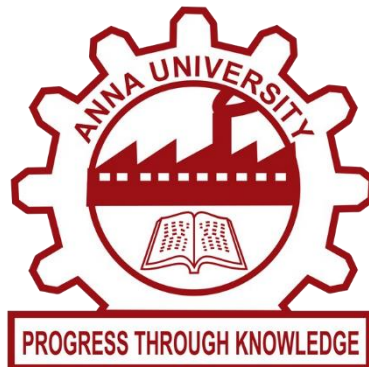


ANNA UNIVERSITY

REGIONAL CAMPUS, COIMBATORE



LABORATORY RECORD

2022 – 2023

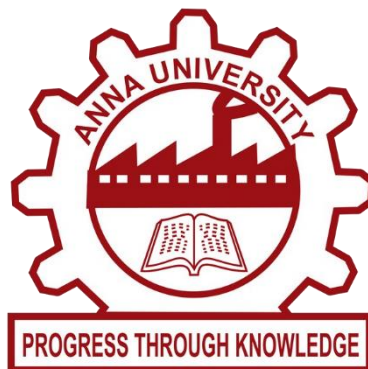
NAME :
REGISTRATION NUMBER :
BRANCH : Computer Science and Engineering
SUBJECT TITLE : Cloud Computing Laboratory
SUBJECT CODE : CS8711

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
ANNA UNIVERSITY REGIONAL CAMPUS
COIMBATORE 641 046

ANNA UNIVERSITY

REGIONAL CAMPUS, COIMBATORE

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING



BONAFIDE CERTIFICATE

Certified that this is the bonafide record of Practical done in **CS8711**
CLOUD COMPUTING LABORATORY by _____
of Registration Number _____ in Fourth Year / Seventh
Semester during 2022-2023.

STAFF IN-CHARGE

HEAD OF THE DEPARTMENT

University Registration Number : _____.

Submitted for the University Practical Examination held on : _____.

INTERNAL EXAMINER

EXTERNAL EXAMINER

TABLE OF CONTENTS

<i>Expt No</i>	<i>Date</i>	<i>Name of the Experiment</i>	<i>Page No</i>	<i>Signature</i>
1		Install Virtualbox / VMware Workstation with different flavours of linux or windows OS on top of windows 10 or 11.		
2		Install a C compiler in the virtual machine created using virtual box and execute Simple Programs		
3		Install Google App Engine. Create <i>hello world</i> app and other simple webapplications 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 trystack(Online Openstack Demo Version)		
8		Install Hadoop single node cluster and run simple applications like wordcount.		

Ex. No : 1
Date :

Installing Virtualbox / VMware Workstation with different flavours of linux or windows OS on top of windows 10 or 11.

AIM:

To Install Virtualbox / VMware Workstation with different flavours of linux or windows OS on top of windows 7 or 8.

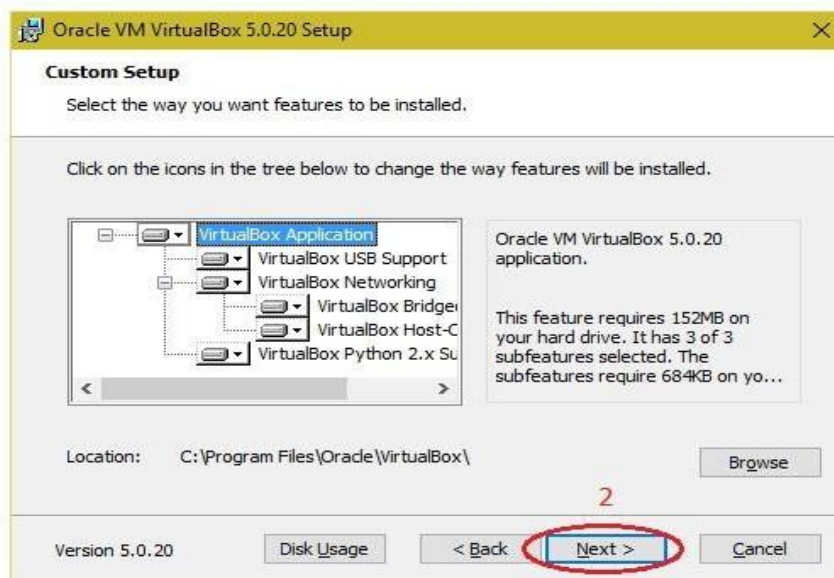
PROCEDURE:

Steps to install Virtual Box:

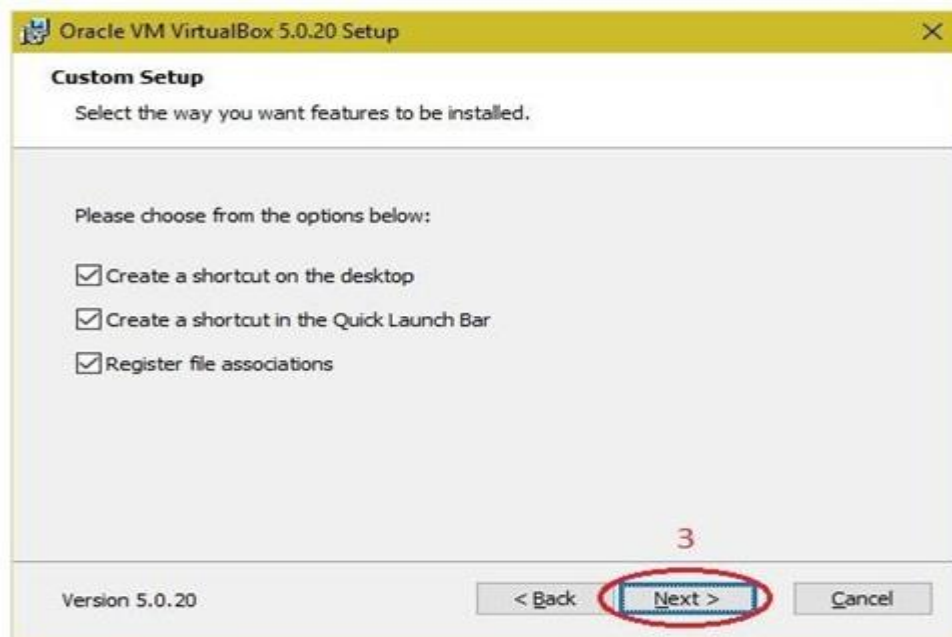
1. Download the Virtual box exe and click the exe file...and select next button.



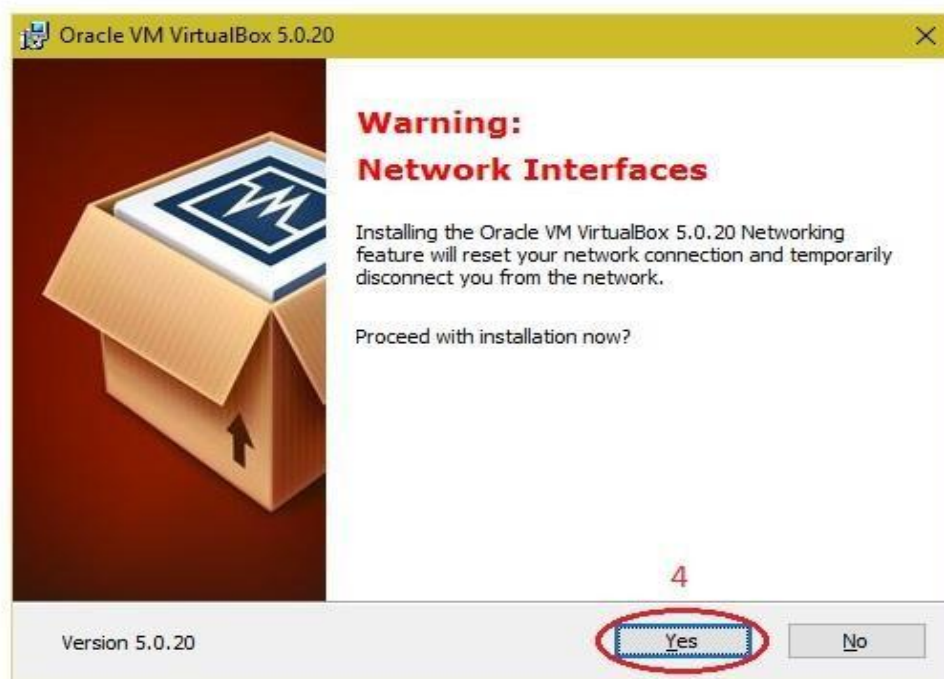
2. Click the next button.



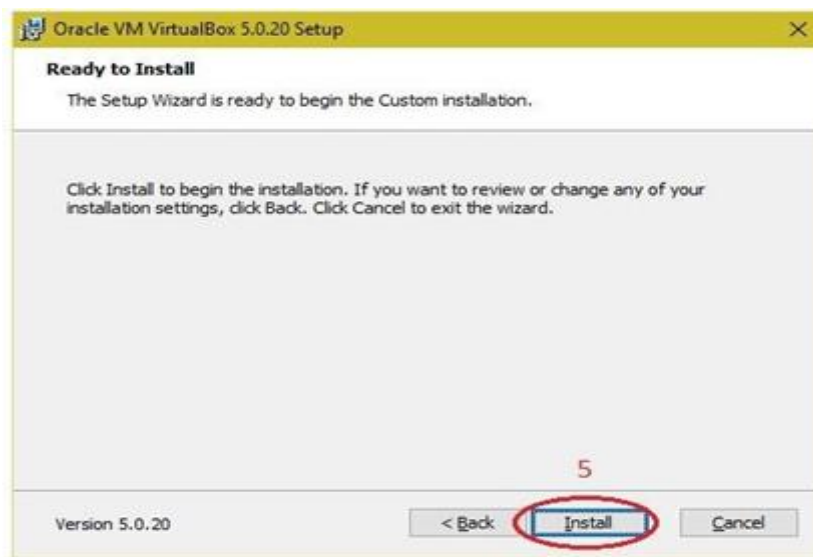
3. Click the next button.



4. Click the YES button..



5. Click the install button...



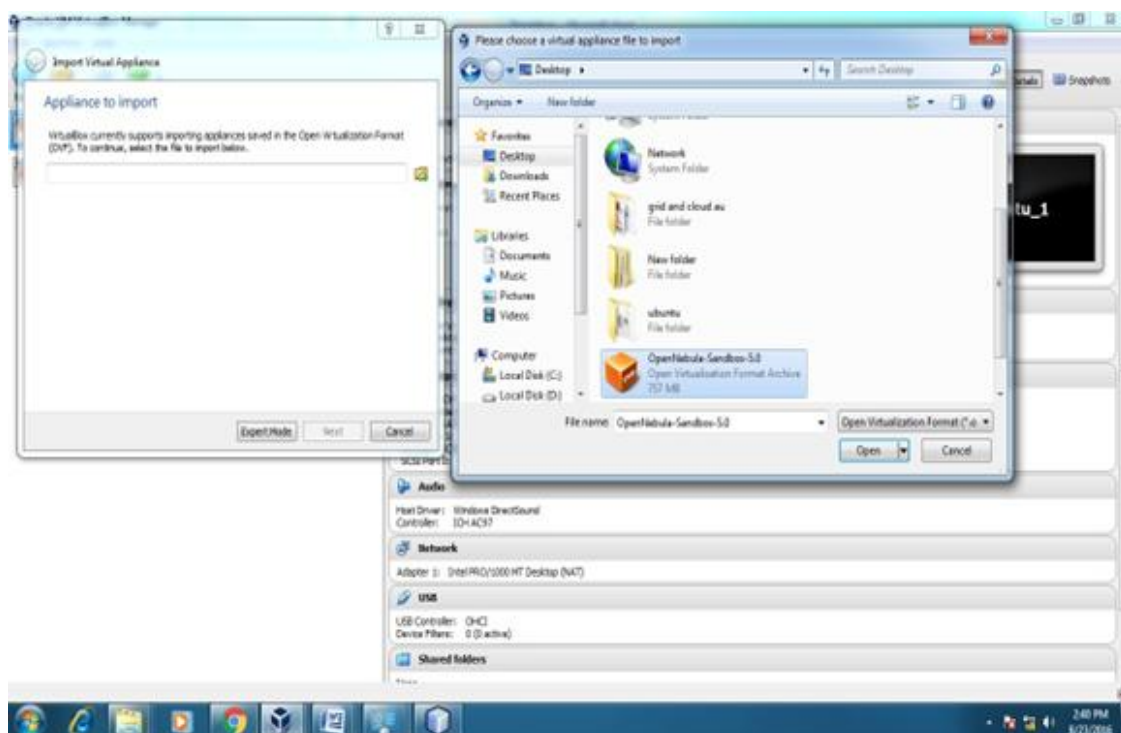
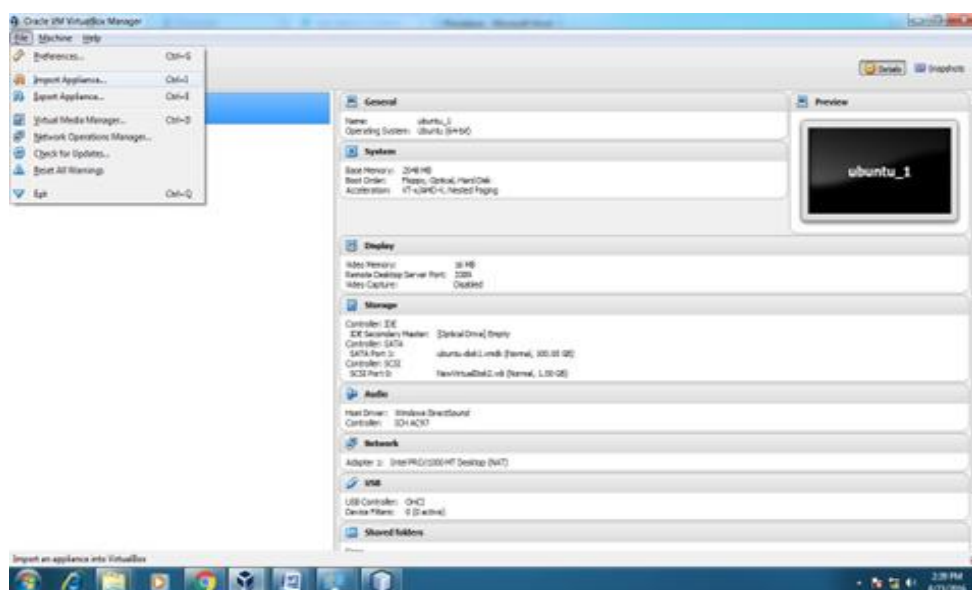
6. Then installation was completed..the show virtual box icon on desktop screen....



VirtualBox

Steps to import Open nebula sandbox:

1. Open Virtual box
2. File → import Appliance
3. Browse OpenNebula-Sandbox-5.0.ova file
4. Then go to setting, select Usb and choose USB 1.1
5. Then Start the Open Nebula
6. Login using username: root, password:opennebula



APPLICATIONS:

There are various applications of cloud computing in today's network world. Many search engines and social websites are using the concept of cloud computing like www.amazon.com, hotmail.com, facebook.com, linkedin.com etc. the advantages of cloud computing in context to scalability is like reduced risk , low cost testing, ability to segment the customer base and auto-scaling based on application load.

RESULT:

Thus the procedure to run the virtual machine of different configuration.

Ex. No : 2
Date :

Install a C compiler in the virtual machine created using virtual box and execute Simple Programs

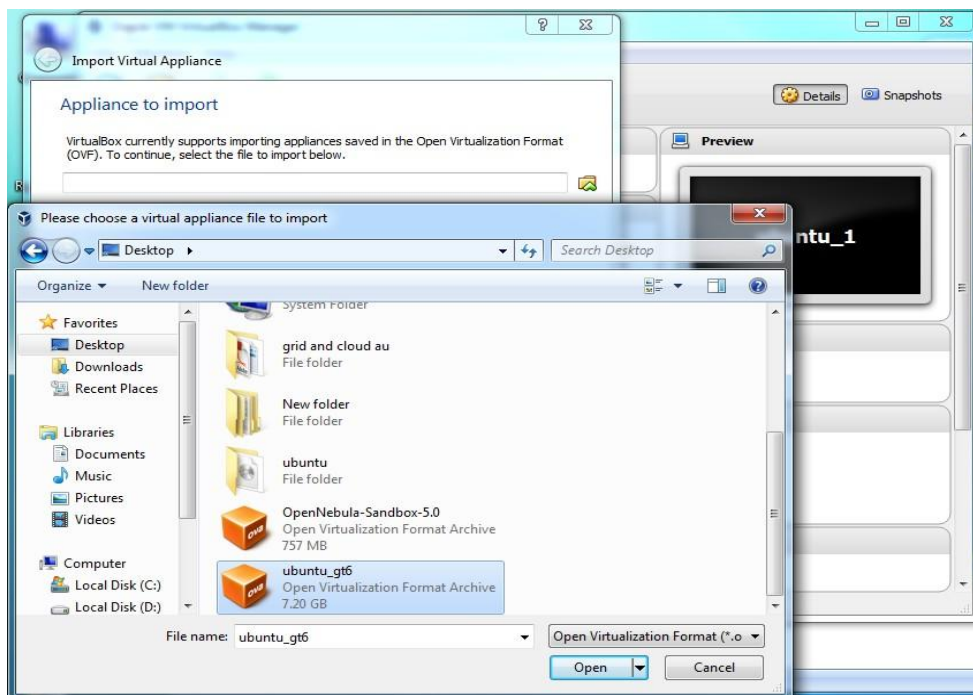
AIM:

To Install a C compiler in the virtual machine created using virtual box and execute Simple Programs`

PROCEDURE:

Steps to import .ova file:

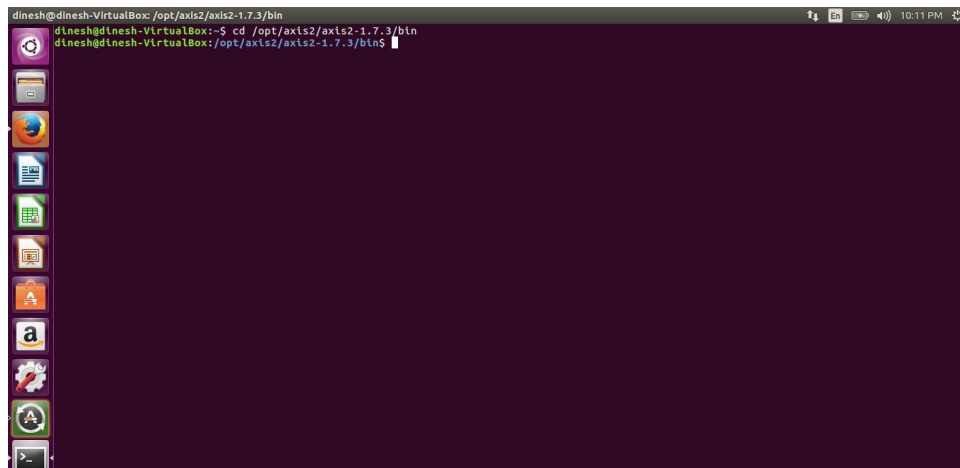
1. Open Virtual box
2. File → import Appliance
3. Browse ubuntu_gt6.ova file
4. Then go to setting, select Usb and choose USB 1.1
5. Then Start the ubuntu_gt6
6. Login using username: dinesh, password:99425.



Steps to run c program:

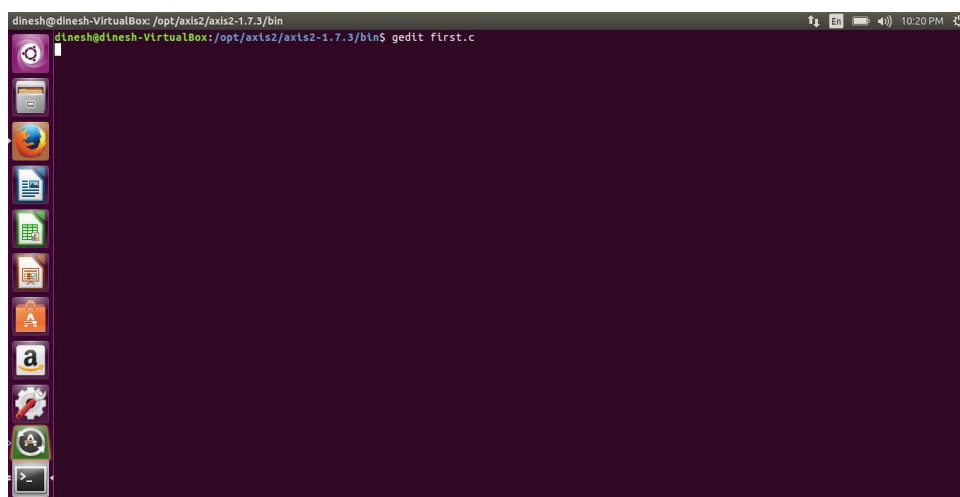
1. Open the terminal
2. Type `cd /opt/axis2/axis2-1.7.3/bin` then press enter
3. `gedit hello.c`
4. `gcc hello.c`
5. `./a.out`

1. Type `cd /opt/axis2/axis2-1.7.3/bin` then press enter

A screenshot of a terminal window in a VirtualBox environment. The window title is 'dinesh@dinesh-VirtualBox: /opt/axis2/axis2-1.7.3/bin'. The terminal shows the command `cd /opt/axis2/axis2-1.7.3/bin` being executed, and the prompt changes to `dinesh@dinesh-VirtualBox:/opt/axis2/axis2-1.7.3/bin$`. The left sidebar shows various application icons, and the top right corner displays the time as 10:11 PM.

```
dinesh@dinesh-VirtualBox: /opt/axis2/axis2-1.7.3/bin
dinesh@dinesh-VirtualBox:~$ cd /opt/axis2/axis2-1.7.3/bin
dinesh@dinesh-VirtualBox:/opt/axis2/axis2-1.7.3/bin$
```

2. Type `gedit first.c`

A screenshot of a terminal window in a VirtualBox environment. The window title is 'dinesh@dinesh-VirtualBox: /opt/axis2/axis2-1.7.3/bin'. The terminal shows the command `gedit first.c` being entered. The left sidebar shows various application icons, and the top right corner displays the time as 10:20 PM.

```
dinesh@dinesh-VirtualBox: /opt/axis2/axis2-1.7.3/bin
dinesh@dinesh-VirtualBox:/opt/axis2/axis2-1.7.3/bin$ gedit first.c
```

3. Type the c program

```

first.c (-/) - gedit
#include<stdio.h>
#include<conio.h>
void main()
{
    int a;
    clrscr();
    printf("Enter the number to find Even Or Not");
    scanf("%d",&a);
    if(a%2==0)
        printf("The Entered number is Even");
    else
        printf("The Entered number is Odd");
}

```

Saving File '/home/dinesh/first.c'...

C Tab Width: 8 Ln 13, Col 2 INS

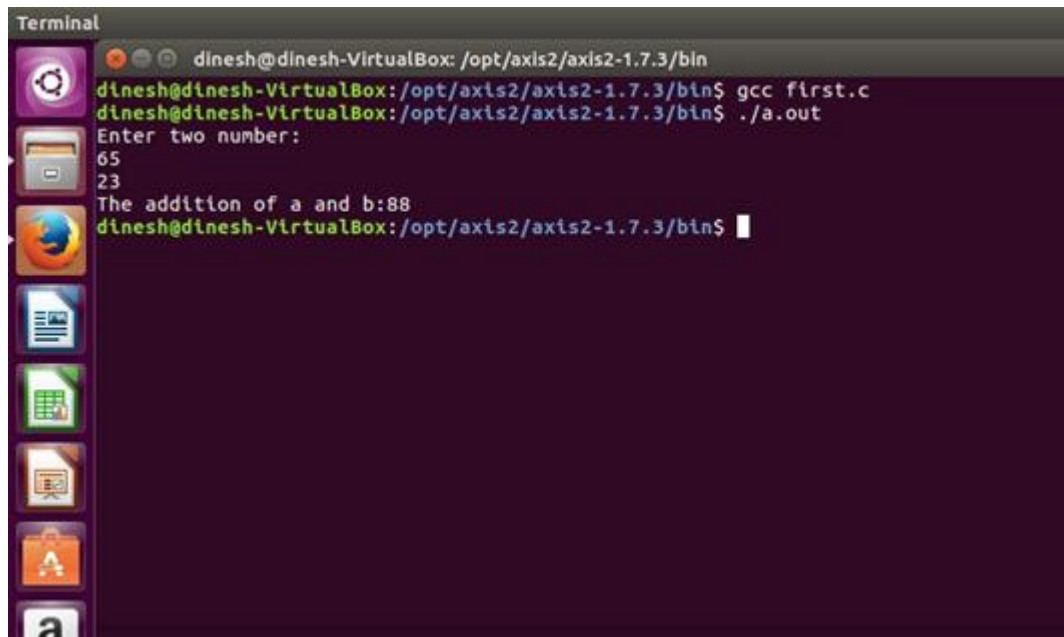
4. Running the C program

```

dinesh@dinesh-VirtualBox: /opt/axis2/axis2-1.7.3/bin
dinesh@dinesh-VirtualBox: /opt/axis2/axis2-1.7.3/bin$ gedit first.c
dinesh@dinesh-VirtualBox: /opt/axis2/axis2-1.7.3/bin$ gcc first.c
dinesh@dinesh-VirtualBox: /opt/axis2/axis2-1.7.3/bin$ ./a.out

```

5. Display the output:

A screenshot of a Linux terminal window titled 'Terminal'. The window shows a user named 'dinesh' at a machine named 'dinesh-VirtualBox'. The current directory is '/opt/axis2/axis2-1.7.3/bin'. The user has compiled a C program 'first.c' using 'gcc' and then executed the resulting binary 'a.out'. The program prompts for two numbers, '65' and '23', and outputs 'The addition of a and b:88'.

```
Terminal
dinesh@dinesh-VirtualBox: /opt/axis2/axis2-1.7.3/bin
dinesh@dinesh-VirtualBox:/opt/axis2/axis2-1.7.3/bin$ gcc first.c
dinesh@dinesh-VirtualBox:/opt/axis2/axis2-1.7.3/bin$ ./a.out
Enter two number:
65
23
The addition of a and b:88
dinesh@dinesh-VirtualBox:/opt/axis2/axis2-1.7.3/bin$
```

APPLICATIONS:

Simply running all programs in grid environment.

RESULT:

Thus the simple C programs executed successfully

Ex. No : 3
Date :

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

AIM:

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

PROCEDURE:

1. Install Google Plugin for Eclipse

Install the Google App Engine Java SDK together with “Google Plugin for Eclipse“, then go to step 2, Or else, get the Google App Engine Java SDK and extract it.

2. Create New Web Application Project

In Eclipse toolbar, click on the Google icon, and select “New Web Application Project...”

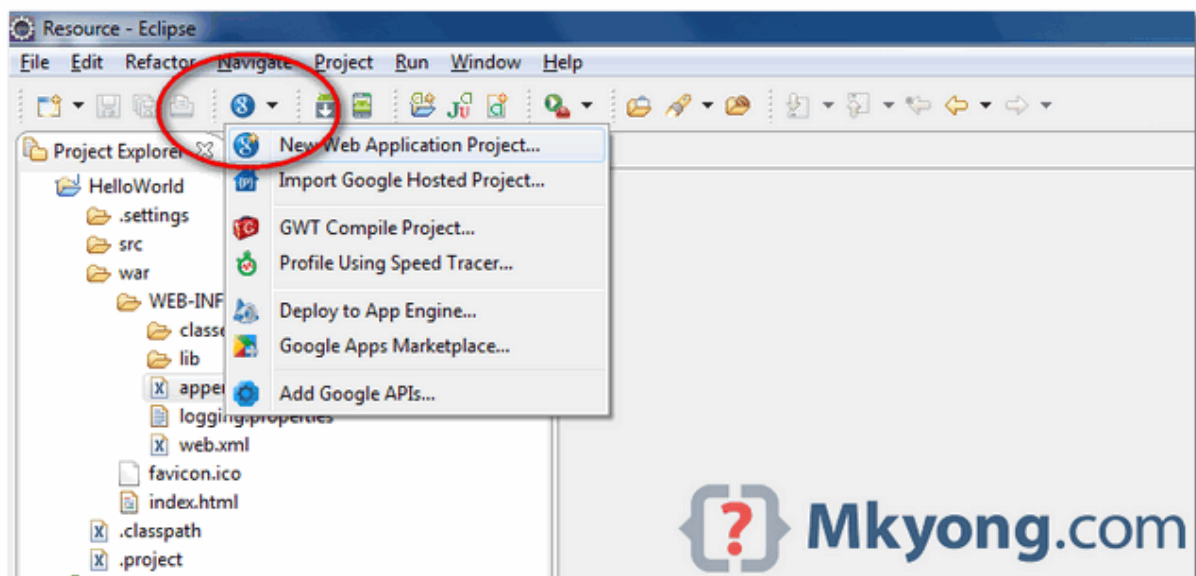
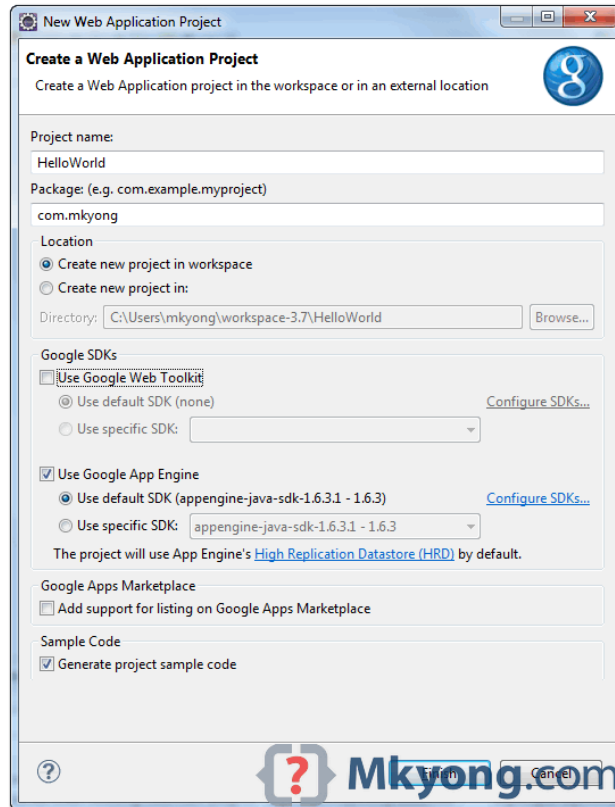


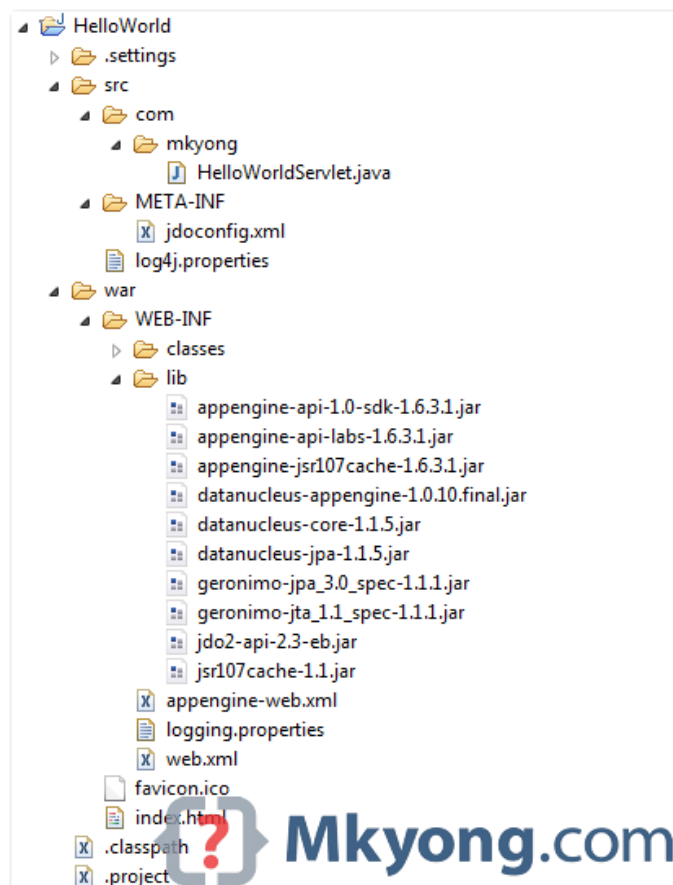
Figure – Deselect the “Google Web ToolKit“, and link your GAE Java SDK via the “configure SDK” link

Click finished, Google Plugin for Eclipse will generate a sample project automatically.



3. *Hello World*

Review the generated project directory.



HelloWorld/ src/
...Java source code... META-INF/
...other configuration... war/
...JSPs, images, data files... WEB-INF/
...app configuration... lib/
...JARs for libraries... classes/
...compiled classes...

Copy

The extra is this file “appengine-web.xml“, Google App Engine need this to run and deploy the application.

File : appengine-web.xml

```
<?xml version="1.0" encoding="utf-8"?>
<appengine-web-app xmlns="http://appengine.google.com/ns/1.0">
<application></application>
<version>1</version>
<!-- Configure java.util.logging -->
<system-properties>
<property name="java.util.logging.config.file" value="WEB-
INF/logging.properties"/>
</system-properties>
</appengine-web-app> Copy
```

4. Run it local

Right click on the project and run as “Web Application“.

Eclipse console :

//...

INFO: The server is running at http://localhost:8888/

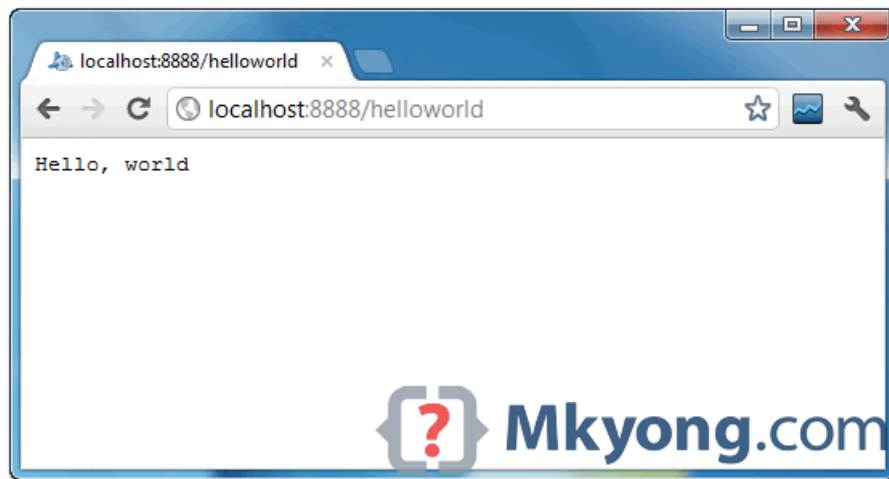
30 Mac 2012 11:13:01 PM

com.google.appengine.tools.development.DevAppServerImpl start INFO: The
admin console is running at http://localhost:8888/_ah/admin

Copy

Access URL http://localhost:8888/, see output

and also the hello world servlet – http://localhost:8888/helloworld



5. *Deploy to Google App Engine*

Register an account on <https://appengine.google.com/>, and create an application ID for your web application.

In this demonstration, I created an application ID, named “mkyong123”, and put it in appengine- web.xml.

File : appengine-web.xml

```
<?xml version="1.0" encoding="utf-8"?>
<appengine-web-app xmlns="http://appengine.google.com/ns/1.0">
  <application>mkyong123</application>
  <version>1</version>

  <!-- Configure java.util.logging -->
  <system-properties>
    <property name="java.util.logging.config.file" value="WEB-INF/logging.properties"/>
  </system-properties>

</appengine-web-app> Copy
```


To deploy, see following steps:

Figure 1.1 – Click on GAE deploy button on the toolbar.

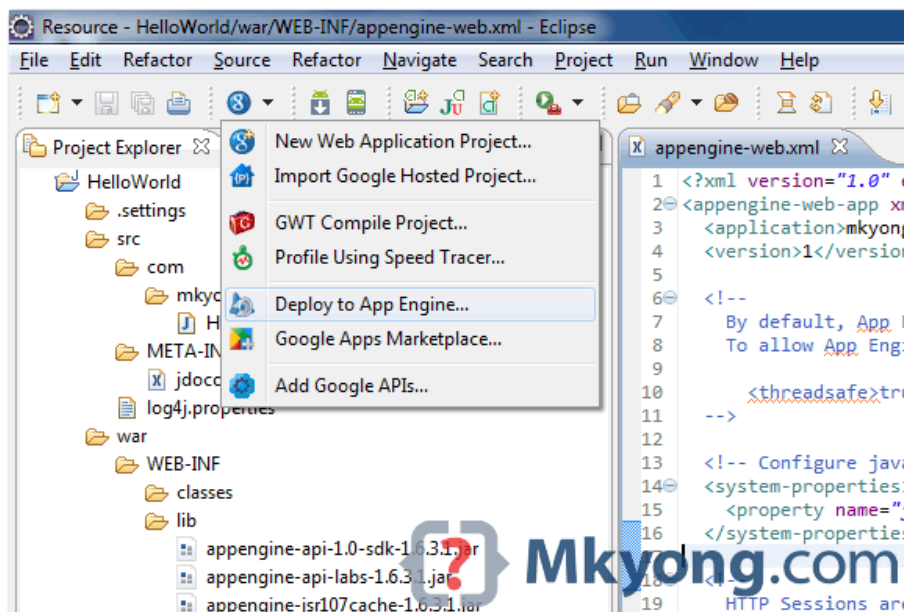


Figure 1.2 – Sign in with your Google account and click on the Deploy button.

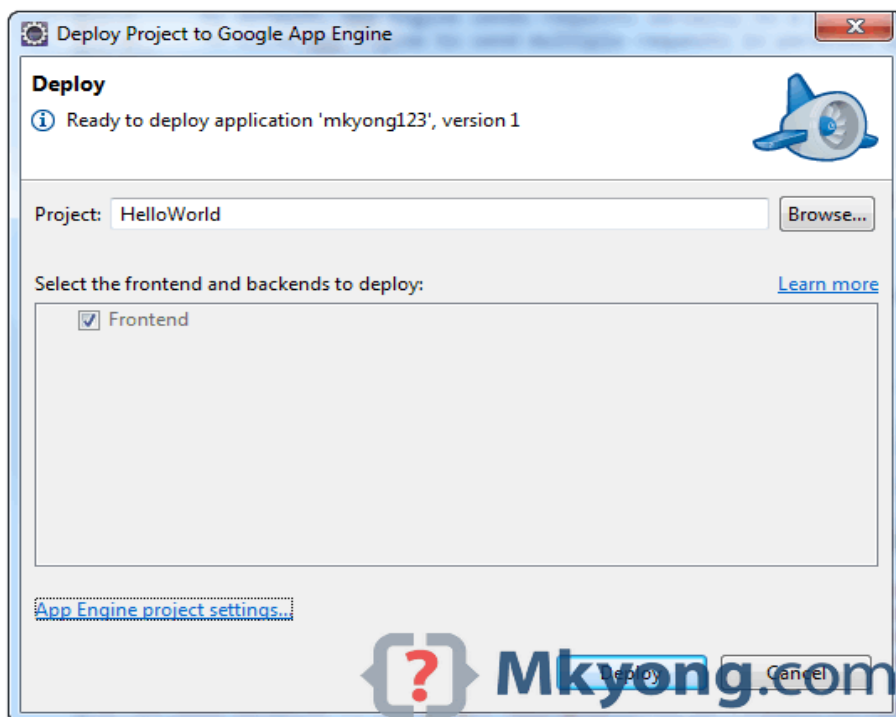


Figure 1.3 – If everything is fine, the hello world web application will be deployed to this URL – <http://mkyong123.appspot.com/>



Result:

Thus the simple application was created successfully.

Ex. No : 4

Date :

Use GAE launcher to launch the web application

AIM:

To Use GAE launcher to launch the web applications.

Steps:

Making your First Application

Now you need to create a simple application. We could use the “+” option to have the launcher make us an application – but instead we will do it by hand to get a better sense of what is going on.

Make a folder for your Google App Engine applications. I am going to make the Folder on my Desktop called “apps” – the path to this folder is:

C:\Documents and Settings\csev\Desktop\apps

And then make a sub-folder in within apps called “ae-01-trivial” – the path to this folder would be:

C:\Documents and Settings\csev\Desktop\apps\ae-01-trivial

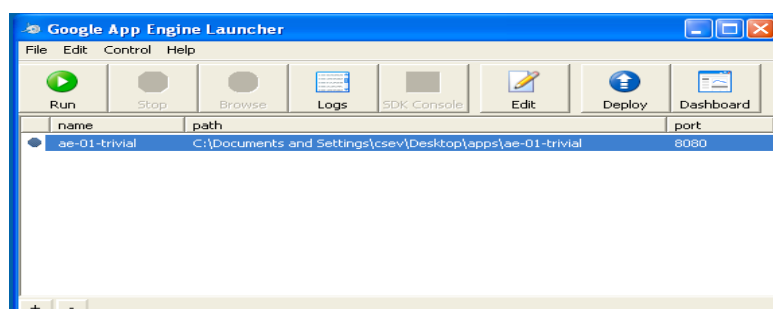
Using a text editor such as JEdit (www.jedit.org), create a file called app.yaml in the ae-01-trivial folder with the following contents:

```
application: ae-01-trivial
version: 1
runtime: python api_
version: 1 handlers:- url: /*
script: index.py
```

```
print 'Content-Type: text/plain' print ' '
```

```
print 'Hello there Chuck'
```

Then start the Google App Engine Launcher program that can be found under Applications. Use the File -> Add Existing Application command and navigate into the apps directory and select the ae-01-trivial folder. Once you have added the application, select it so that you can control the application using the launcher.



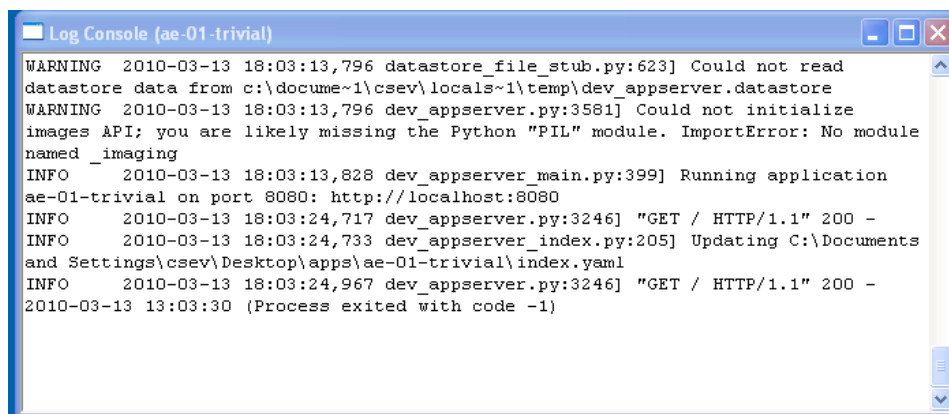
Once you have selected your application and press Run. After a few moments your application will start and the launcher will show a little green icon next to your application. Then press Browse to open a browser pointing at your application which is running at *http://localhost:8080/*

Paste *http://localhost:8080* into your browser and you should see your application as follows:



Watching the Log

You can watch the internal log of the actions that the webserver is performing when you are interacting with your application in the browser. Select your application in the Launcher and press the Logs button to bring up a log window:



Dealing With Errors

With two files to edit, there are two general categories of errors that you may encounter. If you make a mistake on the app.yaml file, the App Engine will not start and your launcher will show a yellow icon near your application:

To get more detail on what is going wrong, take a look at the log for the application:

Ex. No : 5

Date :

Simulate a cloud scenario using CloudSim and run a scheduling algorithm that is not present in CloudSim.

AIM:

To Simulate a cloud scenario using CloudSim and run a scheduling algorithm that is not present in CloudSim.

STEPS:

How to use CloudSim in Eclipse

CloudSim is written in Java. The knowledge you need to use CloudSim is basic Java programming and some basics about cloud computing. Knowledge of programming IDEs such as Eclipse or NetBeans is also helpful. It is a library and, hence, CloudSim does not have to be installed. Normally, you can unpack the downloaded package in any directory, add it to the Java classpath and it is ready to be used. Please verify whether Java is available on your system.

To use CloudSim in Eclipse:

1. Download CloudSim installable files from the link and unzip
<https://code.google.com/p/cloudsim/downloads/list>
2. Open Eclipse
3. Create a new Java Project: File -> New
4. Import an unpacked CloudSim project into the new Java Project

The first step is to initialise the CloudSim package by initialising the CloudSim library, as follows

CloudSim.init(num_user, calendar, trace_flag)

5. Data centres are the resource providers in CloudSim; hence, creation of data centres is a second step. To create Datacenter, you need the DatacenterCharacteristics object that stores the properties of a data centre such as architecture, OS, list of machines, allocation policy that covers the time or spaceshared, the time zone and its price:

Datacenter datacenter9883 = new Datacenter(name, characteristics, new VmAllocationPolicySimple(hostList), s

6. The third step is to create a broker:

DatacenterBroker broker = createBroker();

7. The fourth step is to create one virtual machine unique ID of the VM, userId ID of the VM's owner, mips, number Of Pes amount of CPUs, amount of RAM,

amount of bandwidth, amount of storage, virtual machine monitor, and cloudletScheduler policy for cloudlets:

```
Vm vm = new Vm(vmid, brokerId, mips, pesNumber, ram, bw, size, vmm, new  
CloudletSchedulerTimeShared())
```

8. Submit the VM list to the broker: *broker.submitVmList(vmlist)*

9. Create a cloudlet with length, file size, output size, and utilisation model:

```
Cloudlet cloudlet = new Cloudlet(id, length, pesNumber, fileSize, outputSize,  
utilizationModel, utilizationMode)
```

10. Submit the cloudlet list to the broker:

```
broker.submitCloudletList(cloudletList)
```

OUTPUT:

Cloudlet ID STATUS Data center ID Finish Time 0 SUCCESS 2

0.1 400.1 VM ID Time 0 Start Time 400

*****Datacenter: Datacenter_0*****

User id 3

Debt 35.6

CloudSimExample1 finished!

RESULT:

The simulation was successfully executed.

Ex. No : 6

Date :

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

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).

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).

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.

PROCEDURE:

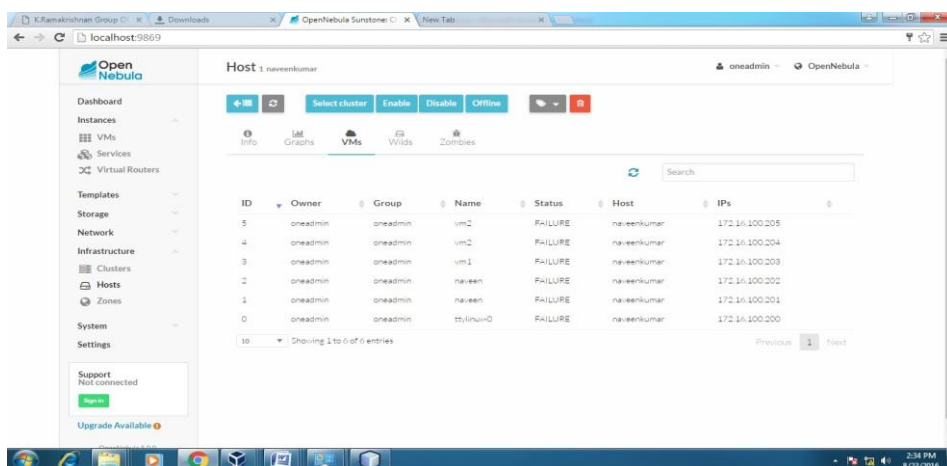
Steps:

1. Open Browser, type localhost:9869

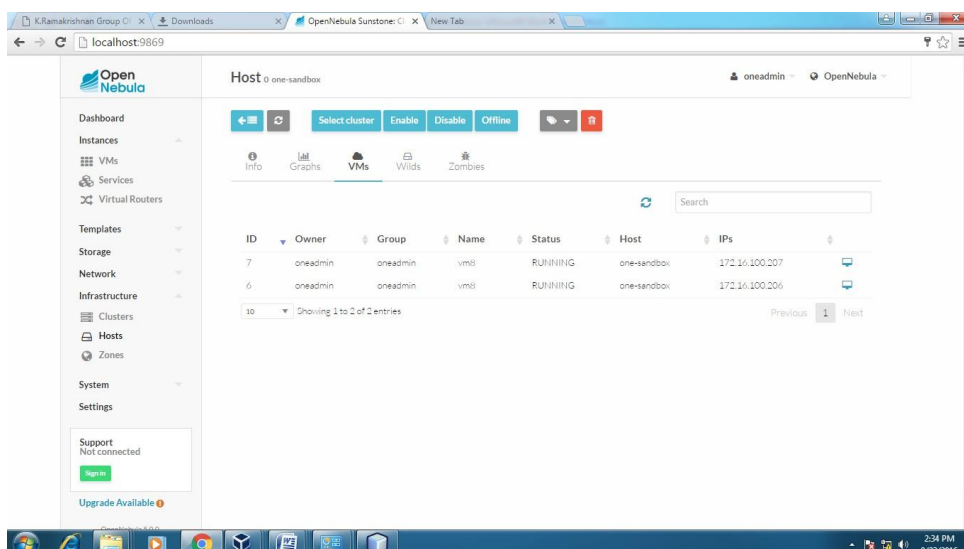
2. Login using username: oneadmin, password: opennebula
3. Then follow the steps to migrate VMs
 - a. Click on infrastructure
 - b. Select clusters and enter the cluster name
 - c. Then select host tab, and select all host
 - d. Then select Vnets tab, and select all vnet
 - e. Then select datastores 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.

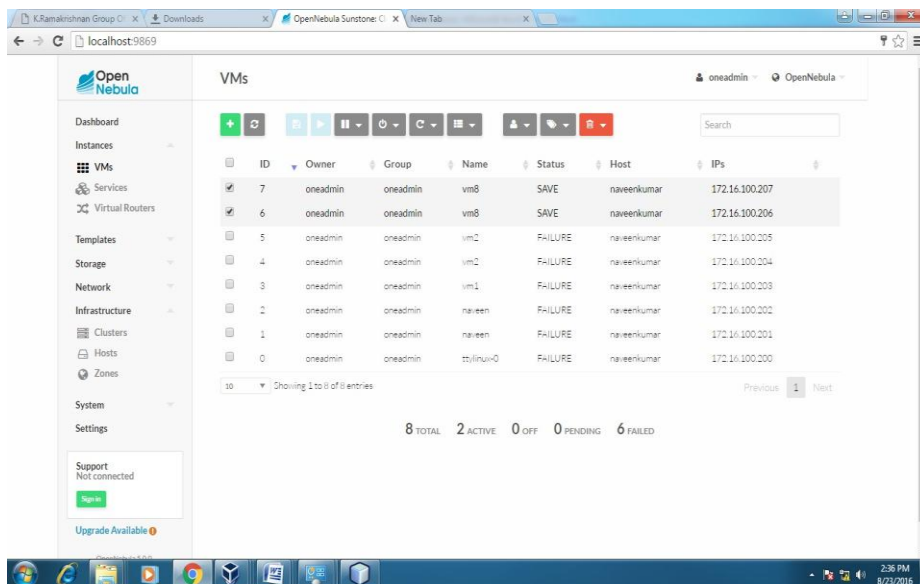
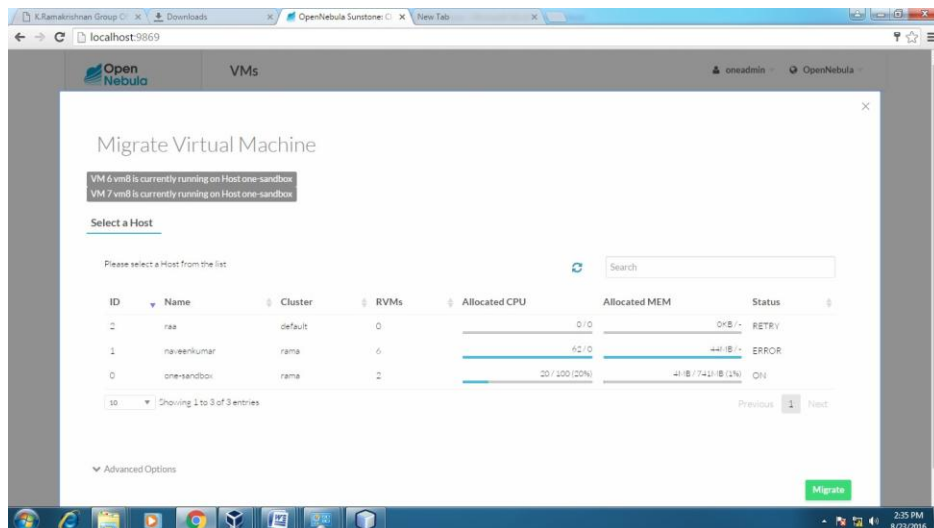
Before migration

Host: SACET

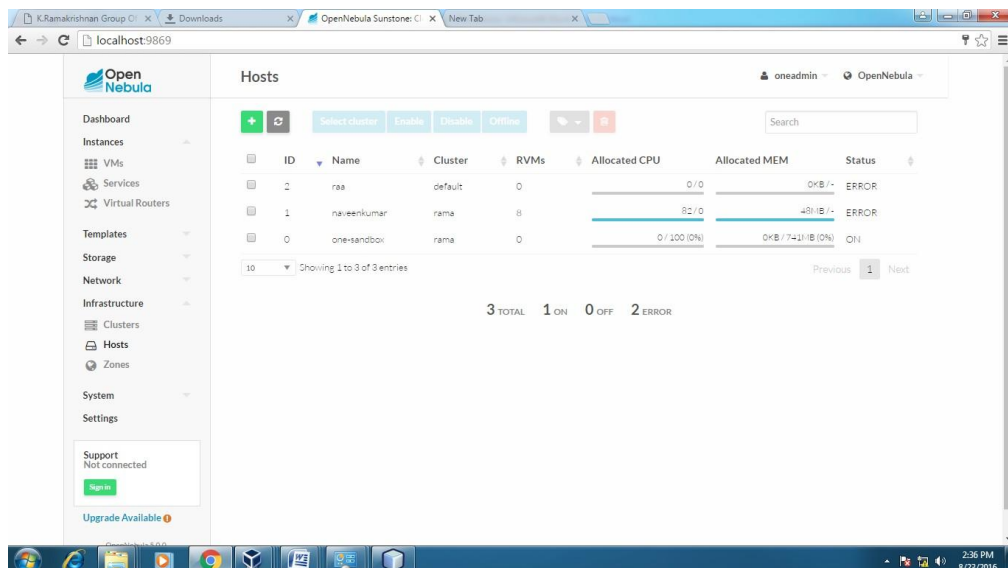


Host: one-sandbox

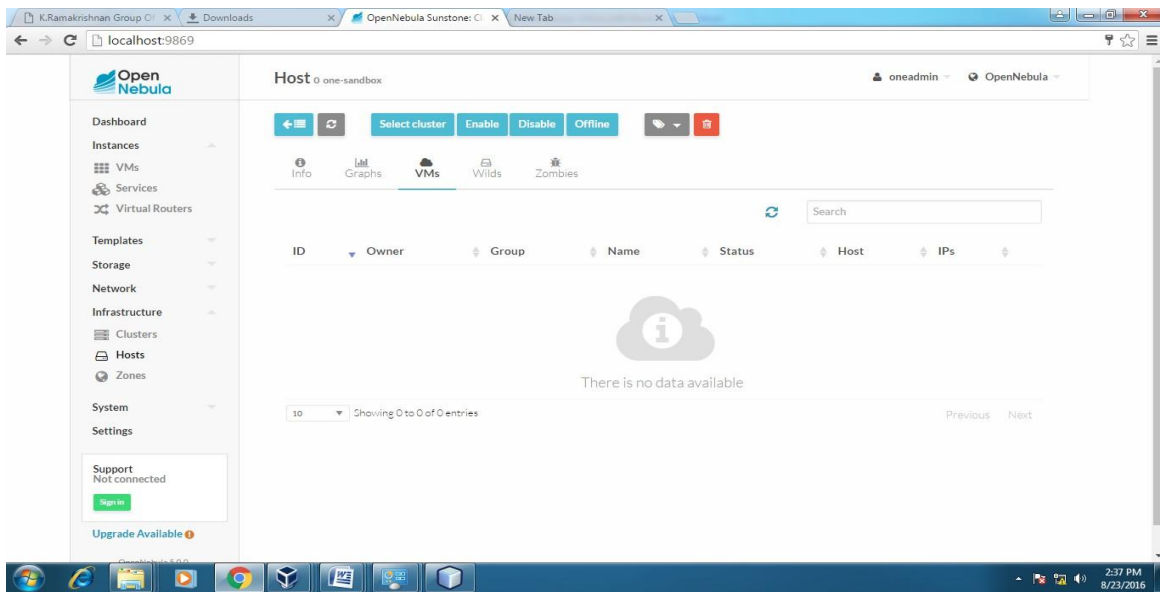




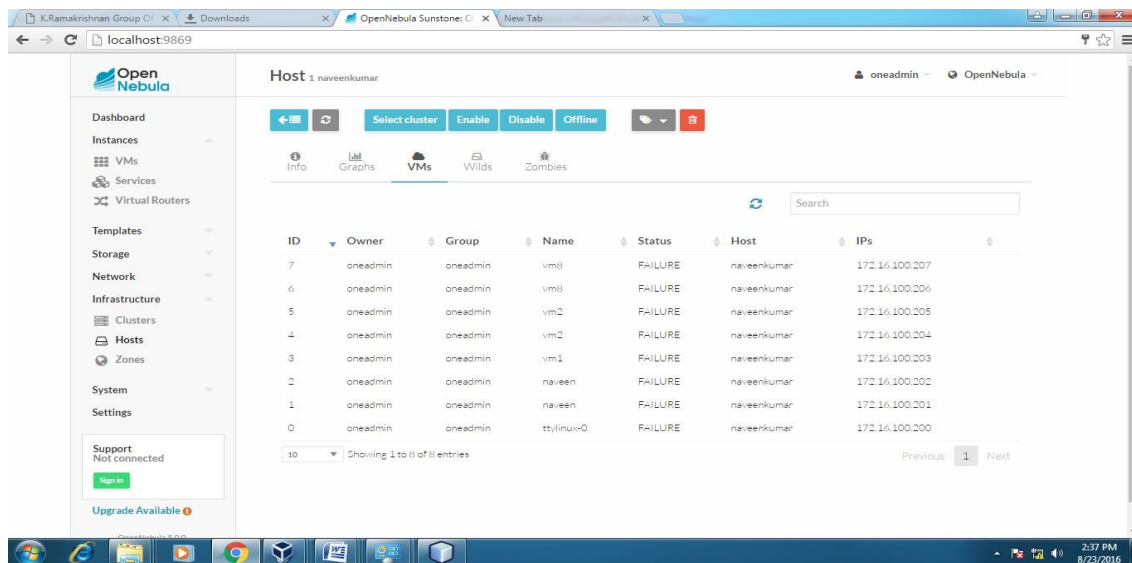
After Migration:



Host: one-sandbox



Host:SACET



APPLICATIONS:

Easily migrate your virtual machine from one pc to another.

RESULT:

Thus the file transfer between VM was successfully completed.....

Ex. No : 7
Date :

Find a procedure to launch virtual machine using trystack (Online Openstack Demo Version)

AIM:

To find a procedure to launch virtual machine using trystack.

STEPS:

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.



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.

1. Go to Network > Networks and then click Create Network.
2. In Network tab, fill Network Name for example internal and then click Next.
3. In Subnet tab,

1. Fill Network Address with appropriate CIDR, for example 192.168.1.0/24.
Use private network CIDR block as the best practice.
2. Select IP Version with appropriate IP version, in this case IPv4.
3. Click Next.
4. In Subnet Details tab, fill DNS Name Servers with 8.8.8.8 (Google DNS) and then click Create.

Step 2: Create Instance

1. Go to Compute > Instances and then click Launch Instance.
2. In Details tab,
 1. Fill Instance Name, for example Ubuntu 1.
 2. Select Flavor, for example m1.medium.
 3. Fill Instance Count with 1.
 4. Select Instance Boot Source with Boot from Image.
 5. 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,
 1. 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.
 2. In Import Key Pair dialog,
 - i. Fill Key Pair Name with your machine name (for example Edward-Key).
 - ii. 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.
 - iii. Click Import key pair.
 3. 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.

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.

1. Go to Network > Routers and then click Create Router.
2. Fill Router Name for example router1 and then click Create router.
3. Click on your router name link, for example router1, Router Details page.
4. Click Set Gateway button in upper right:
 1. Select External networks with external.
 2. Then OK.
5. Click Add Interface button.
 1. Select Subnet with the network that you have been created in Step 1.
 2. Click Add interface.
6. 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 IP's 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.

1. Go to Compute > Instance.
2. In one of your instances, click More > Associate Floating IP.
3. In IP Address, click Plus [+].
4. Select Pool to external and then click Allocate IP.
5. Click Associate.
6. 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.

1. Go to Compute > Access & Security and then open Security Groups tab.
2. In default row, click Manage Rules.
3. Click Add Rule, choose ALL ICMP rule to enable ping into your instance, and then click Add.

4. Click Add Rule, choose HTTP rule to open HTTP port (port 80), and then click Add.
5. Click Add Rule, choose SSH rule to open SSH port (port 22), and then click Add.
6. 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.

RESULT:

Thus the OpenStack demo worked successfully.

Ex. No : 8

Date :

Install Hadoop single node cluster and run simple applications like wordcount.

AIM:

To install hadoop single node cluster and run simple Applications like wordcount.

STEPS:

Install hadoop

Step 1: click here to 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`

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`

Step 4: Extract the Hadoop tar File.

Command: `tar -xvf hadoop-2.7.3.tar.gz`

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.

Command: `vi.bashrc`

Then, save the bash file and close it. For applying all these changes to the current Terminal, execute the source command.

Command: `source.bashrc`

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`

Command: `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:

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*

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*

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.

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

Command: *vi 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.

Command: *vi 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*

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

Command: *cd*

Command: *cd hadoop-2.7.3*

Command: *bin/hadoop namenode -format*

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.

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*

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*

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*

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*

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*

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*

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

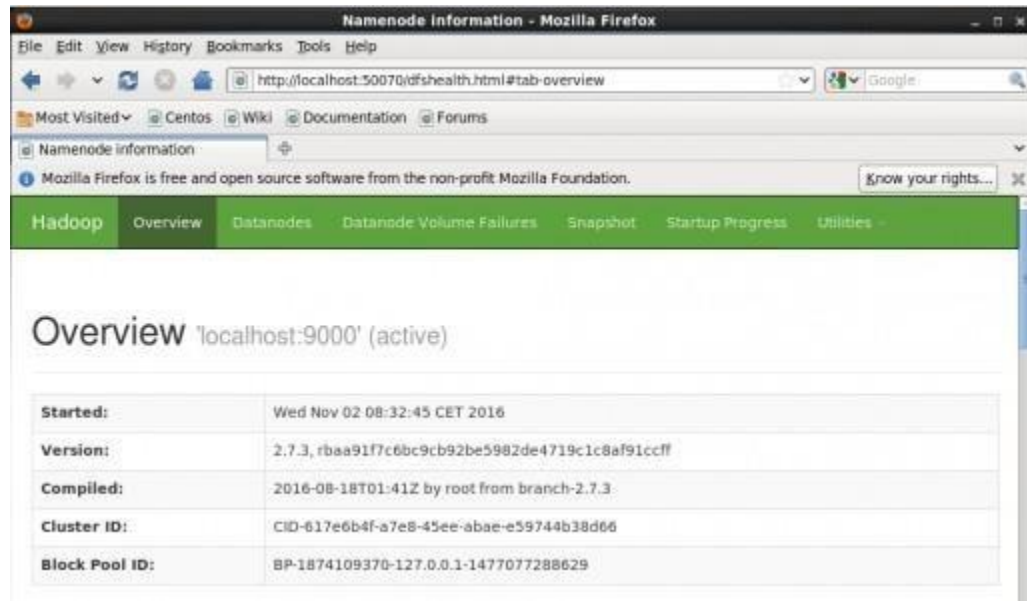


Fig: Hadoop Installation – Starting WebUI

Congratulations, you have successfully installed a single node Hadoop cluster

RESULT:

Thus the Hadoop one cluster was installed and simple applications executed successfully.