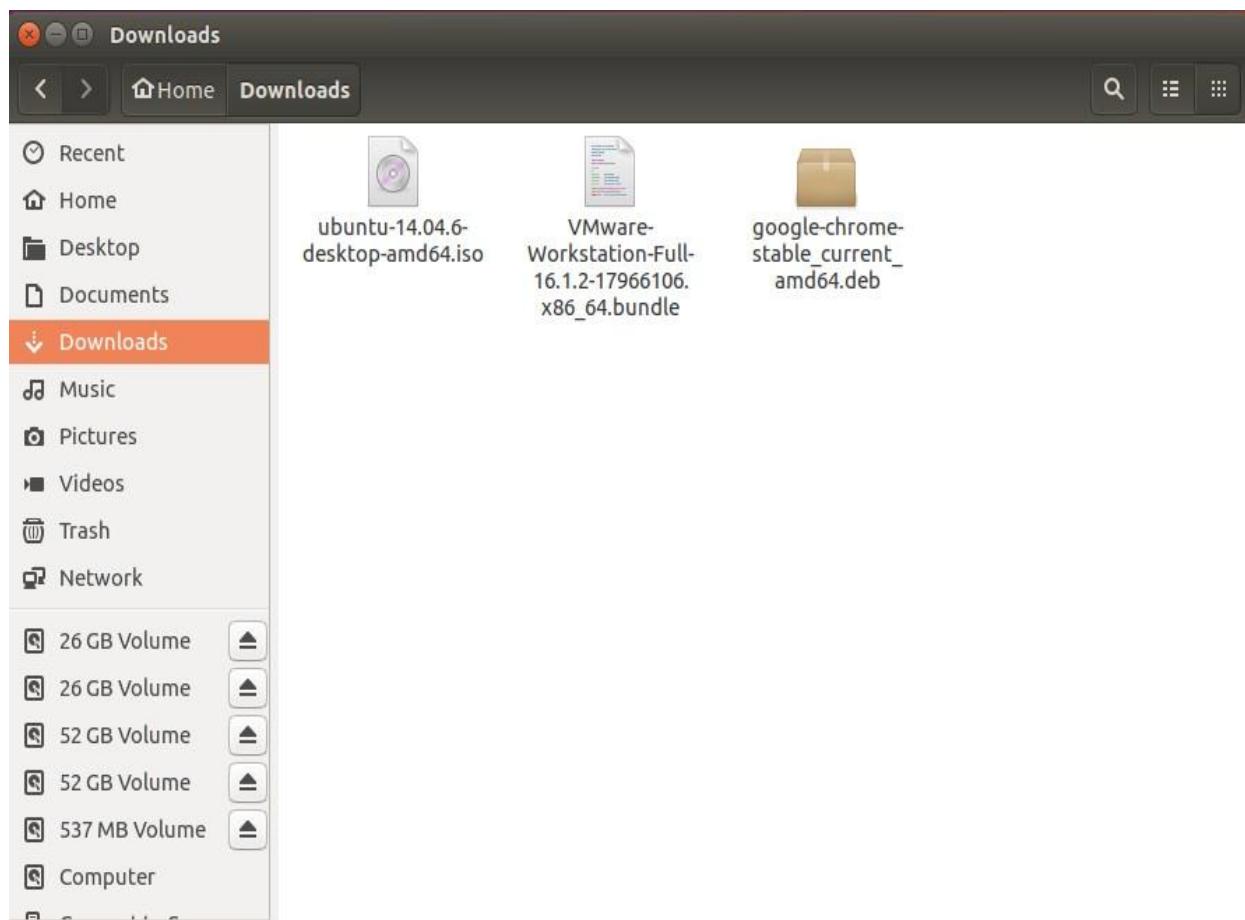


EX.NO.1 INSTALL VIRTUALBOX WITH LINUX OS ON TOP OF WINDOWS**DATE:****AIM:**

To install Virtualbox with different flavours of linux or windows OS on top of windows host operating system.

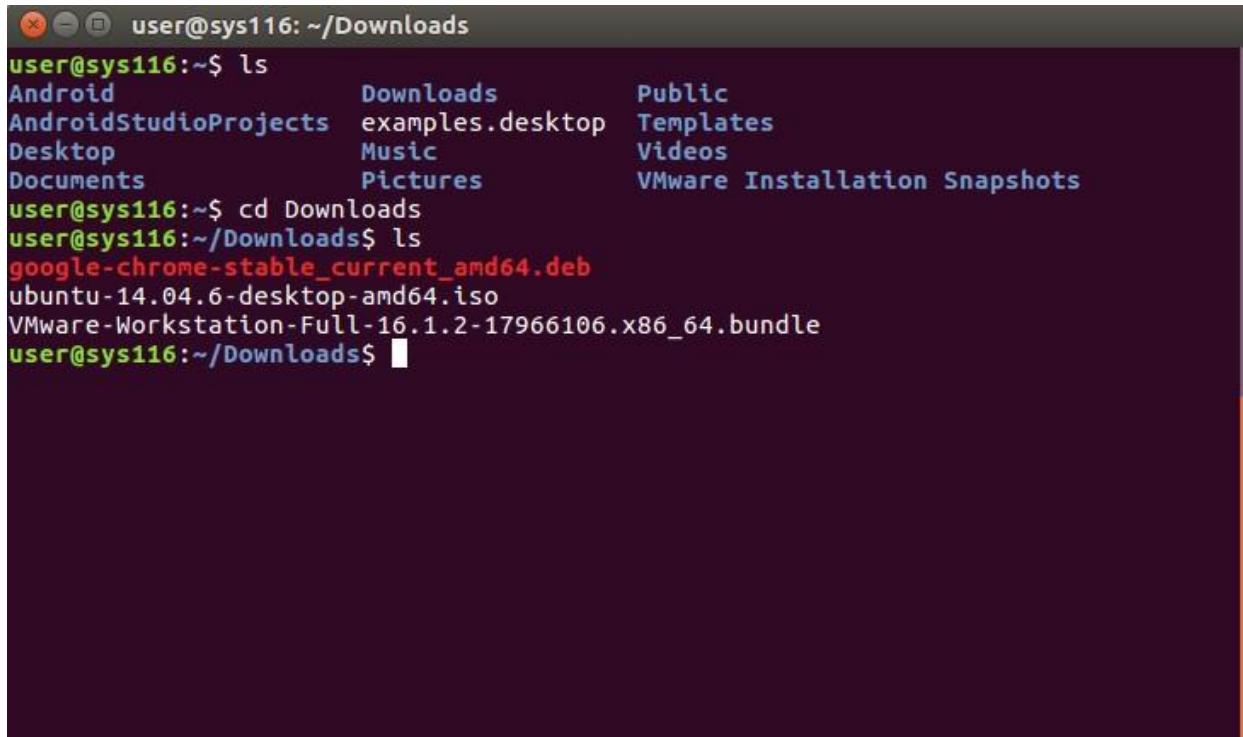
Step 1:

Download the following
VMware package
Guest OS (Ubuntu) ISO file



Step 2:

Navigate to the directory where the above files are downloaded.



```
user@sys116: ~/Downloads
user@sys116:~$ ls
Android           Downloads      Public
AndroidStudioProjects examples.desktop Templates
Desktop           Music         Videos
Documents          Pictures      VMware Installation Snapshots
user@sys116:~$ cd Downloads
user@sys116:~/Downloads$ ls
google-chrome-stable_current_amd64.deb
ubuntu-14.04.6-desktop-amd64.iso
VMware-Workstation-Full-16.1.2-17966106.x86_64.bundle
user@sys116:~/Downloads$
```

Step 3:

Change the permissions of the VMware package. Use the following command, chmod a+x

<VMWARE_PACKAGE_NAME> That is,

chmod a+x

VMware-Workstation-Full-16.1.2-17966106.x86_64

.bundle

Run the next command to install the VMware.

sudo

./VMware-Workstation-Full-16.1.2-17966106.x86_64.bundle

```
user@sys116: ~/Downloads
user@sys116:~/Downloads$ chmod a+x VMware-Workstation-Full-16.1.2-17966106.x86_6
4.bundle
user@sys116:~/Downloads$ sudo ./VMware-Workstation-Full-16.1.2-17966106.x86_64.b
undle
```

```
user@sys116: ~/Downloads
user@sys116:~/Downloads$ chmod a+x VMware-Workstation-Full-16.1.2-17966106.x86_6
4.bundle
user@sys116:~/Downloads$ sudo ./VMware-Workstation-Full-16.1.2-17966106.x86_64.b
undle
[sudo] password for user:
Extracting VMware Installer...done.
Installing VMware Installer 3.0.0
Copying files...
```

```
user@sys116: ~/Downloads
Configuring...
Installing VMware Player Application 16.1.2
Copying files...
Configuring...
Installing VMware OVF Tool component for Linux 4.4.1
Copying files...
Configuring...
Installing VMware Network Editor User Interface 16.1.2
Copying files...
Configuring...
Installing VMware VIX Core Library 1.17.0
Copying files...
Configuring...
Installing VMware VIX Workstation-16.0.0 Library 1.17.0
Copying files...
Configuring...
Installing VMware VProbes component for Linux 16.1.2
Copying files...
Configuring...
Installing VMware Workstation 16.1.2
Copying files...
Configuring...
Installation was successful.
user@sys116:~/Downloads$
```

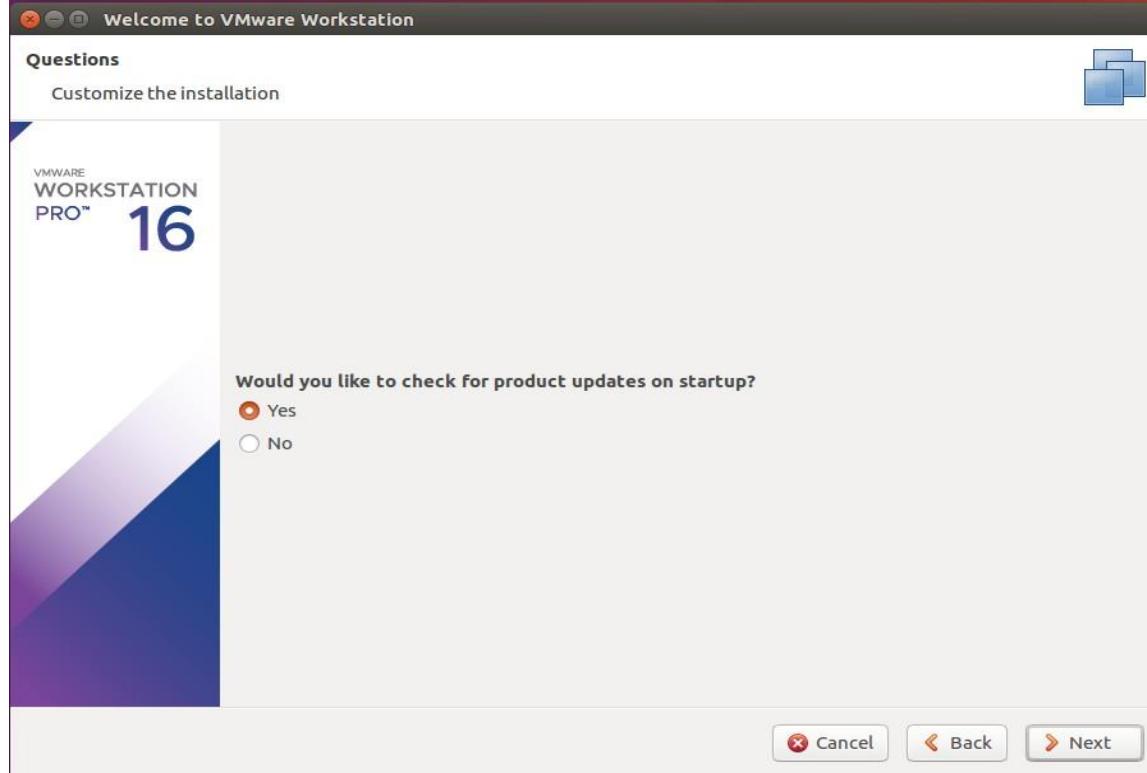
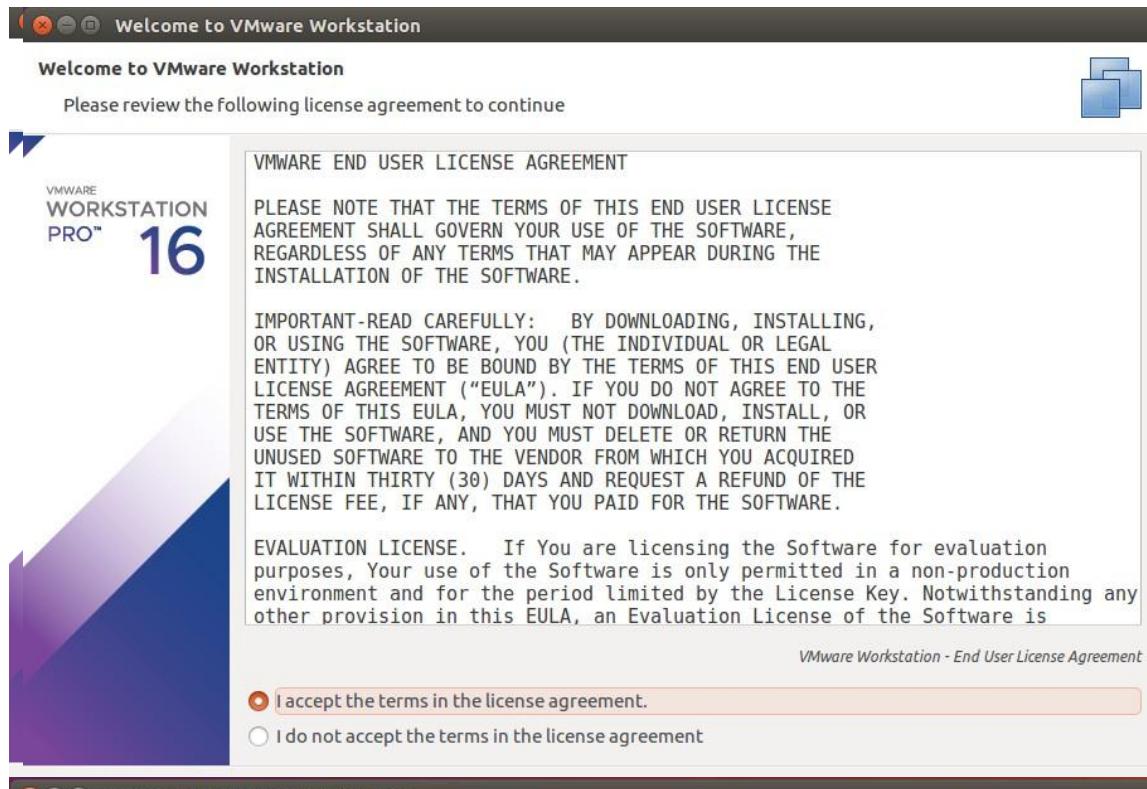
Step 4:

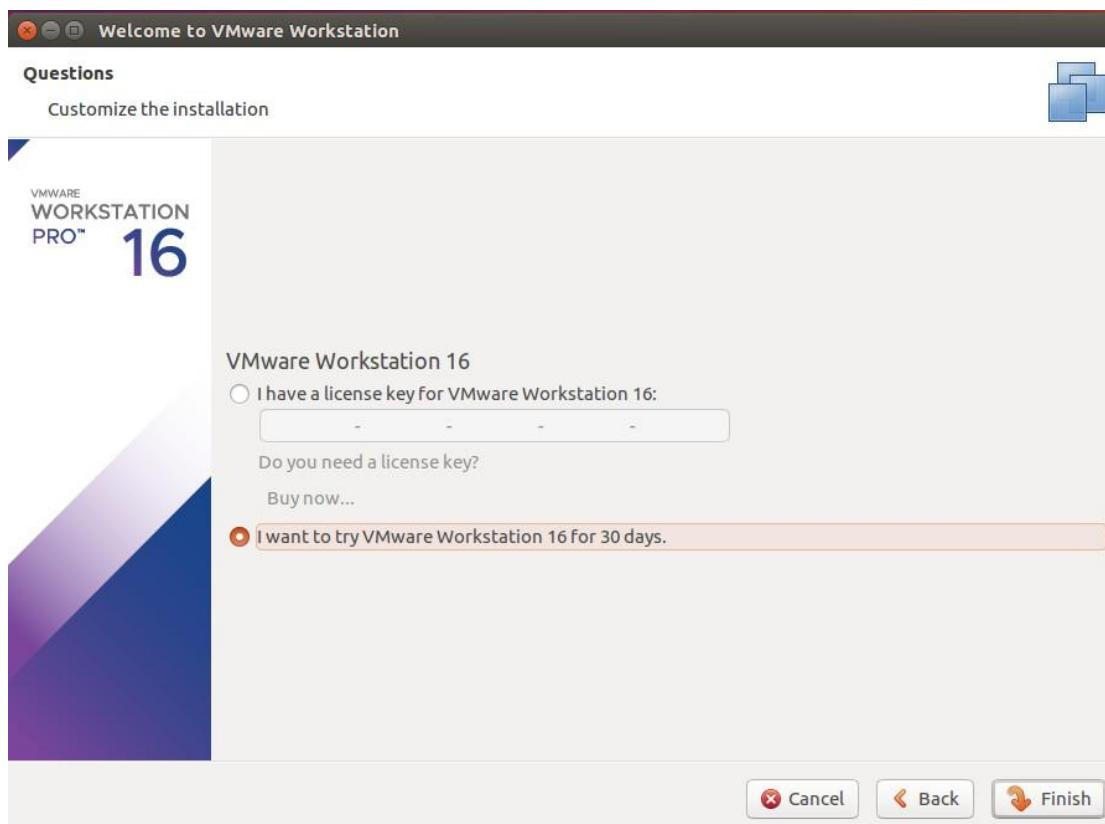
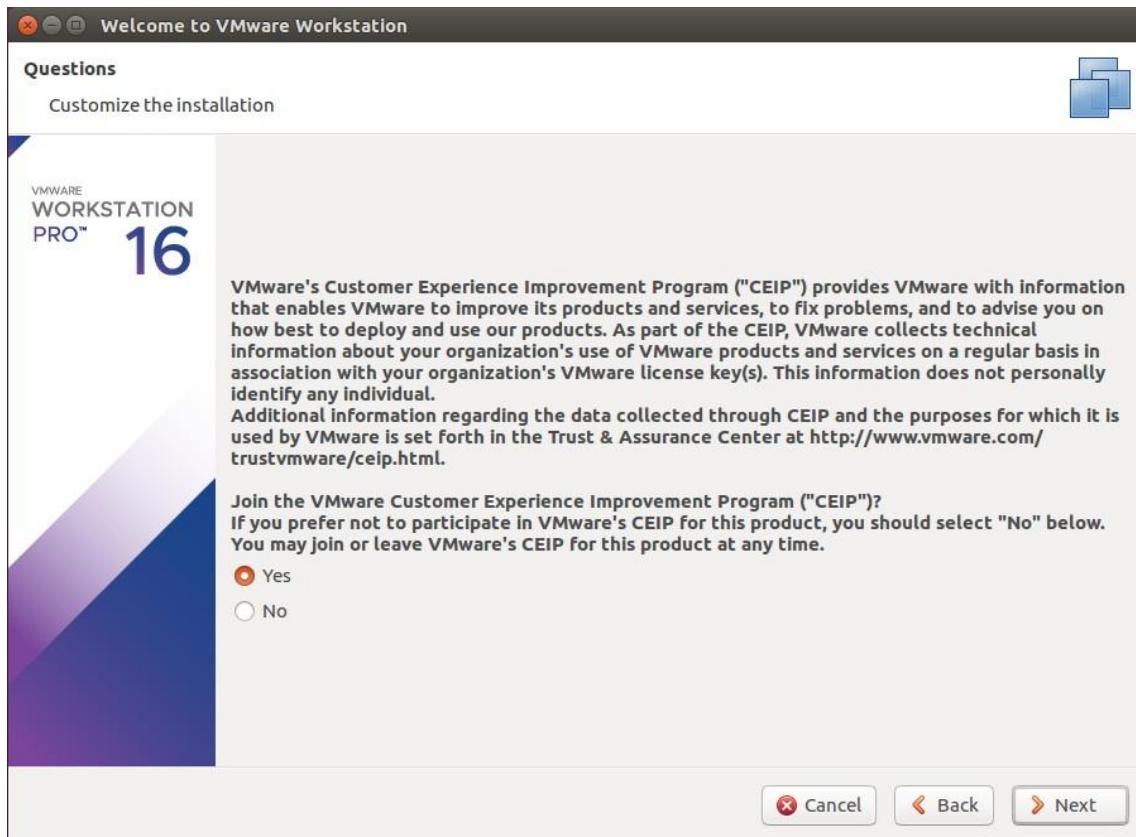
Open VMware Player

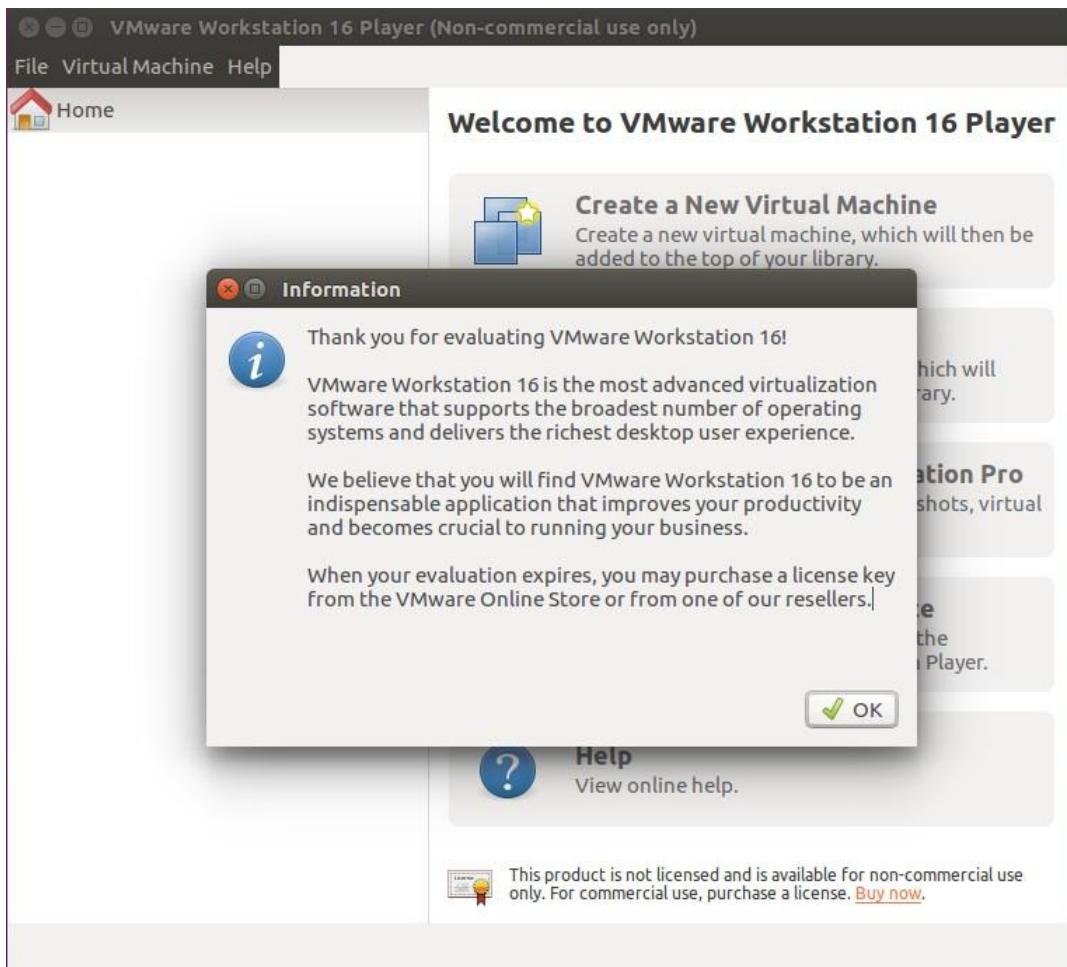
Accept all the agreements, terms and conditions.

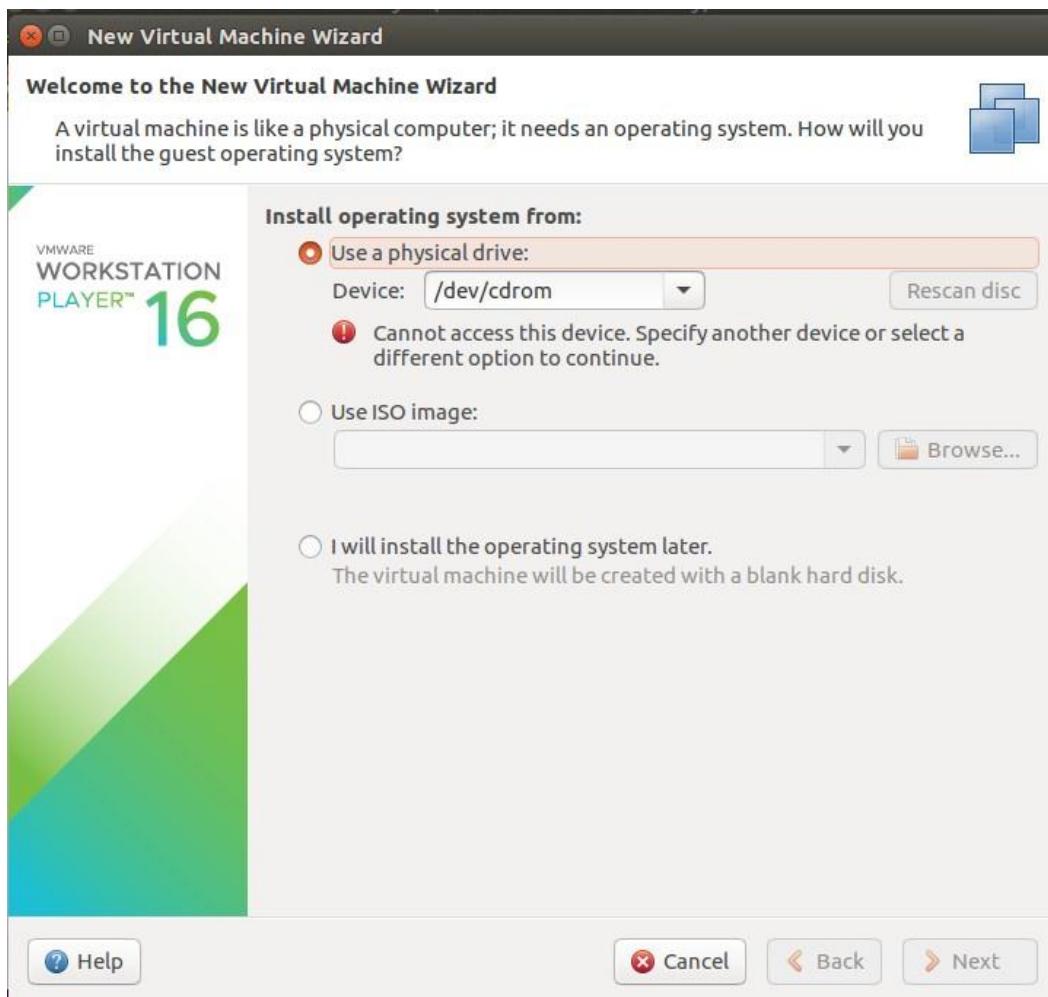
Select all the default settings and continue.

VMware is successfully installed on your Host Machine.







**Step 5:**

Select “Create a new Virtual Machine”

Select “Use ISO image” and Browse your Guest OS ISO image path and click “Next”.

Use your password as “admin” and click “Next”.

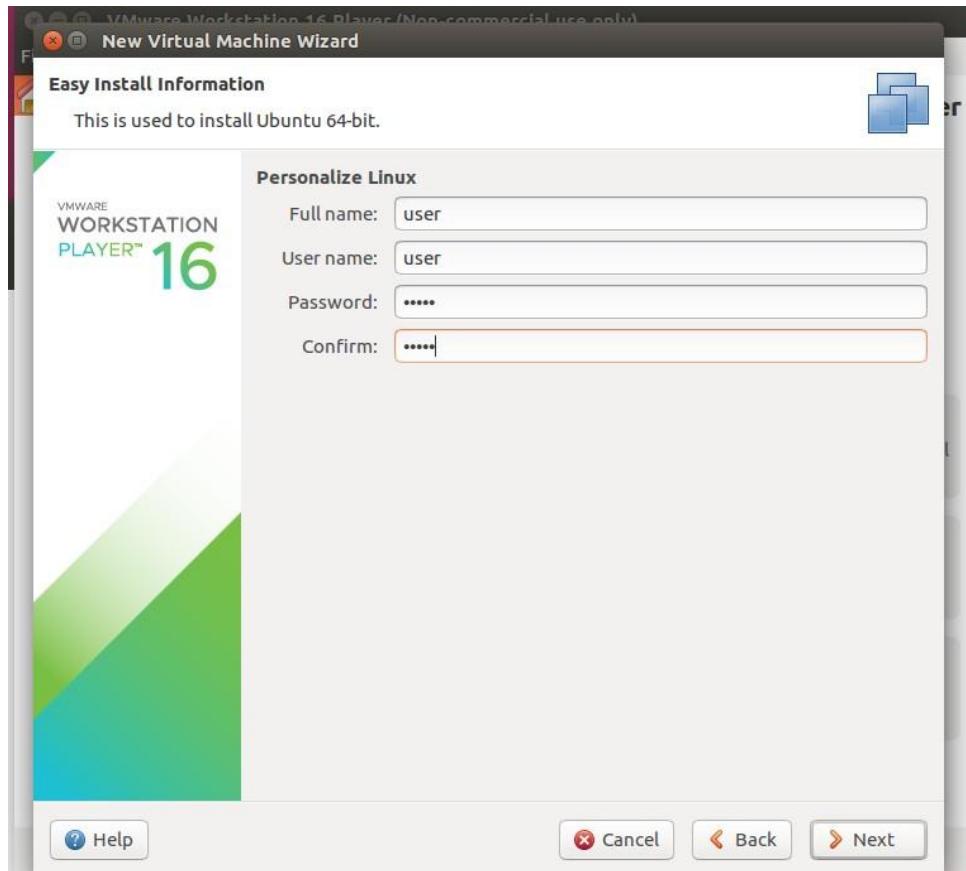
Click “Next”

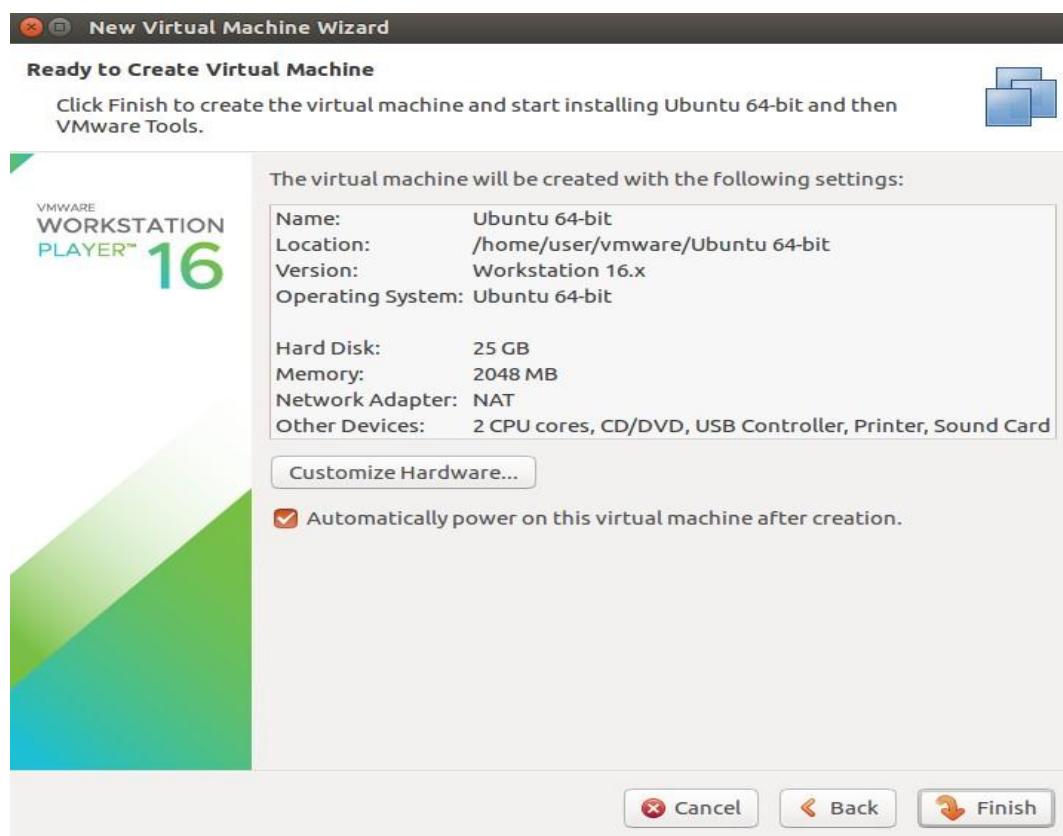
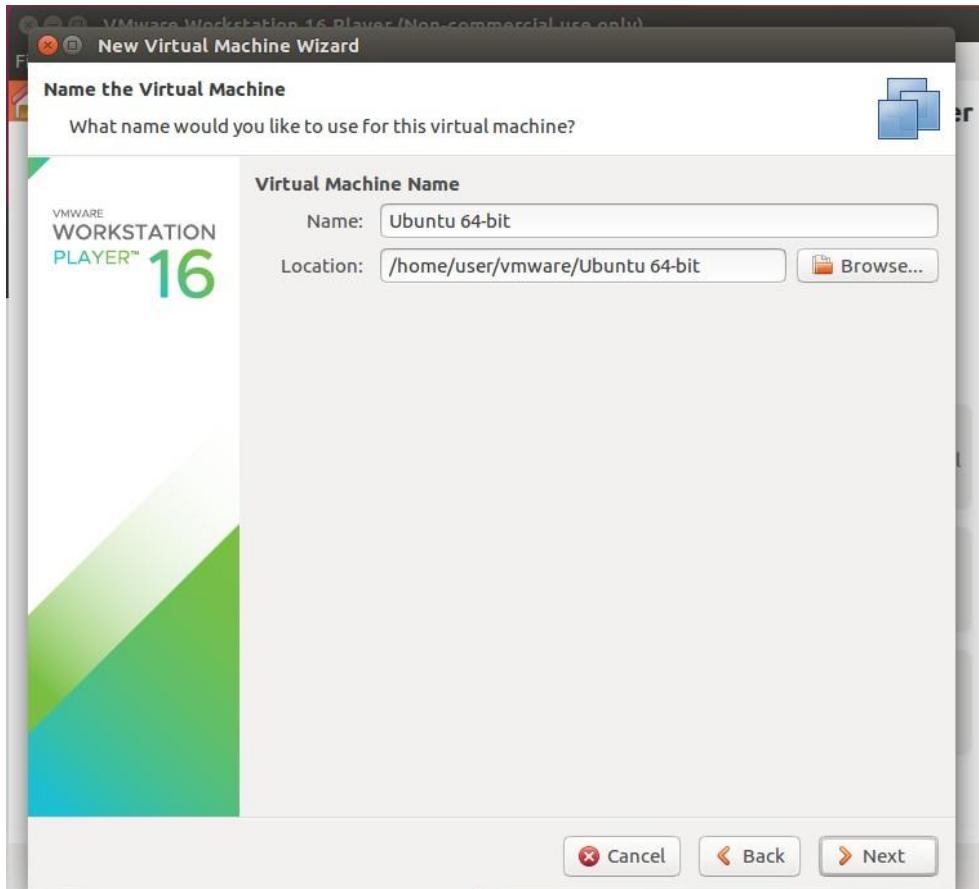
Select your disk size upto a maximum of 25GB and Click “Next”.

Select “Finish”

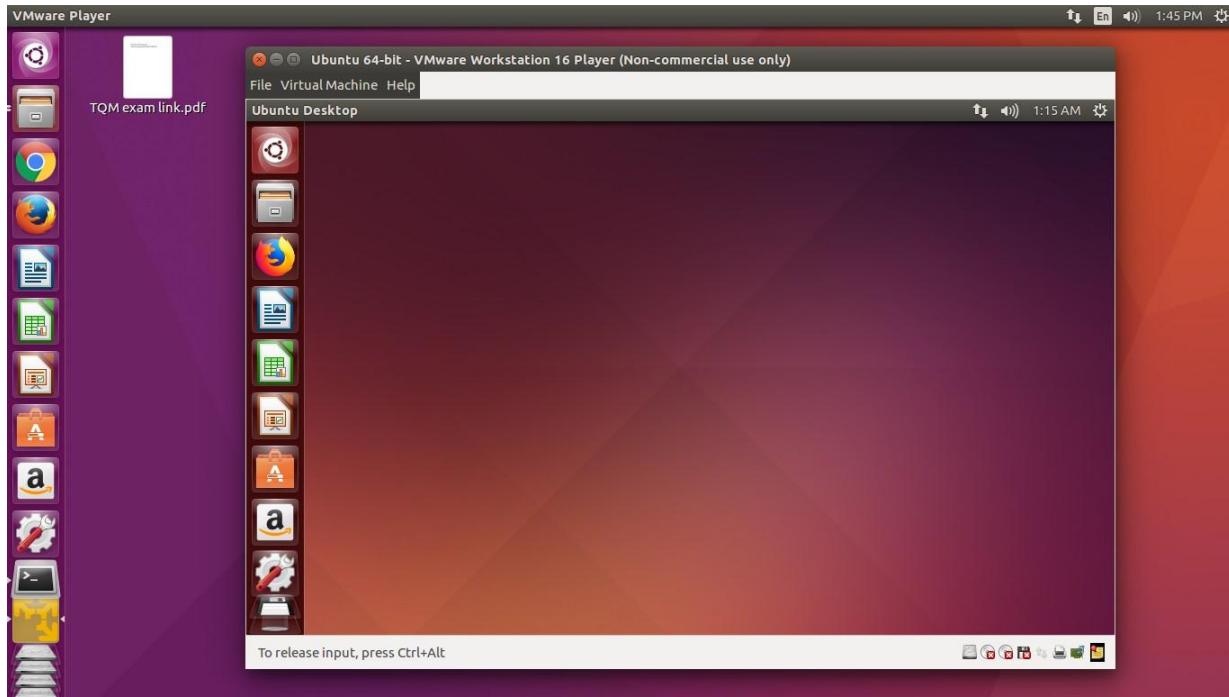
Now sit back and wait until your installation is complete.

Once done, you’ll see your new OS on your VMware installed perfectly





Guest OS is installed Successfully.



RESULT:

Thus installation of Virtualbox with different flavours of linux or windows OS on top of windows host operating system is completed successfully.

Ex.No: 2**Install Gcc Compilers in Virtual Machine****Date :**

Aim: To install a C compiler in the virtual machine and execute a sample program.

Algorithm:

Step 1: To check Gcc compiler is installed or not.

```
$ dpkg -l | grep gcc
```

Step 2: If GCC compiler is not installed in VM, to execute the following command

```
$sudo apt-get install gcc (or) $sudo apt-get install build-essential
```

Step 3: Open a Vi Editor

Step 4: Get the no. of rows and columns for first and second matrix.

Step 5: Get the values of x and y matrix using for loop.

Step 6: Find the product of first and second and store the result in multiply matrix.

```
multiply[i][j]=multiply[i][j]+(first[i][k]*second[k][j]);
```

Step 7: Display the resultant matrix.

Step 8: Stop the program.

Program:

```
#include <stdio.h>
void main()
{
    int m, n, p, q, c, d, k, sum = 0;
    int first[10][10], second[10][10], multiply[10][10];
    printf("Enter the number of rows and columns of first matrix\n");
    scanf("%d%d", &m, &n);
    printf("Enter the elements of first matrix\n");
    for ( c = 0 ; c < m ; c++ )
        for ( d = 0 ; d < n ; d++ )
            scanf("%d", &first[c][d]);
    printf("Enter the number of rows and columns of second matrix\n");
    scanf("%d%d", &p, &q);
    if ( n != p )
        printf("Matrices with entered orders can't be multiplied with each other.\n");
    else
    {
        printf("Enter the elements of second matrix\n");

        for ( c = 0 ; c < p ; c++ ) for ( d
            = 0 ; d < q ; d++ ) scanf("%d",
            &second[c][d]); for ( c = 0 ; c
            < m ; c++ )
        {
            for ( d = 0 ; d < q ; d++ )
            {
                for ( k = 0 ; k < p ; k++ )
                {
                    sum = sum + first[c][k]*second[k][d];
                }
            }
        }
    }
}
```

```
        multiply[c][d] = sum;
        sum = 0;
    }
}
printf("Product of entered matrices:-\n");
for ( c = 0 ; c < m ; c++ )
{
}

}
```

Output:

Compile: cse105@it105-HP-ProDesk-400-G1-SFF:~\$ gcc matrix.c

Run: cse105@cse105-HP-ProDesk-400-G1-SFF:~\$./a.out

Enter the number of rows and columns of first matrix

2 2

Enter the elements of first matrix

2 2 2 2

Enter the number of rows and columns of second matrix

2 2

Enter the elements of second matrix

2 2 2 2

Product of entered matrices:-

8 8

8 8

Result :

Thus the C Complier is installed and program has been successfully executed.

EXNO: 3**INSTALLING GOOGLE APP ENGINE****DATE:****AIM:**

To Install Google App Engine. Create hello world app and other simple web applications using python/java.

STEP 1

- Download google cloud SDK from the google cloud website
[<https://cloud.google.com/sdk/docs/install>]

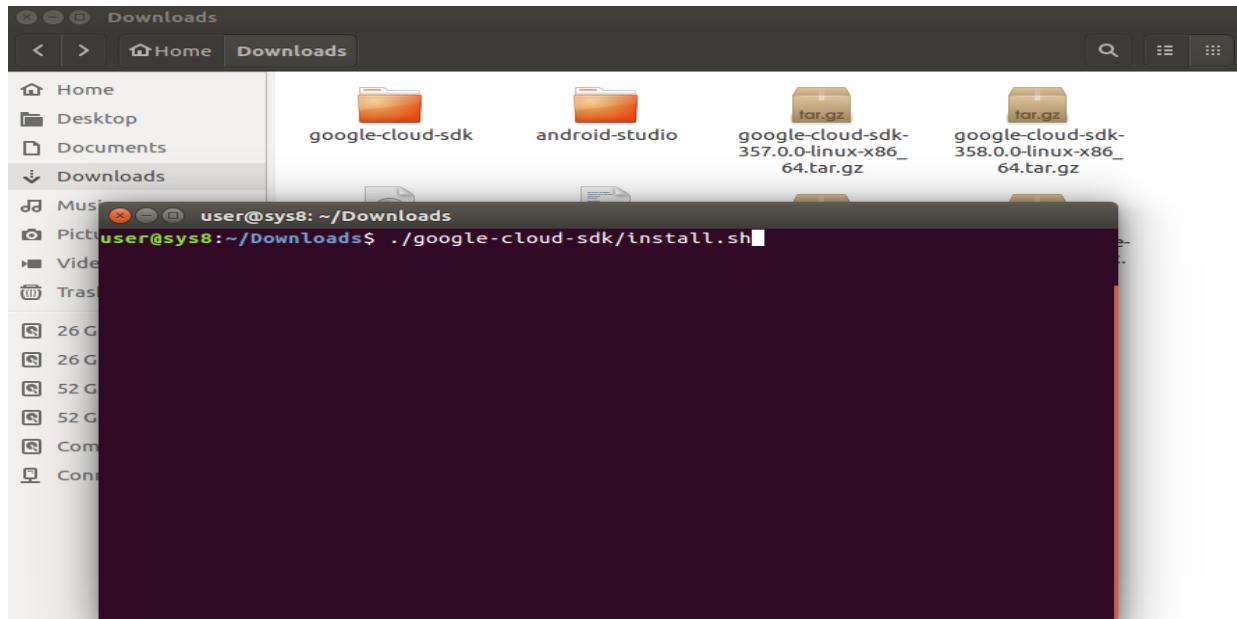
(OR)

• Click the following link to download the required Google Cloud SDK
[https://dl.google.com/dl/cloudsdk/channels/rapid/downloads/google-cloud-sdk-357.0.0-linux-x86_64.tar.gz]

Extract the zip folder in Downloads/ folder

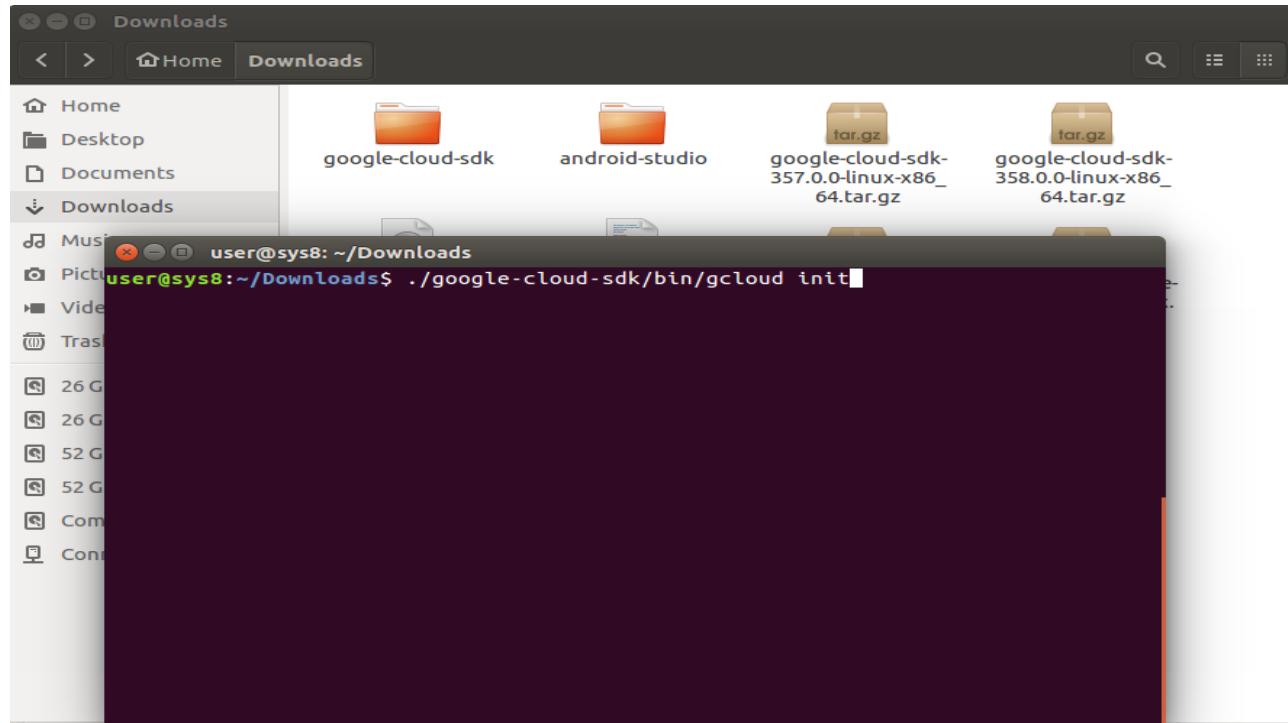
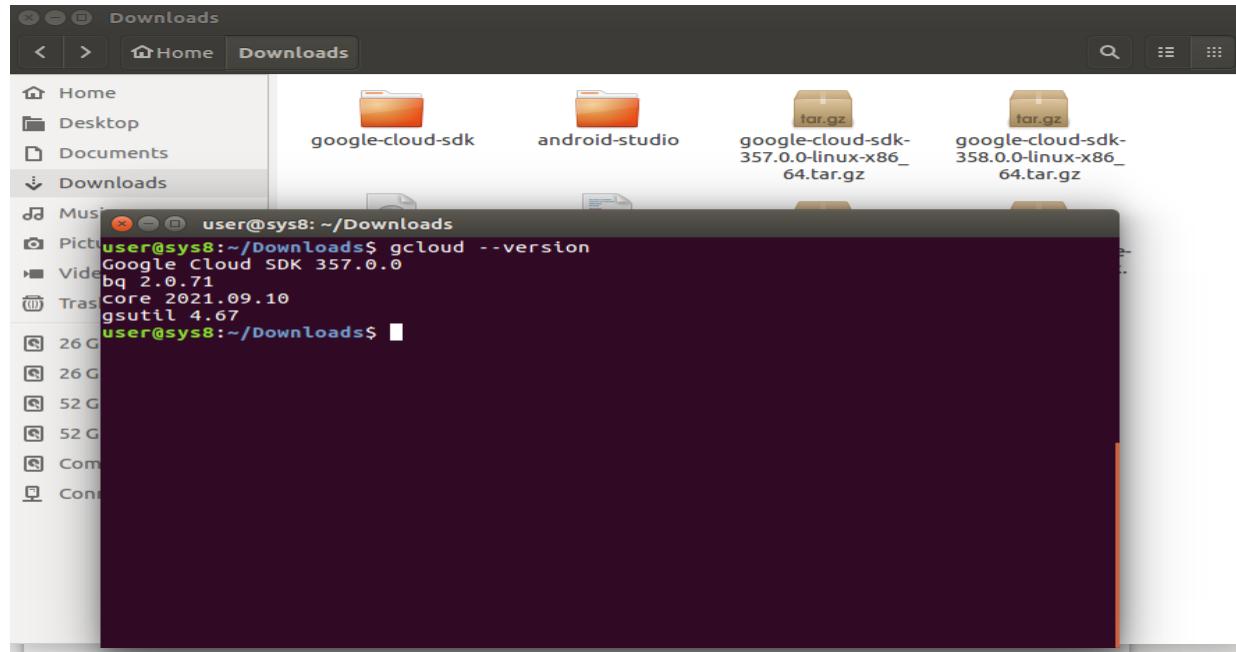
STEP 2

- Open new terminal in Downloads/ folder and install the google cloud
`./google-cloud-sdk/install.sh`



- Open a new terminal and initialized the gcloud

```
./google-cloud-sdk/bin/gcloud init
```



- Login to your google account and continue the process by accepting the terms and conditions
- Now run `dev_appserver.py` command in your terminal.

```
* Run `gcloud topic --help` to learn about advanced features of the SDK like arg
  files and output formatting
user@sys8:~/Downloads$ dev_appserver.py
This action requires the installation of components: [app-engine-
python, cloud-datastore-emulator]

Your current Cloud SDK version is: 357.0.0
Installing components from version: 357.0.0

These components will be installed.

+-----+-----+-----+
| Name | Version | Size   |
+-----+-----+-----+
| Cloud Datastore Emulator | 2.1.0   | 18.4 MiB
| gRPC Python library      | 1.20.0  | 2.1 MiB
| gcloud app Python Extensions | 1.9.93 | 7.7 MiB
+-----+-----+-----+

For the latest full release notes, please visit:
  https://cloud.google.com/sdk/release_notes

Do you want to continue (Y/n)?
```

- Create a `main.py` file and `app.yaml` file in your desired folder
- Inside `main.py` write a simple hello world program

```
app.yaml (~/GAE/Ex1) - gedit
Open Save
runtime: python27
api_version: 1
threadsafe: false

handlers:
- url: /*
  script: main.py
  secure: always
  login: admin
```

YAML Tab Width: 8 Ln 10, Col 1 INS

- Finally run the app using

```
user@sys8:~/GAE/Ex1
user@sys8:~/Downloads$ cd ..
user@sys8:~$ ls
Android           Desktop          GAE-Projects  snap
AndroidStudioProjects  Documents      Music        Templates
curl-7.50.3        Downloads      Pictures     Videos
curl-7.50.3.tar.gz examples.desktop Public      vmware
user@sys8:~$ mkdir -p GAE/Ex1/
user@sys8:~$ cd GAE/Ex1/
user@sys8:~/GAE/Ex1$ touch main.py app.yaml
user@sys8:~/GAE/Ex1$ gedit main.py
user@sys8:~/GAE/Ex1$ gedit app.yaml
user@sys8:~/GAE/Ex1$ dev_appserver.py "app.yaml"
INFO    2021-09-30 08:07:26,554 devappserver2.py:309] Skipping SDK update check
.
WARNING 2021-09-30 08:07:26,964 simple_search_stub.py:1196] Could not read search indexes from /tmp/appengine.None.user/search_indexes
INFO    2021-09-30 08:07:26,965 api_server.py:383] Starting API server at: http://localhost:38679
INFO    2021-09-30 08:07:27,017 dispatcher.py:267] Starting module "default" running at: http://localhost:8080
INFO    2021-09-30 08:07:27,018 admin_server.py:150] Starting admin server at: http://localhost:8000
INFO    2021-09-30 08:07:29,037 instance.py:294] Instance PID: 5617
```

```
$ dev_appserver.py "app.yaml"
```

- Go to <http://localhost:8000>

Result :

Thus installation of Google App Engine is completed and hello world app using python is completed successfully

EXNO: 4**DATE:****GAE LAUNCHER TO LAUNCH THE WEB APPLICATIONS.****AIM:**

To use GAE launcher to launch the web applications.

PROCEDURE :

Step 1 – Go to Cloud SDK

<https://cloud.google.com/sdk/docs/install>

Step 2 – Download Linux 64-bit (x86_64)

The screenshot shows the Google Cloud Cloud SDK: Command Line Interface documentation page. The main content area displays the following information:

- A note: "Your operating system must be able to run one of these supported Python versions in order for Cloud SDK to run. For more information on how to choose and configure your Python interpreter, refer to [gcloud topic startup](#).
- Step 2: "Download one of the following:"
- A note: "★ Note: To determine your OS version, run `getconf LONG_BIT` from your command line."
- A table showing download links for different platforms:

Platform	Package	Size	SHA256 Checksum
Linux 64-bit (x86_64)	google-cloud-sdk-357.0.0-linux-x86_64.tar.gz	87.7 MB	7fa9058eceee419564f53ad768699a78baec5b712b15db63f9e2aed42b5dcde29
Linux 64-bit (arm)	google-cloud-sdk-357.0.0-linux-arm.tar.gz	85.0 MB	d11a08f02db5be7e03688e13fff0ec2a9a9af19e4ab2cf04c75f90015c6c6a4
Linux 32-bit (x86)	google-cloud-sdk-357.0.0-linux-x86.tar.gz	85.0 MB	62efaaa3032780f665bb325aed9eacd6a1273f3fc680f94c9c4168dda4c1933f
- Step 3: "Alternatively, to download the Linux 64-bit archive file from your command-line, run:
`curl -O https://dl.google.com/dl/cloudsdk/channels/rapid/downloads/google-cloud-sdk-357.0.0-linux-x86_64.tar.gz`
- For the 64-bit arm archive file, run:
`curl -O https://dl.google.com/dl/cloudsdk/channels/rapid/downloads/google-cloud-sdk-357.0.0-linux-arm.tar.gz`
- For the 32-bit archive file, run:
`curl -O https://dl.google.com/dl/cloudsdk/channels/rapid/downloads/google-cloud-sdk-357.0.0-linux-x86.tar.gz`

Step 3 – Extract the Zip File from downloads

Step 4 – Open Terminal in Downloads

Step 5 – Run the following command

./google-cloud-sdk/install.sh

Step 6 – Init the cloud sdk using the following command

./google-cloud-sdk/bin/gcloud init

Step 7 – Run “**dev_appserver.py**” in the terminal

Step 8 – Create Project Folder, Download the zip file from the below link and unzip it.

<https://drive.google.com/file/d/18gHSvne6r11yT70X6x7HtmJlpLTBieYq/view?usp=sharing>

Step 9 - Create a App.yaml file

App.yaml

```
runtime: python27  
api_version: 1  
threadsafe: true  
handlers:  
- url: /  
  static_files: project/index.html  
  upload: project/index.html  
- url: /  
  static_dir: project
```

Step 10 - Go to the Project Folder and open the terminal

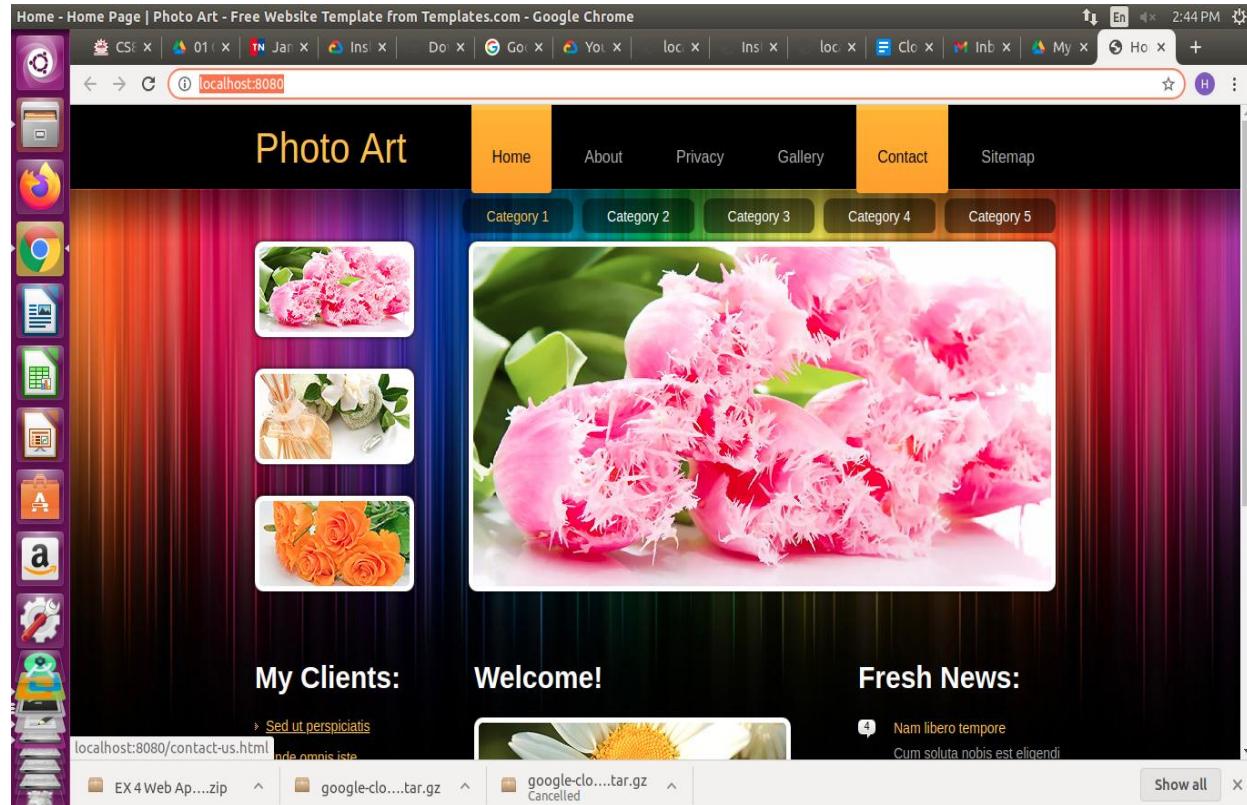
dev_appserver.py “app.yaml”

Step 11 - Go to <http://localhost:8000>

“If Localhost not opening then run this command”

fuser -n tcp -k 8080

OUTPUT :



RESULT :

Thus the GAE launcher has been successfully used to launch the web applications.

EXNO: 5**DATE:****Study on CloudSim is a simulation toolkit****Aim:**

To study about the tools and features of Cloud sim

Description:

CloudSim is a simulation toolkit that supports the modeling and simulation of the core functionality of cloud, like job/task queue, processing of events, creation of cloud entities(datacenter, datacenter brokers, etc), communication between different entities, implementation of broker policies, etc.

This toolkit allows to:

- Test application services in a repeatable and controllable environment.
- Tune the system bottlenecks before deploying apps in an actual cloud.
- Experiment with different workload mix and resource performance scenarios on simulated infrastructure for developing and testing adaptive application provisioning techniques

Core features of CloudSim are:

- The Support of modeling and simulation of large scale computing environment as federated cloud data centers, virtualized server hosts, with customizable policies for provisioning host resources to virtual machines and energy-aware computational resources
- It is a self-contained platform for modeling cloud's service brokers, provisioning, and allocation policies.
- It supports the simulation of network connections among simulated system elements

Support for simulation of federated cloud environment, that inter-networks resources from both private and public domains.

- Availability of a virtualization engine that aids in the creation and management of multiple independent and co-hosted virtual services on a data center node.

- Flexibility to switch between space shared and time shared allocation of processing cores to virtualized services.

The cloudsim allows to model and simulate the cloud system components, therefore to support its function different set of classes has been developed by its developers like:

- To simulating the regions and datacenters the class named “Datacenter.java” is available in org.cloudbus.cloudsim package.
- To simulate the workloads for cloud, the class named as “Cloudlet.java” is available in org.cloudbus.cloudsim package.
- To simulate the load balancing and policy-related implementation the classes named “DatacenterBroker.java”, “CloudletScheduler.java”, “VmAllocationPolicy.java”, etc are available under org.cloudbus.cloudsim package.
- Now because all the different simulated hardware models are required to communicate with each other to share the simulation work updates for this cloudsim has implemented a discrete event simulation engine that keeps track of all the task assignments among the different simulated cloud components.

Cloudsim is used for

- Load Balancing of resources and tasks
- Task scheduling and its migrations
- Optimizing the Virtual machine allocation and placement policies
- Energy-aware Consolidations or Migrations of virtual machines
- Optimizing schemes for Network latencies for various cloud scenarios

Cloudsim simulation toolkit setup is easy. Before you start to setup CloudSim, following resources must be Installed/downloaded on the local system

- **Java Development Kit(JDK)**
- **Eclipse IDE for Java developers**
- **Download CloudSim source code**
- **Download Common Math libraries**

Result:

Thus the study on cloudsim simulation tool is done successfully.

EXNO: 5 a**DATE :**

SIMULATE A CLOUD SCENARIO USING CLOUDSIM

Aim:

To simulate a cloud scenario using CloudSim and run a scheduling algorithm.

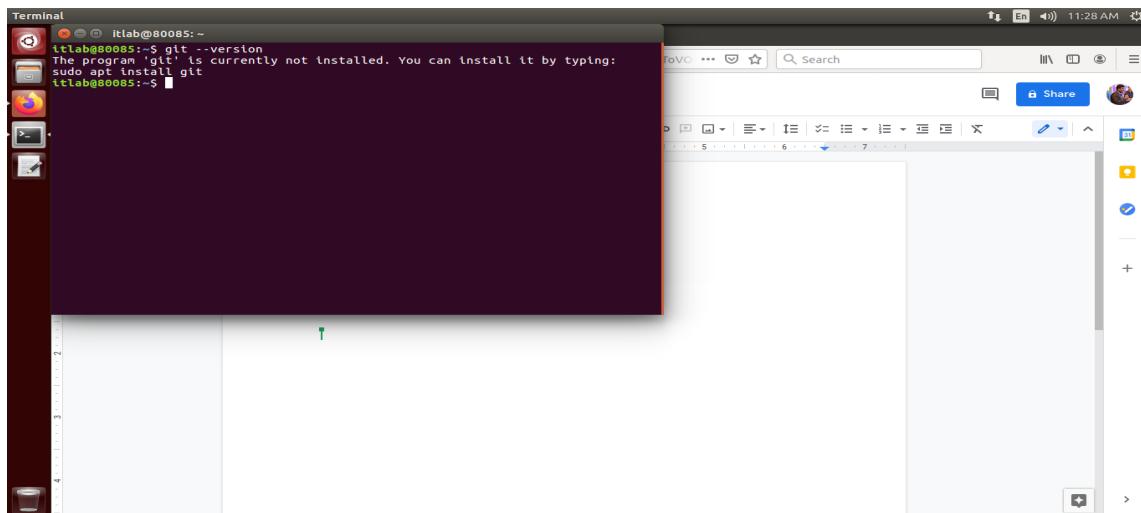
Description:

Cloudsim is a simulation toolkit that supports the modeling and simulation of the core functionality of the cloud. This article discusses the overview of the cloudsim toolkit and helps you get started along with reference to the relevant resources.

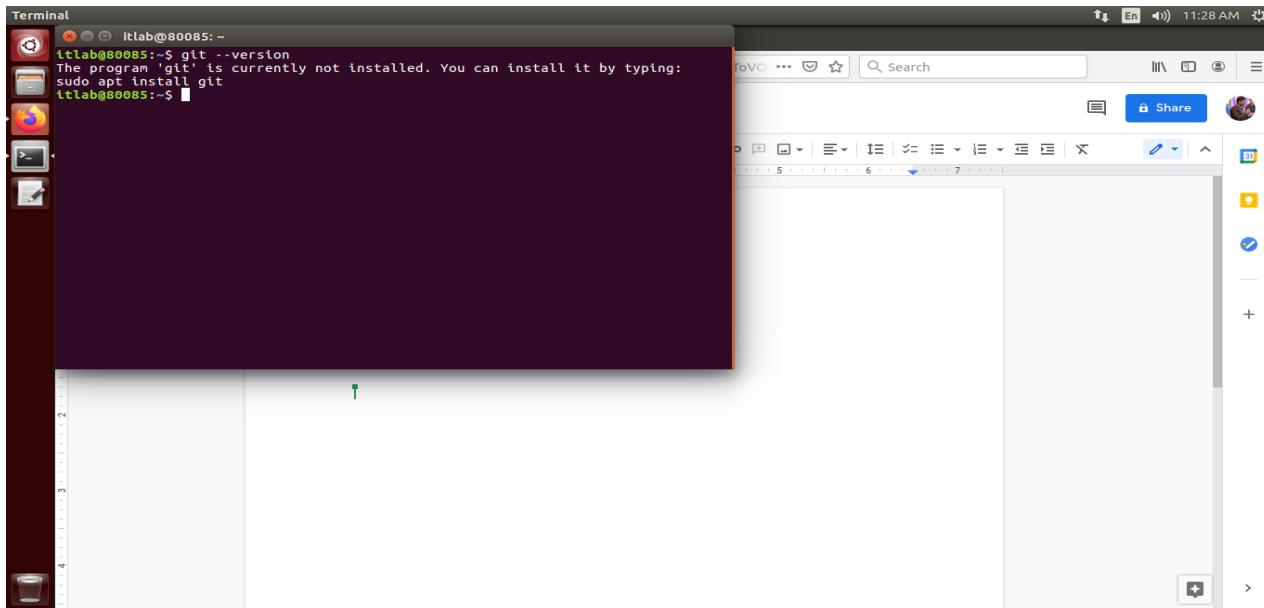
Procedure:

1. Check for git installation

git –version

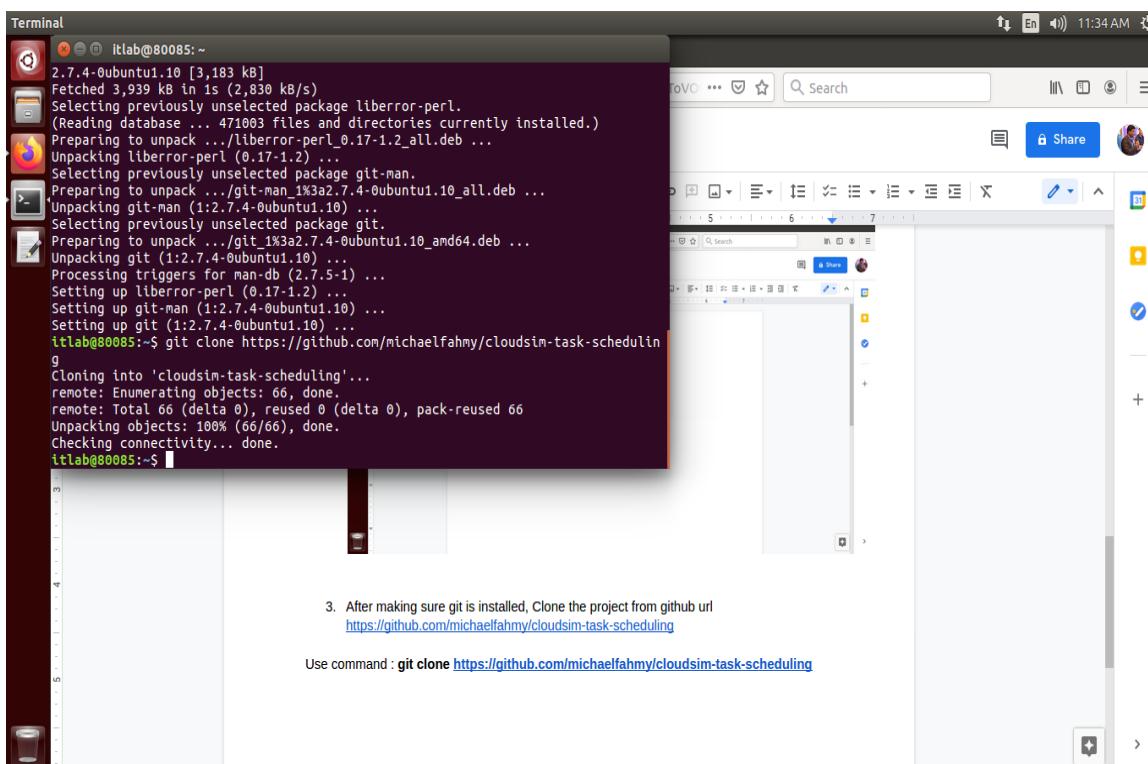


2. If git is not installed,
Use **sudo apt-get install git**



- After making sure git is installed, Clone the project from github url
<https://github.com/michaelfahmy/cloudsim-task-scheduling>

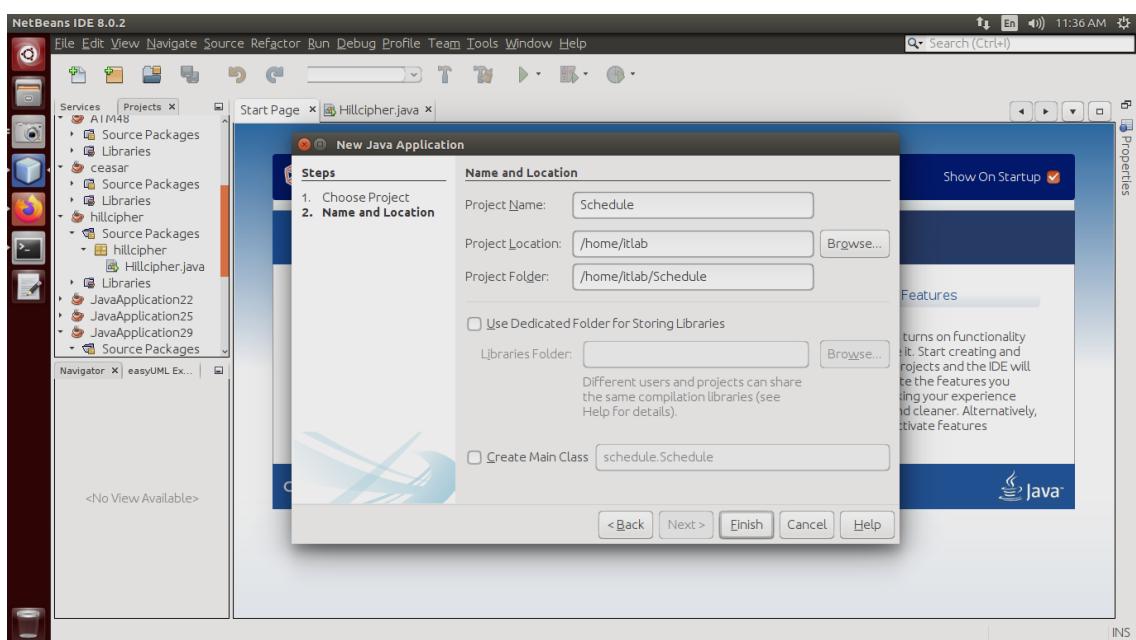
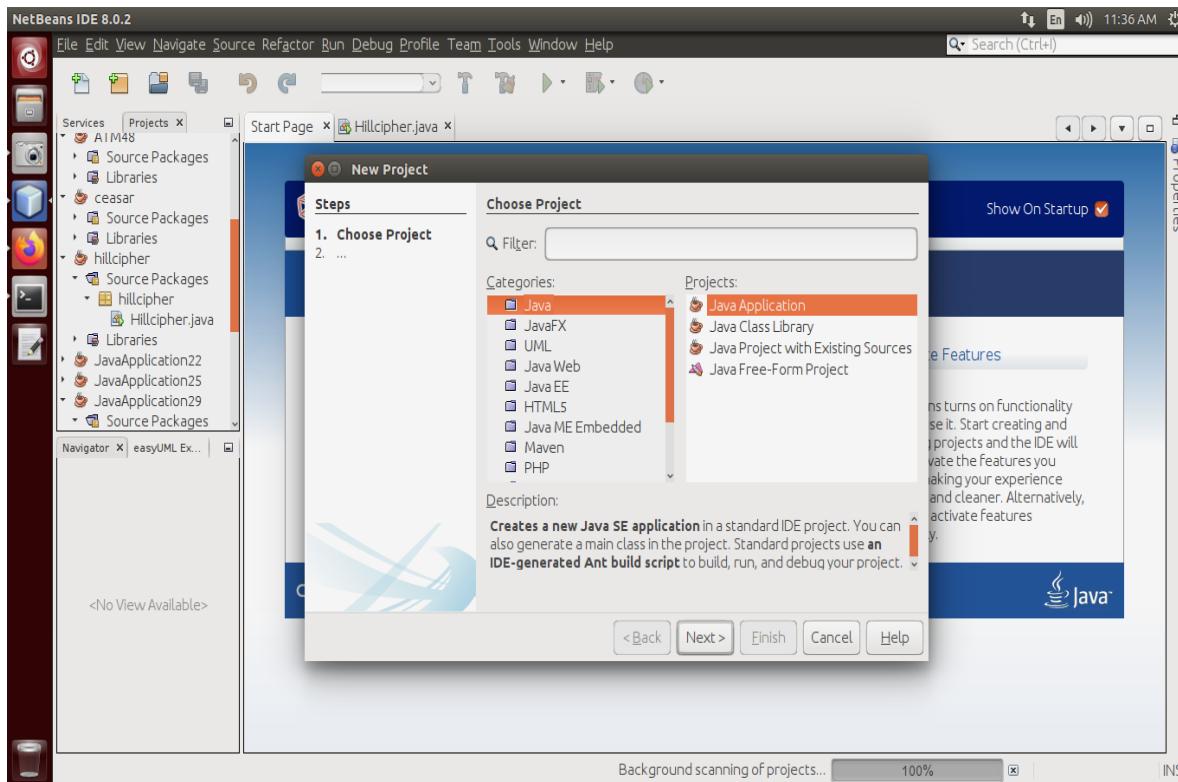
Use command : `git clone https://github.com/michaelfahmy/cloudsim-task-scheduling`



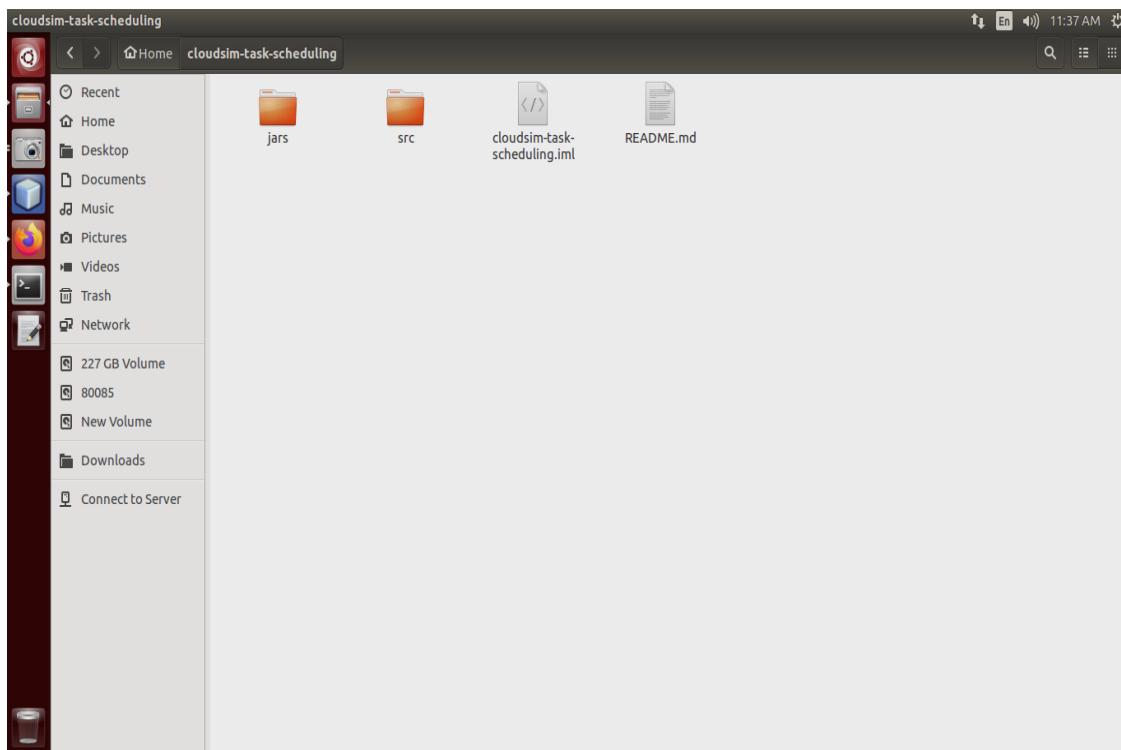
3. After making sure git is installed, Clone the project from github url
<https://github.com/michaelfahmy/cloudsim-task-scheduling>

Use command : `git clone https://github.com/michaelfahmy/cloudsim-task-scheduling`

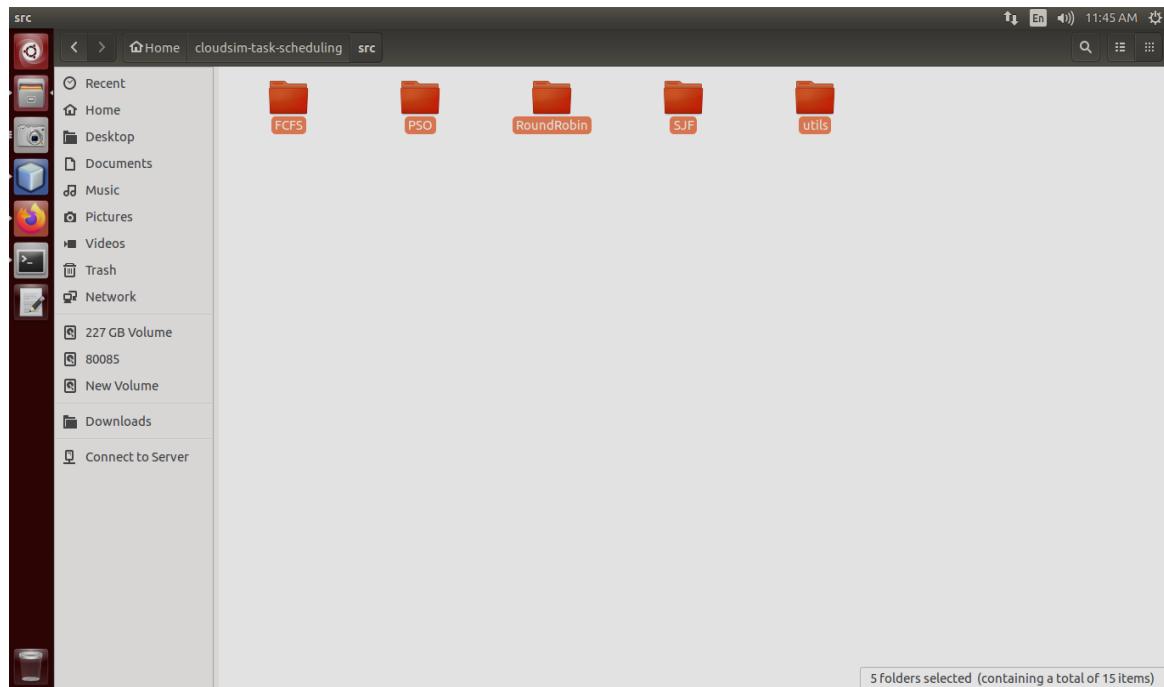
4. Open netbeans 8.x and create a new java project from files



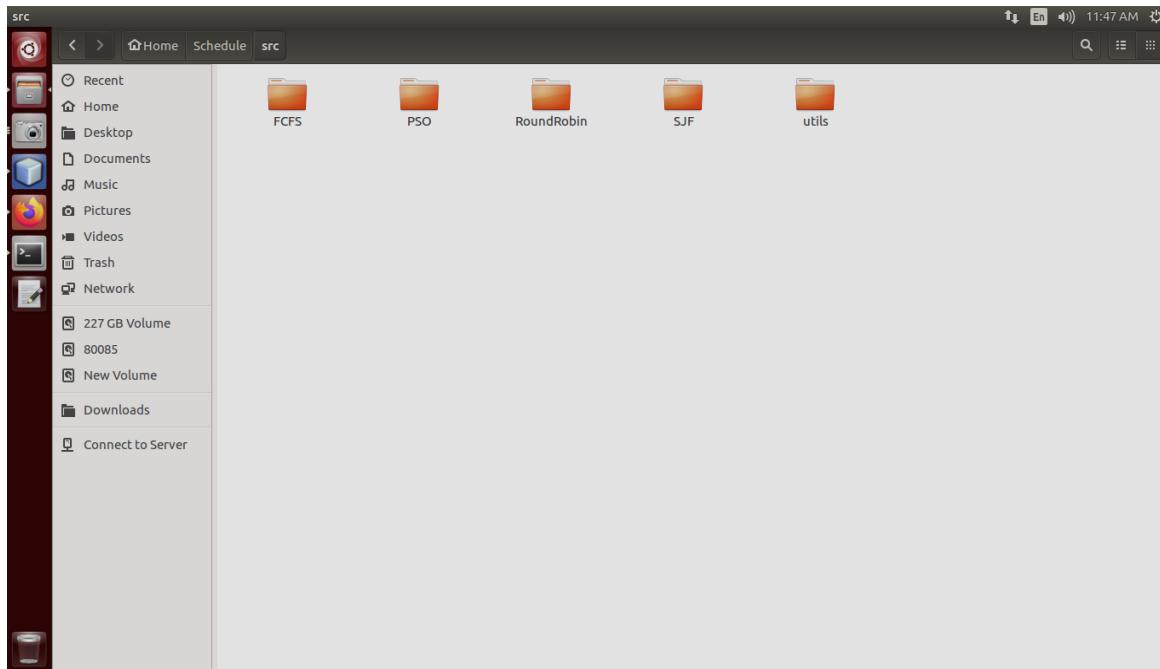
5. Now find the cloned project folder which generally in linux will be in /home/itlab



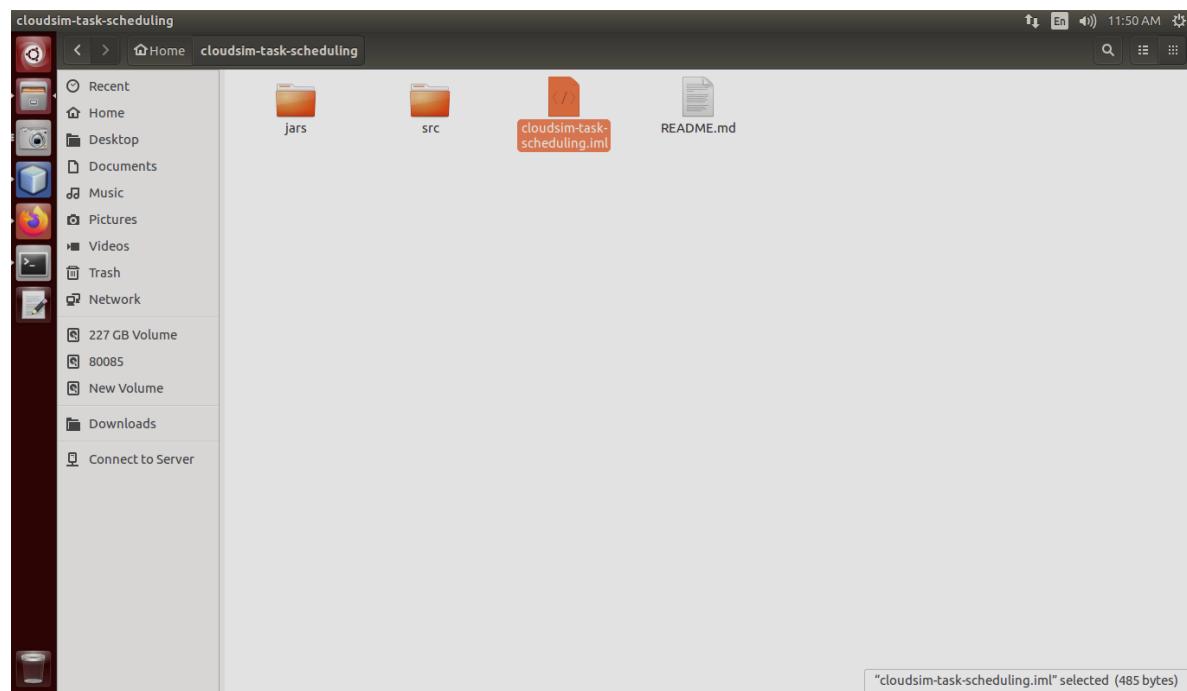
6. Open the src folder and copy the folders

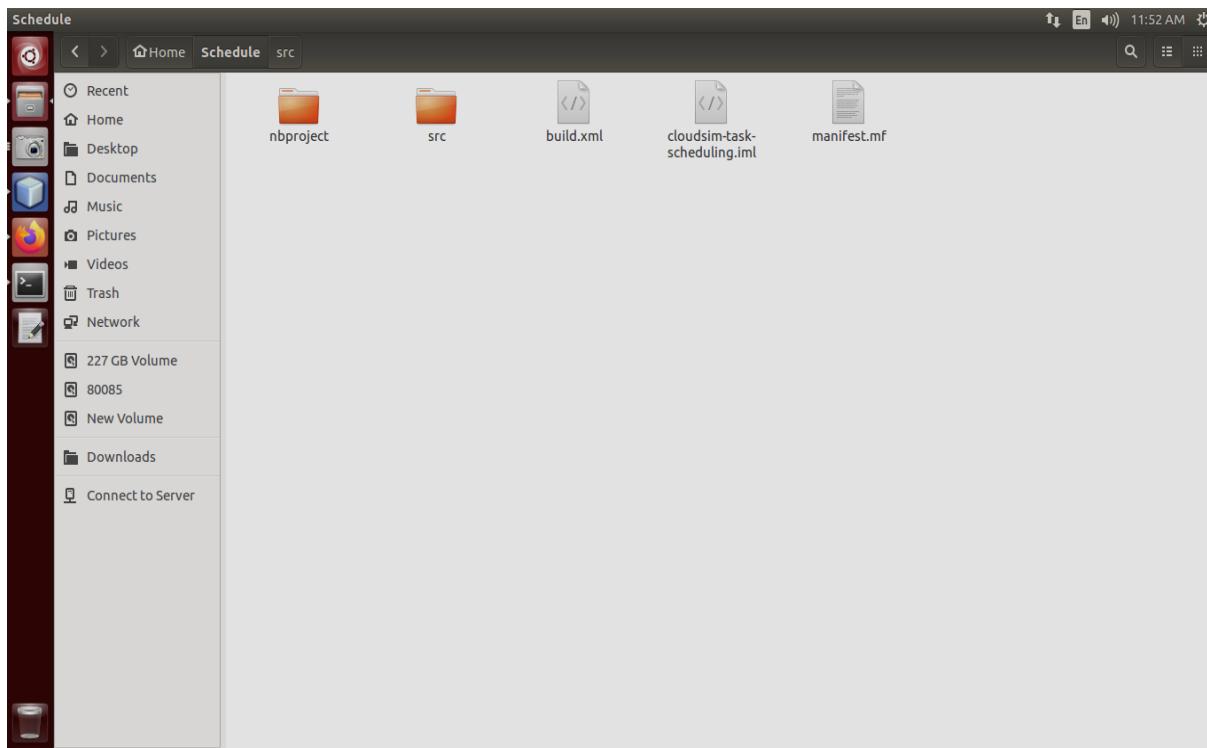


Paste the folders in the src folder of the newly created netbeans project

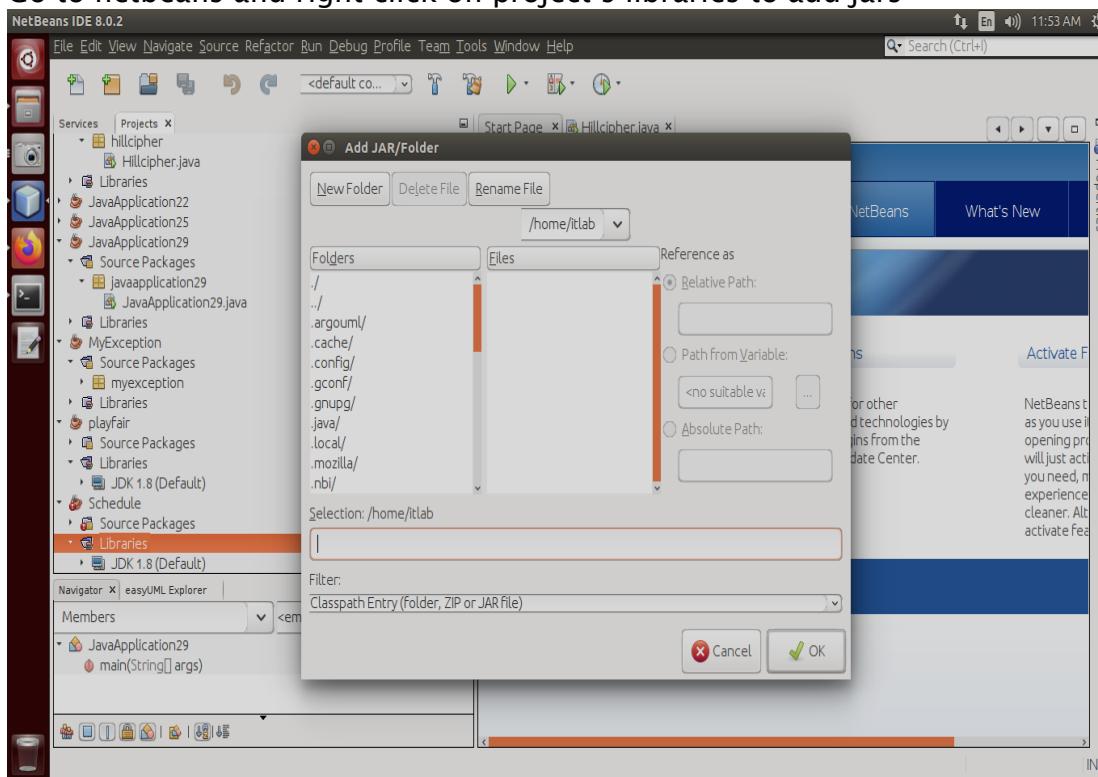


7. Also copy the cloudsim-task-scheduling.iml to the netbeans project

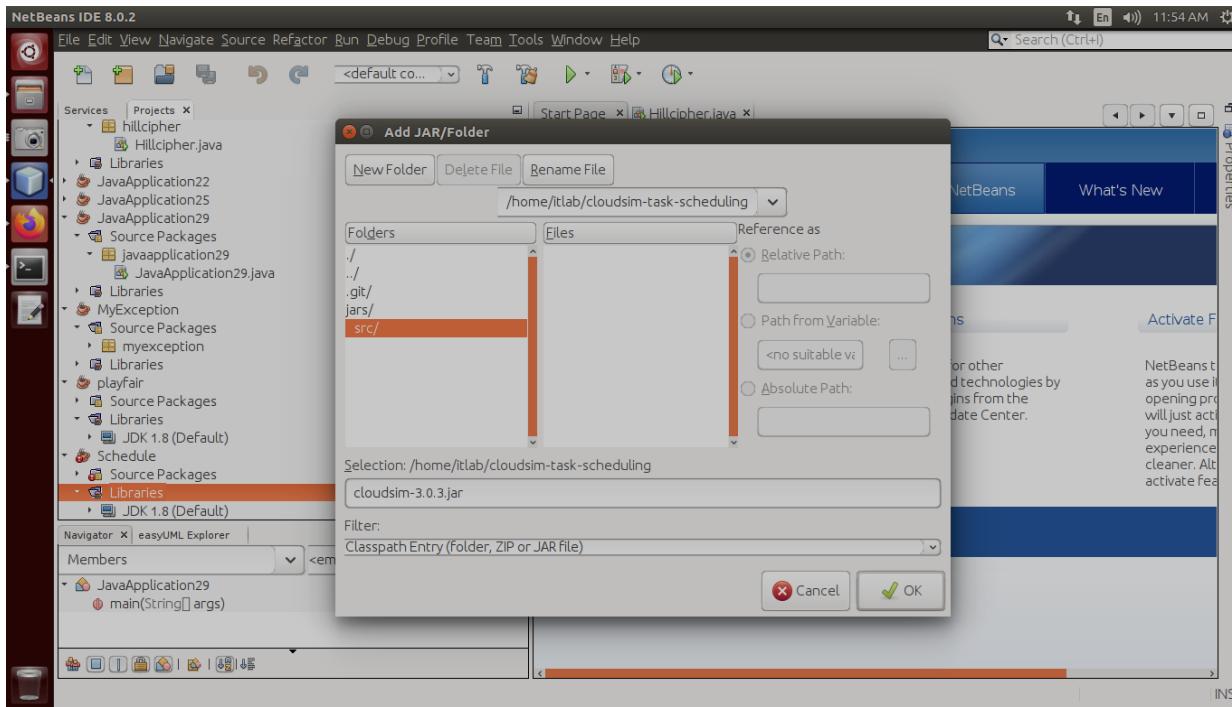




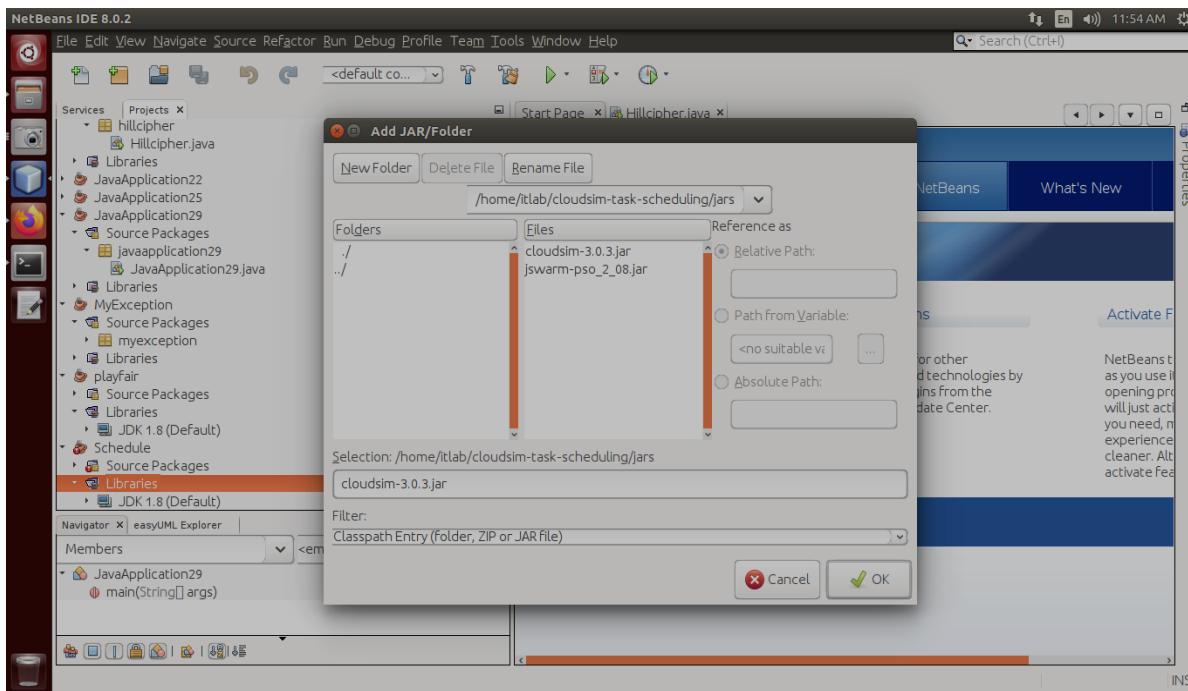
8. Go to netbeans and right click on project's libraries to add jars



Select the folder where you cloned the git project and go to jars



And double click



Select the jar file - cloudsim-3.0.3.jar and click Ok

Similarly add jswarm-psos_2_08.jar

9. To run any scheduling , expand the folder and right click on the file ending on _scheduler and run file
For FCFS, run FCFS_Scheduler.java

Program:

```
package org.cloudbus.cloudsim.examples;

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;

/***
 * An example showing how to create
 * scalable simulations.
 */
```

```

public class CloudSimExample6 {

    /** The cloudlet list. */

    private static List<Cloudlet> cloudletList;

    /** The vmlist. */

    private static List<Vm> vmlist;

    private static List<Vm> createVM(int userId, int vms) {

        //Creates a container to store VMs. This list is passed to the broker later
        LinkedList<Vm> list = new LinkedList<Vm>();

        //VM Parameters

        long size = 10000; //image size (MB)
        int ram = 512; //vm memory (MB) int
        mips = 1000;

        long bw = 1000;

        int pesNumber = 1; //number of cpus
        String vmm = "Xen"; //VMM name

        //create VMs

        Vm[] vm = new Vm[vms];

        for(int i=0;i<vms;i++){

            vm[i] = new Vm(i, userId, mips, pesNumber, ram, bw, size, vmm, new
CloudletSchedulerTimeShared());

            //for creating a VM with a space shared scheduling policy for cloudlets:
            //vm[i] = Vm(i, userId, mips, pesNumber, ram, bw, size, priority, vmm, new
CloudletSchedulerSpaceShared());

            list.add(vm[i]);
        }
    }
}

```

```

        return list;
    }

private static List<Cloudlet> createCloudlet(int userId, int cloudlets){

    // Creates a container to store Cloudlets
    LinkedList<Cloudlet> list = new LinkedList<Cloudlet>();

    //cloudlet parameters
    long length = 1000;
    long fileSize = 300;
    long outputSize = 300;
    int pesNumber = 1;

    UtilizationModel utilizationModel = new UtilizationModelFull();

    Cloudlet[] cloudlet = new Cloudlet[cloudlets];

    for(int i=0;i<cloudlets;i++){

        cloudlet[i] = new Cloudlet(i, length, pesNumber, fileSize, outputSize,
        utilizationModel, utilizationModel, utilizationModel);

        // setting the owner of these Cloudlets
        cloudlet[i].setUserId(userId);

    }

    return list;
}

//////////////// STATIC METHODS ///////////
/*
(
    * Creates main() to run this example
*/
public static void main(String[] args) {
    Log.println("Starting
    CloudSimExample6...");

    try {
    } // First step: Initialize the CloudSim package. It should be called
    // before creating any entities.

}

```

```

i           ndar calendar =
n           Calendar.getInstance(); boolean
t           trace_flag = false; // mean trace
events
n
u
m           // Initialize the CloudSim library
-           CloudSim.init(num_user, calendar,
u           trace_flag);
s
e
r           // Second step: Create Datacenters
=           //Datacenters are the resource providers in CloudSim. We need
at list one of them to run a CloudSim simulation
1
;
@SuppressWarnings("unused")
/
/
Datacenter datacenter0 =
createDatacenter("Datacenter_0");
@SuppressWarnings("unused")
n           Datacenter datacenter1 = createDatacenter("Datacenter_1");
u
m
b           //Third step: Create Broker
e           DatacenterBroker broker =
r           createBroker(); int brokerId =
o           broker.getId();
f

g           //Fourth step: Create VMs and Cloudlets and send
r           them to broker
i           vmlist = createVM(brokerId,20);
d           //creating 20 vms
cloudletList = createCloudlet(brokerId,40); // creating 40 cloudlets
u
s           broker.submitVmList(vmlist);
e
r           broker.submitCloudletList(cloudletList);
s
r           // Fifth step: Starts
s           the simulation
C           CloudSim.startSimu
a           lation();
l
e

```

```

/
    s over List<Cloudlet> newList =
/
    broker.getCloudletReceivedList();

F
i      CloudSim.stopSimulation();
n
a
l      printCloudletList(newList);

s      Log.println("CloudSimExample"
t      6 finished!");
e
p      }
:
catch (Exception e)

P      {
r
i
n      e.printStackTrace();
t
Log.println("The simulation has been terminated due to an unexpected
r
}
e

// 3. Create PEs and add these into the list.

u
l      //for a quad-core machine, a list of 4 PEs is required:
t
s      peList1.add(new Pe(0, new PeProvisionerSimple(mips))); // need to store
Pe id and

```

MIPS Rating

```

h
peList1.add(new Pe(1, new PeProvisionerSimple(mips)));
peList1.add(new Pe(2, new PeProvisionerSimple(mips)));
peList1.add(new Pe(3, new PeProvisionerSimple(mips)));

s
i
m
u
l
a
t
i
o
n

i

```

```

//Another list, for a dual-core machine
List<Pe> peList2 = new ArrayList<Pe>();

peList2.add(new Pe(0, new PeProvisionerSimple(mips)));
peList2.add(new Pe(1, new PeProvisionerSimple(mips)));

//4. Create Hosts with its id and list of PEs and add them to the list of machinesint
hostId=0;

int ram = 2048; //host memory (MB) long
storage = 1000000; //host storageint bw =
10000;

hostList.add( new Host( hostId,
                        new
                        RamProvisionerSim
                        ple(ram),new
                        wProvisionerSimpl
                        e(bw), storage,
                        peList1,
                        new VmSchedulerTimeShared(peList1)
                      )
);

// This is our
first machine

hostId++;
}

private static Datacenter createDatacenter(String name){

// Here are the steps needed to create a PowerDatacenter:

// 1. We need to create a list to store one or more
// Machines

List<Host> hostList = new ArrayList<Host>();

// 2. A Machine contains one or more PEs or CPUs/Cores. Therefore,
should

// create a list to store these PEs before creating
// a Machine.

```

```

L      yList<Pe>();int mips=
i          1000;

s      hostList.add(
t          new Host(
<              hostId,
P              new
e                  RamProvisionerSim
>                  ple(ram),new
BwProvisionerSimp
le(bw), storage,
peList2,
p          new VmSchedulerTimeShared(peList2)
e      )
L      ); // Second machine
i      //To create a host with a space-shared allocation policy for PEs to VMs:
s
t          //hostList.add(
l          //      new Host(
//              hostId,
=          //      new CpuProvisionerSimple(peList1),
//              new RamProvisionerSimple(ram),
//              new BwProvisionerSimple(bw),
n          //              storage,
e          //              new VmSchedulerSpaceShared(peList1)
w          //          )
//      );
A
r      //To create a host with a oportunistic space-shared allocation policy for
r          PEs to VMs:
a          //hostList.add(

```

```

//           new Host // Create a DatacenterCharacteristics object that stores the
//           hostId properties of a data center: architecture, OS, list of
//           Resource //   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";
//           new VmSchedulerOportunisticSpaceShared(peList1)
//           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 = 0.1; // the cost of using storage in this resource
double costPerBw = 0.1; // the cost of using bw in this
/

```

5

```
LinkedList<Storage> storageList = new LinkedList<Storage>();
```

```
C
r
```

```
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;
}
```

//We strongly encourage users to develop their own broker policies, to submit vms andcloudlets according

```

//to the specific rules of the simulated scenario private
static DatacenterBroker createBroker(){

    DatacenterBroker broker = null;
    try {

        broker = new DatacenterBroker("Broker");

    } catch (Exception e) {

        e.printStackTrace();
        return null;

    }return broker;

}

/**
 * Prints the Cloudlet objects
 * @param list list of Cloudlets
 */

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 + 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() +

```

```

dft.format(cloudlet.getActualCPUTime()) + indent + indent + indent +
indent + indent + dft.format(cloudlet.getExecStartTime())+ indent + indent +
indent +dft.format(cloudlet.getFinishTime()));
}
}
}
}
```

OUTPUT :

```
Schedule - NetBeans IDE 8.0.2
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help
Output - Schedule (rpn)
Datacenter_1 is shutting down...
Datacenter_2 is shutting down...
Datacenter_3 is shutting down...
Datacenter_4 is shutting down...
Broker_0 is shutting down...
Simulation completed.

=====
Cloudlet ID STATUS Data center ID VM ID Time Start Time Finish Time
00 SUCCESS 02 02 1891.67 09.2 1891.87
01 SUCCESS 02 03 2537.03 09.2 2537.29
02 SUCCESS 03 06 2060.92 09.2 2060.97
03 SUCCESS 02 05 1702.66 09.2 1702.86
04 SUCCESS 02 03 2089.98 2537.23 4627.21
05 SUCCESS 02 05 4135.88 1702.85 5885.74
06 SUCCESS 03 06 2700.98 2060.87 4797.20
07 SUCCESS 03 06 2050.39 4797.21 6847.6
08 SUCCESS 03 06 2489.52 6847.6 8773.12
09 SUCCESS 02 04 2489.52 09.2 2490.11
10 SUCCESS 02 04 1146.58 2490.11 3636.69
11 SUCCESS 02 04 3356.47 3636.69 6993.16
12 SUCCESS 03 06 2385.93 6993.16 11381.9
13 SUCCESS 02 03 1226.38 4627.21 5853.6
14 SUCCESS 02 04 2018.62 6993.16 9011.78
15 SUCCESS 02 03 1302.85 5853.6 7755.08
16 SUCCESS 02 05 1551.92 5858.74 7930.66
17 SUCCESS 02 04 1506.95 9011.78 10518.72
18 SUCCESS 02 04 1539.95 1539.95 12029.78
19 SUCCESS 02 02 1996.46 1891.72 3888.32
20 SUCCESS 02 03 2109.49 7755.6 9865.09
21 SUCCESS 02 04 2097.49 12029.78 14047.54
22 SUCCESS 02 03 2717.99 9865.09 12583.08
23 SUCCESS 02 05 2662.71 7930.66 10053.36
24 SUCCESS 02 05 2241.99 10053.36 12220.08
25 SUCCESS 03 06 3105.89 11138.3 14244.8
26 SUCCESS 02 04 4097.36 14094.54 18191.89
27 SUCCESS 02 02 2622.66 3888.32 6711.18
28 SUCCESS 02 03 3310.98 12583.08 15889.46
29 SUCCESS 02 03 1497.01 15899.46 17981.27

Makespan using FCFS: 4724.949185867268
FCFS_Scheduler finished!
BUILD SUCCESSFUL (total time: 0 seconds)
```

```
Starting PSO Scheduler...
Reading the metrics...
Gloabl best at iteration (0): 7630.060628
Gloabl best at iteration (10): 6754.056424
Gloabl best at iteration (20): 6754.446232
Gloabl best at iteration (30): 6754.474742
Gloabl best at iteration (40): 5761.584259
Gloabl best at iteration (50): 5748.730665
Gloabl best at iteration (60): 5748.730665
Gloabl best at iteration (70): 5748.347002
Gloabl best at iteration (80): 5748.347002
Gloabl best at iteration (90): 5748.347002
Gloabl best at iteration (100): 5748.347002
Gloabl best at iteration (110): 5748.347002
Gloabl best at iteration (120): 5739.491267
Gloabl best at iteration (130): 5709.742746
Gloabl best at iteration (140): 5709.742746
Gloabl best at iteration (150): 5709.742746
Gloabl best at iteration (160): 5709.742746
Gloabl best at iteration (170): 5709.742746
Gloabl best at iteration (180): 5709.742746
Gloabl best at iteration (190): 5709.742746
Gloabl best at iteration (200): 5709.742746
Gloabl best at iteration (210): 5709.742746
Gloabl best at iteration (220): 5709.742746
Gloabl best at iteration (230): 5709.742746
Gloabl best at iteration (240): 5709.742746
Gloabl best at iteration (250): 5709.742746
Gloabl best at iteration (260): 5709.742746
Gloabl best at iteration (270): 5709.742746
Gloabl best at iteration (280): 5709.742746
Gloabl best at iteration (290): 5709.742746
Gloabl best at iteration (300): 5709.742746
Gloabl best at iteration (310): 5709.742746
Gloabl best at iteration (320): 5709.742746
Gloabl best at iteration (330): 5709.742746
Gloabl best at iteration (340): 5709.742746
Gloabl best at iteration (350): 5709.742746
```

Result:

Thus the study on cloudsim simulation tool is done successfully.

EX:NO: 6
DATE:

PROCEDURE TO LAUNCH VIRTUAL MACHINE USING TRYSTACK

Aim:

To understand the procedure to launch virtual machine using trystack.

Description:

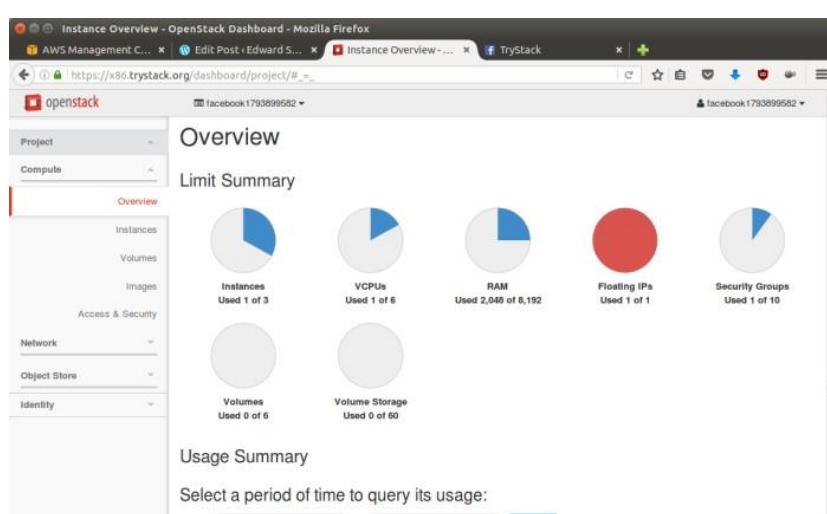
OpenStack is an open-source software cloud computing platform. OpenStack is primarily used for deploying an infrastructure as a service (IaaS) solution like Amazon Web Service (AWS). In other words, you can *make your own AWS* by using OpenStack.

Procedure:

1. Open trystack page using trystack.org



2. Login to trystack by creating account.



3. Create Network

- a) Go to Network > Networks and then click Create Network.
- b) In Network tab, fill Network Name for example internal and then click Next.
- c) In Subnet tab,
- d) Fill Network Address with appropriate CIDR, for example 192.168.1.0/24. Use private network CIDR block as the best practice.
- e) Select IP Version with appropriate IP version, in this case IPv4.
- f) Click Next.
- g) In Subnet Details tab, fill DNS Name Servers with 8.8.8.8 (Google DNS) and then click Create.

4. Create Instance

1. Go to Compute > Instances and then click Launch Instance.
2. 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.
3. 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 (usually is in ~/.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.
4. In Networking tab,
 - In Selected Networks, select network that have been created in Step 1, for example internal.
5. Click Launch.
6. If you want to create multiple instances, you can repeat step 1-5. I created one more instance with instance name Ubuntu 2.
7. Create Router
 - 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.

- a) Click Add Interface button.
 - Select Subnet with the network that you have been created in Step 1.
 - Click Add interface.
- b) 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.

6. 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 are collected in a pool and managed by admin.

- a) Go to Compute > Instance.
- b) In one of your instances, click More > Associate Floating IP.
- c) In IP Address, click Plus [+].
- d) Select Pool to external and then click Allocate IP.
- e) Click Associate.
- f) Now you will get a public IP, e.g. 8.21.28.120, for your instance.

7. Configure Access & Security

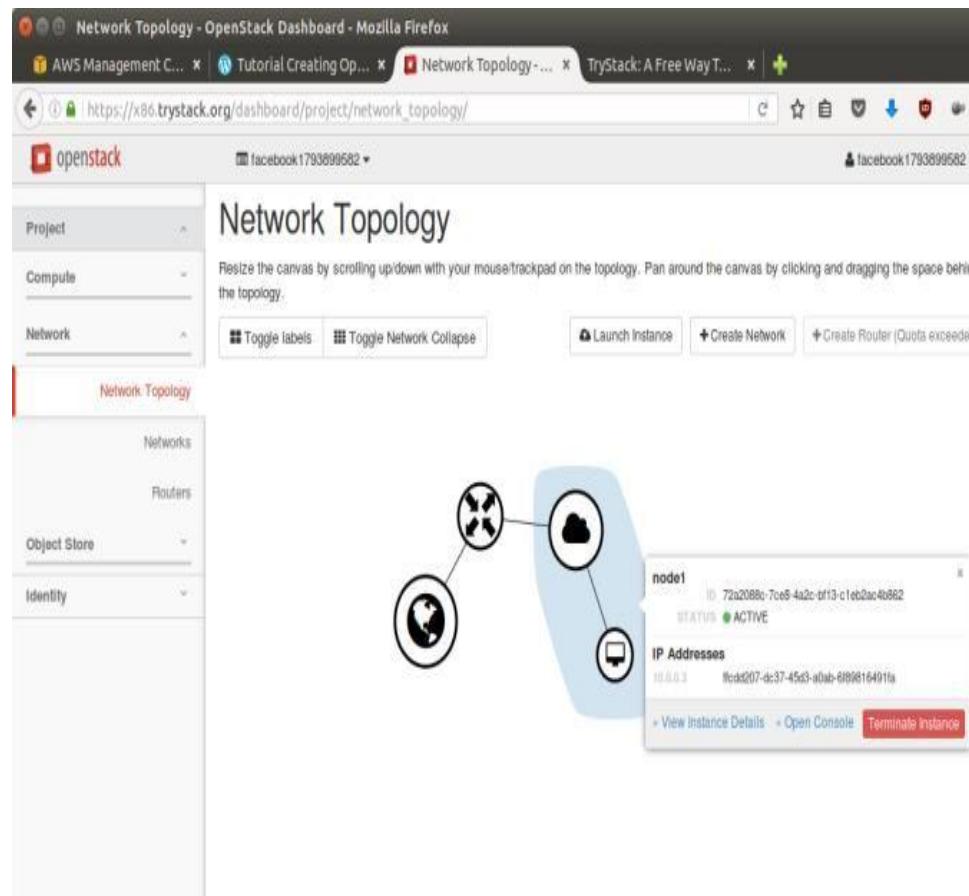
OpenStack has a feature like a firewall. It can whitelist/blacklist your in/out connection.

It is called *Security Group*.

- a) Go to Compute > Access & Security and then open Security Groups tab.
- b) In default row, click Manage Rules.
- c) Click Add Rule, choose ALL ICMP rule to enable ping into your instance, and then click Add.
- d) Click Add Rule, choose HTTP rule to open HTTP port (port 80), and then click Add.
- e) Click Add Rule, choose SSH rule to open SSH port (port 22), and then click Add.
- f) You can open other ports by creating new rules.

8. 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.

OUTPUT :

Result: Thus the procedure to launch virtual machine in trystack is studied

EXNO: 7**DATE:****Procedure File Transfer in Client & Server using virtual machine.****Aim:**

To procedure File Transfer in Client & Server using virtual machine

Steps:

Steps to perform File Transfer in Client & Server using virtual machine.

Step 1: Open a virtual machine to do file transfer.

Step 2: Write the java program for FTP Client and FTP Server.

Step 3: Run the program.

Source Code:

```
FTPClient.java
import java.io.*;
import java.net.*;
import java.util.*;
public class FTPClient{
    public static void main(String args[])throws IOException {
        try {
            int number;
            Socket s=new Socket("127.0.0.1",10087);
            Scanner sc=new Scanner(System.in);
            System.out.println("Enter the file name:");
            String fn=sc.next();
            DataOutputStream dos=new DataOutputStream(s.getOutputStream());
            dos.writeUTF(fn);
            DataInputStream dis=new DataInputStream(s.getInputStream());
            String input=(String)dis.readUTF();
            FileInputStream fis=new FileInputStream(input);
            System.out.println("Even Numbers in the " +fn+ " are");
            int i=0;
            while((i=fis.read())!=-1){
                System.out.println((char)i);
            }
            s.close();
        }
        catch(Exception e){
            System.out.println("Port not available "+e);
        }
    }
}
```

```
FTPServer.java

import java.io.*;
import java.net.*;
import java.util.*;

public class FTPServer{

    public static void main(String args[])throws IOException{
        try{
            int num;
            Scanner sc=new Scanner(System.in);
            ServerSocket ss=new ServerSocket(10087);
            Socket s=ss.accept();
            System.out.println("Waiting. ... ");
            DataInputStream dis=new DataInputStream(s.getInputStream());
            String input=(String)dis.readUTF();
            DataOutputStream dos=new DataOutputStream(s.getOutputStream());
            FileInputStream fis = new FileInputStream("out.txt");
            FileOutputStream fos = new FileOutputStream(input);
            while((num=fis.read())!= -1) {
                if(num%2==0) {
                    fos.write(num);
                }
            }
            dos.writeUTF(input);
            System.out.println("File is sent to client");
            ss.close();
            s.close();
        }
    }
}
```

```
}

catch(Exception e) {

System.out.println("Port not available"+e);

} } }
```

Out.txt

```
1  
2  
3  
4  
5  
6  
7  
8  
9
```

Output:

The image shows two windows from a Windows command-line interface (cmd.exe). Both windows are running on Microsoft Windows [Version 6.1.7601].

The left window (FTP Server) shows the following command sequence:

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\      .d:
D:\>cd      \Programs\FTP
D:\      Programs\FTP>javac FTPServer.java
D:\      Programs\FTP>java FTPServer
Waiting.....
File is sent to client
D:\      \Programs\FTP>
```

The right window (FTP Client) shows the following command sequence:

```
C:\Windows\system32\cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Users\      >d:
D:\>cd      \Programs\FTP
D:\      \Programs\FTP>javac FTPClient.java
D:\      \Programs\FTP>java FTPClient
Enter the file name:
out.txt
Even Numbers in theout.txt are

2
4
6
8
```

Result:

Thus the program to the File transfer operation using virtual machine was successfully executed and verified.

EX.NO: 8 Hadoop Single Node ClusterDate:**DATE:****Aim :**

To find procedure to set up the one node Hadoop cluster.

Procedure :

1. Install openssh server

```
sudo apt-get install openssh-server
```

2. Create the new group

```
sudo addgroup <groupname>
```

eg:<groupname> cluster

3. Create the new user in newly created group

```
sudo adduser --ingroup <groupname> <username>
```

eg: <username> hadoop

4. Login to newly created user.

5. Generate the SSH key

```
ssh-keygen -t rsa -P " -f ~/.ssh/id_rsa
```

```
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

```
ssh localhost
```

6. Unzip the JDK and Hadoop files

```
tar -xzf jdk-8u45-linux-x64.tar.gz
```

```
tar -xzf hadoop-2.6.0.tar.gz
```

7. Open the „bashrc“ file

```
gedit .bashrc
```

8. Type the following command in the „bashrc“ file bottom.

```
//add the following at the end of file
```

```
export JAVA_HOME=jdk1.8.0_45
```

```
export HADOOP_HOME=hadoop-2.6.0
```

```
export PATH=$PATH:$JAVA_HOME/bin:$HADOOP_HOME/bin:$HADOOP/sbin
```

```
export PATH
```

9. Execute the „bashrc“ file

```
exec bash
```

```
source .bashrc
```

```
hadoop version
```

10. Open the „hadoop-env.sh“ file and add add java home dir or java folder path

```
gedit hadoop-env.sh
```

11. Copy and Paste the following file to hadoop folder-->etc-->hadoop

```
vim core-site.xml
```

```
<configuration>
```

```
  <property>
```

```
    <name>fs.default.name</name>
```

```
    <value>hdfs://localhost:9000</value>
```

```
  </property>
```

```
</configuration>
```

```
vim mapred-site.xml
```

```
<configuration>
```

```
  <property>
```

```
    <name>mapreduce.framework.name</name>
```

```
    <value>yarn</value>
```

```
  </property>
```

```
  <property>
```

```
    <name>mapreduce.job.tracker</name>
```

```
    <value>localhost:54311</value>
```

```
  </property>
```

```
  <property>
```

```
    <name>mapreduce.tasktracker.map.tasks.maximum</name>
```

```
<value>4</value>
</property>
<property>
<name>mapreduce.map.tasks</name>
<value>4</value>
</property>
</configuration>

vim hdfs-site.xml // to edit the username in this file

<configuration>
<property>
<name>dfs.replication</name>
<value>3</value>
</property>
<property>
<name>dfs.namenode.name.dir</name>
<value>file:/home/hduser/hdfs/namenode</value>
</property>
<property>
<name>dfs.datanode.data.dir</name>
<value>file:/home/hduser/hdfs/datanode</value>
</property>
</configuration>
```

12. Format the hadoop namenode

Hadoop/hdfs namenode –format

13. Start the hadoop

start-all.sh (or) hadoop-2.6.0/sbin/start-all.sh

14. Check the nodes that are created or not

15. If name node is not created execute the below command and repeat step 12 & 13

```
rm -r hdfs/
```

16. Open the namenode and datanode. In browser type the following port number

localhost:50070

localhost:8088

Output:

NameNode:

DataNode:

Result:

Thus the procedure to set up the one node Hadoop cluster is successfully completed.

EX.NO: 8 A**WORD COUNT PROGRAM****DATE:****Aim:**

To write a word count program to demonstrate the use of Map and Reduce tasks

Procedure:

1. After created one node hadoop cluster. Create the HDFS input directory.

```
hdfs dfs -mkdir /inp
```

2. Copy the input file to created input directory

```
hdfs dfs -copyFromLocal hadoop-2.6.0/etc/hadoop/a.txt /inp
```

3. if name node is running in safemode to execute the bellow command

```
hadoop dfsadmin -safemode leave
```

4. Execute hadoop mapreduce JAR file and specify an input & output directory.

```
hadoop jar hadoop-2.6.0/share/hadoop/mapreduce/hadoop-mapreduce-examples-2.6.0.jar  
wordcount /inp /out
```

5. Open the namenode and datanode. In browser type the following port number.

localhost:50070

localhost:8088

6. Download the file from output directory and view the wordcout program output.

Program:

```
import java.io.IOException;  
  
import java.util.*;  
  
import org.apache.hadoop.conf.*;  
  
import org.apache.hadoop.fs.*;  
  
import org.apache.hadoop.conf.*;  
  
import org.apache.hadoop.io.*;  
  
import org.apache.hadoop.mapreduce.*;  
  
import org.apache.hadoop.mapreduce.lib.input.*;
```

```
import org.apache.hadoop.mapreduce.lib.output.*;
import org.apache.hadoop.util.*;
public class WordCount extends Configured implements Tool {
    public static void main(String args[]) throws Exception {
        int res = ToolRunner.run(new WordCount(), args);
        System.exit(res);
    }
    public int run(String[] args) throws Exception {
        Path inputPath = new Path(args[0]);
        Path outputPath = new Path(args[1]);
        Configuration conf = getConf();
        Job job = new Job(conf, this.getClass().toString());
        FileInputFormat.setInputPaths(job, inputPath);
        FileOutputFormat.setOutputPath(job, outputPath);
        job.setJobName("WordCount");
        job.setJarByClass(WordCount.class);
        job.setInputFormatClass(TextInputFormat.class);
        job.setOutputFormatClass(TextOutputFormat.class);
        job.setMapOutputKeyClass(Text.class);
        job.setMapOutputValueClass(IntWritable.class);
        job.setOutputKeyClass(Text.class);
        job.setOutputValueClass(IntWritable.class);
        job.setMapperClass(Map.class);
        job.setCombinerClass(Reduce.class);
        job.setReducerClass(Reduce.class);
        return job.waitForCompletion(true) ? 0 : 1;
    }
}
```

```
public static class Map extends Mapper<LongWritable, Text, Text, IntWritable> {  
    private final static IntWritable one = new IntWritable(1);  
    private Text word = new Text();  
    public void map(LongWritable key, Text value,  
                    Mapper.Context context) throws IOException, InterruptedException {  
        String line = value.toString();  
        StringTokenizer tokenizer = new StringTokenizer(line);  
        while (tokenizer.hasMoreTokens()) {  
            word.set(tokenizer.nextToken());  
            context.write(word, one);  
        } } }  
  
public static class Reduce extends Reducer<Text, IntWritable, Text, IntWritable> {  
    public void reduce(Text key, Iterable<IntWritable> values, Context context) throws  
                      IOException, InterruptedException {  
        int sum = 0;  
        for (IntWritable value : values) {  
            sum += value.get();  
        }  
        context.write(key, new IntWritable(sum));  
    }  
}
```

NameNode:

Browsing HDFS - Mozilla Firefox
localhost:50070/explorer.html#/uma
Hadoop Overview Datanodes Snapshot Startup Progress Utilities

Browse Directory

Permission	Owner	Group	Size	Replication	Block Size	Name
-rw-r--r--	h1	supergroup	200 B	3	128 MB	1.txt

Hadoop, 2014.

Firefox automatically sends some data to Mozilla so that we can improve your experience. Choose What I Share

DataNode:

Mozilla Firefox
localhost:8088/cluster/app/application_1472623896056_0001
Logged in as: dhruv

hadoop

Application Overview

User: h1
Name: word count
Application Type: MAPREDUCE
Application Tags:
State: FINISHED
Final Status: SUCCEEDED
Started: 31-Aug-2016 11:46:56
Elapsed: 13sec
Tracking URL: History
Diagnostics:

Application Metrics

Total Resource Preempted: <memory:0, vCores:0>
Total Number of Non-AM Containers Preempted: 0
Total Number of AM Containers Preempted: 0
Resource Preempted from Current Attempt: <memory:0, vCores:0>
Number of Non-AM Containers Preempted from Current Attempt: 0
Aggregate Resource Allocation: 48382 MB-seconds, 25 vcore-seconds

ApplicationMaster

Attempt Number	Start Time	Node	Logs
1	31-Aug-2016 11:46:56	t1-HP-ProDesk-400-G1-SFF-8042	logs

Firefox automatically sends some data to Mozilla so that we can improve your experience. Choose What I Share

Input:**a.txt**

```
sss    sss  sss
ppp    ppp  ppp
ccc    ccc  ccc
```

Output:

```
sss    3
ppp    3
ccc    3
```

Result:

Thus the word count program to demonstrate the use of Map and Reduce tasks is successfully executed.