SEMESTER - VI

20ITPL602 SDG NO. 4	CLOUD COMPUTING AND VIRTUALIZATION LABORATORY	L	T	P	C
		0	0	3	1.5

OBJECTIVES:

- To develop Web Applications in Cloud
- To learn the design and development process involved in creating a Cloud Based Application
- To understand the installation of Cloud Simulation tools and Cloud Setup tools
- To learn to implement and use Parallel programming using Hadoop

LIST OF EXPERIMENTS:

1. Install Virtual Box/VMware Workstation with different flavors of Linux or Windows OS on top of Windows 7 or 8.
2. Install a C compiler in the virtual machine created using virtual box and executes simple programs.
3. Install Google App Engine. Create hello world app and other simple web applications using Python/Java.
4. Use GAE launcher to launch the Web Applications.
5. Simulate a cloud scenario using CloudSim and run a scheduling algorithm that is not present in CloudSim.
6. Find a procedure to transfer the files from one Virtual Machine to another Virtual Machine.
7. Find a procedure to launch Virtual Machine using try stack (Online Openstack Demo Version)
8. Install Hadoop single node cluster and run simple applications like wordcount.

TOTAL: 45 PERIODS

LAB REQUIREMENTS:

1. Virtual box
2. VMware Workstation
3. Openstack, Hadoop
4. Cloudism
5. GAE launcher

OUTCOMES :

On completion of this laboratory course, the student should be able to

1. Configure various virtualization tools such as Virtual Box, VMware.
2. Design and deploy a web application in a PAAS environment.
3. Learn how to simulate a cloud environment to implement new schedulers.
4. Manipulate large data sets in a parallel environment.
5. Design and Develop Cloud based applications.
6. Learn to use Hadoop Map Reduce software and solve word count problem

CO- PO, PSO MAPPING :

	PO1	PO2	PO3	PO4	PO5	PO6	PO7	PO8	PO9	PO10	PO11	PO12
CO1	3	3	3	3	3							3
CO2	2	2	3	3	3							3
CO3	3	3	3	3	3	3			3	3		3
CO4	2	2	2	3	3	2	2	1	1	1	1	1
CO5	3	3	3	2	2	1	1			1	1	1
CO6	3	3	3	2	2	1				1	1	2

Ex No 1	**Install Virtual Box/VMware Workstation with different flavors of Linux or Windows OS on top of Windows 7 or 8.**
Date	

Virtual Machine in Cloud Computing

A VM is **a virtualized instance of a computer that can perform almost all of the same functions as a computer, including running applications and operating systems**. Virtual machines run on a physical machine and access computing resources from software called a hypervisor.

What is a virtual machine?

A Virtual Machine (VM) is a compute resource that uses software instead of a physical computer to run programs and deploy apps. One or more virtual "guest" machines run on a physical "host" machine. Each virtual machine runs its own operating system and functions separately from the other VMs, even when they are all running on the same host. This means that, for example, a virtual MacOS virtual machine can run on a physical PC.

Virtual machine technology is used for many use cases across on-premises and cloud environments. More recently, public cloud services are using virtual machines to provide virtual application resources to multiple users at once, for even more cost efficient and flexible compute.

What are virtual machines used for?

Virtual machines (VMs) allow a business to run an operating system that behaves like a completely separate computer in an app window on a desktop. VMs may be deployed to accommodate different levels of processing power needs, to run software that requires a different operating system, or to test applications in a safe, sandboxed environment.

Virtual machines have historically been used for server virtualization, which enables IT teams to consolidate their computing resources and improve efficiency. Additionally, virtual machines can perform specific tasks considered too risky to carry out in a host environment, such as accessing virus-infected data or testing operating systems. Since the virtual machine is separated from the rest of the system, the software inside the virtual machine cannot tamper with the host computer.

How do virtual machines work?

The virtual machine runs as a process in an application window, similar to any other application, on the operating system of the physical machine. Key files that make up a virtual machine include a log file, NVRAM setting file, virtual disk file and configuration file.

Advantages of virtual machines

- Virtual machines are easy to manage and maintain, and they offer several advantages over physical machines:
- VMs can run multiple operating system environments on a single physical computer, saving physical space, time and management costs.
- Virtual machines support legacy applications, reducing the cost of migrating to a new operating system. For example, a Linux virtual machine running a distribution of Linux as the guest operating system can exist on a host server that is running a non-Linux operating system, such as Windows.
- VMs can also provide integrated disaster recovery and application provisioning options.

Disadvantages of virtual machines

- While virtual machines have several advantages over physical machines, there are also some potential disadvantages:
- Running multiple virtual machines on one physical machine can result in unstable performance if infrastructure requirements are not met.
- Virtual machines are less efficient and run slower than a full physical computer. Most enterprises use a combination of physical and virtual infrastructure to balance the corresponding advantages and disadvantages.

What are 5 types of virtualization?

All the components of a traditional data center or IT infrastructure can be virtualized today, with various specific types of virtualization:

1. **Hardware virtualization:** When virtualizing hardware, virtual versions of computers and operating systems (VMs) are created and consolidated into a single, primary, physical server. A hypervisor communicates directly with a physical server's disk space and CPU to manage the VMs. Hardware virtualization, which is also known as server virtualization, allows hardware resources to be utilized more efficiently and for one machine to simultaneously run different operating systems.
2. **Software virtualization**: Software virtualization creates a computer system complete with hardware that allows one or more guest operating systems to run on a physical host machine. For example, Android OS can run on a host machine that is natively using a Microsoft Windows OS, utilizing the same hardware as the host machine does. Additionally, applications can be virtualized and delivered from a server to an end user's device, such as a laptop or smartphone. This allows employees to access centrally hosted applications when working remotely.

3. **Storage virtualization:** Storage can be virtualized by consolidating multiple physical storage devices to appear as a single storage device. Benefits include increased performance and speed, load balancing and reduced costs. Storage virtualization also helps with disaster recovery planning, as virtual storage data can be duplicated and quickly transferred to another location, reducing downtime.
4. **Network virtualization:** Multiple sub-networks can be created on the same physical network by combining equipment into a single, software-based virtual network resource. Network virtualization also divides available bandwidth into multiple, independent channels, each of which can be assigned to servers and devices in real time. Advantages include increased reliability, network speed, security and better monitoring of data usage. Network virtualization can be a good choice for companies with a high volume of users who need access at all times.
5. **Desktop virtualization:** This common type of virtualization separates the desktop environment from the physical device and stores a desktop on a remote server, allowing users to access their desktops from anywhere on any device. In addition to easy accessibility, benefits of virtual desktops include better data security, cost savings on software licenses and updates, and ease of management.

VM WARE

- **VMware** is a virtualization and cloud computing software provider based in Palo Alto, Calif. Founded in 1998, VMware is a subsidiary of Dell Technologies.

- EMC Corporation originally acquired VMware in 2004; EMC was later acquired by Dell Technologies in 2016. VMware bases its virtualization technologies on its bare-metal hypervisor ESX/ESXi in x86 architecture.

- With VMware server virtualization, a hypervisor is installed on the physical server to allow for multiple **virtual machines (VMs)** to run on the same physical server.

- Each VM can run its own operating system (OS), which means multiple OSes can run on one physical server.

- All the VMs on the same physical server share resources, such as networking and RAM. In 2019, VMware added support to its hypervisor to run containerized workloads in a Kubernetes cluster in a similar way.

- These types of workloads can be managed by the infrastructure team in the same way as virtual machines and the DevOps teams can deploy containers as they were used to.

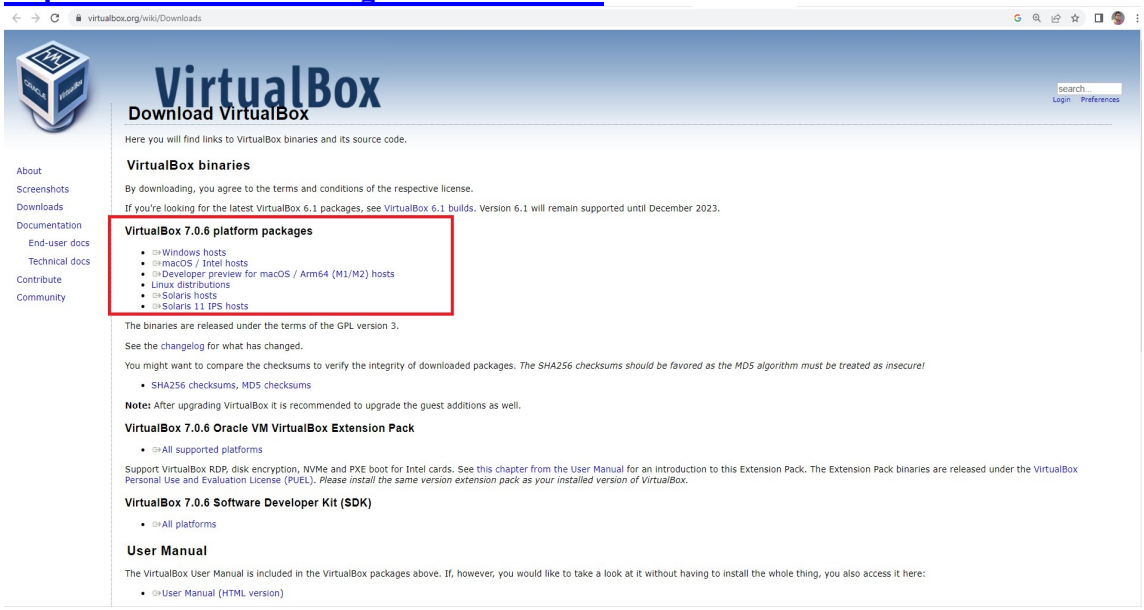
- Diane Greene, Scott Devine, Mendel Rosenblum, Edward Wang and Edouard Bugnion founded VMware, which launched its first product -- VMware Workstation -- in 1999. The company released its second product, VMware ESX in 2001.

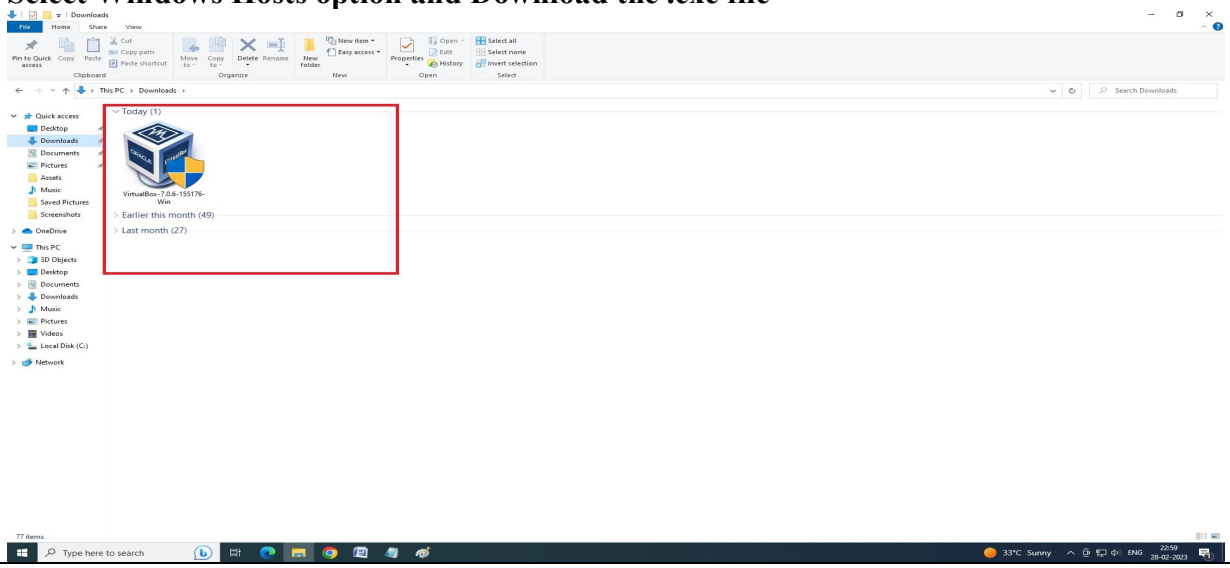
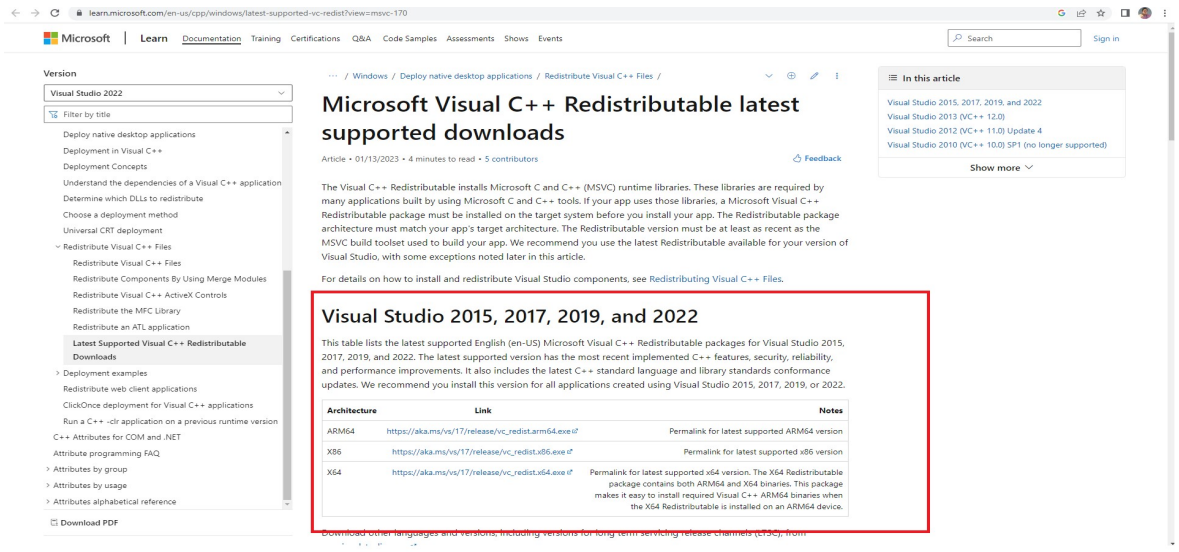
- VMware's current CEO is Patrick Gelsinger, appointed in 2012.

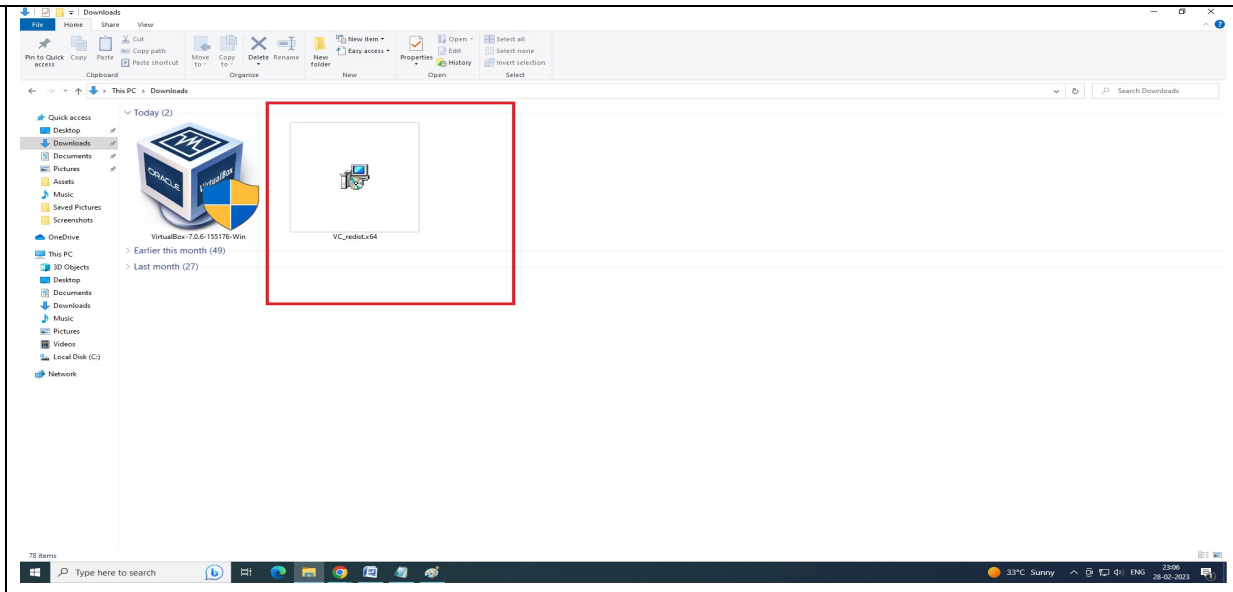
Oracle Virtual Box

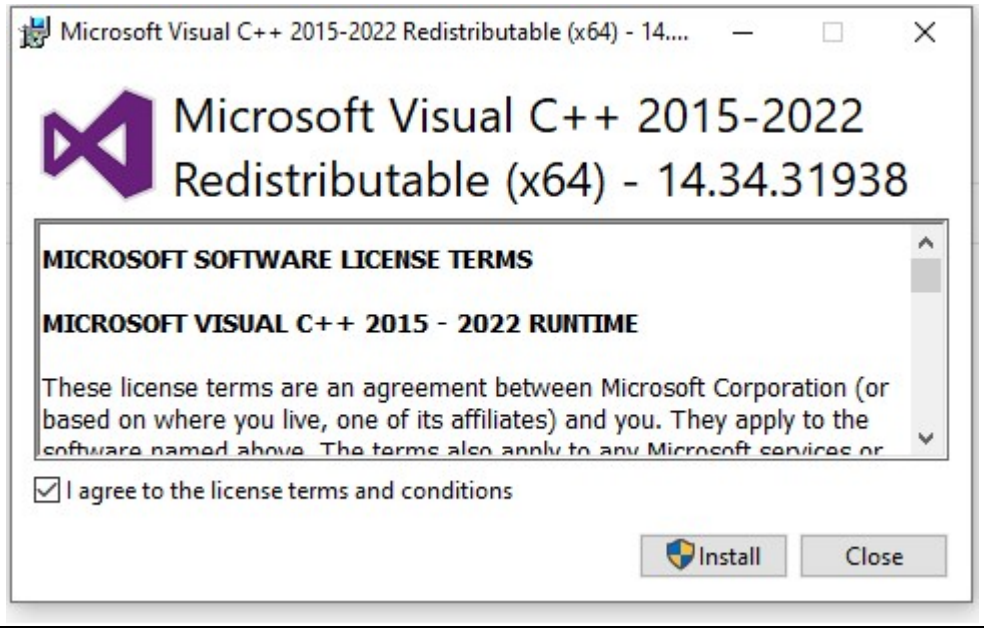
- Oracle VM VirtualBox (formerly Sun VirtualBox, Sun xVM VirtualBox and InnoTek VirtualBox) is a type-2 hypervisor for x86 virtualization developed by Oracle Corporation. VirtualBox was originally created by InnoTek Systemberatung GmbH, which was acquired by Sun Microsystems in 2008, which was in turn acquired by Oracle in 2010.
- VirtualBox may be installed on Microsoft Windows, macOS, Linux, Solaris and OpenSolaris. There are also ports to FreeBSD and Genode.[6] It supports the creation and management of guest virtual machines running Windows, Linux, BSD, OS/2, Solaris, Haiku, and OSx86, as well as limited virtualization of macOS guests on Apple hardware.
- For some guest operating systems, a "Guest Additions" package of device drivers and system applications is available, which typically improves performance, especially that of graphics, and allows changing the resolution of the guest OS automatically when the window of the virtual machine on the host OS is resized.
- Released under the terms of the GNU General Public License and, optionally, the CDDL for most files of the source distribution, VirtualBox is free and open-source software, though the Extension Pack is proprietary software. The License to VirtualBox was relicensed to GPLv3 with linking exceptions to the CDDL and other GPL-incompatible licenses.

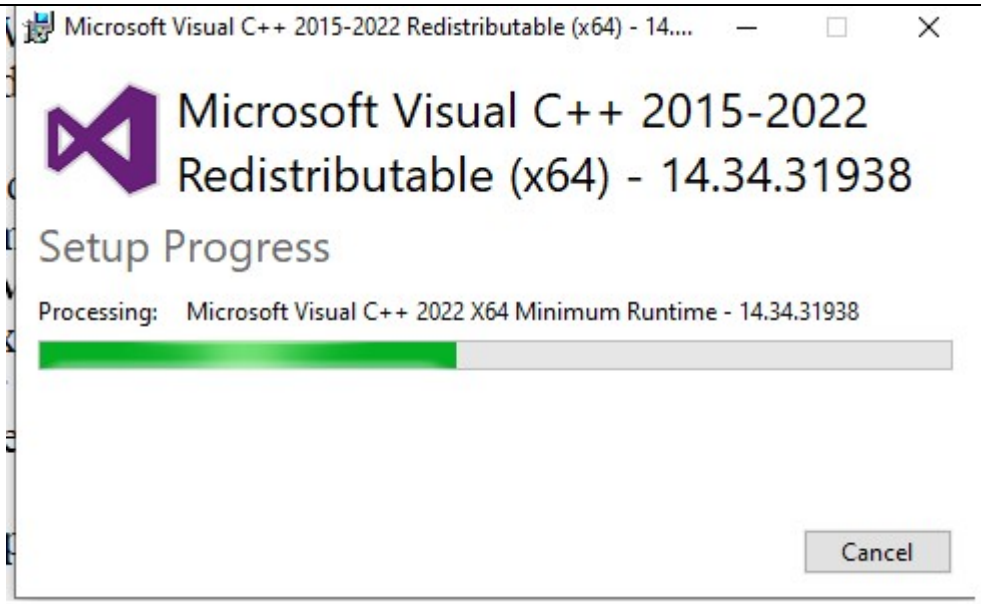
Oracle Virtual Box Installation in Windows Machine

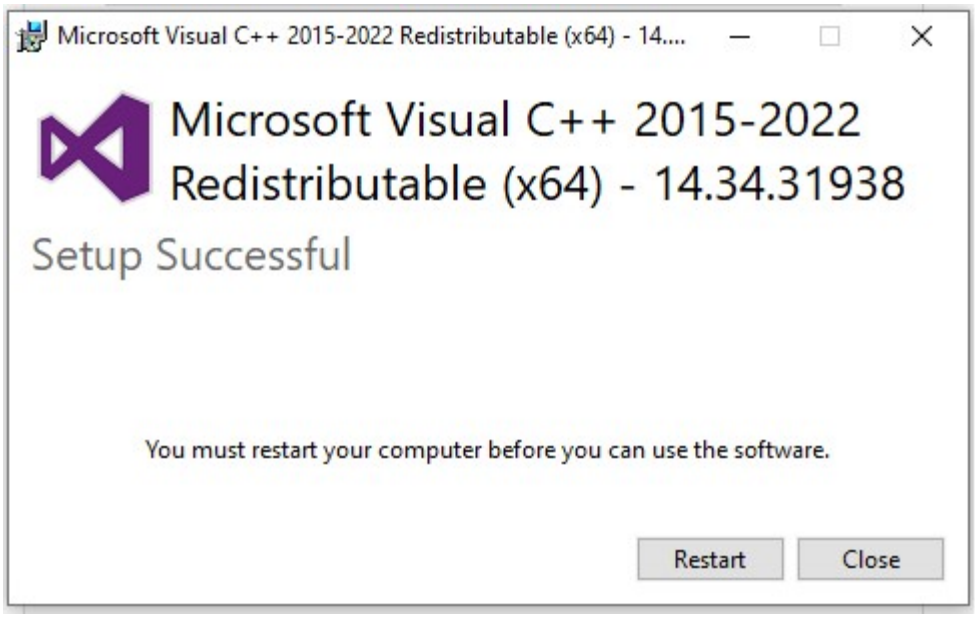
Step 1	**Download Virtual Box from the following Web Site** **https://www.virtualbox.org/wiki/Downloads** 

Step 2	**Select Windows Hosts option and Download the .exe file** 
Step 3	**The pre requisite of Virtual Box is the presence of Microsoft Visual C++ . If Microsoft Visual C++ is already present in the Machine, Skip Step 3 and Move on to Step 4** **If Visual C++ is not available download and install the same** **https://learn.microsoft.com/en-us/cpp/windows/latest-supported-vc-redist?view=msvc-170** 

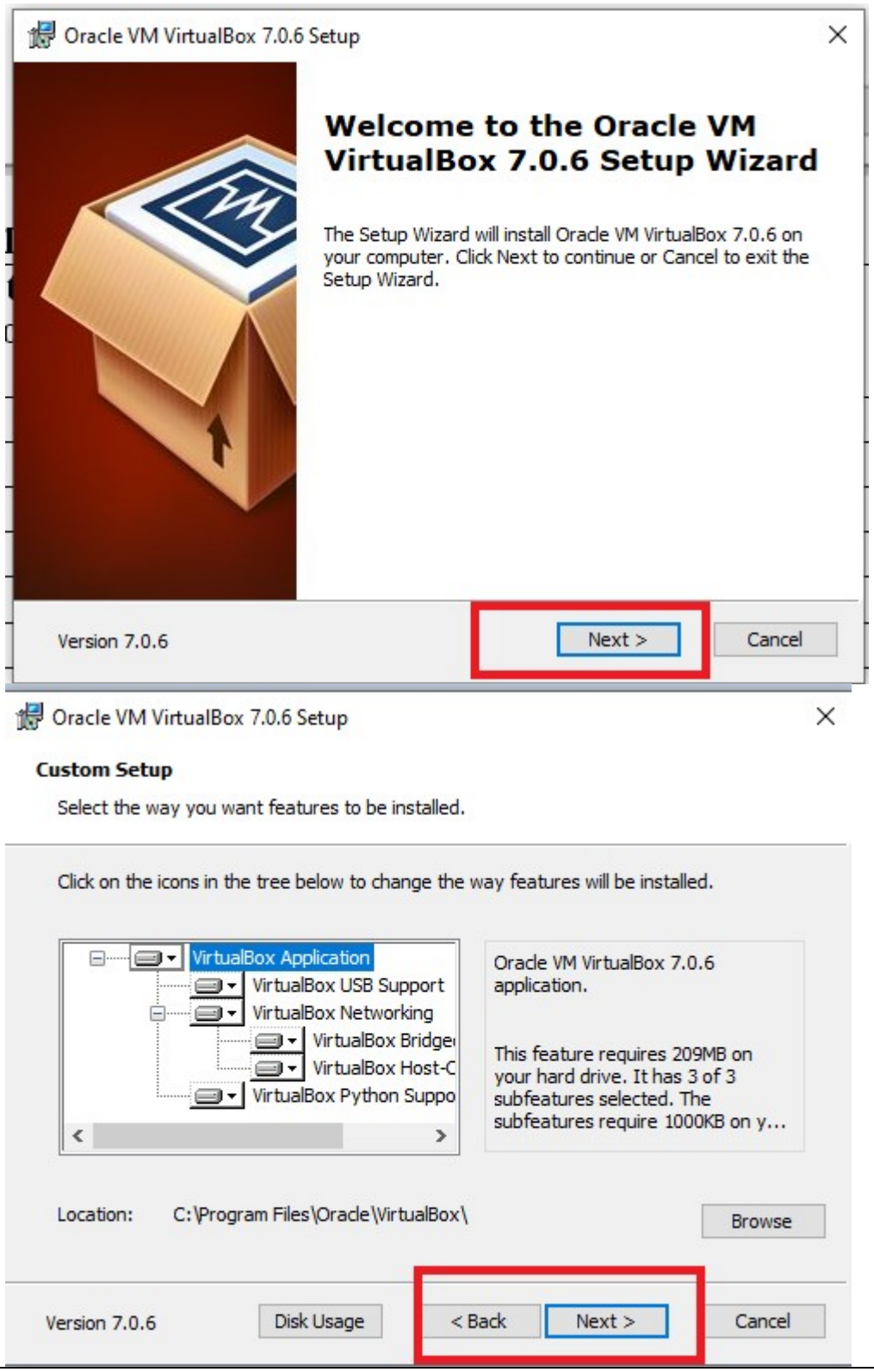

Downloads
File
Home
Share
View
This PC
Downloads
Search Downloads
Quick access
Desktop
Downloads
Documents
Pictures
Assets
Music
Saved Pictures
Screenshots
OneDrive
This PC
3D Objects
Desktop
Documents
Downloads
Music
Pictures
Videos
Local Disk (C:)
Network
Today (2)
VirtualBox-7.0.6-155176-Win
VC_redist.x64
Earlier this month (49)
Last month (27)
78 items
Type here to search
33°C Sunny
ENG
23:06
28-02-2023

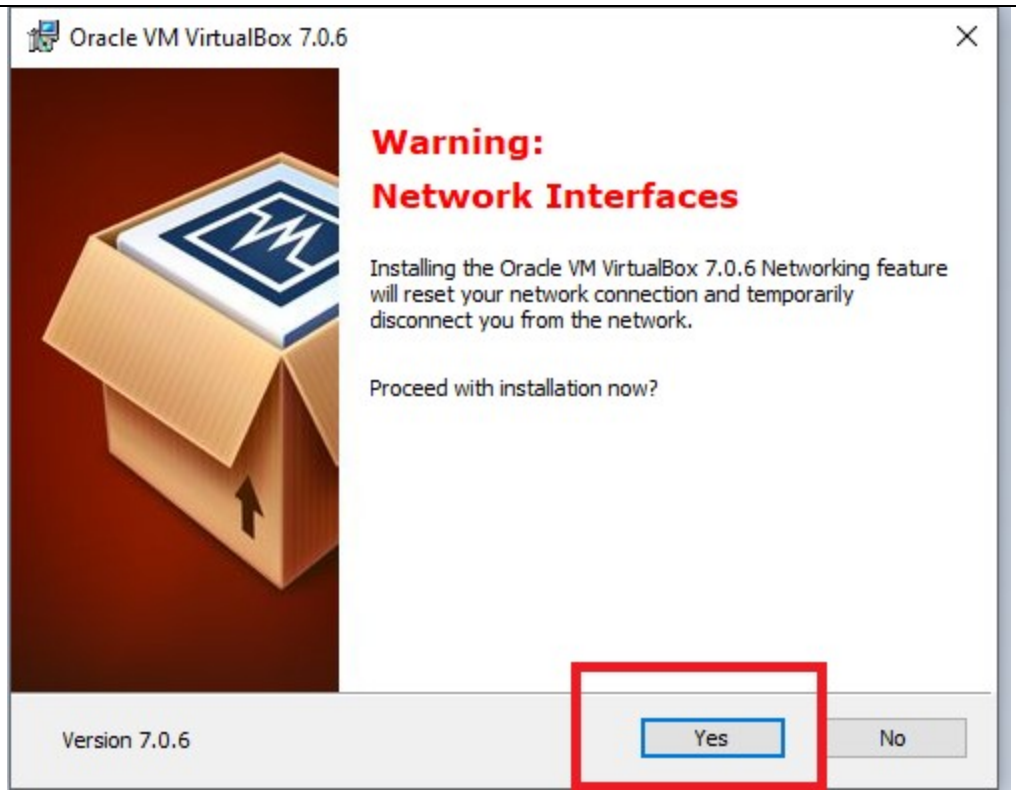

Microsoft Visual C++ 2015-2022 Redistributable (x64) - 14....
Microsoft Visual C++ 2015-2022 Redistributable (x64) - 14.34.31938
MICROSOFT SOFTWARE LICENSE TERMS
MICROSOFT VISUAL C++ 2015 - 2022 RUNTIME
These license terms are an agreement between Microsoft Corporation (or based on where you live, one of its affiliates) and you. They apply to the
I agree to the license terms and conditions
Install
Close

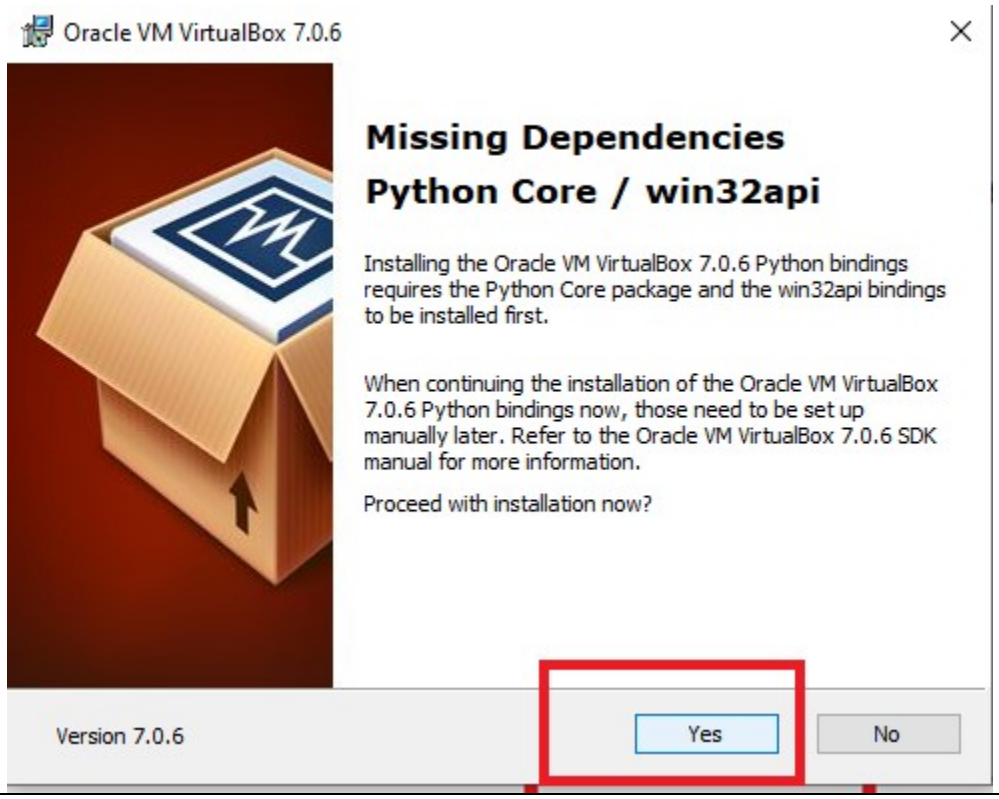


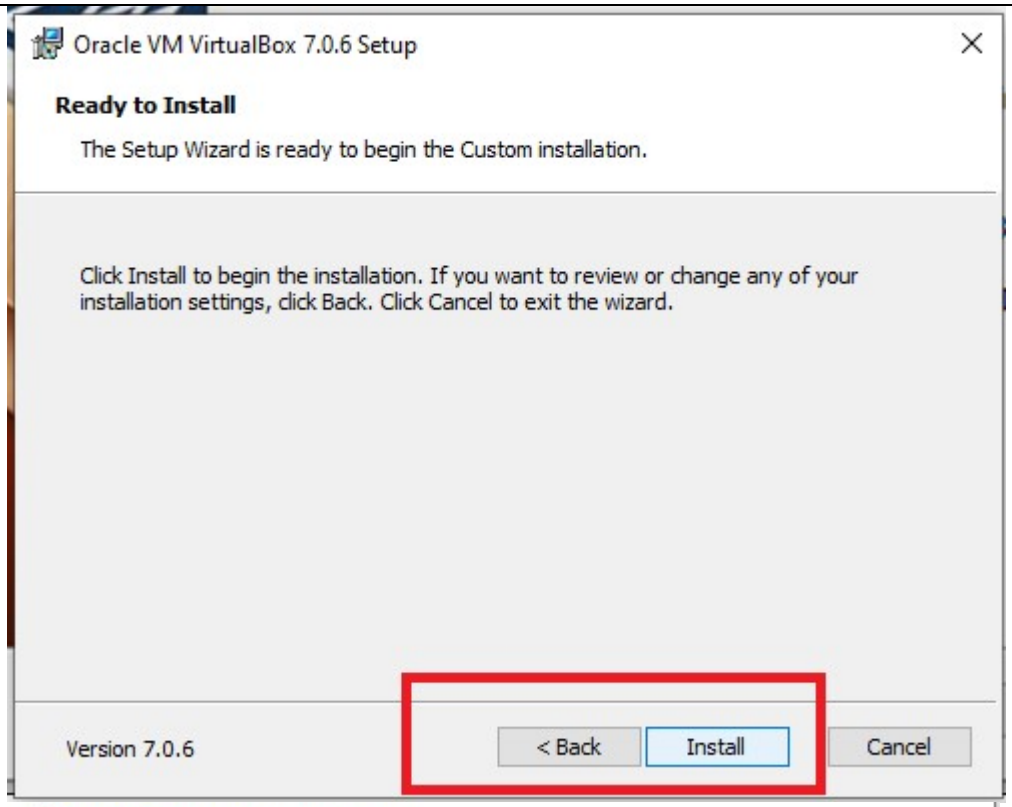


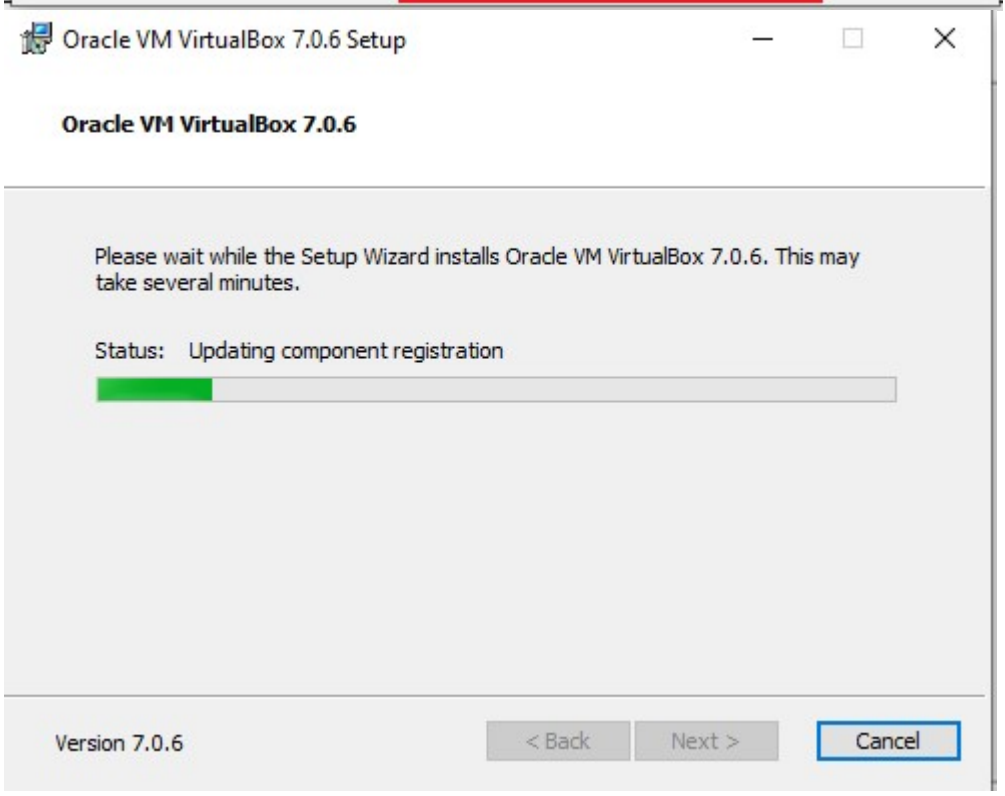
Restart the Machine

Step 4	**Installing Virtual Box** **Double click on the installer file to start with the installation** 

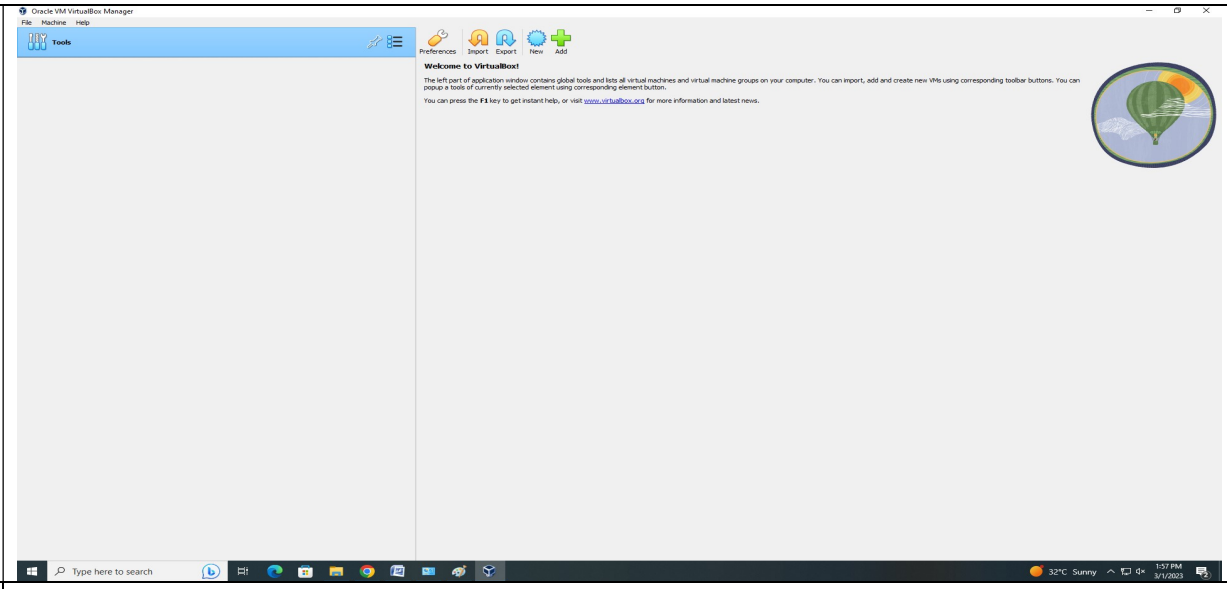

Oracle VM VirtualBox 7.0.6
Warning:
Network Interfaces
Installing the Oracle VM VirtualBox 7.0.6 Networking feature will reset your network connection and temporarily disconnect you from the network.
Proceed with installation now?
Version 7.0.6
Yes
No


Oracle VM VirtualBox 7.0.6
Missing Dependencies
Python Core / win32api
Installing the Oracle VM VirtualBox 7.0.6 Python bindings requires the Python Core package and the win32api bindings to be installed first.
When continuing the installation of the Oracle VM VirtualBox 7.0.6 Python bindings now, those need to be set up manually later. Refer to the Oracle VM VirtualBox 7.0.6 SDK manual for more information.
Proceed with installation now?
Version 7.0.6
Yes
No





Step 5	**After successful installation of Virtual Box, Open Virtual Box**



Step 6

Installing New Machine in the Virtual Box

- **ISO files** support the archiving of file folders such as system and installation files of operating systems, software programs, file folders, movies, videos, or even computer games. ISOs are popular among gamers into older classic games (e.g. SNES or Playstation 2) based on PC emulators.

- **An ISO file is an exact copy of an entire optical disk such as a CD, DVD, or Blu-ray archived into a single file**. This file, which is also sometimes referred to as an ISO image, is a smaller sized duplicate of large sets of data.

- In order to perform its function, the ISO file needs to be opened and assembled so the data can be viewed.

- Many people use ISO files to back up their optical disks or store their data in a more functional way.

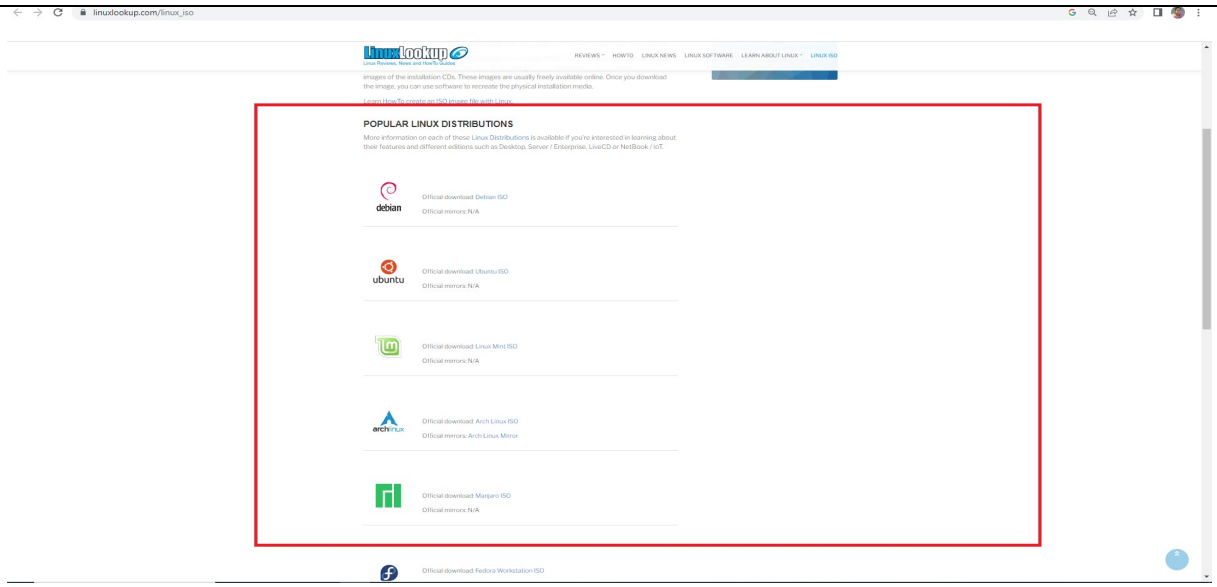
- Its actual function is to replicate an original optical disk and store it until it is needed to burn a new disk having the same data within it. You can use it to transfer a game, for instance, from an old disk to your laptop when there is no physical disk around.

- However, **ISO files are used for much greater purposes when it comes to operating systems**. They can be excellent vessels for large programs or operating systems themselves, since they are a single file that contains huge amounts of data.

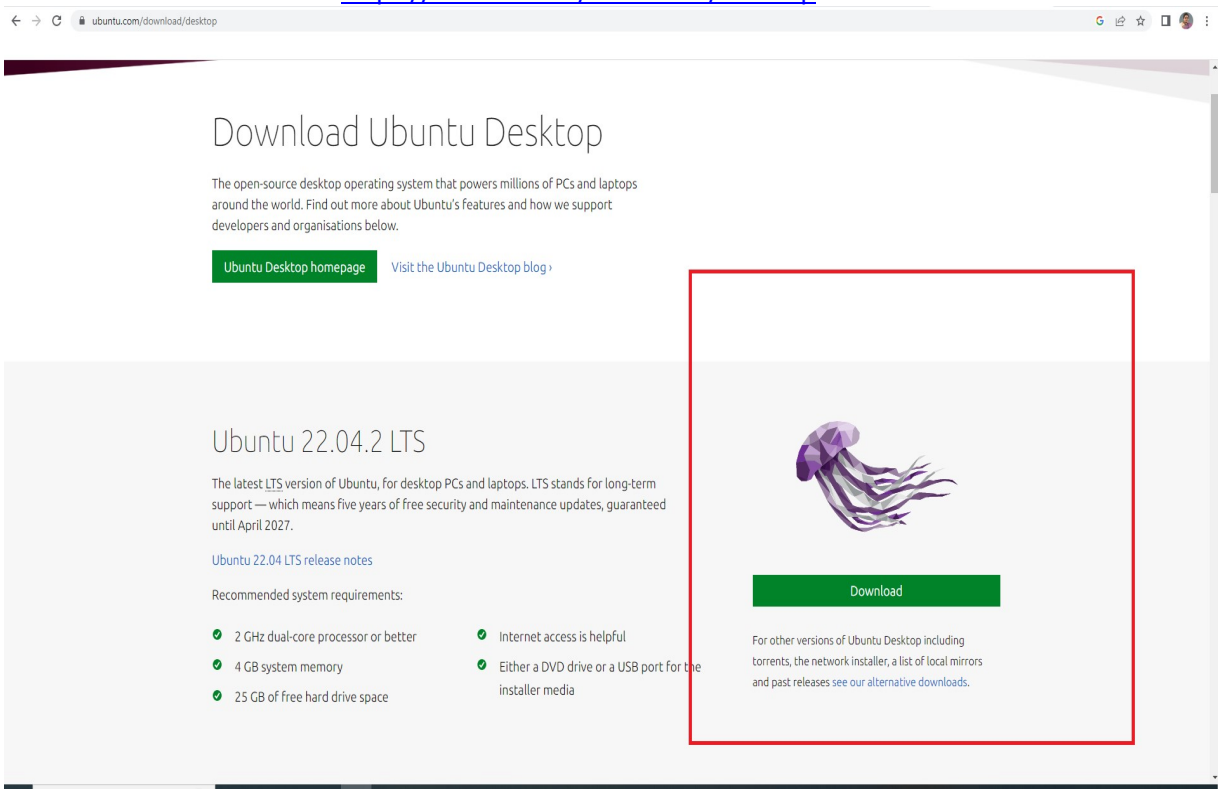
- By uploading them on a ISO file, these operating systems can be downloaded, mounted, or burned on an optical disk easily.

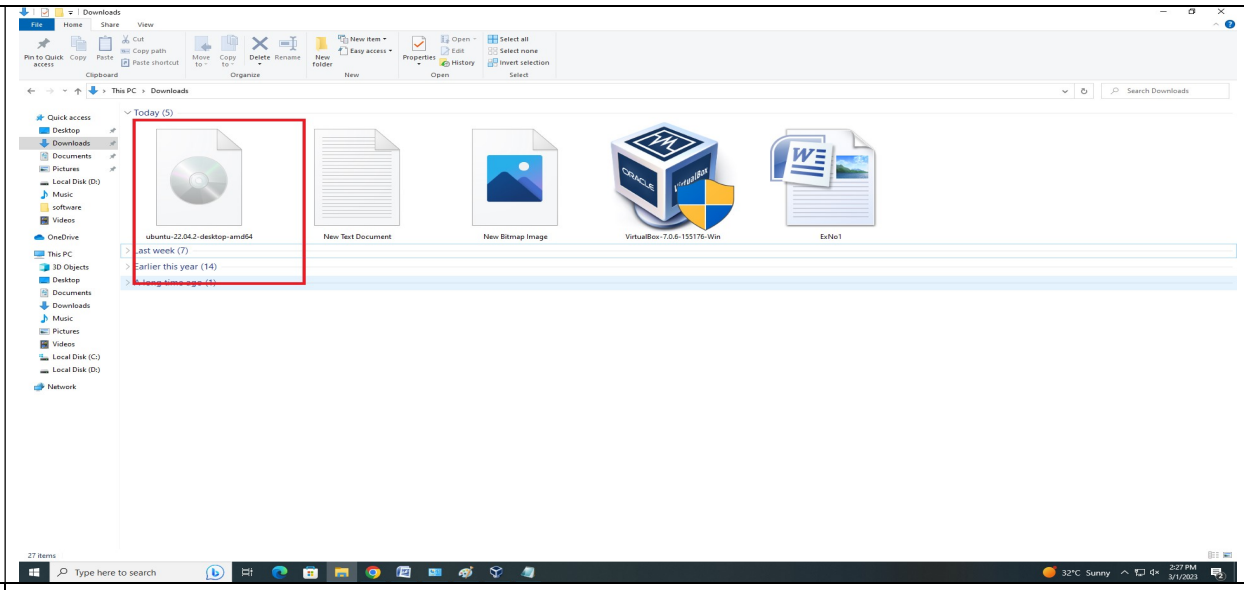
Open Source OS available in Market

Visit the https://www.linuxlookup.com/linux_iso site to know about the flavours of linux OS



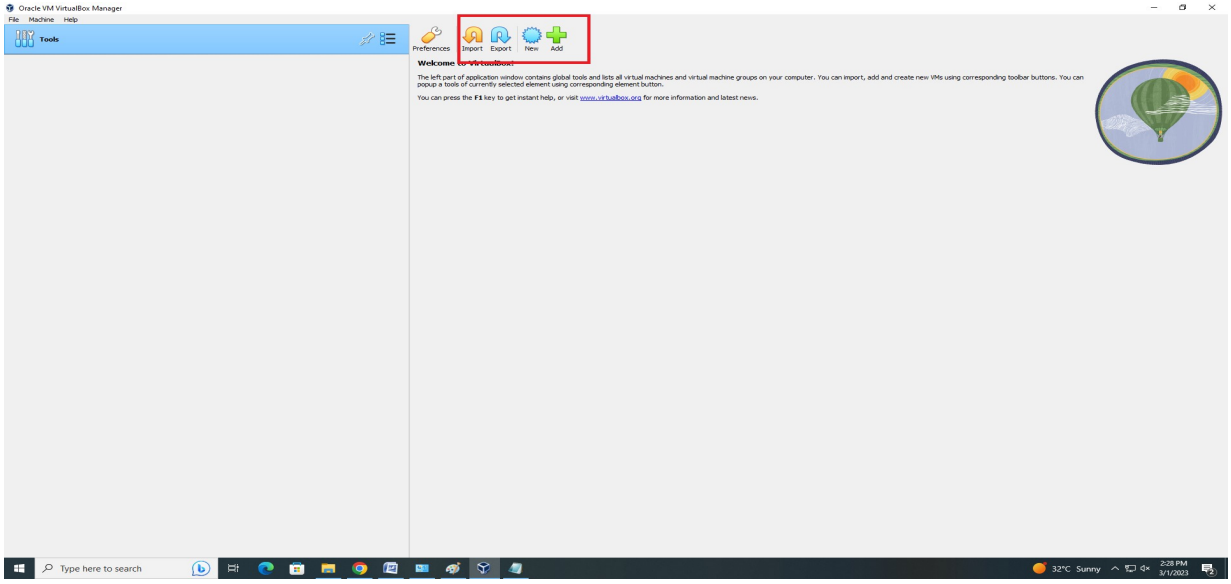
Download Ubuntu ISO from https://ubuntu.com/download/desktop

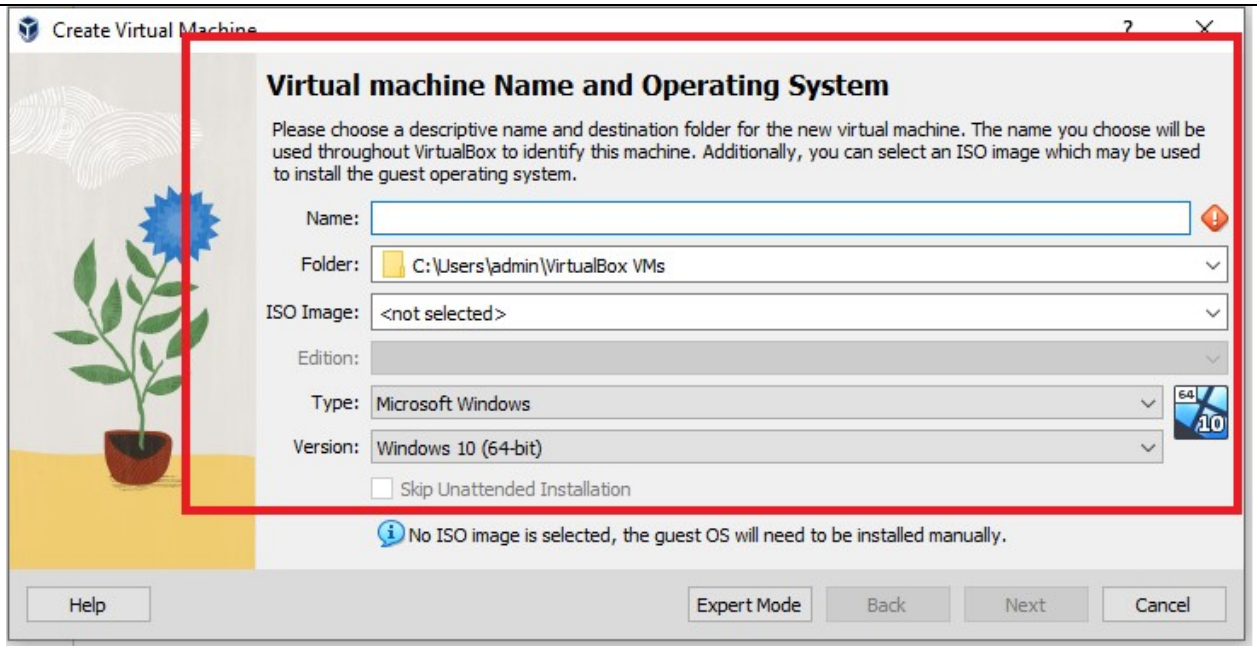


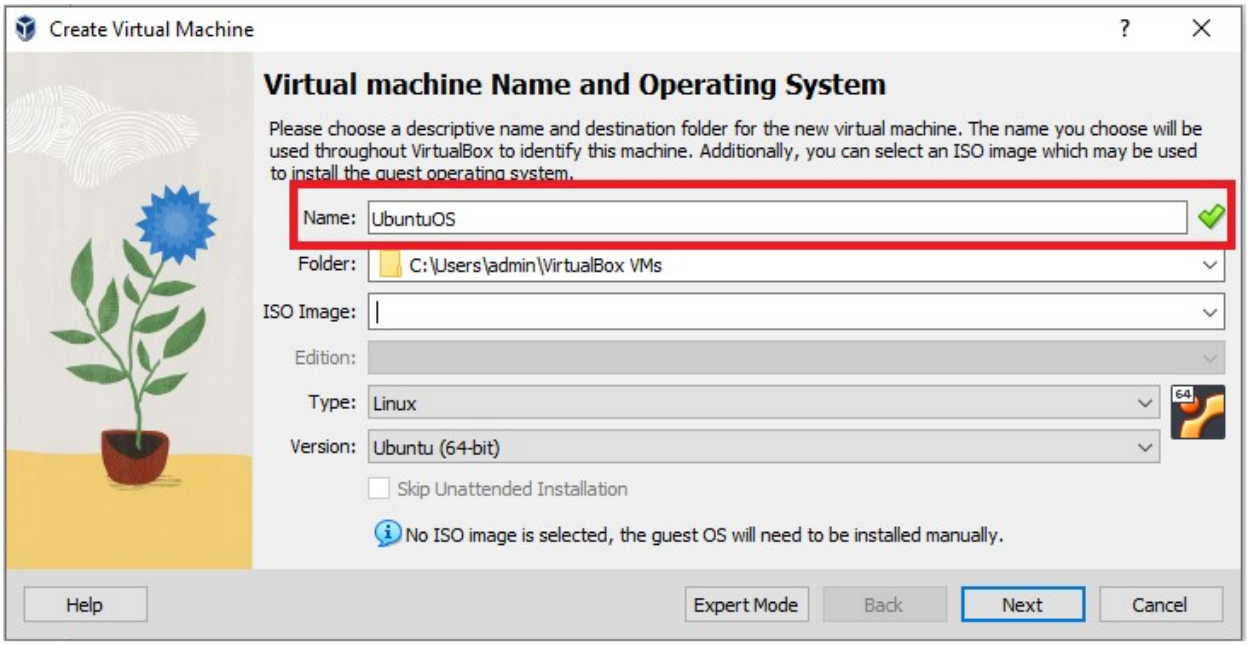


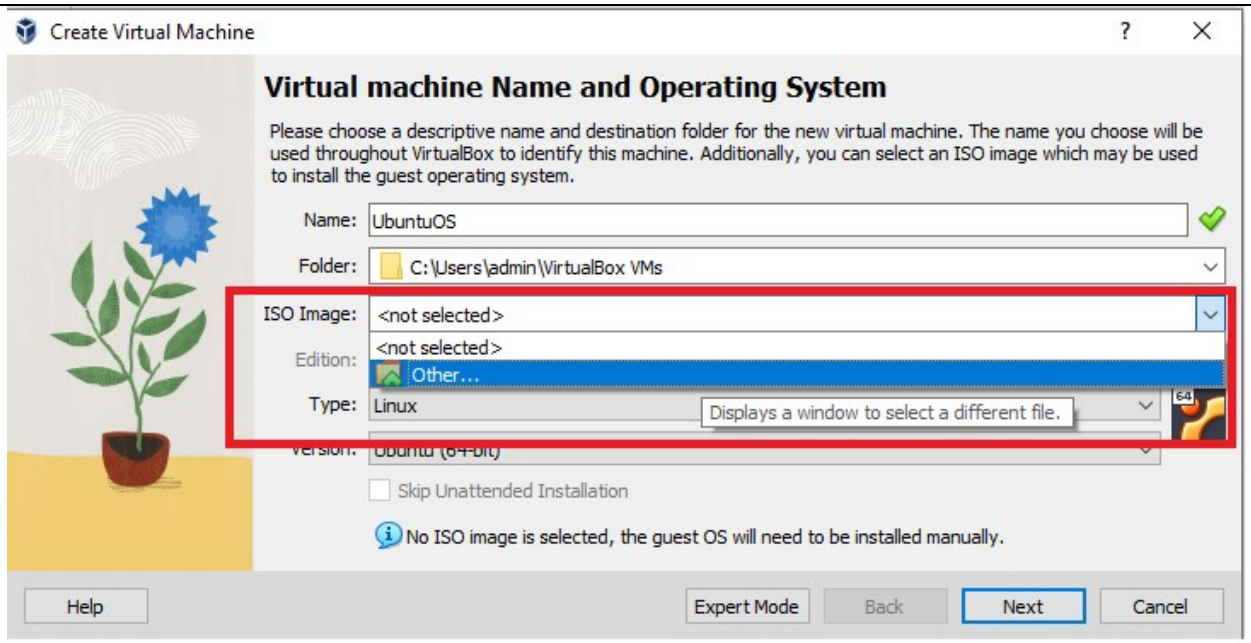
Step 7 **Creating a New Machine with Ubuntu OS in Virtual Box**

Click on New Icon in Virtual Box

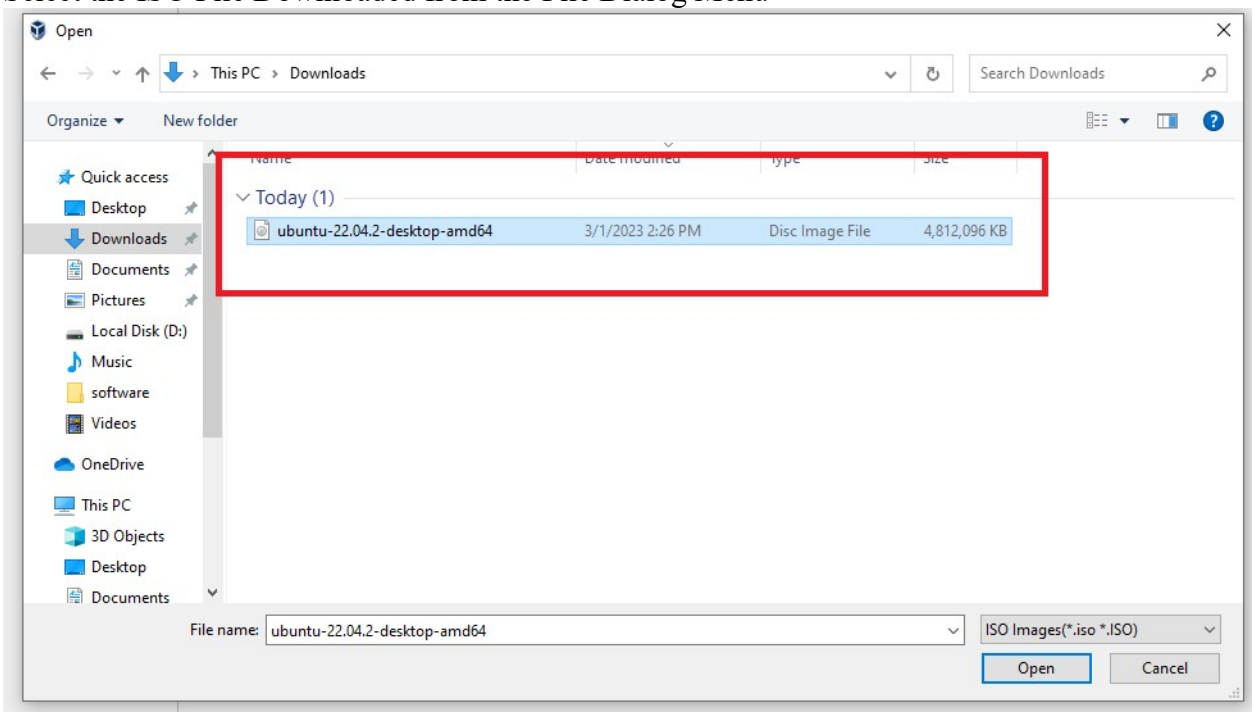



Create Virtual Machine
Virtual machine Name and Operating System
Please choose a descriptive name and destination folder for the new virtual machine. The name you choose will be used throughout VirtualBox to identify this machine. Additionally, you can select an ISO image which may be used to install the guest operating system.
Name:
Folder:
C:\Users\admin\VirtualBox VMs
ISO Image:
<not selected>
Edition:
Type:
Microsoft Windows
Version:
Windows 10 (64-bit)
Skip Unattended Installation
No ISO image is selected, the guest OS will need to be installed manually.
Help
Expert Mode
Back
Next
Cancel


Create Virtual Machine
Virtual machine Name and Operating System
Please choose a descriptive name and destination folder for the new virtual machine. The name you choose will be used throughout VirtualBox to identify this machine. Additionally, you can select an ISO image which may be used to install the guest operating system.
Name:
UbuntuOS
Folder:
C:\Users\admin\VirtualBox VMs
ISO Image:
Edition:
Type:
Linux
Version:
Ubuntu (64-bit)
Skip Unattended Installation
No ISO image is selected, the guest OS will need to be installed manually.
Help
Expert Mode
Back
Next
Cancel

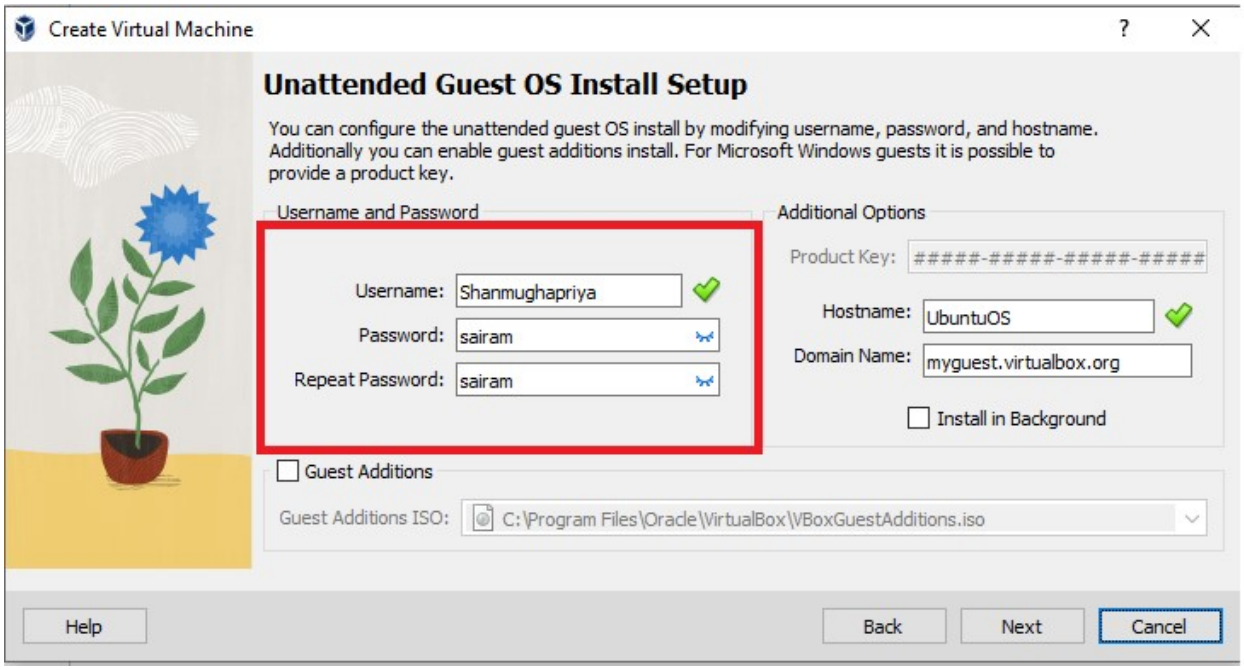


Select the ISO File Downloaded from the File Dialog Menu

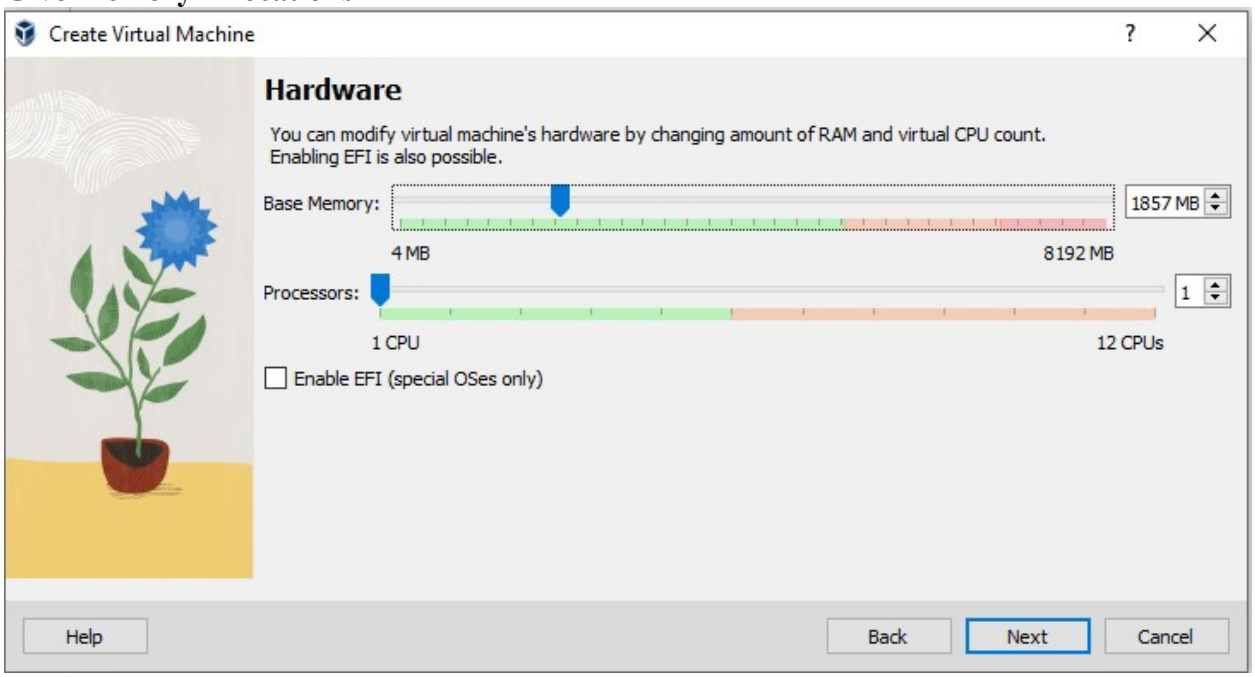


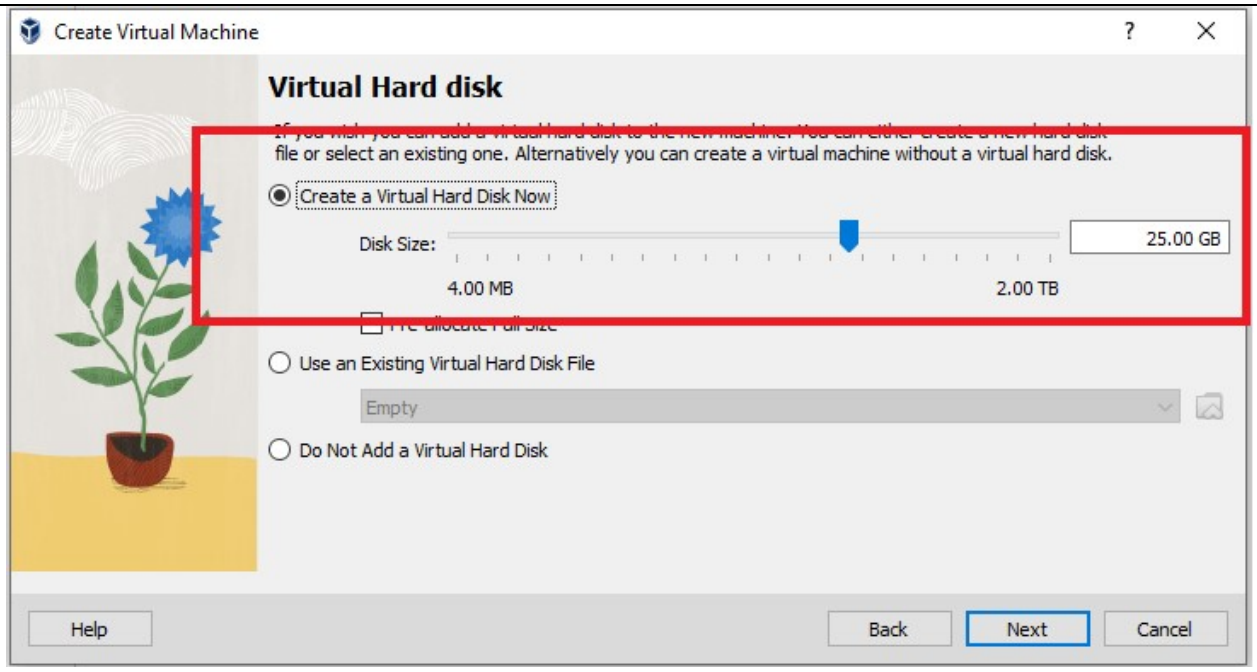
Click on Next and Provide necessary User Name and Password Details of your choice

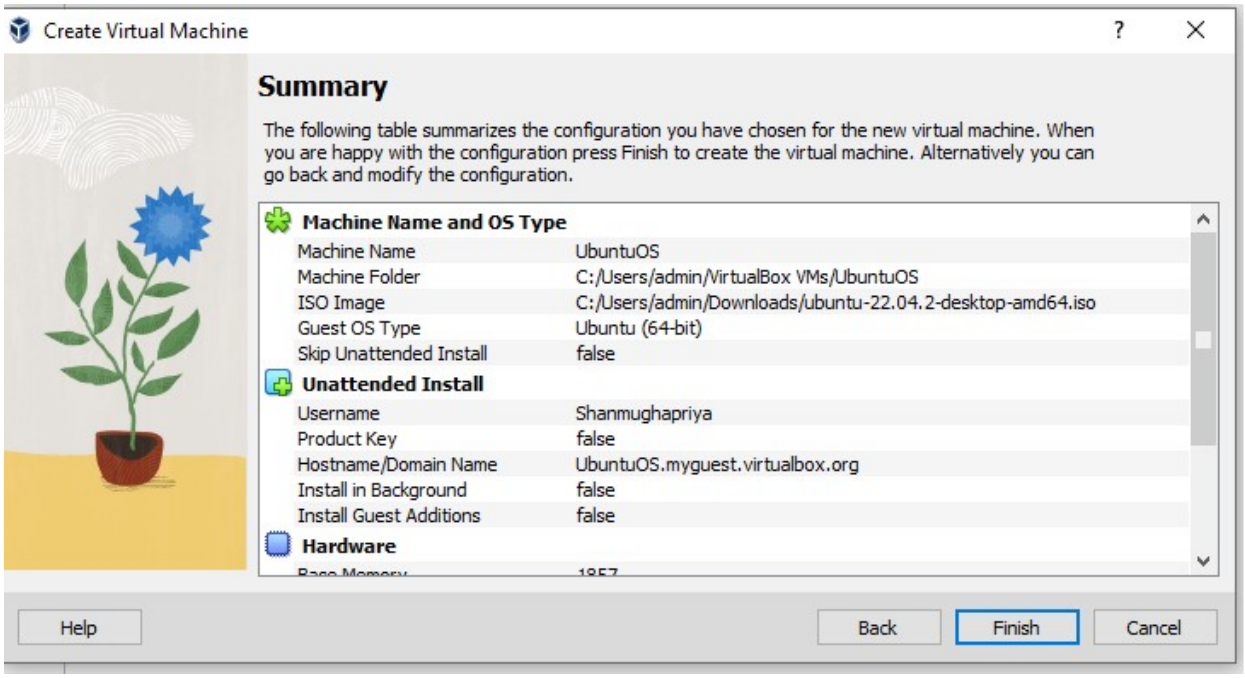
User Name should be lower case



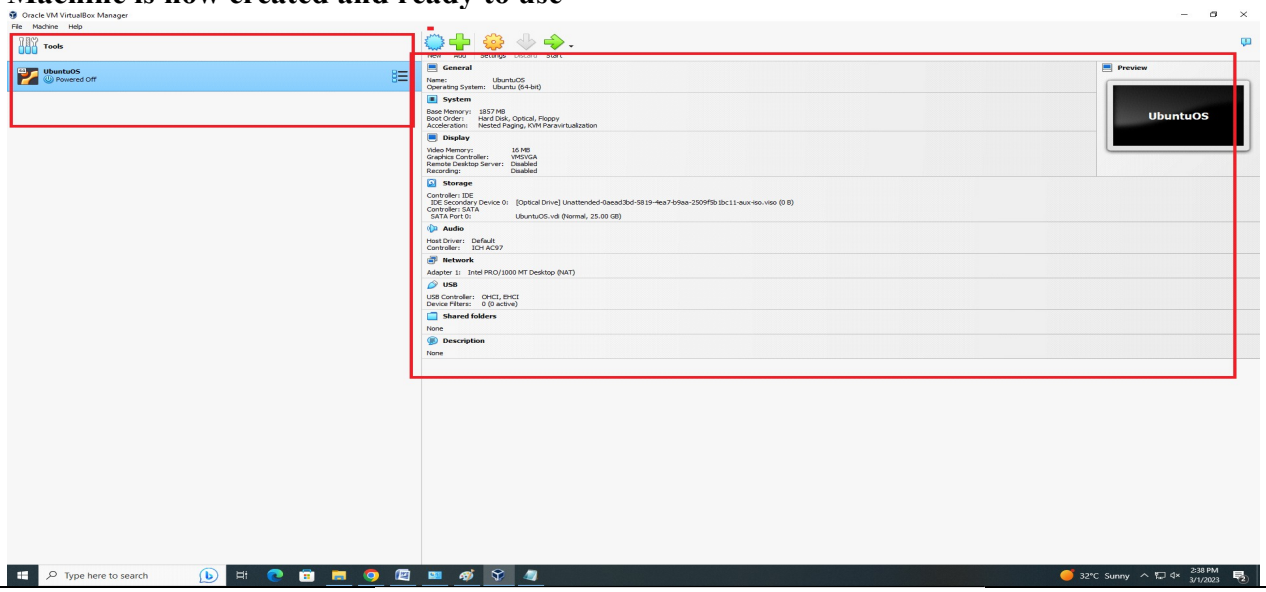
Give Memory Allocations




Create Virtual Machine
Virtual Hard disk
file or select an existing one. Alternatively you can create a virtual machine without a virtual hard disk.
Create a Virtual Hard Disk Now
Disk Size:
25.00 GB
4.00 MB
2.00 TB
Use an Existing Virtual Hard Disk File
Empty
Do Not Add a Virtual Hard Disk
Help
Back
Next
Cancel

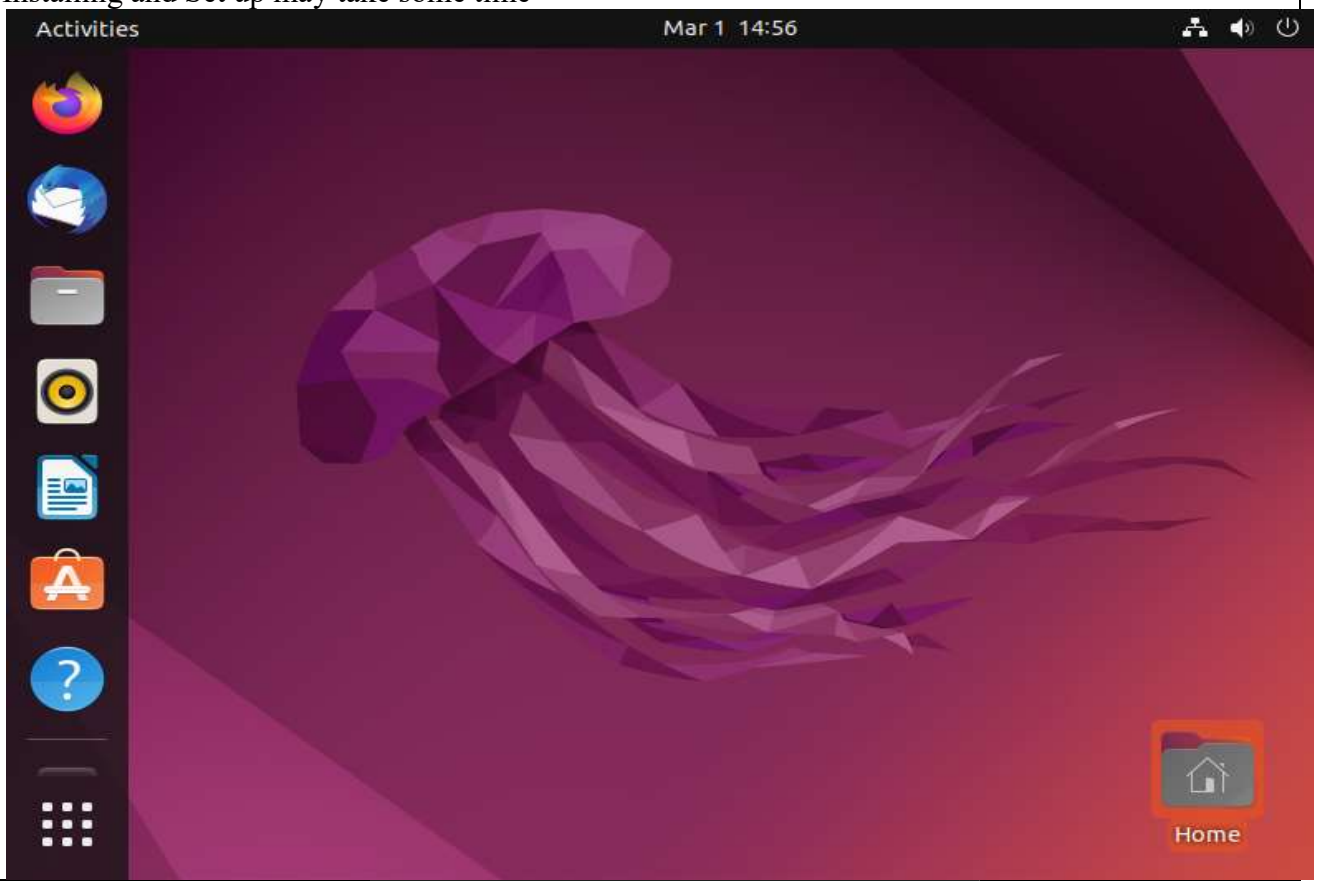

Create Virtual Machine
Summary
The following table summarizes the configuration you have chosen for the new virtual machine. When you are happy with the configuration press Finish to create the virtual machine. Alternatively you can go back and modify the configuration.
Machine Name and OS Type
Machine Name UbuntuOS
Machine Folder C:/Users/admin/VirtualBox VMs/UbuntuOS
ISO Image C:/Users/admin/Downloads/ubuntu-22.04.2-desktop-amd64.iso
Guest OS Type Ubuntu (64-bit)
Skip Unattended Install false
Unattended Install
Username Shanmughapriya
Product Key false
Hostname/Domain Name UbuntuOS.myguest.virtualbox.org
Install in Background false
Install Guest Additions false
Hardware
Help
Back
Finish
Cancel

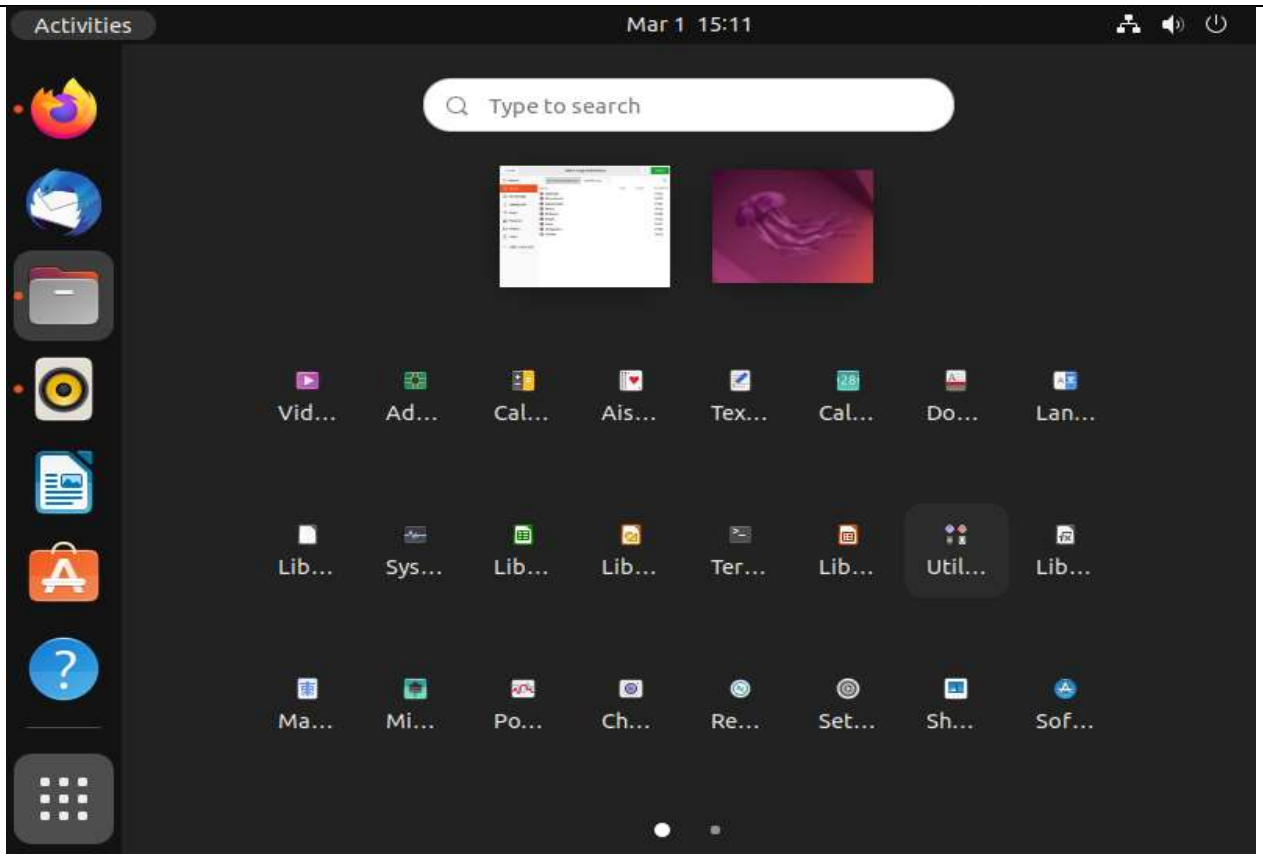
Machine is now created and ready to use

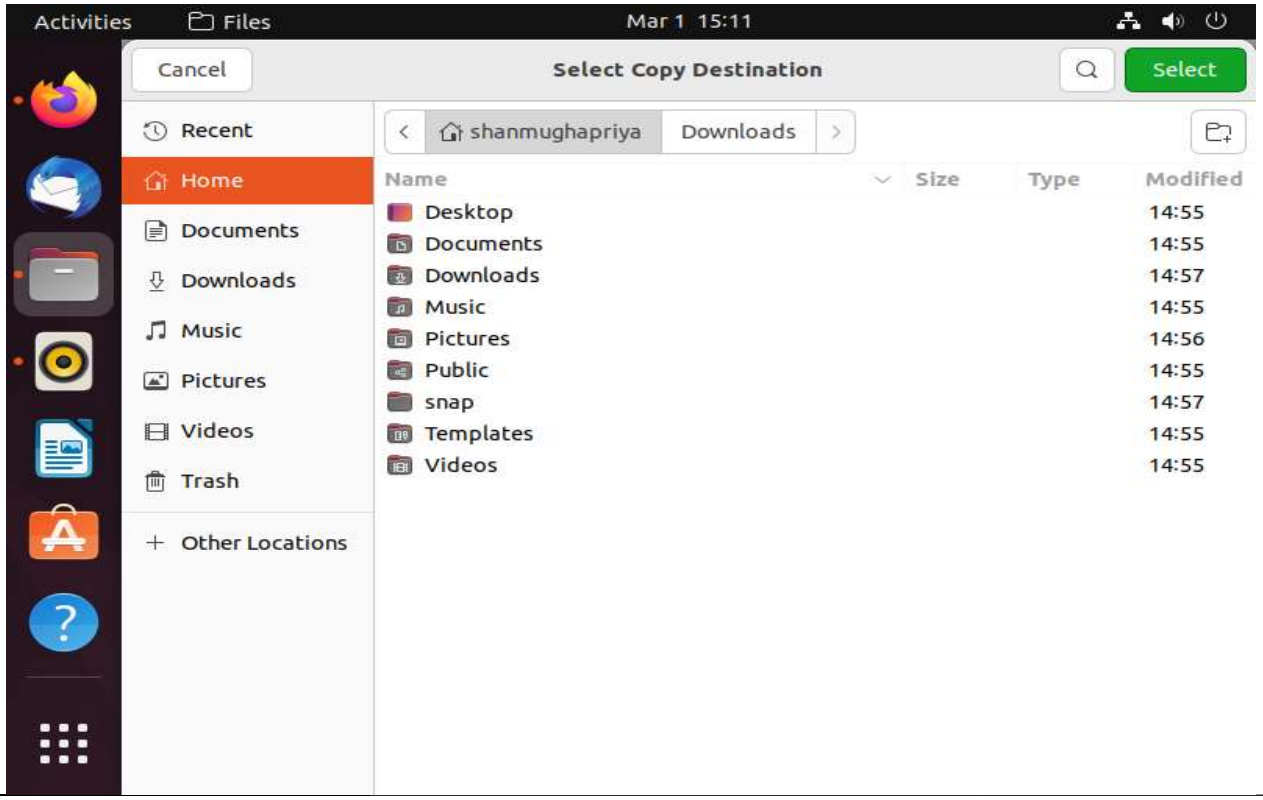


Step 8

Power on the Machine by double clicking on Machine Name
The installation is verified and can be signed in using User Name and Password, Copying, Installing and Set up may take some time



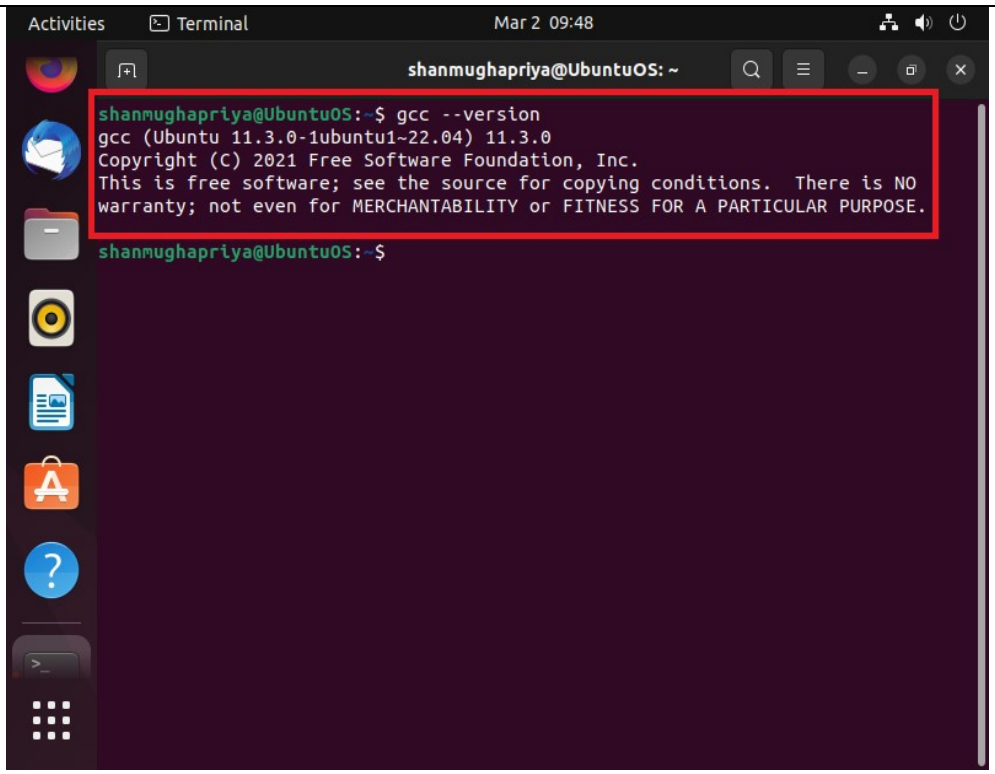

Activities
Mar 1 15:11
Type to search
Vid...
Ad...
Cal...
Ais...
Tex...
Cal...
Do...
Lan...
Lib...
Sys...
Lib...
Lib...
Ter...
Lib...
Util...
Lib...
Ma...
Mi...
Po...
Ch...
Re...
Set...
Sh...
Sof...

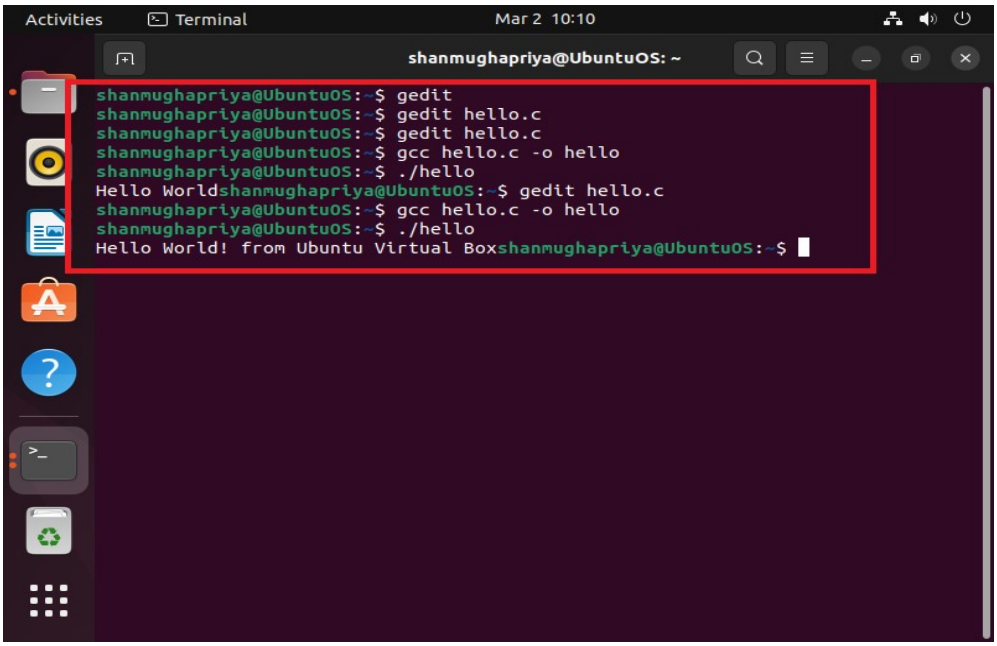

Activities
Files
Mar 1 15:11
Cancel
Select Copy Destination
Select
Recent
Home
Documents
Downloads
Music
Pictures
Videos
Trash
Other Locations
shanmughapriya
Downloads
Name
Size
Type
Modified
Desktop 14:55
Documents 14:55
Downloads 14:57
Music 14:55
Pictures 14:56
Public 14:55
snap 14:57
Templates 14:55
Videos 14:55

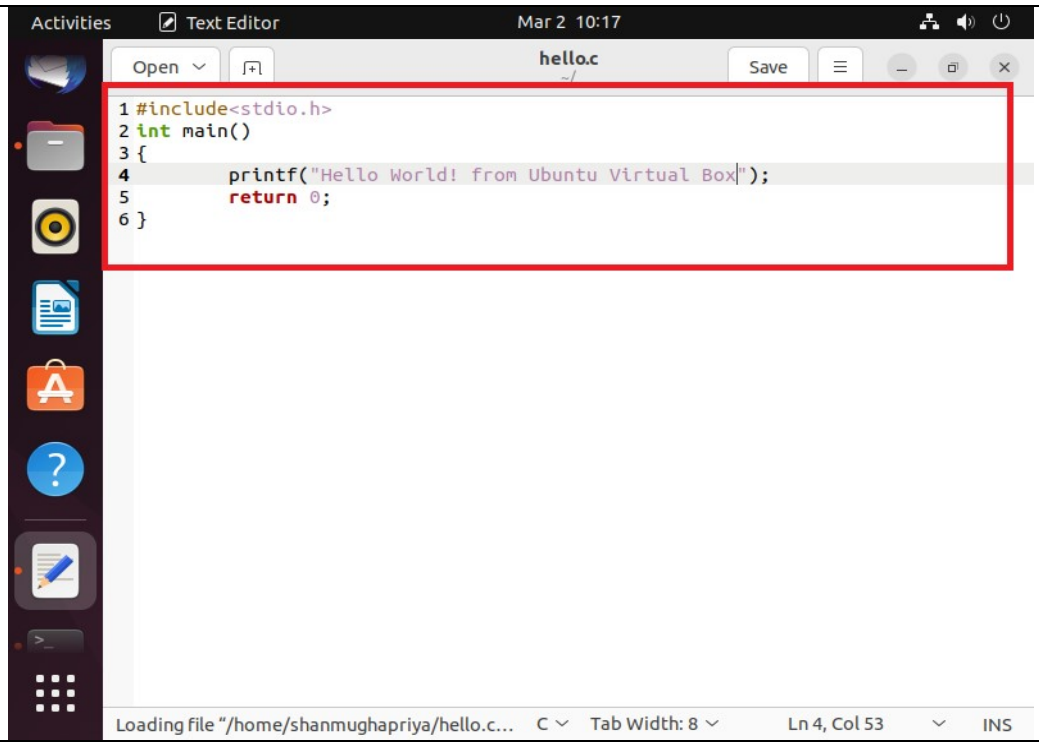
Ex No 2	Install a C compiler in the virtual machine created using virtual box and executes simple programs.
Date	

To install C Compiler in UbuntuOS Machine in Virtual Box, follow the steps below

Step 1	You can find the terminal icon on your Desktop screen. If not, you can find it through the search menu. A shortcut to open terminal is: Press **Ctrl + Alt + T.** Open terminal so that we can proceed with the further steps to install the GCC Compiler. The main command for installing the GCC compiler using terminal on Ubuntu is: Use following commands one by one 1. su root 2. **sudo apt install GCC** **To check the version of C installed** **sudo gcc –version**
Step 2	Use the following command **gedit hello.c** **#include<stdio.h>** **int main()** **{** **printf("Hello World! From Ubuntu in VM Box");** **return 0;** **}** **To create a output file use the commad** **gcc hello.c –o hello** **Output file will be created** **To run the C program use the command** ./hello


Activities
Terminal
Mar 2 09:48
shanmughapriya@UbuntuOS: ~
shanmughapriya@UbuntuOS:~$ gcc --version
gcc (Ubuntu 11.3.0-1ubuntu1~22.04) 11.3.0
Copyright (C) 2021 Free Software Foundation, Inc.
This is free software; see the source for copying conditions. There is NO
warranty; not even for MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.
shanmughapriya@UbuntuOS:~$


Activities
Terminal
Mar 2 10:10
shanmughapriya@UbuntuOS: ~
shanmughapriya@UbuntuOS:~$ gedit
shanmughapriya@UbuntuOS:~$ gedit hello.c
shanmughapriya@UbuntuOS:~$ gedit hello.c
shanmughapriya@UbuntuOS:~$ gcc hello.c -o hello
shanmughapriya@UbuntuOS:~$ ./hello
Hello Worldshanmughapriya@UbuntuOS:~$ gedit hello.c
shanmughapriya@UbuntuOS:~$ gcc hello.c -o hello
shanmughapriya@UbuntuOS:~$ ./hello
Hello World! from Ubuntu Virtual Boxshanmughapriya@UbuntuOS:~$


Activities
Text Editor
Mar 2 10:17
Open
hello.c
~/
Save
1 #include<stdio.h>
2 int main()
3 {
4 printf("Hello World! from Ubuntu Virtual Box");
5 return 0;
6 }
Loading file "/home/shanmughapriya/hello.c...
C
Tab Width: 8
Ln 4, Col 53
INS

Ex No	3. Install Google App Engine. Create hello world app and other simple web applications using Python/Java. 4. Use GAE launcher to launch the Web Applications.
Date	

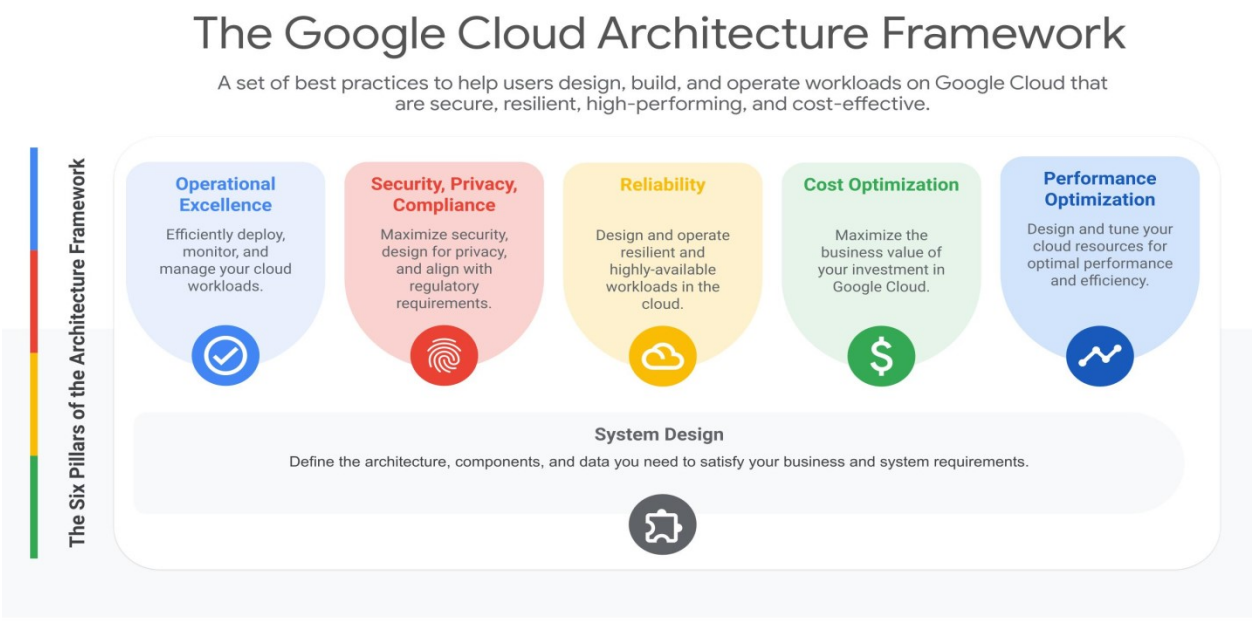
Google Cloud

Google Cloud Platform (GCP), offered by Google, is a suite of cloud computing services that runs on the same infrastructure that Google uses internally for its end-user products, such as Google Search, Gmail, Google Drive, and YouTube. disputed – discuss] Alongside a set of management tools, it provides a series of modular cloud services including computing, data storage, data analytics and machine learning. Registration requires a credit card or bank account details.

Google Cloud Platform provides infrastructure as a service, platform as a service, and serverless computing environments.

In April 2008, Google announced App Engine, a platform for developing and hosting web applications in Google-managed data centers, which was the first cloud computing service from the company. The service became generally available in November 2011. Since the announcement of App Engine, Google added multiple cloud services to the platform.

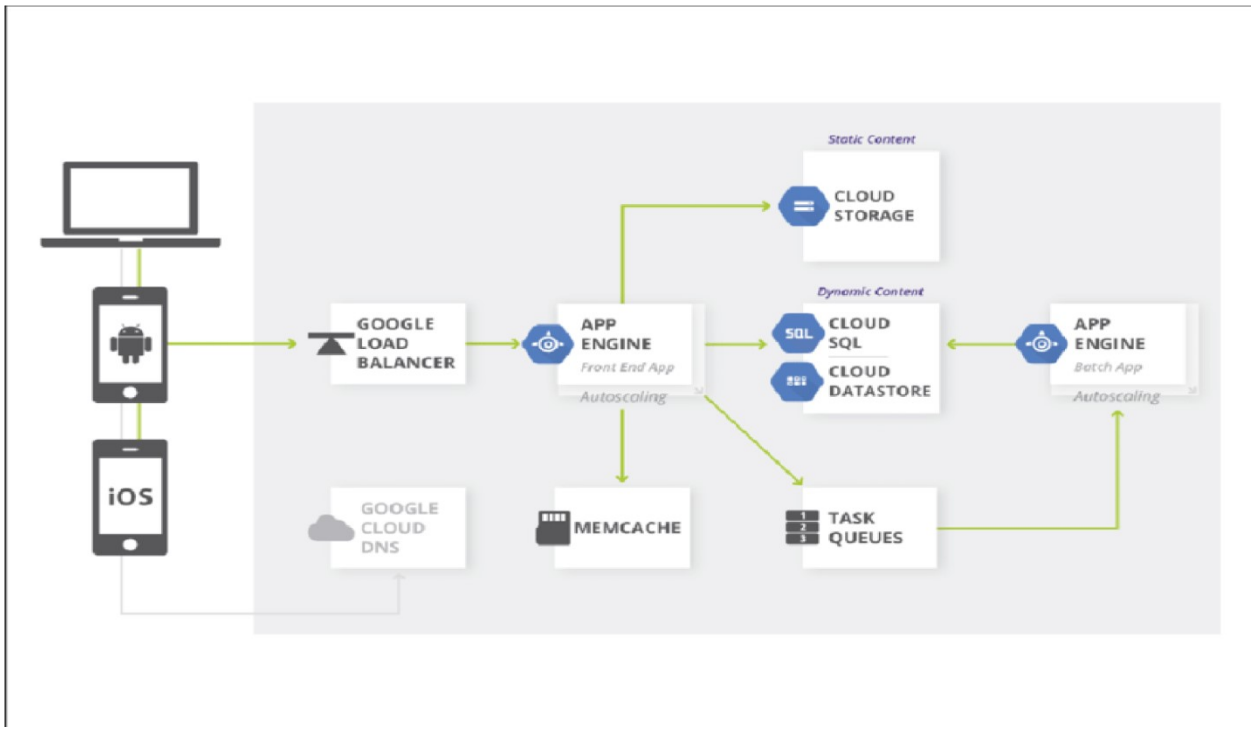
Google Cloud Platform is a part of Google Cloud, which includes the Google Cloud Platform public cloud infrastructure, as well as Google Workspace (G Suite), enterprise versions of Android and ChromeOS, and application programming interfaces (APIs) for machine learning and enterprise mapping services.

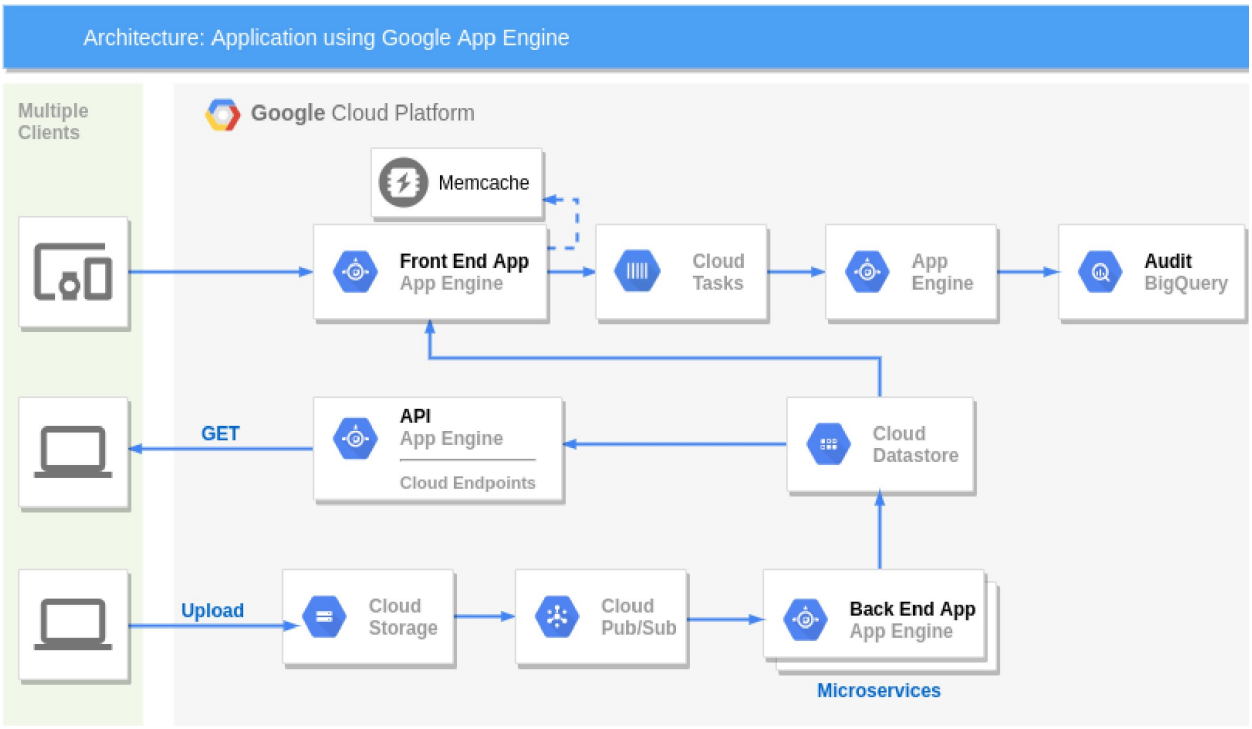


Google App Engine

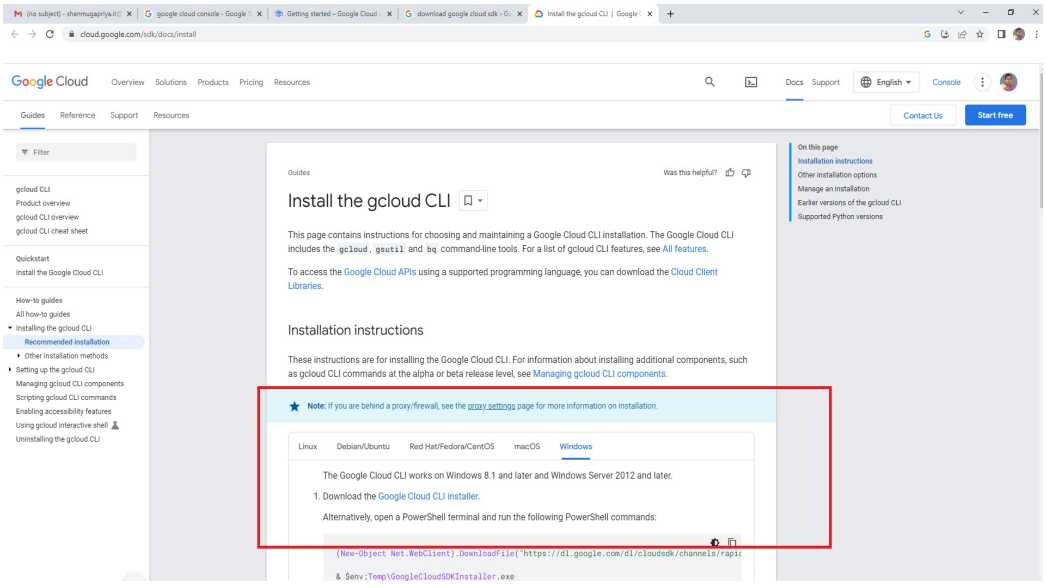
Google App Engine (often referred to as GAE or simply App Engine) is a cloud computing platform as a service for developing and hosting web applications in Google-managed data centers. Applications are sandboxed and run across multiple servers. App Engine offers automatic scaling for web applications—as the number of requests increases for an application, App Engine automatically allocates more resources for the web application to handle the additional demand.

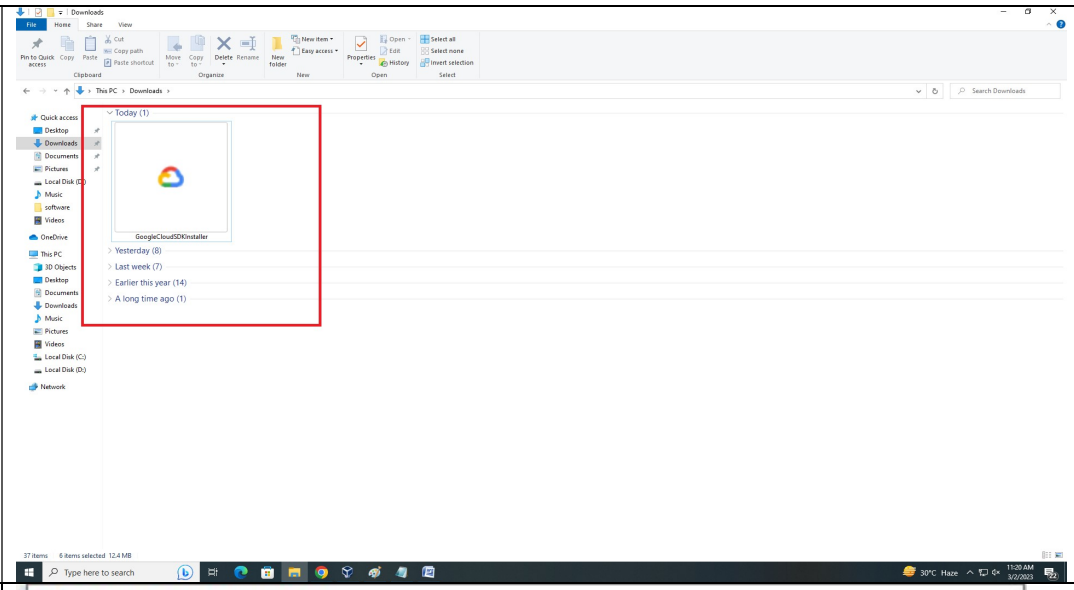
Google App Engine primarily supports Go, PHP, Java, Python, Node.js, .NET, and Ruby applications, although it can also support other languages via "custom runtimes". The service is free up to a certain level of consumed resources and only in standard environment but not in flexible environment. Fees are charged for additional storage, bandwidth, or instance hours required by the application. It was first released as a preview version in April 2008 and came out of preview in September 2011.

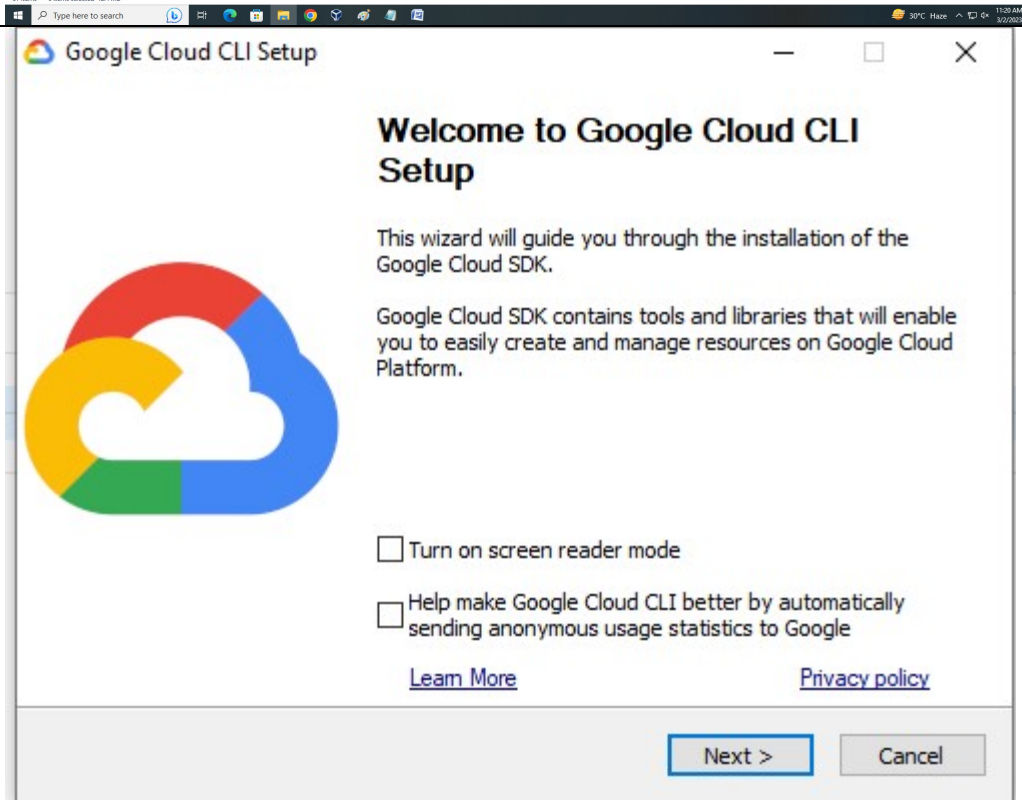


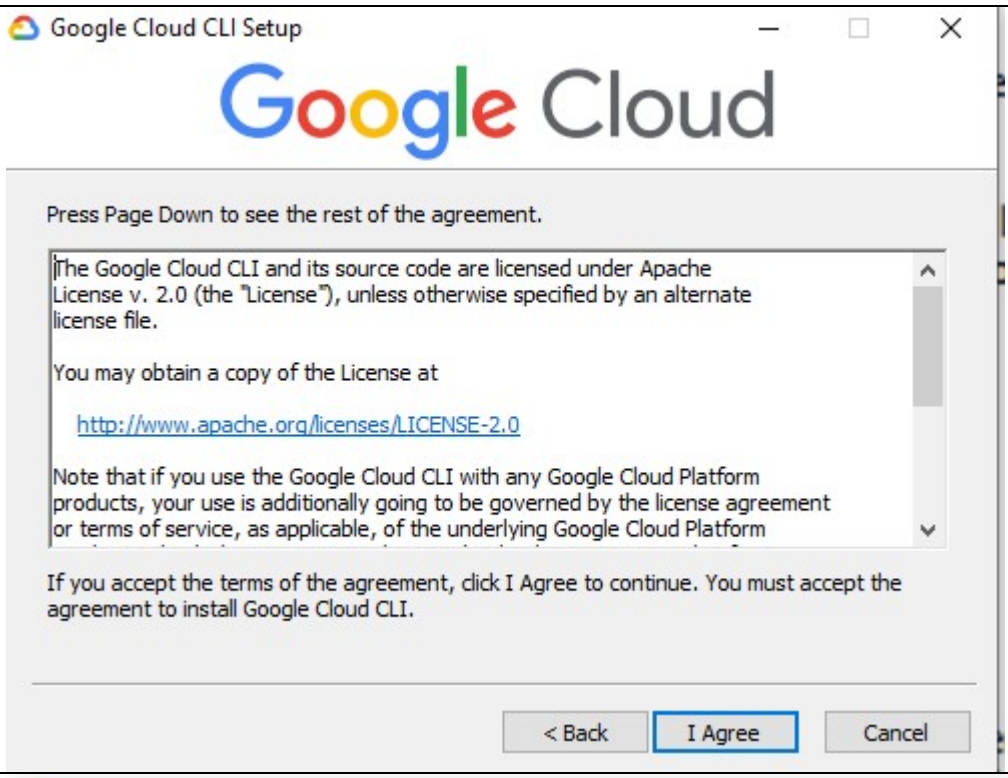


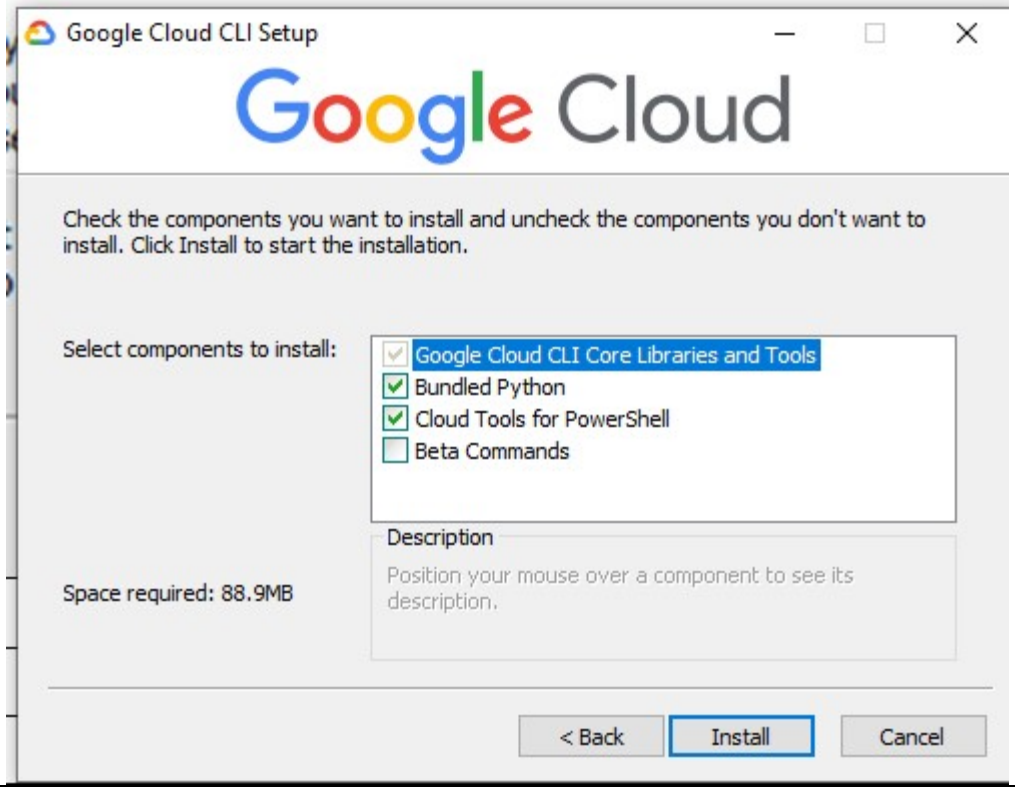
Download and Install Google Cloud Console – Google Cloud SDK and CLI (Command Line Interface)

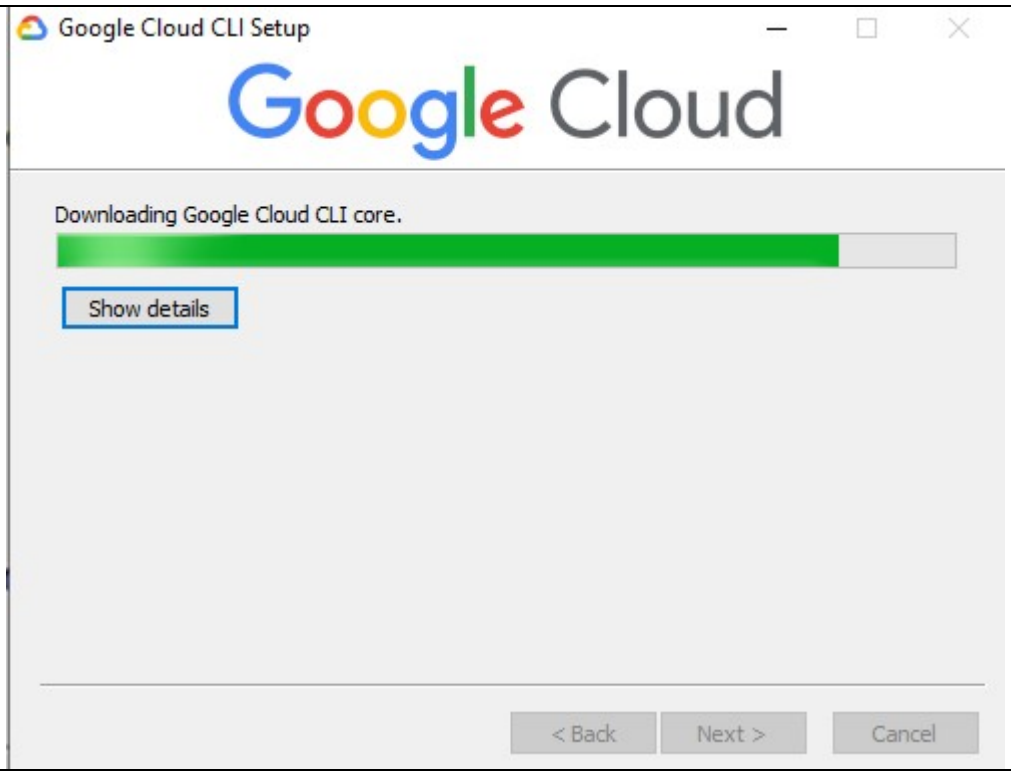
Step 1	**Visit the following website** **https://cloud.google.com/sdk/docs/install** 


Downloads
Today (1)
GoogleCloudSDKInstaller
Yesterday (8)
Last week (7)
Earlier this year (14)
A long time ago (1)


Google Cloud CLI Setup
Welcome to Google Cloud CLI Setup
This wizard will guide you through the installation of the Google Cloud SDK.
Google Cloud SDK contains tools and libraries that will enable you to easily create and manage resources on Google Cloud Platform.
Turn on screen reader mode
Help make Google Cloud CLI better by automatically sending anonymous usage statistics to Google
Learn More
Privacy policy
Next >
Cancel

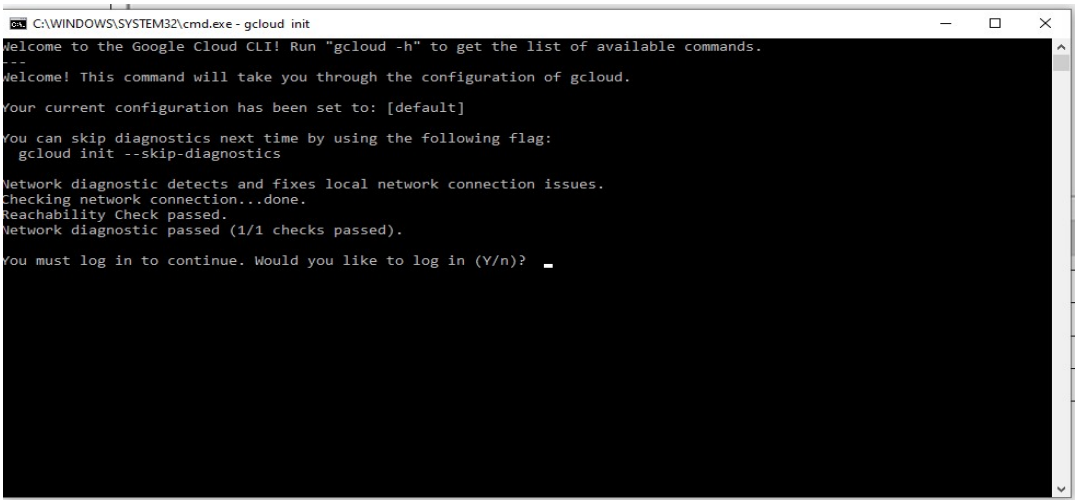

Google Cloud CLI Setup
Google Cloud
Press Page Down to see the rest of the agreement.
The Google Cloud CLI and its source code are licensed under Apache License v. 2.0 (the "License"), unless otherwise specified by an alternate license file.
You may obtain a copy of the License at
http://www.apache.org/licenses/LICENSE-2.0
Note that if you use the Google Cloud CLI with any Google Cloud Platform products, your use is additionally going to be governed by the license agreement or terms of service, as applicable, of the underlying Google Cloud Platform
If you accept the terms of the agreement, click I Agree to continue. You must accept the agreement to install Google Cloud CLI.
< Back
I Agree
Cancel

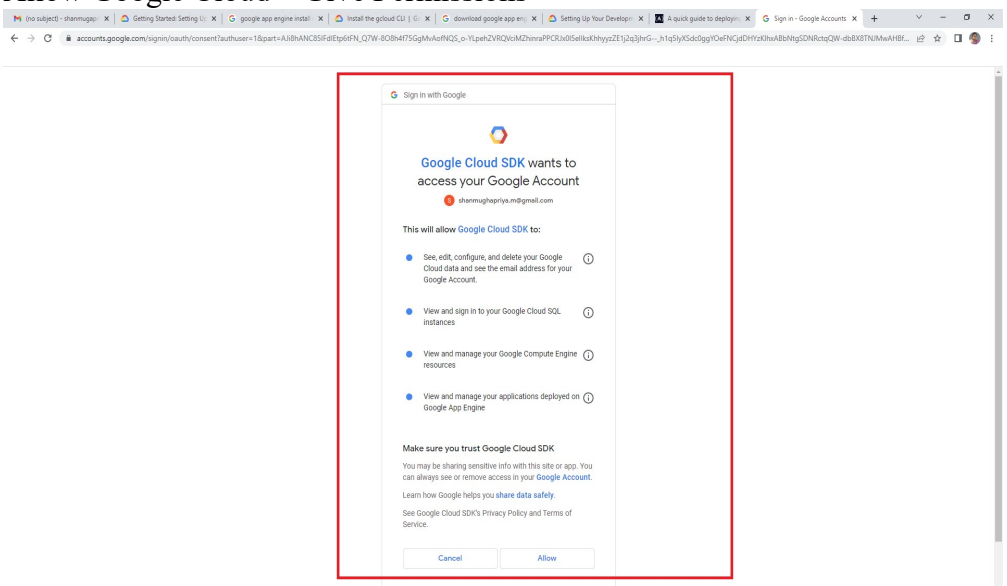
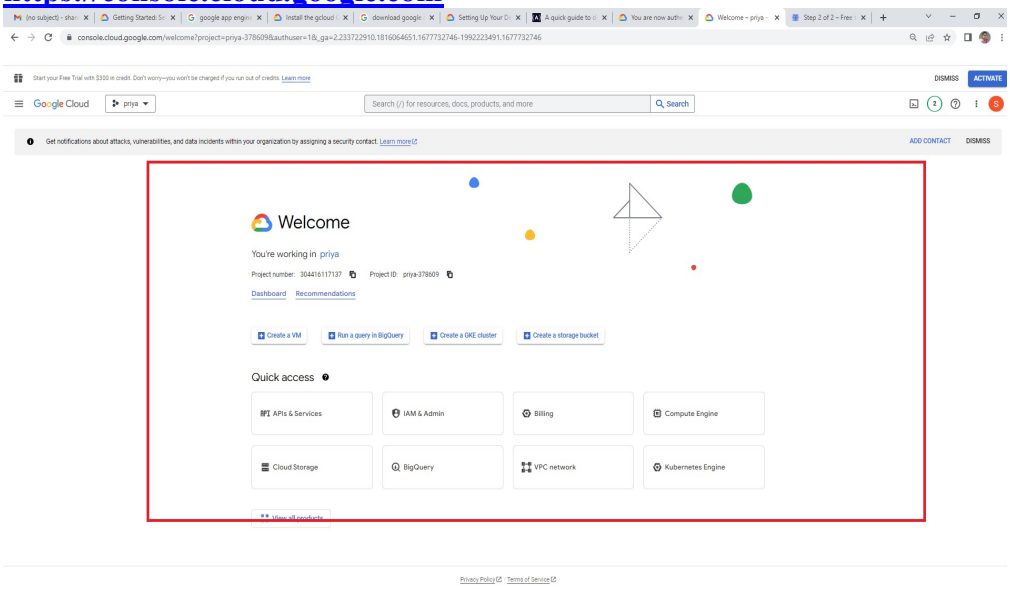

Google Cloud CLI Setup
Google Cloud
Check the components you want to install and uncheck the components you don't want to install. Click Install to start the installation.
Select components to install:
Google Cloud CLI Core Libraries and Tools
Bundled Python
Cloud Tools for PowerShell
Beta Commands
Description
Position your mouse over a component to see its description.
Space required: 88.9MB
< Back
Install
Cancel

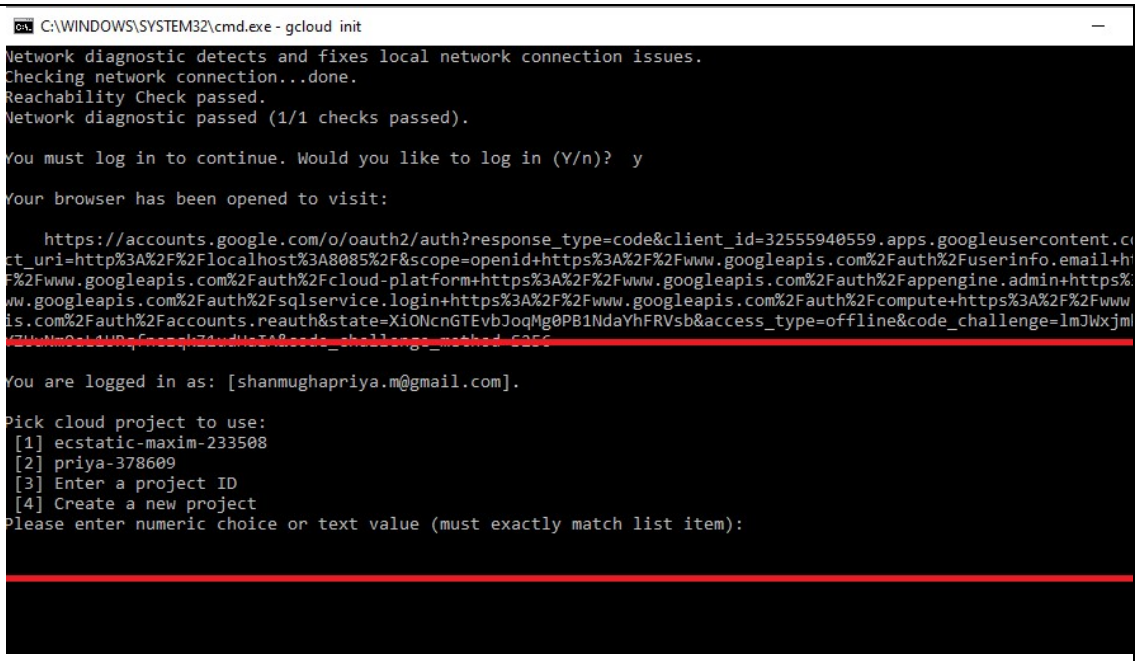


Step 2

Google Cloud SDK CLI – Sign in Using Google Account – Follow the Steps and Sign in

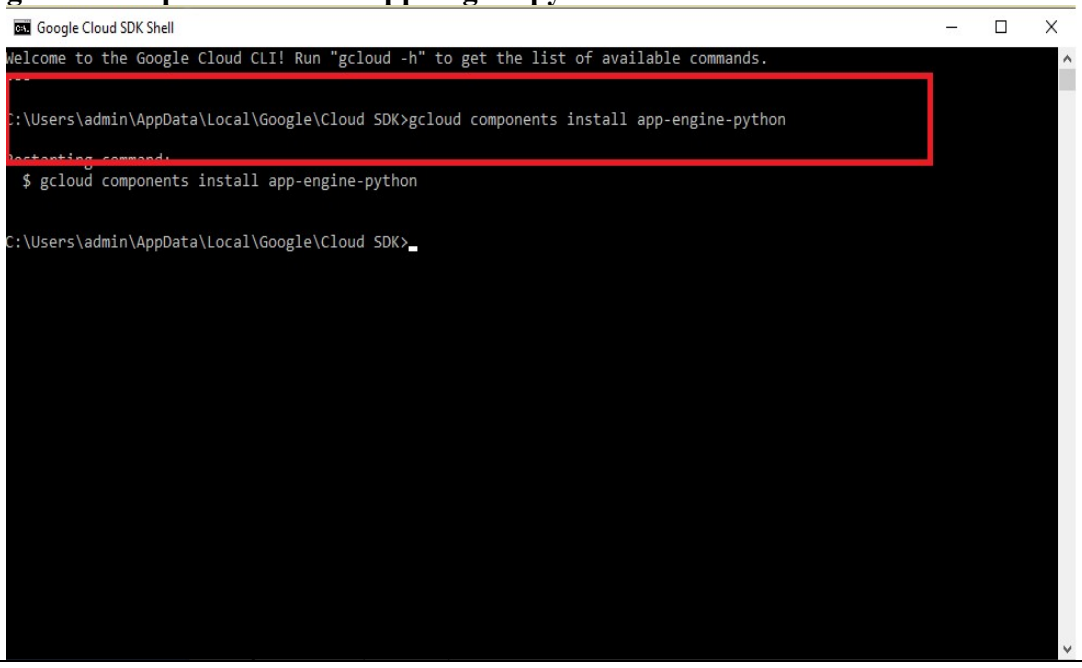


	Allow Google Cloud – Give Permissions 
	Provide necessary information to sign into Google Cloud – Visit the https://console.cloud.google.com 
	After successful sign in Google Cloud, Cloud SDK CLI will look like below

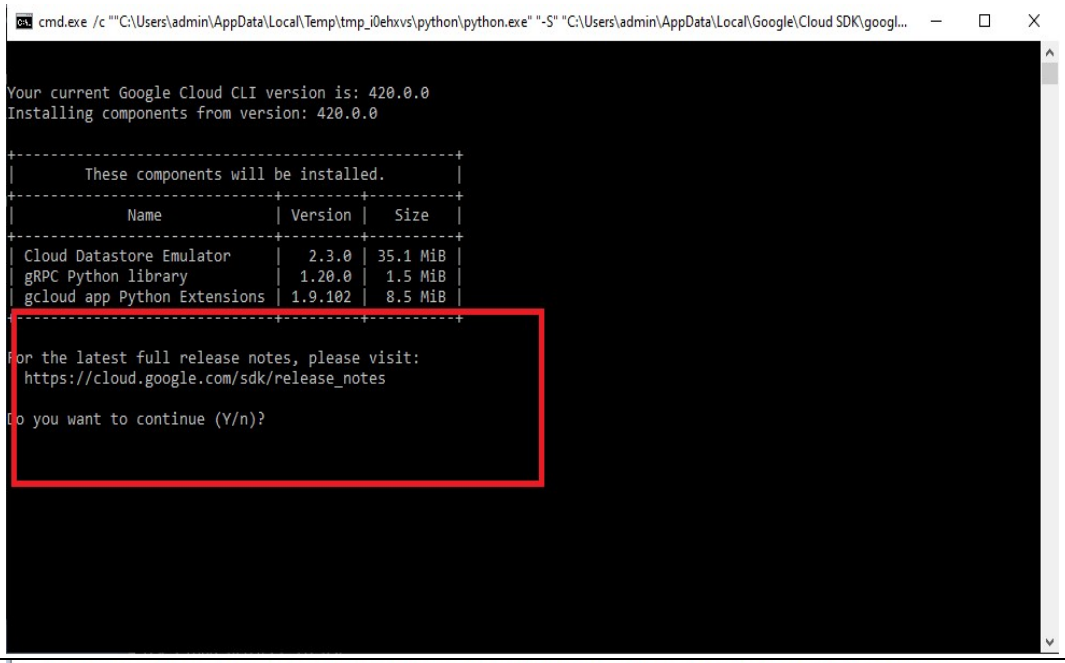


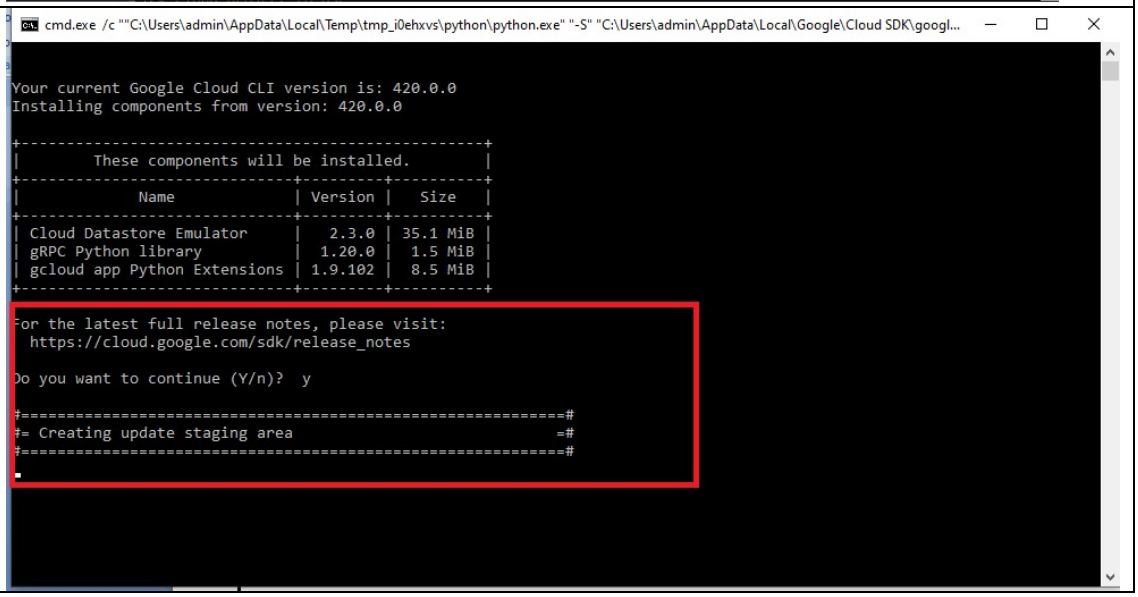
Installing Python Google App Engine in Google Cloud SDK Shell

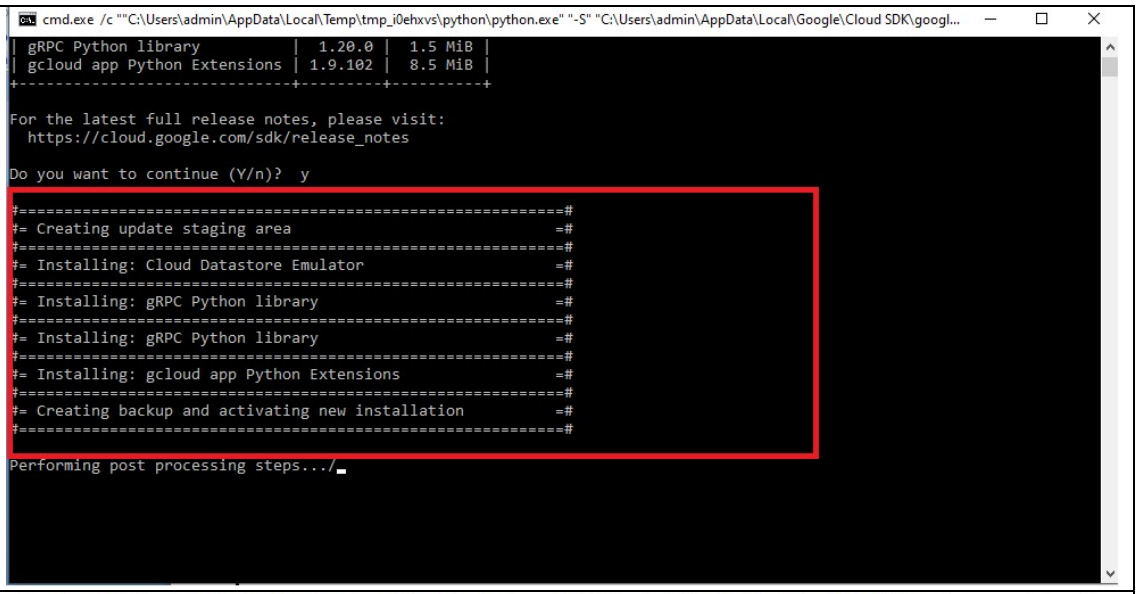
Step 1	**If Python IDLE is a prerequisite for GAE Python,** **If Python is not available, download and install Python from** **https://www.python.org/downloads/**
Step 2	**Open Google Cloud SDK Shell** **Use the following command to n** **gcloud components install app-engine-python**

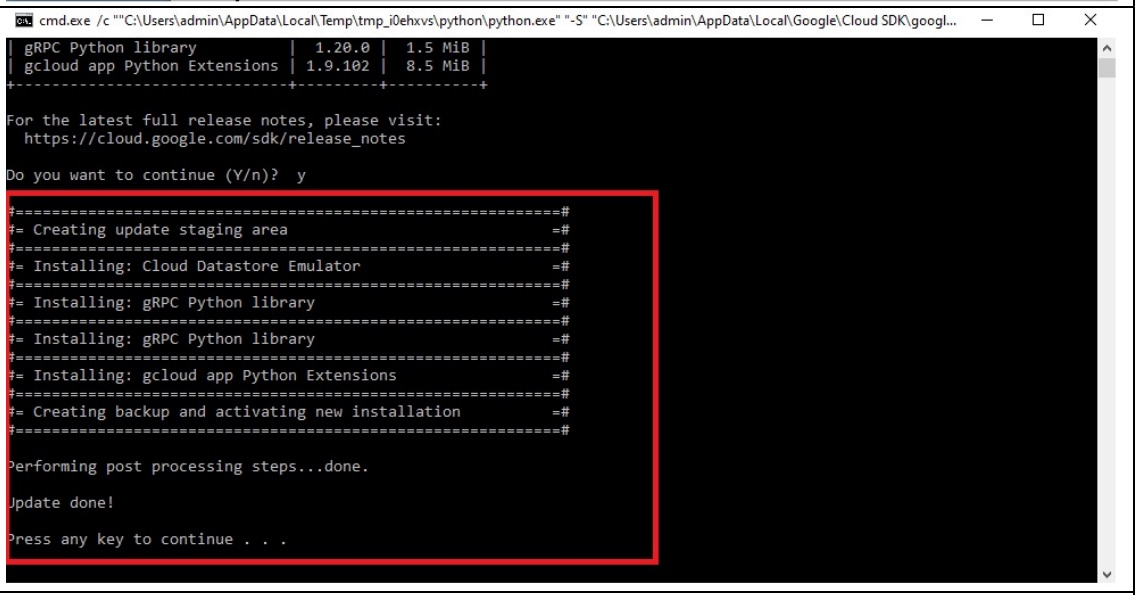


	Give Yes to continue download









File Structure of Python Project File

- `building-an-app/`
  - `app.yaml`
  - `main.py`
  - `requirements.txt`
  - `static/`
    - `script.js`
    - `style.css`
  - `templates/`
    - `index.html`

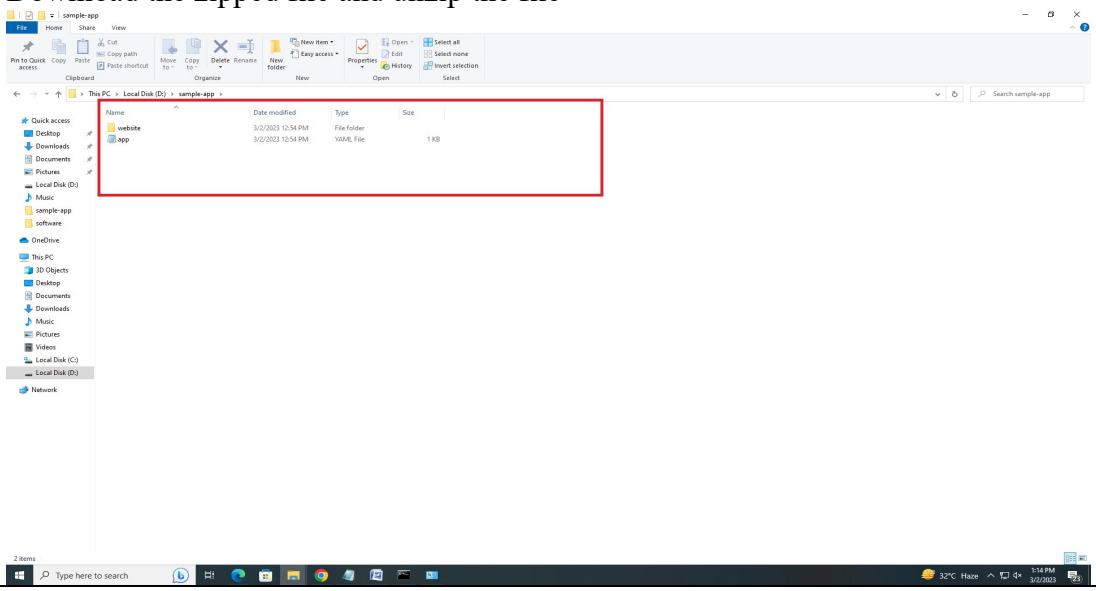
The app. yaml file also **contains information about your app's code, such as the runtime and the latest version identifier**. Each service in your app has its own app. yaml file, which acts as a descriptor for its deployment.

App.yaml designed for serving a static site on Google App Engine (Python). Copy your static html and files into a folder called "static" next to app.yaml. Contains a bunch of mimetype declarations from html5boilerplate's .htaccess. May not be necessary for most situations.

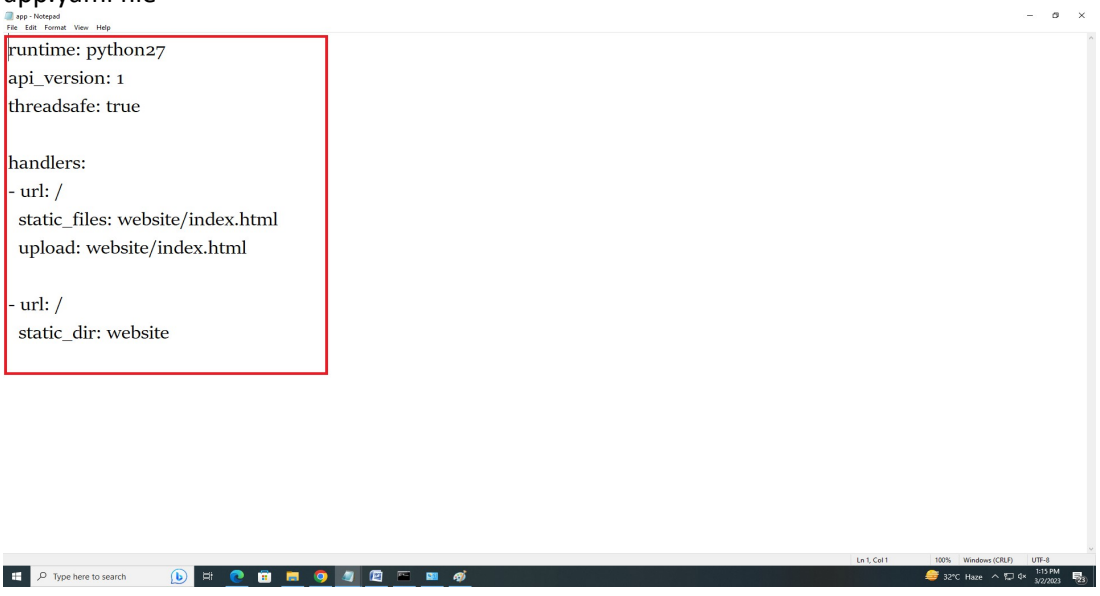
Get the Sample Web App from the following

https://drive.google.com/file/d/1zi9d-W5js0oS2SKg8jap3A2DiDRiABsE/view?usp=sharing

Download the zipped file and unzip the file



app.yaml file

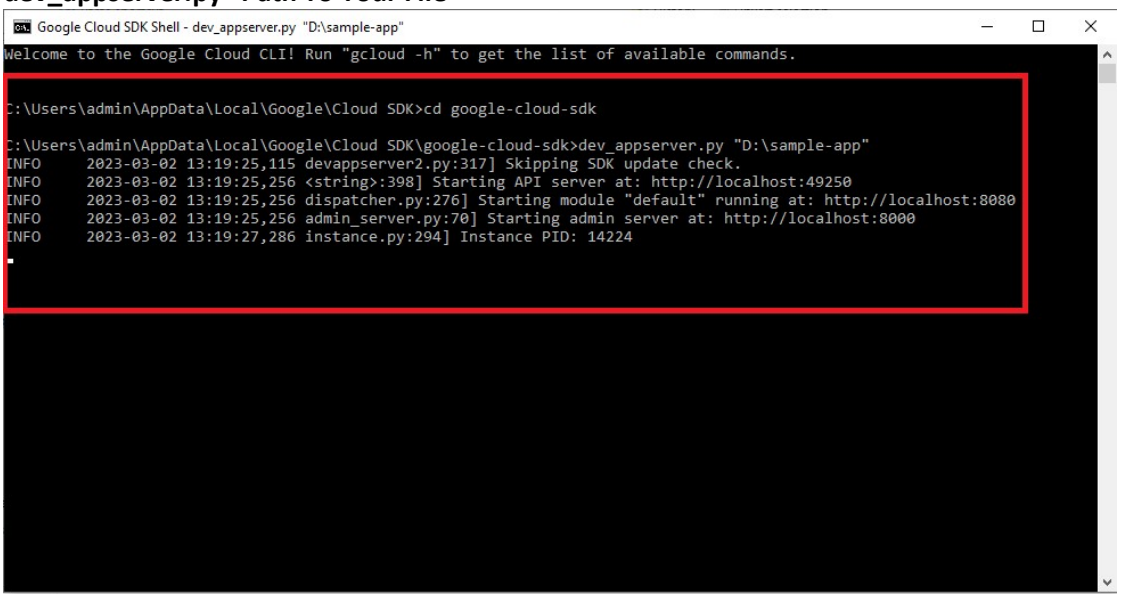


```
runtime: python27
api_version: 1
threadsafe: true

handlers:
- url: /
  static_files: website/index.html
  upload: website/index.html

- url: /
  static_dir: website
```

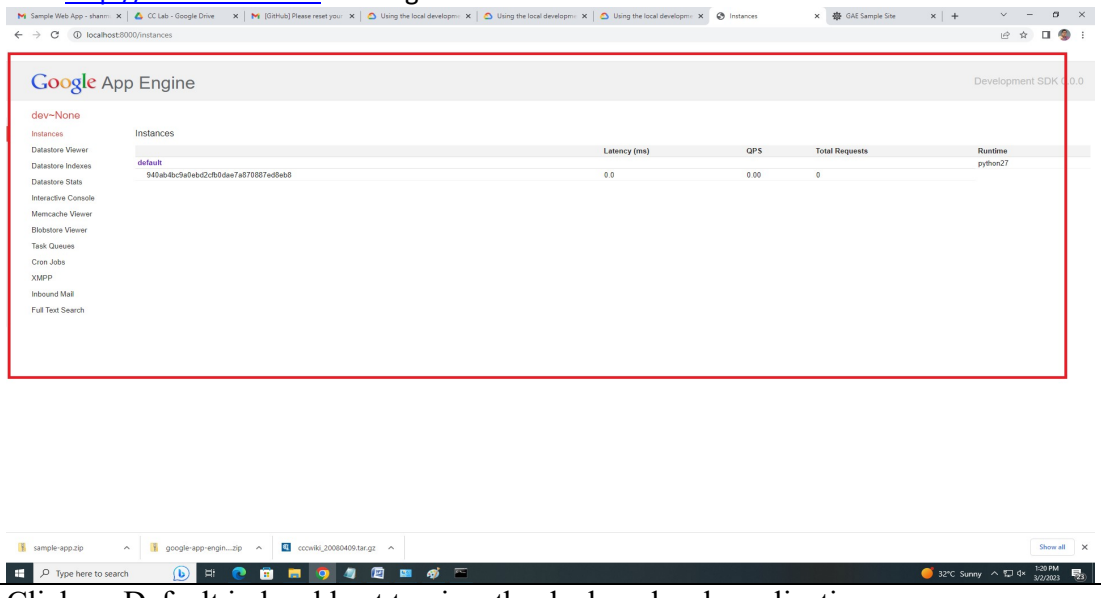
Run the Sample-App using following commands

cd google-cloud-sdk

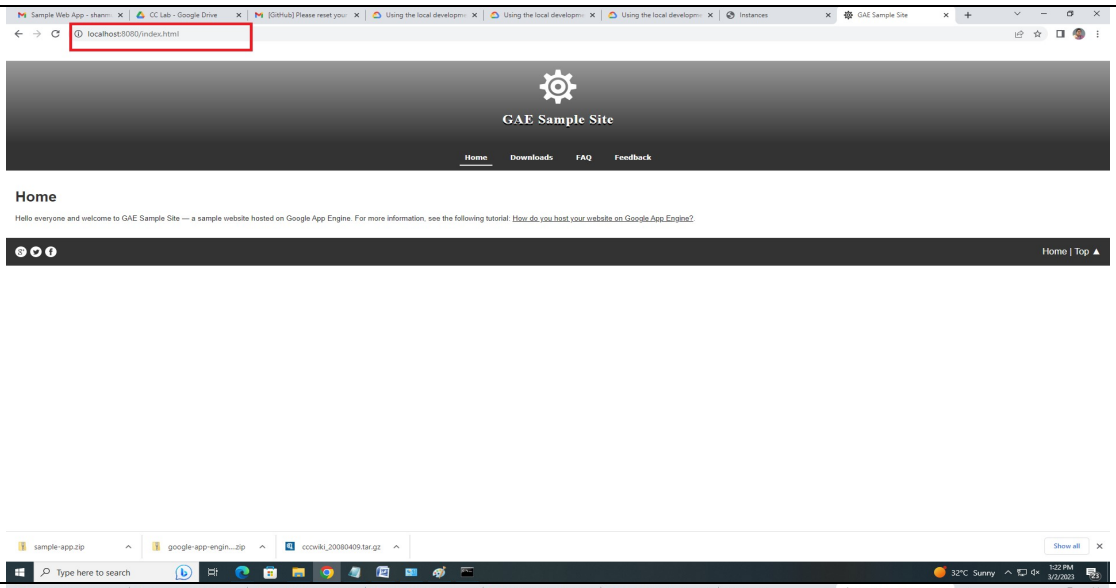
dev_appserver.py "Path To Your File"

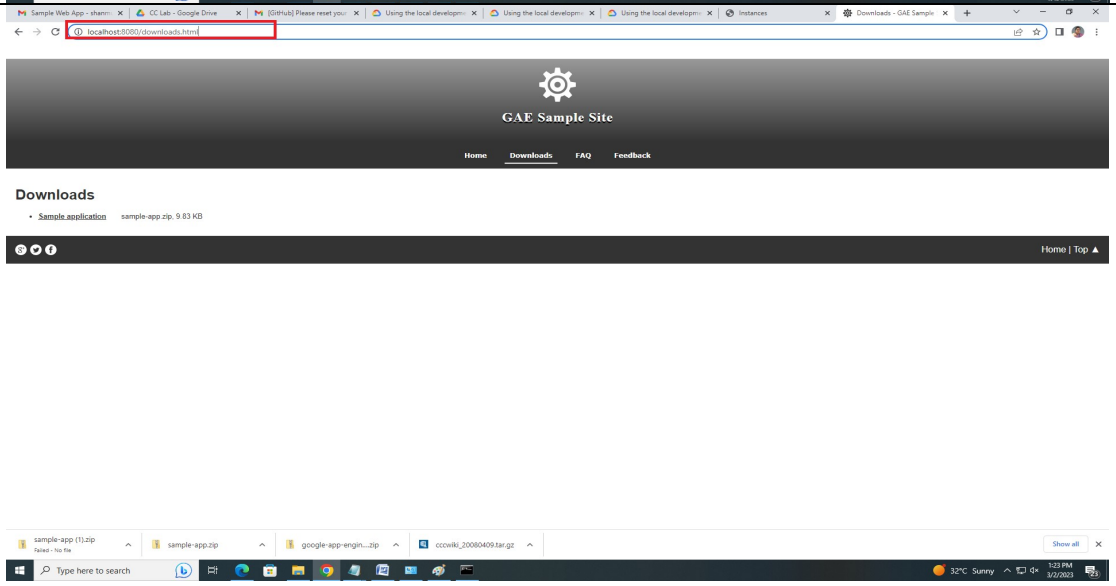


Visit http://localhost:8000 in Google Chrome

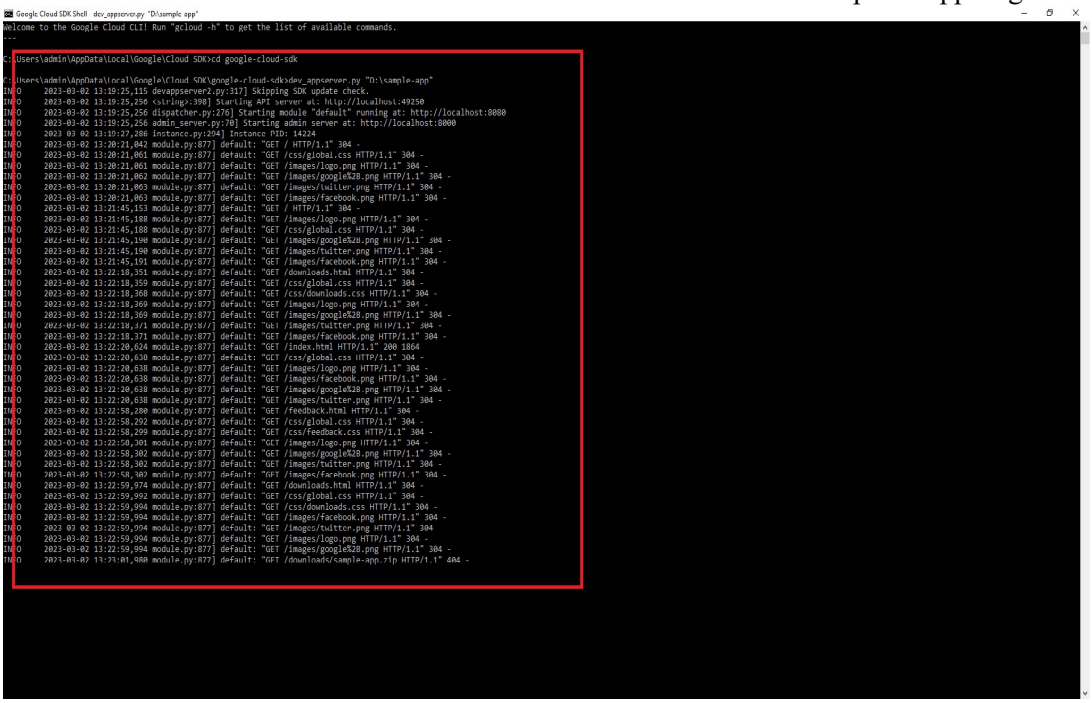


Click on Default in local host to view the deployed web applications


localhost:8080/index.html
GAE Sample Site
Home
Downloads
FAQ
Feedback
Home
Hello everyone and welcome to GAE Sample Site — a sample website hosted on Google App Engine. For more information, see the following tutorial: How do you host your website on Google App Engine?.
Home | Top ▲


localhost:8080/downloads.html
GAE Sample Site
Home
Downloads
FAQ
Feedback
Downloads
Sample application
sample-app.zip, 9.83 KB
Home | Top ▲

Status can be viewed in the Cloud SDK – Use **control+break** to stop the App Engine



Ex No 5	Simulate a cloud scenario using CloudSim and run a scheduling algorithm that is not present inCloudSim.
Date	

Cloud Computing is one of the hottest topics in town. It has completely transformed how modern-day applications are developed and maintained with high scalability and low latency.

CloudSim is an open-source framework, which is used to simulate cloud computing infrastructure and services. It is developed by the CLOUDS Lab organization and is written entirely in Java. It is used for modelling and simulating a cloud computing environment as a means for evaluating a hypothesis prior to software development in order to reproduce tests and results.

For example, if you were to deploy an application or a website on the cloud and wanted to test the services and load that your product can handle and also tune its performance to overcome bottlenecks before risking deployment, then such evaluations could be performed by simply coding a simulation of that environment with the help of various flexible and scalable classes provided by the CloudSim package, free of cost.

Benefits of Simulation over the Actual Deployment:

Following are the benefits of CloudSim:

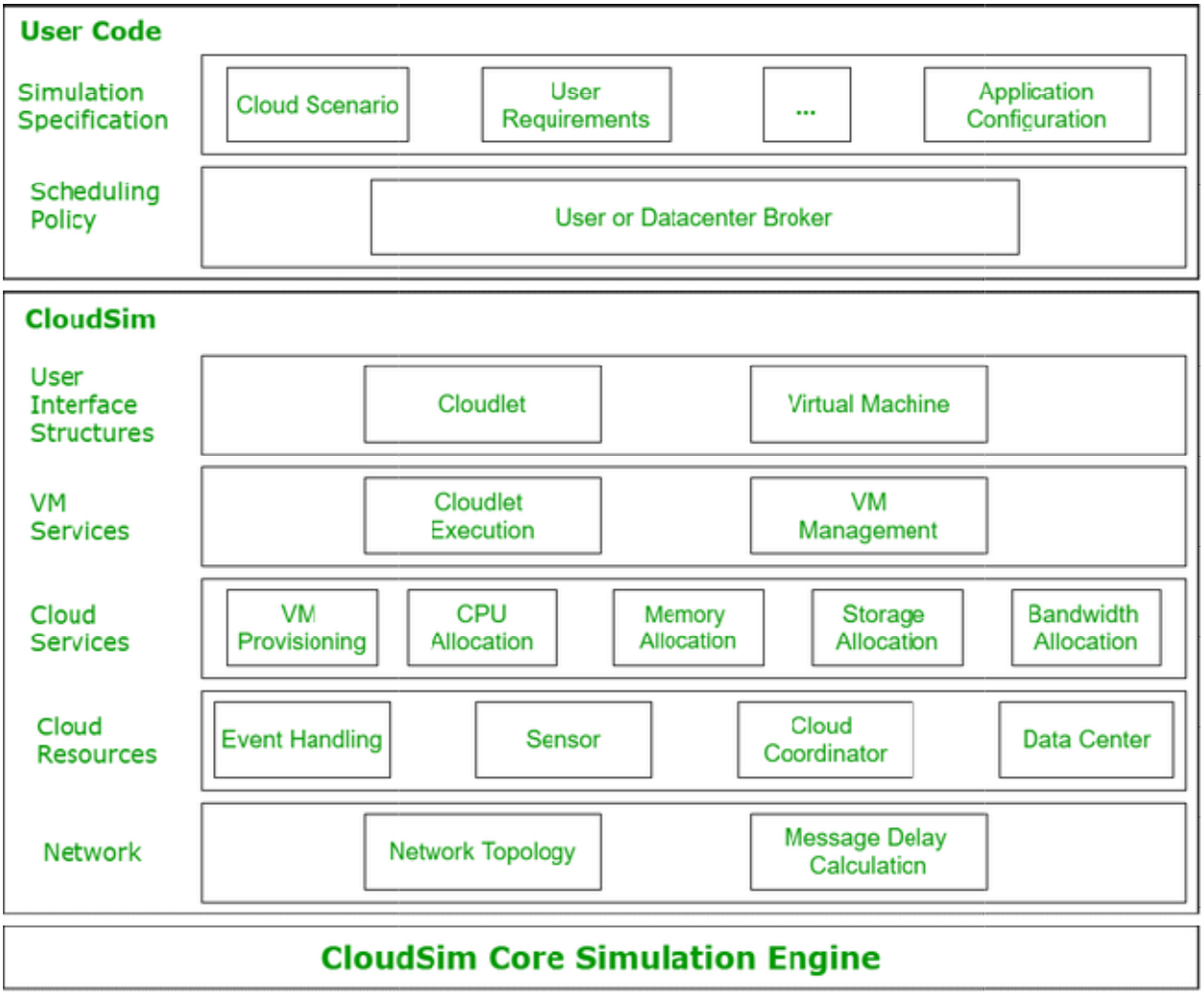
- No capital investment involved. With a simulation tool like CloudSim there is no installation or maintenance cost.
- Easy to use and Scalable. You can change the requirements such as adding or deleting resources by changing just a few lines of code.
- Risks can be evaluated at an earlier stage. In Cloud Computing utilization of real testbeds limits the experiments to the scale of the testbed and makes the reproduction of results an extremely difficult undertaking. With simulation, you can test your product against test cases and resolve issues before actual deployment without any limitations.
- No need for try-and-error approaches. Instead of relying on theoretical and imprecise evaluations which can lead to inefficient service performance and revenue generation, you can test your services in a repeatable and controlled environment free of cost with CloudSim.

Why use CloudSim?

Below are a few reasons to opt for CloudSim:

- Open source and free of cost, so it favours researchers/developers working in the field.
- Easy to download and set-up.
- It is more generalized and extensible to support modelling and experimentation.
- Does not require any high-specs computer to work on.
- Provides pre-defined allocation policies and utilization models for managing resources, and allows implementation of user-defined algorithms as well.
- The documentation provides pre-coded examples for new developers to get familiar with the basic classes and functions.
- Tackle bottlenecks before deployment to reduce risk, lower costs, increase performance, and raise revenue.

CloudSim Architecture:



CloudSim Core Simulation Engine provides interfaces for the management of resources such as VM, memory and bandwidth of virtualized Datacenters.

CloudSim layer manages the creation and execution of core entities such as VMs, Cloudlets, Hosts etc. It also handles network-related execution along with the provisioning of resources and their execution and management.

User Code is the layer controlled by the user. The developer can write the requirements of the hardware specifications in this layer according to the scenario.

Some of the most common classes used during simulation are:

1. Datacenter: used for modelling the foundational hardware equipment of any cloud environment, that is the Datacenter. This class provides methods to specify the functional requirements of the Datacenter as well as methods to set the allocation policies of the VMs etc.
2. Host: this class executes actions related to management of virtual machines. It also defines policies for provisioning memory and bandwidth to the virtual machines, as well as allocating CPU cores to the virtual machines.
3. VM: this class represents a virtual machine by providing data members defining a VM's bandwidth, RAM, mips (million instructions per second), size while also providing setter and getter methods for these parameters.
4. Cloudlet: a cloudlet class represents any task that is run on a VM, like a processing task, or a memory access task, or a file updating task etc. It stores parameters defining the characteristics of a task such as its length, size, mi (million instructions) and provides methods similarly to VM class while also providing methods that define a task's execution time, status, cost and history.
5. DatacenterBroker: is an entity acting on behalf of the user/customer. It is responsible for functioning of VMs, including VM creation, management, destruction and submission of cloudlets to the VM.
6. CloudSim: this is the class responsible for initializing and starting the simulation environment after all the necessary cloud entities have been defined and later stopping after all the entities have been destroyed.

Features of CloudSim:

- CloudSim provides support for simulation and modelling of:
- Large scale virtualized Datacenters, servers and hosts.
- Customizable policies for provisioning host to virtual machines.
- Energy-aware computational resources.
- Application containers and federated clouds (joining and management of multiple public clouds).
- Datacenter network topologies and message-passing applications.
- Dynamic insertion of simulation entities with stop and resume of simulation.
- User-defined allocation and provisioning policies.

Installation:

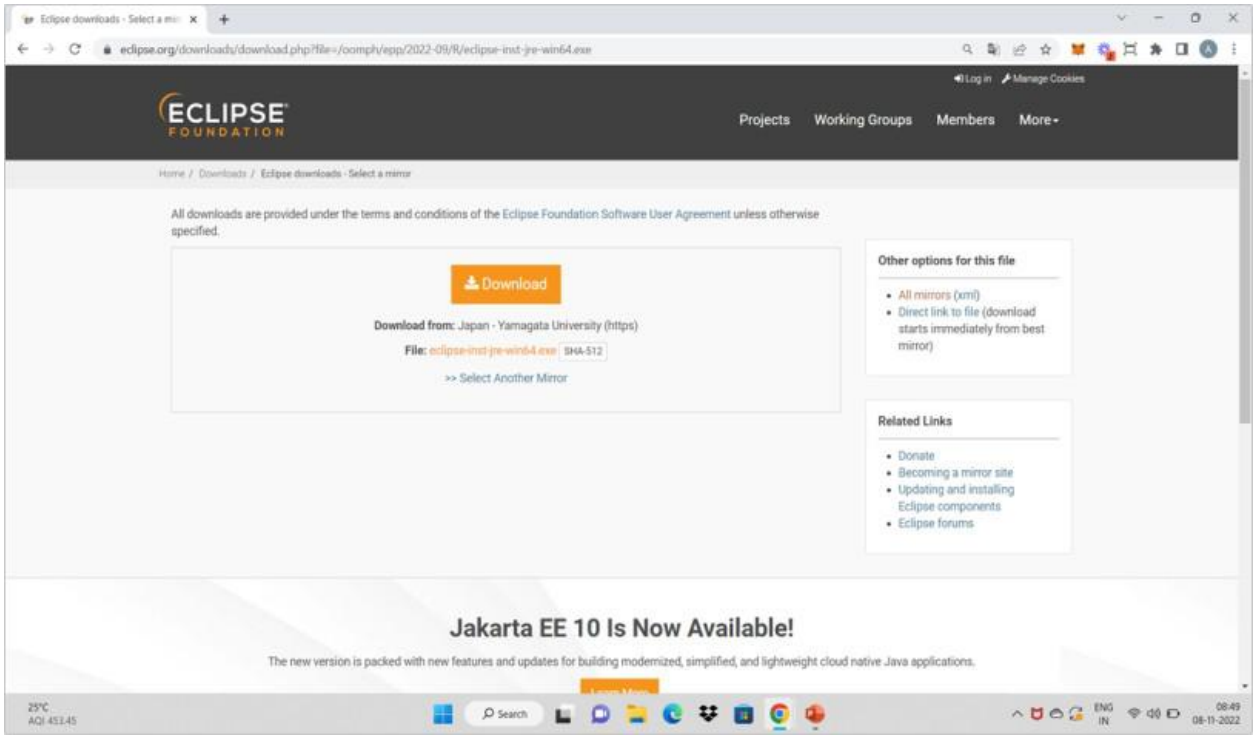
Prerequisites:

- Knowledge of Core Java language features such as OOP and Collections.

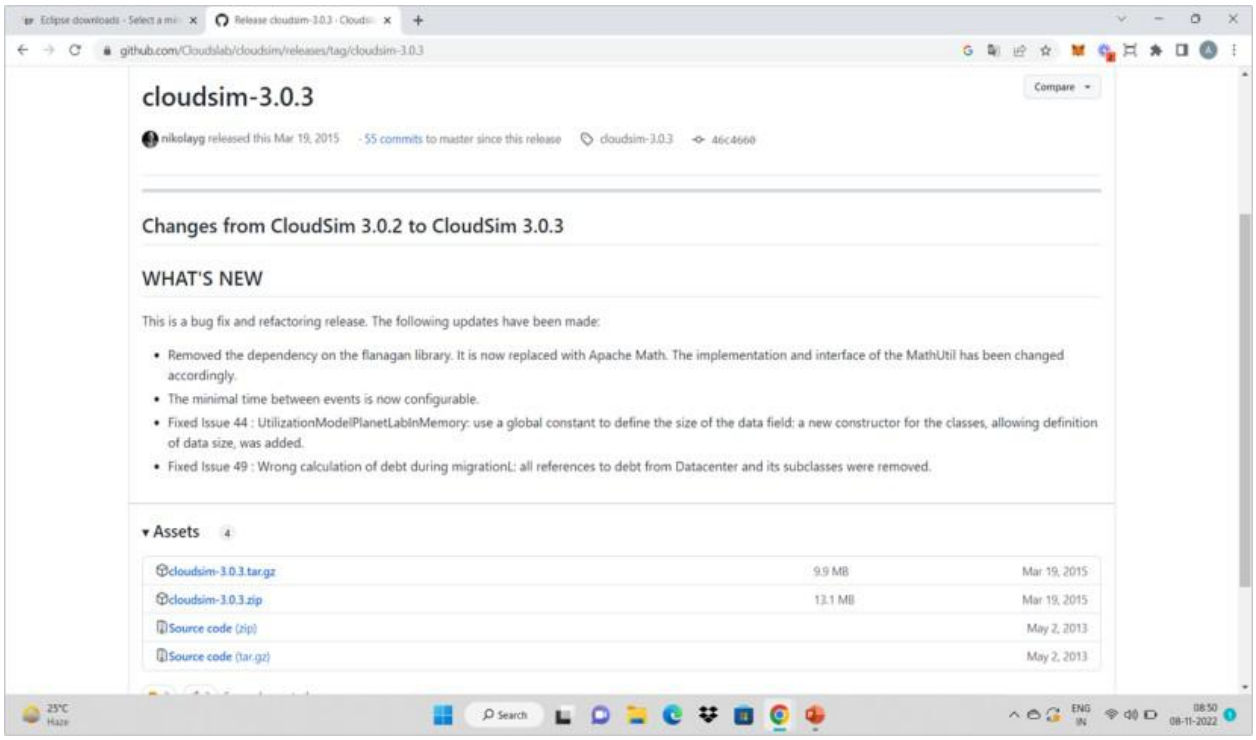
- Basics of Cloud Computing.
- CloudSim is available for download here.
- For this tutorial, we have downloaded zip file of CloudSim 3.0.3.
- Note: CloudSim also uses some utilities of Apache's commons-math3 library. Download its Binaries zip file from here.

Step-by-Step Implementation

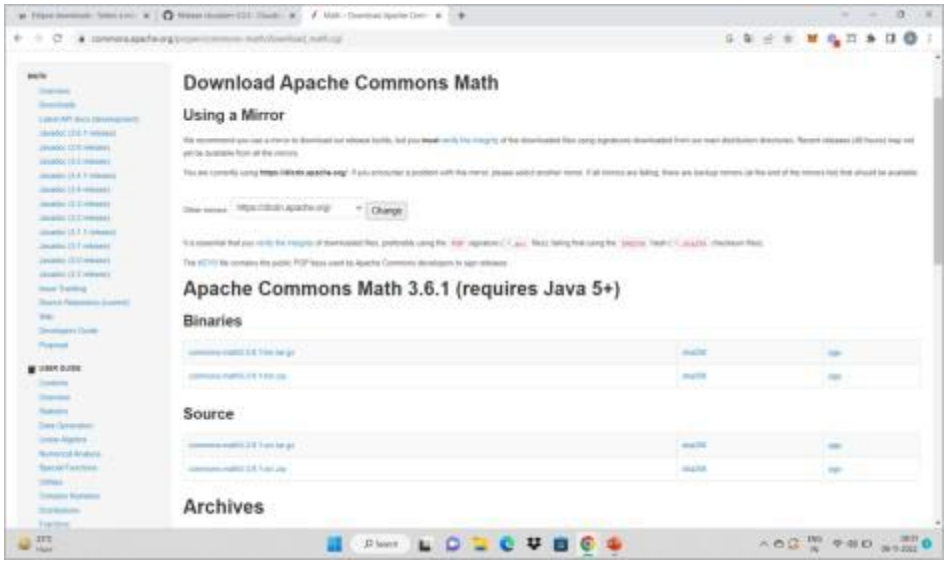
Step 1: Download Eclipse IDE for Java Developers. It is developed by the CLOUDS Lab organization and is written entirely in Java so we would need a Java IDE to simulate a cloud scenario using CloudSim.



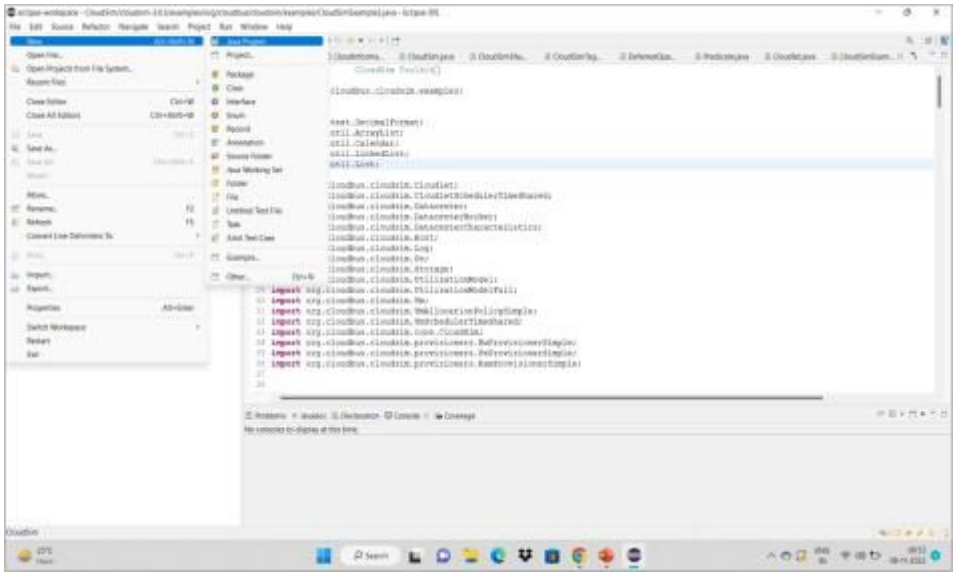
Step 2: Download the CloudSim 3.0.3 zip file from GitHub and extract the folders within a folder in our computer storage.



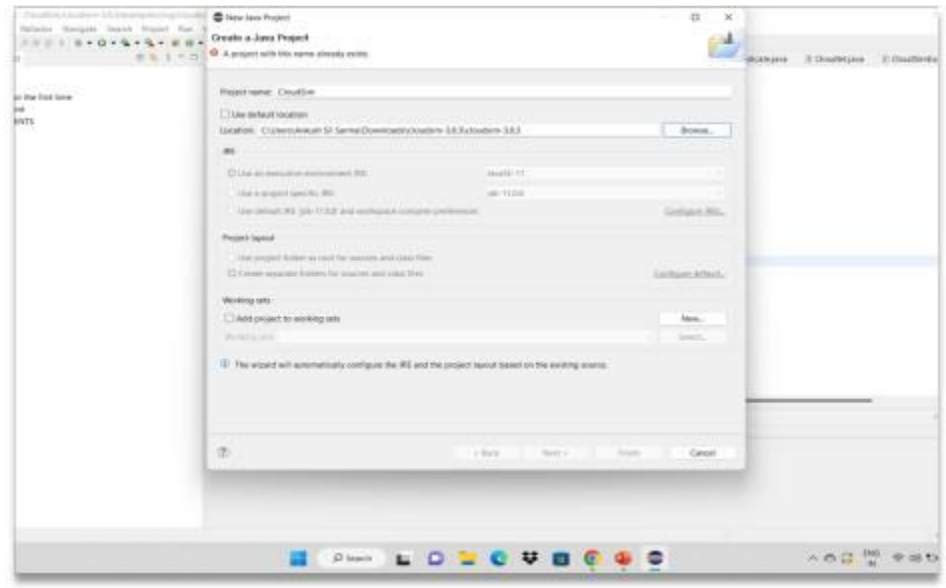
Step 3: Download Apache Commons Math 3.6.1 zip file. It is required as it provides a faster, more accurate, portable alternative to the regular Math and StrictMath classes for large-scale computation in java files.

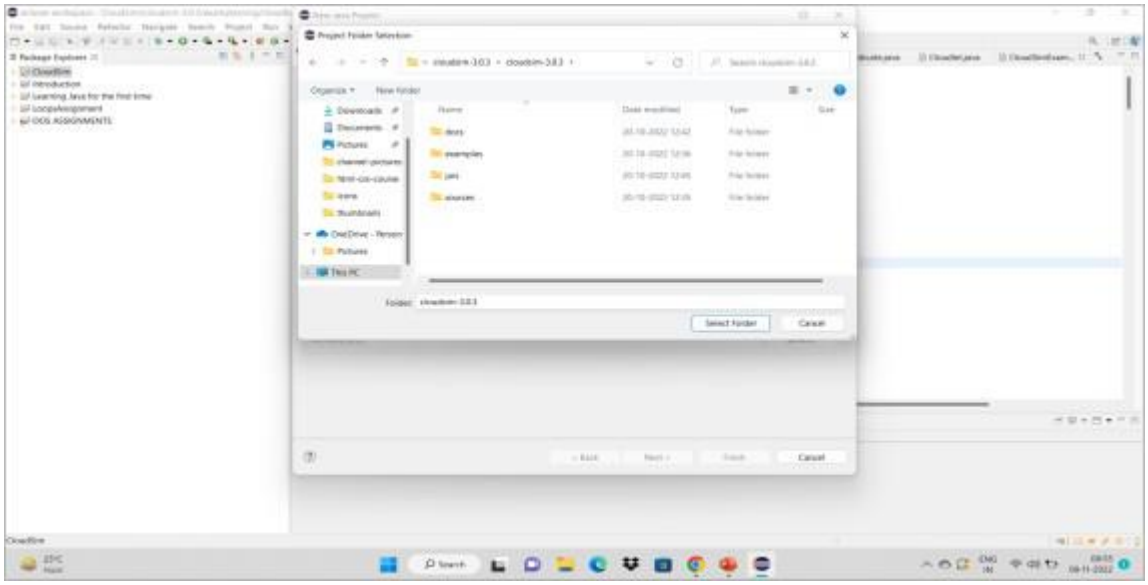


Step 4: Open Eclipse IDE and create a new Java Project.

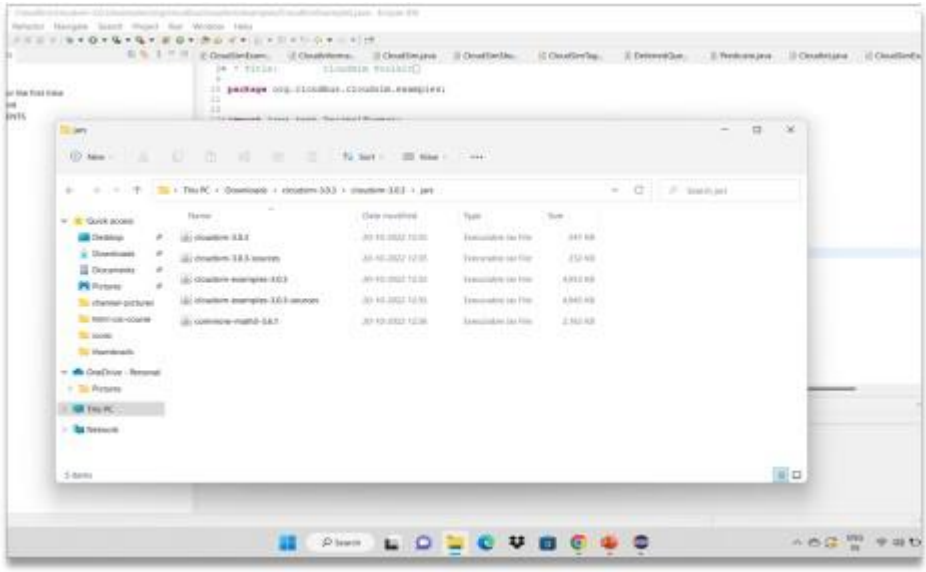


Step 5: The above-mentioned Java Project should be created with the location of the previously downloaded and extracted CloudSim 3.0.3 folder.

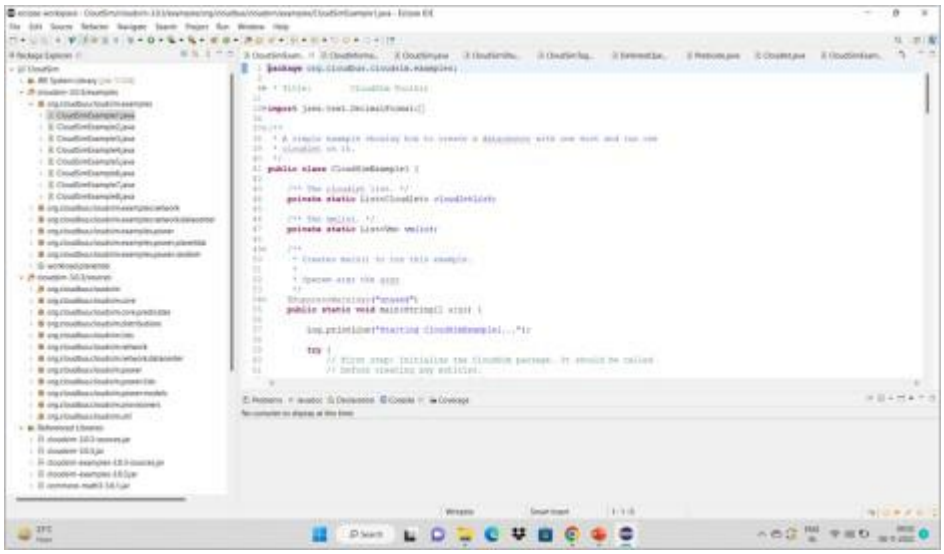




Step 6: The JAR file from the extracted Apache Commons Math 3.6.1 folder needs to be added to the JAR files of CloudSim.

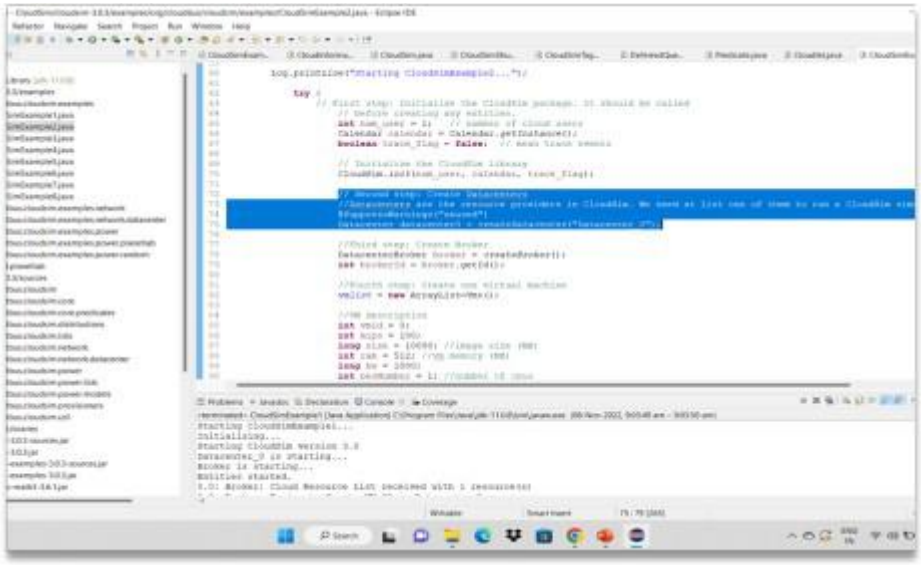


Step 7: Now the CloudSim Environment has been setup in the Eclipse IDE.



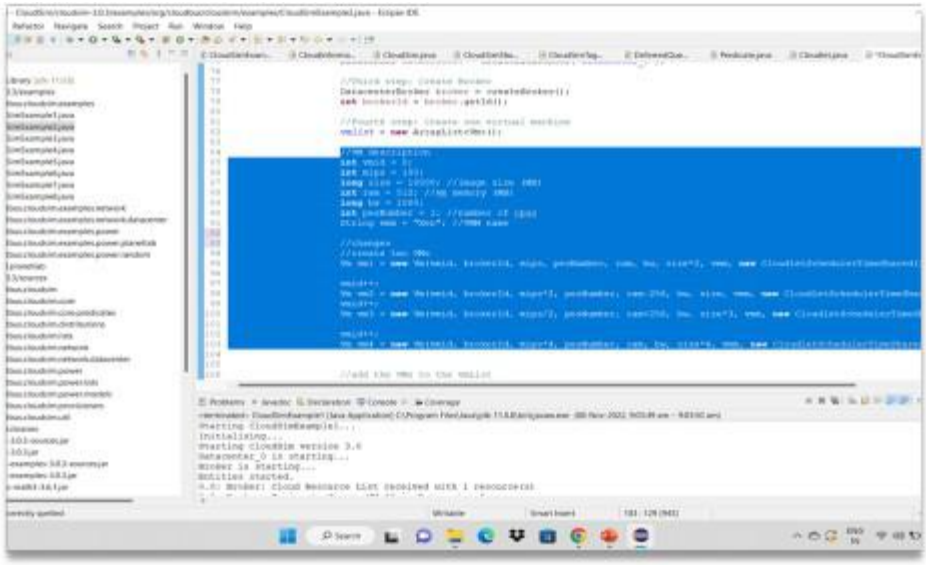
Now Creating Datacenters, Virtual Machines, And Cloudlets In The CloudSim Environment And Checking Its Output

Step 1: We open CloudSimExample2.java from the library on the left and create a data center first.



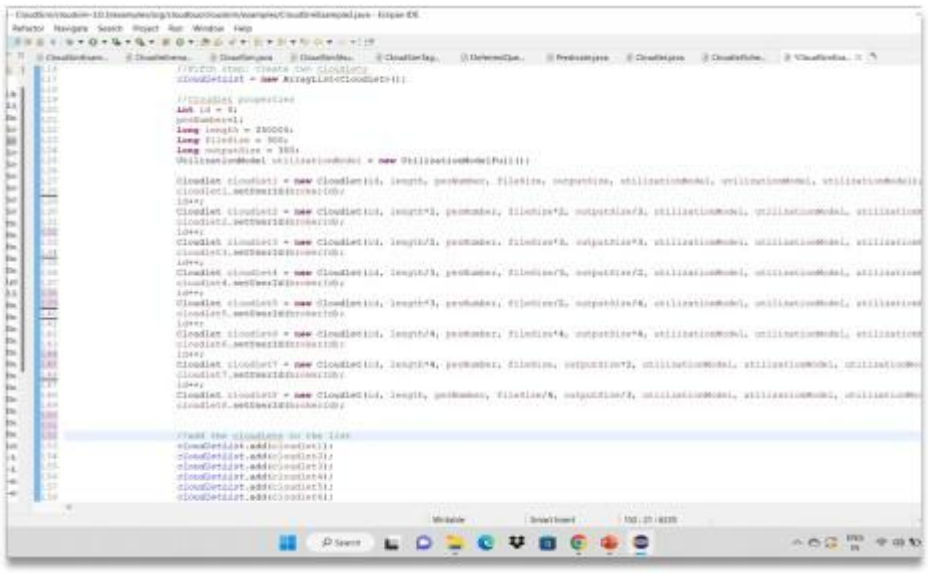
Creating a datacenter in CloudSimExample2.java

Step 2: The initial configuration of one of VM(Virtual machine) is done i.e mips(million instructions), ram(size of RAM), bw(bandwidth), etc. Here we create 4 VMs which are initialized with different configuration and added to vmlist, which is the array list created to store all the 4VMs.



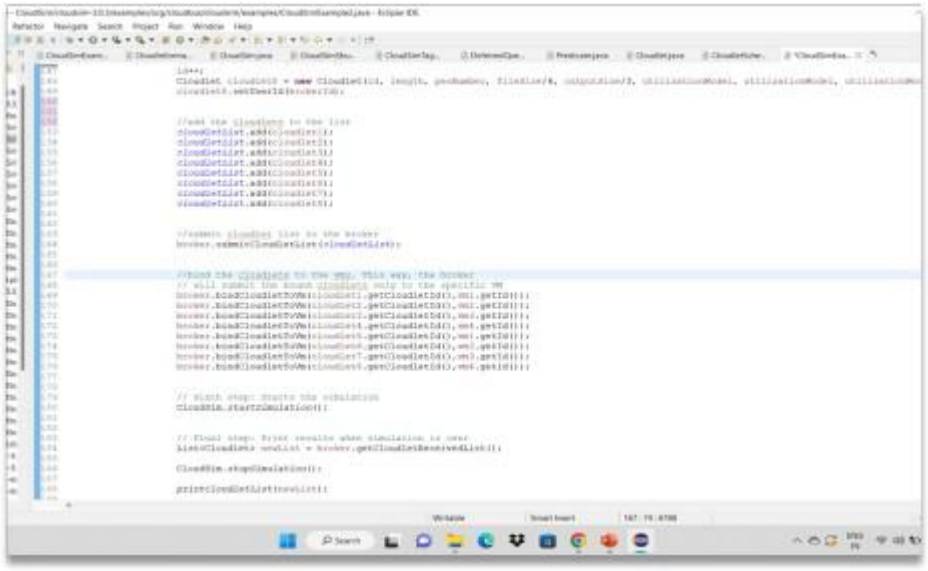
Configuring Virtual Machine

Step 3: We create 8 cloudlets that are initialized with different properties or characteristics i.e id length, outputsize, and filesize.



Creating Cloudlets With Different Characteristics

Step 4: Now the VM-cloudlet binding is done using the broker.



VM-Cloudlet Binding

Below is the code for the above implementation of steps:

Note: You have to run this Java program on your Eclipse IDE after extracting CloudSim 3.0.3 and Apache Commons Math 3.6.1 zip file. Only then will the code run properly.

```
import java.text.DecimalFormat;
```

```
import java.util.ArrayList;
import java.util.Calendar;
import java.util.LinkedList;
import java.util.List;
import org.cloudbus.cloudsim.Cloudlet;
import org.cloudbus.cloudsim.CloudletSchedulerTimeShared;
import org.cloudbus.cloudsim.Datacenter;
import org.cloudbus.cloudsim.DatacenterBroker;
import org.cloudbus.cloudsim.DatacenterCharacteristics;
import org.cloudbus.cloudsim.Host;
import org.cloudbus.cloudsim.Log;
import org.cloudbus.cloudsim.Pe;
import org.cloudbus.cloudsim.Storage;
import org.cloudbus.cloudsim.UtilizationModel;
import org.cloudbus.cloudsim.UtilizationModelFull;
import org.cloudbus.cloudsim.Vm;
import org.cloudbus.cloudsim.VmAllocationPolicySimple;
import org.cloudbus.cloudsim.VmSchedulerTimeShared;
import org.cloudbus.cloudsim.core.CloudSim;
import org.cloudbus.cloudsim.provisioners.BwProvisionerSimple;
import org.cloudbus.cloudsim.provisioners.PeProvisionerSimple;
import org.cloudbus.cloudsim.provisioners.RamProvisionerSimple;

// A simple example showing how to create a data center
// with one host and run eight cloudlets on it
public class CloudSimExample1 {
   // The cloudlet list
   private static List<Cloudlet> cloudletList;

   // The vmlist
   private static List<Vm> vmlist;

   @SuppressWarnings("unused")
   public static void main(String[] args)
   {
      Log.printLine("Starting CloudSimExample2...");

      try {
         // First step: Initialize the CloudSim package.
         // It should be called before creating any
         // entities. number of cloud users
         int num_user = 1;

         // Calendar whose fields have been initialized
         // with the current date and time.
         Calendar calendar = Calendar.getInstance();
```

```
// trace events
boolean trace_flag = false;

CloudSim.init(num_user, calendar, trace_flag);

// Second step: Create Datacenters
// Datacenters are the resource providers in
// CloudSim. We need at list one of them to run
// a CloudSim simulation
Datacenter datacenter0
   = createDatacenter("Datacenter_0");

// Third step: Create Broker
DatacenterBroker broker = createBroker();
int brokerId = broker.getId();

// Fourth step: Create four virtual machine
vmlist = new ArrayList<Vm>();

// VM description
int vmid = 0;
int mips = 1000;
long size = 10000; // image size (MB)
int ram = 512; // vm memory (MB)
long bw = 1000; // bandwidth
int pesNumber = 1; // number of cpus
String vmm = "Xen"; // VMM name

// create 4 VMs
Vm vm1
   = new Vm(vmid, brokerId, mips, pesNumber,
         ram, bw, size, vmm,
         new CloudletSchedulerTimeShared());
vmid++;
Vm vm2 = new Vm(
   vmid, brokerId, mips * 2, pesNumber,
   ram - 256, bw, size * 2, vmm,
   new CloudletSchedulerTimeShared());
vmid++;
Vm vm3 = new Vm(
   vmid, brokerId, mips / 2, pesNumber,
   ram + 256, bw, size * 3, vmm,
   new CloudletSchedulerTimeShared());
vmid++;
Vm vm 4
```

```
    = new Vm(vmid, brokerId, mips * 4,
          pesNumber, ram, bw, size * 4, vmm,
          new CloudletSchedulerTimeShared());
vmid++;

// add the VM to the vmList
vmlist.add(vm1);
vmlist.add(vm2);
vmlist.add(vm3);
vmlist.add(vm4);

// submit vm list to the broker
broker.submitVmList(vmlist);

// Fifth step: Create eight Cloudlets
cloudletList = new ArrayList<Cloudlet>();

// Cloudlet properties
int id = 0;
long length = 400000;
long fileSize = 300;
long outputSize = 300;
UtilizationModel utilizationModel
   = new UtilizationModelFull();

Cloudlet cloudlet1 = new Cloudlet(
   id, length, pesNumber, fileSize, outputSize,
   utilizationModel, utilizationModel,
   utilizationModel);
cloudlet1.setUserId(brokerId);
id++;
Cloudlet cloudlet2 = new Cloudlet(
   id, length * 2, pesNumber, fileSize * 2,
   outputSize / 3, utilizationModel,
   utilizationModel, utilizationModel);
cloudlet2.setUserId(brokerId);
id++;
Cloudlet cloudlet3 = new Cloudlet(
   id, length / 2, pesNumber, fileSize * 3,
   outputSize * 3, utilizationModel,
   utilizationModel, utilizationModel);
cloudlet3.setUserId(brokerId);
Cloudlet cloudlet4 = new Cloudlet(
   id, length / 3, pesNumber, fileSize / 3,
   outputSize / 2, utilizationModel,
   utilizationModel, utilizationModel);
```

```
        cloudlet4.setUserId(brokerId);
        Cloudlet cloudlet5 = new Cloudlet(
           id, length * 3, pesNumber, fileSize / 2,
           outputSize / 4, utilizationModel,
           utilizationModel, utilizationModel);
        cloudlet5.setUserId(brokerId);
        Cloudlet cloudlet6 = new Cloudlet(
           id, length / 4, pesNumber, fileSize * 4,
           outputSize * 4, utilizationModel,
           utilizationModel, utilizationModel);
        cloudlet6.setUserId(brokerId);
        Cloudlet cloudlet7 = new Cloudlet(
           id, length * 4, pesNumber, fileSize,
           outputSize * 2, utilizationModel,
           utilizationModel, utilizationModel);
        cloudlet7.setUserId(brokerId);
        Cloudlet cloudlet8 = new Cloudlet(
           id, length, pesNumber, fileSize / 4,
           outputSize / 3, utilizationModel,
           utilizationModel, utilizationModel);
        cloudlet8.setUserId(brokerId);

        // add the cloudlet to the list
        cloudletList.add(cloudlet1);
        cloudletList.add(cloudlet2);
        cloudletList.add(cloudlet3);
        cloudletList.add(cloudlet4);
        cloudletList.add(cloudlet5);
        cloudletList.add(cloudlet6);
        cloudletList.add(cloudlet7);
        cloudletList.add(cloudlet8);

        // submit cloudlet list to the broker
        broker.submitCloudletList(cloudletList);

        // bind the cloudlets to the vms,This way the
        // broker will submit the bound cloudlets only
        // to the specific VM
        broker.bindCloudletToVm(
           Cloudlet1.getCloudletId(), vm1.getId());
        broker.bindCloudletToVm(
           Cloudlet2.getCloudletId(), vm2.getId());
        broker.bindCloudletToVm(
           Cloudlet3.getCloudletId(), vm3.getId());
        broker.bindCloudletToVm(
           Cloudlet4.getCloudletId(), vm4.getId());
```

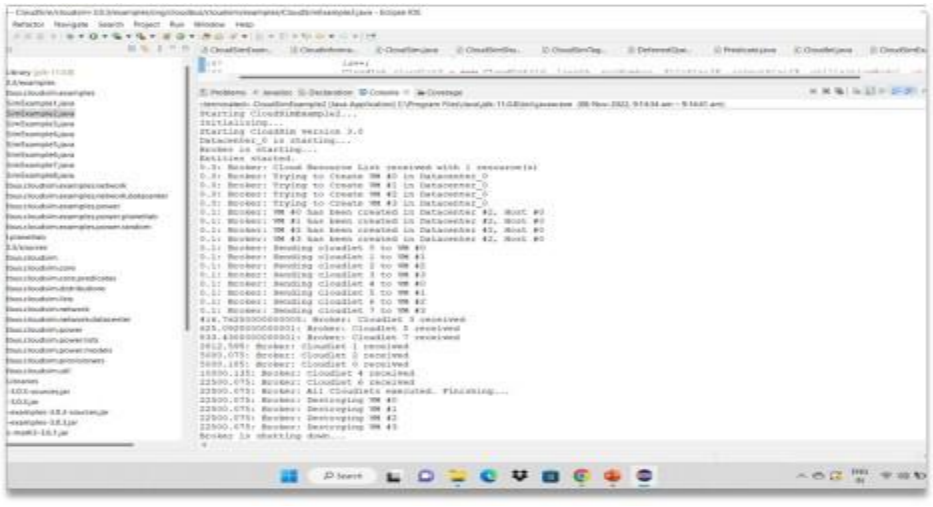
```
        broker.bindCloudletToVm(
           Cloudlet5.getCloudletId(), vm1.getId());
        broker.bindCloudletToVm(
           Cloudlet6.getCloudletId(), vm2.getId());
        broker.bindCloudletToVm(
           Cloudlet7.getCloudletId(), vm3.getId());
        broker.bindCloudletToVm(
           Cloudlet8.getCloudletId(), vm4.getId());

        // Sixth step: Starts the simulation
        CloudSim.startSimulation();

        CloudSim.stopSimulation();

        // Final step: Print results when simulation is
        // over
        List<Cloudlet> newList
           = broker.getCloudletReceivedList();
        printCloudletList(newList);

        Log.printLine("CloudSimExample1 finished!");
    }
    catch (Exception e) {
        e.printStackTrace();
        Log.printLine("Unwanted errors happen");
    }
}

private static Datacenter createDatacenter(String name)
{

    // Here are the steps needed to create a
    // PowerDatacenter:
    // 1. We need to create a list to store
    // our machine
    List<Host> hostList = new ArrayList<Host>();

    // 2. A Machine contains one or more PEs or
    // CPUs/Cores. In this example, it will have only
    // one core.
    List<Pe> peList = new ArrayList<Pe>();

    int mips = 1000;

    // 3. Create PEs and add these into a list.
    // need to store Pe id and MIPS Rating
```

```
peList.add(
   new Pe(0, new PeProvisionerSimple(mips)));

// 4. Create Host with its id and list of PEs and
// add them to the list of machines
int hostId = 0;
int ram = 2048; // host memory (MB)
long storage = 1000000; // host storage
int bw = 10000;

hostList.add(new Host(
   hostId, new RamProvisionerSimple(ram),
   new BwProvisionerSimple(bw), storage, peList,
   new VmSchedulerTimeShared(
      peList))); // This is our machine

// 5. Create a DatacenterCharacteristics object that
// stores the properties of a data center:
// architecture, OS, list of Machines, allocation
// policy: time- or space-shared, time zone and its
// price (G$/Pe time unit).
String arch = "x86"; // system architecture
String os = "Linux"; // operating system
String vmm = "Xen";
double time_zone
   = 10.0; // time zone this resource located
double cost = 3.0; // the cost of using processing
                   // in this resource
double costPerMem = 0.05; // the cost of using
                          // memory in this resource
double costPerStorage
   = 0.001; // the cost of using storage in this
            // resource
double costPerBw
   = 0.0; // the cost of using bw in this resource
LinkedList<Storage> storageList
   = new LinkedList<Storage>(); // we are not
                                // adding SAN
                                // devices by now

DatacenterCharacteristics characteristics
   = new DatacenterCharacteristics(
      arch, os, vmm, hostList, time_zone, cost,
      costPerMem, costPerStorage, costPerBw);

// 6. Finally, we need to create a PowerDatacenter
```

```
    // object.
    Datacenter datacenter = null;
    try {
        datacenter = new Datacenter(
            name, characteristics,
            new VmAllocationPolicySimple(hostList),
            storageList, 0);
    }
    catch (Exception e) {
        e.printStackTrace();
    }

    return datacenter;
}

private static DatacenterBroker createBroker()
{
    DatacenterBroker broker = null;
    try {
        broker = new DatacenterBroker("Broker");
    }
    catch (Exception e) {
        e.printStackTrace();
        return null;
    }
    return broker;
}

private static void
printCloudletList(List<Cloudlet> list)
{
    int size = list.size();
    Cloudlet cloudlet;

    String indent = "    ";
    Log.printLine();
    Log.printLine("========== OUTPUT ==========");
    Log.printLine("Cloudlet ID" + indent + "STATUS"
            + indent + "Data center ID" + indent
            + "VM ID" + indent + "Time" + indent
            + "Start Time" + indent
            + "Finish Time");

    DecimalFormat dft = new DecimalFormat("###.##");
    for (int i = 0; i < size; i++) {
        cloudlet = list.get(i);
```

```
            Log.print(indent + cloudlet.getCloudletId()
                    + indent + indent);

            if (cloudlet.getCloudletStatus()
               == Cloudlet.SUCCESS) {
               Log.print("SUCCESS");

               Log.printLine(
                  indent + indent
                  + cloudlet.getResourceId() + indent
                  + indent + indent + cloudlet.getVmId()
                  + indent + indent
                  + dft.format(
                     cloudlet.getActualCPUTime())
                  + indent + indent
                  + dft.format(
                     cloudlet.getExecStartTime())
                  + indent + indent
                  + dft.format(cloudlet.getFinishTime()));
            }
         }
      }
}
```

Step 5: Output after running the program is shown. One can see all the VM and cloudlets are created and their finish time(performance) is different as their VM and cloudlet configurations were different.



If we change the VM-cloudlet binding, we will get different outputs.

Ex No 6	Find a procedure to transfer the files from one Virtual Machine to another Virtual Machine
Date	

Aim:

To Find a procedure to transfer the files from one virtual machine to another virtual machine.

Steps:

1. You can copy few (or more) lines with copy & paste mechanism.

For this you need to share clipboard between host OS and guest OS, installing Guest Addition on both the virtual machines (probably setting bidirectional and restarting them). You copy from guest OS in the clipboard that is shared with the host OS.

Then you paste from the host OS to the second guest OS.

2. You can enable drag and drop too with the same method (Click on the machine, settings, general, advanced, drag and drop: set to bidirectional)

3. You can have common Shared Folders on both virtual machines and use one of the directory shared as buffer to copy.

Installing Guest Additions you have the possibility to set Shared Folders too. As you put a file in a shared folder from host OS or from guest OS, is immediately visible to the other. (Keep in mind that can arise some problems for date/time of the files when there are different clock settings on the different virtual machines).

If you use the same folder shared on more machines you can exchange files directly copying them in this folder.

4. You can use usual method to copy files between 2 different computer with client-server application. (e.g. scp with sshd active for linux, winscp... you can get some info about SSH servers e.g. here)

You need an active server (sshd) on the receiving machine and a client on the sending machine. Of course you need to have the authorization setted (via password or, better, via an automatic authentication method).

Note: many Linux/Ubuntu distribution install sshd by default: you can see if it is running with pgrep sshd from a shell. You can install with sudo apt-get install openssh-server.

5. You can mount part of the file system of a virtual machine via NFS or SSHFS on the other, or you can share file and directory with Samba. You may find interesting the article Sharing files between guest and host without VirtualBox shared folders with detailed step by step instructions.

You should remember that you are dialling with a little network of machines with different operative systems, and in particular:

- Each virtual machine has its own operative system running on and acts as a physical machine.

- Each virtual machine is an instance of a program owned by an user in the hosting operative system and should undergo the restrictions of the user in the hosting OS.

E.g Let we say that Hastur and Meow are users of the hosting machine, but they did not allow each other to see their directories (no read/write/execute authorization). When each of them run a virtual machine, for the hosting OS those virtual machine are two normal programs owned by Hastur and Meow and cannot see the private directory of the other user. This is a restriction due to the hosting OS. It's easy to overcame it: it's enough to give authorization to read/write/execute to a directory or to chose a different directory in which both users can read/write/execute.

- Windows likes mouse and Linux fingers. :-)

I mean I suggest you to enable Drag & drop to be cosy with the Windows machines and the Shared folders or to be cosy with Linux.

When you will need to be fast with Linux you will feel the need of ssh-keygen and

to Generate once SSH Keys to copy files on/from a remote machine without writing password anymore. In this way it functions bash auto-completion remotely too!

PROCEDURE:

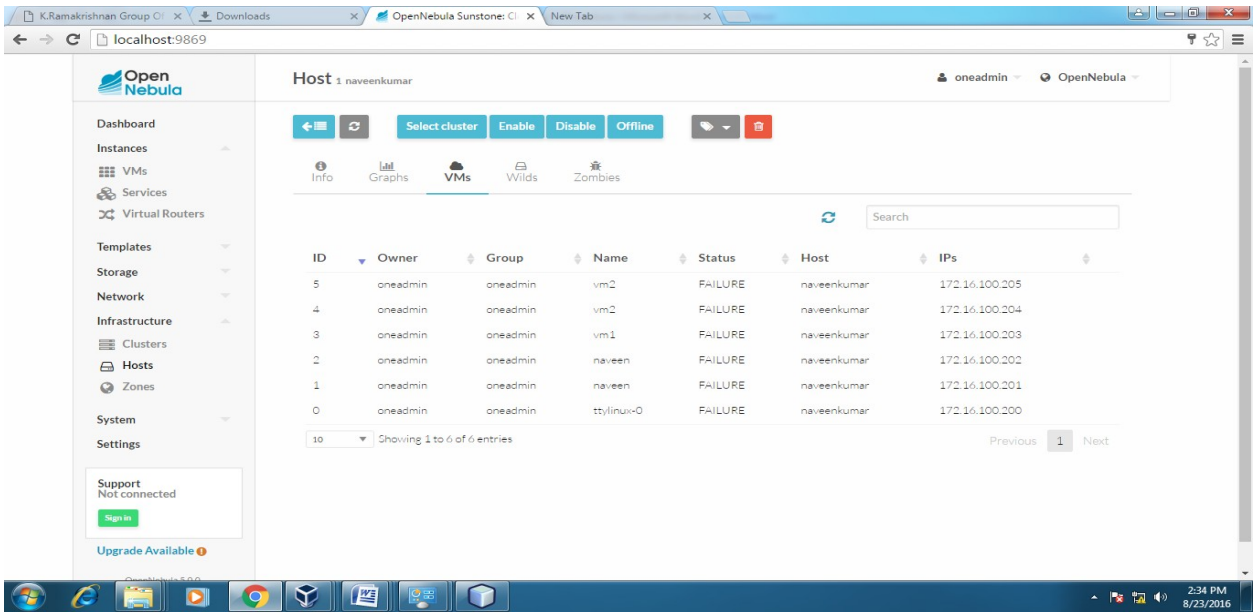
Steps:

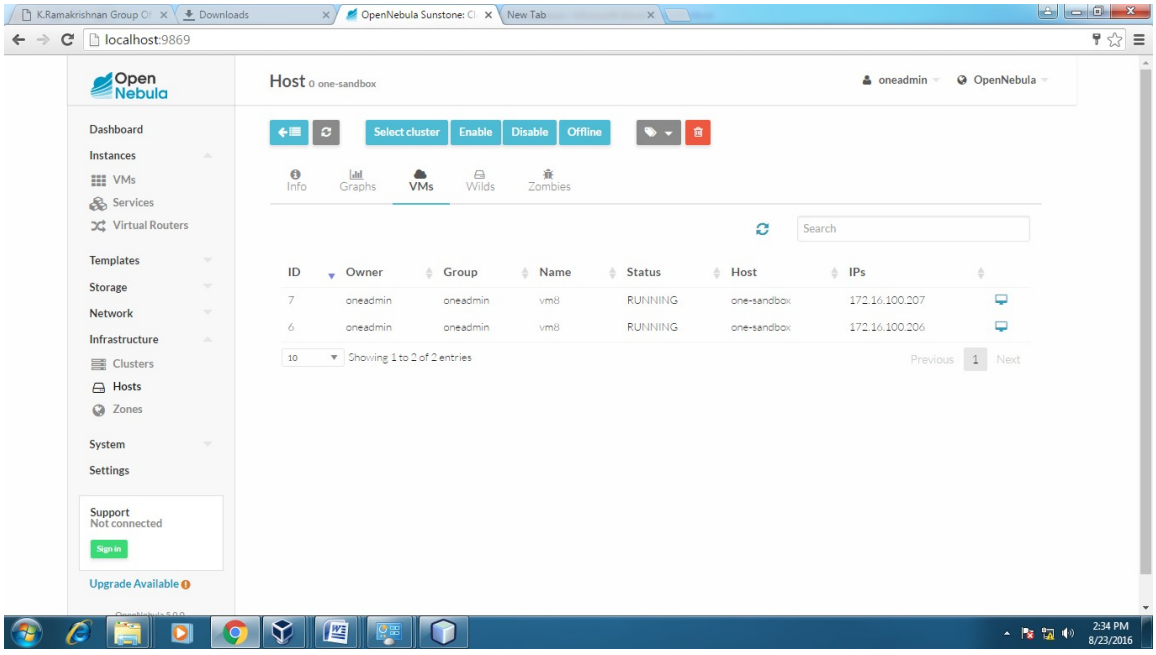
1. Open Browser, type localhost:9869
2. Login using username: oneadmin, password: opennebula
3. Then follow the steps to migrate VMs

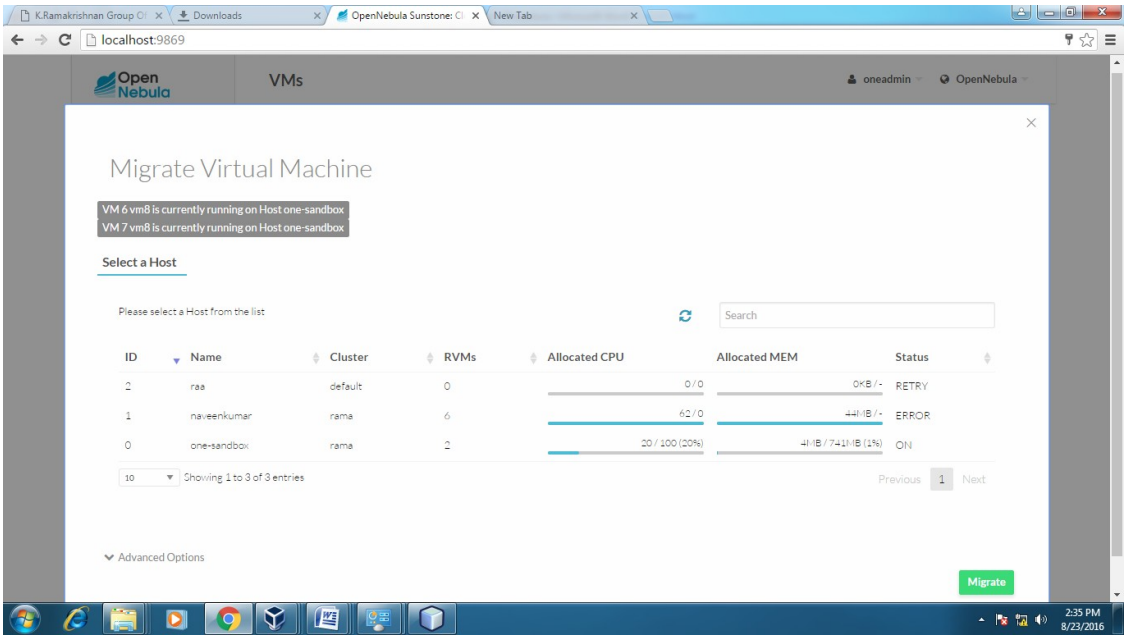
a. Click on infrastructure
b. Select clusters and enter the cluster name
c. Then select host tab, and select all host
d. Then select Vnets tab, and select all vnet
e. Then select datastores tab, and select all datastores
f. And then choose host under infrastructure tab
g. Click on + symbol to add new host, name the host then click on create.

4. on instances, select VMs to migrate then follow the stpes

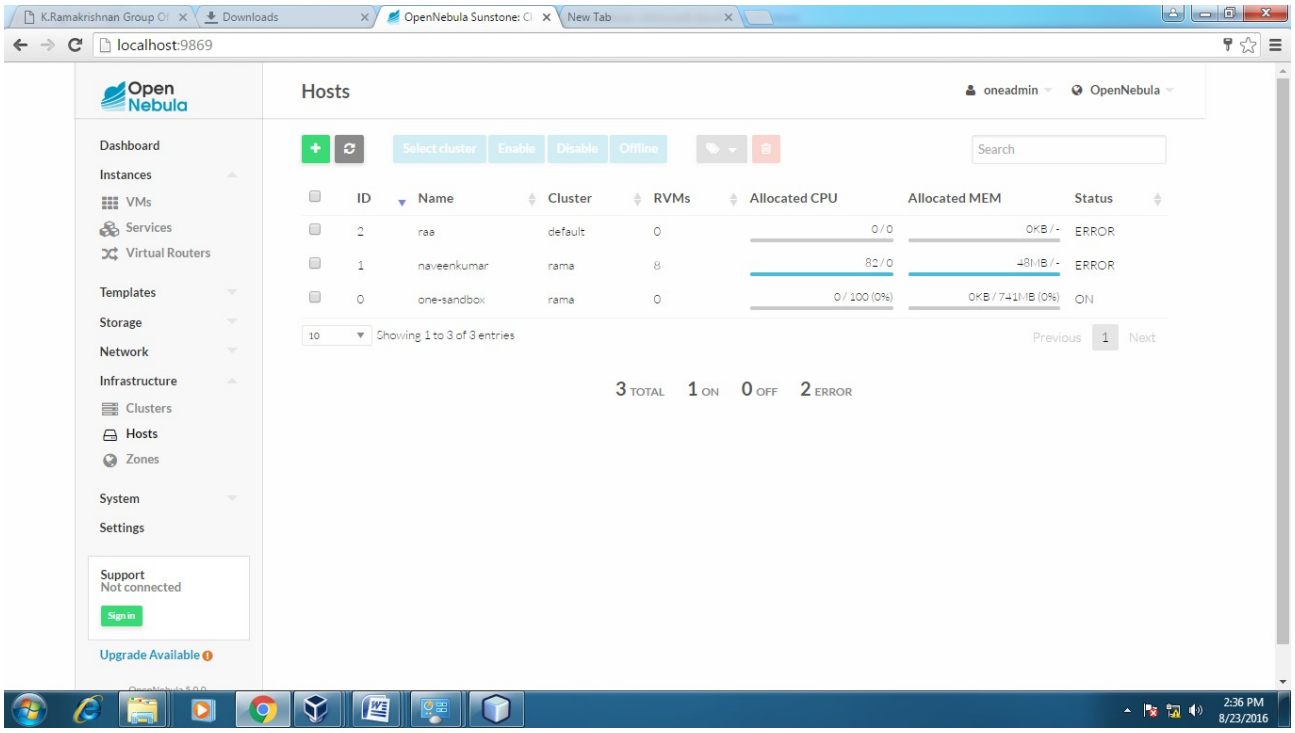
a. Click on 8th icon ,the drop down list display
b. Select migrate on that ,the popup window display
c. On that select the target host to migrate then click on migrate.


K.Ramakrishnan Group Of
Downloads
OpenNebula Sunstone: Cl
New Tab
localhost:9869
Open Nebula
Host 1 naveenkumar
oneadmin
OpenNebula
Dashboard
Instances
VMs
Services
Virtual Routers
Templates
Storage
Network
Infrastructure
Clusters
Hosts
Zones
System
Settings
Support
Not connected
Sign in
Upgrade Available
Select cluster
Enable
Disable
Offline
Info
Graphs
VMs
Wilds
Zombies
Search
ID Owner Group Name Status Host IPs
5 oneadmin oneadmin vm2 FAILURE naveenkumar 172.16.100.205
4 oneadmin oneadmin vm2 FAILURE naveenkumar 172.16.100.204
3 oneadmin oneadmin vm1 FAILURE naveenkumar 172.16.100.203
2 oneadmin oneadmin naveen FAILURE naveenkumar 172.16.100.202
1 oneadmin oneadmin naveen FAILURE naveenkumar 172.16.100.201
0 oneadmin oneadmin ttylinux-0 FAILURE naveenkumar 172.16.100.200
10
Showing 1 to 6 of 6 entries
Previous
1
Next
2:34 PM
8/23/2016


K.Ramakrishnan Group Of
Downloads
OpenNebula Sunstone: Cl
New Tab
localhost:9869
Open Nebula
Host 0 one-sandbox
oneadmin
OpenNebula
Dashboard
Instances
VMs
Services
Virtual Routers
Templates
Storage
Network
Infrastructure
Clusters
Hosts
Zones
System
Settings
Support
Not connected
Sign in
Upgrade Available
Select cluster
Enable
Disable
Offline
Info
Graphs
VMs
Wilds
Zombies
Search
ID Owner Group Name Status Host IPs
7 oneadmin oneadmin vm8 RUNNING one-sandbox 172.16.100.207
6 oneadmin oneadmin vm8 RUNNING one-sandbox 172.16.100.206
10
Showing 1 to 2 of 2 entries
Previous
1
Next
2:34 PM
8/23/2016



After Migration



Ex No 7	Find a procedure to launch Virtual Machine using try stack (Online Openstack Demo Version)
Date	

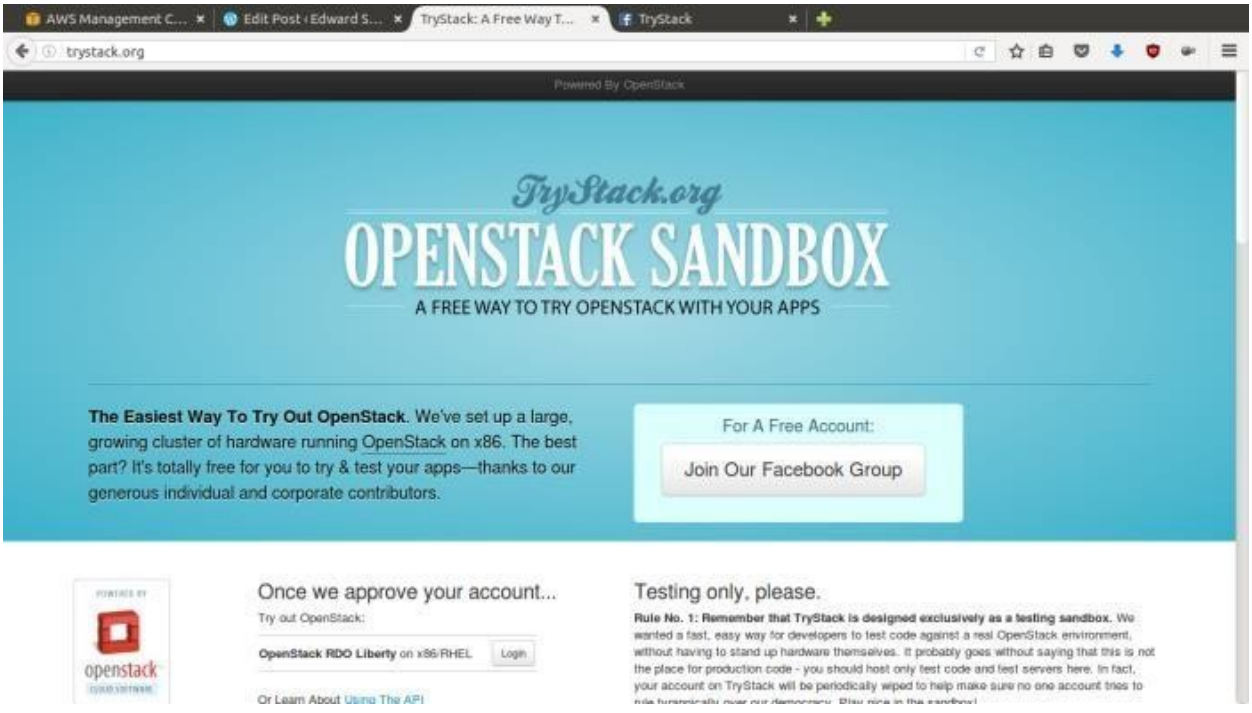
Aim:

To Find a procedure to launch virtual machine using trystack.

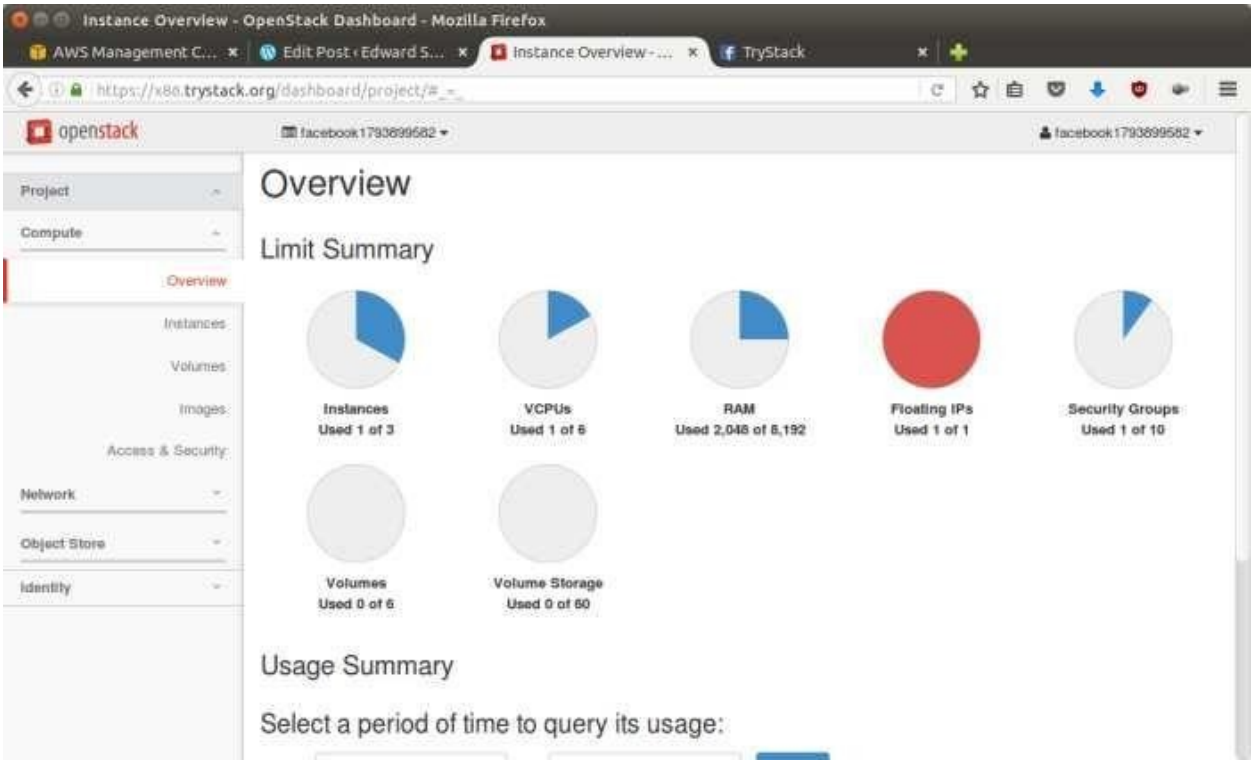
OpenStack is an open-source software cloud computing platform.

OpenStack is primarily used for deploying an infrastructure as a service (IaaS) solution like Amazon Web Service (AWS). In other words, you can make your own AWS by using OpenStack. If you want to try out OpenStack, TryStack is the easiest and free way to do it.

In order to try OpenStack in TryStack, you must register yourself by joining TryStack Facebook Group. The acceptance of group needs a couple days because it's approved manually. After you have been accepted in the TryStack Group, you can log in TryStack.



I assume that you already join to the Facebook Group and login to the dashboard. After you log in to the TryStack, you will see the Compute Dashboard like:



OpenStack Compute Dashboard

Step 1: Create Network

- Network? Yes, the network in here is our own local network. So, your instances will be not mixed up with the others. You can imagine this as your own LAN (Local Area Network) in the cloud.
- Go to Network > Networks and then click Create Network.
- In Network tab, fill Network Name for example internal and then click Next.
- In Subnet tab,
- Fill Network Address with appropriate CIDR, for example 192.168.1.0/24. Use private network CIDR block as the best practice.
- Select IP Version with appropriate IP version, in this case IPv4.
- Click Next.
- In Subnet Details tab, fill DNS Name Servers with 8.8.8.8 (Google DNS) and then click Create.

Step 2: Create Instance

- Now, we will create an instance. The instance is a virtual machine in the cloud, like AWS EC2. You need the instance to connect to the network that we just created in the previous step.
- Go to Compute > Instances and then click Launch Instance.
- In Details tab,
- Fill Instance Name, for example Ubuntu 1.
- Select Flavor, for example m1.medium.
- Fill Instance Count with 1.
- Select Instance Boot Source with Boot from Image.
- Select Image Name with Ubuntu 14.04 amd64 (243.7 MB) if you want install Ubuntu
- 14.04 in your virtual machine.
- In Access & Security tab,
- Click [+] button of Key Pair to import key pair. This key pair is a public and private key that we will use to connect to the instance from our machine.
- In Import Key Pair dialog,
- Fill Key Pair Name with your machine name (for example Edward-Key).
- Fill Public Key with your SSH public key (usuallyin~/.ssh/id_rsa.pub). See description in Import Key Pair dialog box for more information. If you are using Windows, you can use Puttygen to generate key pair.
- Click Import key pair.
- In Security Groups, mark/check default.
- In Networking tab, In Selected Networks, select network that have been created in Step 1, for example internal.
- Click Launch.
- If you want to create multiple instances, you can repeat step 1-5. I created one more instance with instance name Ubuntu 2.

Step 3: Create Router

- I guess you already know what router is. In the step 1, we created our network, but it is isolated. It doesn't connect to the internet. To make our network has an internet connection, we need a router that running as the gateway to the internet.
- Go to Network > Routers and then click Create Router.
- Fill Router Name for example router1 and then click Create router.
- Click on your router name link, for example router1, Router Details page.
- Click Set Gateway button in upper right:
- Select External networks with external.
- Then OK.
- Click Add Interface button.
- Select Subnet with the network that you have been created in Step 1.
- Click Add interface.
- Go to Network > Network Topology. You will see the network topology. In the example, there are two network, i.e. external and internal, those are bridged by a router. There are instances those are joined to internal network.

Step 4: Configure Floating IP Address

- Floating IP address is public IP address. It makes your instance is accessible from the internet. When you launch your instance, the instance will have a private network IP, but no public IP. In OpenStack, the public Ips is collected in a pool and managed by admin (in our case is TryStack). You need to request a public (floating) IP address to be assigned to your instance.
- Go to Compute > Instance.
- In one of your instances, click More > Associate Floating IP.
- In IP Address, click Plus [+].
- Select Pool to external and then click Allocate IP.
- Click Associate.
- Now you will get a public IP, e.g. 8.21.28.120, for your instance.

Step 5: Configure Access & Security

- OpenStack has a feature like a firewall. It can whitelist/blacklist your in/out connection. It is called Security Group.
- Go to Compute > Access & Security and then open Security Groups tab.
- In default row, click Manage Rules.
- Click Add Rule, choose ALL ICMP rule to enable ping into your instance, and then click Add.
- Click Add Rule, choose HTTP rule to open HTTP port (port 80), and then click Add.
- Click Add Rule, choose SSH rule to open SSH port (port 22), and then click Add.
- You can open other ports by creating new rules.
- Step 6: SSH to Your Instance
- Now, you can SSH your instances to the floating IP address that you got in the step 4. If you are using Ubuntu image, the SSH user will be ubuntu.

Ex No 8	Install Hadoop single node cluster and run simple applications like wordcount.
Date	

Install Hadoop: Setting up a Single Node Hadoop Cluster

You must have got a theoretical idea about Hadoop, HDFS and its architecture. But to get Hadoop Certified you need good hands-on knowledge. I hope you would have liked our previous blog on ***HDFS Architecture***, now I will take you through the practical knowledge about Hadoop and HDFS. The first step forward is to install Hadoop.

There are two ways to install Hadoop, i.e. **Single node** and **Multi-node**.

A single node cluster means only one DataNode running and setting up all the NameNode, DataNode, ResourceManager, and NodeManager on a single machine. This is used for studying and testing purposes. For example, let us consider a sample data set inside the healthcare industry. So, for testing whether the Oozie jobs have scheduled all the processes like collecting, aggregating, storing, and processing the data in a proper sequence, we use a single node cluster. It can easily and efficiently test the sequential workflow in a smaller environment as compared to large environments which contain terabytes of data distributed across hundreds of machines.

While in a **Multi-node cluster**, there are more than one DataNode running and each DataNode is running on different machines. The multi-node cluster is practically used in organizations for analyzing Big Data. Considering the above example, in real-time when we deal with petabytes of data, it needs to be distributed across hundreds of machines to be processed. Thus, here we use a multi-node cluster.

Prerequisites

- *VIRTUAL BOX*: it is used for installing the operating system on it.
- *OPERATING SYSTEM*: You can install Hadoop on Linux-based operating systems. Ubuntu and CentOS are very commonly used. In this tutorial, we are using CentOS.
- *JAVA*: You need to install the Java 8 package on your system.
- *HADOOP*: You require Hadoop 2.7.3 package.

Install Hadoop

Step 1: Download the Java 8 Package. Save this file in your home directory.

Step 2: Extract the Java Tar File.

***Command*:** tar -xvf jdk-8u101-linux-i586.tar.gz

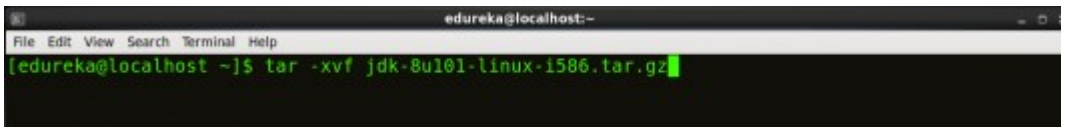


Fig: Hadoop Installation – Extracting Java Files

Step 3: Download the Hadoop 2.7.3 Package.
***Command*:** wget https://archive.apache.org/dist/hadoop/core/hadoop-2.7.3/hadoop-2.7.3.tar.gz

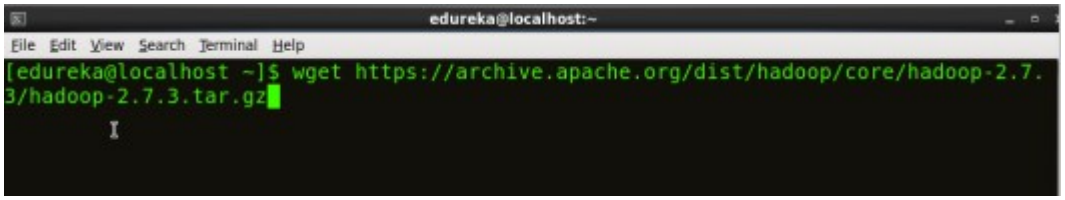


Fig: Hadoop Installation – Downloading Hadoop

Step 4: Extract the Hadoop tar File.
Command: tar -xvf hadoop-2.7.3.tar.gz

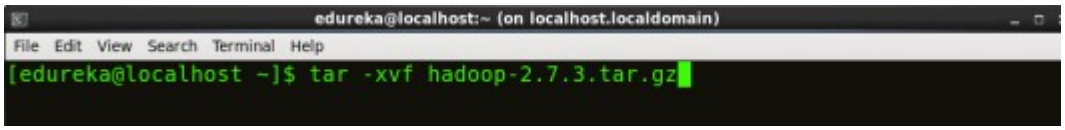
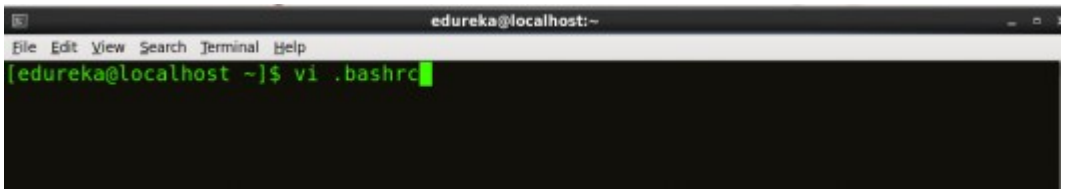


Fig: Hadoop Installation – Extracting Hadoop Files

Step 5: Add the Hadoop and Java paths in the bash file (.bashrc).
Open**. bashrc** file. Now, add Hadoop and Java Path as shown below.

Learn more about the Hadoop Ecosystem and its tools with the Hadoop Certification.

***Command*:** vi .bashrc



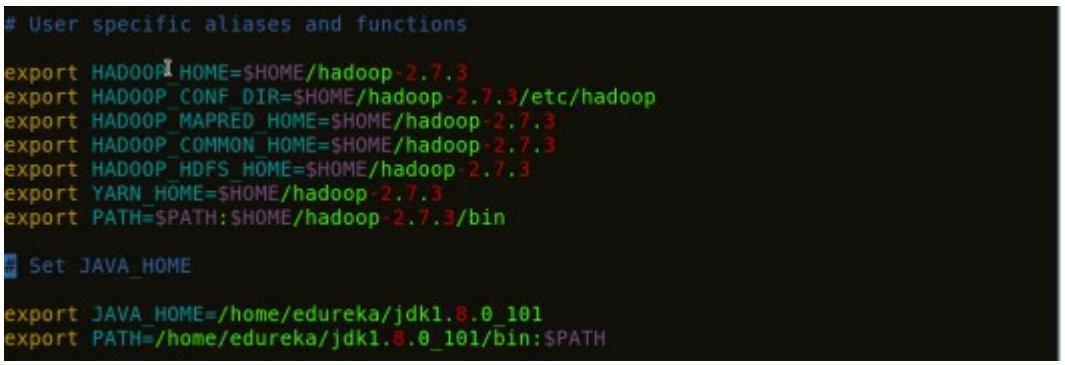


Fig: Hadoop Installation – Setting Environment Variable

Then, save the bash file and close it.

For applying all these changes to the current Terminal, execute the source command.

***Command*:** source .bashrc

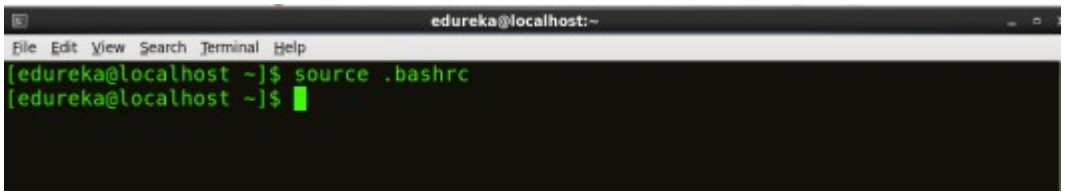


Fig: Hadoop Installation – Refreshing environment variables

To make sure that Java and Hadoop have been properly installed on your system and can be accessed through the Terminal, execute the java -version and hadoop version commands.

***Command*:** java -version

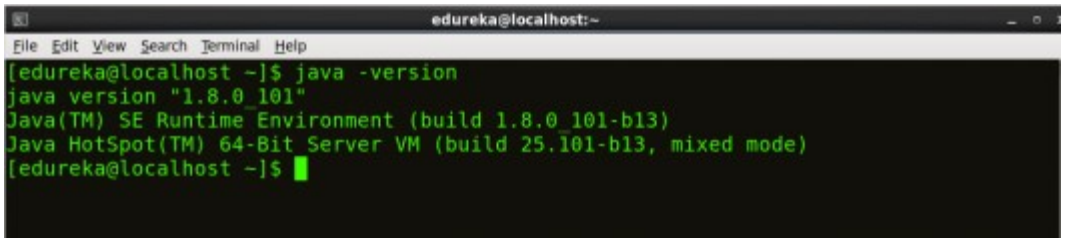


Fig: Hadoop Installation – Checking Java Version

***Command*:** hadoop version

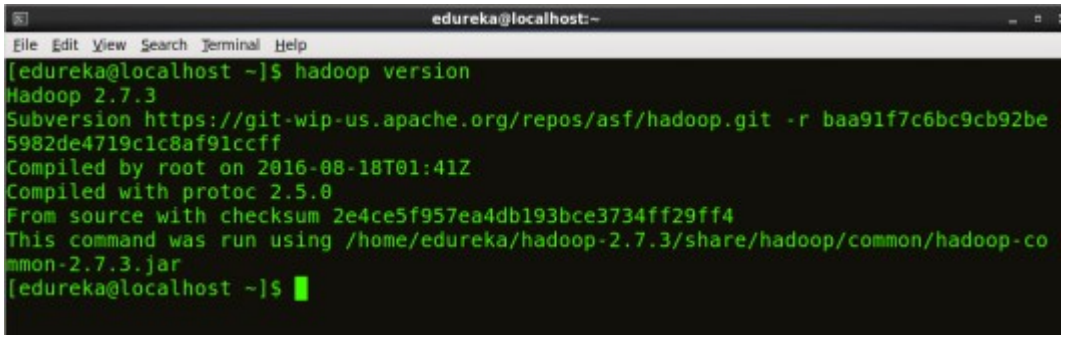


Fig: Hadoop Installation – Checking Hadoop Version

Step 6: Edit the **Hadoop Configuration files**.
Command: cd hadoop-2.7.3/etc/hadoop/

Command: ls

All the Hadoop configuration files are located in **hadoop-2.7.3/etc/hadoop** directory as you can see in the snapshot below:

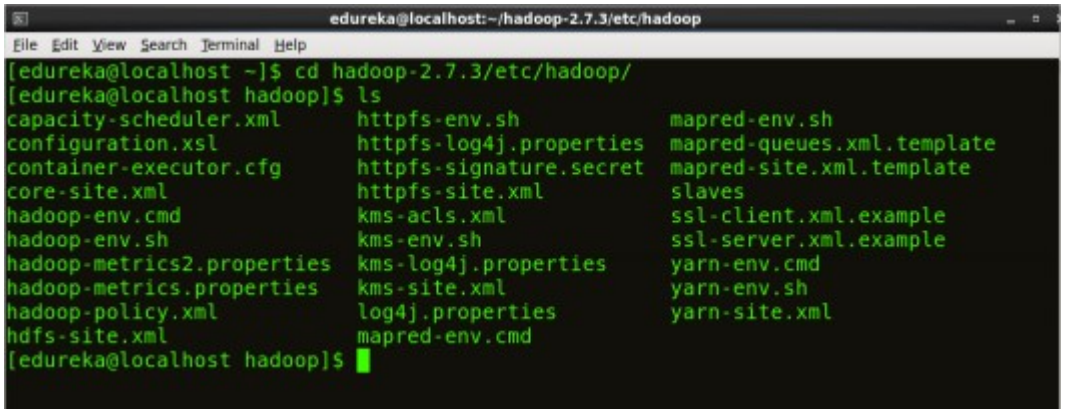


Fig: Hadoop Installation – Hadoop Configuration Files

Step 7: Open *core-site.xml* and edit the property mentioned below inside configuration tag:
core-site.xml informs Hadoop daemon where NameNode runs in the cluster. It contains configuration settings of Hadoop core such as I/O settings that are common to HDFS & MapReduce.

***Command*:** vi core-site.xml

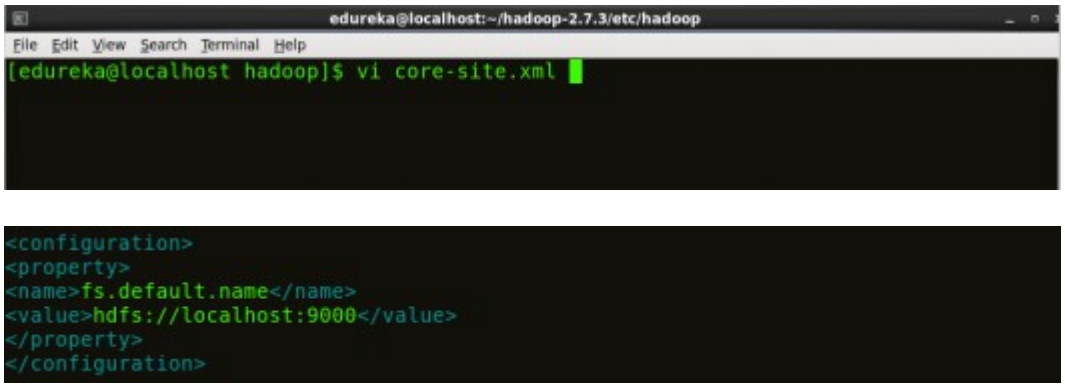


Fig: Hadoop Installation – Configuring core-site.xml

Step 8: Edit *hdfs-site.xml* and edit the property mentioned below inside configuration tag:
hdfs-site.xml contains configuration settings of HDFS daemons (i.e. NameNode, DataNode, Secondary NameNode). It also includes the replication factor and block size of HDFS.

***Command*:** vi hdfs-site.xml

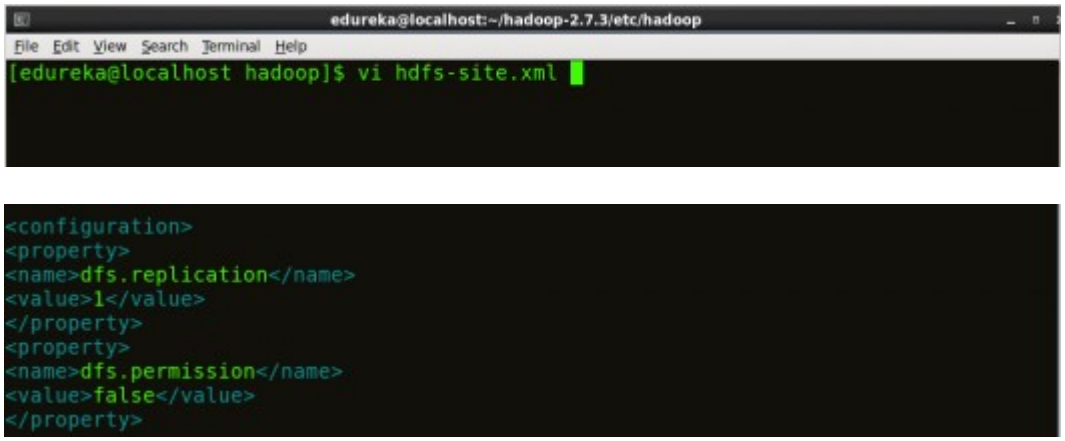


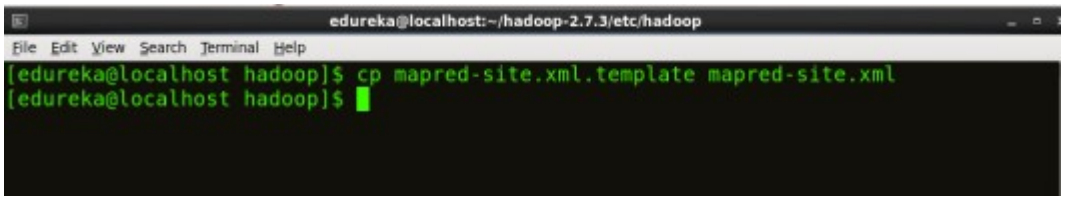
Fig: Hadoop Installation – Configuring hdfs-site.xml

Step 9: Edit the *mapred-site.xml* file and edit the property mentioned below inside configuration tag:
mapred-site.xml contains configuration settings of MapReduce application like number of JVM that can run in parallel, the size of the mapper and the reducer process, CPU cores available for a process, etc.

In some cases, mapred-site.xml file is not available. So, we have to create the mapred-site.xml file using mapred-site.xml template.

***Command*:** cp mapred-site.xml.template mapred-site.xml

***Command*:** vi mapred-site.xml.



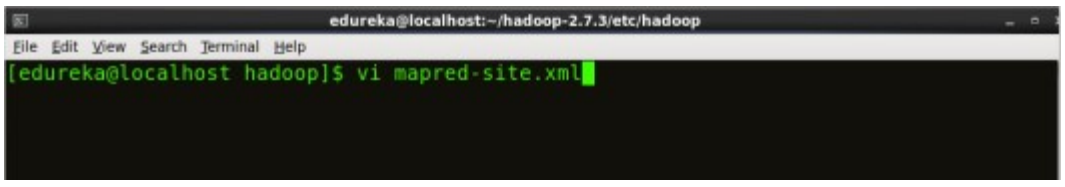


Fig: Hadoop Installation – Configuring mapred-site.xml

Step 10: Edit *yarn-site.xml* and edit the property mentioned below inside configuration tag: *yarn-site.xml* contains configuration settings of ResourceManager and NodeManager like application memory management size, the operation needed on program & algorithm, etc.

You can even check out the details of Big Data with the Azure Data Engineering Certification in Hyderabad.

***Command*:** vi yarn-site.xml

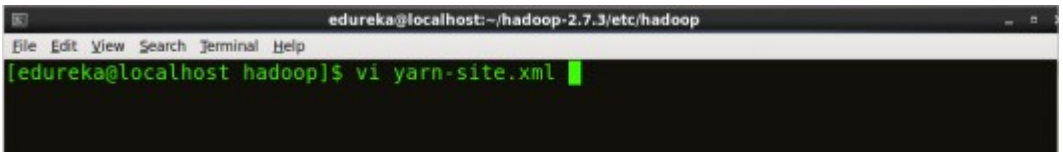


Fig: Hadoop Installation – Configuring yarn-site.xml

Step 11: **Edit *hadoop-env.sh* and add the Java Path as mentioned below:**

hadoop-env.sh contains the environment variables that are used in the script to run Hadoop like Java home path, etc.

***Command*:** vi hadoop–env.sh

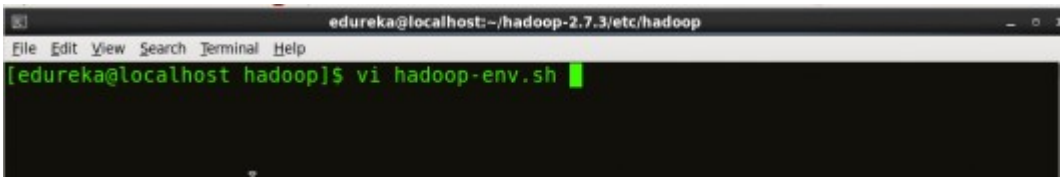


Fig: Hadoop Installation – Configuring hadoop-env.sh

Step 12: Go to Hadoop home directory and format the NameNode.
***Command*:** cd

***Command*:** cd hadoop-2.7.3

***Command*:** bin/hadoop namenode -format

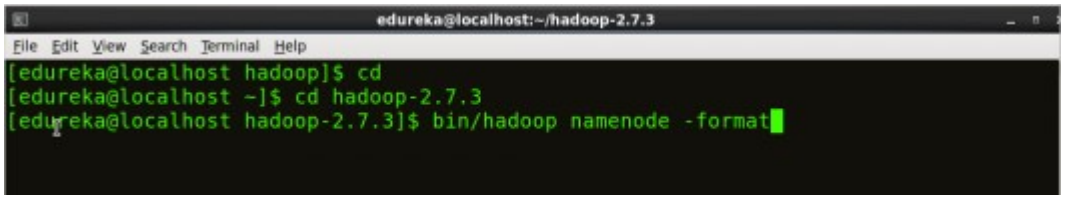


Fig: Hadoop Installation – Formatting NameNode

This formats the HDFS via NameNode. This command is only executed for the first time. Formatting the file system means initializing the directory specified by the dfs.name.dir variable.

Never format, up and running Hadoop filesystem. You will lose all your data stored in the HDFS.

Step 13: Once the NameNode is formatted, go to hadoop-2.7.3/sbin directory and start all the daemons.
Command: cd hadoop-2.7.3/sbin

Either you can start all daemons with a single command or do it individually.

Command: ./start-all.sh

The above command is a combination of ***start-dfs.sh, start-yarn.sh*** & ***mr-jobhistory-daemon.sh***

Or you can run all the services individually as below:

Start NameNode:
The NameNode is the centerpiece of an HDFS file system. It keeps the directory tree of all files stored in the HDFS and tracks all the file stored across the cluster.

Command: ./hadoop-daemon.sh start namenode

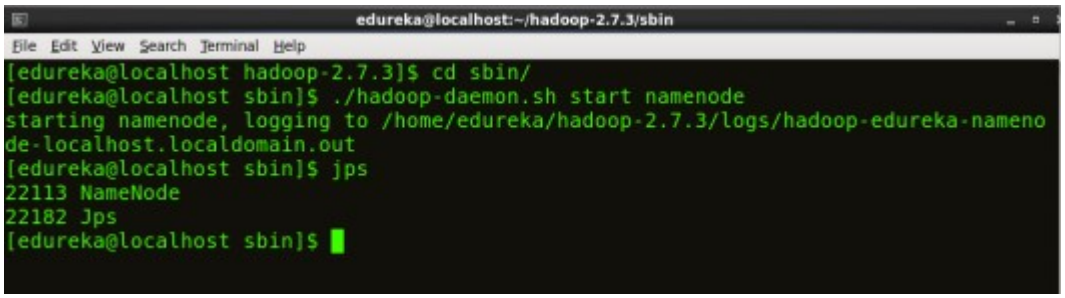


Fig: Hadoop Installation – Starting NameNode

Start DataNode:
On startup, a DataNode connects to the Namenode and it responds to the requests from the Namenode for different operations.

Command: ./hadoop-daemon.sh start datanode

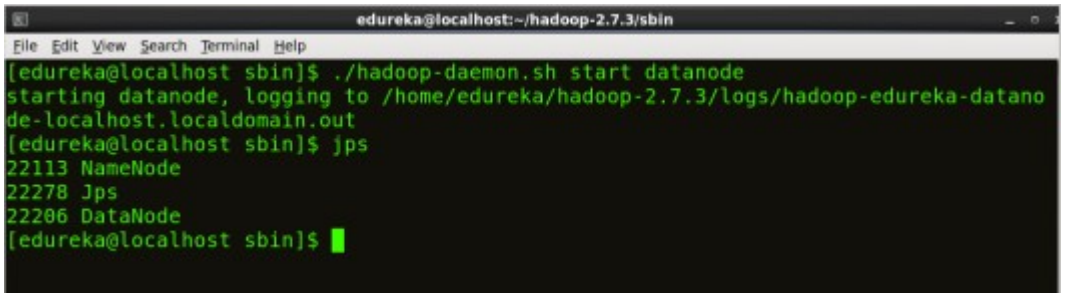


Fig: Hadoop Installation – Starting DataNode

Start ResourceManager:
ResourceManager is the master that arbitrates all the available cluster resources and thus helps in managing the distributed applications running on the YARN system. Its work is to manage each NodeManagers and the each application's ApplicationMaster.

Command: ./yarn-daemon.sh start resourcemanager

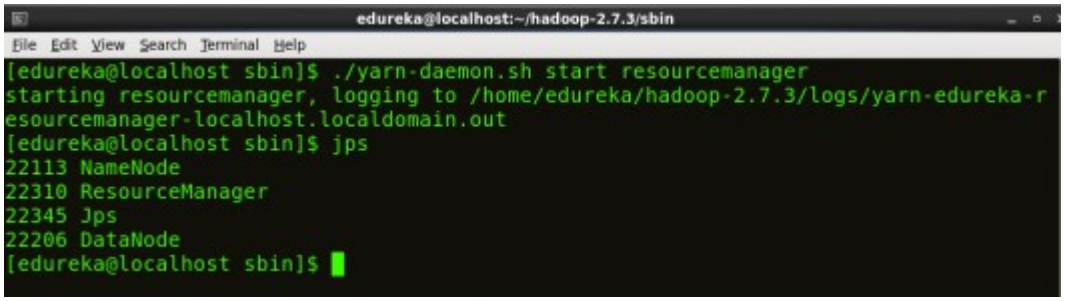


Fig: Hadoop Installation – Starting ResourceManager

Start NodeManager:
The NodeManager in each machine framework is the agent which is responsible for managing containers, monitoring their resource usage and reporting the same to the ResourceManager.

Command: ./yarn-daemon.sh start nodemanager

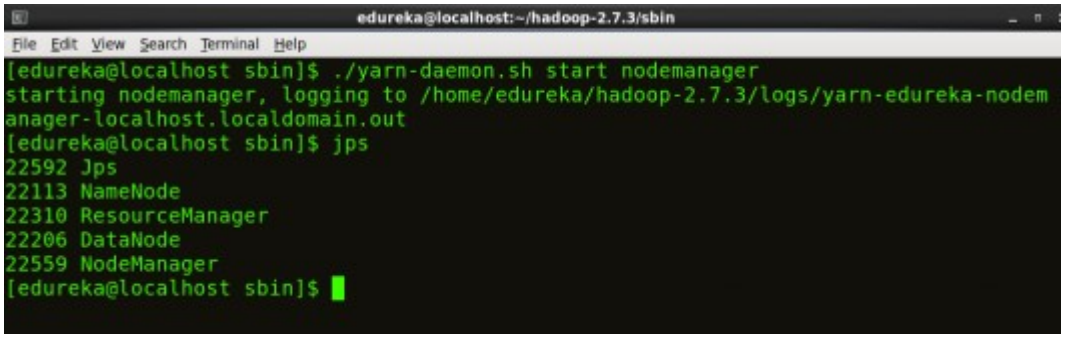


Fig: Hadoop Installation – Starting NodeManager

Start JobHistoryServer:
JobHistoryServer is responsible for servicing all job history related requests from client.

***Command*:** ./mr-jobhistory-daemon.sh start historyserver

Step 14: To check that all the Hadoop services are up and running, run the below command.
Command: jps

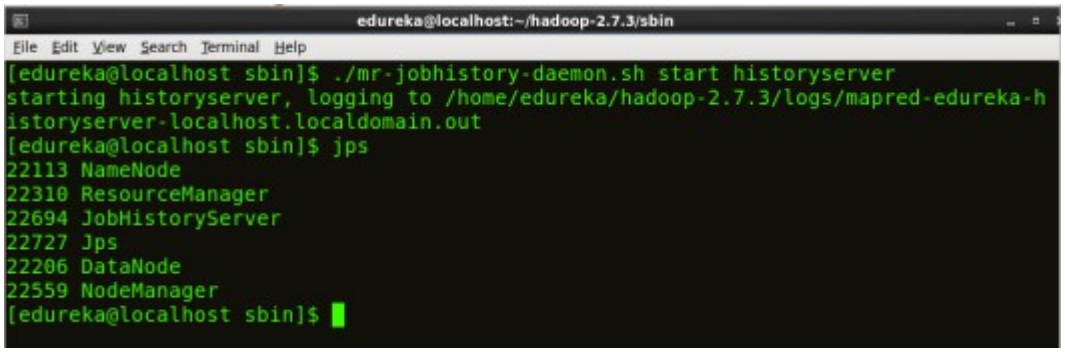


Fig: Hadoop Installation – Checking Daemons

Step 15: Now open the Mozilla browser and go to **localhost:50070/dfshealth.html** to check the NameNode interface.

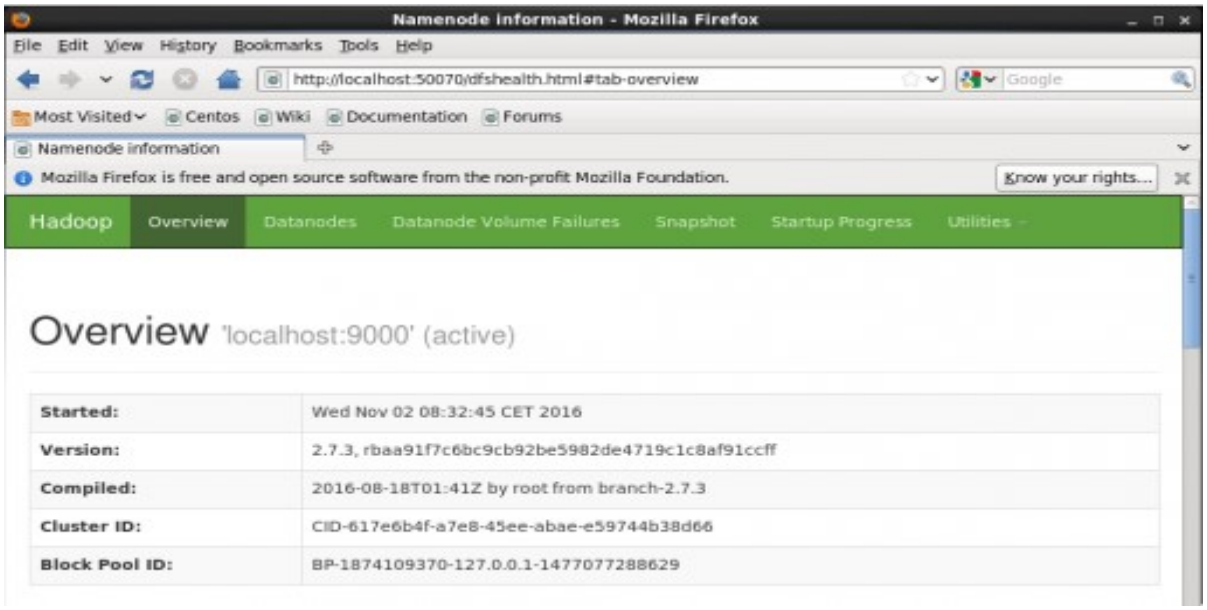


Fig: Hadoop Installation – Starting WebUI

MapReduce Word Count Example

In MapReduce word count example, we find out the frequency of each word. Here, the role of Mapper is to map the keys to the existing values and the role of Reducer is to aggregate the keys of common values. So, everything is represented in the form of Key-value pair.

Pre-requisite

- Java Installation - Check whether the Java is installed or not using the following command.
  java -version
- Hadoop Installation - Check whether the Hadoop is installed or not using the following command.
  hadoop version

File: WC_Mapper.java

```
1. package com.javatpoint;
2.
3. import java.io.IOException;
4. import java.util.StringTokenizer;
5. import org.apache.hadoop.io.IntWritable;
6. import org.apache.hadoop.io.LongWritable;
7. import org.apache.hadoop.io.Text;
8. import org.apache.hadoop.mapred.MapReduceBase;
9. import org.apache.hadoop.mapred.Mapper;
```

```
10. import org.apache.hadoop.mapred.OutputCollector;
11. import org.apache.hadoop.mapred.Reporter;
12. public class WC_Mapper extends MapReduceBase implements Mapper<LongWritable,Text,Text,IntWritable>{
13.     private final static IntWritable one = new IntWritable(1);
14.     private Text word = new Text();
15.     public void map(LongWritable key, Text value,OutputCollector<Text,IntWritable> output,
16.             Reporter reporter) throws IOException{
17.         String line = value.toString();
18.         StringTokenizer  tokenizer = new StringTokenizer(line);
19.         while (tokenizer.hasMoreTokens()){
20.             word.set(tokenizer.nextToken());
21.             output.collect(word, one);
22.         }
23.     }
24.
25. }
```

File: WC_Reducer.java

```
1.  package com.javatpoint;
2.      import java.io.IOException;
3.      import java.util.Iterator;
4.      import org.apache.hadoop.io.IntWritable;
5.      import org.apache.hadoop.io.Text;
6.      import org.apache.hadoop.mapred.MapReduceBase;
7.      import org.apache.hadoop.mapred.OutputCollector;
8.      import org.apache.hadoop.mapred.Reducer;
9.      import org.apache.hadoop.mapred.Reporter;
10.
11.     public class WC_Reducer  extends MapReduceBase implements Reducer<Text,IntWritable,Text,IntWritable> {
12.     public void reduce(Text key, Iterator<IntWritable> values,OutputCollector<Text,IntWritable> output,
13.      Reporter reporter) throws IOException {
14.     int sum=0;
15.     while (values.hasNext()) {
16.     sum+=values.next().get();
17.     }
18.     output.collect(key,new IntWritable(sum));
19.     }
20.     }
```

File: WC_Runner.java

```
1.  package com.javatpoint;
2.
3.      import java.io.IOException;
4.      import org.apache.hadoop.fs.Path;
5.      import org.apache.hadoop.io.IntWritable;
6.      import org.apache.hadoop.io.Text;
7.      import org.apache.hadoop.mapred.FileInputFormat;
8.      import org.apache.hadoop.mapred.FileOutputFormat;
```

```
9.  import org.apache.hadoop.mapred.JobClient;
10. import org.apache.hadoop.mapred.JobConf;
11. import org.apache.hadoop.mapred.TextInputFormat;
12. import org.apache.hadoop.mapred.TextOutputFormat;
13. public class WC_Runner {
14.     public static void main(String[] args) throws IOException{
15.         JobConf conf = new JobConf(WC_Runner.class);
16.         conf.setJobName("WordCount");
17.         conf.setOutputKeyClass(Text.class);
18.         conf.setOutputValueClass(IntWritable.class);
19.         conf.setMapperClass(WC_Mapper.class);
20.         conf.setCombinerClass(WC_Reducer.class);
21.         conf.setReducerClass(WC_Reducer.class);
22.         conf.setInputFormat(TextInputFormat.class);
23.         conf.setOutputFormat(TextOutputFormat.class);
24.         FileInputFormat.setInputPaths(conf,new Path(args[0]));
25.         FileOutputFormat.setOutputPath(conf,new Path(args[1]));
26.         JobClient.runJob(conf);
27.     }
28. }
```