SRI RAMAKRISHNA ENGINEERING COLLEGE

[Educational Service: SNR Sons Charitable Trust]
[Autonomous Institution, Reaccredited by NAAC with 'A+' Grade]
[Approved by AICTE and Permanently Affiliated to Anna University, Chennai]
[ISO 9001:2015 Certified and all Eligible Programmes Accredited by NBA]
VATTAMALAIPALAYAM, N.G.G.O. COLONY POST, COIMBATORE – 641 022.

DEPARTMENT OF INFORMATION TECHNOLOGY

20IT278 – DEVELOPMENT OPERATIONS

Class: III B.Tech IT A **Semester: V**

Certified that this is the bonafide record of work done by _____in the course **20IT278 - DEVELOPMENT OPERATIONS** of this institution for V semester during the academic year 2023-2024 (ODD).

Faculty In-charge **Head of the Department**

Roll No. | |

Submitted for the Autonomous Practical Examination held on _____

Internal Examiner **Subject Expert**

DEPARTMENT OF INFORMATION TECHNOLOGY

Vision

To evolve the programme to global standards in Information Technology domain attuning to the needs and challenges of industry and society

Mission

- To inculcate professional adeptness and motivation for life-long learning through a comprehensive curriculum, innovative teaching and industry interaction.
- To promote entrepreneurship, research and consultancy.
- To produce technologists with social responsibilities and ethical values.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO I: Be proficient in their professional career, higher education, entrepreneurial endeavours through a sound foundation in mathematical, computing skills and communication technologies in the field of Information Technology.

PEO II: Possess the proficiency to analyze, design and develop products using current tools and techniques for solving complex problems

PEO III: Exhibit effective communication skills, teamwork, professional ethics and multidisciplinary approach in their profession with a capability to relate engineering issues to meet the current trends to a broader societal and environmental context.

PROGRAM OUTCOMES (POs)

1. **Engineering knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/development of solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct investigations of complex problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern tool usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modelling to complex engineering activities with an understanding of the limitations.
6. **The engineer and society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and team work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

11. **Project management and finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-long learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO 1: Use fundamental knowledge in Mathematics, basic Engineering and in-depth knowledge in Information Technology to optimize software development with due consideration for ethical, societal and environmental sustainability.

PSO 2: Design and develop solutions as the member of project team for Information Communication Technology using modern tools, techniques and methodologies.

PSO 3: Apply skill sets to develop and enhance professional career in the fields related to human-computer interaction for the design and implementation of intelligent systems.

Index

Ex. No	Date	List of Exercises	Page No.	Signature
1		Basic Installation of Linux Distribution with Prerequisites of DevOps		
2		Setting up of Virtualization Environment using VM Virtual Box		
3		Implementing SSH Protocol on a Multiple Virtual Machines		
4		Installation and Configuration of Docker Services on Linux		
5		Exploring the Installation and Configuration of Chef and Puppet		
6		Exploring the Installation and Configuration of Jenkins and MVN tools		
7		Exploring the Services of ELK- (Elastic Search, Logstash, Kibana)		
8		Setting up of LAMP Service using Nginx Engine		
9		Setting up of Version Control System Using GIT		
10		Setting up of GitLaboratory as a Repository service		
CONTENT BEYOND SYLLABUS				
11		Study on Kubernetes		

Ex. No. 1	BASIC INSTALLATION OF LINUX DISTRIBUTION WITH PREREQUISITES OF DEVOPS

AIM:

To install Linux operating system and to study the pre-requisites of DevOps.

LINUX DISTRIBUTION:

Linux is open-source, free to use kernel. It is used by programmers, organizations, profit and non-profit companies around the world to create Operating systems to suit their individual requirements. To prevent hacking attempts, many organizations keep their Linux operating systems private. Many others make their variations of Linux available publicly so the whole world can benefit at large.

Some of the famous Linux distributions are:

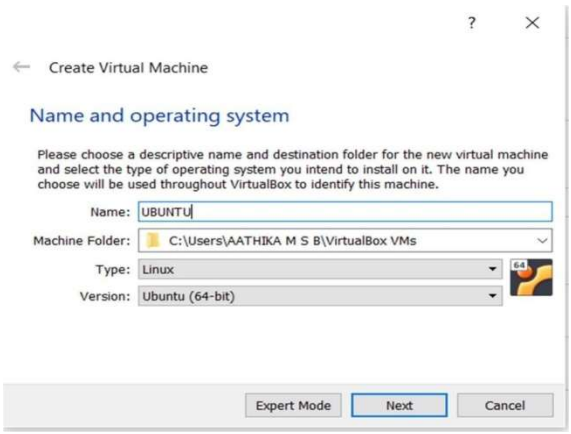
1. Ubuntu
2. Fedora
3. Linuxmint
4. openSUSE
5. PCLinuxOS
6. Debian
7. Mandriva
8. Sabayon/Gentoo

Ubuntu:

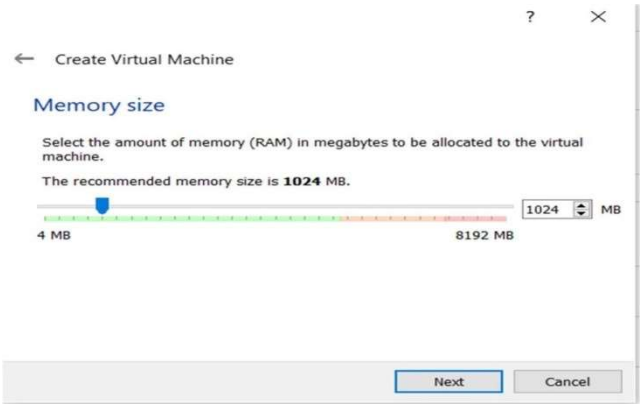
Ubuntu Linux is the most popular open source operating system. There are many reasons to use Ubuntu Linux that make it a worthy Linux distro. Apart from being free and open source, it's highly customizable and has a Software Centre full of apps. There are numerous Linux distributions designed to serve different needs. Being an open source software, Linux allows the developers to pick its code and create something new and exciting.

PROCEDURE:

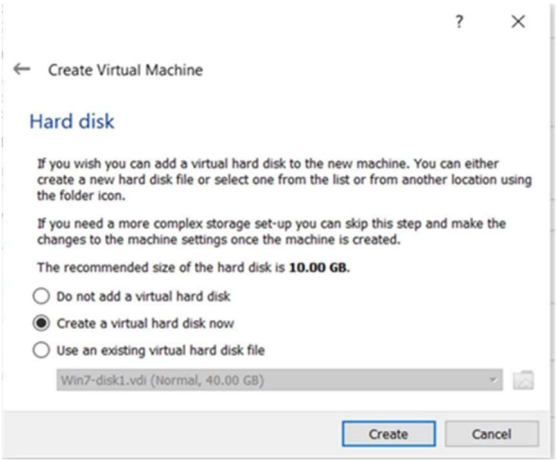
Step 1: Open Virtual box and click on new button. In next window**,** give the name of your OS which you are installing in virtual box. And select OS like Linux and version as Ubuntu 32 bit. And click on next.



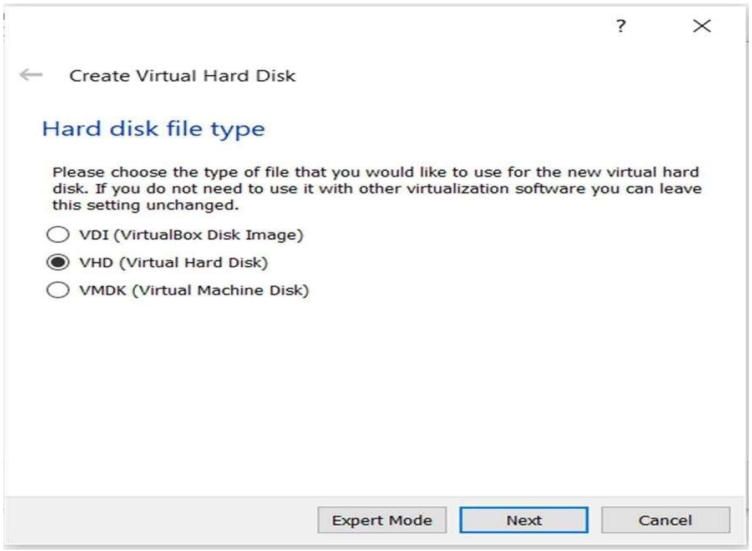
Step 2: Now Allocate Ram Size To your Virtual OS. It is recommended keeping1024mb(1GB) RAM to run Ubuntu better. And click on next.



Step 3: Now to run OS in virtual box we have to create virtual hard disk, click on create a virtual hard drive now and click on create button. The virtual hard disk is where the OS installation files and data/applications you create/install in this Ubuntu machine will reside.

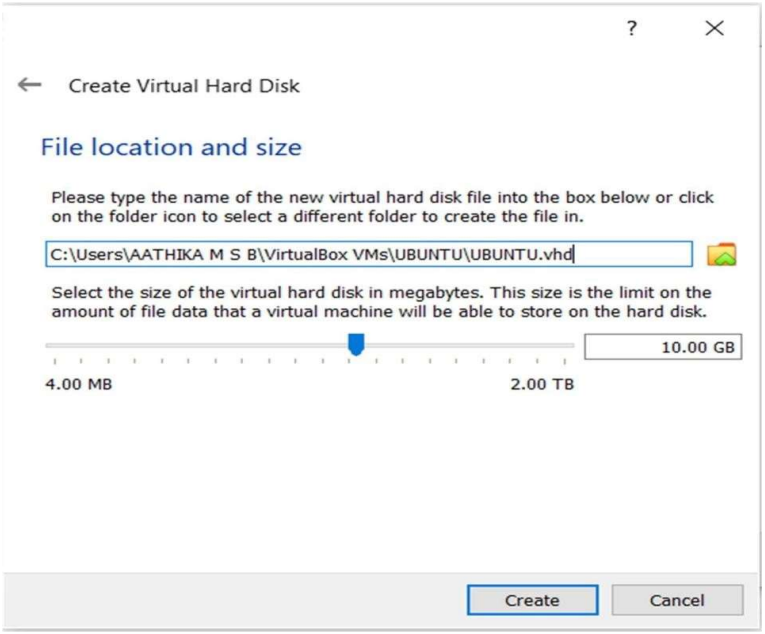


Step 4: Select VHD(virtual hard disk) option and click on next.

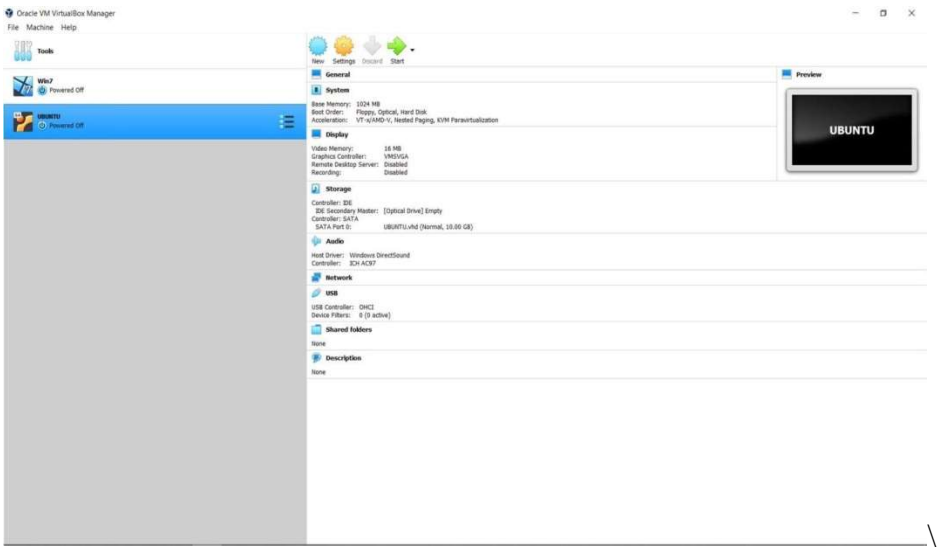


Step 5: Click on dynamical located and click on next. This means that the size of the disk will increased dynamically as per requirement.

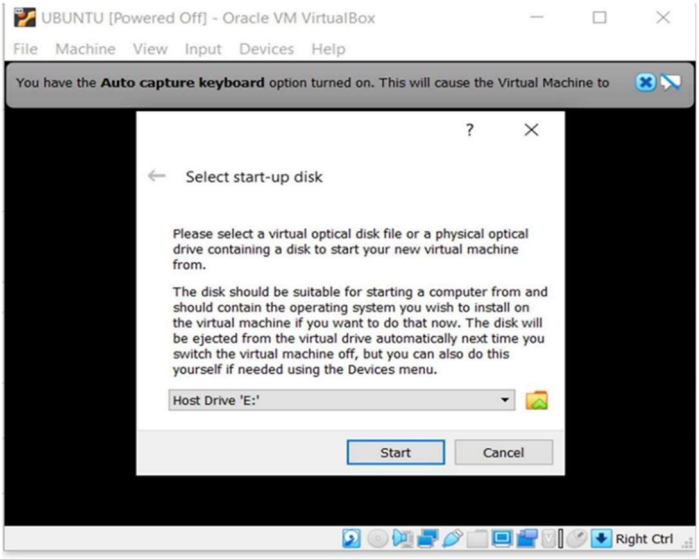
Step 6: Allocate memory to your virtual hard drive. 8GB recommended. Click on create button.



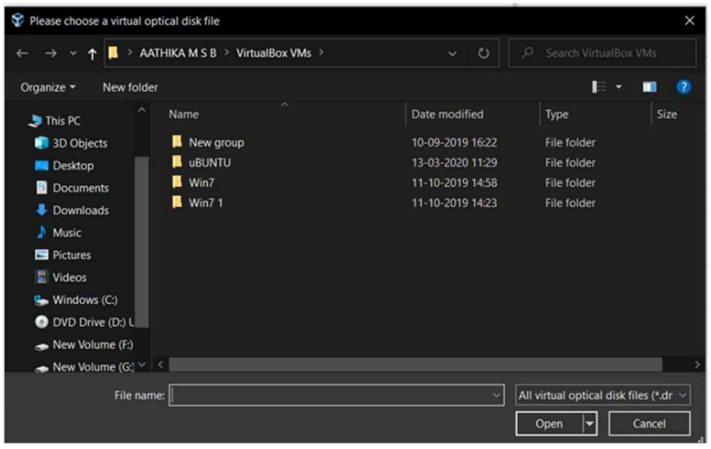
Step 7: Now you can see the machine name in left panel



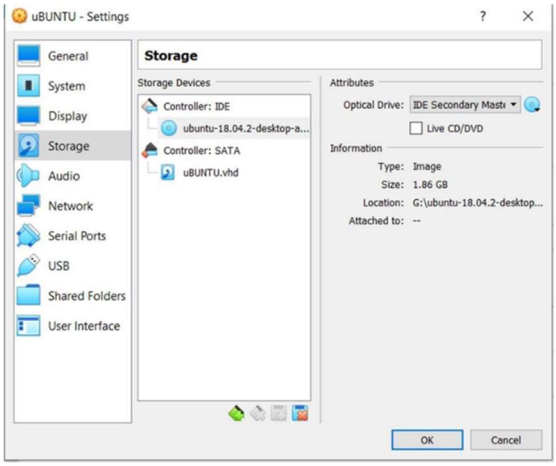
Step 8: Select the Folder Option



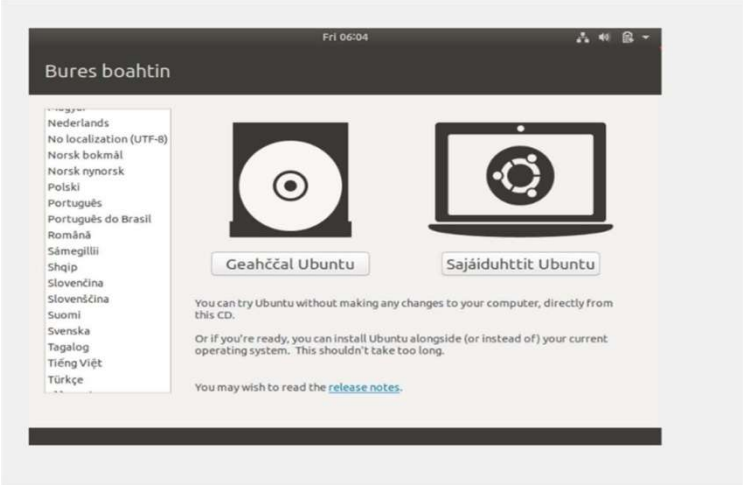
Step 9: Select the Ubuntu iso file.



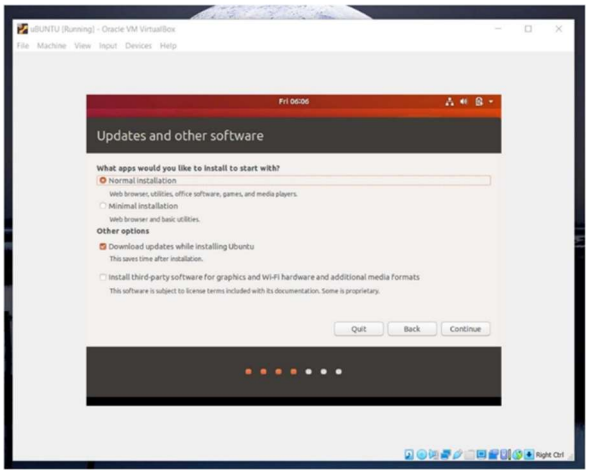
Step 10: In setting you can see the selected file.



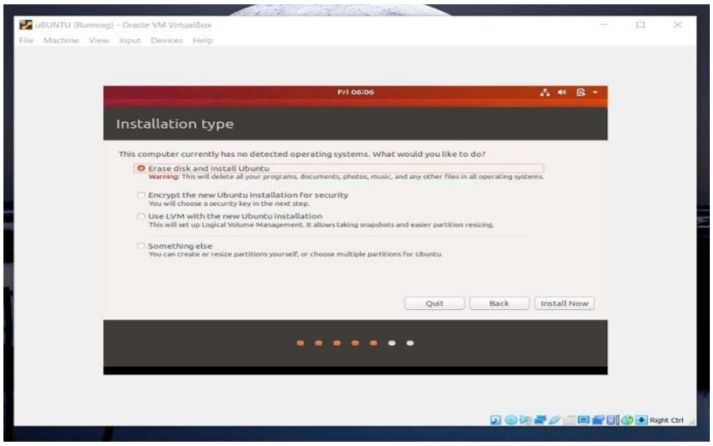
Step 11: You have an option to Run Ubuntu WITHOUT installing.



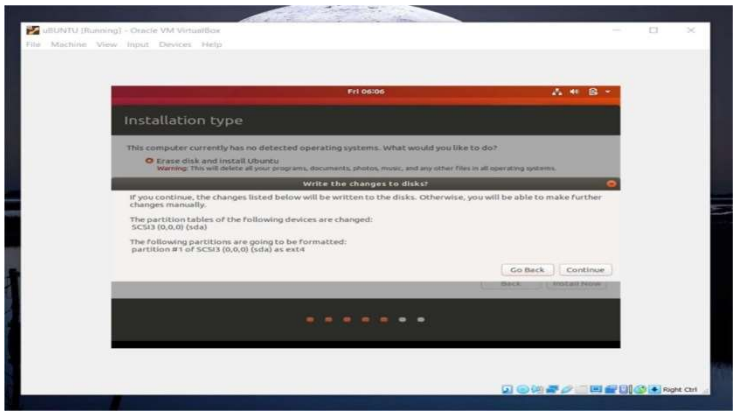
Step 12: Click continue.



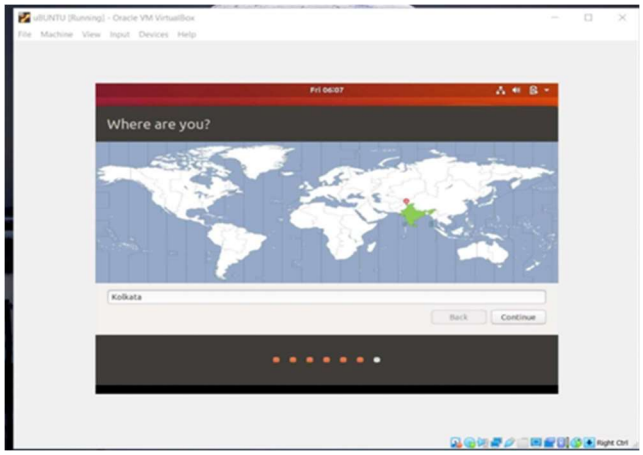
Step 13: Select option to erase the disk and install Ubuntu and click on install now. This option installs Ubuntu into our virtual hard drive which is we made earlier. It will not harm your PC or Windows installation.



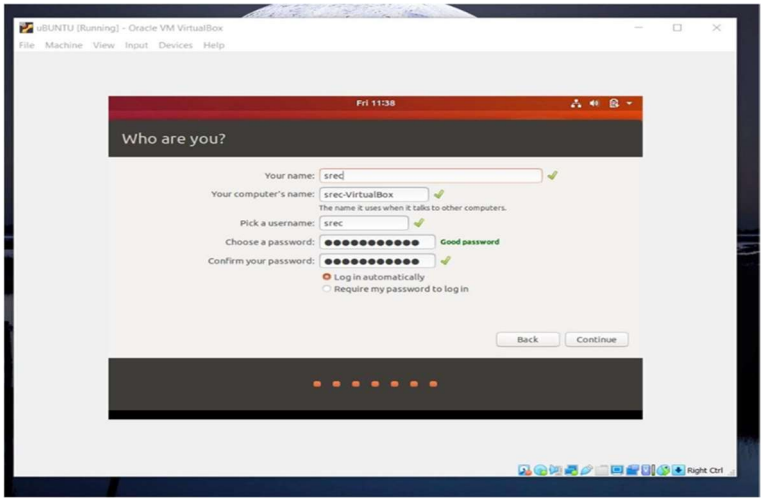
Step14: Click continue.



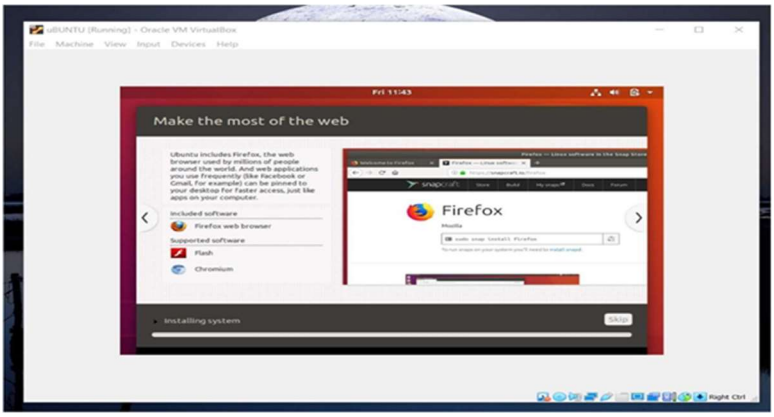
Step 15: Select your location for setting up time zone, and click on continue

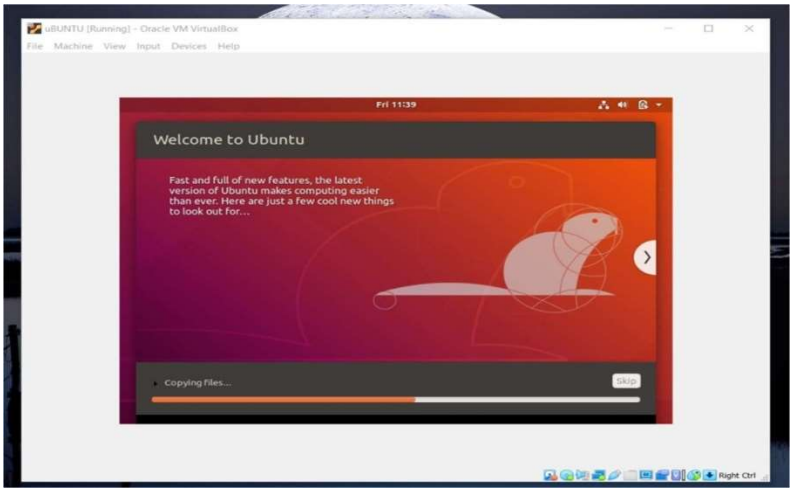


Step 16: Select our username and password for our Ubuntu admin account. This information has been needed for installing any software package into Ubuntu and also for login to your OS. Fill up your details and tick on login automatically to ignore login attempt and click on continue.

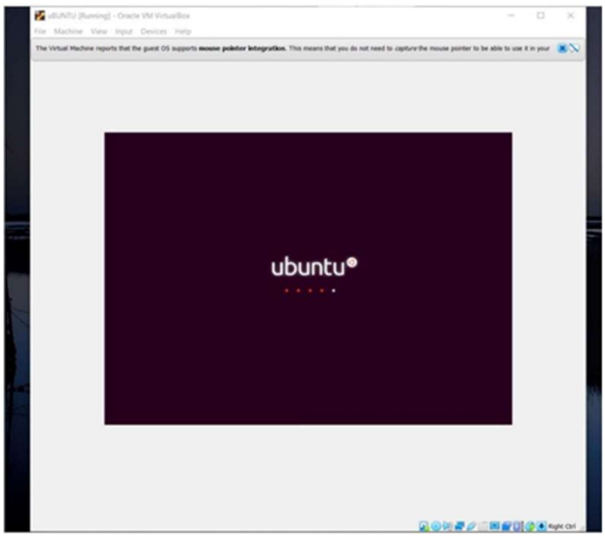


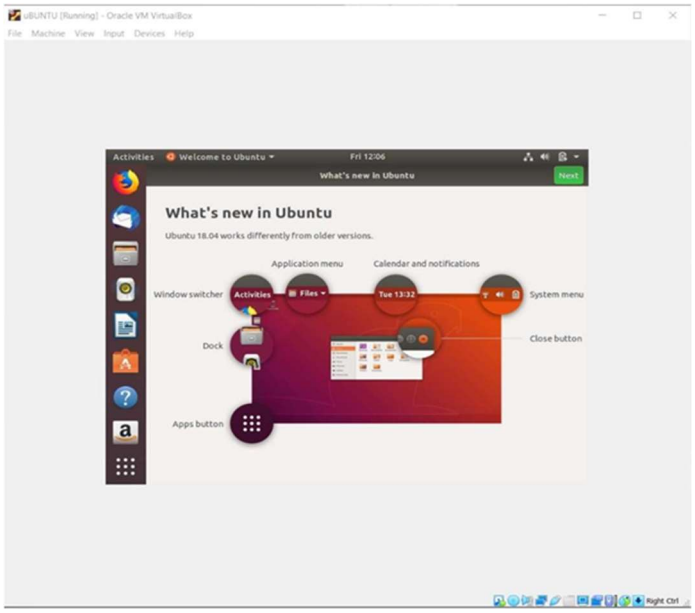
Step 17: Installation process starts. May take up to 30 minutes. Please wait until installation process completes.





Step 18: After finishing the installation, you will see Ubuntu Desktop.


ubuntu


Activities
Welcome to Ubuntu
Fri 12:06
What's new in Ubuntu
Next
What's new in Ubuntu
Ubuntu 18.04 works differently from older versions.
Application menu
Calendar and notifications
Window switcher
Activities
Files
Tue 13:32
System menu
Dock
Close button
Apps button

PRE-REQUISITES OF DEVOPS:

- Organizational commitment
- Automation with discipline
- Tools and infrastructure
- Understanding of Linux/ Unix system concepts
- Basics of core Java
- Familiarity with command-line interface
- Know how of build and deployment process
- Basic knowledge of tools
- Knowledge of Operating system
- Understanding of server setup.
- Understanding of Agile methodology which is currently using to deliver software product.
- Understanding of each entity which plays role to deliver software product.

RESULT:

Thus, the Linux Operating system has been successfully installed in the system and pre-requisites of DevOps has been learned successfully.

Ex. No:2	SETTING UP OF VIRTUALIZATION ENVIRONMENT USING VM VIRTUAL BOX

AIM:

To setup virtualization environment using Virtual Box.

VIRTUALBOX:

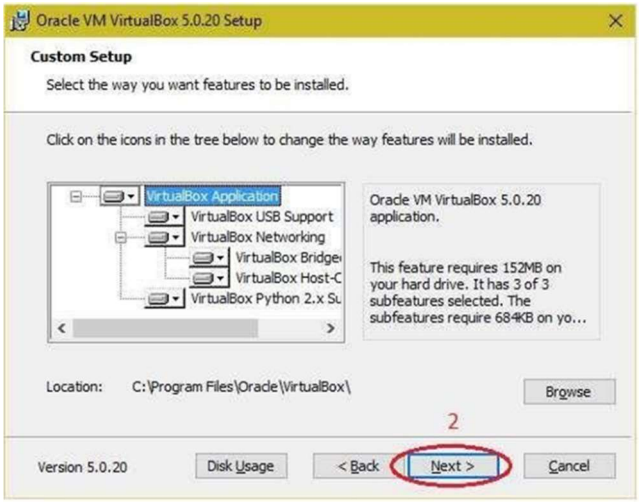
Oracle VM VirtualBox is a cross-platform virtualization application. It installs on existing Intel or AMD-based computers, whether they are running Windows, MacOSX, Linux, or Oracle Solaris operating systems. Secondly, it extends the capabilities of existing computer so that it can run multiple OS, inside multiple virtual machines, at the same time. The only practical limits are diskspace and memory.

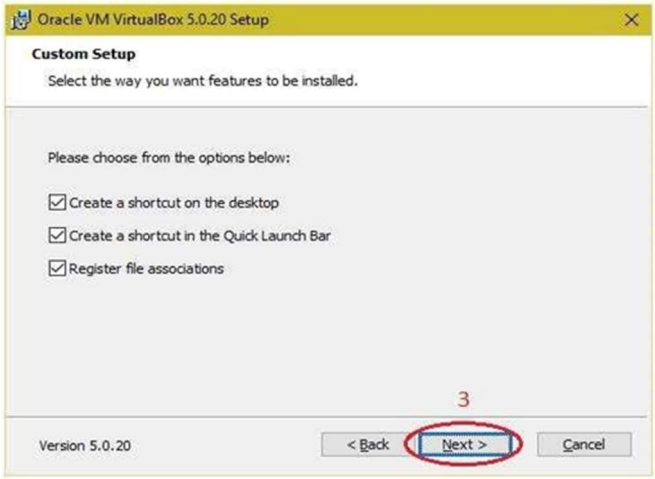
Oracle VM VirtualBox is deceptively simple yet also very powerful. It can run everywhere from small embedded systems order sktop class machine all the way upto data center deployment sand even Cloud environments.

STEPS TO INSTALLVIRTUALBOX:

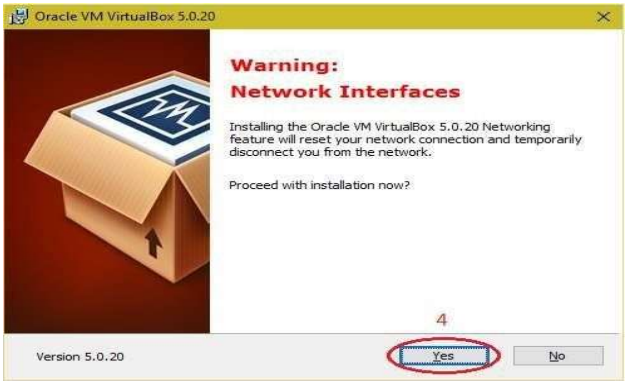
1. Run the virtual box set up and click next

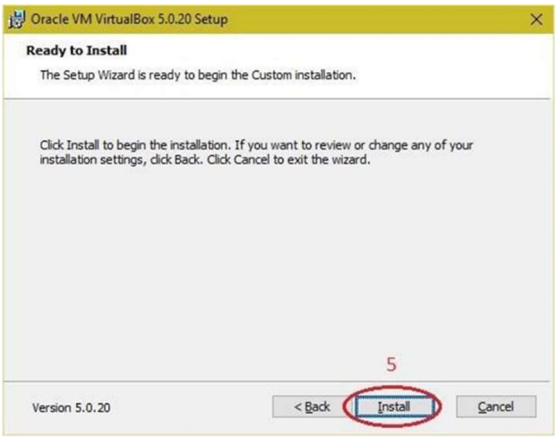
2. Click “Next”





3. Clickon ‘Next’ button
4. Clickon “Yes” button

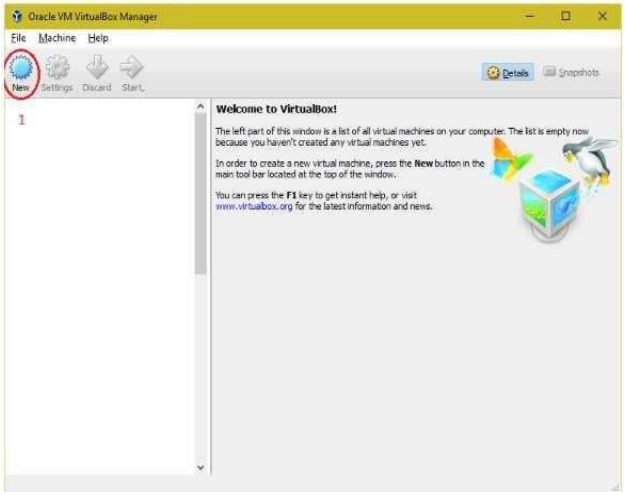




5. Click on Install button

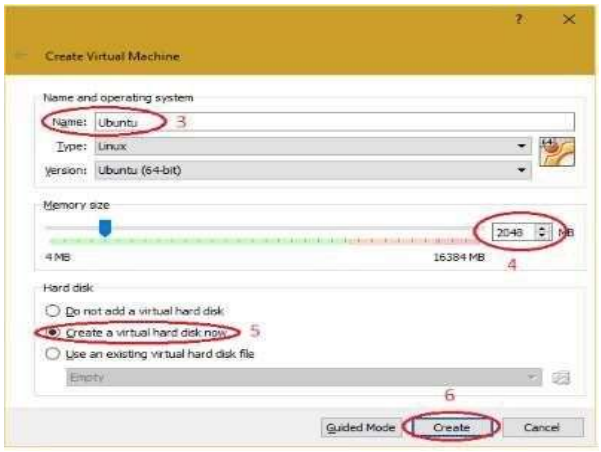
STEPS TO CREATE VIRTUAL MACHINES:

1. Open “Oracle VM Virtual Box Manager”

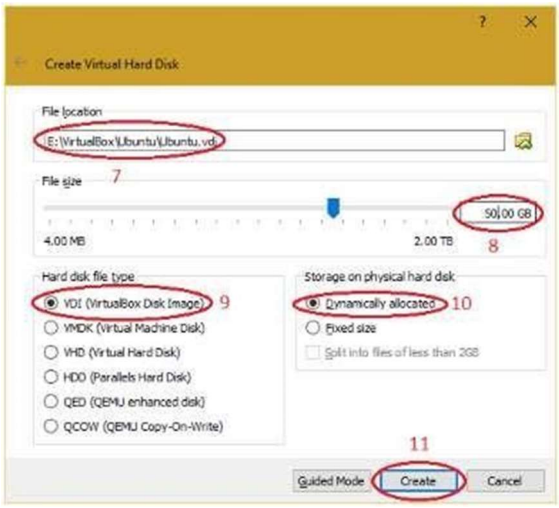


2. Clickon “New“ Button and select “Expert Mode”.

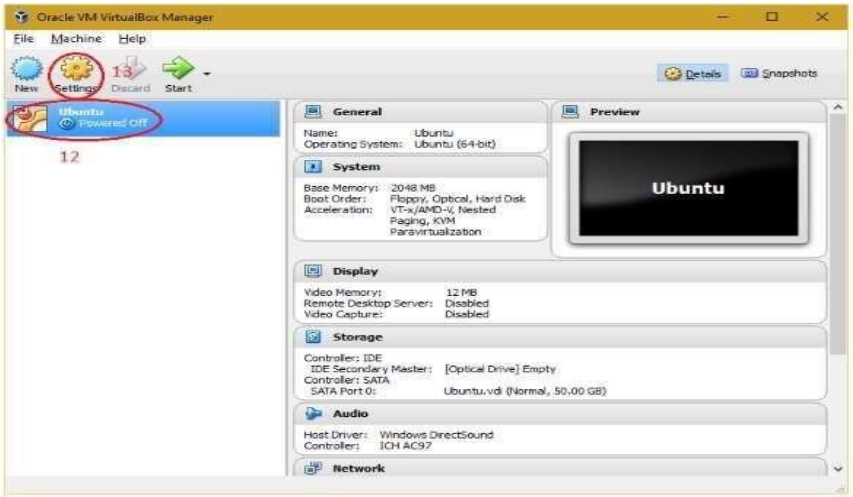
3. Provide the name and operating system information for virtual machine.



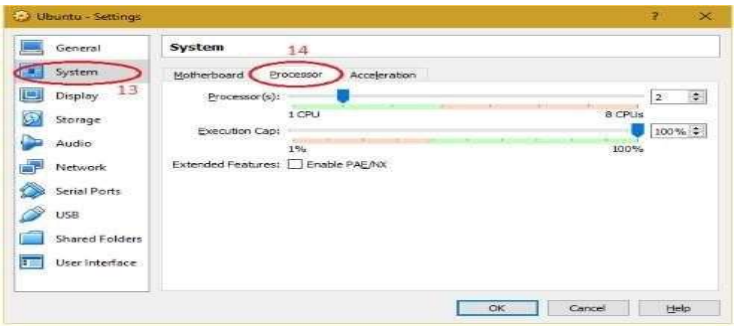
4. Select the path for the virtual hard disk and click on “Create” button.



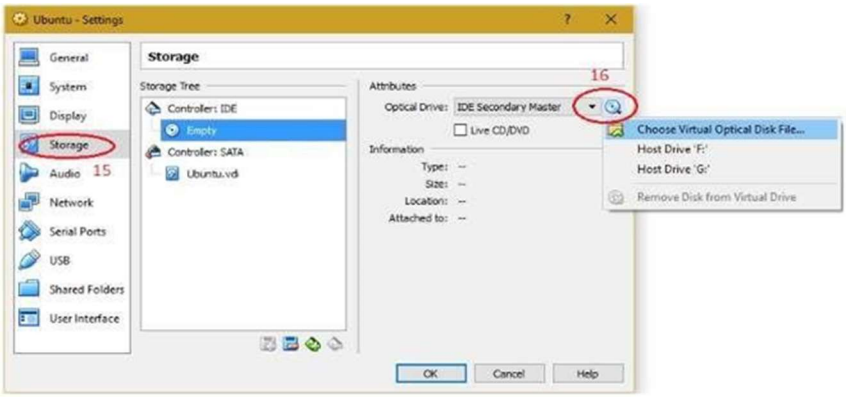
5. Select the virtual machine from the virtual box manager and click on “Settings” button

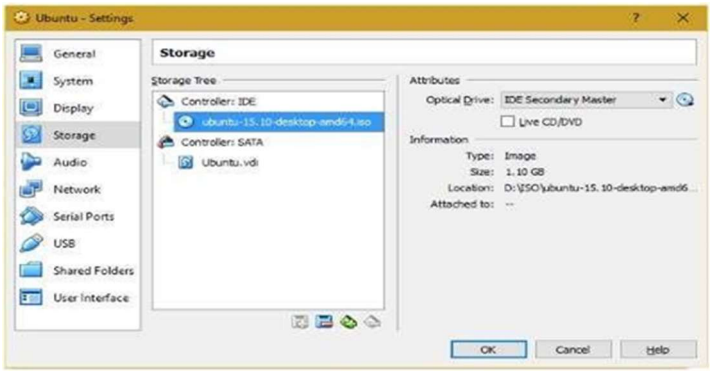


6. Select “System” and navigate to “Processor” tab to adjust number of processor of virtual machine for better performance.

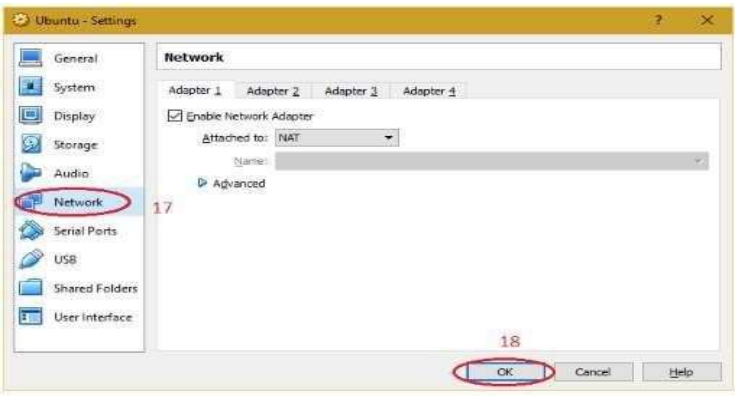


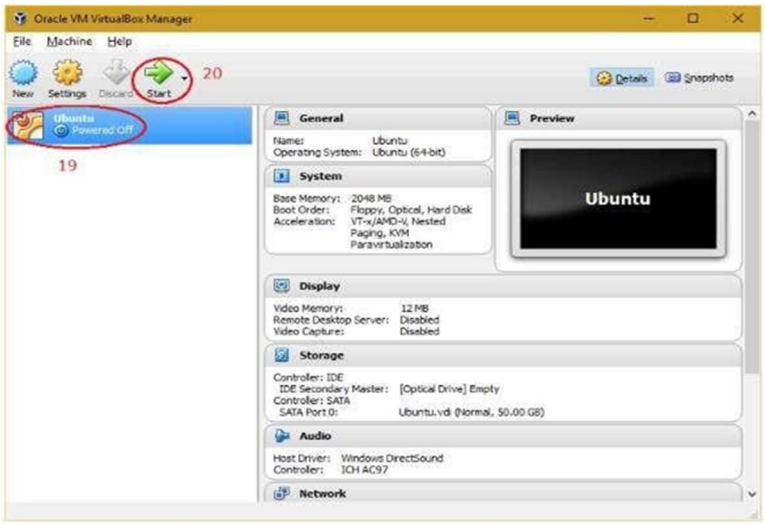
7. Select “Storage” and choose the installation media of operating system (ISO/CD/DVD). Preferred Linux”. iso” can be downloaded from CCftp site. Also many different flavours of Linux are available Fedora, CentOS, Ubuntu, Debian, Mageia, openSUSE, Arch Linux,Slackware Linux etc.



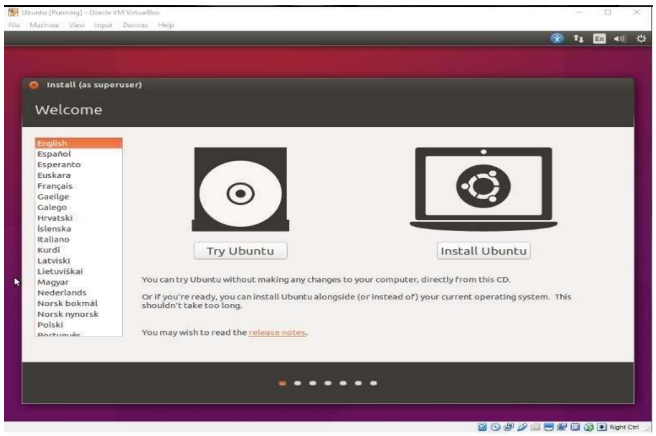


8. Select “Network” to make changes required for network setting of virtual machine and click on “OK”.





9. Select the created virtual Machine and click on “Start” button.
10. Proceed with the installation of operating system in virtual machine.



RESULT:

Thus, the virtualization environment setup was created successfully.

Ex. No:3	IMPLEMENTING SSH PROTOCOL ON A MULTIPLE VIRTUAL MACHINES

AIM:

To create a secure shell(SSH) tunnel to access the services and resources provided by your Oracle Java Cloud Service instance's virtual machine(VM).

PROCEDURE:

Background:

- Oracle Cloud services such as Oracle Java Cloud Service and Oracle Database Cloud Service are built on top of infrastructure and functionality that are provided by Oracle Compute Cloud Service.
- When we create an instance of one of these services, all the Oracle Compute VMs required to support the service are provisioned and configured for you.
- We can access the services and resources provided by the VMs by logging into the machine through a secure shell (SSH).
- If the port for are source provided by a VM is not directly accessible through the internet, we can access that resource by creating an SSH "tunnel" to a port in the VM hosting the service instance.
- For example, a tunnel is required for connecting a local Integrated Development Environment (IDE) such as Eclipse to your Oracle Java Cloud Service instance.

Pre-requisities:

- We need an SSH client to create an SSH tunnel.
- This describes two ways to create a tunnel, one using a WindowsGUIclient, called PuTTY, and one using the ssh utility at the command line.
- We'll need one or both of the following

PuTTY

- PuTTY is a free, open-source implementation of several network protocols, including SSH.
- PuTTY is available from many sites, but you can reach the main download site http://www.putty.org.

An SSH client

- ❖ It includes support for the SSH protocol and a command line shell. Many implementations of UNIX and UNIX-like operating systems already include ssh;
- ❖ OpenSSH is available from http://www.openssh.com/portable.html.
- ❖ An SSH private key file that matches the SSH public key used when the Oracle Java cloud Service instance was created.
- ❖ For PuTTY, we need a private key in the proprietary PuTTY(.ppk) format.
- ❖ For ssh at the command line, we need an SSH format such as OpenSSH.

Finding the IP Address of the VM:

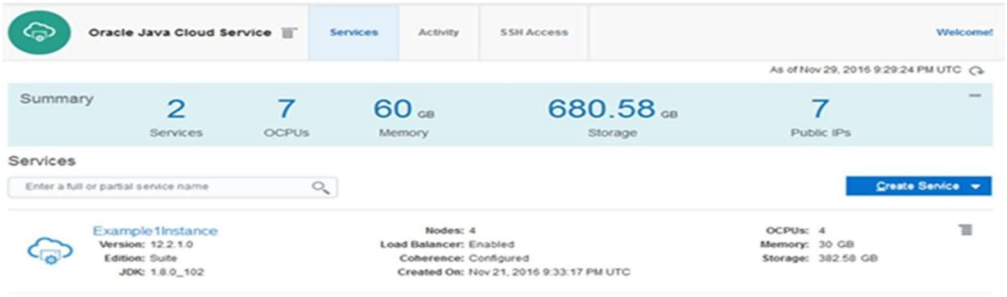
Whether you use PuTTY or a command line shell, you must find the IP address of the VM hosting your Oracle Java Cloud Service instance before you can create an SSH tunnel.

1. Sign in to the My Services application by clicking the link in your Welcome email. Or, you can go to http://cloud.oracle.com, click Sign In, and select the Public CloudServices value for your region from the My Services - Select Data Center dropdown list. Either way you must provide an identity domain, user name, and password to sign in.

When logged in, you will see the MyServices Dashboard.

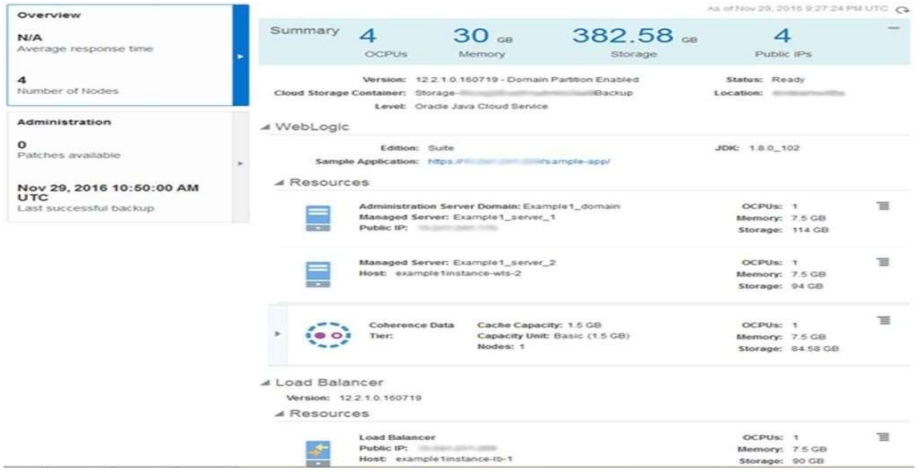
2. From the navigation menu at the top of the page, choose **Java**.

The Oracle Java Cloud Service Console is displayed. An example is shown in the following illustration.

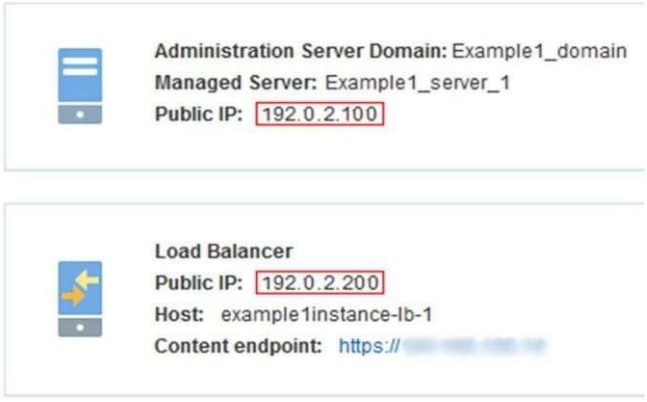


3. Find the service instance you want to manage and click the name.

The service instance Overview page is displayed, showing details about the instance. An example is shown in the following illustration.

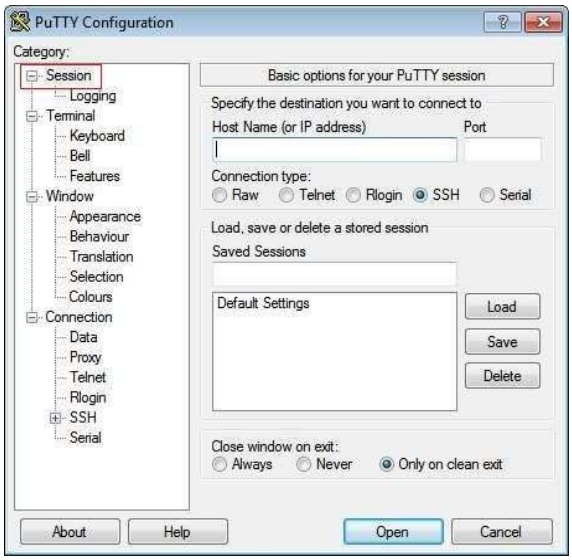


4. Make a note of the IP address of the Administration Server and, if you have one configured, the load balancer.



USING PUTTY TO CREATE AN SSH TUNNEL

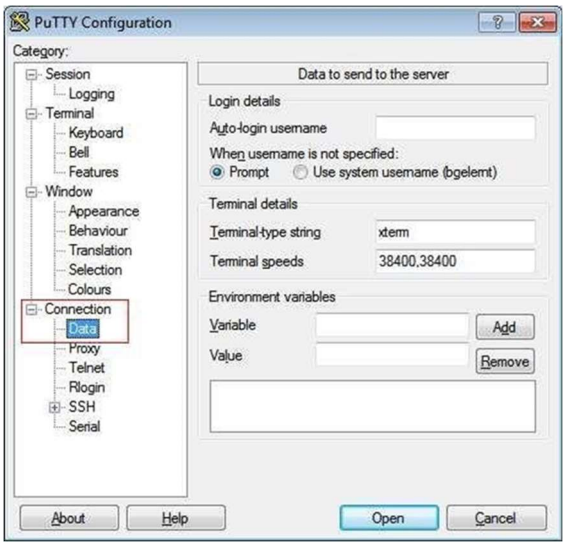
1. Find putty.exe in the PuTTY folder on your computer, for example, C:\Program Files(x86)\PuTTY. Double-click putty.exe. to open it.The PuTTY Configuration window is displayed, showing the Session panel.



2. In the **HostName (orIPaddress)** box, enter the IPaddress of the VM. Leave the port number at the default 22.

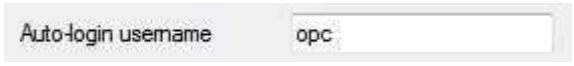
3. Confirm that the Connection type option is set to SSH.

4. In the Category tree, expand Connection if necessary and then click **Data**.

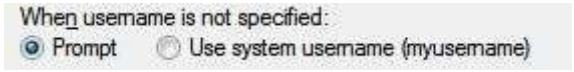


The Data panel is displayed.

5. In Auto-log in username box, enter opc.

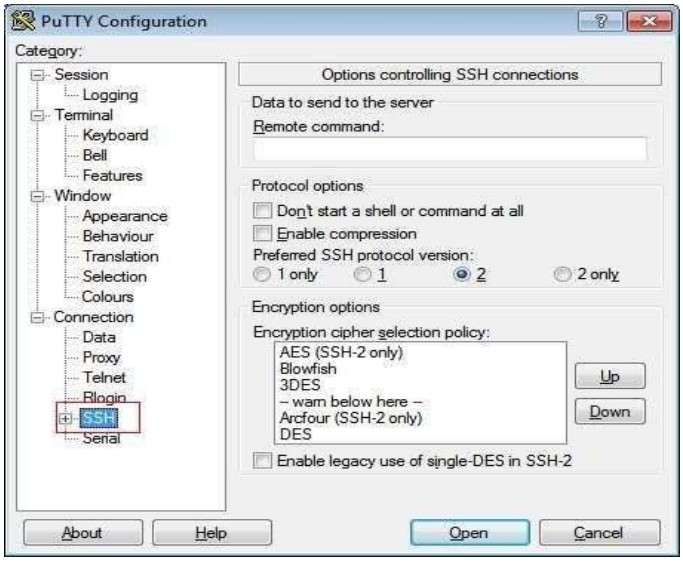


6. Confirm that the **When username is not specified** option is set to **Prompt**.



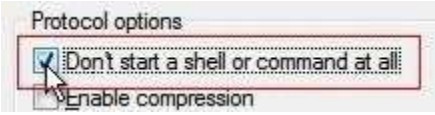
7. In the Categorytree, click **SSH**.

The Options controlling SSH connections panel is displayed.



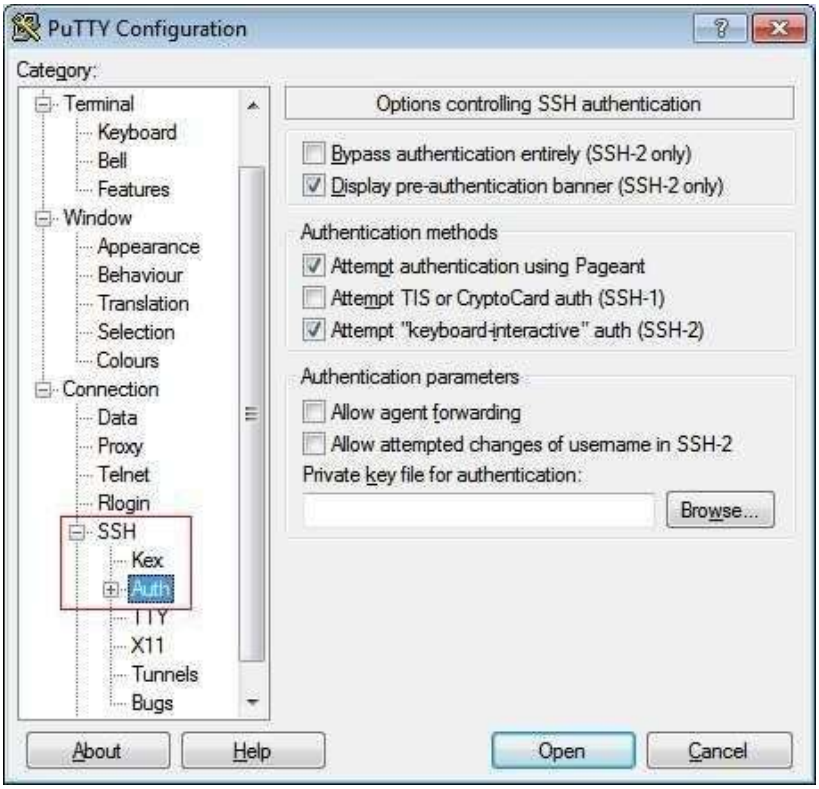
8. Under **Protocol options**, check **Don't start a shell command at all**.

This optional step ensures that only the SSH tunnel is enabled. You will not be able to use the SSH session to run commands in the command shell (although you will be able to enter the pass phrase for your SSH key, as described later in this tutorial).



9. In the Categorytree, expand **SSH**, and then click **Auth**.

   The Options controlling SSH authentication panel is displayed.



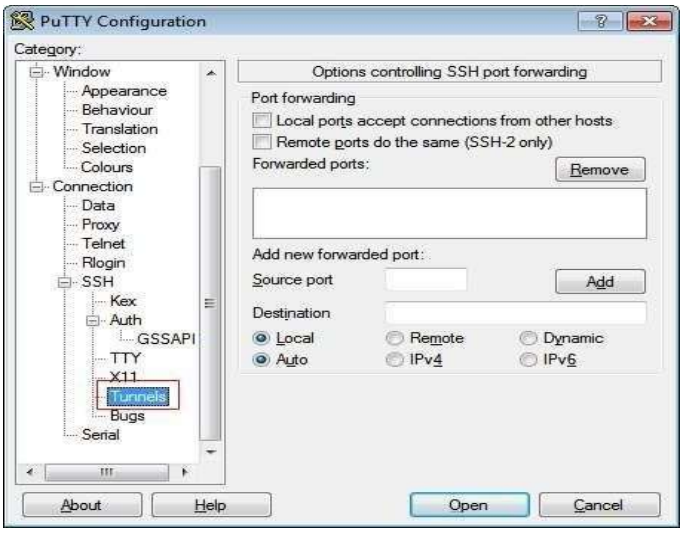
10. Under **Private key file for authentication**, click **Browse.**

11. In the **Select private key file** window, click **PuTTY Private KeyFiles(.ppk)** to find and open the private key file that matches the public key used when the instance was created.

Note: The .ppk extension indicates that the private key is in PuTTY's proprietary format. You must use a key of this format when using PuTTY. If you have to use a key saved in a different

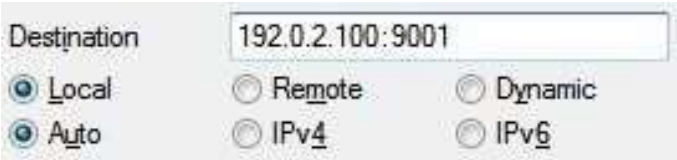
format, see the PuTTY documentation.

12. In the Categorytree, click **Tunnels**.The Options controlling SSH port for warding panel is displayed.



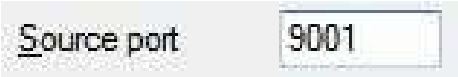
13. In the **Destination** box, enteradmin_server_ip:9001, where admin_server_ip is the public IPaddress for the Administration Server that you found and recorded earlier in this tutorial. Also select **Local** and **Auto**, if they aren't already selected.

Note: Port9001isthedefaultportcreatedspecificallyforconnectingtotheAdministrationServer via SSH tunnels.

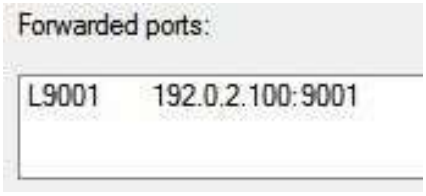


14. In the **Source Port** box, type 9001, to match the VM's port number.

Note: It is not a general requirement of SSH tunnels that the port numbers match. However, it is a requirement of the JMX/RMI protocol that is used for communicating with the port on the Administration Server.



15. Click **Add** to add the forwarded port. The local and remote ports appear in the Forwarded ports list.



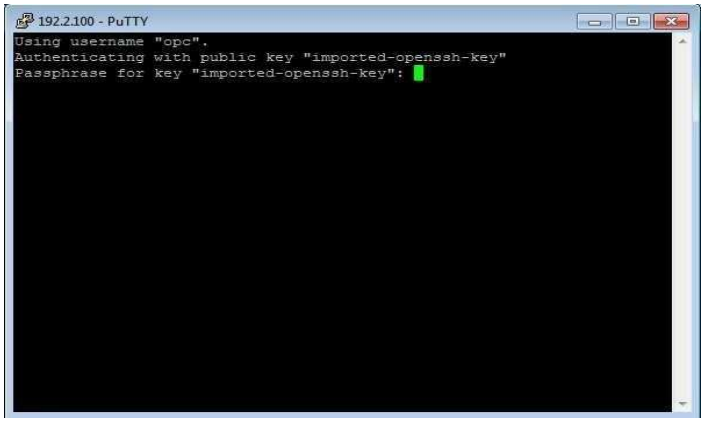
16. In the Categorytree, click **Session** to display the Session panel again.

17. In the **Saved Sessions** box, enter a name for this connection configuration. For the tutorial, type vm_session, and then click **Save**. When you open PuTTY the next time, you can load this configuration by selecting it and clicking **Load**.

18. Click **Open** to open the connection to the VM. If this is the first time you are connecting to the VM, the PuTTY Security Alert window is displayed, prompting you to confirm the public key.

Click **Yes** to continue connecting.

ThePuTTYConfigurationwindowclosesandthePuTTYcommandwindowisdisplayed.Theuser name is the value you supplied earlier, in the **Auto-login username box** in step 5.



19. When prompted, enter the pass phrase for the key, if one was defined.

20. The tunnel is now created. Any packets sent to the client's port 9001 will reach the VM's port 9001. In this tutorial, you established a connection between port 9001 on your client and port 9001 on the VM that hosts Web LogicServer. So you can now access port 9001on the Administration Server(on the VM) by connecting to your client's port 9001.

Note: This "port forwarding" established by the tunnel is different from a regular SSH session, which simply provides the secure shell for logging into and issuing commands on a remote computer.

You can see that the tunnel is working by displaying the Web LogicServer Administration Console for the service instance. Enter the following into a web browser's address bar:

localhost:9001/console

The Web LogicServer Administration Consolelog- in page is displayed.

21. Login with your usual Web LogicServer administration log-in credentials for the Oracle Java Cloud Service instance.

The WebLogic Server Administration Console is displayed. You can see from the localhost:9001 in the address that the tunnel is working. The local port is being forwarded to the remote host(VM) and port.

Also, under Domain Structure in the Administration Console, you can see the name of the WebLogic Server domain in your Oracle Java Cloud Service instance.

CREATING AN SSH TUNNEL AT THE COMMANDLINE:

1. In a commandline shell, set the file permissions of the private key file so that only you have access to it:

$chmod600private-key-file

Where private-key-file is the path to the SSH private key file that matches the public key used when your Oracle Java Cloud Service instance was created.

2. Run the ssh utility:

$ssh-iprivate-key-file-Llocal-port:vm-ip-address:vm-portopc@vm-ip-address-N

where:

a) -L specifies that the local (client) port is to be forwarded to the remote host and port. This allocates a socket to listen to the local port. Whenever a connection is made to the local port, it is forwarded over these cure channel, and a connection is made to the host port from the remote machine.

b) **private-key-file** is the path to the SSH private key file that matches the public key used when your instance was created.

c) **Note:** When using an SSH client at the command line, you cannot use a private key saved in PuTTY (.ppk) format. If the only private key you have is in PuTTY (.ppk) format, see the PuTTY documentation for instructions on how to convert it to OpenSSH format.

d) **local-port** is the port number on your client.

e) BecausetheVMuses9001as the listen port for the Administration Server, and because the client and the VM port numbers must match, use 9001 for local-port.

f) **vm-ip-address** is the IP address of the VM in x.x.x.x format. For an Oracle Java Cloud Service instance, you can use the IP address of the Administration Server or the IP address of the load balancer, if one is enabled.

g) **vm-port**istheportnumberontheVMtowhichyouwanttocreateatunnel.OracleJava Cloud Service uses 9001 as the listen port for the Administration Server. Therefore, use 9001 as the vm-port.

h) **opc@vm-ip-address** is the user account(opc) and the VM's IP address.

i) -N ensures only the SSH tunnel is enabled. You will not be able to use the SSH session to run commands in the command shell.

For example:

$ssh-ikeys/id_rsa-L9001:192.0.2.100:9001opc@192.0.2.100

3. If this is the first time you are connecting to the VM, the ssh utility prompts you to confirm the public key. In response to the prompt, enter yes.

4. At the prompt, enter the passphrase for the SSH key, if one was created.

```
$ ssh -i keys/id_rsa -L 9001:192.0.2.100:9001 opc@192.0.2.100
The authenticity of host '192.0.2.100(192.0.2.100)' can't be established.
RSA key fingerprint is [redacted].
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '192.0.2.100' (RSA) to the list of known hosts.
Enter passphrase for key 'keys/id_rsa':
Authorized uses only. All activity may be monitored and reported.
[opc@jcsdoc-wls-1 ~]$
```

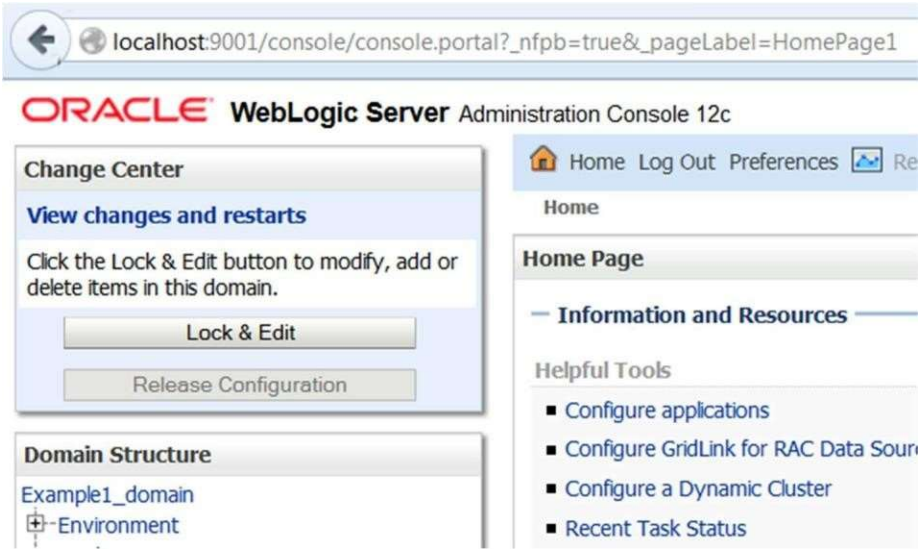
5. Now you are logged in, and you can use any resources of that server as if it were local.

6. You can see that the tunnel is working by displaying the Web Logic Server Administration Console for the service instance. Enter the following into a web browser's address bar:

localhost:9001/console

The Web logic Server Administration Console log-in page is displayed.

7. Login with your usual Web Logic Server administration log-in credentials for the Oracle Java Cloud Service instance. The WebLogic Server Administration Console for your instance is displayed.



RESULT:

Thus, the implementation of SSH protocol on multiple virtual machines was successfully implemented and the output was verified.

Ex. No:4	INSTALLATION AND CONFIGURATION OF DOCKER SERVICES ON LINUX

AIM:

To install and configure Docker services on Linux.

PROCEDURE:

Step-1. Login to Linux host and check prerequisite

Dockercomesin3flavors:

- DockerEngine–Community
- DockerEngine–Enterprise
- DockerEnterprise

1.1) Login to Linux host.

Digital Ocean droplet as linuxhost.

```
bash-3.2$ ssh root@45.56.94.4
root@45.56.94.4's password:
Welcome to Ubuntu 19.04 (GNU/Linux 5.0.0-13-generic x86_64)
```

1.2) Check Linux OS Architecture:

We need 64-bit architecture.

```
root@localhost:~# arch

x86_64
```

1.3) Check Linux OS Kernel Level:

Kernel Level should be greater than 3.0.

```
root@localhost:~# uname -r

5.0.0-13-generic
```

Step-2. Install Docker Community Edition

```
root@localhost:~# curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key
OK

root@localhost:~#
```

2.1) Make sure to add Docker's official PGP key to system.

```
root@localhost:~# sudo apt-key fingerprint 0EBFCD88 | grep 5822

Warning: apt-key output should not be parsed (stdout is not a terminal)
      9DC8 5822 9FC7 DD38 854A  E2D8 8D81 803C 0EBF CD88
```

2.2) Make sure we got right PGP key:

2.3) Make sure to install Ubuntu OS to latest version

Command: sudoapt-getupdate

```
root@localhost:~# sudo apt-get update

Hit:1 http://mirrors.linode.com/ubuntu disco InRelease
Get:2 http://mirrors.linode.com/ubuntu disco-updates InRelease [97.5 kB]
Get:3 http://mirrors.linode.com/ubuntu disco-backports InRelease [88.8 kB]
Hit:4 https://download.docker.com/linux/ubuntu disco InRelease
Get:5 http://mirrors.linode.com/ubuntu disco-updates/main amd64 Packages [226 kB]
Get:6 http://security.ubuntu.com/ubuntu disco-security InRelease [97.5 kB]
Get:7 http://mirrors.linode.com/ubuntu disco-updates/main i386 Packages [194 kB]
Get:8 http://mirrors.linode.com/ubuntu disco-updates/universe i386 Packages [252 kB]
Get:9 http://mirrors.linode.com/ubuntu disco-updates/universe amd64 Packages [254 kB]
Get:10 http://mirrors.linode.com/ubuntu disco-updates/universe Translation-en [80.1 kB]
Get:11 http://security.ubuntu.com/ubuntu disco-security/main i386 Packages [144 kB]
Get:12 http://security.ubuntu.com/ubuntu disco-security/main amd64 Packages [173 kB]
Get:13 http://security.ubuntu.com/ubuntu disco-security/universe amd64 Packages [220 kB]
Get:14 http://security.ubuntu.com/ubuntu disco-security/universe i386 Packages [217 kB]
Get:15 http://security.ubuntu.com/ubuntu disco-security/universe Translation-en [55.4 kB]
Fetched 2,098 kB in 2s (1,188 kB/s)
Reading package lists... Done
```

2.4) Set up latest stable Docker Repository

Command: sudo add-apt-repository "deb [arch=amd64]

https://download.docker.com/linux/ubuntu$(lsb_release-cs)stable"

```
root@localhost:~# sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/

Hit:1 http://mirrors.linode.com/ubuntu disco InRelease
Hit:2 http://mirrors.linode.com/ubuntu disco-updates InRelease
Hit:3 http://mirrors.linode.com/ubuntu disco-backports InRelease
Hit:4 https://download.docker.com/linux/ubuntu disco InRelease
Hit:5 http://security.ubuntu.com/ubuntu disco-security InRelease
Reading package lists... Done
```

2.5) Install Docker

Command: sudoapt-getinstalldocker-cer

```
root@localhost:~# sudo apt-get install docker-ce

Reading package lists... Done
Building dependency tree
Reading state information... Done
The following packages were automatically installed and are no longer required:
  bridge-utils dns-root-data dnsmasq-base ubuntu-fan
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  aufs-tools containerd.io docker-ce-cli libltdl7
The following packages will be REMOVED:
  containerd docker.io runc
The following NEW packages will be installed:
  aufs-tools containerd.io docker-ce docker-ce-cli libltdl7
0 upgraded, 5 newly installed, 3 to remove and 101 not upgraded.
Need to get 87.9 MB of archives.
After this operation, 133 MB of additional disk space will be used.
Do you want to continue? [Y/n] Y

Get:1 http://mirrors.linode.com/ubuntu disco/universe amd64 aufs-tools amd64 1:4.9+20170
Get:2 http://mirrors.linode.com/ubuntu disco/main amd64 libltdl7 amd64 2.4.6-10 [38.3 kB
Get:3 https://download.docker.com/linux/ubuntu disco/stable amd64 containerd.io amd64 1.
Get:4 https://download.docker.com/linux/ubuntu disco/stable amd64 docker-ce-cli amd64 5:
Get:5 https://download.docker.com/linux/ubuntu disco/stable amd64 docker-ce amd64 5:19.0
Fetched 87.9 MB in 3s (34.9 MB/s)
(Reading database ... 83080 files and directories currently installed.)
Removing docker.io (18.09.5-0ubuntu1) ...
'/usr/share/docker.io/contrib/nuke-graph-directory.sh' -> '/var/lib/docker/nuke-graph-di
Removing containerd (1.2.6-0ubuntu1) ...
Removing runc (1.0.0~rc7+git20190403.029124da-0ubuntu1) ...
Selecting previously unselected package aufs-tools.
(Reading database ... 82836 files and directories currently installed.)
Preparing to unpack .../aufs-tools_1%3a4.9+20170918-2_amd64.deb ...
Unpacking aufs-tools (1:4.9+20170918-2) ...
Selecting previously unselected package containerd.io.
Preparing to unpack .../containerd.io_1.2.6-3_amd64.deb ...
Unpacking containerd.io (1.2.6-3) ...
Selecting previously unselected package docker-ce-cli.
Preparing to unpack .../docker-ce-cli_5%3a19.03.1~3-0~ubuntu-disco_amd64.deb ...
Unpacking docker-ce-cli (5:19.03.1~3-0~ubuntu-disco) ...
Selecting previously unselected package docker-ce.
Preparing to unpack .../docker-ce_5%3a19.03.1~3-0~ubuntu-disco_amd64.deb ...
Unpacking docker-ce (5:19.03.1~3-0~ubuntu-disco) ...
Selecting previously unselected package libltdl7:amd64.
Preparing to unpack .../libltdl7_2.4.6-10_amd64.deb ...
Unpacking libltdl7:amd64 (2.4.6-10) ...
Setting up aufs-tools (1:4.9+20170918-2) ...
Setting up containerd.io (1.2.6-3) ...
Setting up libltdl7:amd64 (2.4.6-10) ...
Setting up docker-ce-cli (5:19.03.1~3-0~ubuntu-disco) ...
Setting up docker-ce (5:19.03.1~3-0~ubuntu-disco) ...
Installing new version of config file /etc/init.d/docker ...
Installing new version of config file /etc/init/docker.conf ...
Created symlink /etc/systemd/system/multi-user.target.wants/docker.service → /lib/systemd
Processing triggers for systemd (240-6ubuntu5) ...
Processing triggers for man-db (2.8.5-2) ...
Processing triggers for libc-bin (2.29-0ubuntu2) ...
```

Check Docker version

```
root@localhost:~# docker -v
Docker version 19.03.1, build 74b1e89
```

Step-3. Start Docker and run HelloWorld

```
root@localhost:~# sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest: sha256:6540fc08ee6e6b7b63468dc3317e3303aae178cb8a45ed3123180328bcc1d20f
Status: Downloaded newer image for hello-world:latest

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

3.1 Simple command to run Docker on Linux

As we above during installation Docker registers itself as a system service: /lib/systemd/system/docker.service.

```
root@localhost:~# sudo systemctl start docker
```

3.2 Let'spullHelloWorldDockerandrun

Command:sudodockerrunhello-world

```
root@localhost:~# sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest: sha256:6540fc08ee6e6b7b63468dc3317e3303aae178cb8a45ed3123180328bcc1d20f
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:~#
```

Above command will pull docker image and we will be able to run Docker Hello World application.

Step-4. How to check Docker Images/repositories? Before running Hello World App:

```
root@localhost:~# docker images
REPOSITORY                    TAG       IMAGE ID       CREATED
keyansoftwares/logviewersvc   latest    04612d6a457d   2 months ago
keyansoftwares/logreadagent   latest    0385a757deb5   2 months ago
keyansoftwares/logreadagent   <none>    d75a02abb278   2 months ago
grafana/grafana               latest    f96bf1723e2a   3 months ago
```

After running Hello World App:

As we see here-we will see new repository hello-world below

```
root@localhost:~# docker images
REPOSITORY                    TAG                 IMAGE ID            CREATED
keyansoftwares/logviewersvc   latest              04612d6a457d        2 months ago
keyansoftwares/logreadagent   latest              0385a757deb5        2 months ago
keyansoftwares/logreadagent   <none>              d75a02abb278        2 months ago
grafana/grafana               latest              f96bf1723e2a        3 months ago
hello-world                   latest              fce289e99eb9        7 months ago
```

Congratulations. we are all set. We have setup your Linux environment for Docker, Install Docker and ran your 1st Docker Application too.

Extra Step:

We recommend you to follow these post-install steps.

How to auto-start docker after VM reboot?

Command: sudo system ctl enable docker

```
root@localhost:~# sudo systemctl enable docker

Synchronizing state of docker.service with SysV service script with /lib/systemd/systemd-
Executing: /lib/systemd/systemd-sysv-install enable docker
```

Just execute above command and docker will start automatically next time you reboot VM/host.

Want to install Docker on CentOS?

JustreplaceStep-2abovewithbelowsteps:

```
root@localhost:~# sudo yum install -y yum-utils device-mapper-persistent-data lvm2

root@localhost:~# sudo yum-config-manager --add-repo https://download.docker.com/linux/ce

root@localhost:~# sudo yum install docker-ce docker-ce-cli containerd.io

root@localhost:~# sudo systemctl start docker

root@localhost:~# sudo docker run hello-world
```

RESULT:

Thus, the installation and configuration of Docker services on LINUX was executed successfully and the output was verified.

Ex. No. 5	EXPLORING THE INSTALLATION AND CONFIGURATION OF CHEF AND PUPPET

AIM:

To explore the installation and configuration of DevOps tools Chef and Puppet.

INSTALLATION AND CONFIGURATION OF CHEF:

PRERQUISITES:

General Requirements

- Three backend servers; as many frontend servers as required
- 1 x GigE NIC

Front end Requirements

- 4 cores(physical or virtual)
- 4GB RAM
- 20GB off reedisk space(SSDifonpremises,PremiumStorageinMicrosoftAzure,EBS-OptimizedGP2inAWS)

Backend Requirements

- 2 cores(physical or virtual)
- 8GB RAM
- 50GB/backend server(SSD if on premises, Premium Storage in MicrosoftAzure,EBS-Optimized GP2 in AWS)

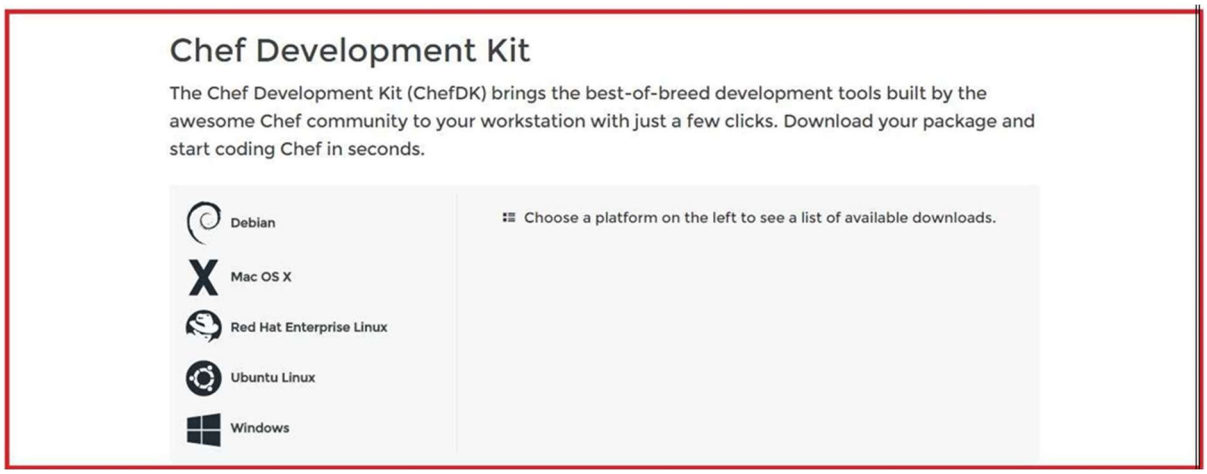
PROCEDURE:

Following are the steps to install Chef:

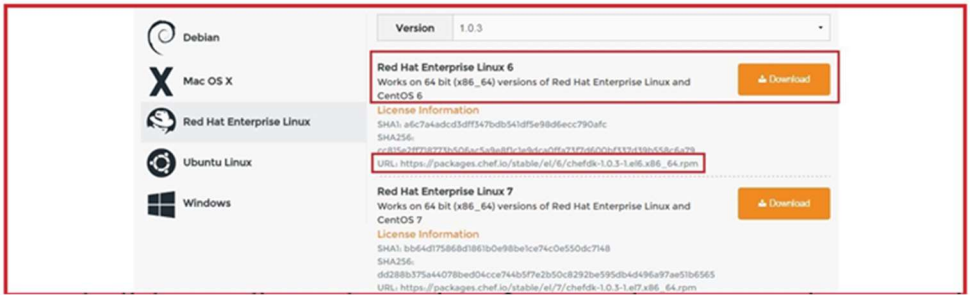
1. Install Chef DK(Development Kit) on Chef Workstation
2. Set up a Chef Server
3. Create a Recipe or a Cookbook/download a Cook book from Chef Supermarket in Workstation
4. Upload a Cook book on the Chef Server
5. Connect A Node To The Chef Server
6. Deploy the Cook book from the Server to the Node

1. Install Chef DK(Development Kit)

In my Chef Work station I will install Chef DK. Chef DK is a package that contains all the development tools that you will need when coding Chef. Here is the link to download *ChefDK*.



Here, choose the operating system that you are using. I am using CentOS 6.8. So, I will click on Red Hat Enterprise Linux.



Copy the link according to the version of CentOS that you are using. I am using CentOS 6, as you can see that I have highlighted in the above screenshot.

Go to your Workstation terminal and download the Chef DK by using wget command and paste the link.

Execute this command:

Wget https://packages.chef.io/stable/el/6/chefdk-1.0.3-1.el6.x86_64.rpm

```
[root@Workstation ~]# wget https://packages.chef.io/stable/el/6/chefdk-1.0.3-1.el6.x86_64.rpm
--2016-11-25 07:40:38--  https://packages.chef.io/stable/el/6/chefdk-1.0.3-1.el6.x86_64.rpm
Resolving packages.chef.io... 151.101.8.65
Connecting to packages.chef.io|151.101.8.65|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://packages.chef.io/files/stable/chefdk/1.0.3/el/6/chefdk-1.0.3-1.el6.x86_64.rpm [following]
--2016-11-25 07:40:44--  https://packages.chef.io/files/stable/chefdk/1.0.3/el/6/chefdk-1.0.3-1.el6.x86_64.rpm
Reusing existing connection to packages.chef.io:443.
HTTP request sent, awaiting response... 200 OK
Length: 105276444 (100M) [application/x-rpm]
Saving to: “chefdk-1.0.3-1.el6.x86_64.rpm”

100%[======================================================================>] 105,276,444  300K/s   in 3m 19s

2016-11-25 07:44:15 (517 KB/s) - “chefdk-1.0.3-1.el6.x86_64.rpm” saved [105276444/105276444]
```

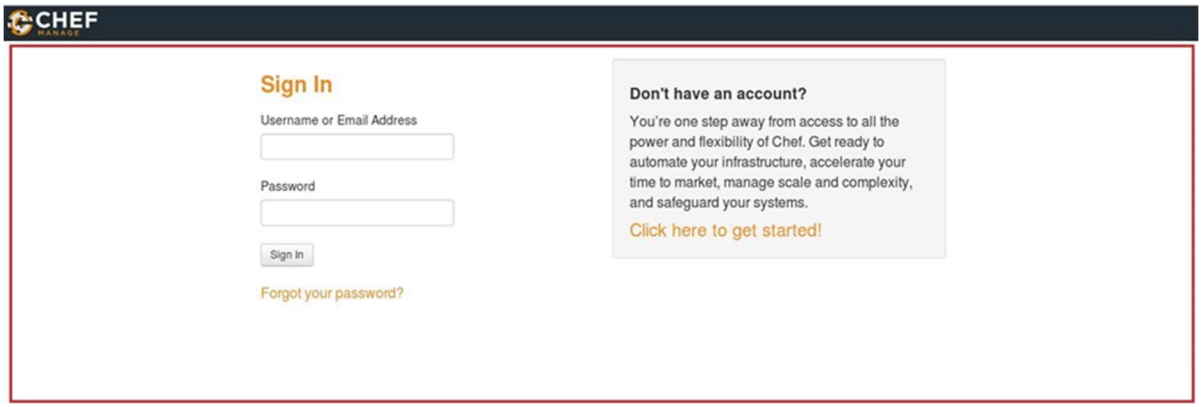
The package is now downloaded. Now I will install this package using rpm.

Execute this:

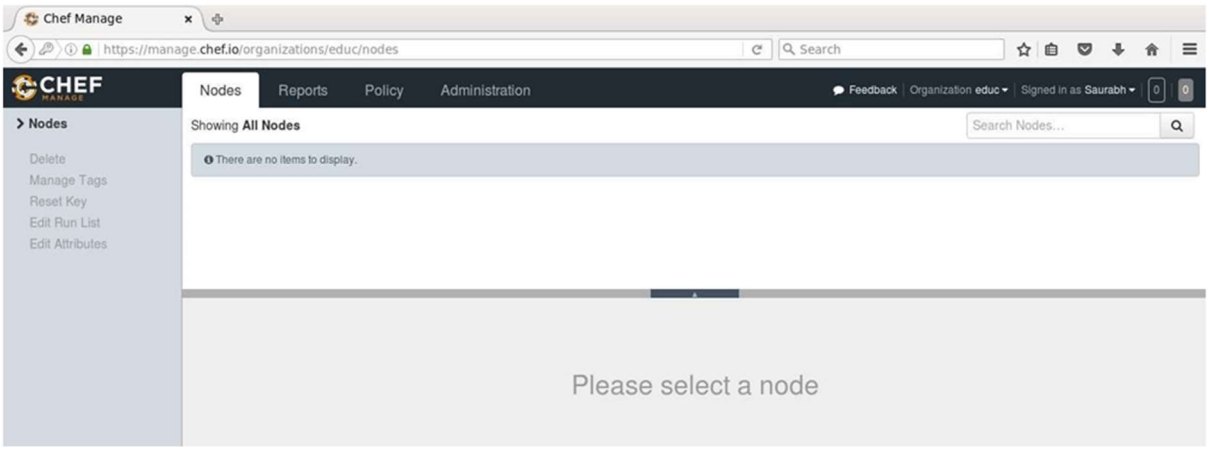
rpm-ivhchefdk-1.0.3-1.el6.x86_64.rpm

2. Setup Chef Server

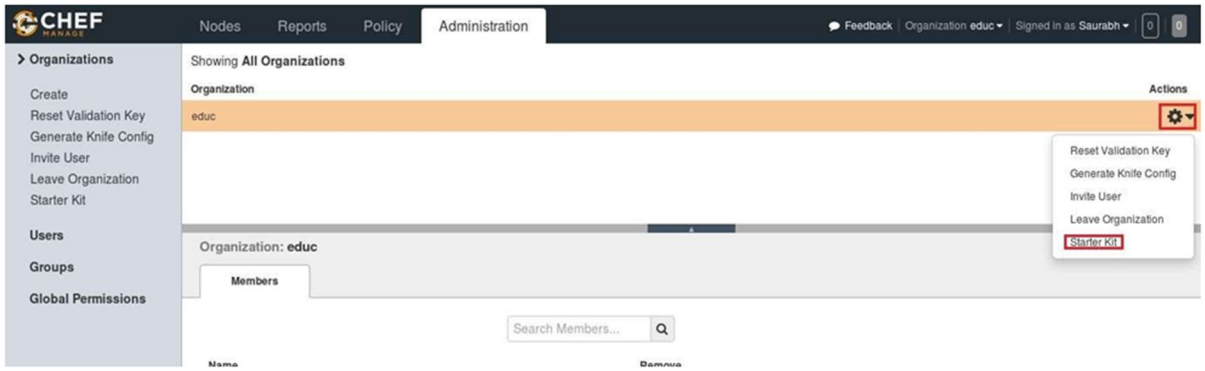
I will use the hosted version of Chef Server on the cloud but you can use a physical machine as well. This Chef-Server is present at manage.chef.io



Over here, create an account if you don’t have one. Once you have created an account, sign-in with your login credentials.



This is how Chef Server looks like.



Now you will get an option to download the Starter Kit. Just click on it to download the Starter Kit zip file.

Move this file to your root directory. Now unzip this file by using unzip command in your terminal. You will notice that it includes a directory called chef-repo.

Execute this:

Unzip chef-starter.zip

```
[root@Workstation ~]# unzip chef-starter.zip
Archive:  chef-starter.zip
  inflating: chef-repo/cookbooks/chefignore
   creating: chef-repo/cookbooks/starter/
   creating: chef-repo/cookbooks/starter/files/
   creating: chef-repo/cookbooks/starter/files/default/
  inflating: chef-repo/cookbooks/starter/files/default/sample.txt
  inflating: chef-repo/cookbooks/starter/metadata.rb
   creating: chef-repo/cookbooks/starter/attributes/
  inflating: chef-repo/cookbooks/starter/attributes/default.rb
   creating: chef-repo/cookbooks/starter/templates/
   creating: chef-repo/cookbooks/starter/templates/default/
  inflating: chef-repo/cookbooks/starter/templates/default/sample.erb
   creating: chef-repo/cookbooks/starter/recipes/
  inflating: chef-repo/cookbooks/starter/recipes/default.rb
  inflating: chef-repo/README.md
  inflating: chef-repo/.gitignore
   creating: chef-repo/.chef/
   creating: chef-repo/roles/
  inflating: chef-repo/.chef/knife.rb
  inflating: chef-repo/roles/starter.rb
  inflating: chef-repo/.chef/saurabh010.pem
```

Now move this starter kit to the cookbook directory in chef-repo directory.

Execute this:

mv starter /root/chef-repo/cookbooks

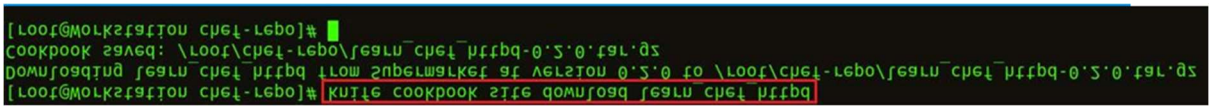
3. Download A Cookbook From Chef Supermarket In Workstation

Chef Cookbooks are available in the Cookbook Supermarket, we can go to the Chef Supermarket. Download the required Cookbooks from supermarket.chef.io. I'm downloading one of the Cookbook to install Apache from there.

Execute this:

cd chef-repo

knife cookbook site download learn_chef_httpd



There is Tar ball downloaded for the Apache Cookbook. Now, I will extract the contents from this downloaded Tar file. For that, I will use tar command.

Execute this:

tar -xvf learn_chef_httpd-0.2.0.tar.gz

All the required files are automatically created under this Cookbook. There is no need to make any modifications. Let's check the Recipe description inside my recipe folder.

Execute this:

cd /root/chef-repo/learn_chef_httpd/recipes

cat default.rb

```
[root@Workstation recipes]# cat default.rb
#
# Cookbook Name:: learn_chef_httpd
# Recipe:: default
#
# Copyright (C) 2014
#
#
#
package 'httpd'

service 'httpd' do
  action [:enable, :start]
end

template '/var/www/html/index.html' do
  source 'index.html.erb'
end

service 'iptables' do
  action :stop
end
[root@Workstation recipes]#
```

4.Upload A Cookbook In The Chef Server

In order to upload the Apache Cookbook that I have downloaded, first move this learn_chef_httpd file to the Cookbooks folder in the chef-repo. Then change your directory to cookbooks.

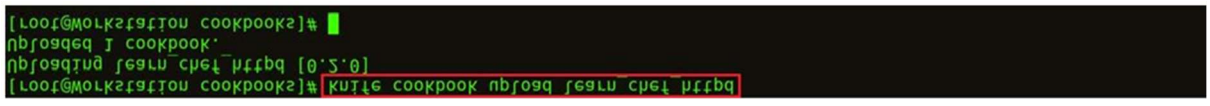
Execute this:

mv /root/chef-repo/learn_chef_httpd /root/chef-repo/cookbooks

cd /root/chef-repo/cookbooks

Next

knife cookbook upload learn_chef_httpd



Now, our final step is to add Chef Node. We've setup a Workstation, a Chef Server and we need to add our Nodes to the Chef Server for automation.

5.Connect A Node To The Chef Server

The terminal color of my Node machine is different from the Workstation so that you will be able to differentiate between both.

I just need the IP address of my Node for that I will execute the below command in my Node machine.

Execute this:

Ifconfig

```
[root@ChefNode ~]# ifconfig
eth1      Link encap:Ethernet  HWaddr 08:00:27:BA:23:1B
          inet addr:10.0.2.16  Bcast:10.255.255.255  Mask:255.0.0.0
          inet6 addr: fe80::a00:27ff:feba:231b/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:62 errors:0 dropped:0 overruns:0 frame:0
          TX packets:66 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:7822 (7.6 KiB)  TX bytes:5885 (5.7 KiB)

eth2      Link encap:Ethernet  HWaddr 08:00:27:85:36:02
          inet addr:192.168.56.102  Bcast:192.168.56.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fe85:3602/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:1909 errors:0 dropped:0 overruns:0 frame:0
          TX packets:34 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:254899 (248.9 KiB)  TX bytes:8260 (8.0 KiB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
```

I will add my Chef Node to the Server by executing Knife Bootstrap command in which I will specify the IP address of The Chef Node and its name. Execute the command shown below:

Execute this:

knife bootstrap 192.168.56.102 --ssh-user root --ssh-password edureka --node-name chefNode

```
[root@Workstation cookbooks]# knife bootstrap 192.168.56.102 --ssh-user root --ssh-password edureka --node-name
chefNode
Node chefNode exists, overwrite it? (Y/N) y
Client chefNode exists, overwrite it? (Y/N) y
Creating new client for chefNode
Creating new node for chefNode
Connecting to 192.168.56.102
192.168.56.102 -----> Installing Chef Omnibus (-v 12)
192.168.56.102 downloading https://omnitruck-direct.chef.io/chef/install.sh
192.168.56.102   to file /tmp/install.sh.5743/install.sh
192.168.56.102 trying wget...
192.168.56.102 el 6 x86_64
192.168.56.102 Getting information for chef stable 12 for el...
192.168.56.102 downloading https://omnitruck-direct.chef.io/stable/chef/metadata?v=12&p=el&pv=6&m=x86_64
192.168.56.102   to file /tmp/install.sh.5754/metadata.txt
192.168.56.102 trying wget...
192.168.56.102 sha1      abe23132483c1ff015148ad72becf092c0fac428
192.168.56.102 sha256    bca097d9107bd4b20df88045b676a1d032b68a8b66bcf751dabc2e58715ae0ae
192.168.56.102 url       https://packages.chef.io/files/stable/chef/12.16.42/el/6/chef-12.16.42-1.el6.x86_64.rpm
192.168.56.102 version   12.16.42
192.168.56.102 downloaded metadata file looks valid...
192.168.56.102 downloading https://packages.chef.io/files/stable/chef/12.16.42/el/6/chef-12.16.42-1.el6.x86_64.r
```

```
192.168.56.102   to file /tmp/install.sh.5754/chef-12.16.42-1.el6.x86_64.rpm
192.168.56.102 trying wget...
192.168.56.102 Comparing checksum with sha256sum...
192.168.56.102 Installing chef 12
192.168.56.102 installing with rpm...
192.168.56.102 warning: /tmp/install.sh.5754/chef-12.16.42-1.el6.x86_64.rpm: Header V4 DSA/SHA1 Signature, key I
D 83ef826a: NOKEY
192.168.56.102 Preparing...                ########################################### [100%]
192.168.56.102    1:chef                   ########################################### [100%]
192.168.56.102 Thank you for installing Chef!
192.168.56.102 Starting the first Chef Client run...
192.168.56.102 Starting Chef Client, version 12.16.42
192.168.56.102 resolving cookbooks for run list: []
192.168.56.102 Synchronizing Cookbooks:
192.168.56.102 Installing Cookbook Gems:
192.168.56.102 Compiling Cookbooks...
192.168.56.102 [2016-11-28T11:13:27+00:00] WARN: Node chefNode has an empty run list.
192.168.56.102 Converging 0 resources
192.168.56.102
192.168.56.102 Running handlers:
192.168.56.102 Running handlers complete
192.168.56.102 Chef Client finished, 0/0 resources updated in 49 seconds
```

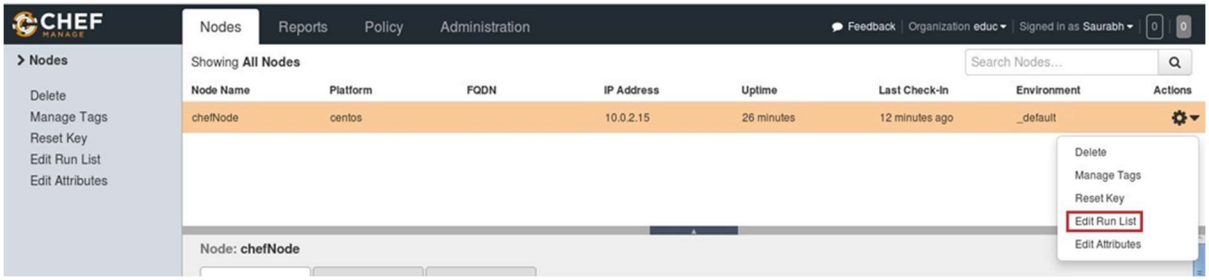
This command will also initialize the installation of the Chef-Client in the Chef Node. You can verify it from the CLI on the Workstation using the knife command, as shown below:

Execute this:

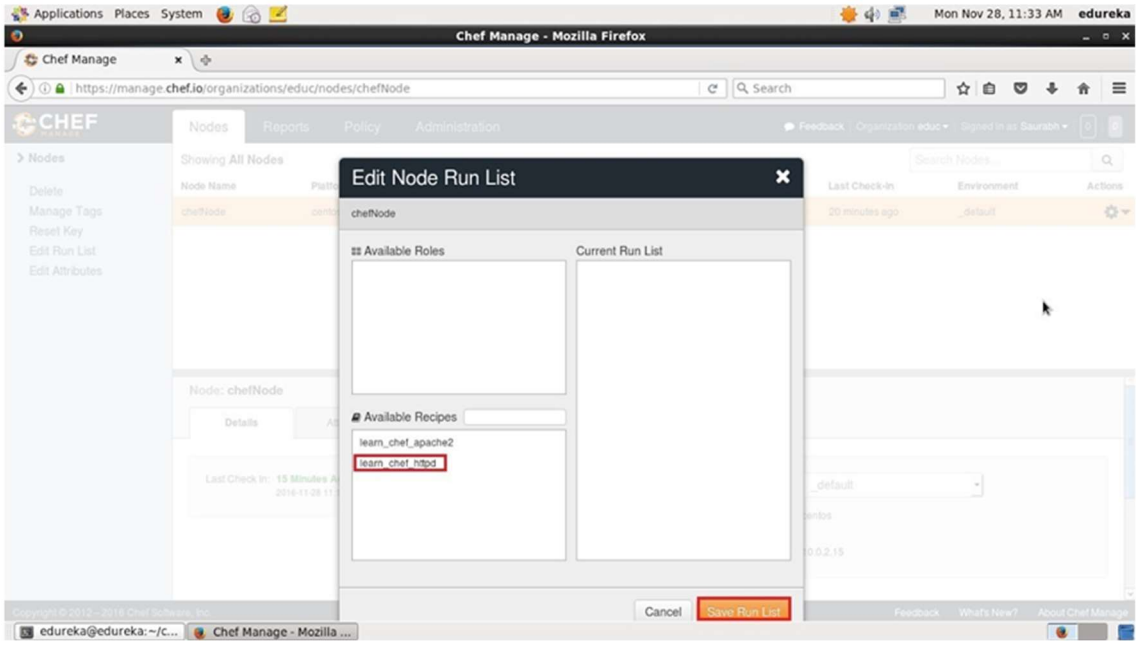
Knife node list

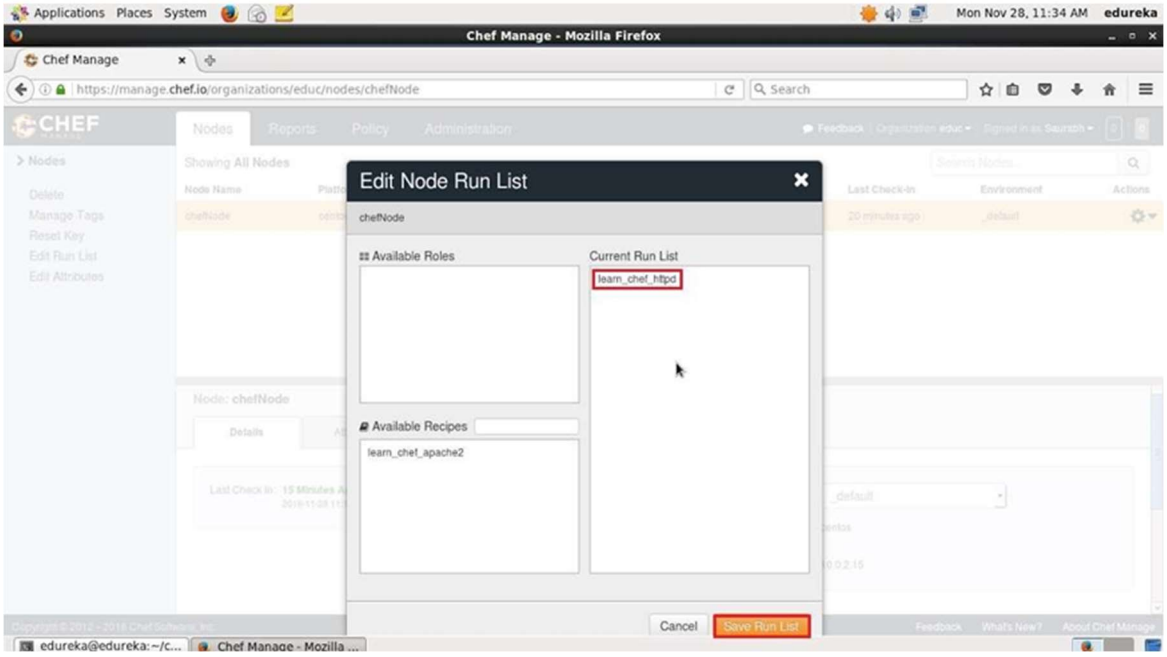
```
[root@Workstation cookbooks]# knife node list
chefNode
[root@Workstation cookbooks]#
```

6. Deploy the Cookbook From The Server To The Node



In the Available Recipes, you can see ourlearn_chef_httpd Recipe, you can drag that from the available packages to the current Run List and save the Run list.





Execute this:

chef-client

```
[root@ChefNode ~]# chef-client
Starting Chef Client, version 12.16.42
resolving cookbooks for run list: ["learn_chef_httpd"]
Synchronizing Cookbooks:
  - learn_chef_httpd (0.2.0)
Installing Cookbook Gems:
Compiling Cookbooks...
Converging 4 resources
Recipe: learn_chef_httpd::default
  * yum_package[httpd] action install (up to date)
  * service[httpd] action enable
    - enable service service[httpd]
  * service[httpd] action start
    - start service service[httpd]
  * template[/var/www/html/index.html] action create
    - create new file /var/www/html/index.html
    - update content in file /var/www/html/index.html from none to ef4ffd
    --- /var/www/html/index.html        2016-11-28 11:47:17.414304893 +0000
    +++ /var/www/html/.chef-index20161128-6284-hzok74.html      2016-11-28 11:47:17.414304893 +0000
    @@ -1 +1,6 @@
    +<html>
```

INSTALLATION AND CONFIGURATION OF PUPPET:

PRERQUISITES:

- **Hardware requirements:**

  These hardware requirements are based on internal testing at Puppet and are meant only as guidelines to help you determine your hardware needs.

- **Supported operating systems:**

  Puppet Enterprise supports various operating systems depending on the role a machine assumes in your infrastructure.

- **Supported browsers:**

  The following browsers are supported for use with the console.

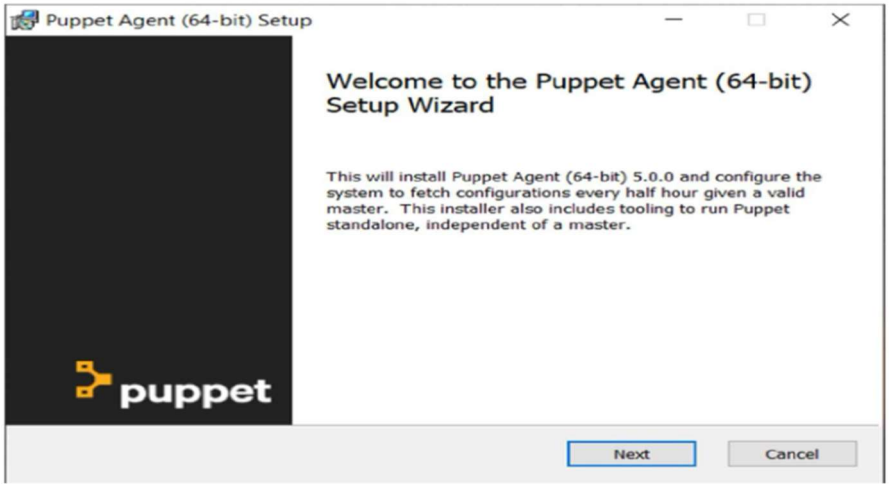
  Google chrome, Mozilla firefox, Microsoft Edge and Apple Safari(10 or Later);

- **System configuration:**

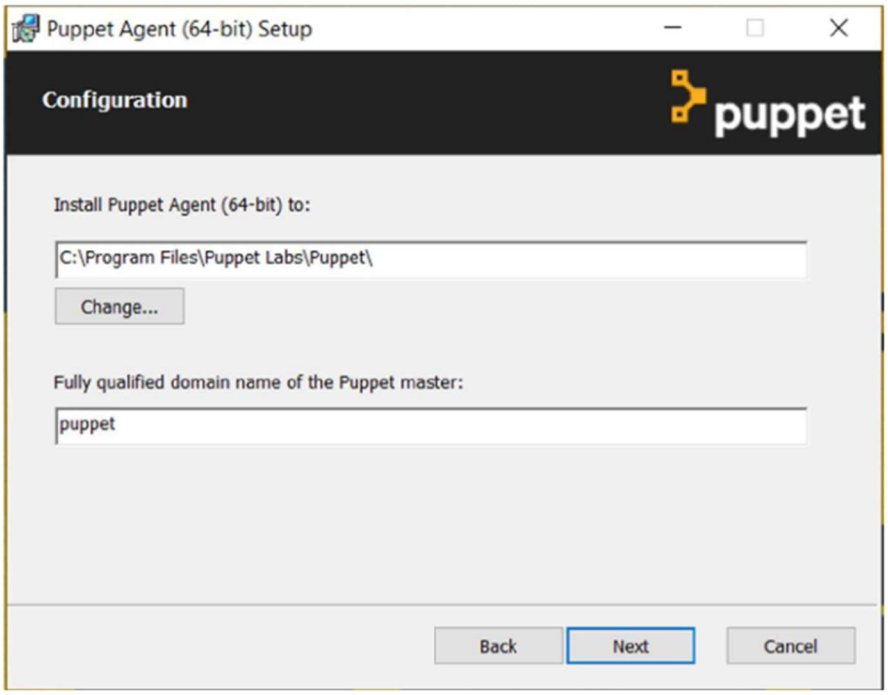
  Before installing Puppet Enterprise, make sure that your nodes and network are properly configured.

PROCEDURE:

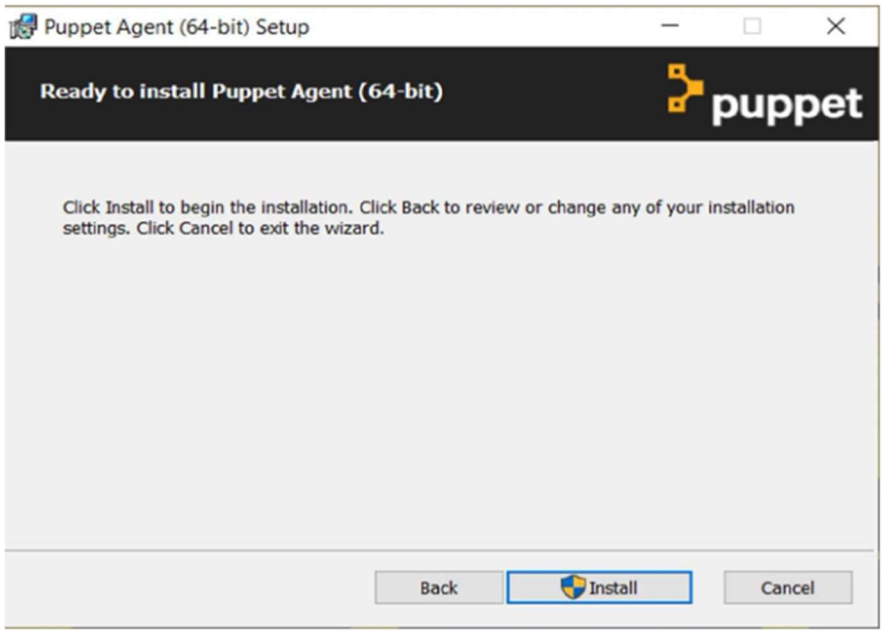
1. Download the puppet agent .msi file from the official site. Open the setup wizard.



2. Check the ‘Accept the terms and conditions’ option and Click Next.
3. Set the path for the puppet Agent and the domain name for the puppet master.



4. Click ‘Install’ to begin Installation.



RESULT:

Thus, the experiment to explore the installation and configuration of Chef and Puppet tools was executed and verified successfully.

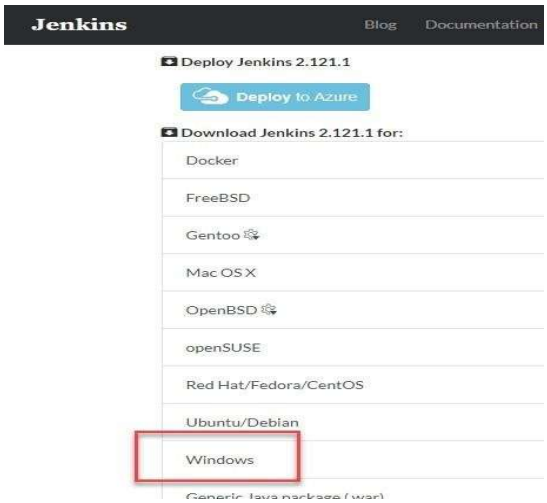
Ex. No. 6	INSTALLATION AND CONFIGURATION OF JENKINS AND MVN TOOLS

AIM:

To install and configure Jenkins and MVN tools.

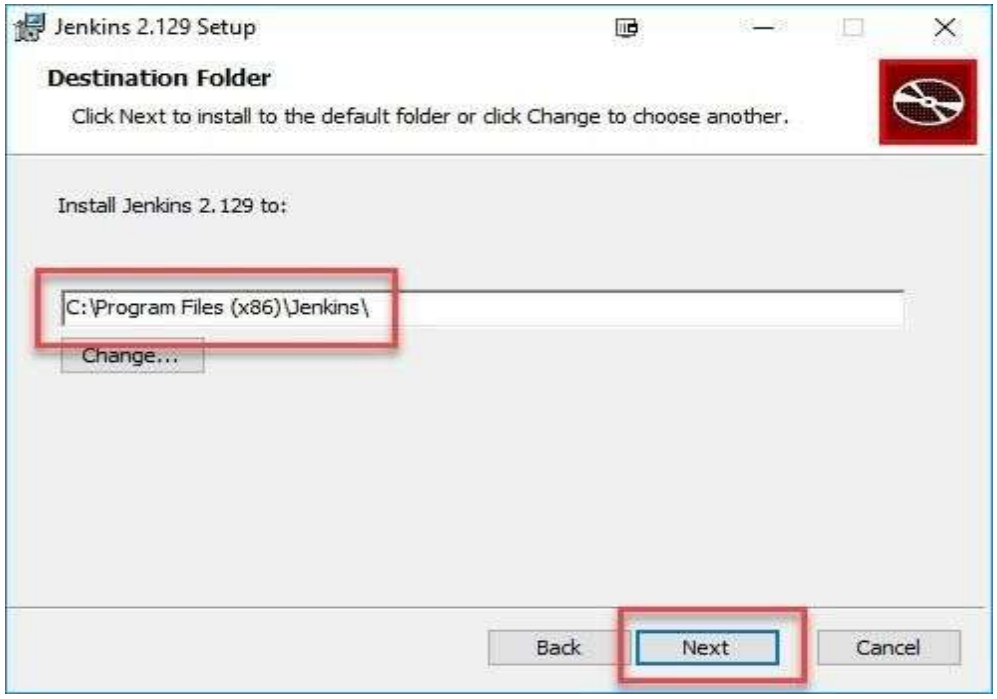
PROCEDURE:

STEP 1: Got to **https://jenkins.io/download/** and select the platform. Select ***WINDOWS*** as OS.



STEP 2: Open the 'Jenkins.msi' file. In the setup wizard, Click Next.

STEP 3: Choose the location where you want to have the Jenkins instance installed (defaultlocation is C:\ProgramFiles (x86)\Jenkins), then click on **Next** button.



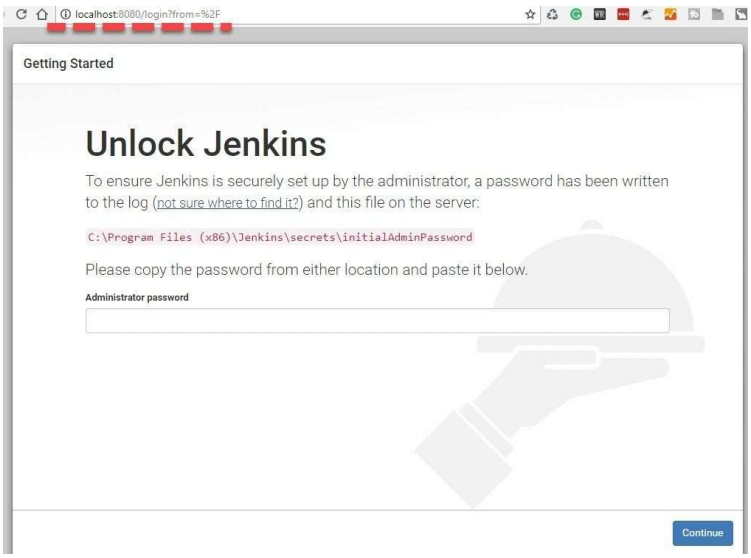
STEP 4: Click on the Install button. Once the installation is complete, Click Finish.

STEP 5: After completing the Jenkins installation phase, you should proceed further and start its configuration. Next steps will guide you how you can unlock Jenkins application:

i. After completing the Jenkins installation process, a browser tab will pop-up asking for the initial Administrator password. To access Jenkins, you need to go to browsethe following path in your web browser.

http://localhost:8080

If you can access the above URL, then it confirms that Jenkins issuccessfully installed in your system.



ii. The initial Administrator password should be found under the Jenkinsinstallation path.

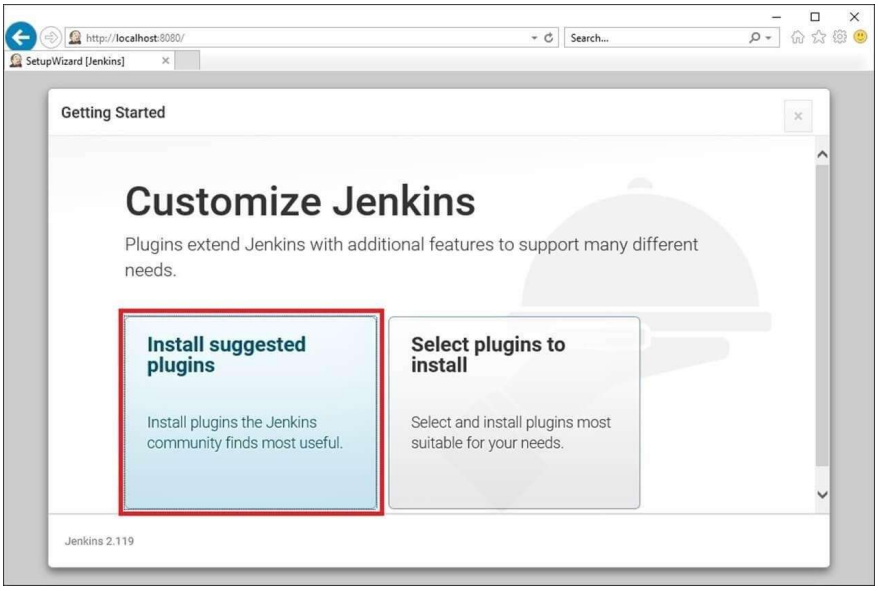
iii. For default installation location to C:\Program Files (x86)\Jenkins, a file called **initial Admin Password** can be found under ***C:\Program Files (x86)\Jenkins\secrets.***

iv. However, If a custom path for Jenkins installation was selected, thenyou should check that location for **initial Admin Password** file.

v. Open the highlighted file and copy the content of the **initial Admin Password** file.

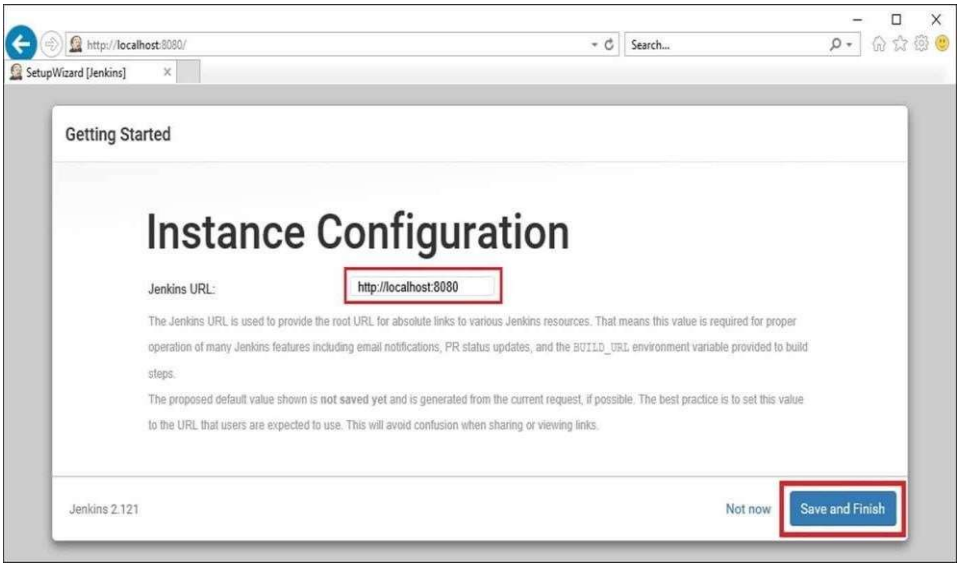
vi. Paste the password it into browser's pop-up tab (**http://localhost:8080/login?form=%2F**) and click on Continue button

vii. If the Jenkins needs to be customized, the necessary plugins can be installed.

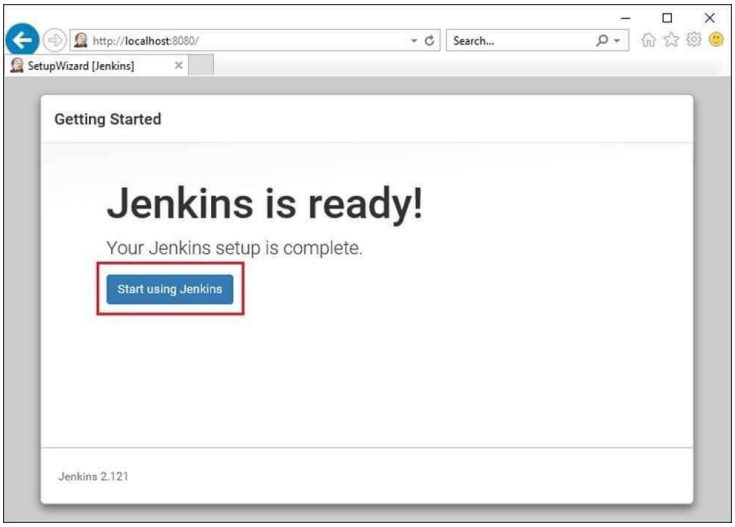


viii. After all suggested plugins were installed, the "Create First Admin User" panel will show up. Fill all the fields with desired account detailsand hit the "**Save and Finish**" button.

ix. Finally it will ask for URL information where you can configure the default instance path for Jenkins. Hit the "**Save and Continue**" button.



x. The Jenkins Server has successfully been installed

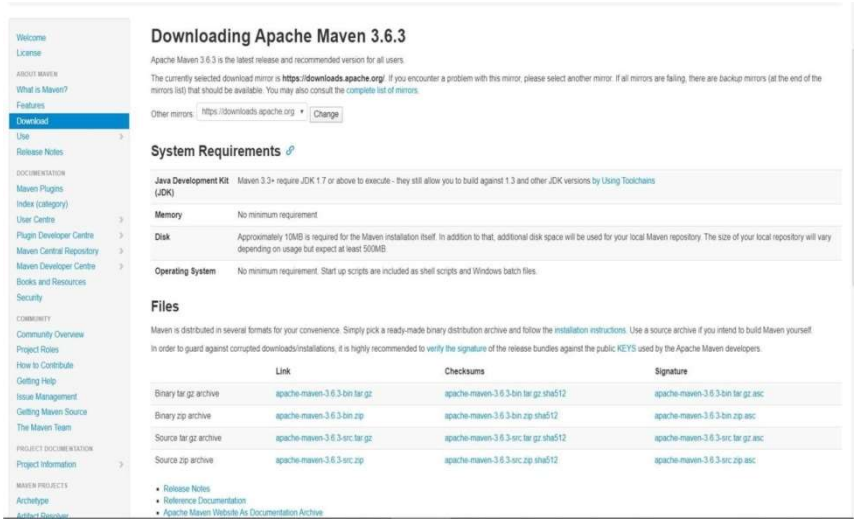


INSTALLATION AND CONFIGURATION OF MVN TOOLS:

The installation of Apache Maven is a simple process of extracting the archive andadding the `bin` folder with the‘mvn’ command to the ‘PATH’.

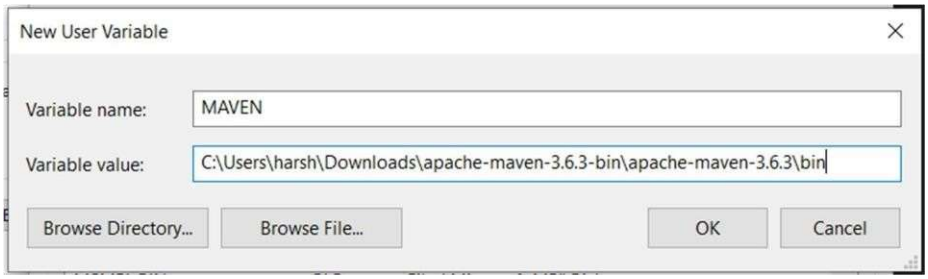
PROCEDURE:

1. From the Apache Maven site, download the Maven zip file of the latest version, and un zip the file to the folder that you want to use Maven in.

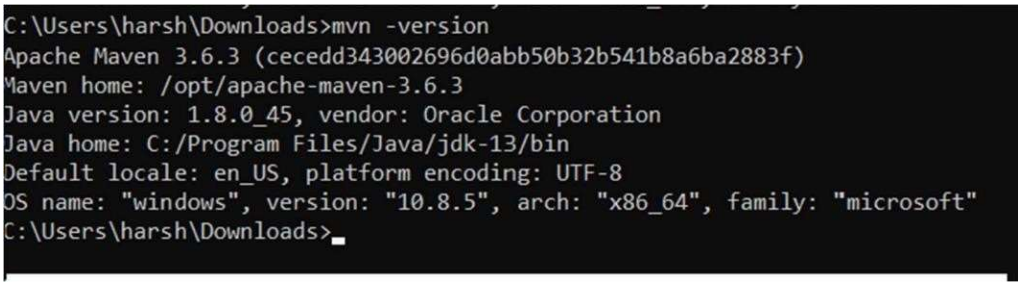


2. Add both M2_HOME and MAVEN_HOME variables to the Windows environment variables (using system properties), and point them to your Maven folder.

3. Update the PATH variable by appending the Maven bin folder %M2_HOME%\bin. This way Maven’s commands can be run anywhere.



4. To check whether or not the above steps have been completed successfully, run the following command in the command prompt: mvn –version.
5. The command should display the Maven version, the Java version, and the operating system information.



RESULT:

Thus, the experiment to explore the installation and configuration of Jenkins and MVN tools was executed and verified successfully.

Ex. No. 7	EXPLORING THE SERVICES OF ELK – (ELASTIC SEARCH, LOGSTASH, KIBANA)

AIM:

To explore ELK Stack, which is a collection of three open-source products - Elasticsearch, Logstash, and Kibana.

PRERQUISITES:

- OS: Ubuntu 14.04
- RAM: 4GB
- CPU: 2
- JDK 8
- Nginx

PROCEDURE:

1) Install Elastic Search:

   $ sudo apt-get -y install elastic search

2) Open elastic search.yml and edit following the line “network.host: localhost” to lock access down to the localhost.

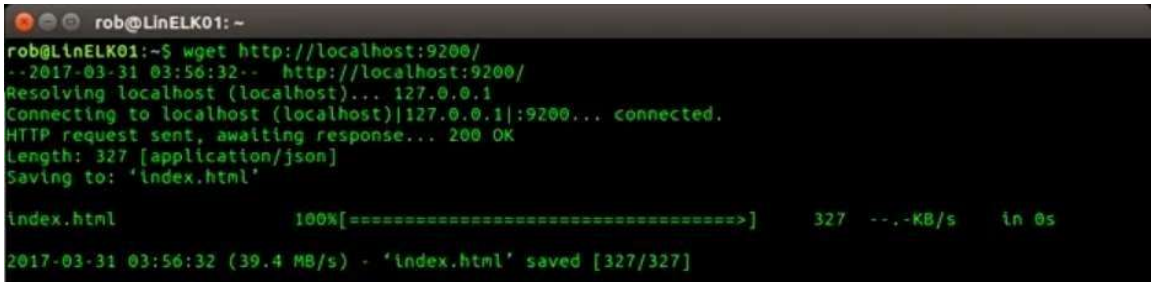
3) Restart the service and enable it to start with the server:

   $ sudo service elastic search restart

   $ sudo system ctl enable elastic search

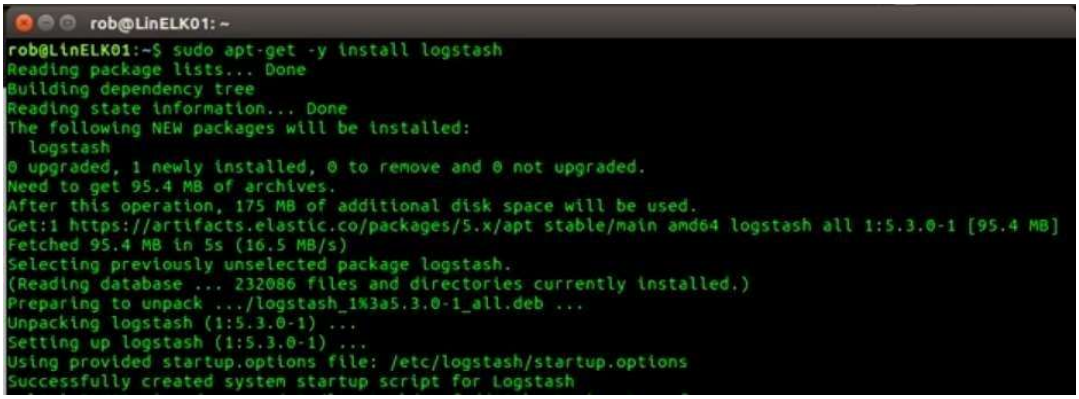
4) Verify Elasticsearch is running:

   $ wget http://localhost:9200/



5) Install Logstash:

$ sudo apt-get -y install logstash



6) Create and open the syslog filter configuration file (logstash/conf.d/10-syslog-filter.conf) and add the following tothe configuration file

```
filter {
if [type] == "syslog" {grok {
match => { "message" => "%{SYSLOGTIMESTAMP:syslog_timestamp}
%{SYSLOGHOST:syslog_hostname}
%{DATA:syslog_program}(?:\[%{POSINT:syslog_pid}\])?:
%{GREEDYDATA:syslog_message}" }
add_field => [ "received_at", "%{@timestamp}" ]add_field => [ "received_from",
"%{host}" ]
}
syslog_pri { }date {
match => [ "syslog_timestamp", "MMM d HH:mm:ss", "MMM dd HH:mm:ss" ]
}
}
}
```

7) Create and open the Elasticsearch output configuration file (logstash/conf.d/30-elasticsearch-output.conf) and addthe following to the configuration file:

```
output { elasticsearch
{
hosts => ["localhost:9200"]sniffing => true manage_template => false
index => "%{[@metadata][beat]}-%{+YYYY.MM.dd}"document_type =>
```

"%{[@metadata][type]}"

}

}

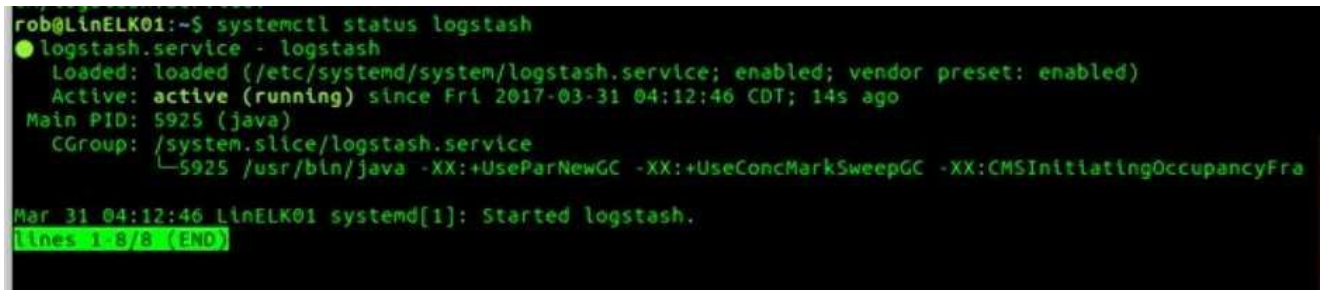
8) Restart Logstash and enable it to start with the server:

$ sudo service logstash restart

$ sudo systemctl enable logstash

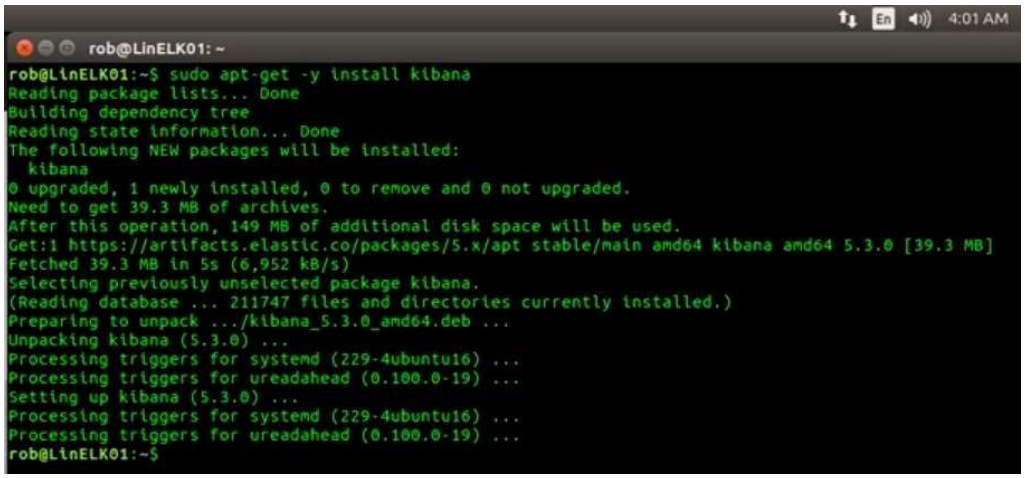
9) Verify the service is running using the command:

$ systemctl status logstash



10) Install Kibana:

$ sudo apt-get -y install kibana



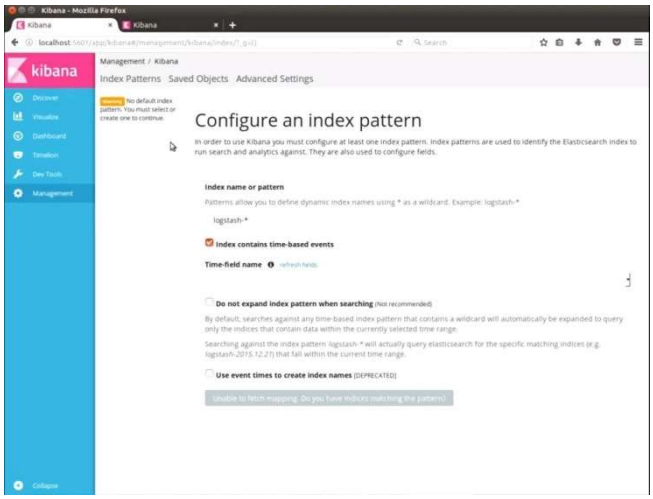
11) Open kibana.yml and edit following the line “network.host: localhost” to lock access down to the localhost.

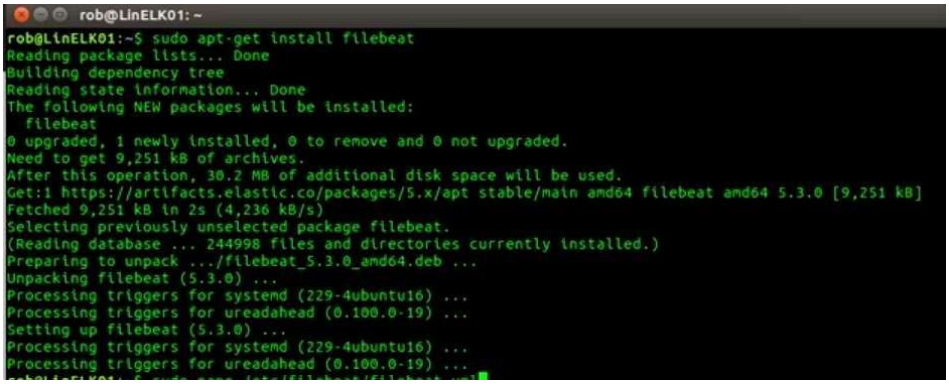
12) Restart the service and enable it to start with the server:

$ sudo service kibana restart

$ sudo systemctl enable kibana

13) Verify Kibana is running by browsing to http://localhost:5601/



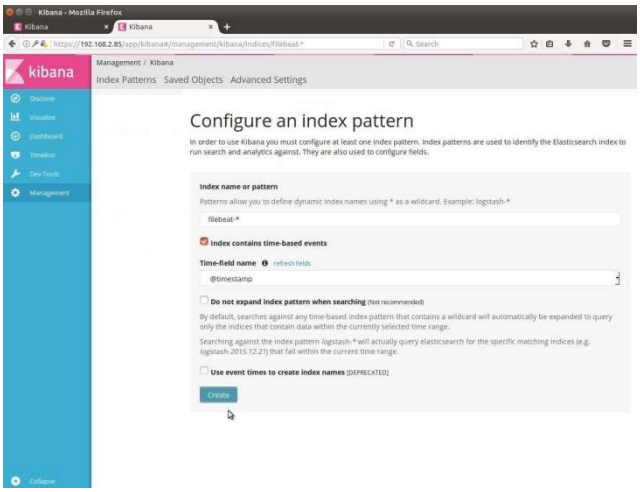


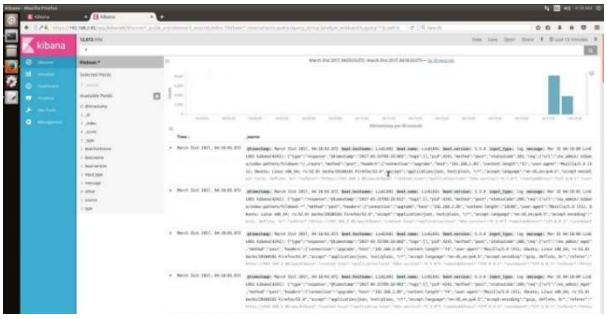
14) The Elastic Stack uses several lightweight data shippers called Beats to collect data from various sources and transport them to Logstash or Elasticsearch. Install filebeat using the command :

$ sudo apt-get install filebeat

15) Go back to Kibana and configure the index pattern for Filebeat: *filebeat-**

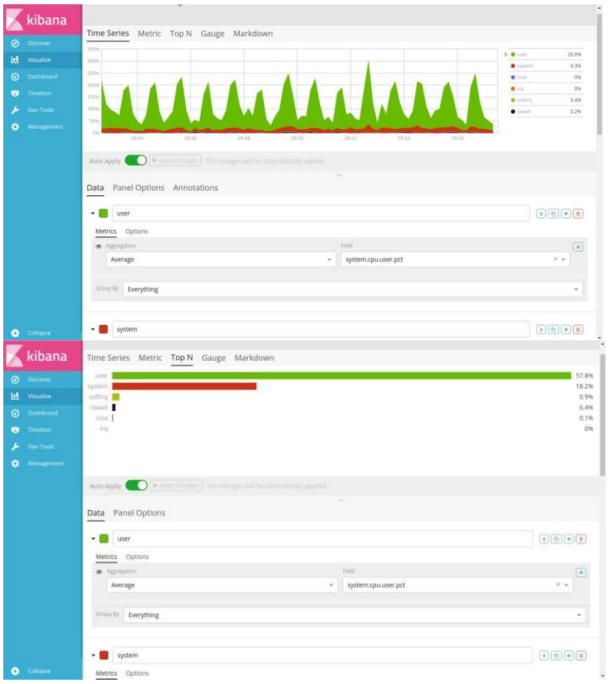
Click the “Create” button.

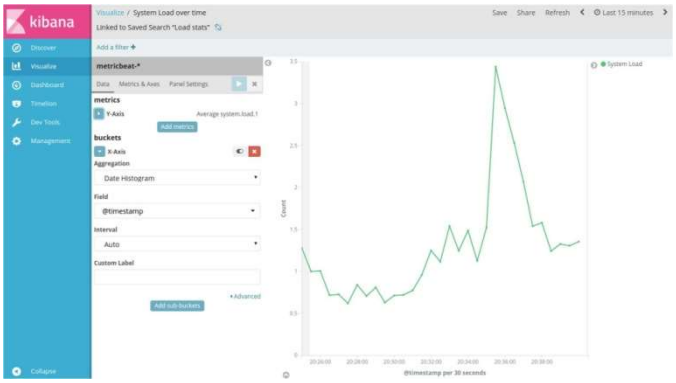


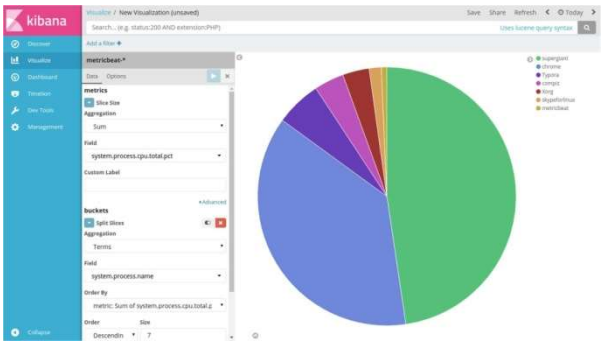


16) Now, we can search against the filebeat index pattern and view the logs that filebeat is feeding the stack:

17) In the “Visualize” tab, various methods to visualize the data can be explored.







RESULT:

Thus, the experiment to explore the ELK stack was executed and verified successfully.

Ex. No. 8	SETTING UP OF LAMP SERVICES USING NGINX ENGINE

AIM:

To install Nginx, MariaDB and PHP7-FPM(LEMPstack) on Ubuntu16.04

PROCEDURE:

Step 1: Install Nginx Web Server

sudo apt install nginx

sudosystemctl enable nginx

Then start Nginx with this command:

sudosystemctl start nginx

Now check out its status.

systemctl status nginx

OUTPUT:

```
● nginx.service - A high performance web server and a reverse proxy serve
r
   Loaded: loaded (/lib/systemd/system/nginx.service; enabled; vendor pre
set: enabled)
   Active: active (running) since Sat 2016-06-04 08:31:23 EDT; 1 day 2h a
go
 Main PID: 298 (nginx)
   CGroup: /system.slice/nginx.service
     ├─298 nginx: master process /usr/sbin/nginx -g daemon on; master_pro
cess on
     └─299 nginx: worker process
```

"**enabled**" indicates that auto start at boot time is enabled and we can see that Nginx is running.

Now in your browser's address bar, type the public IP address of Ubuntu 16.04 LTS server. You should see the "Welcome to nginx!" Web page which means Nginx Web server is running correctly.

Use the following command to fetch the public IP address of Ubuntu 16.04 server.

sudo apt install curl

curl http://icanhazip.com

sudochown www-data /usr/share/nginx/html -R

Step2: InstallMariaDB

MariaDB is a drop-in replacement for MySQL. It is developed by former members of MySQL team who concerned that Oracle might turn MySQL into a closed-source product. Many Linux distributions (Arch Linux, Fedora etc), companies and organizations (Google, Wikipedia) have migrated to MariaDB. So, we're going to install MariaDB instead of MySQL.

Sudo apt install maria db-server maria db-client

After it's installed, MariaDB server should be automatically started. Use systemctl to check its status.

Systemctl status mysql

OUTPUT:

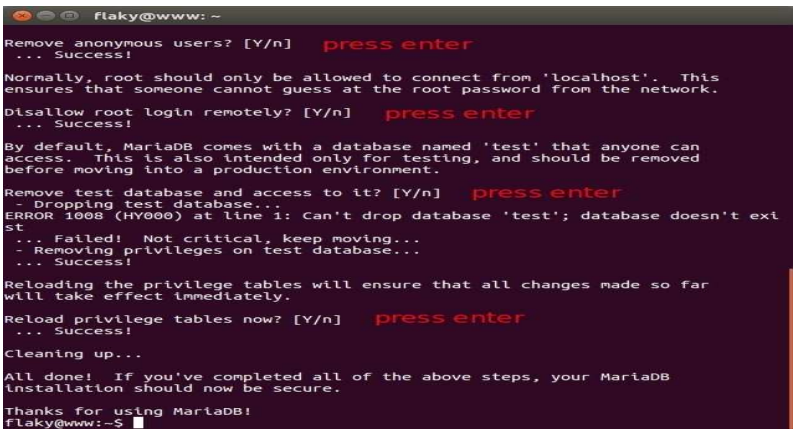
```
● mysql.service - LSB: Start and stop the mysql database server daemon
 Loaded: loaded (/etc/init.d/mysql; bad; vendor preset: enabled)
 Active: active (running) since Wed 2016-04-20 18:52:01 EDT; 1min 30s ago
 Docs: man:systemd-sysv-generator(8)
```

If it's not running, start it with this command:
sudosystemctl start mysql

To enable MariaDB to automatically start when Ubuntu 16.04 is rebooted:
sudosystemctl enable mysql

Now run the post installation security script.
sudomysql_secure_installation

Next you can just press Enter to answer all the remaining questions. This will remove anonymous user, disable remote root login and remove test database. This step is a basic requirement for MariaDB database security.



Step 3: Install PHP7

Enter the following command to install PHP7 and PHP7 extensions.
sudo apt install php7.0-fpm php7.0-mbstring php7.0-xml php7.0-mysql php7.0-common php7.0-gd php7.0-json php7.0-cli php7.0-curl

PHP extensions are commonly needed for content management systems (CMS) like WordPress. For example, if your installation lacks php7.0-xml, then some of your WordPress site page may be blank and you can find an error in Nginx error log like:
PHP message: PHP Fatal error: Uncaught Error: Call to undefined function xml_parser_create()

Installing these PHP extensions ensures that your CMS runs smoothly.

Nowstartphp7.0-fpm.
sudosystemctl start php7.0-fpm
Check status:
systemctl status php7.0-fpm

OUTPUT:

```
● php7.0-fpm.service - The PHP 7.0 FastCGI Process Manager
 Loaded: loaded (/lib/systemd/system/php7.0-fpm.service; enabled; vendor
pre
set: enabled)
 Active: active (running) since Wed 2016-04-20 19:21:05 EDT; 2s ago
```

Step 4: Create a Default Nginx Server Block File

Remove the **default** symlink in **sites-enabled** directory.

sudo rm /etc/nginx/sites-enabled/default

Then create a new default server block file under **/etc/nginx/conf.d/** directory.

sudo nano /etc/nginx/conf.d/default.conf

Paste the following text into the file. Replace 12.34.56.78 with your actual server IP.

```
server { listen 80;
listen [::]:80;
server_name 12.34.56.78; root /usr/share/nginx/html/;
index index.php index.html index.htm index.nginx-debian.html;

location / {
try_files $uri $uri/ =404;
}

error_page 404 /404.html;
error_page 500 502 503 504 /50x.html;

location = /50x.html {
root /usr/share/nginx/html;
}

location ~ \.php$ {
fastcgi_passunix:/run/php/php7.0-fpm.sock;
fastcgi_param SCRIPT_FILENAME $document_root$fastcgi_script_name; include
fastcgi_params;
include snippets/fastcgi-php.conf;
}

location ~ /\.ht{ deny all;
}
}
```

Save and close the file. Then test nginx configuration and reload it.
sudo nginx -t
sudosystemctl reload nginx

Step 5: Test PHP

To test the cli version of PHP7, we just need to enter this command:
php --version

Sample output:

```
PHP 7.0.15-7ubuntu2 (cli) ( NTS )
Copyright (c) 1997-2016 The PHP Group
Zend Engine v3.0.0, Copyright (c) 1998-2016 Zend Technologies
  with Zend OPcache v7.0.6-dev, Copyright (c) 1999-2016, by Zend Technolo
gies
```

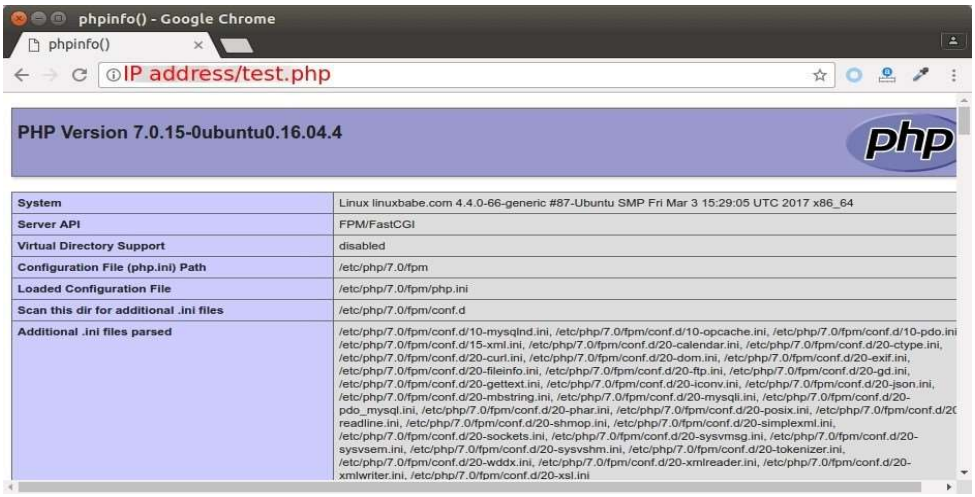
To test PHP-FPM, first create a

sudo nano /usr/share/nginx/html/test.php

Paste the following PHP code into the file.

<?phpphpinfo(); ?>

Save and close the file. Now in the browser address bar, enter
address with your actual IP. You should see your server's PHP information. This means PHP processing is fine.



How to Install PHP7.2

PHP7.2 is the latest stable version of PHP, released on November 30, 2017 and it has minor performance boost over PHP7.0. We can add the PPA from Ondrej Sury to install PHP7.2 on Ubuntu 17.10. That guy is also the maintainer of Certbot PPA.

sudo apt install software-properties-commonsudo add-apt-repository ppa:ondrej/php -ysudoapt

update

Then we can install PHP7.2 and common extensions by using the following command.

sudo apt install php7.2 php7.2-fpm php7.2-mysql php-common php7.2-cli php7.2-common php7.2-json php7.2-opcachephp7.2-readline php7.2-mbstringphp7.2-xml php7.2-gd php7.2-curl

NowstartPHP7.2-FPM.

sudosystemctlstartphp7.2-fpm

Enableauto-startatsystemboot time.

sudosystemctlenable php7.2-fpm

Checkitsstatus:

systemctl statusphp7.2-fpm

UsingPHP7.2-FPM withNginx

To make Nginx use PHP7.2-FPM instead of PHP7.0-FPM, we need to edit the Nginx server block file.

sudo nano /etc/nginx/conf.d/default.conf

Find the following line.

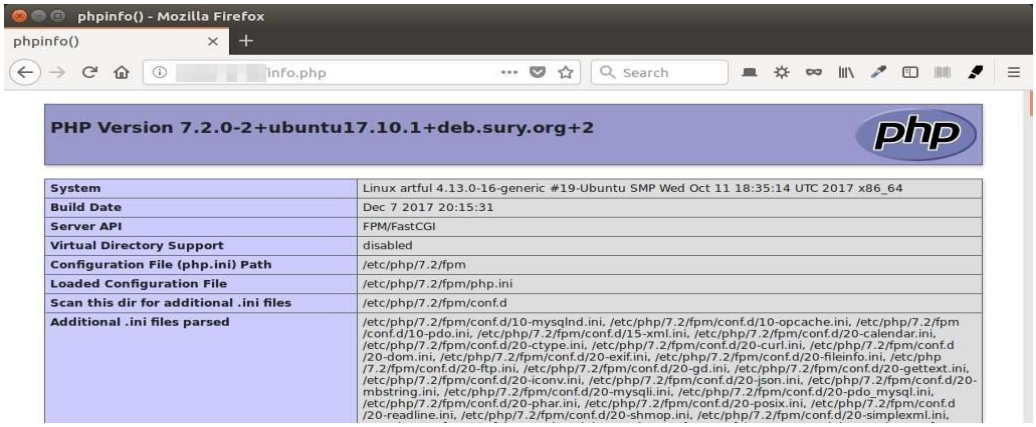
fastcgi_passunix:/run/php/php7.0-fpm.sock;

php7.0-fpmtophp7.2-fpm.

fastcgi_passunix:/run/php/php7.2-fpm.sock;

Save and close the file. Then reload Nginx for the change to take effect.

sudosystemctl reload nginx



RESULT:

Thus, Nginx, MariaDB and PHP7(LEMPstack) was installed on Ubuntu16.04 successfully.

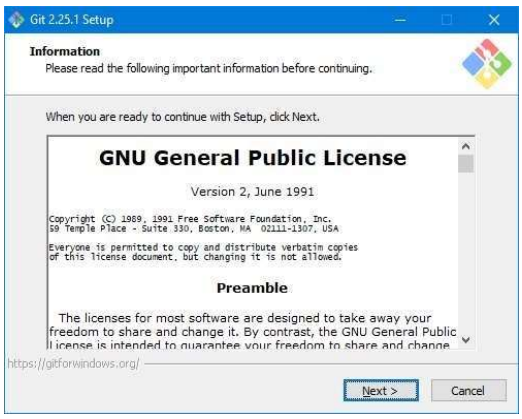
Ex. No. 9	SETTING OF VERSION CONTROL SYSTEM USING GIT

AIM:

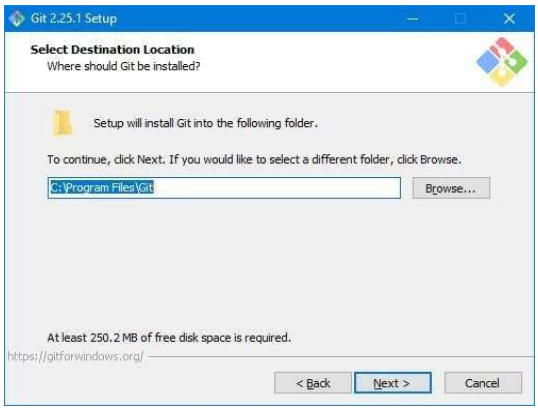
To setup version control system using git and basic operations of it.

PROCEDURE:

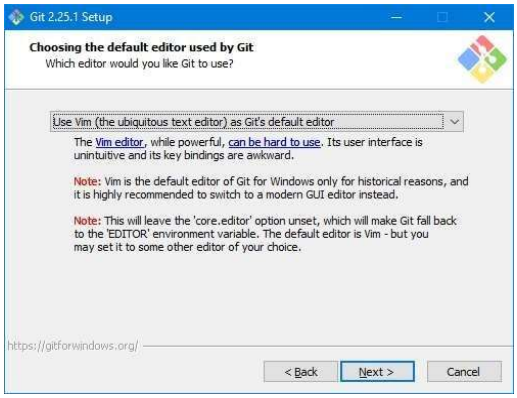
Step - 1: Download the latest version of git and run the setup file.



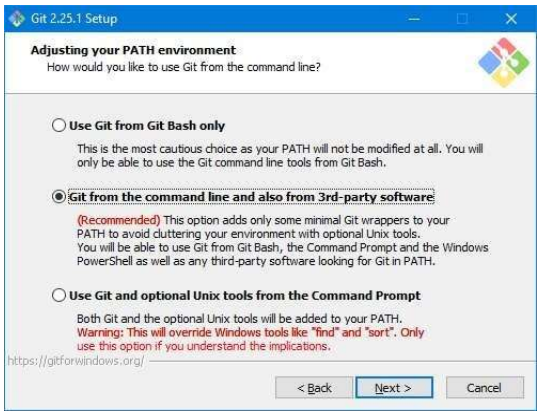
Step - 2: Choose installation directory and click next.



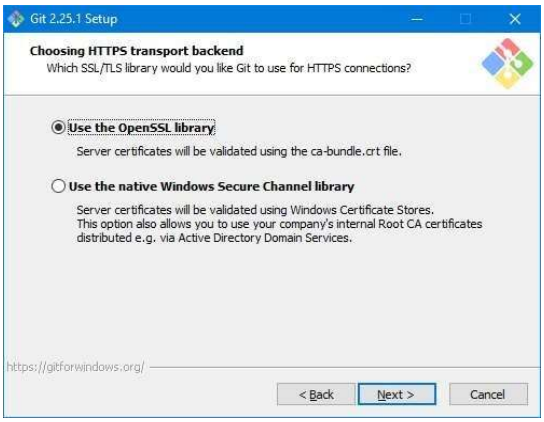
Step - 3: Select < Use Vim (the ubiquitous text editor) as Git's default editor >



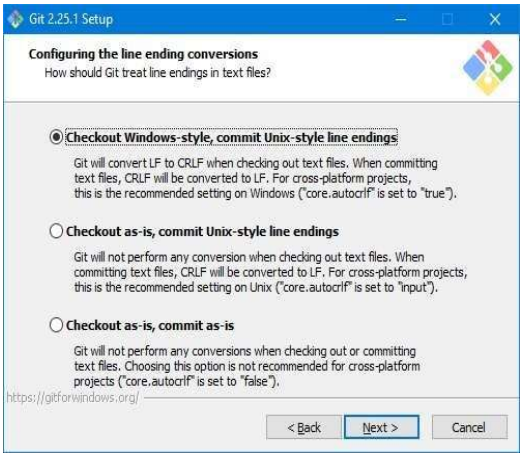
Step - 4: Select < Git from the command line and also from 3rd - party software >



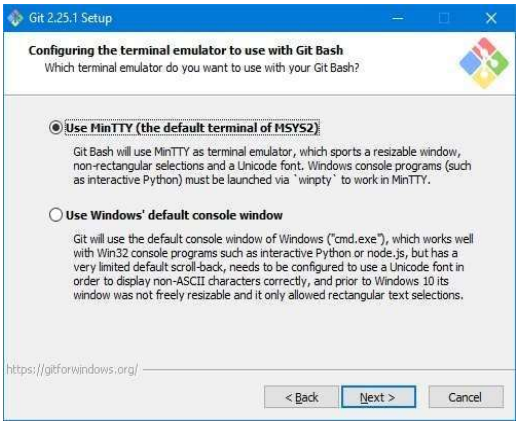
Step - 5: Select < Use the OpenSSL library >



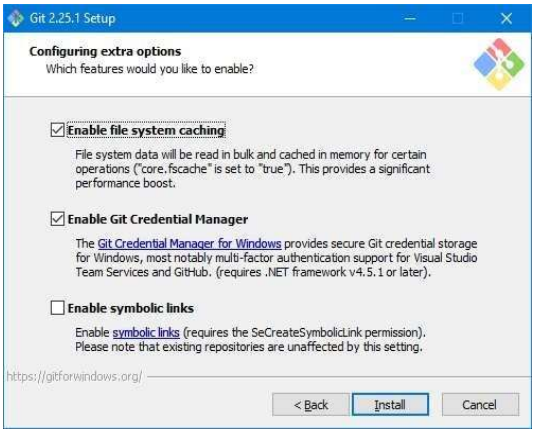
Step - 6: Select < Checkout Windows-style, commit Unix-style line endings >



Step - 7: Select < Use MinTTY (the default terminal of MSYS2) >

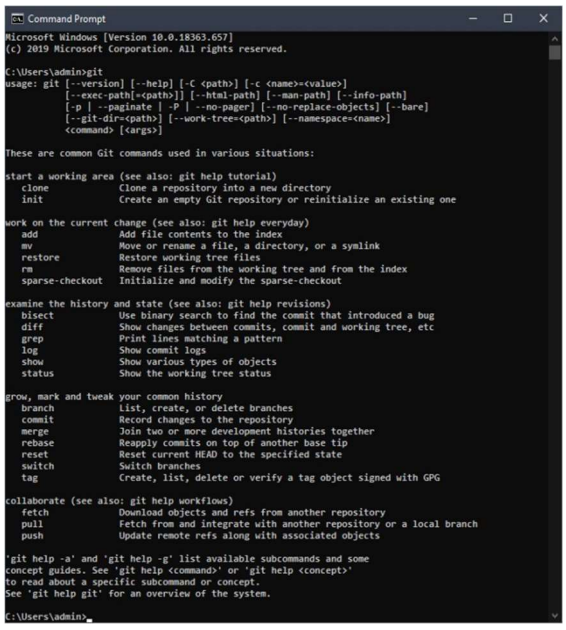


Step - 8: Check < Enable file system caching > and < Enable Git Credential Manager >



Step - 9: Installation process is done.

Step - 10: To verify installation, run the following command in cmd**> git**

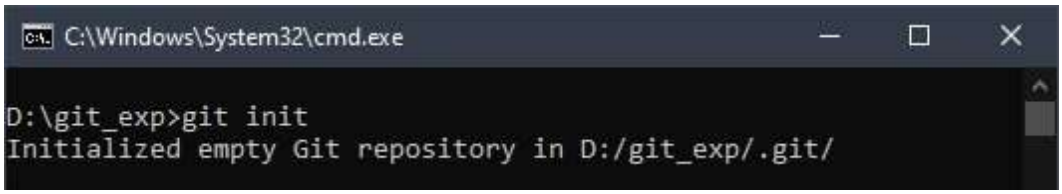


Basic Git commands:

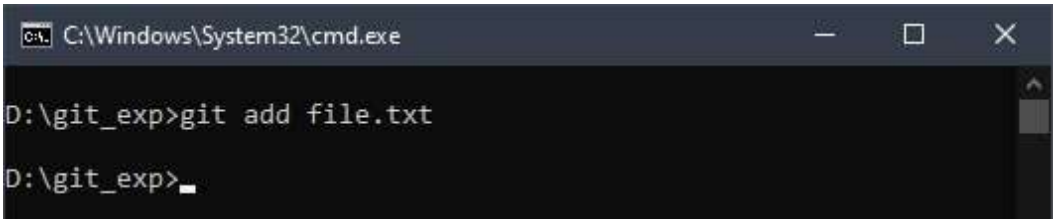
COMMANDS	FUNCTION
git init	Initialize git repository for the directory.
git add <file>	Creates or adds your file to local git repository.
git diff	Difference between previous versions and current version of the file.
git status	Git status for the directory i.e. displays the untracked files.
git commit –a –m "<commit name>"	To create local commit of the repository.
git log	Displays the commit logs of the local repository.
git checkout <file>	To restore the data of the file from previous commit.
git clone <URL>	To clone the github repository in local machine.
git push	To push the local commits to the github repository.
git remote add <name><URL>	To push the local commit to a remote group.

OUTPUT:

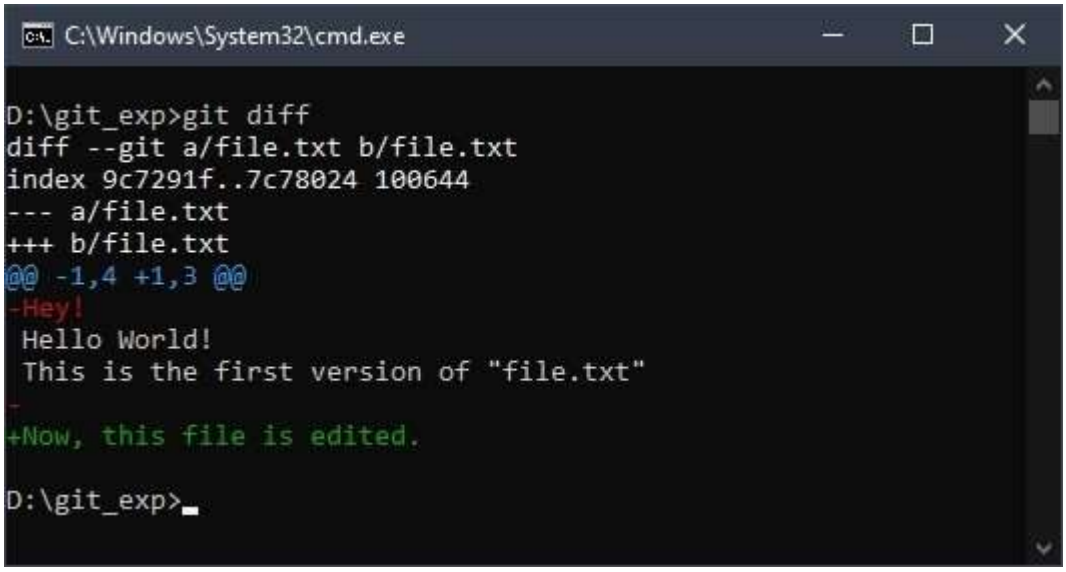
git init



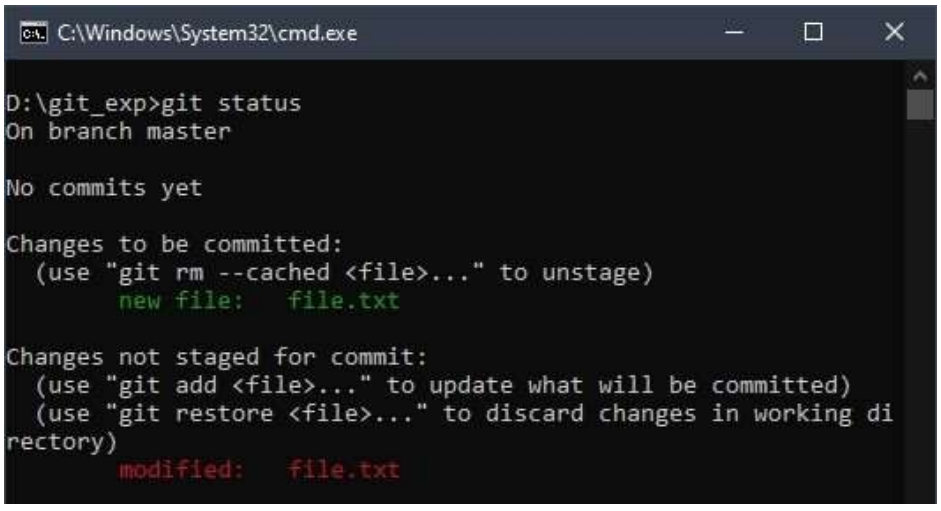
git add <file>



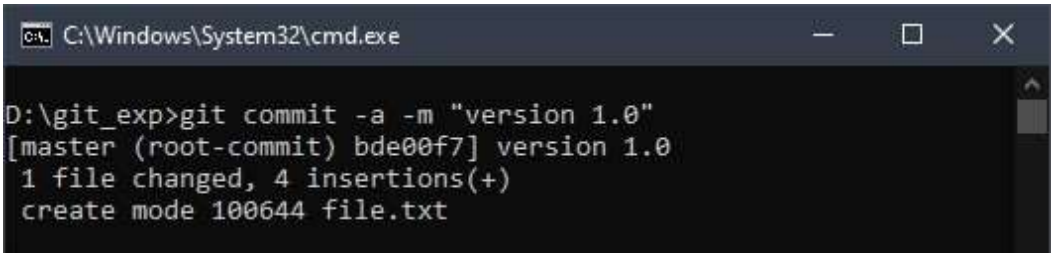
git diff



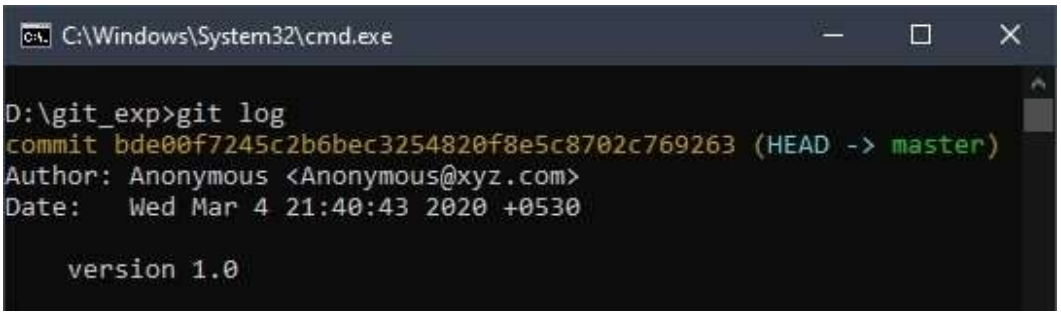
git status



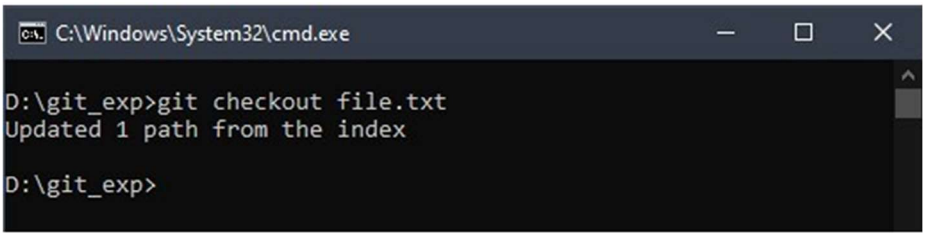
git commit –a –m "<commit name>"



git log



git checkout



RESULT:

Thus, the Git was configured and basic operations of version control system are executed successfully.

Ex.No. 10	SETTING UP OF GITLABORATORY AS A REPOSITORY SERVICE

AIM:

To set up GitLab as a repository service by creating and forking repositories.

REQUIREMENTS:

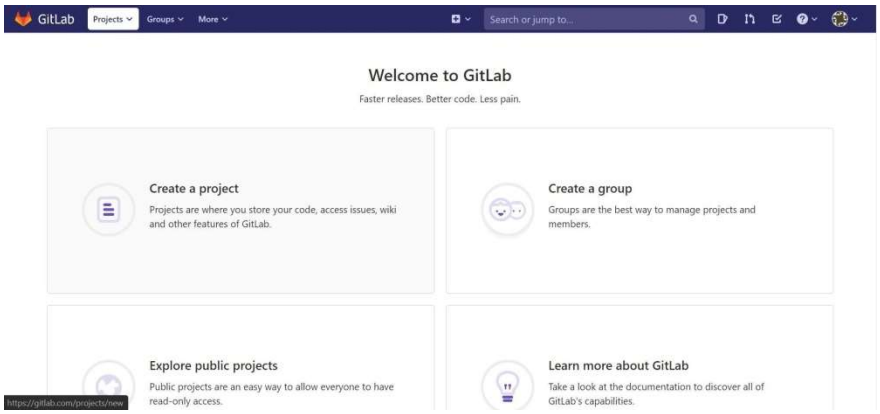
1. GitLab Account
2. Command Prompt (CLI)

PROCEDURE:

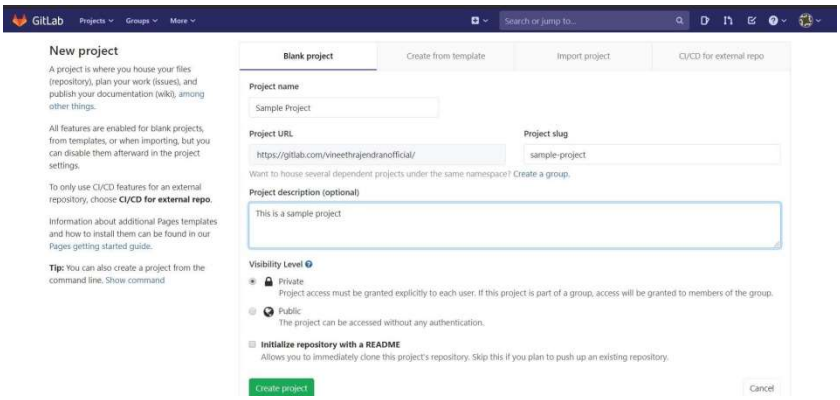
Creating a Project:

To create a new repository, all you need to do is create a new project or fork an existing project.

Step 1 − To create new project, login to your GitLab account and click on the ***Create a project*** button in the dashboard

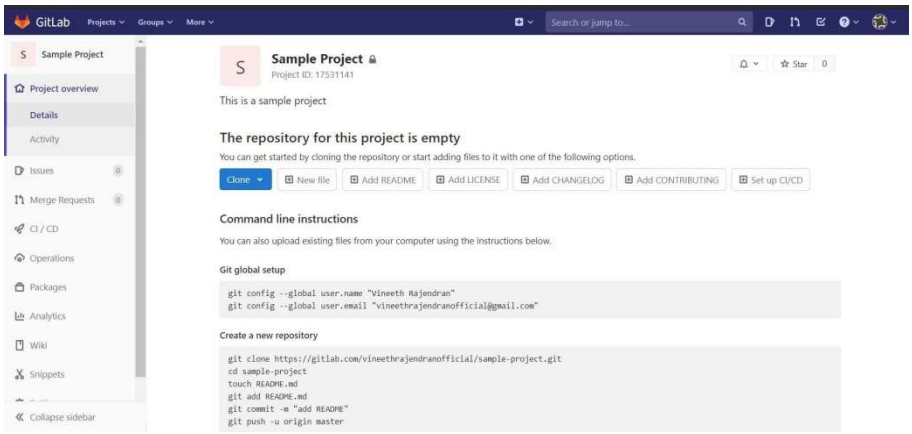


Step 2 − It will open the New project screen as shown below in the image

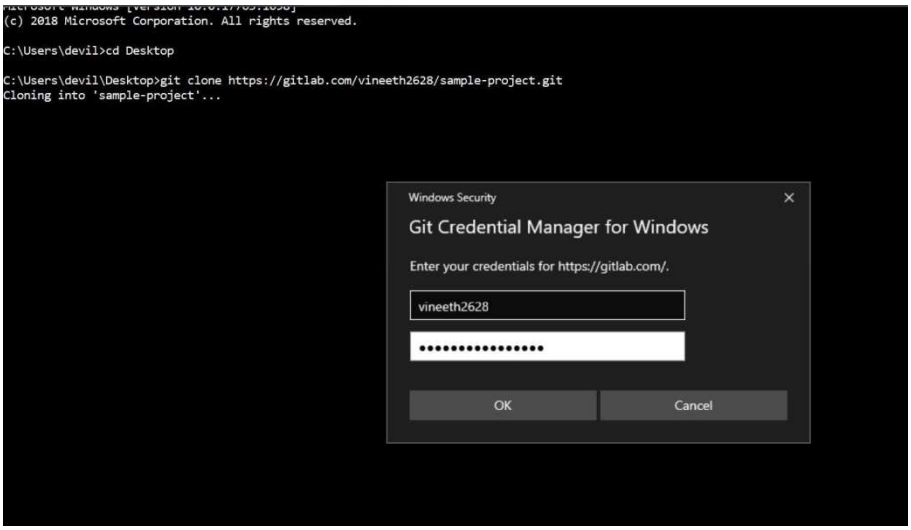


Enter the project name, description for the project, visibility level (accessing the project's visibility in publicly or internally) and click on the ***Create project*** button.

Step 3 − Next it will create a new project (here given the project name as sample-project) with successful message as shown below



Step 4 − You can clone the repository to your local system by using the ***git-clone*** command



The clone command makes a copy of repository into a new directory called ***sample-project.***

Step 5 − Now go to your newly created directory and type the below command

```
C:\Users\devil\Desktop\sample-project>echo.> README.md
```

The above command creates a ***README.md*** file in which you can put the information about your folder. When a README or index file is present in a repository, its contents will be automatically pre-rendered by GitLab without opening it.

Step 6 − Add the ***README.md*** file to your created directory by using the below command

```
C:\Users\devil\Desktop\sample-project>git add README.md
```

Step 7 − Now store the changes to the repository along with the log message as shown below

```
C:\Users\devil\Desktop\sample-project>git commit -m "add README"
[master (root-commit) d83eea3] add README
 1 file changed, 1 insertion(+)
 create mode 100644 README.md
```

The flag ***-m*** is used for adding a message on the commit.

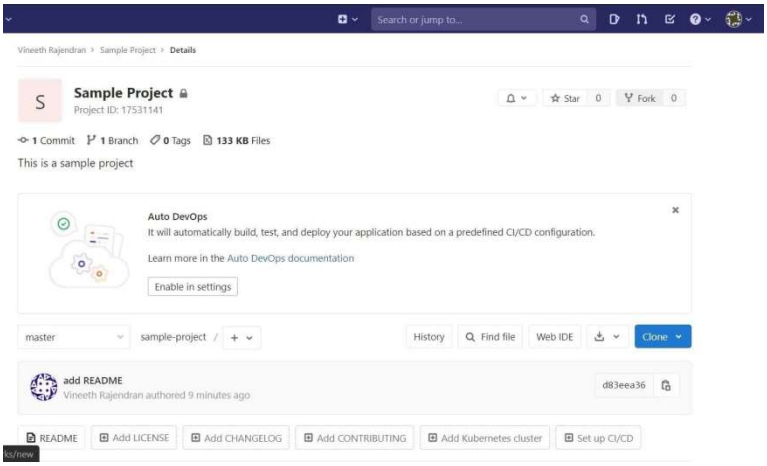
Step 8 − Push the commits to remote repository which are made on the local branch. The below image depicts the usage of above commands in pushing the commits to remote repository.

```
C:\Users\devil\Desktop\sample-project>git push -u origin master
Enumerating objects: 3, done.
Counting objects: 100% (3/3), done.
Writing objects: 100% (3/3), 233 bytes | 116.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://gitlab.com/vineeth2628/sample-project.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
```

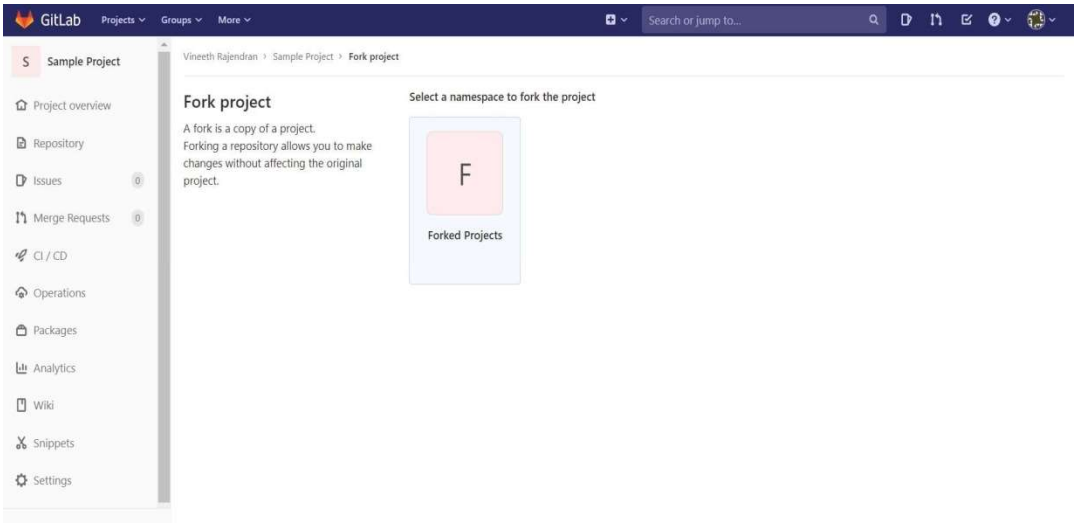
When you commit your changes, you are introducing those changes to your branch. Via command line, you can commit multiple times before pushing.

Forking a Project:

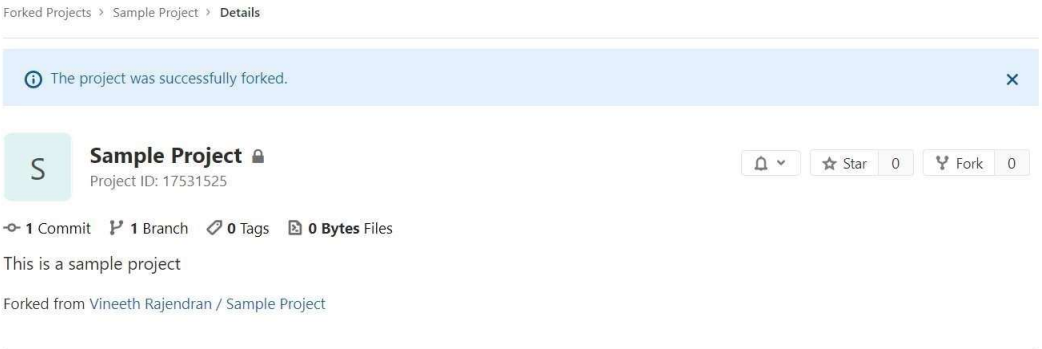
Step 1 − To fork a project, click on the ***Fork*** button as shown below



Step 2 − After forking the project, you need to add the forked project to a **fork group** by clicking on it.



Step 3 − It will display the success message after completion of forking the project process



Supported markup languages and extensions:

GitLab supports a number of markup languages (sometimes called lightweight markup languages) that you can use for the content of your files in a repository. They are mostly used for documentation purposes.

Just pick the right extension for your files and GitLab will render them according to the markup language.

Markup language	Extensions
Plain text	txt
Markdown	mdown, mkd, mkdn, md, markdown
reStructuredText	rst
AsciiDoc	adoc, ad, asciidoc
Textile	textile
rdoc	rdoc
Orgmode	org
creole	creole
Mediawiki	wiki, mediawiki

Project and repository size:

A project's size is reported on the project's **Details** page. The reported size is updated every 15 minutes at most, so may not reflect recent activity. The displayed files size includes repository files, artifacts, and LFS. The project size may differ slightly from one instance to another due to compression, housekeeping, and other factors.

Repository size limit may be set by admins. GitLab.com's repository size limit is set by GitLab.

RESULT:

Thus, Git Laboratory was successfully set as a repository service.

CONTENT BEYOND SYLLABUS

Ex. No. 11	CONTENT BEYOND SYLLABUS STUDY ON KUBERNETES

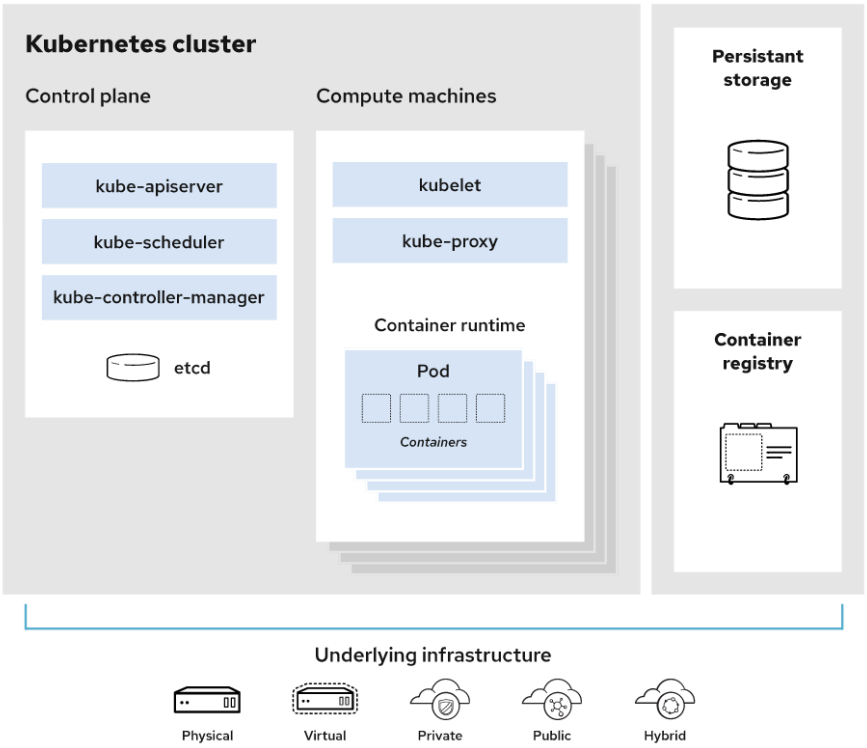
AIM:

To study about installation and working of Kubernetes.

KUBERNETES:

Kubernetes, often abbreviated as K8s, is an open-source container orchestration platform used for automating, managing, and scaling containerized applications. Developed by Google, it simplifies the deployment, scaling, and maintenance of containerized applications, making it easier to manage container workloads across a cluster of machines. Key features include automated scaling, load balancing, self-healing, and declarative configuration using YAML files. Kubernetes has become the de facto standard for container orchestration and is widely used in cloud-native application development and container management.

How does Kubernetes work?



A working Kubernetes deployment is called a cluster. You can visualize a Kubernetes cluster as two parts: the control plane and the compute machines, or nodes.

Each node is its own environment, and could be either a physical or virtual machine. Each node runs pods, which are made up of containers.

The control plane is responsible for maintaining the desired state of the cluster, such as which applications are running and which container images they use. Compute machines actually run the applications and workloads.

Kubernetes runs on top of an operating system and interacts with pods of containers running on the nodes.

The Kubernetes control plane takes the commands from an administrator (or DevOps team) and relays those instructions to the compute machines.

This handoff works with a multitude of services to automatically decide which node is best suited for the task. It then allocates resources and assigns the pods in that node to fulfill the requested work.

The desired state of a Kubernetes cluster defines which applications or other workloads should be running, along with which images they use, which resources should be made available to them, and other such configuration details.

From an infrastructure point of view, there is little change to how you manage containers. Your control over containers just happens at a higher level, giving you better control without the need to micromanage each separate container or node.

Your work involves configuring Kubernetes and defining nodes, pods, and the containers within them. Kubernetes handles orchestrating the containers.

Where you run Kubernetes is up to you. This can be on bare metal servers, virtual machines, public cloud providers, private clouds, and hybrid cloud environments. One of Kubernetes' key advantages is it works on many different kinds of infrastructure.

PROCEDURE TO INSTALL:

Prerequisites:

- Configure IP Tables.
- Disable SWAP.
- Install Docker and configure.
- Install Kubeadm-Kubelet and Kubectl.
- Create Default Audit Policy.
- Install NFS Client Drivers

Step 1: Disable swap on all nodes by removing the line with swap in /etc/fstab

Step 2: Install Docker runtime on all nodes by running the following commands:

```
sudo apt-get update
sudo apt-get install \
apt-transport-https \
ca-certificates \
curl \
gnupg \
lsb-release
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/docker-archive-keyring.gpg
echo \
"deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg]
https://download.docker.com/linux/ubuntu \  $(lsb_release -cs) stable" | sudo tee
/etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update
sudo apt-get install docker-ce docker-ce-cli containerd.io
```

Step 3: Configure cgroup on all nodes by running the following command:

```
sudo vim /etc/default/grub
```

Add cgroup_enable=cpuset cgroup_enable=memory cgroup_memory=1 to the GRUB_CMDLINE_LINUX_DEFAULT line, then run sudo update-grub and reboot the machine

Step 4: Install kubeadm, kubelet, and kubectl on all nodes by running the following commands:

```
sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg
https://packages.cloud.google.com/apt/doc/apt-key.gpg
echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg]
https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee
/etc/apt/sources.list.d/kubernetes.list
```

sudo apt-get update
sudo apt-get install -y kubelet kubeadm kubectl
sudo apt-mark hold kubelet kubeadm kubectl

Step 5: Initialize the master node by running the following command:
sudo kubeadm init --pod-network-cidr=192.168.0.0/16

This will create a new Kubernetes cluster with the specified pod network CIDR

Step 6: Set up the kubectl configuration by running the following commands:
mkdir -p $HOME/.kube
sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
sudo chown $(id -u):$(id -g) $HOME/.kube/config

This will allow you to use kubectl to manage your Kubernetes cluster

Step 7: Install a pod network add-on to enable communication between pods in the cluster. For example, to install Calico, run the following command:
kubectl apply -f https://docs.projectcalico.org/manifests/calico.yaml

This will install Calico and configure it to use the pod network CIDR specified in step 5

KUBERNETES CASE STUDY ON ADIDAS:

1. Challenge Faced by Adidas:

In recent years, the adidas team was happy with its software choices from a technology perspective—but accessing all of the tools was a problem. For instance, "just to get a developer VM, you had to send a request form, give the purpose, give the title of the project, who's responsible, give the internal cost center a call so that they can do recharges," says Daniel Eichten, Senior Director of Platform Engineering. "The best case is you got your machine in half an hour. Worst case is half a week or sometimes even a week."

2. Solution to overcome this challenge:

To improve the process, "we started from the developer point of view," and looked for ways to shorten the time it took to get a project up and running and into the adidas infrastructure, says Senior Director of Platform Engineering Fernando Cornago. They found the solution with containerization, agile development, continuous delivery, and a cloud native platform that includes Kubernetes and Prometheus.

3. Impact after using Kubernetes:

The implementation of Kubernetes at Adidas had a significant impact. Within six months, 100% of their e-commerce site ran on Kubernetes, leading to a 50% reduction in load times. Release frequency increased from every 4-6 weeks to 3-4 times a day. With 4,000 pods, 200 nodes, and 80,000 monthly builds, they now run 40% of their most critical systems on Kubernetes. Adoption among engineers was rapid, with teams onboarding and learning the platform, leading to a successful and culture-aligned transition.