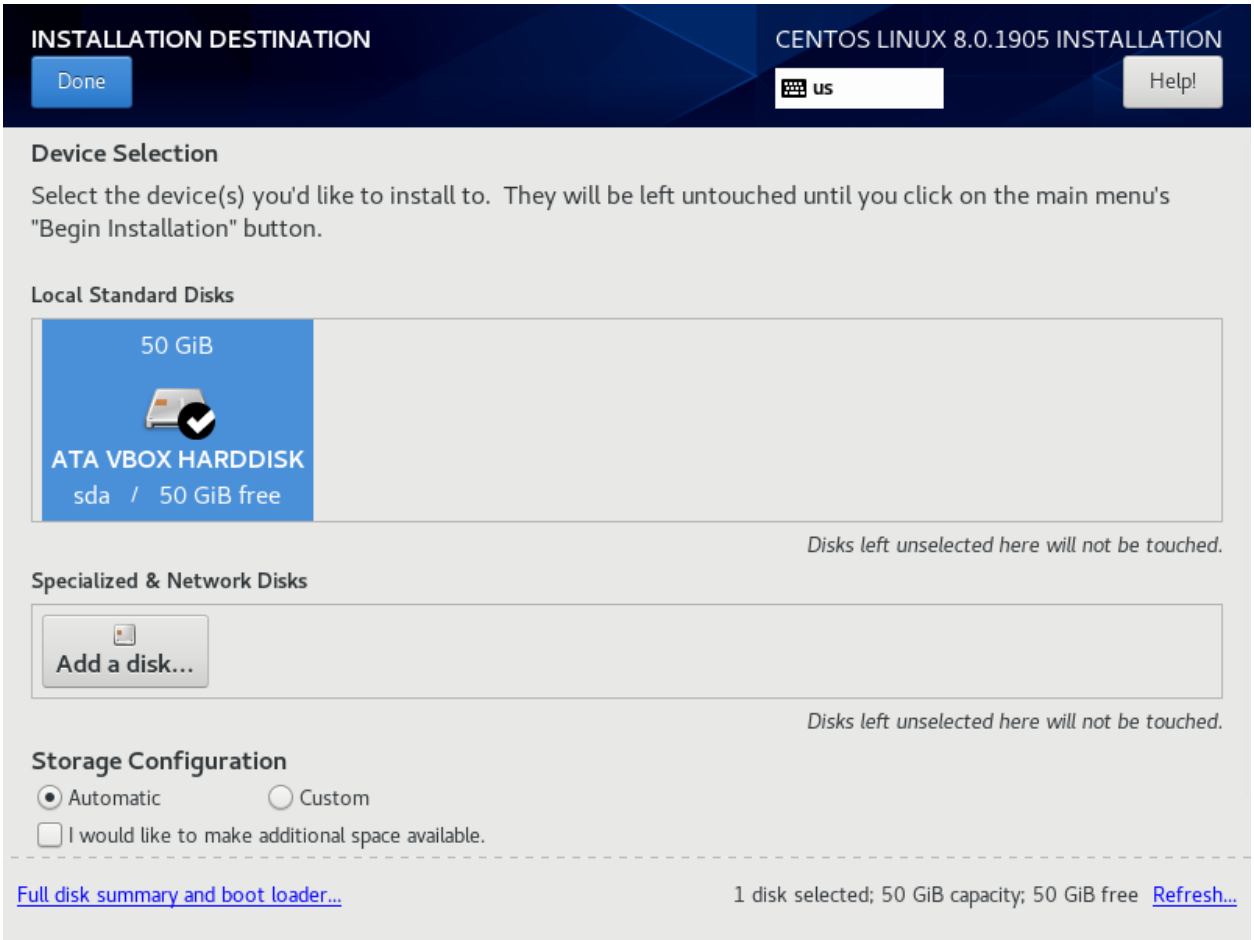
And then as we can see I browsed to get the iso file, open it and start the process for downloading.



And that’s it for now we will choose install and it will start installing.

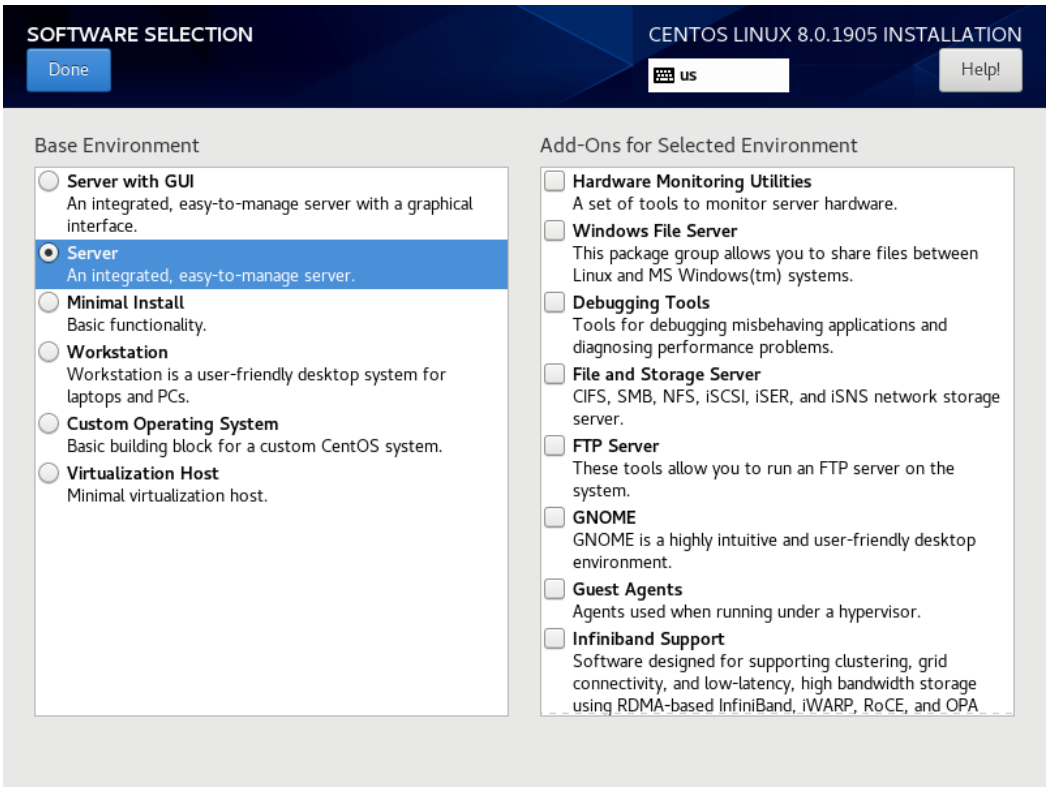


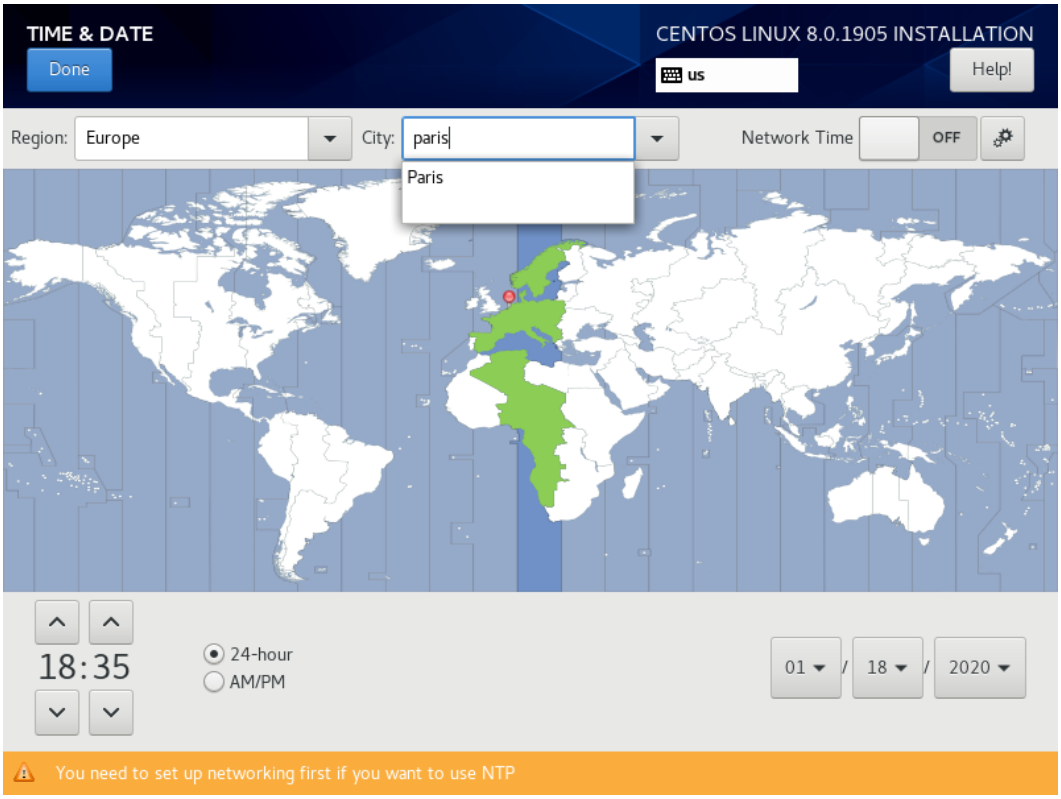
And then we have in here for the destination we will choose the local standard disk then click Done.



Then we will choose the language, the time you can choose depending on your place, for me

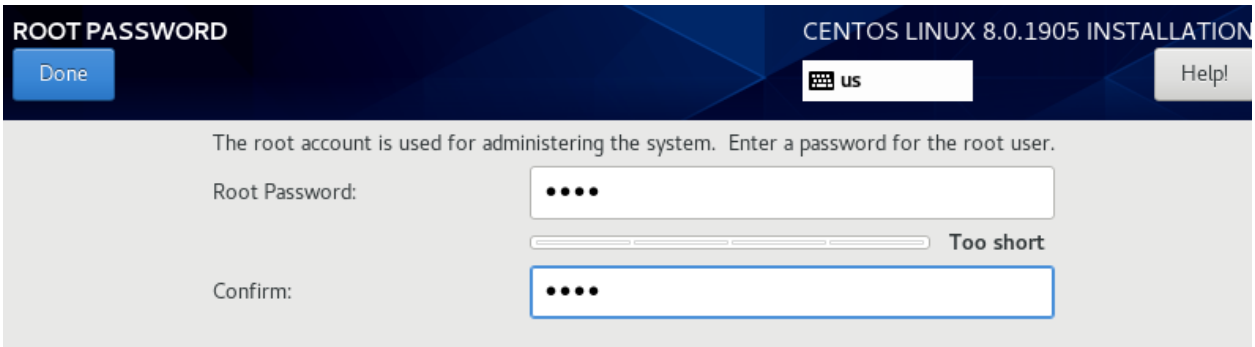
I choose server only which means everything will be done by using the terminal. and then we will have to choose the country in which we will have to set the time line so for me it is PARIS.

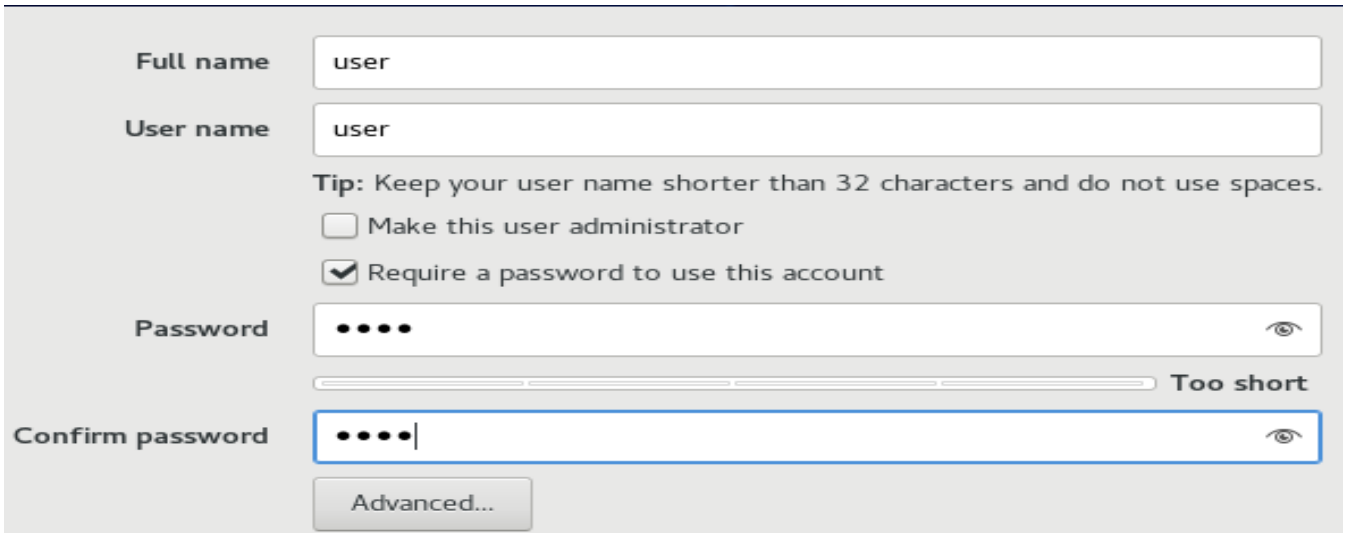

SOFTWARE SELECTION
Done
CENTOS LINUX 8.0.1905 INSTALLATION
us
Help!
Base Environment
Server with GUI
An integrated, easy-to-manage server with a graphical interface.
Server
An integrated, easy-to-manage server.
Minimal Install
Basic functionality.
Workstation
Workstation is a user-friendly desktop system for laptops and PCs.
Custom Operating System
Basic building block for a custom CentOS system.
Virtualization Host
Minimal virtualization host.
Add-Ons for Selected Environment
Hardware Monitoring Utilities
A set of tools to monitor server hardware.
Windows File Server
This package group allows you to share files between Linux and MS Windows(tm) systems.
Debugging Tools
Tools for debugging misbehaving applications and diagnosing performance problems.
File and Storage Server
CIFS, SMB, NFS, iSCSI, iSER, and iSNS network storage server.
FTP Server
These tools allow you to run an FTP server on the system.
GNOME
GNOME is a highly intuitive and user-friendly desktop environment.
Guest Agents
Agents used when running under a hypervisor.
Infiniband Support
Software designed for supporting clustering, grid connectivity, and low-latency, high bandwidth storage using RDMA-based InfiniBand, iWARP, RoCE, and OPA


TIME & DATE
Done
CENTOS LINUX 8.0.1905 INSTALLATION
us
Help!
Region:
Europe
City:
paris
Paris
Network Time
OFF
18:35
24-hour
AM/PM
01
18
2020
You need to set up networking first if you want to use NTP

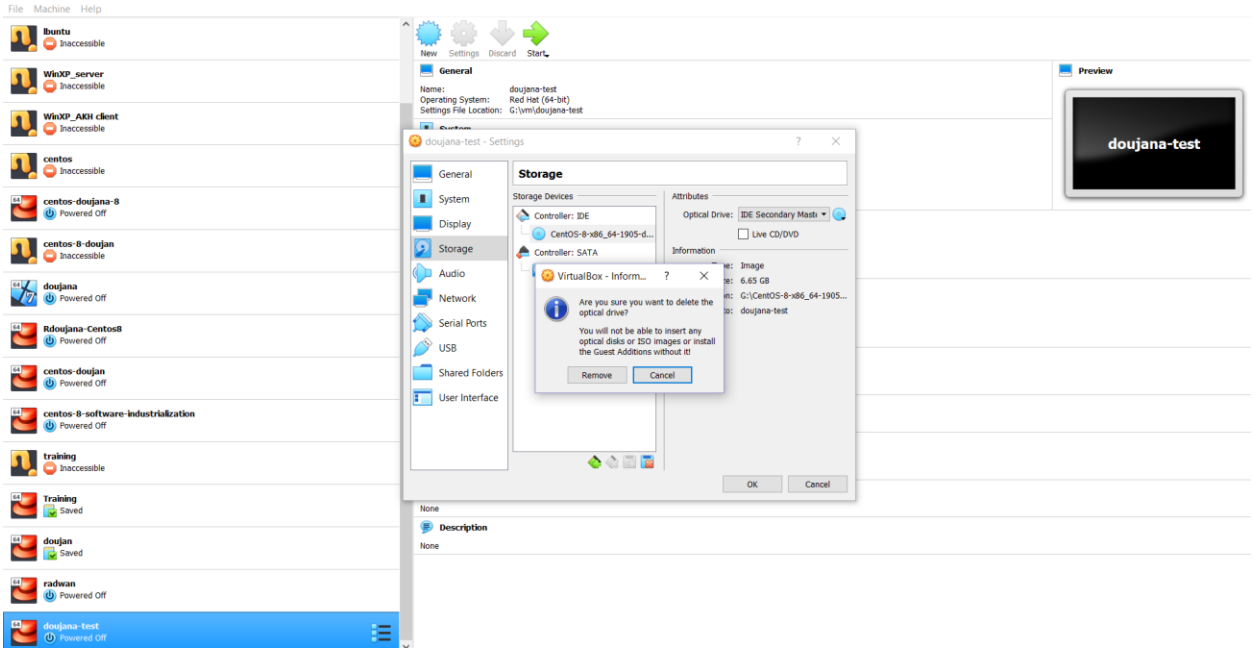
So , the downloading will start as we can see the root password should be set .

And then we go to set the root and the user creation for me it is root root and for the user it is user user I mean the **name: root** and the **password:root** same for the user.





After the installation is finished we click on reboot and then as a tip we should delete the image otherwise the installation process we be started from the beginning so after clicking rebooting you should delete it and restart the virtual machine and everything will be working fine.



Then start the virtual machine again you will choose the first option and then we will have to write the login name and the password which is in my case is name: root and password: root

```
CentOS Linux (4.18.0-80.el8.x86_64) 8 (Core)
CentOS Linux (0-rescue-e932ef704d89429dbea96e611653b66b) 8 (Core)
```

```
CentOS Linux 8 (Core)
Kernel 4.18.0-80.el8.x86_64 on an x86_64

Activate the web console with: systemctl enable --now cockpit.socket

localhost login: root
Password:
[root@localhost ~]#
```

Okay perfect now everything is set, we will start to build our software factory, first we need to create a folder in which it will set our environment int the file system.

```
CentOS Linux 8 (Core)
Kernel 4.18.0-80.el8.x86_64 on an x86_64

Activate the web console with: systemctl enable --now cockpit.socket

localhost login: root
Password:
[root@localhost ~]# cd ..
[root@localhost /]# mkdir test_
```

So, now we have internet connection to make sure that we have it in our virtual machine we use the command **ifup enp0s3** which the default one and then we will write **ifconfig** to check the ip address we get in which we will have internet connection.

```
CentOS Linux 8 (Core)
Kernel 4.18.0-80.el8.x86_64 on an x86_64

Activate the web console with: systemctl enable --now cockpit.socket

localhost login: root
Password:
Last login: Sat Jan 18 19:47:18 on tty1
[root@localhost ~]# ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        ether 08:00:27:be:df:5b  txqueuelen 1000  (Ethernet)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 0  bytes 0 (0.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 0  bytes 0 (0.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

[root@localhost ~]# ifup enp0s3
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnecti
on/1)
[root@localhost ~]#
```

Now the first thing we have to install the docker ce then docker-compose then docker.

-Installing docker CE:

Docker is a daemon-based container engine which allows us to deploy applications inside containers. With the release of CentOS 8, docker package has been removed from their default package repositories, docker has been replaced with **podman** and **buildah**.

Following the following step will guide you on how to install and use community version of docker on CentOS 8 system.

Docker is available in two versions,

- Docker CE (Community Edition)
- Enterprise Edition (EE)

In this guide I will install Docker CE (Community Edition) on CentOS.

System Requirements for Docker CE

- Minimal CentOS 8
- Sudo or root privileges
- Internet Connection as I mentioned before, it will be by command ifup enp0s3

-Installation Steps of Docker CE on CentOS 8

Step:1) Enable Docker CE Repository:
As I said above, docker packages are not available anymore on CentOS 8, so run following dnf command to enable Docker CE package repository.

```
[root@localhost ~]# dnf config-manager --add-repo=https://download.docker.com/linux/centos/docker-ce
.repo
Adding repo from: https://download.docker.com/linux/centos/docker-ce.repo
[root@localhost ~]# _
```

Step:2) Install Docker CE using dnf command:

Once the docker ce repo has been configured successfully then run following command to verify which version of docker is available for installation.

```
docker-ce.x86_64                          3:19.03.5-3.el7                          docker-ce-stable
[root@localhost ~]# cd ..
[root@localhost /]# mkdir test
mkdir: cannot create directory 'test': File exists
[root@localhost /]# cd test
[root@localhost test]# dnf config-manager --add-repo=https://download.docker.com/linux/centos/docker
-ce.repo
Adding repo from: https://download.docker.com/linux/centos/docker-ce.repo
[root@localhost test]# dnf list docker-ce
Last metadata expiration check: 0:02:15 ago on Sat 18 Jan 2020 07:57:10 PM CET.
Available Packages
docker-ce.x86_64                          3:19.03.5-3.el7                          docker-ce-stable
```

Now, use beneath dnf command to install latest version of docker,

```
[root@localhost test]# dnf install docker-ce --nobest -y
Docker CE Stable - x86_64                                          2.5 kB/s | 3.5 kB     00:01
Dependencies resolved.

 Problem: package docker-ce-3:19.03.5-3.el7.x86_64 requires containerd.io >= 1.2.2-3, but none of th
e providers can be installed
  - cannot install the best candidate for the job
  - package containerd.io-1.2.10-3.2.el7.x86_64 is excluded
  - package containerd.io-1.2.2-3.3.el7.x86_64 is excluded
  - package containerd.io-1.2.2-3.el7.x86_64 is excluded
  - package containerd.io-1.2.4-3.1.el7.x86_64 is excluded
  - package containerd.io-1.2.5-3.1.el7.x86_64 is excluded
  - package containerd.io-1.2.6-3.3.el7.x86_64 is excluded
================================================================================================
 Package               Arch           Version                     Repository                Size
================================================================================================
Installing:
 docker-ce             x86_64         18.06.3.ce-3.el7            docker-ce-stable          41 M
Installing dependencies:
 libcgroup             x86_64         0.41-19.el8                 BaseOS                    70 k
 libtool-ltdl          x86_64         2.4.6-25.el8                BaseOS                    58 k
Skipping packages with broken dependencies:
 docker-ce             x86_64         3:19.03.5-3.el7             docker-ce-stable          24 M

Transaction Summary
```

After the installation of docker, start and enable its service using the following systemctl commands:

-systemctl start docker

-systemctl enable docker

After the installation of docker, start and enable its service using the following systemctl commands.

```
[root@localhost test]# systemctl start docker
[root@localhost test]# systemctl enable docker
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/syste
/docker.service.
[root@localhost test]#
```

Run the following command to verify installed docker version

```
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /usr/lib/systemd/syster
/docker.service.
[root@localhost test]# docker --version
Docker version 18.06.3-ce, build d7080c1
[root@localhost test]#
```

Step:3) Verify and test Docker CE Engine
To verify whether docker ce engine has been setup correctly or not, try to spin up a "hello-world" container using below docker command,

```
[root@localhost test]# docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest: sha256:9572f7cdcee8591948c2963463447a53466950b3fc15a247fcad1917ca215a2f
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

[root@localhost test]# _
```

As we can see the informational message in above output which confirms docker engine is setup correctly on CentOS 8**. Let's move to docker compose installation**.

Step:4) Install Docker Compose

As we know that docker compose is used to link multiple containers using a single command. In other words, Docker Compose is useful where we need to launch multiple containers and these containers depends on each other. Examples like launching LAMP/ LEMP applications (WordPress & Joomla etc) insider containers.

Run the following commands to install docker compose on CentOS 8.

```
[root@localhost test]# dnf install curl -y
CentOS-8 - AppStream                                             1.9 kB/s | 4.3 kB     00:02
CentOS-8 - Base                                                  1.5 kB/s | 3.8 kB     00:02
CentOS-8 - Extras                                                684  B/s | 1.5 kB     00:02
Docker CE Stable - x86_64                                        2.4 kB/s | 3.5 kB     00:01
Package curl-7.61.1-8.el8.x86_64 is already installed.
Dependencies resolved.
================================================================================================
 Package              Arch                Version                     Repository           Size
================================================================================================
Upgrading:
 curl                 x86_64              7.61.1-11.el8               BaseOS               353 k
 libcurl              x86_64              7.61.1-11.el8               BaseOS               298 k

Transaction Summary
================================================================================================
Upgrade  2 Packages

Total download size: 651 k
Downloading Packages:
(1/2): curl-7.61.1-11.el8.x86_64.rpm                             408 kB/s | 353 kB     00:00
(2/2): libcurl-7.61.1-11.el8.x86_64.rpm                          314 kB/s | 298 kB     00:00
------------------------------------------------------------------------------------------------
Total                                                            285 kB/s | 651 kB     00:02
Running transaction check
Transaction check succeeded.
Running transaction test
```

```
Transaction Summary
================================================================================================
Upgrade  2 Packages

Total download size: 651 k
Downloading Packages:
(1/2): curl-7.61.1-11.el8.x86_64.rpm                             408 kB/s | 353 kB     00:00
(2/2): libcurl-7.61.1-11.el8.x86_64.rpm                          314 kB/s | 298 kB     00:00
------------------------------------------------------------------------------------------------
Total                                                            285 kB/s | 651 kB     00:02
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :                                                                        1/1
  Upgrading        : libcurl-7.61.1-11.el8.x86_64                                           1/4
  Upgrading        : curl-7.61.1-11.el8.x86_64                                              2/4
  Cleanup          : curl-7.61.1-8.el8.x86_64                                               3/4
  Cleanup          : libcurl-7.61.1-8.el8.x86_64                                            4/4
  Running scriptlet: libcurl-7.61.1-8.el8.x86_64                                            4/4
  Verifying        : curl-7.61.1-11.el8.x86_64                                              1/4
  Verifying        : curl-7.61.1-8.el8.x86_64                                               2/4
  Verifying        : libcurl-7.61.1-11.el8.x86_64                                           3/4
  Verifying        : libcurl-7.61.1-8.el8.x86_64                                            4/4

Upgraded:
  curl-7.61.1-11.el8.x86_64                        libcurl-7.61.1-11.el8.x86_64

Complete!
[root@localhost test]# curl -L "https://github.com/docker/compose/releases/download/1.25.0/docker-co
mpose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   617    0   617    0     0    858      0 --:--:-- --:--:-- --:--:--   856
 34 16.2M   34 5762k    0     0   424k      0  0:00:39  0:00:13  0:00:26  553k_
```

Then you have to set the executable permissions for docker compose binary by writing.

chmod +x /usr/local/bin/docker-compose

then we have to Verify the docker compose version by running the following command.

```
[root@localhost test]# chmod +x /usr/local/bin/docker-compose
[root@localhost test]# docker-compose --version
docker-compose version 1.25.0, build 0a186604
[root@localhost test]#
```

-setting up the environment:

So now we go and create a folder we name docker, we do ll to check everything is fine, then we create inside docker another folder let's name it software-factory the point is when we will deploy this software-factory in another pc and we do docker-compose up it should work with the same data inside it.

```
[root@localhost /]# mkdir docker
[root@localhost /]# cd docker
[root@localhost docker]# ll
total 0
[root@localhost docker]# mkdir software-factory
[root@localhost docker]# ll
total 0
drwxr-xr-x. 2 root root 6 Jan 18 21:07 software-factory
[root@localhost docker]# _
```

So now we will create a **docker-compose. Yml**, so in here yml is a configuration file we will get sections and this file relies on indentation, this file always starts with version and then for me we will have the first services we will start by jenkins service in here I name it jenkins then we have the ports we are using 80 because it is a web and we said for the ports actually it is a port binding it will be 10380:8080

And for the other service which is nexus I chose he port 8081 and the service number is 10680 so eventually we will have for our services like this:

-jenkins: 10380:8080

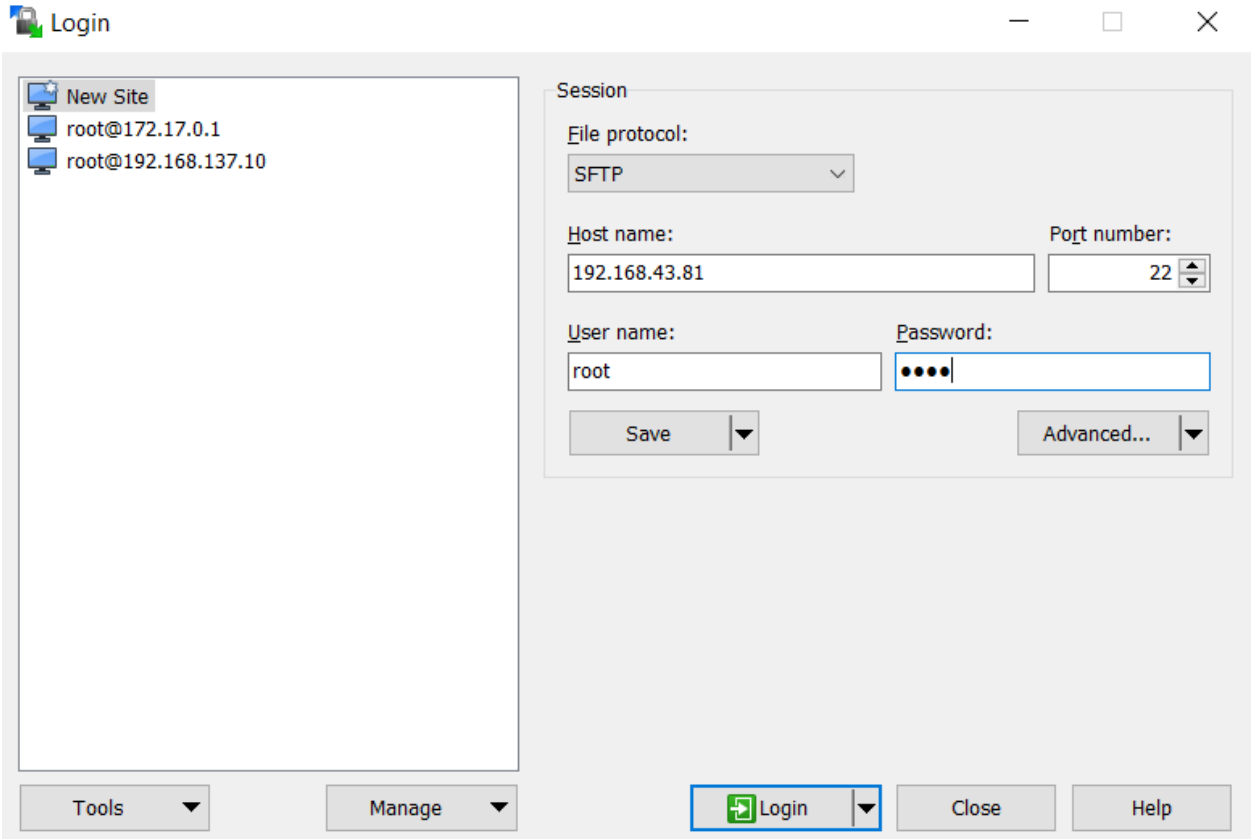
-nexus: 10680:8081

So for now we went to software-factory directory and then to create the file simply I wrote

touch docker-compose.yml, now as we can see it is created.

```
[root@localhost software-factory]# ll
total 4
-rw-r--r--. 1 root root 144 Jan 19 09:17 docker-compose.yml
drwxr-xr-x. 2 root root   6 Jan 19 09:18 jenkins
```

we write **nano docker-compose.yml** to write the lines for our services nexus and jenkins for me to make things much easier I used the winscp software to make this winscp connects with your virtual machine you have to write ifconfig you will get the ip you have for me it is **192.168.43.81**



Now after logging on we go to our docker-compose.yml in this way:

./docker/software-factory/docker-compose.yml

Then we have these lines to configure jenkins and nexus in the .yml file.

/docker/software-factory/docker-compose.yml - root@192.168.43.81 - Editor - WinSCP

```yaml
version: '3.5'
services:
  jenkins:
    image: "jenkins/jenkins:lts"
    volumes:
      - "./jenkins:/var/jenkins_home"
    ports:
      - "10380:8080"

    restart: always
  nexus:
    image: "sonatype/nexus:oss"
    ports:
      - "10680:8081"
    volumes:
      - "./nexus:/sonatype-work"
```

Now just to understand what is written in this file:

It always begins with a version definition then the version in our case it is 3.5 then we have the list of our services, so our first service is jenkins, so we took the image that we will pull the service from then we have the ports, we have for each different port of course for nexus it is going to be another port.

so the first part of the port we will make it representative for the group of application so project1 which has project 1 everything is related to this group of application and then we attribute a number for this service and then the kind of service we have it web service so it is 80 port which indicate it is a web application if we want to do the nexus it will run with 10680 and so on so we should select and order for our software-factory, so for the port it is internal to external in other words from the host to the container, Preserve volume data when containers are created, Compose preserves all volumes used by your services. When docker-compose up runs, if it finds any containers from previous runs, it copies the volumes from the old container to the new container. This process ensures that any data you've created in volumes isn't lost.

Then as we can see why we have the var in the url for each service, compose supports variables in the Compose file. You can use these variables to customize your composition for different environments, or different users.

eventually our service it will be like this:

-jenkins: 10380:8080

-nexus: 10680:8081

-Starting the services jenkins+nexus:

So now we have to start the docker service

```
software-factory_jenkins_1 exited with code 1
[root@localhost software-factory]# service docker start
Redirecting to /bin/systemctl start docker.service
[root@localhost software-factory]#
```

then we will start our services using the command: **docker-compose up**

```
[root@localhost software-factory]# docker-compose up
Starting software-factory_jenkins_1 ... done
Attaching to software-factory_jenkins_1
jenkins_1  | Running from: /usr/share/jenkins/jenkins.war
jenkins_1  | webroot: EnvVars.masterEnvVars.get("JENKINS_HOME")
jenkins_1  | 2020-01-19 08:25:51.824+0000 [id=1]        INFO    org.eclipse.jetty.util.log.Lo
alized: Logging initialized @632ms to org.eclipse.jetty.util.log.JavaUtilLog
jenkins_1  | 2020-01-19 08:25:52.034+0000 [id=1]        INFO    winstone.Logger#logInternal:
ng extraction from war file
jenkins_1  | 2020-01-19 08:25:53.569+0000 [id=1]        WARNING o.e.j.s.handler.ContextHandle
ntextPath: Empty contextPath
jenkins_1  | 2020-01-19 08:25:53.666+0000 [id=1]        INFO    org.eclipse.jetty.server.Serv
art: jetty-9.4.22.v20191022; built: 2019-10-22T13:37:13.455Z; git: b1e6b55512e008f7fbdf1cbea4
6d1073b; jvm 1.8.0_232-b09
jenkins_1  | 2020-01-19 08:25:54.161+0000 [id=1]        INFO    org.eclipse.jetty.util.TypeUt
nit>: JVM Runtime does not support Modules
jenkins_1  | 2020-01-19 08:25:54.182+0000 [id=1]        INFO    o.e.j.w.StandardDescriptorPro
visitServlet: NO JSP Support for /, did not find org.eclipse.jetty.jsp.JettyJspServlet
jenkins_1  | 2020-01-19 08:25:54.267+0000 [id=1]        INFO    o.e.j.s.s.DefaultSessionIdMan
Start: DefaultSessionIdManager workerName=node0
jenkins_1  | 2020-01-19 08:25:54.267+0000 [id=1]        INFO    o.e.j.s.s.DefaultSessionIdMan
Start: No SessionScavenger set, using defaults
jenkins_1  | 2020-01-19 08:25:54.273+0000 [id=1]        INFO    o.e.j.server.session.HouseKee
rtScavenging: node0 Scavenging every 660000ms
jenkins_1  | 2020-01-19 08:25:55.005+0000 [id=1]        INFO    hudson.WebAppMain#contextInit
: Jenkins home directory: /var/jenkins_home found at: EnvVars.masterEnvVars.get("JENKINS_HOME
jenkins_1  | 2020-01-19 08:25:55.234+0000 [id=1]        INFO    o.e.j.s.handler.ContextHandle
rt: Started w.@379ab47b{Jenkins v2.214,/,file:///var/jenkins_home/war/,AVAILABLE}{/var/jenkin
```

```
jenkins_1   | 2020-01-19 08:26:02.861+0000 [id=25]        INFO     o.s.c.s.AbstractApplicationContext#p
repareRefresh: Refreshing org.springframework.web.context.support.StaticWebApplicationContext@eb8287
8: display name [Root WebApplicationContext]; startup date [Sun Jan 19 08:26:02 UTC 2020]; root of c
ontext hierarchy
jenkins_1   | 2020-01-19 08:26:02.862+0000 [id=25]        INFO     o.s.c.s.AbstractApplicationContext#o
btainFreshBeanFactory: Bean factory for application context [org.springframework.web.context.support
.StaticWebApplicationContext@eb82878]: org.springframework.beans.factory.support.DefaultListableBean
Factory@3f5a5249
jenkins_1   | 2020-01-19 08:26:02.862+0000 [id=25]        INFO     o.s.b.f.s.DefaultListableBeanFactory
#preInstantiateSingletons: Pre-instantiating singletons in org.springframework.beans.factory.support
.DefaultListableBeanFactory@3f5a5249: defining beans [filter,legacy]; root of factory hierarchy
jenkins_1   | 2020-01-19 08:26:03.677+0000 [id=25]        INFO     jenkins.install.SetupWizard#init:
jenkins_1   |
jenkins_1   | *************************************************************
jenkins_1   | *************************************************************
jenkins_1   | *************************************************************
jenkins_1   |
jenkins_1   | Jenkins initial setup is required. An admin user has been created and a password genera
ted.
jenkins_1   | Please use the following password to proceed to installation:
jenkins_1   |
jenkins_1   | 15c50790f8fd4516a2d2eb83324053fc
jenkins_1   |
jenkins_1   | This may also be found at: /var/jenkins_home/secrets/initialAdminPassword
jenkins_1   |
jenkins_1   | *************************************************************
jenkins_1   | *************************************************************
jenkins_1   | *************************************************************
jenkins_1   |
jenkins_1   | 2020-01-19 08:26:21.451+0000 [id=39]        INFO     hudson.util.Retrier#start: The attem
pt #1 to do the action check updates server failed with an allowed exception:
jenkins_1   | java.net.UnknownHostException: updates.jenkins.io
jenkins_1   |     at java.net.AbstractPlainSocketImpl.connect(AbstractPlainSocketImpl.java:184)
jenkins_1   |     at java.net.SocksSocketImpl.connect(SocksSocketImpl.java:392)
jenkins_1   |     at java.net.Socket.connect(Socket.java:607)
jenkins_1   |     at sun.security.ssl.SSLSocketImpl.connect(SSLSocketImpl.java:666)
jenkins_1   |     at sun.net.NetworkClient.doConnect(NetworkClient.java:175)
```

This is for jenkins service now we have to do the same for the nexus service so

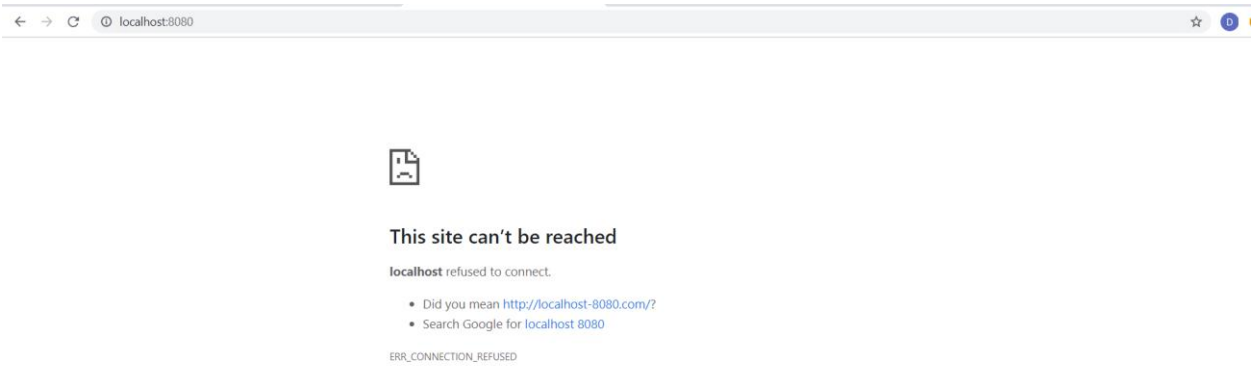
docker up nexus

```
a:1570)
jenkins_1   |     at sun.net.www.protocol.http.HttpURLConnection.getInputStream(HttpURLConnection.java
:1498)
jenkins_1   |     at sun.net.www.protocol.https.HttpsURLConnectionImpl.getInputStream(HttpsURLConnecti
onImpl.java:268)
jenkins_1   |     at hudson.model.DownloadService.loadJSON(DownloadService.java:114)
jenkins_1   |     at hudson.model.UpdateSite.updateDirectlyNow(UpdateSite.java:185)
jenkins_1   |     at hudson.PluginManager.checkUpdatesServer(PluginManager.java:1765)
jenkins_1   |     at hudson.util.Retrier.start(Retrier.java:62)
jenkins_1   |     at hudson.PluginManager.doCheckUpdatesServer(PluginManager.java:1736)
jenkins_1   |     at jenkins.DailyCheck.execute(DailyCheck.java:93)
jenkins_1   |     at hudson.model.AsyncPeriodicWork.lambda$doRun$0(AsyncPeriodicWork.java:100)
jenkins_1   |     at java.lang.Thread.run(Thread.java:748)
jenkins_1   | 2020-01-19 10:33:04.916+0000 [id=39]        INFO     hudson.util.Retrier#start: Calling t
he listener of the allowed exception 'updates.jenkins.io' at the attempt #1 to do the action check u
pdates server
jenkins_1   | 2020-01-19 10:33:04.919+0000 [id=39]        INFO     hudson.util.Retrier#start: Attempted
 the action check updates server for 1 time(s) with no success
jenkins_1   | 2020-01-19 10:33:04.941+0000 [id=39]        SEVERE   hudson.PluginManager#doCheckUpdatesS
erver: Error checking update sites for 1 attempt(s). Last exception was: UnknownHostException: updat
es.jenkins.io
jenkins_1   | 2020-01-19 10:33:04.943+0000 [id=39]        INFO     hudson.model.AsyncPeriodicWork#lambd
a$doRun$0: Finished Download metadata. 20,494 ms
^CGracefully stopping... (press Ctrl+C again to force)
Stopping software-factory_jenkins_1 ...
Killing software-factory_jenkins_1  ... done
[root@localhost software-factory]# docker-compose start jenkins
Starting jenkins ... done
[root@localhost software-factory]# docker-compose up nexus
Pulling nexus (sonatype/nexus:oss)...
oss: Pulling from sonatype/nexus
34971b2d1eb9: Downloading [==============>                                    ]  20.87MB/76.24MB
4fbc3bafa3d4: Download complete
d505d07e1d07: Downloading [================>                                  ]  17.22MB/55.41MB
c96d925fb046: Downloading [===>                                               ]  6.446MB/82.73MB
33b982b7be10: Waiting
_
```

Now, nexus is working too, as we can see form the picture below.

```
faultRepositoryRegistry - Added repository "Releases" [id=releases][contentClass=Maven2][mainFacet=o
rg.sonatype.nexus.proxy.maven.MavenHostedRepository]
nexus_1     | 2020-01-25 14:00:52,765+0000 INFO  [jetty-main-1]  org.sonatype.nexus.proxy.registry.De
faultRepositoryRegistry - Added repository "Snapshots" [id=snapshots][contentClass=Maven2][mainFacet
=org.sonatype.nexus.proxy.maven.MavenHostedRepository]
nexus_1     | 2020-01-25 14:00:52,962+0000 INFO  [jetty-main-1]  org.sonatype.nexus.proxy.registry.De
faultRepositoryRegistry - Added repository "3rd party" [id=thirdparty][contentClass=Maven2][mainFace
t=org.sonatype.nexus.proxy.maven.MavenHostedRepository]
nexus_1     | 2020-01-25 14:00:53,023+0000 INFO  [jetty-main-1]  org.sonatype.nexus.proxy.registry.De
faultRepositoryRegistry - Added repository "Central M1 shadow" [id=central-m1][contentClass=Maven1][
mainFacet=org.sonatype.nexus.proxy.maven.MavenShadowRepository]
nexus_1     | 2020-01-25 14:00:53,111+0000 INFO  [jetty-main-1]  org.sonatype.nexus.proxy.registry.De
faultRepositoryRegistry - Added repository "Public Repositories" [id=public][contentClass=Maven2][ma
inFacet=org.sonatype.nexus.proxy.repository.GroupRepository]
nexus_1     | 2020-01-25 14:00:53,111+0000 INFO  [jetty-main-1]  org.sonatype.scheduling.DefaultSched
uler - Initializing Scheduler...
nexus_1     | 2020-01-25 14:00:53,111+0000 INFO  [jetty-main-1]  org.sonatype.nexus.scheduling.intern
al.DefaultTaskConfigManager - 0 task(s) to load.
nexus_1     | 2020-01-25 14:00:53,112+0000 INFO  [jetty-main-1]  org.sonatype.nexus.NxApplication - N
exus Work Directory : /sonatype-work
nexus_1     | 2020-01-25 14:00:53,137+0000 INFO  [jetty-main-1]  org.sonatype.nexus.NxApplication - S
tarted Nexus Repository Manager OSS 2.14.16-01
nexus_1     | 2020-01-25 14:00:53,687+0000 INFO  [pxpool-1-thread-1] *TASK org.sonatype.nexus.index.t
asks.UpdateIndexTask - Scheduled task (UpdateIndexTask) started :: Updating repository index "Public
 Repositories" from path null and below.
nexus_1     | 2020-01-25 14:00:53,851+0000 INFO  [pxpool-1-thread-1] *TASK org.sonatype.nexus.index.N
exusScanningListener - Scanning of repositoryID="releases" started.
nexus_1     | 2020-01-25 14:00:54,329+0000 INFO  [pxpool-1-thread-1] *TASK org.sonatype.nexus.index.N
exusScanningListener - Scanning of repositoryID="releases" finished: scanned=0, added=0, updated=0,
removed=0, scanningDuration=0:00:00.472
nexus_1     | 2020-01-25 14:00:54,406+0000 INFO  [pxpool-1-thread-1] *TASK org.sonatype.nexus.index.D
efaultIndexerManager - Publishing index for repository releases
nexus_1     | 2020-01-25 14:00:55,058+0000 INFO  [pxpool-1-thread-1] *TASK org.sonatype.nexus.index.N
exusScanningListener - Scanning of repositoryID="snapshots" started.
nexus_1     | 2020-01-25 14:00:55,440+0000 INFO  [jetty-main-1]  com.sonatype.nexus.plugins.nuget.oda
ta.NugetODataGalleryMigrator - Starting Nuget DB migration.
nexus_1     | 2020-01-25 14:00:55,452+0000 INFO  [jetty-main-1]  com.sonatype.nexus.plugins.nuget.oda
```

Now the other thing is that I wanted to login to my services but I couldn't I was getting this error as it shown in the picture below



The solution was by turning off the firewall in my virtual machine and having a fixed ip.
For the firewall it was by the following command:

systemctl disable firewalld

systemctl stop firewalld

-for fixing the ip it's going to be by following this lines:

To enable this configuration, edit the following file:

```
nano /etc/sysconfig/network-scripts/ifcfg-enp0s3
```

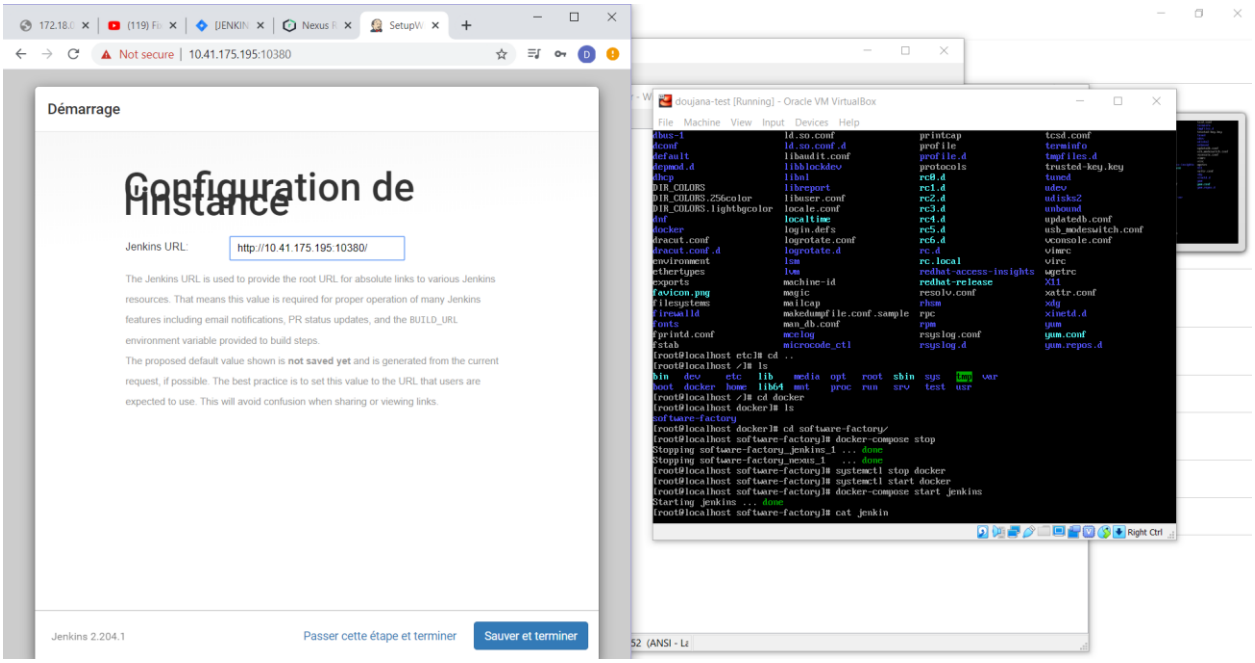
then apply the following settings:

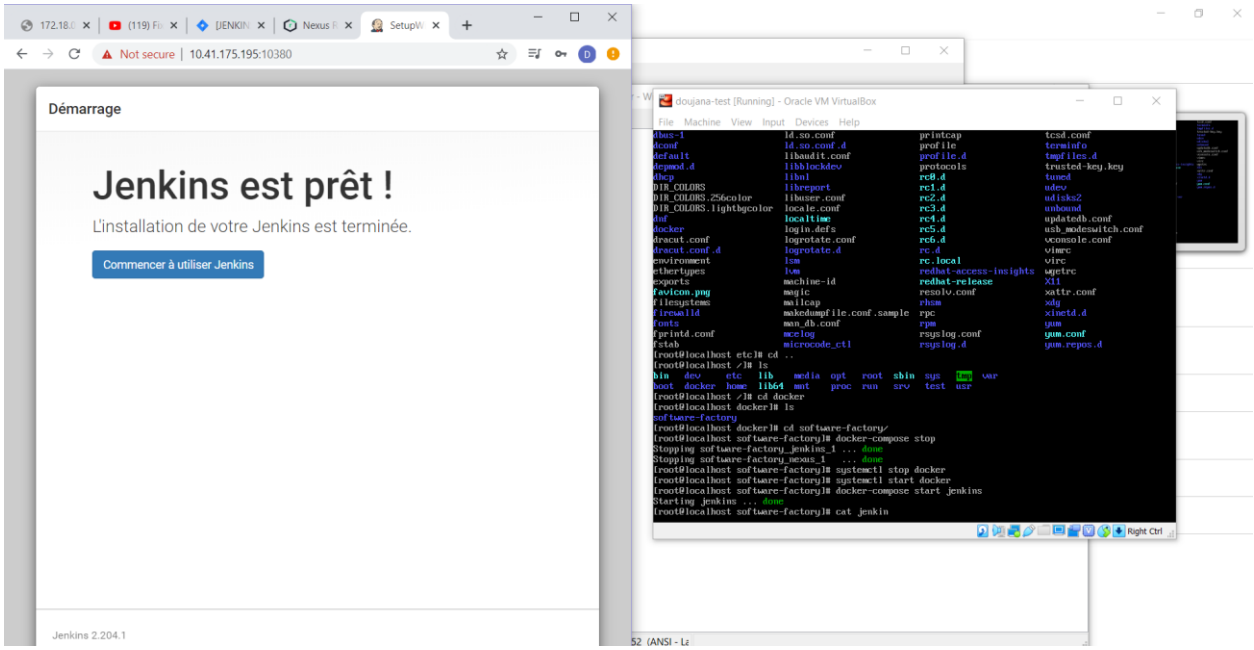
```
TYPE=Ethernet
PROXY_METHOD=none
BROWSER_ONLY=no
BOOTPROTO=none
DEFROUTE=yes
IPV4_FAILURE_FATAL=no
IPV6INIT=yes
IPV6_AUTOCONF=yes
IPV6_DEFROUTE=yes
IPV6_FAILURE_FATAL=no
IPV6_ADDR_GEN_MODE=stable-privacy
NAME=enp0s3
UUID=48e40e15-c66e-45b5-8e94-c092134b0d8f
DEVICE=enp0s3
ONBOOT=yes
IPADDR=192.168.137.10
PREFIX=24
GATEWAY=192.168.137.1
DNS1=8.8.8.8
```

Now let's start by jenkins I opened my browser I wrote ip:port for jenkins more precisely like this:
Well at first I used 10.45.175.195 and then I couldn't log in so after fixing the ip I had the following:

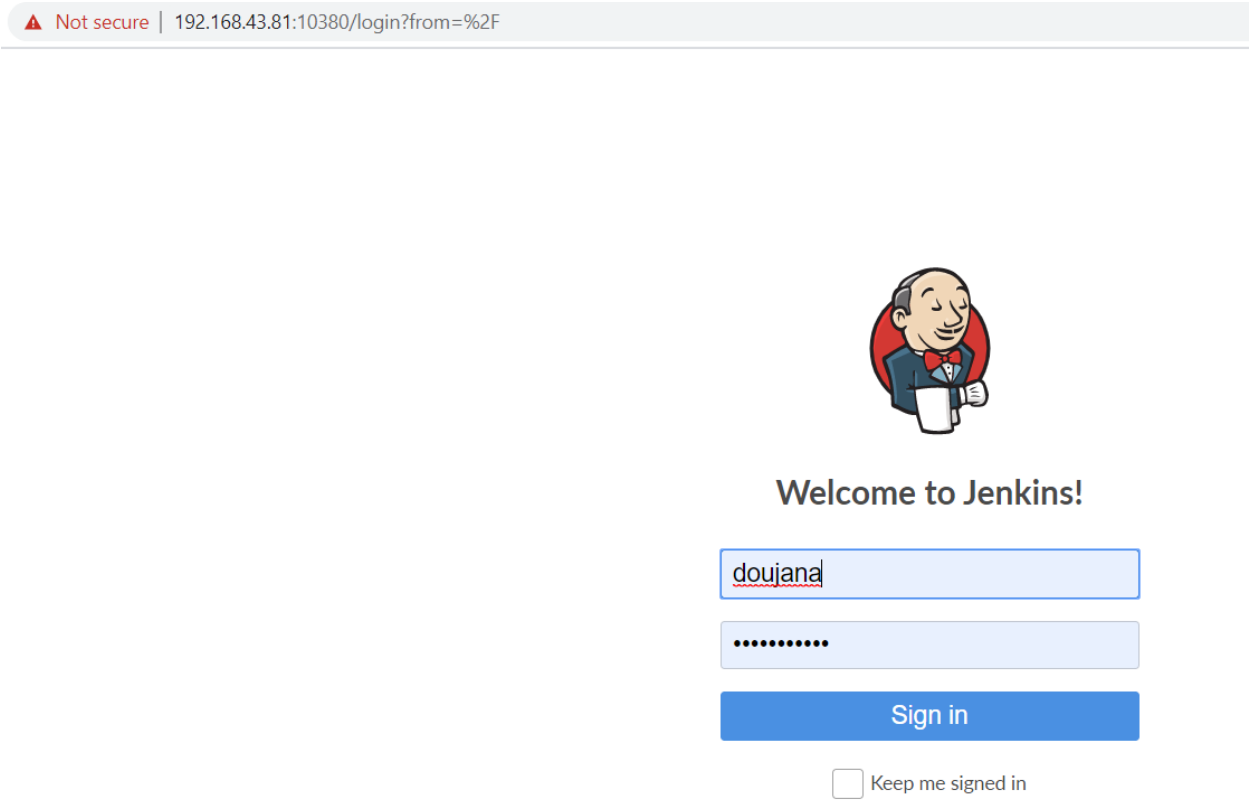
192.168.43.81:10380

-Starting jenkins:





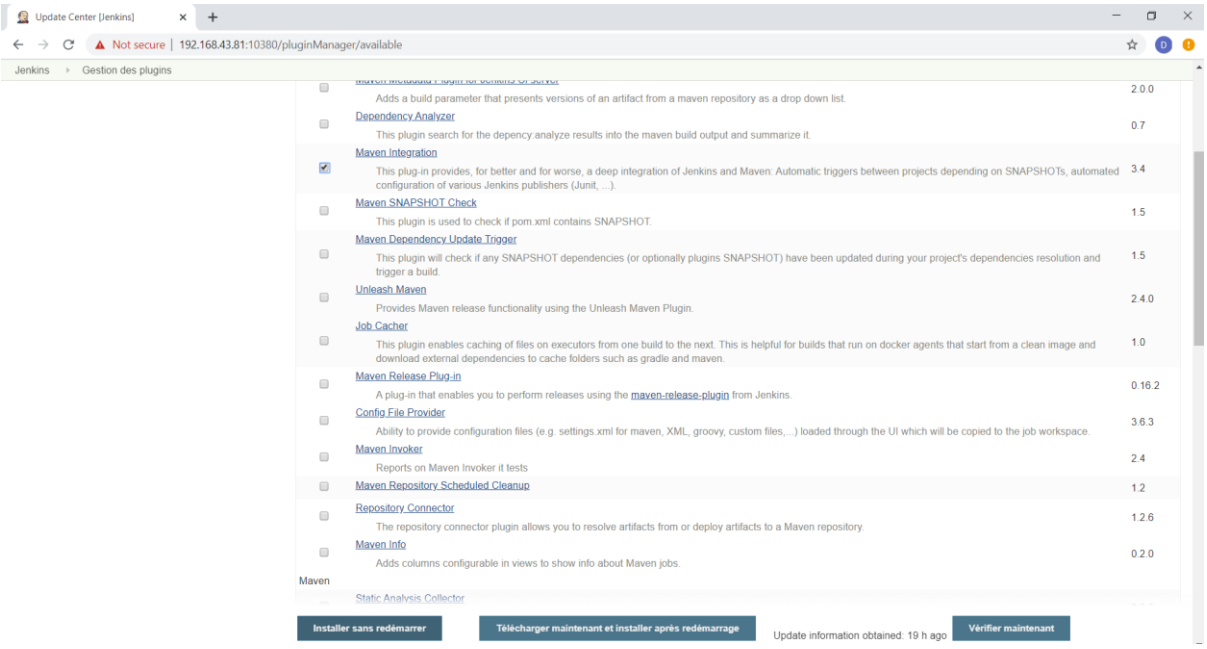
Now, everything is working we will login I set the name as doujana and the password I set it too, now we will deploy our first project in jenkins:

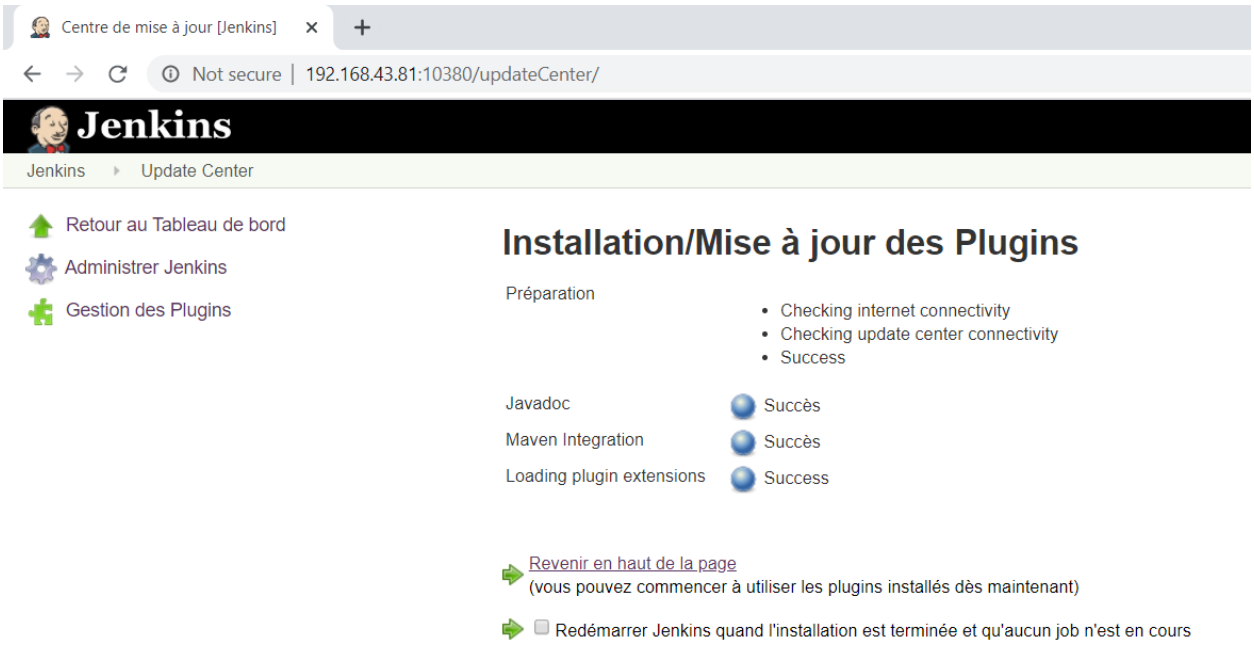


Okay now we have to install maven plugins to deploy our first maven project.so we go to:

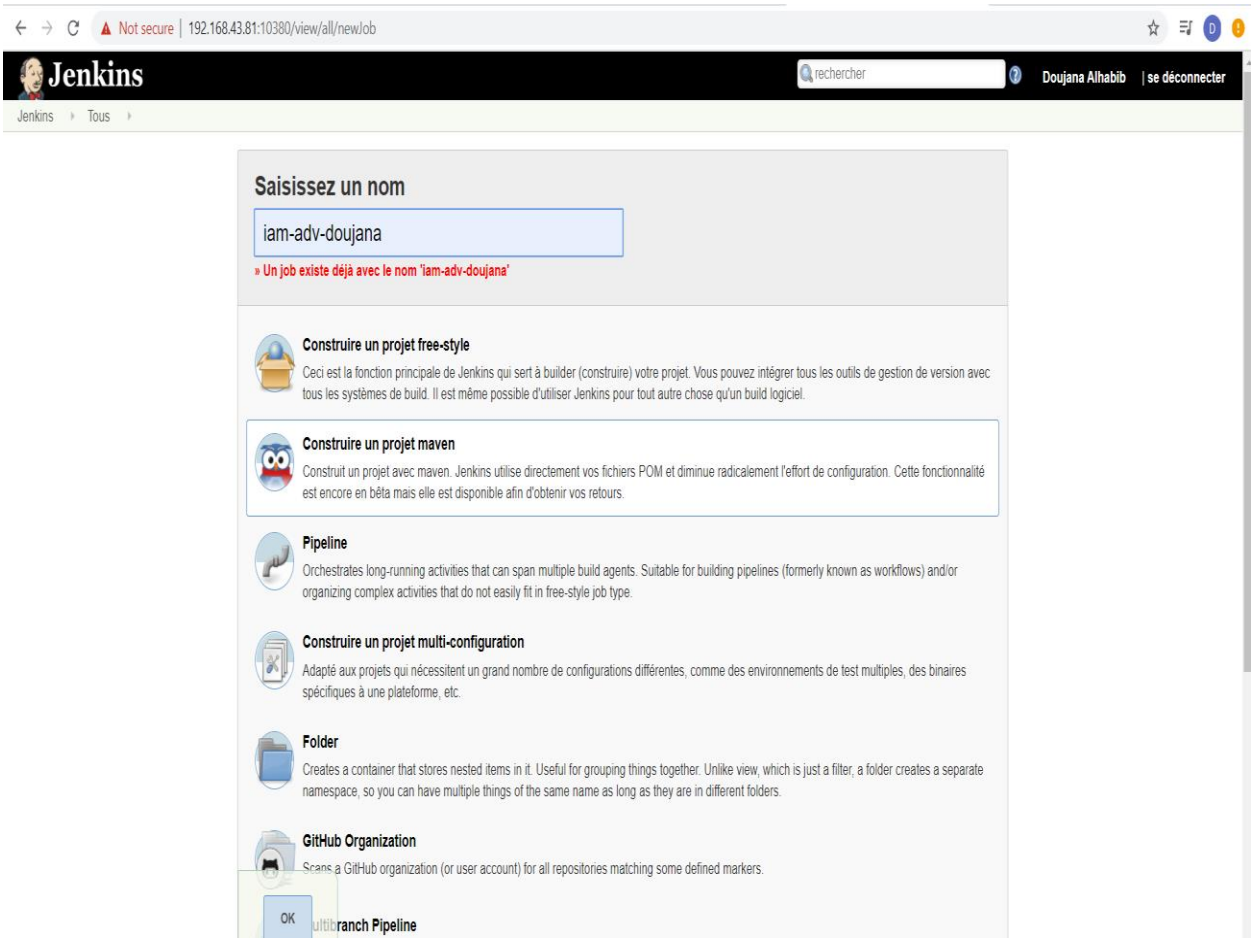
jenkins administration----plugin management----available

we choose maven integration and we install it as will be shown in the pictures below:



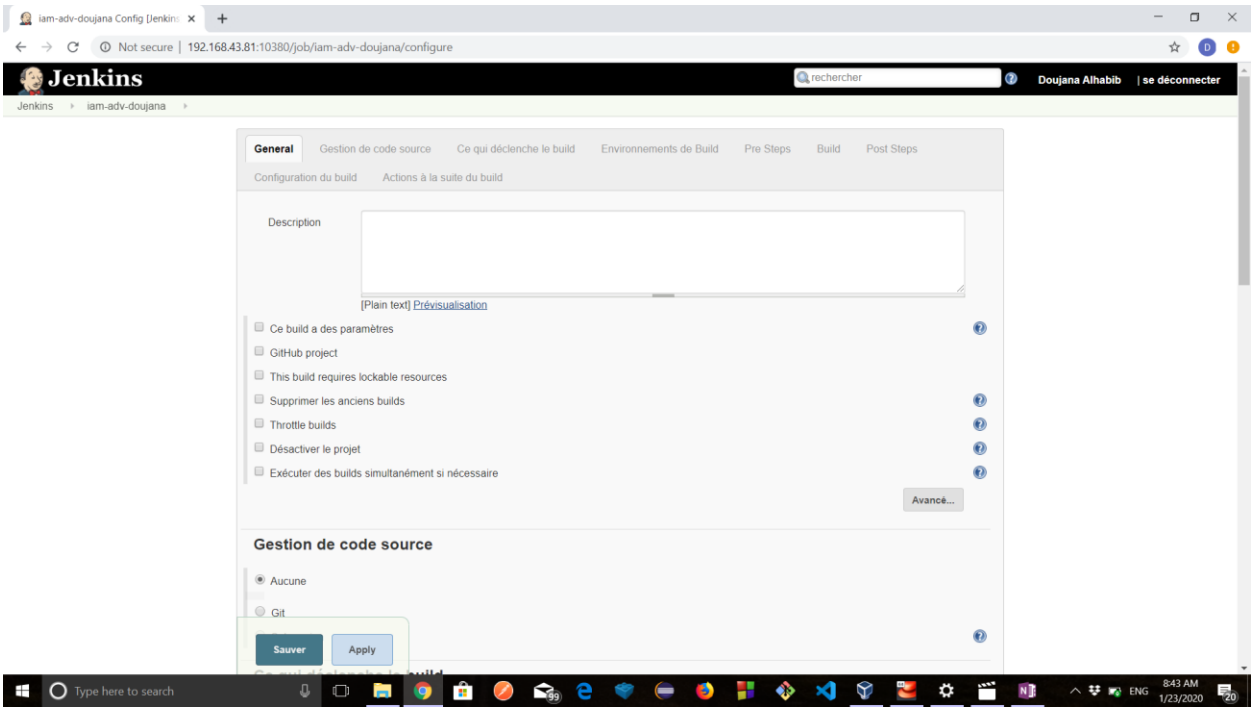


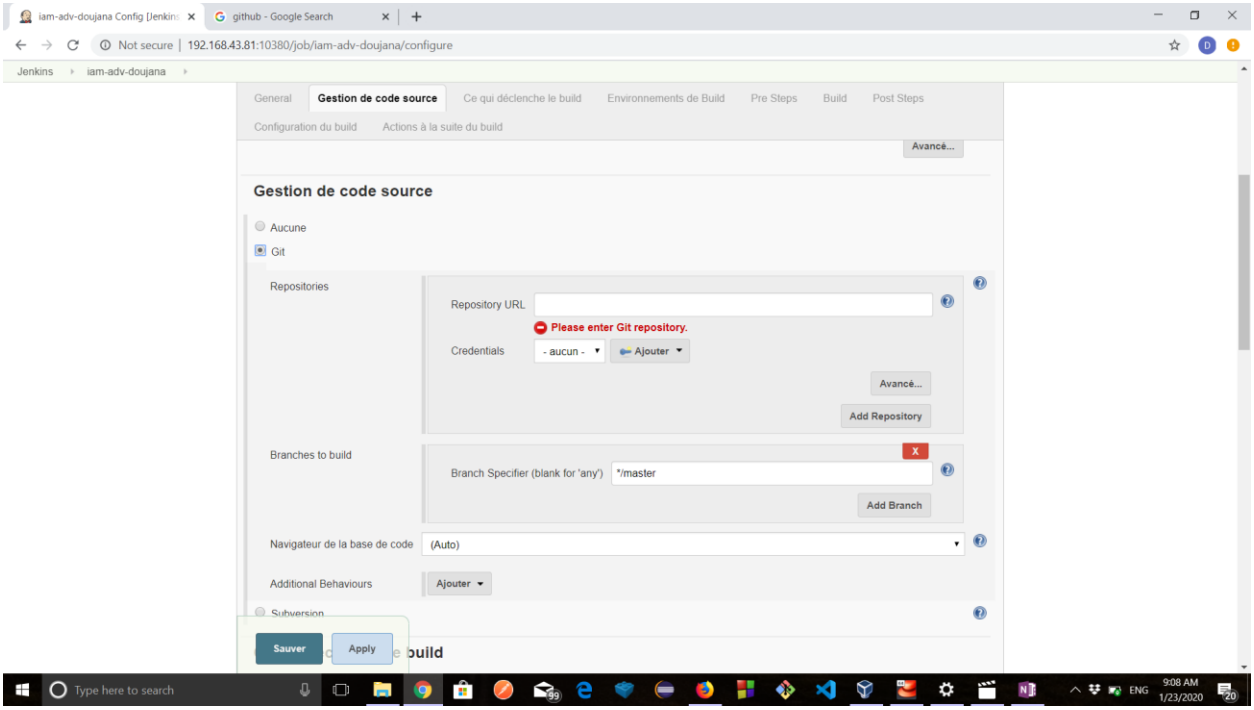
Now, I will go to new item and then we will choose maven project for me I named it iam-adv-doujana



Now let's see what we have to do for the project we need to add the url we can use the github url after choosing that the source will be Git, then I added the goal for each one at first it was install and then it will be deploy.

And then we will put lancer un build we start building the project and that's for now for the configuration for jenkins:





iam-adv-doujana Config [Jenkins] | github - Google Search | GitHub - doujana/Epita-Java-Adv

Not secure | 192.168.43.81:10380/job/iam-adv-doujana/configure

Jenkins › iam-adv-doujana

General | **Gestion de code source** | Ce qui déclenche le build | Environnements de Build | Pre Steps | Build | Post Steps

Configuration du build | Actions à la suite du build

Avancé...

Gestion de code source

- Aucune
- Git

Repositories

Repository URL: https://github.com/doujana/Epita-Java-Advanced.git

Credentials: - aucun - | Ajouter

Avancé...

Add Repository

Branches to build

X

Branch Specifier (blank for 'any'): */master

Add Branch

Navigateur de la base de code: (Auto)

Additional Behaviours: Ajouter

- Subversion

Sauver | Apply

build

Type here to search | ENG | 9:10 AM 1/23/2020

iam-adv-doujana Config [Jenkins] | github - Google Search | GitHub - doujana/Epita-Java-Adv

Not secure | 192.168.43.81:10380/job/iam-adv-doujana/configure

Jenkins › iam-adv-doujana

General | Gestion de code source | Ce qui déclenche le build | **Environnements de Build** | Pre Steps | Build | Post Steps

Configuration du build | Actions à la suite du build

- Use secret text(s) or file(s)
- Abort the build if it's stuck
- Add timestamps to the Console Output
- Inspect build log for published Gradle build scans
- With Ant

Pre Steps

Ajouter une étape pré-build

Build

Version de Maven: Jenkins a besoin de savoir où Maven est installÉ. Please do so from the tool configuration.

POM Racine: pom.xml

Goals et options: clean install

Avancé...

Post Steps

- Run only if build succeeds
- Run only if build succeeds or is unstable
- Run regardless of build result

Should the post-build steps run only for successful builds, etc.

Ajouter une étape post-build

Sauver | Apply

Type here to search | ENG | 9:11 AM 1/23/2020

iam-adv-doujana Config [Jenkins] | github - Google Search | GitHub - doujana/Epita-Java-Adv

Not secure | 192.168.43.81:10380/job/iam-adv-doujana/configure

Jenkins › iam-adv-doujana ›

General | Gestion de code source | Ce qui déclenche le build | **Environnements de Build** | Pre Steps | Build | Post Steps | Configuration du build | Actions à la suite du build

Use secret text(s) or file(s)

Abort the build if it's stuck

Add timestamps to the Console Output

Inspect build log for published Gradle build scans

With Ant

Pre Steps

Ajouter une étape pré-build

Build

Version de Maven — Jenkins a besoin de savoir où Maven est installÉ. Please do so from the tool configuration.

POM Racine: pom.xml

Goals et options: clean install

Avancé...

Post Steps

Run only if build succeeds | Run only if build succeeds or is unstable | Run regardless of build result

Should the post-build steps run only for successful builds, etc.

Ajouter une étape post-build

Sauver | Apply

Configuration du build

Type here to search — ENG — 9:11 AM 1/23/2020

Configuration globale des outils | github - Google Search | GitHub - doujana/Epita-Java-Adv

Not secure | 192.168.43.81:10380/configureTools/

Jenkins › Configuration globale des outils

Installations Ant — Ajouter Ant

Liste des installations Ant sur ce système

Maven

Installations Maven — Ajouter Maven

Maven

Nom: default

Install automatically

Install from Apache

Version: 3.6.3

Supprimer un installateur

Ajouter un installateur

Supprimer Maven

Ajouter Maven

Liste des installations Maven sur ce système

Docker

Installations Docker — Ajouter Docker

Liste des installations Docker sur ce système

Enregistrer | Appliquer

Page générée: 23 janv. 2020 08:12:34 UTC — Jenkins ver. 2.204.1

Type here to search — ENG — 9:14 AM 1/23/2020

iam-adv-doujana Config [Jenkins] | github - Google Search | GitHub - doujana/Epita-Java-Adv

Not secure | 192.168.43.81:10380/job/iam-adv-doujana/configure

Jenkins › iam-adv-doujana ›

General | Gestion de code source | Ce qui déclenche le build | **Environnements de Build** | Pre Steps | Build | Post Steps | Configuration du build | Actions à la suite du build

With Ant

Pre Steps

Invoquer les cibles Maven de haut niveau

Version de Maven: (Par défaut)

Cibles Maven:

Avancé...

Ajouter une étape pré-build

Build

POM Racine: quiz-core/pom.xml

Goals et options: clean install

Avancé...

Post Steps

Run only if build succeeds | Run only if build succeeds or is unstable | Run regardless of build result

Should the post-build steps run only for successful builds, etc.

Sauver | Apply

Type here to search | ENG | 9:23 AM 1/23/2020

iam-adv-doujana #5 Console [Jenkins] | github - Google Search | GitHub - thomasbroussard/2019

Not secure | 192.168.43.81:10380/job/iam-adv-doujana/5/console

Jenkins › iam-adv-doujana › #5

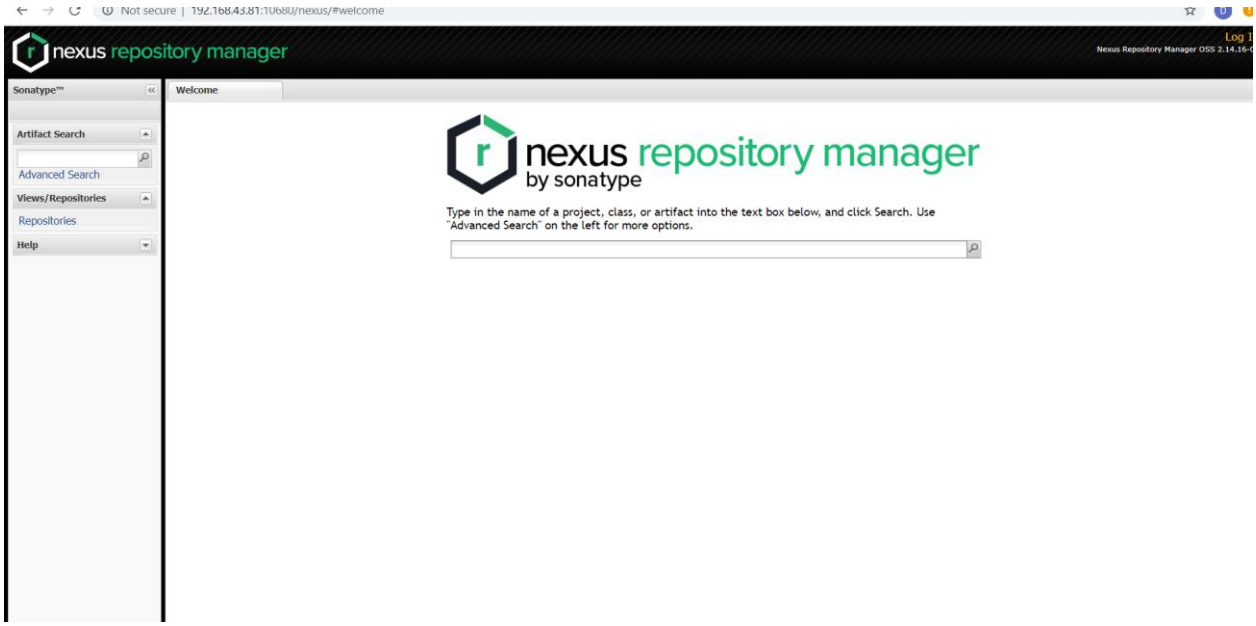
```
Progress (3): 87 kB | 180/230 kB | 12 kB
Progress (3): 87 kB | 184/230 kB | 12 kB
Progress (3): 87 kB | 188/230 kB | 12 kB
Progress (3): 87 kB | 193/230 kB | 12 kB
Progress (3): 87 kB | 197/230 kB | 12 kB
Progress (3): 87 kB | 201/230 kB | 12 kB
Progress (3): 87 kB | 205/230 kB | 12 kB
Progress (3): 87 kB | 209/230 kB | 12 kB
Progress (3): 87 kB | 213/230 kB | 12 kB
Progress (3): 87 kB | 217/230 kB | 12 kB
Progress (3): 87 kB | 221/230 kB | 12 kB
Progress (3): 87 kB | 225/230 kB | 12 kB
Progress (3): 87 kB | 229/230 kB | 12 kB
Progress (3): 87 kB | 230 kB | 12 kB

Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-digest/1.0/plexus-digest-1.0.jar (12 kB at 13 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/codehaus/plexus/plexus-utils/3.0.5/plexus-utils-3.0.5.jar (230 kB at 255 kB/s)
Downloaded from central: https://repo.maven.apache.org/maven2/org/apache/maven/maven-artifact/2.0.6/maven-artifact-2.0.6.jar (87 kB at 96 kB/s)
[INFO] Installing /var/jenkins_home/workspace/iam-adv-doujana/./epita-parent/pom.xml to /var/jenkins_home/.m2/repository/fr/epita/quiz/epita-parent/0.0.1-
SNAPSHOT/epita-parent-0.0.1-SNAPSHOT.pom
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time:  11.585 s
[INFO] Finished at: 2020-01-23T09:19:30Z
[INFO] ------------------------------------------------------------------------
Parsing POMs
Discovered a new module fr.epita.quiz:quiz-core quiz-core
Modules changed, recalculating dependency graph
Established TCP socket on 42655
[quiz-core] $ java -cp /var/jenkins_home/plugins/maven-plugin/WEB-INF/lib/maven35-agent-
1.13.jar:/var/jenkins_home/tools/hudson.tasks.Maven_MavenInstallation/default/boot/plexus-classworlds-
2.6.0.jar:/var/jenkins_home/tools/hudson.tasks.Maven_MavenInstallation/default/conf/logging jenkins.maven3.agent.Maven35Main
/var/jenkins_home/tools/hudson.tasks.Maven_MavenInstallation/default /var/jenkins_home/war/WEB-INF/lib/remoting-3.36.jar /var/jenkins_home/plugins/maven-
plugin/WEB-INF/lib/maven35-interceptor-1.13.jar /var/jenkins_home/plugins/maven-plugin/WEB-INF/lib/maven3-interceptor-commons-1.13.jar 42655
<===[JENKINS REMOTING CAPACITY]===>channel started
```

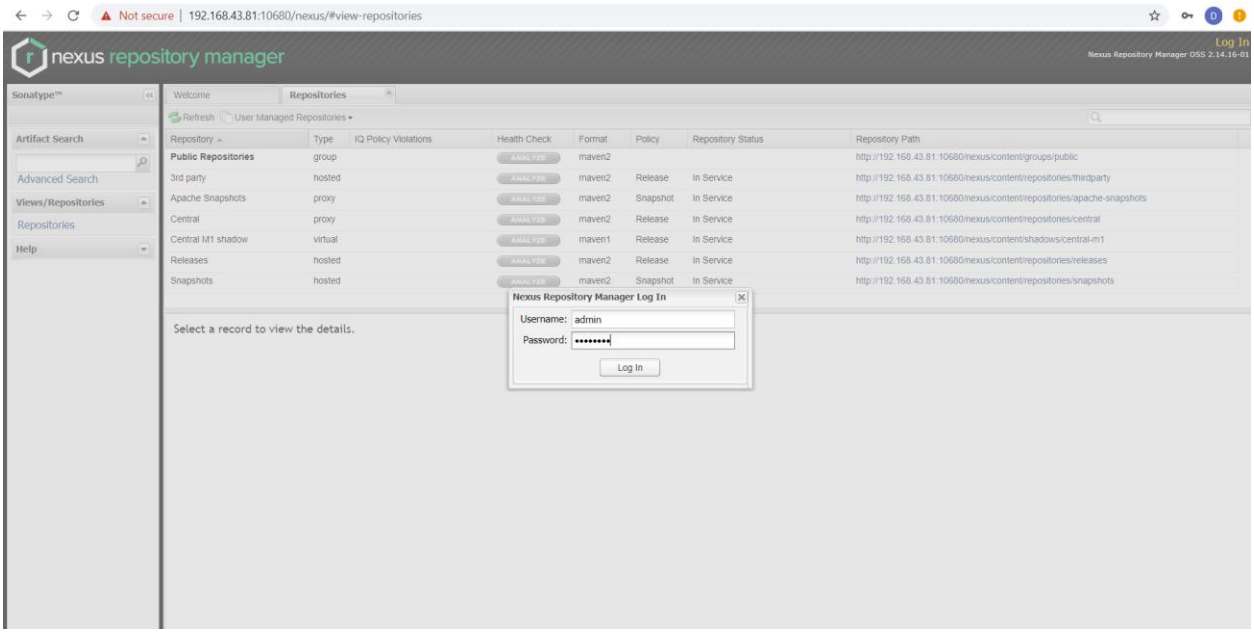
Page générée: 23 janv. 2020 09:19:20 UTC | REST API | Jenkins ver. 2.204.1

Type here to search | ENG | 10:19 AM 1/23/2020

-Nexus configuration:

Now for nexus we will use **192.168.43.81:10680** and when we will log in as an admin, the name for now will be admin password is admin123.Everything will be shown in the below pictures.





-Configuring nexus and jenkins:

Now for jenkins we will go to deploy our project this time and we will add these lines next to deploy

Then we have to go the setting file in the jenkins folder in the software-factory folder and we will add these lines in this location as described in here

Path to maven settings file:

/forge/jenkins/tools/hudson.tasks.Maven_MavenInstallation/default/conf

edit settings.xml :

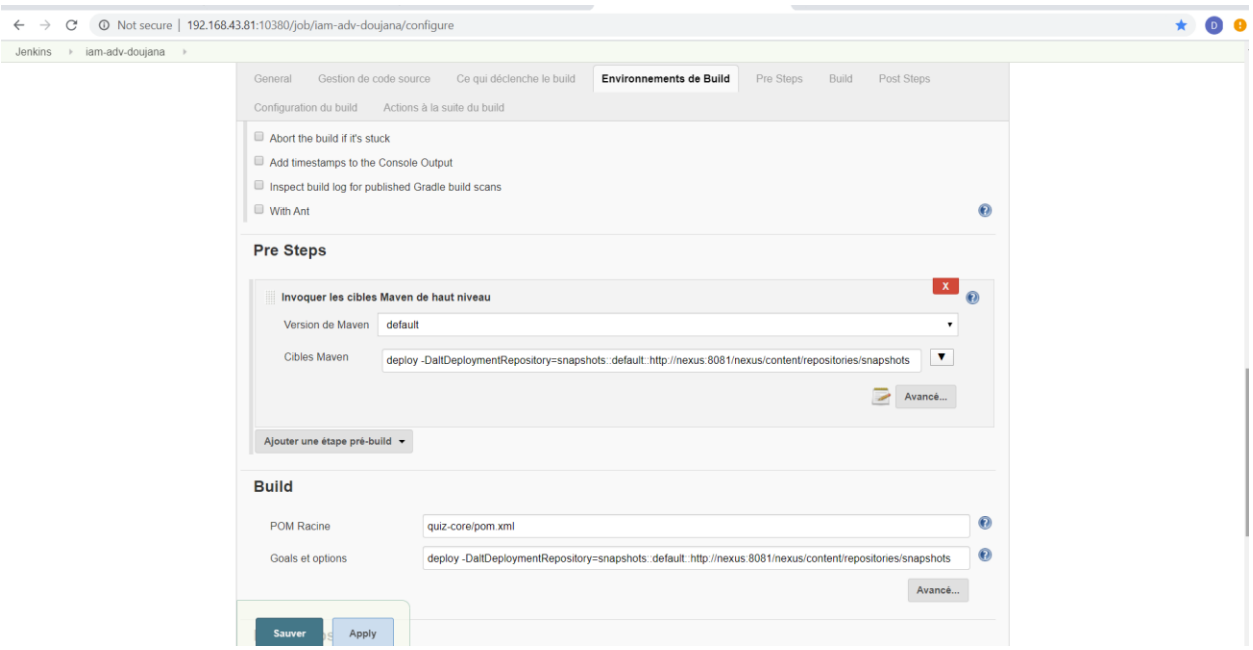
```
  <servers>
    <server>
      <id>releases</id>
      <username>admin</username>
      <password>***</password>
    </server>
    <server>
      <id>snapshots</id>
      <username>admin</username>
      <password>***</password>
    </server>
  </servers>
 <!-- mirrors-->
```

Then in the project pom file (or project parent pom file), add the following lines :

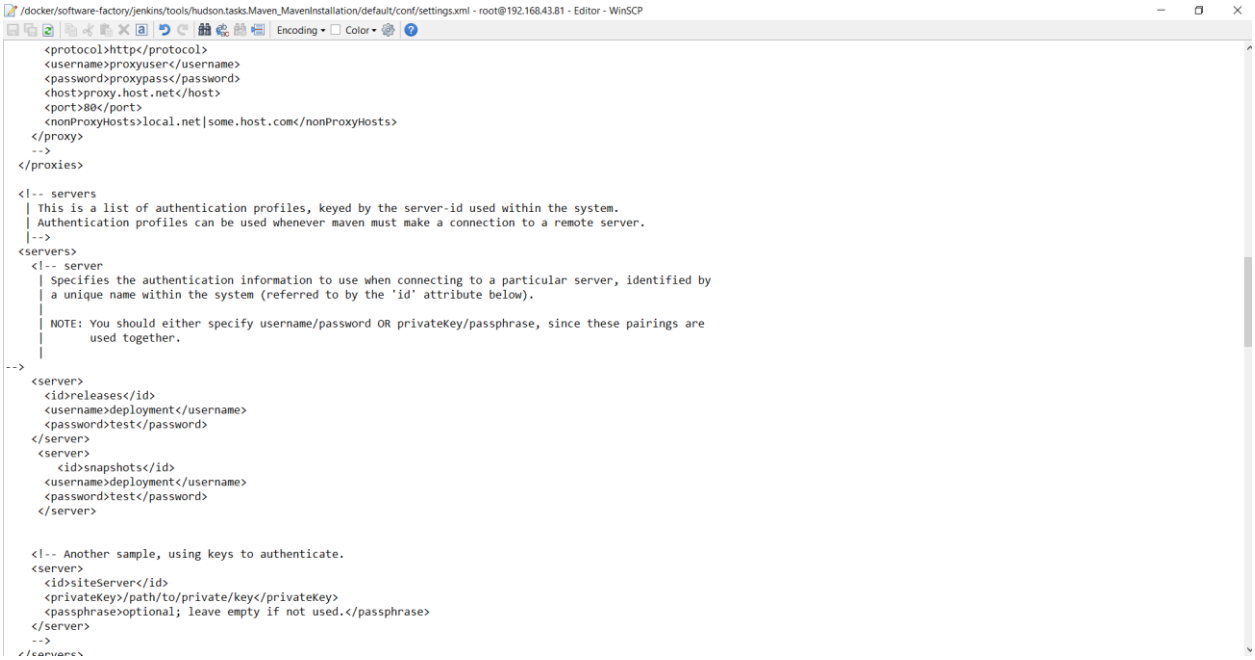
```
<distributionManagement>
               <repository>
                       <id>releases</id>

        <url>http://nexus:8081/nexus/content/repositories/releases</url>
               </repository>
               <snapshotRepository>
                       <id>snapshots</id>

        <url>http://nexus:8081/nexus/content/repositories/snapshots</url>
               </snapshotRepository>
        </distributionManagement>
```

Those lines will tell Jenkins (while invoking the maven goal "deploy"), to deploy to nexus repositories.

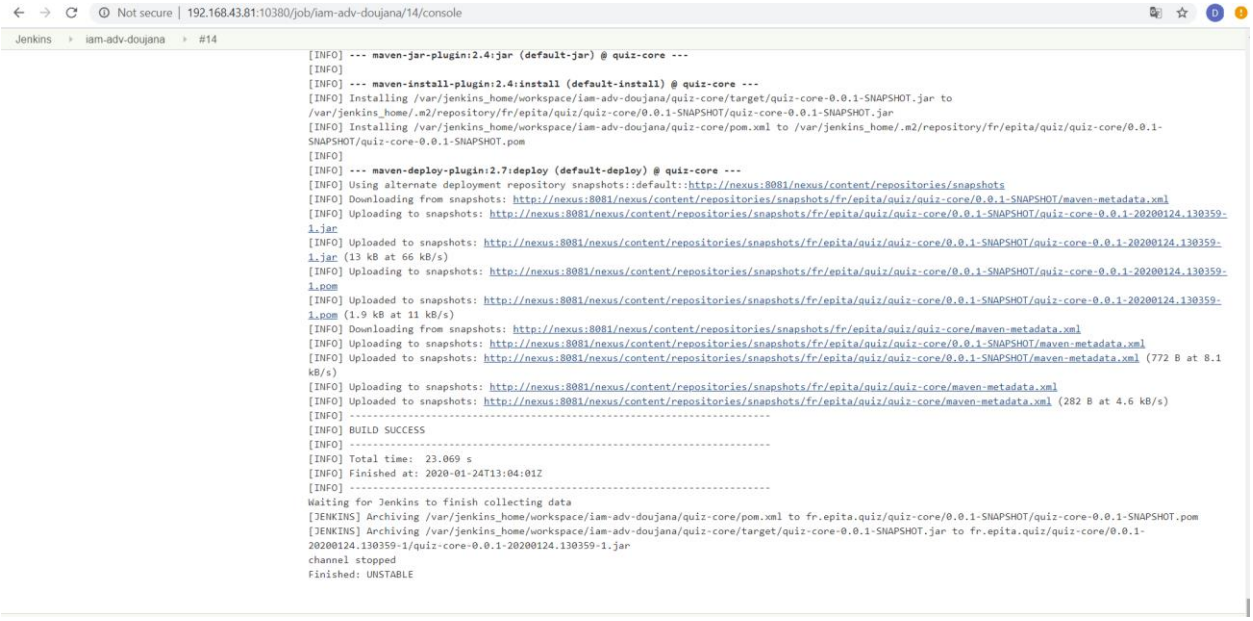
in jenkins we add these lines in the goals



in the setting file it will be like this:

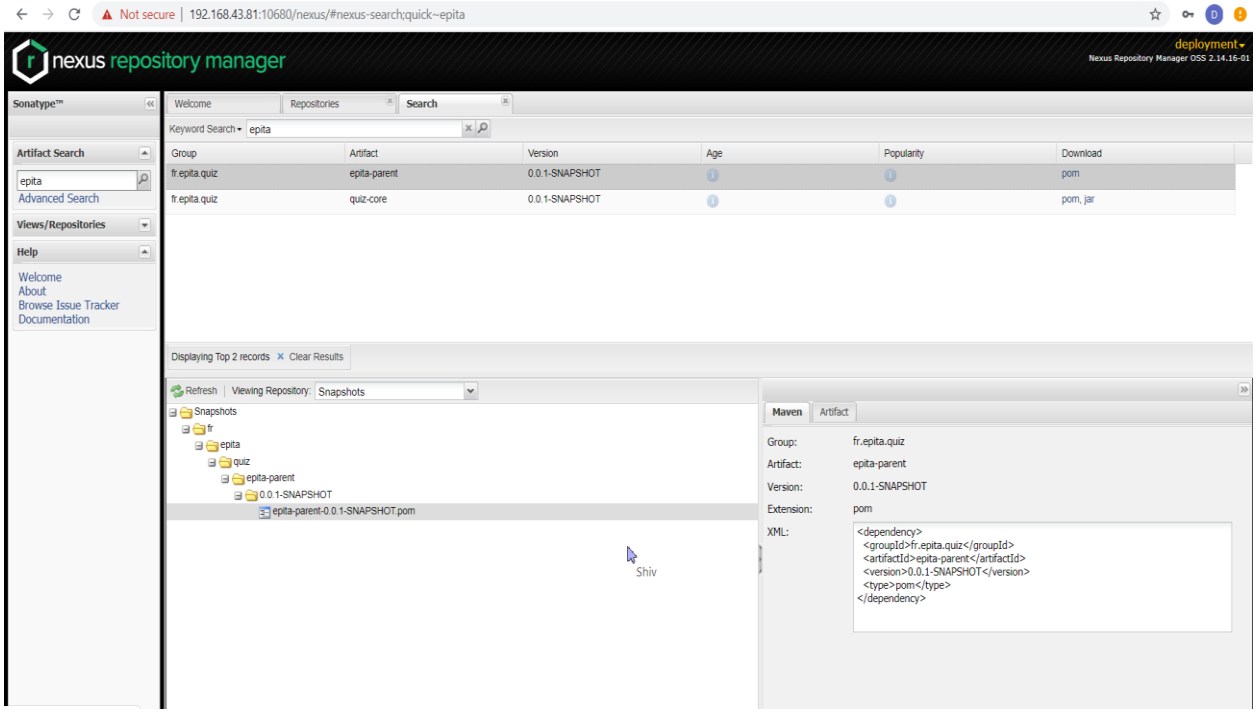


the result I got it is successfully build.



And now in nexus if we search for epita we will see that it is perfectly configured, knowing that in the setting file I changed in the servers the name and the password to deployment: test

And in nexus also I went there I went to security ----users---deployment I reset the password and now I logged in as a deployment I searched for epita and he found it as In the picture below.



Everything will be show in in the attached video.