



NEHRU INSTITUTE OF ENGINEERING AND TECHNOLOGY
Autonomous



An ISO 9001: 2015 & 14001: 2015 Certified Institution, Affiliated to Anna University, Chennai
Approved by AICTE, New Delhi, Recognized by UGC with Section 2(f) and 12(B)
Re-accredited by NAAC “A+”, NBA Accredited (UG Courses: AERO & CSE)
“Nehru Garden”, Thirumalayampalayam, Coimbatore - 641 105.

DEPARTMENT OF INFORMATION TECHNOLOGY

LAB MANUAL

ANNA UNIVERSITY REGULATION 2021

Laboratory Code : CSS335

Laboratory Name : Cloud Computing Laboratory

Semester / Year : V / III

Degree / Branch : B.Tech / IT

VISION AND MISSION OF THE DEPARTMENT

VISION

To produce highly competent and innovative Computing and Business Systems Professionals with Managerial Skills, Social Values to serve the Nation and to meet the industry challenges.

MISSION

M1: To impart technical knowledge through innovative students-centric teaching learning processes and research.

M2: To groom students technologically superior and ethically stronger and responsible throughout the professional career to compete globally.

M3: To produce competent engineers with professional ethics, spirit of innovation and managerial skills to cater the needs of the industries and society.

PEOS & PSOS

PROGRAM EDUCATIONAL OBJECTIVES

The Graduates of the Information Technology Programme will be able to

PEOs	PROGRAM EDUCATIONAL OBJECTIVES
PEO1	Acquire and Apply knowledge in Computer Science, Mathematics, Science and inter-disciplinary engineering principles in order to excel in computer professional career.
PEO2	Analyze real life problems adapting to new Computing Technologies for professional excellence and ethical attitude in order to provide economically feasible engineering solutions.
PEO3	Carry out complex engineering problems with best practices exhibiting communication skills, team work and interpersonal skills to enable continued computer professional development through life-long learning.

PROGRAM SPECIFIC OUTCOMES

The Students of the Information Technology Programme will be able to

PSO	PROGRAM SPECIFIC OUTCOMES
PSO1	Professional Skills: Acquaint in-depth knowledge on the basic and advanced computer science domains like Data Sciences, Cryptography, Cloud and Distributed Computing, Neural Networks and Artificial Intelligence.
PSO2	Analyze and recommend the appropriate IT infrastructure required for the implementation of a project.
PSO3	Entrepreneurship and Successful Career: Apply the standard practices to have successful career path in the field of information and communication technology and entrepreneurship.

PROGRAM OUTCOMES

PO	PROGRAM OUTCOMES (POS)-12 GRADUATE ATTRIBUTES
PO1	Engineering knowledge: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
PO2	Problem analysis: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
PO3	Design/development of solutions: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
PO4	Conduct investigations of complex problems: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
PO5	Modern tool usage: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
PO6	The engineer and society: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
PO7	Environment and sustainability: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
PO8	Ethics: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
PO9	Individual and team work: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
PO10	Communication: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
PO11	Project management and finance: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
PO12	Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

Please DO!



- Be on time, Clean up yourself; enter the lab with proper dress code and lab coat
- Sign the login register and occupy only the allotted system
- Do use the computer equipments properly and inform the instructor, if there is a problem
- Touch the keyboard lightly and maintain silence
- Do some example exercises beyond the syllabus
- Read and understand how to carry out an experiment thoroughly & complete the lab workbook before coming to the laboratory
- Do a proper shutdown after using computers and arrange the chairs properly before leaving the lab
- Respect the equipment. Don't damage, remove, or disconnect any labels, parts, cables, or equipment.

Please DON'T!



- Don't bring food or drinks inside the room
- Do not remove anything from the computer laboratory without permission
- Don't play games on the computers
- Don't make undue noise in the laboratories
- Don't disturb other users
- Don't use the on/off switch to reboot
- Do not read or modify other users' files.
- Do not install or download any software or modify or delete any system files on any lab computers.
- Leave the bags outside and do not use pen drives
- Do not touch, connect or disconnect any plug or cable without your instructor/laboratory technician's permission
- Do not misbehave in the computer laboratory
- If you leave the lab, do not leave your personal belongings unattended.

Syllabus		
V Semester		
Course Code: CSS335		Course Name: Cloud Computing
Objective(s)	Students will be made to 1. To understand the principles of cloud architecture, models and infrastructure. 2. To understand the concepts of virtualization and virtual machines. 3. To gain knowledge about virtualization Infrastructure. 4. To explore and experiment with various Cloud deployment environments. 5. To learn about the security issues in the cloud environment	
Outcome(s)	At the end of this course, the students will be able to: 1. Explain the design challenges in the cloud. 2. Apply the concept of virtualization and its types 3. Experiment with virtualization of hardware resources and Docker. 4. Develop and deploy services on the cloud and set up a cloud environment. 5. Elaborate the security challenges in the cloud environment.	
LIST OF EXPERIMENTS		
1	Install Virtual box/VMware/ Equivalent open source cloud Workstation with different flavours of Linux or Windows OS on top of windows 8 and above.	
2	Install a C compiler in the virtual machine created using a virtual box and execute Simple Programs	
3	Install Google App Engine. Create a hello world app and other simple web applications using python/java	
4	Use the 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	Install Hadoop single node cluster and run simple applications like word count.	
8	Creating and Executing Your First Container Using Docker.	
9	Run a Container from Docker Hub	
Total hours		30
CONTENT BEYOND SYLLABUS:		
1	Find a procedure to launch virtual machine using trystack (Online Openstack Demo Version)	

CO's-PO's & PSO's MAPPING

CO's	PO's												PSO's		
	1	2	3	4	5	6	7	8	9	10	11	12	1	2	3
1	3	2	1	1	1	-	-	-	2	3	1	3	2	1	3
2	3	1	2	2	1	-	-	-	1	2	1	3	2	2	1
3	2	3	2	3	1	-	-	-	3	1	1	3	1	1	1
4	1	2	3	3	3	-	-	-	3	3	1	2	1	3	3
5	2	3	3	1	3	-	-	-	2	2	1	2	2	2	3
AVg.	2.2	2.2	2.2	2	1.8	-	-	-	2.2	2.2	1	2.6	1.6	1.8	2.2

1-low, 2-medium, 3-high, '-'-no correlation

EXNO.:1	Install Virtual box / VMware Workstation with different flavours of Linux or windows OS on top of windows7 or 8.
DATE:	

Aim:

To Install Virtual box / VMware Workstation with different flavours of Linux or windows OS on top of Windows 7 or 8.

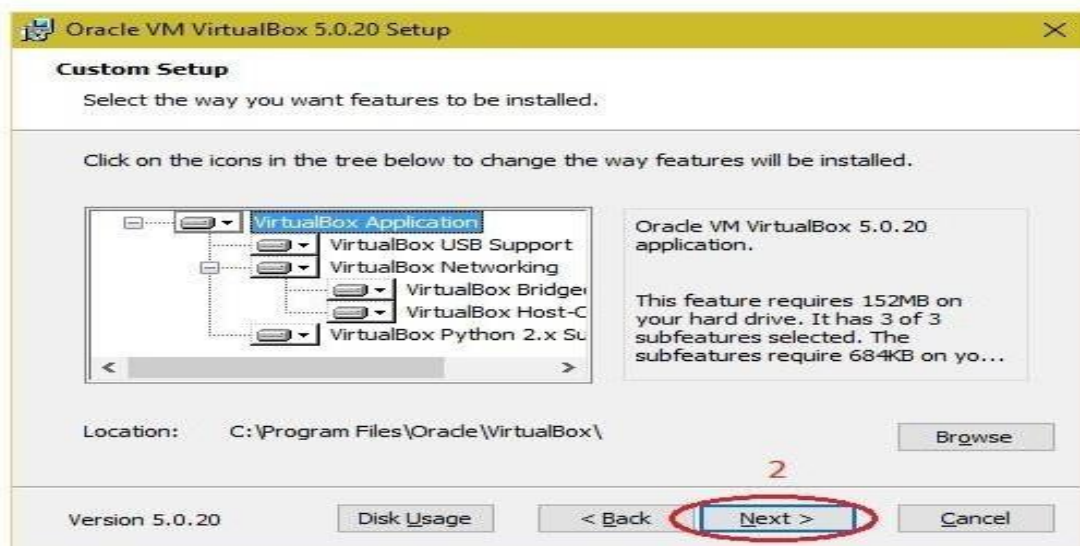
PROCEDURE:

Steps to install Virtual Box:

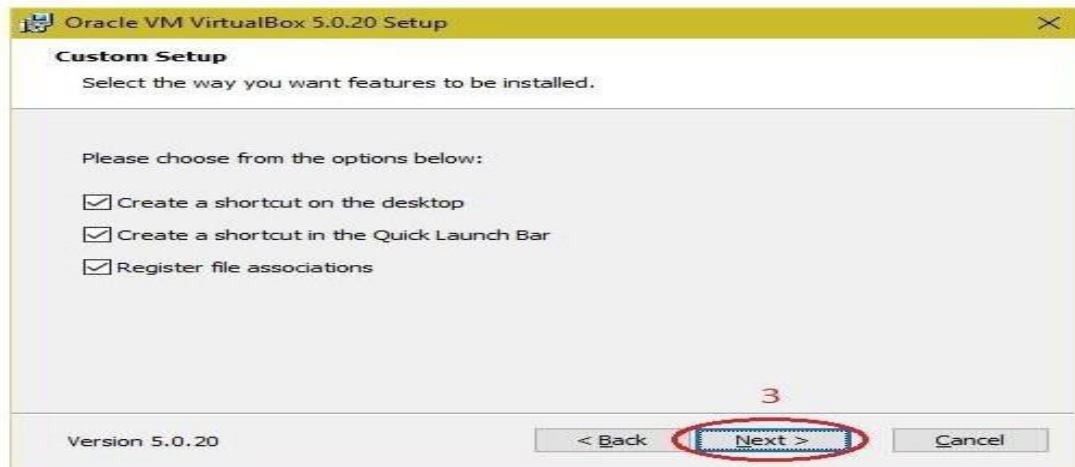
1. Download the Virtual box exe and click the exe file...and select next button.



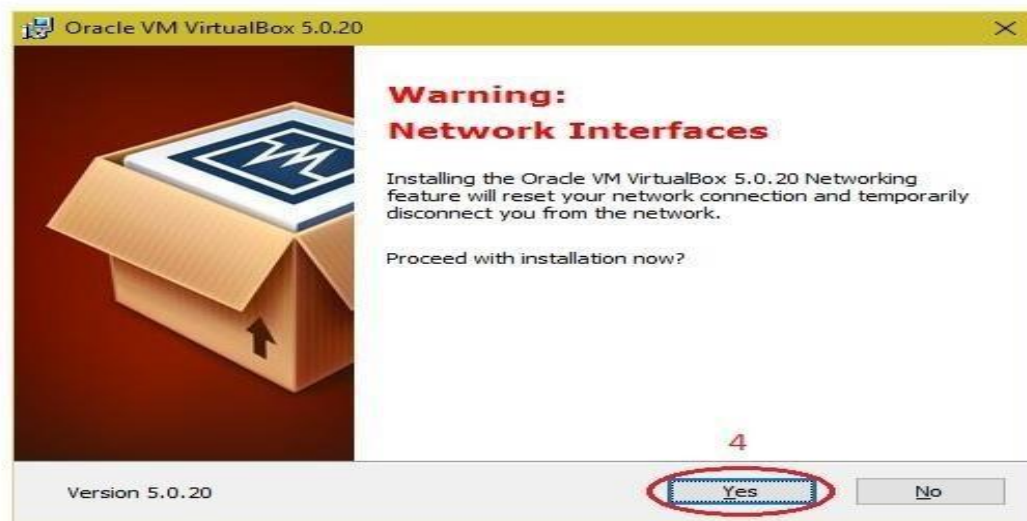
2. Click the next button.



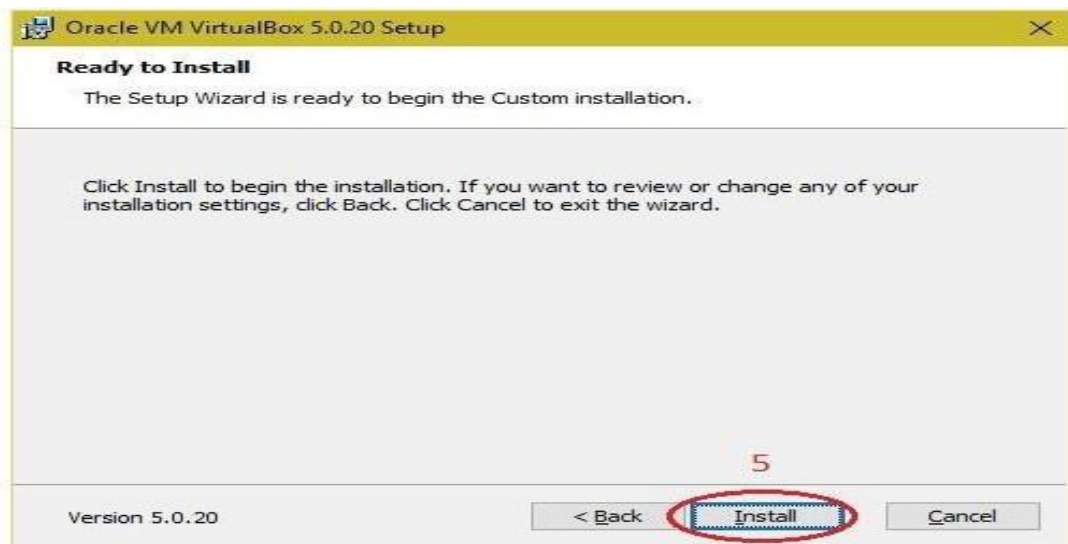
3. Click the next button



4. Click the YES button.



5. Click the install button

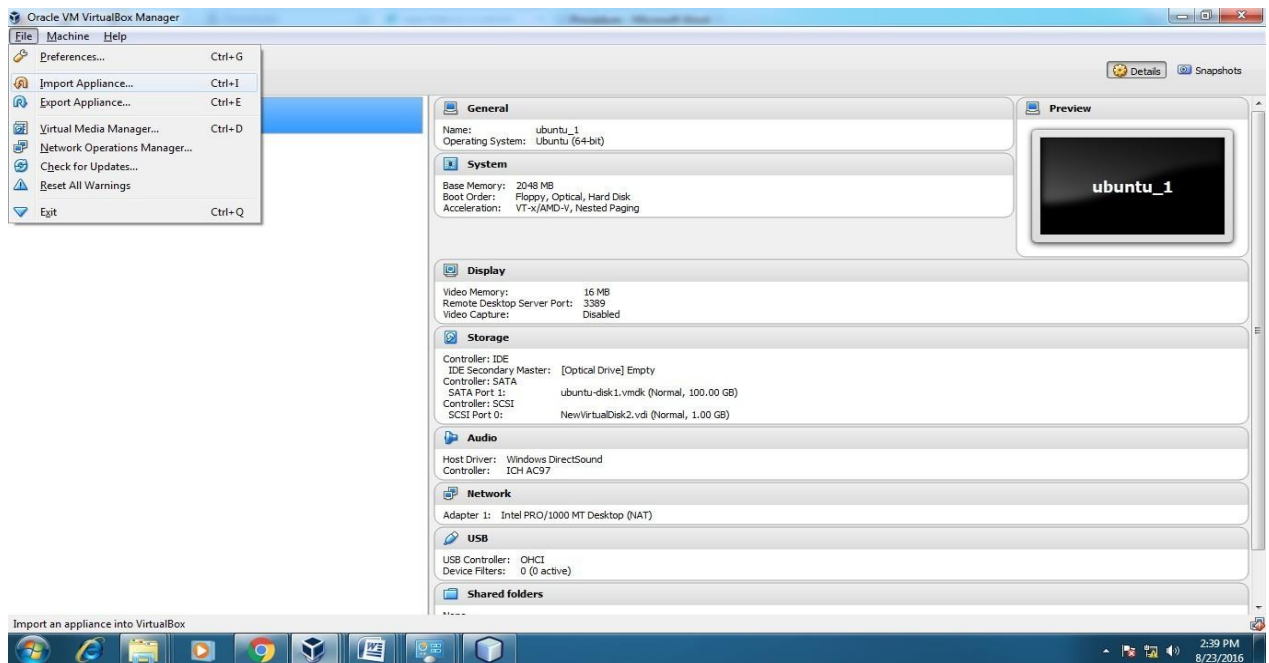


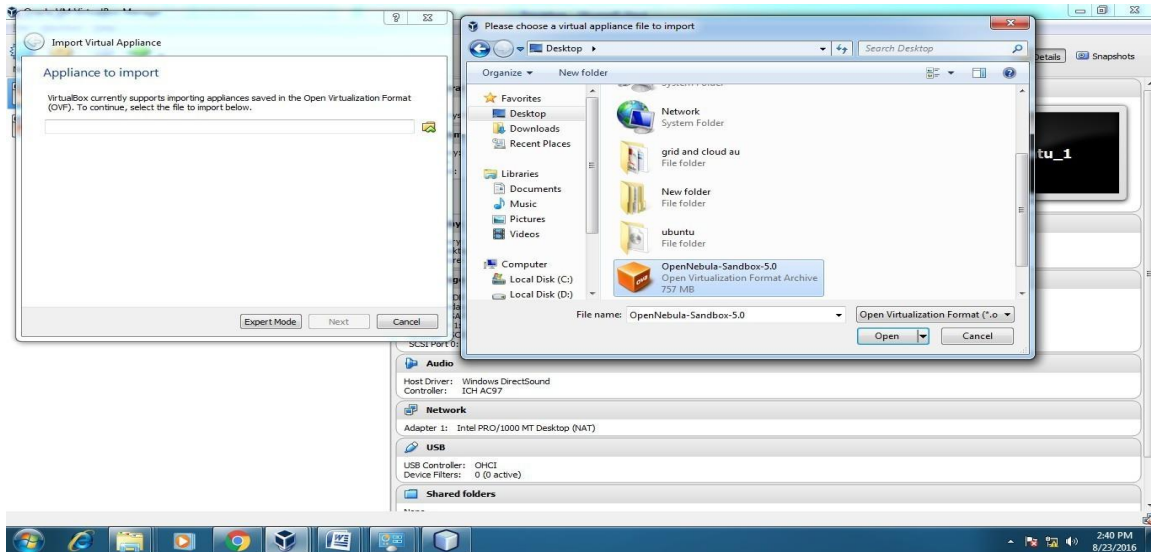
6. Then installation was completed. the show virtual box icon on desktop screen....



Steps to import Open nebula sandbox:

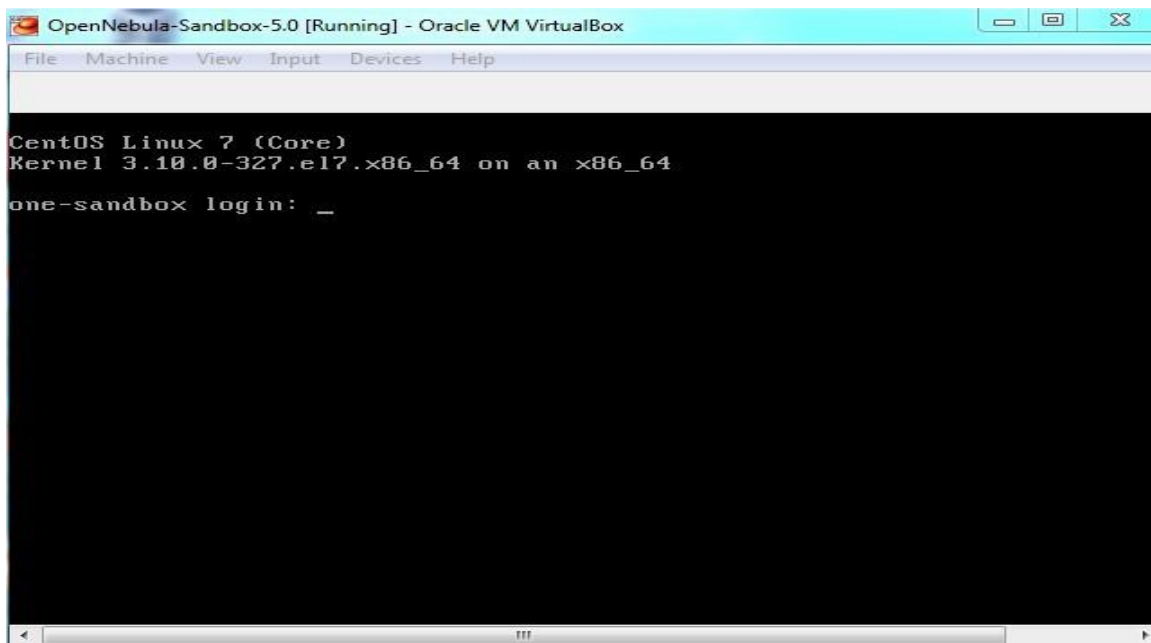
- Open Virtual box
- File ☰ import Appliance
- BrowseOpenNebula-Sandbox-5.0.ova file
- Thengotosetting,selectUsbandchooseUSB1.1
- Then Start the Open Nebula
- Login using username: root, password :open nebula

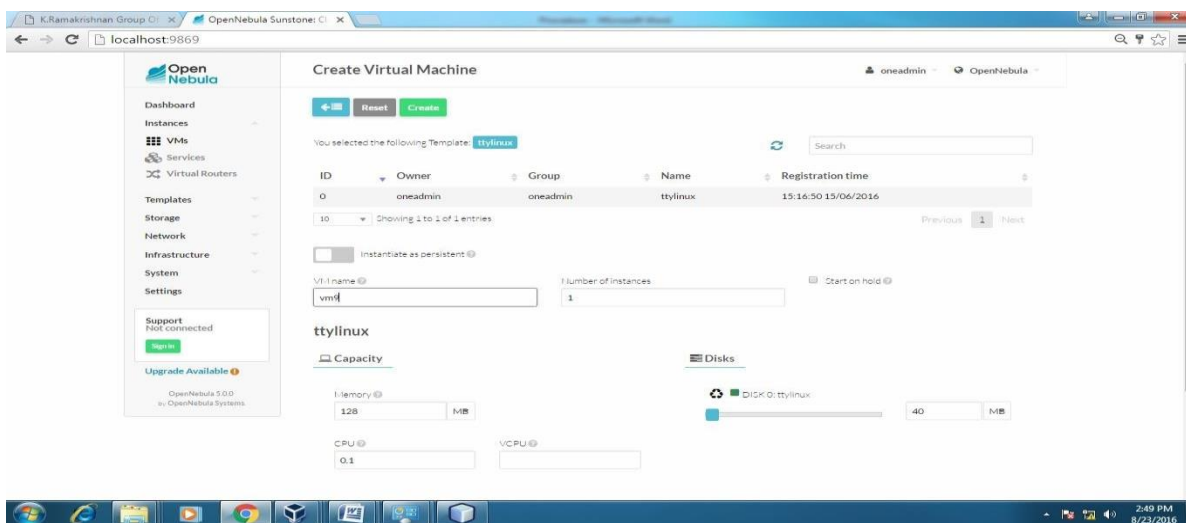




Steps to create Virtual Machine through open nebula

1. Open Browser, type localhost:9869
2. Login using username: one admin ,password: open nebula
3. Click on instances, select VMs then follow the steps to create Virtual machine
 - a. Expand the + symbol
 - b. Select user one admin
 - c. Then enter the VM name, no. of instance, cpu.
 - d. Then click on create button.
 - e. Repeat the steps the C,D for creating more than one VMs.





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

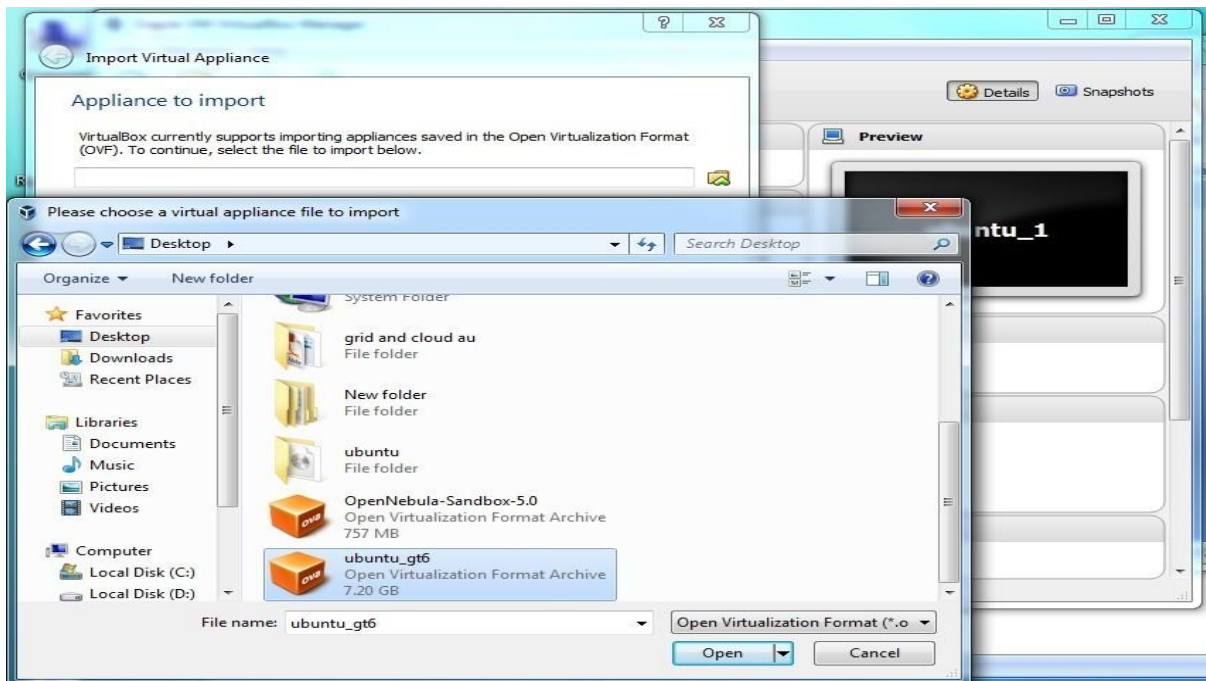
Aim:

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

PROCEDURE:

Steps to import.ova file:

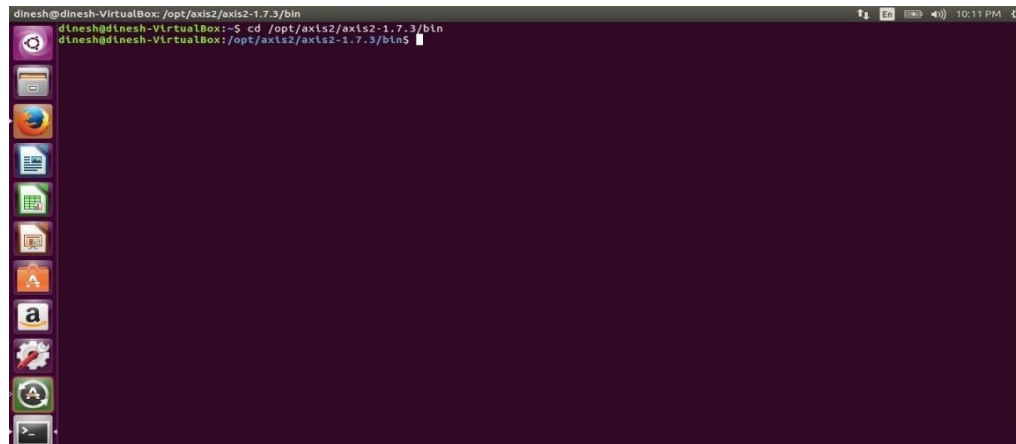
1. Open Virtual box
2. File ☰ import Appliance
3. Browseubuntu_gt6.ova file
4. Then go to setting, select Usb and choose USB1.1
5. Then Start the ubuntu_gt6
6. Login using username: dinesh, password: 99425.



Steps to run c program:

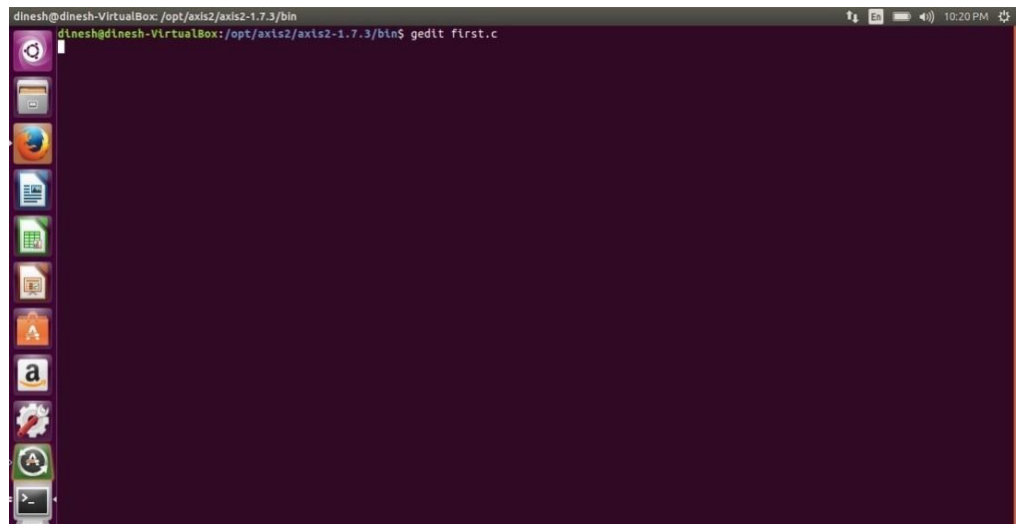
1. Open the terminal
2. Type `cd /opt/axis2/axis2-1.7.3/bin` then press enter
3. `gedit hello.c`
4. `gcc hello.c`
5. `./a.out`

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



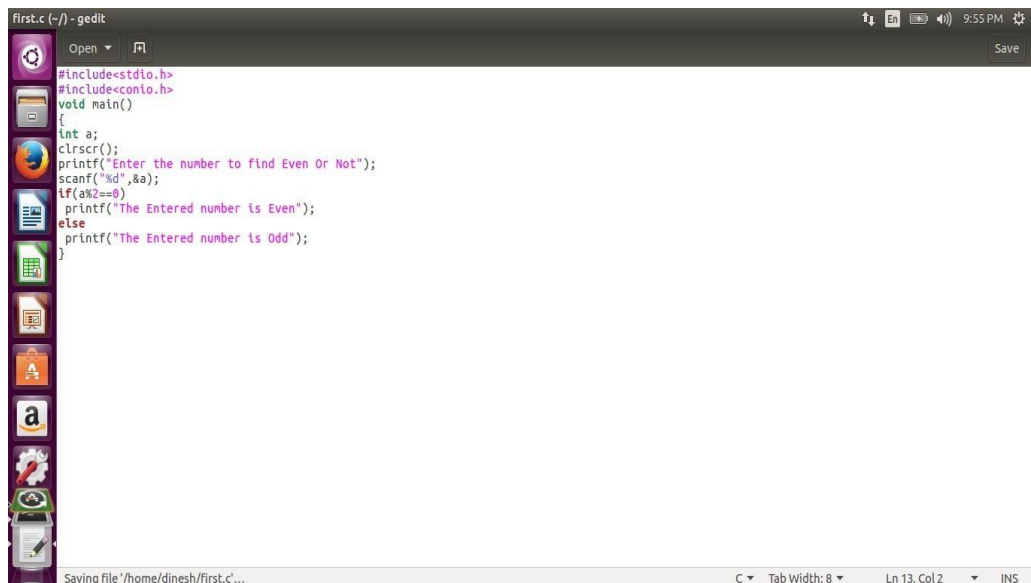
```
dinesh@dinesh-VirtualBox: /opt/axis2/axis2-1.7.3/bin
dinesh@dinesh-VirtualBox:~$ cd /opt/axis2/axis2-1.7.3/bin
dinesh@dinesh-VirtualBox: /opt/axis2/axis2-1.7.3/bin$
```

2. Type `gedit first.c`



```
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

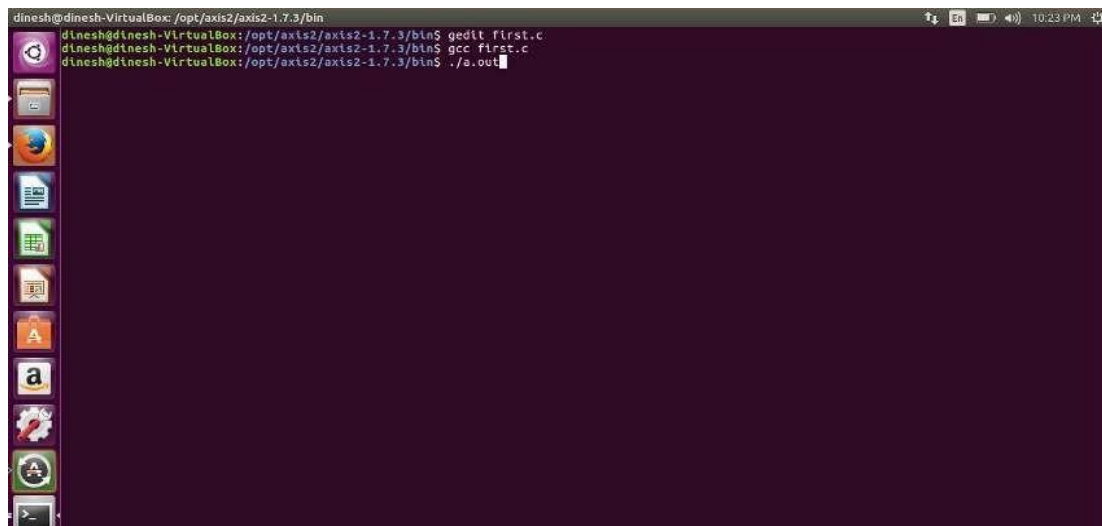


```
first.c (~) - gedit
#include<stdio.h>
#include<conio.h>
void main()
{
    int a;
    clrscr();
    printf("Enter the number to find Even Or Not");
    scanf("%d",&a);
    if(a%2==0)
        printf("The Entered number is Even");
    else
        printf("The Entered number is Odd");
}
```

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

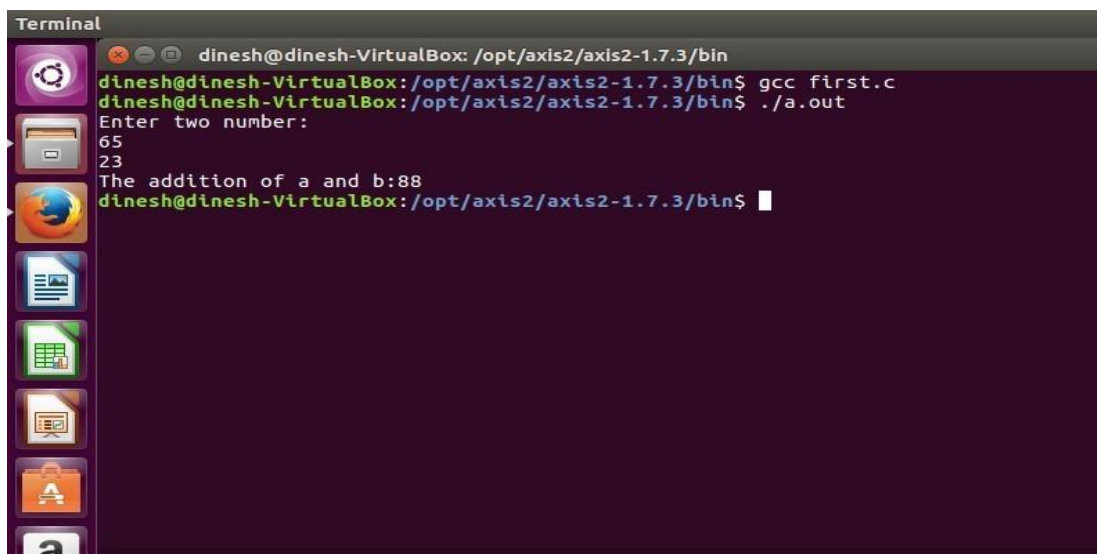
C Tab Width: 8 Ln 13, Col 2 INS

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$ gcc 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:



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

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

Read this guide – how to install Google Plugin for Eclipse. If you install the Google App Engine Java SDK together with “Google Plugin for Eclipse”, then go to step2, Otherwise, get the Google App Engine Java SDK and extract it.

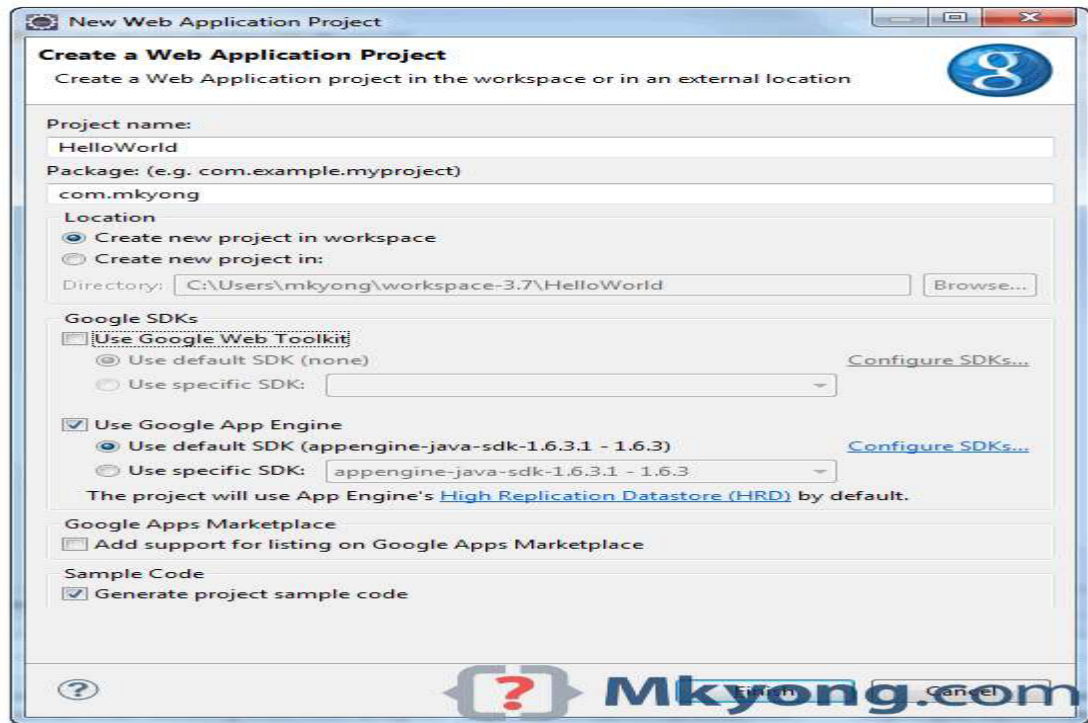
2. Create New Web Application Project

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

Figure–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.



Nothing special, a standard Java web project structure

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:

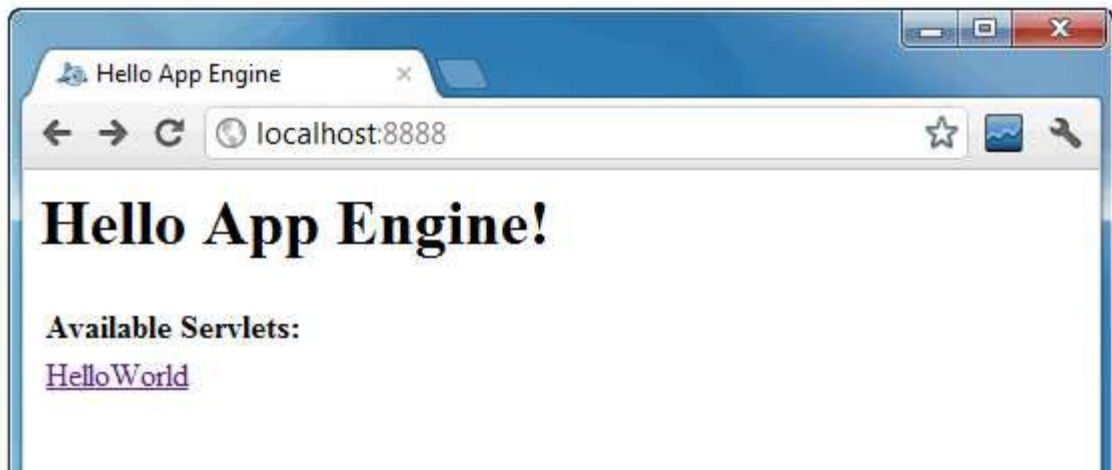
//...

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

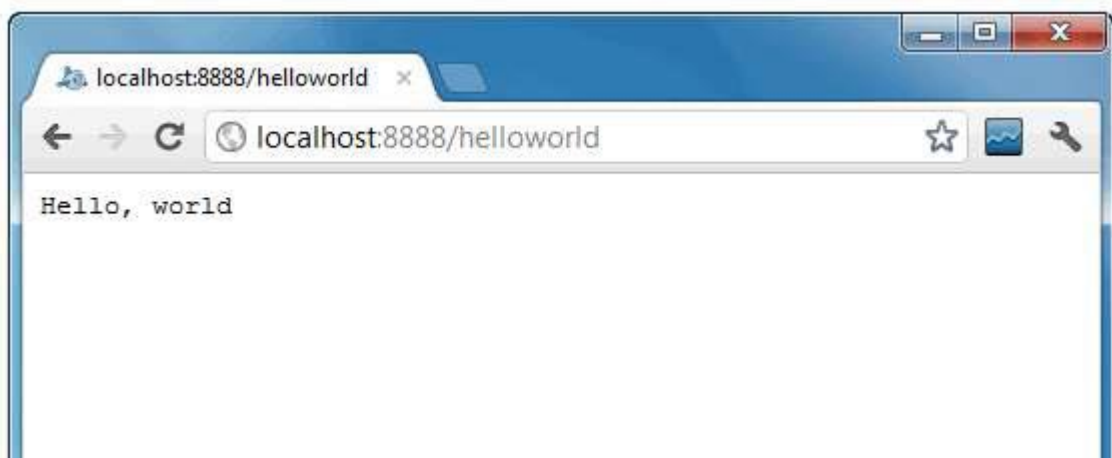
30 Mac 2012 11:13:01 PM com.google.appengine.tools.development.DevAppServerImpl start INFO: The admin console is running at http://localhost:8888/_ah/admin

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.

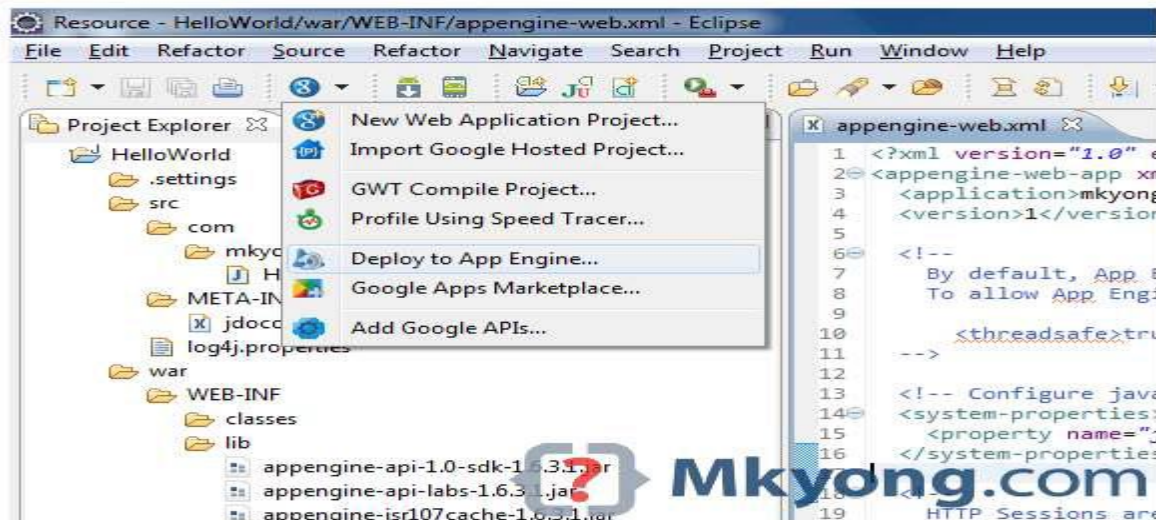


Figure 1.2 – Sign in with your Google account and click on the Deploy button.



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	Use GAE launcher to launch the web applications.
DATE:	

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

Note: Please do not copy and paste these lines into your text editor– you might end up with strange characters – simply type them into your editor.

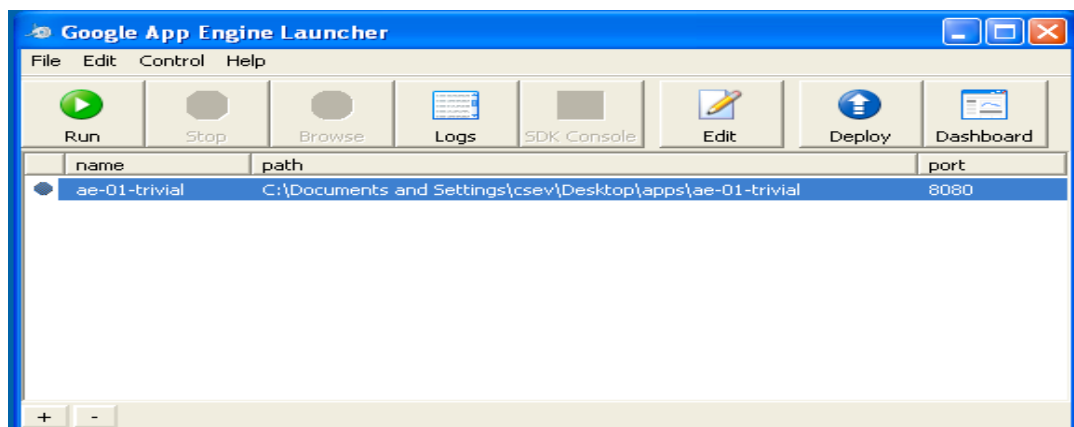
Then create a file in the ae-•01-•trivial folder called index.py with three lines in it:

```
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:

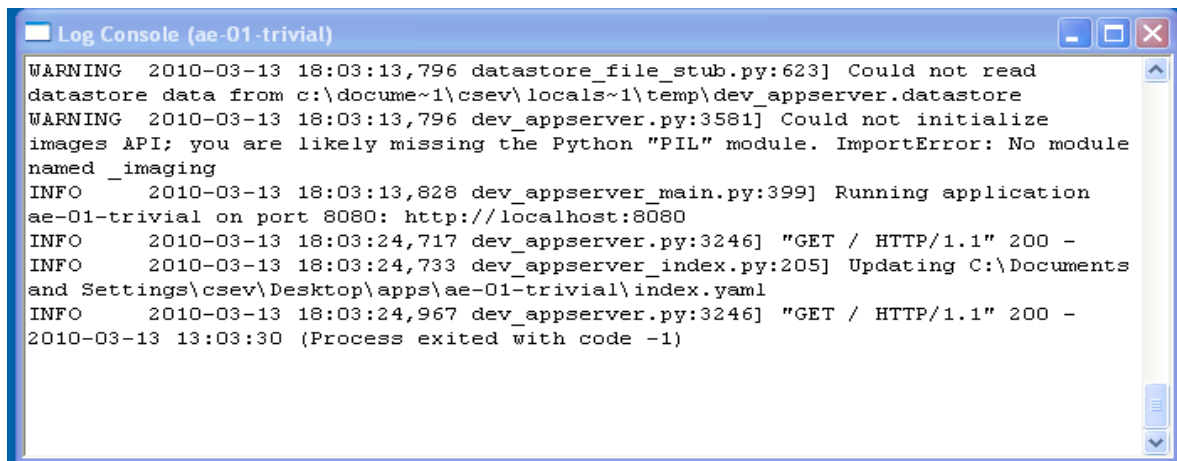


Just for fun, edit the `index.pyto` change the name “Chuck” to you row nname and press Refresh in the browser to verify your updates.

Watching the Log

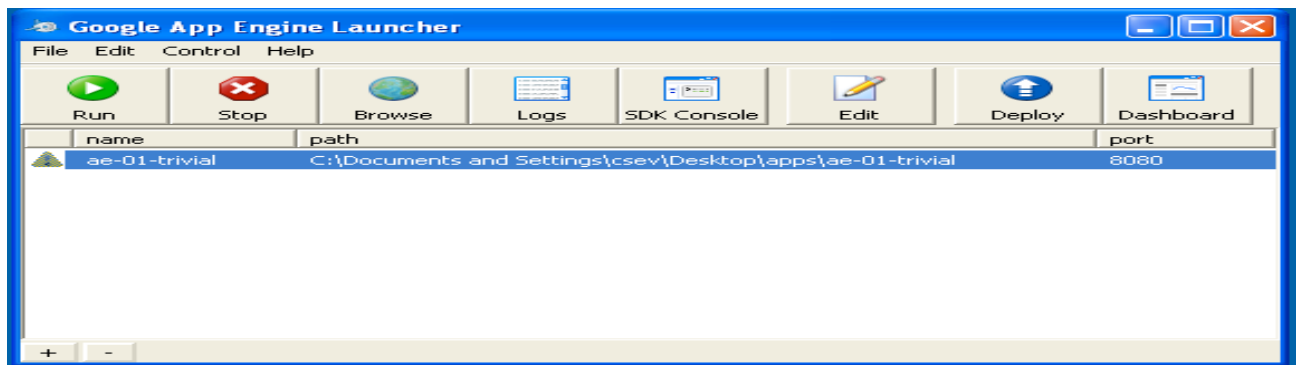
You can watch the internal log of the actions that the web server is performing when you are interacting with your application in the browser. Select your application in the Launcher and press the Logs button to bring up a log window:

Each time you press Refresh in your browser—you can see it retrieving the output with a GET request.

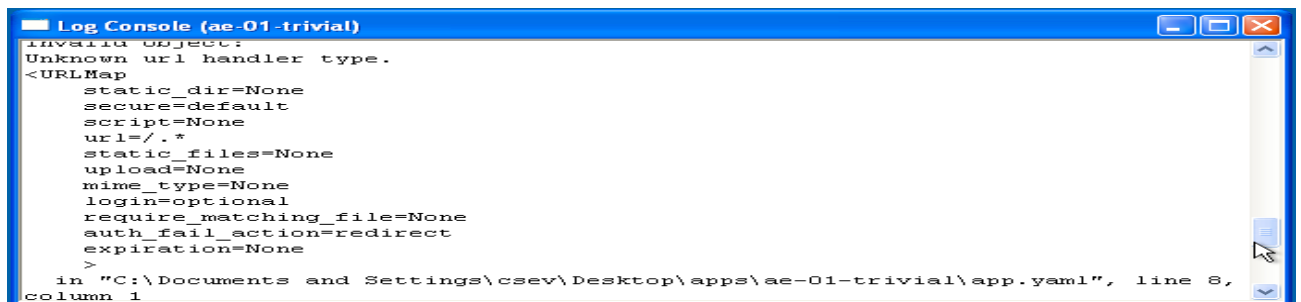


Dealing with Errors

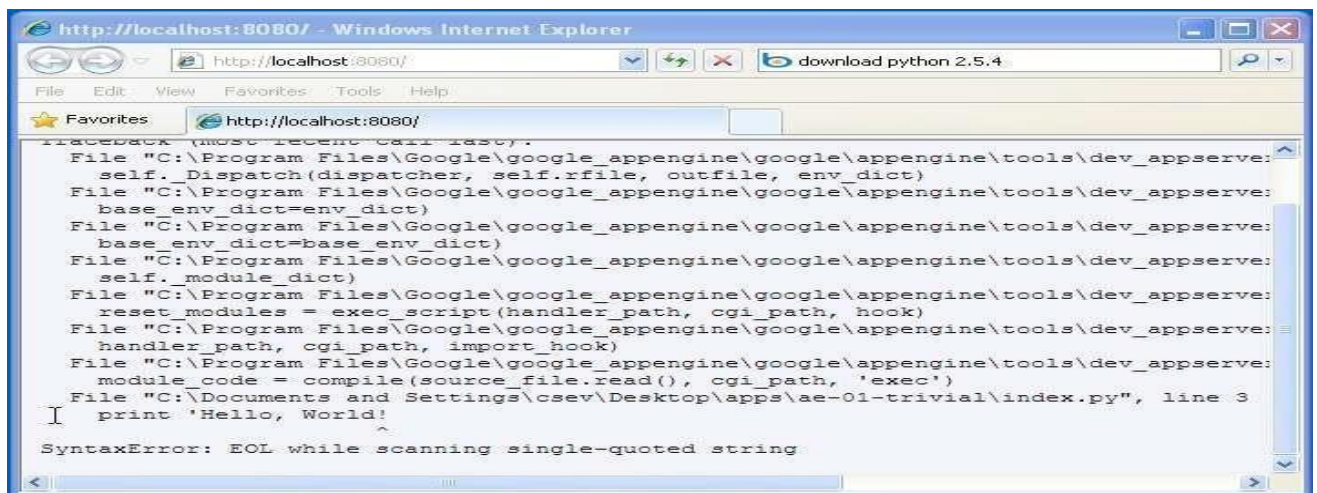
With two files to edit, there are two general categories of errors that you may encounter. If you make a mistake on the `app.yaml` file, the App Engine will not start and your launcher will show a yellow icon near your application:



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



In this instance – the mistake is mis--indenting the last line in the app.yaml (line 8). If you make a syntax error in the index.pyfile, a Python trace back error will appear in your browser.



syntax error in the index.pyfile, a Python trace backerrorwill appear in your browser.

The error you need to see is likely to be the last few lines of the output – in this case I made a Python syntax error on line one of our one--line application.

Reference: http://en.wikipedia.org/wiki/Stack_trace

When you make a mistake in the app.yaml file – you must the fix the mistake and attempt to start the application again.

If you make a mistake in a file like index.py, you can simply fix the file and press refresh in your browser – there is no need to restart the server.

Shutting Down the Server

To shut down the server, use the Launcher, select your application and press the Stop button.

Result:

Thus the GAE web applications was created.

EX.NO:5	Simulate a cloud scenario using CloudSim and run a scheduling algorithm that is not present in CloudSim
DATE:	

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 Net Beans 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 class path 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 <https://code.google.com/p/cloudsim/downloads/list> and unzip
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 Datacenter Characteristics object that stores the properties of a data centre such as architecture, OS, list of machines, allocation policy that covers the time or space shared, 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:

```
Datacenter Broker broker = create Broker ();
```

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 cloudlet Scheduler 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) Sample
```

Output from the Existing Example:

Starting

CloudSimExample1...

Initialising...

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 computers 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 Virtual Box shared folders with detailed step by step instructions.

You should remember that you are dealing 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 overcome 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-key gen 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: one admin, password: open nebula
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 data stores tab, and select all data stores
 - 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 steps
 - 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 migrates.

Before migration

Host:SACET

The screenshot shows the OpenNebula web interface in a browser window. The URL bar shows 'localhost:9869'. The interface has a sidebar on the left with navigation links: Dashboard, Instances (VMs, Services, Virtual Routers), Templates, Storage, Network, Infrastructure (Clusters, Hosts, Zones), System, and Settings. The main content area is titled 'Host 1 naaveenkumar' and shows a table of VMs. The table has columns for ID, Owner, Group, Name, Status, Host, and IPs. All VMs listed have a status of 'FAILURE'. Below the table, it says 'Showing 1 to 6 of 6 entries'.

ID	Owner	Group	Name	Status	Host	IPs
5	oneadmin	oneadmin	vm2	FAILURE	naaveenkumar	172.16.100.205
4	oneadmin	oneadmin	vm2	FAILURE	naaveenkumar	172.16.100.204
3	oneadmin	oneadmin	vm1	FAILURE	naaveenkumar	172.16.100.203
2	oneadmin	oneadmin	naveen	FAILURE	naaveenkumar	172.16.100.202
1	oneadmin	oneadmin	naveen	FAILURE	naaveenkumar	172.16.100.201
0	oneadmin	oneadmin	tylinux-Q	FAILURE	naaveenkumar	172.16.100.200

Host:one-sandbox

The screenshot shows the OpenNebula web interface at localhost:9869. The 'Hosts' page is active, displaying a table of hosts. The host 'one-sandbox' is selected, showing two running VMs.

ID	Owner	Group	Name	Status	Host	IPs
7	oneadmin	oneadmin	vm7	RUNNING	one-sandbox	172.16.100.207
6	oneadmin	oneadmin	vm6	RUNNING	one-sandbox	172.16.100.206

Showing 1 to 2 of 2 entries

The screenshot shows the 'Migrate Virtual Machine' dialog box in the OpenNebula web interface. It displays a table of hosts available for migration.

ID	Name	Cluster	RVMs	Allocated CPU	Allocated MEM	Status
2	raa	default	0	0/0	0KB/0	RETRY
1	naveenkumar	rama	8	82/0	481KB/0	ERROR
0	one-sandbox	rama	2	20/100 (20%)	41B/741B (1%)	ON

Showing 1 to 3 of 3 entries

Advanced Options

Migrate

After Migration:

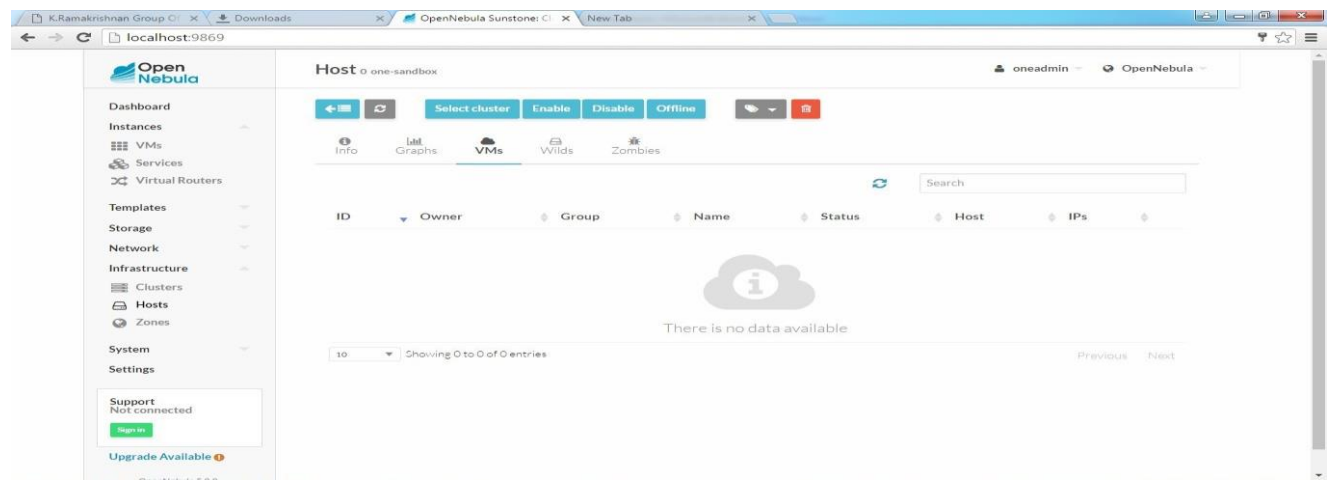
The screenshot shows the OpenNebula web interface after migration. The 'Hosts' page displays the updated status of the hosts.

ID	Name	Cluster	RVMs	Allocated CPU	Allocated MEM	Status
2	raa	default	0	0/0	0KB/0	ERROR
1	naveenkumar	rama	8	82/0	481KB/0	ERROR
0	one-sandbox	rama	0	0/100 (0%)	0KB/741B (0%)	ON

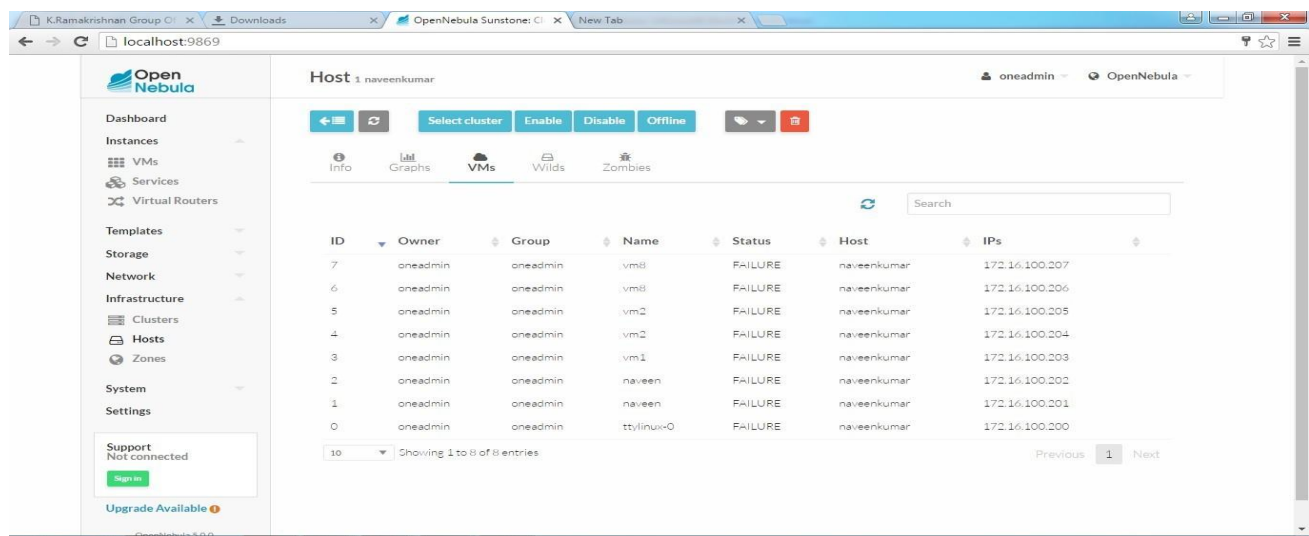
Showing 1 to 3 of 3 entries

3 TOTAL 1 ON 0 OFF 2 ERROR

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.

EXNO.:7	Install Hadoop single node cluster and run simple applications like word count
DATE:	

Aim:

To Install Hadoop single node cluster and run simple applications like word count.

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`

```
edureka@localhost:~$ tar -xvf jdk-8u101-linux-i586.tar.gz
```

Fig: Hadoop Installation–Extracting Java Files

Step3:DownloadtheHadoop2.7.3Package.

Command: `wget-https://archive.apache.org/dist/hadoop/core/hadoop-2.7.3/hadoop- 2.7.3.tar.gz`

```
edureka@localhost:~$ wget https://archive.apache.org/dist/hadoop/core/hadoop-2.7.3/hadoop-2.7.3.tar.gz
```

Fig: Hadoop Installation–Downloading Hadoop

Step4: Extract the Hadoop tar File.

Command: `tar-xvfhadoop-2.7.3.tar.gz`

```
edureka@localhost:~$ 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.

Command: `vi .bashrc`

```
edureka@localhost:~$ vi .bashrc
```

Fig: Hadoop Installation–Setting Environment Variable

```
# User specific aliases and functions

export HADOOP_HOME=$HOME/hadoop-2.7.3
export HADOOP_CONF_DIR=$HOME/hadoop-2.7.3/etc/hadoop
export HADOOP_MAPRED_HOME=$HOME/hadoop-2.7.3
export HADOOP_COMMON_HOME=$HOME/hadoop-2.7.3
export HADOOP_HDFS_HOME=$HOME/hadoop-2.7.3
export YARN_HOME=$HOME/hadoop-2.7.3
export PATH=$PATH:$HOME/hadoop-2.7.3/bin

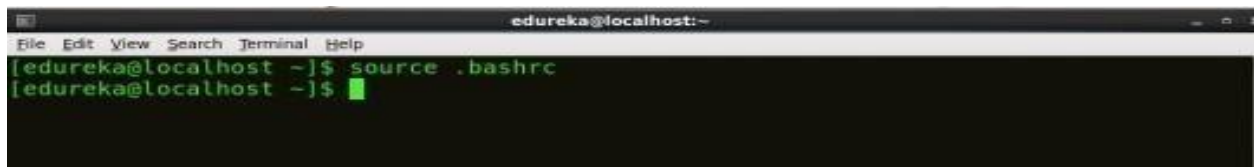
# Set JAVA_HOME
export JAVA_HOME=/home/edureka/jdk1.8.0_101
export PATH=/home/edureka/jdk1.8.0_101/bin:$PATH
```

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

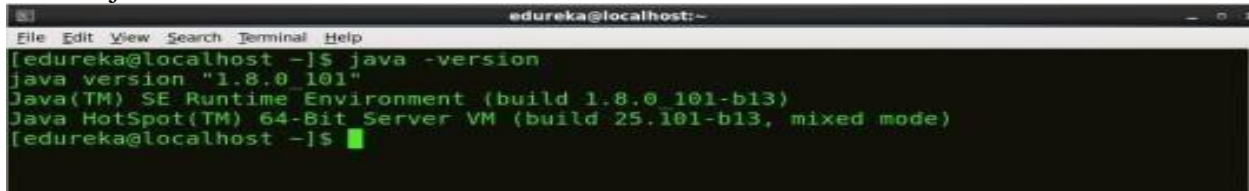


```
edureka@localhost:~  
File Edit View Search Terminal Help  
[edureka@localhost ~]$ source .bashrc  
[edureka@localhost ~]$
```

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.

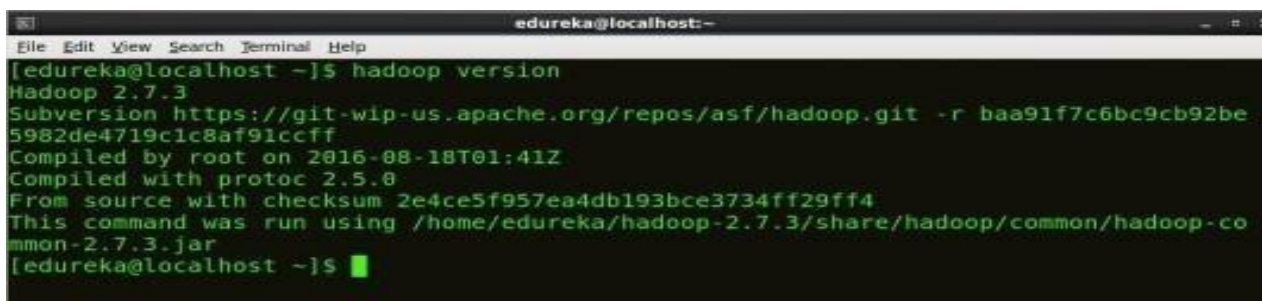
Command: java -version



```
edureka@localhost:~  
File Edit View Search Terminal Help  
[edureka@localhost ~]$ java -version  
java version "1.8.0_101"  
Java(TM) SE Runtime Environment (build 1.8.0_101-b13)  
Java HotSpot(TM) 64-Bit Server VM (build 25.101-b13, mixed mode)  
[edureka@localhost ~]$
```

Fig: Hadoop Installation – Checking Java Version

Command: hadoop version



```
edureka@localhost:~  
File Edit View Search Terminal Help  
[edureka@localhost ~]$ hadoop version  
Hadoop 2.7.3  
Subversion https://git-wip-us.apache.org/repos/asf/hadoop.git -r baa91f7c6bc9cb92be5982de4719c1c8af91ccff  
Compiled by root on 2016-08-18T01:41Z  
Compiled with protoc 2.5.0  
From source with checksum 2e4ce5f957ea4db193bce3734ff29ff4  
This command was run using /home/edureka/hadoop-2.7.3/share/hadoop/common/hadoop-common-2.7.3.jar  
[edureka@localhost ~]$
```

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:

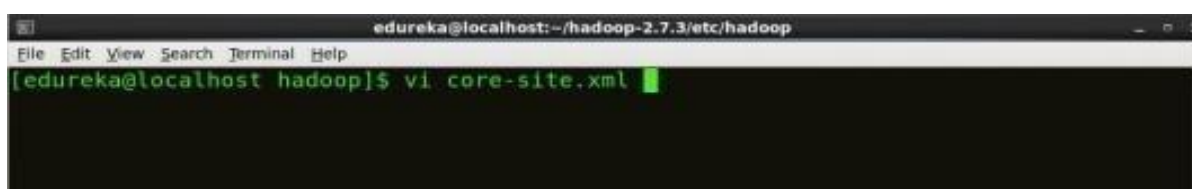


```
edureka@localhost:~/hadoop-2.7.3/etc/hadoop  
File Edit View Search Terminal Help  
[edureka@localhost ~]$ cd hadoop-2.7.3/etc/hadoop/  
[edureka@localhost hadoop]$ ls  
capacity-scheduler.xml  httpfs-env.sh  mapred-env.sh  
configuration.xml      httpfs-log4j.properties  mapred-queues.xml.template  
container-executor.cfg  httpfs-signature.secret  mapred-site.xml.template  
core-site.xml           httpfs-site.xml  slaves  
hadoop-env.cmd          kms-acls.xml     ssl-client.xml.example  
hadoop-env.sh           kms-env.sh       ssl-server.xml.example  
hadoop-metrics2.properties  kms-log4j.properties  yarn-env.cmd  
hadoop-metrics.properties  kms-site.xml      yarn-env.sh  
hadoop-policy.xml        log4j.properties  yarn-site.xml  
hdfs-site.xml            mapred-env.cmd  
[edureka@localhost hadoop]$
```

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 Name Node runs in the cluster. It contains configuration settings of Hadoop core such as I/O settings that are common to HDFS & Map Reduce.

Command: vi core-site.xml



```
edureka@localhost:~/hadoop-2.7.3/etc/hadoop  
File Edit View Search Terminal Help  
[edureka@localhost hadoop]$ vi core-site.xml
```

Fig: Hadoop Installation – Configuring core-site.xml


```

<configuration>
<property>
<name>fs.default.name</name>
<value>hdfs://localhost:9000</value>
</property>
</configuration>

```

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

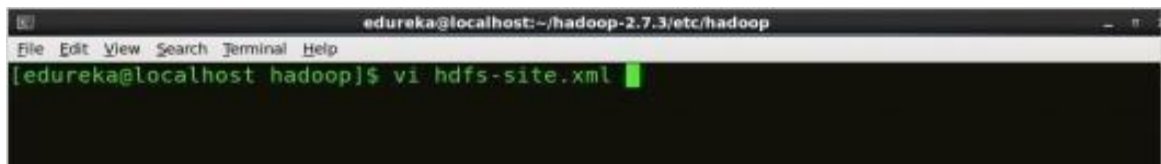


Fig: Hadoop Installation – Configuring hdfs-site.xml

```

<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
<property>
<name>dfs.permission</name>
<value>>false</value>
</property>

```

Fig: Hadoop Installation – Configuring hdfs-site.xml

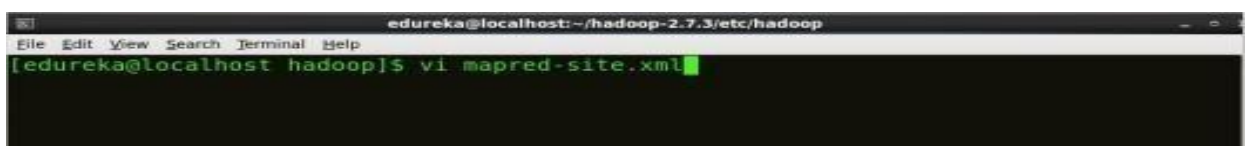
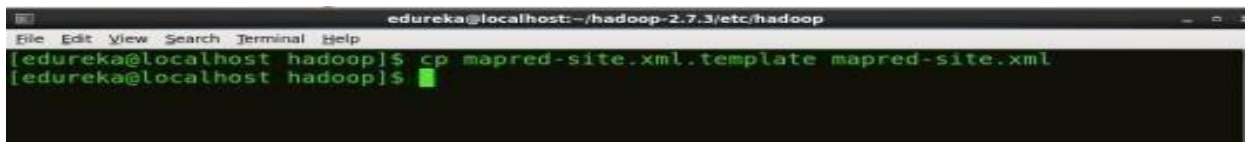
Step 9: Edit the *mapred-site.xml* file and edit the property mentioned below **inside configuration tag**:

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



```

<configuration>
<property>
<name>mapreduce.framework.name</name>
<value>yarn</value>
</property>
</configuration>

```

Fig: Hadoop Installation–Configuring mapred-site.xml

```

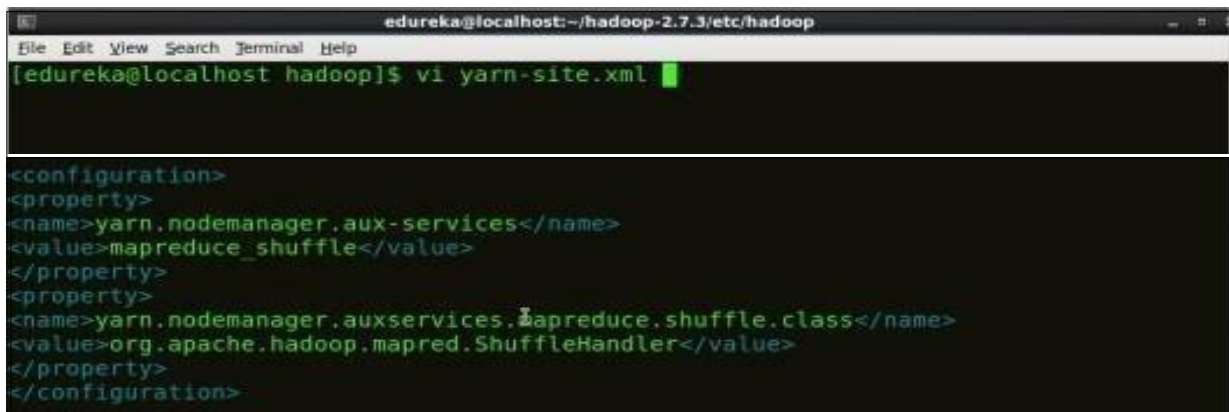
<configuration>
<property>
<name>dfs.replication</name>
<value>1</value>
</property>
<property>
<name>dfs.permission</name>
<value>>false</value>
</property>

```

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



```
edureka@localhost:~/hadoop-2.7.3/etc/hadoop
File Edit View Search Terminal Help
[edureka@localhost hadoop]$ vi yarn-site.xml

<configuration>
<property>
<name>yarn.nodemanager.aux-services</name>
<value>mapreduce_shuffle</value>
</property>
<property>
<name>yarn.nodemanager.auxservices.mapreduce.shuffle.class</name>
<value>org.apache.hadoop.mapred.ShuffleHandler</value>
</property>
</configuration>
```

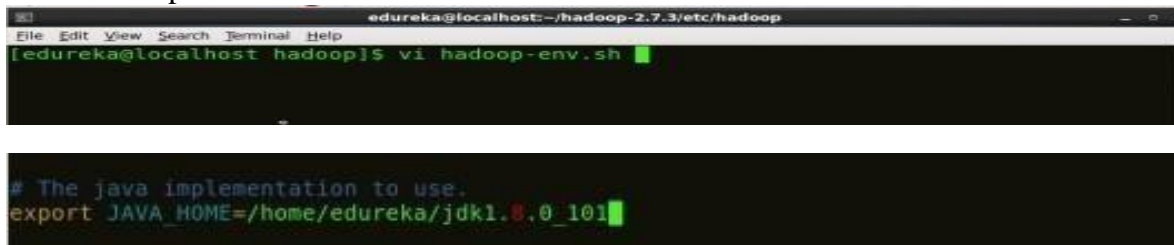
Fig: Hadoop Installation–Configuring yarn-site.xml

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

```
1
2      <?xmlversion="1.0">
3      <configuration>
4      <property>
5      <property>
8      name>varn.nodemanager.auxservices.mapreduce.shuffle.class</name>
9      <value>org.apache.hadoop.mapred.ShuffleHandler</value>
10
11
```

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



```
edureka@localhost:~/hadoop-2.7.3/etc/hadoop
File Edit View Search Terminal Help
[edureka@localhost hadoop]$ vi hadoop-env.sh

# The java implementation to use.
export JAVA_HOME=/home/edureka/jdk1.8.0_101
```

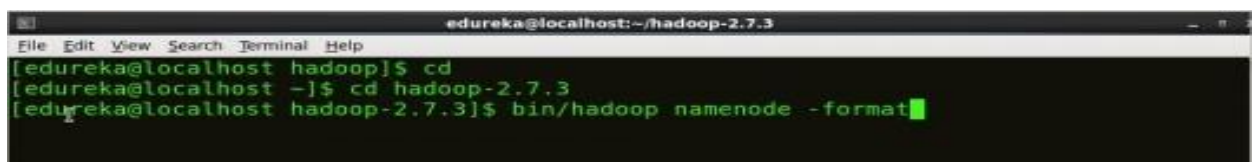
Fig: Hadoop Installation – Configuring hadoop-env.sh

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

Command: cd

Command: cd hadoop-2.7.3

Command: bin/hadoop namenode -format



```
edureka@localhost:~/hadoop-2.7.3
File Edit View Search Terminal Help
[edureka@localhost hadoop]$ cd
[edureka@localhost ~]$ cd hadoop-2.7.3
[edureka@localhost hadoop-2.7.3]$ bin/hadoop namenode -format
```

Fig: Hadoop Installation – Formatting Name Node

This formats the HDFS via Name Node. 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 file system. You will lose all your data stored in the HDFS.

Step 13: Once the Name Node 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.

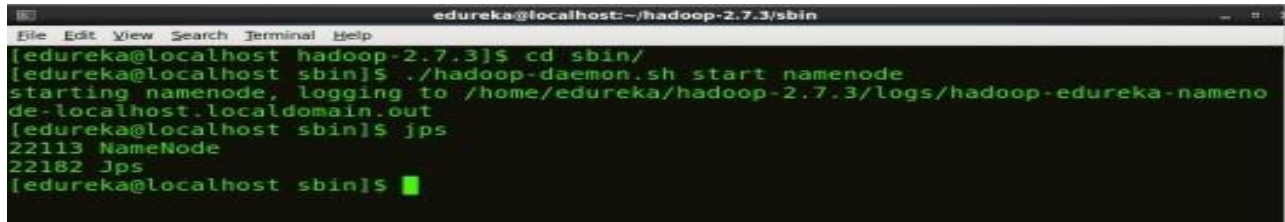
Command: `./start-all.sh`

The above command is a combination of *start-dfs.sh*, *start-yarn.sh* & *mr-jobhistorydaemon.sh*
Or you can run all the services individually as below:

Start Name Node:

The Name Node is the centerpiece of an HDFS file system. It keeps the directory tree of all files stored in the HDFS and tracks the entire file stored across the cluster.

Command: `./hadoop-daemon.sh start name node`



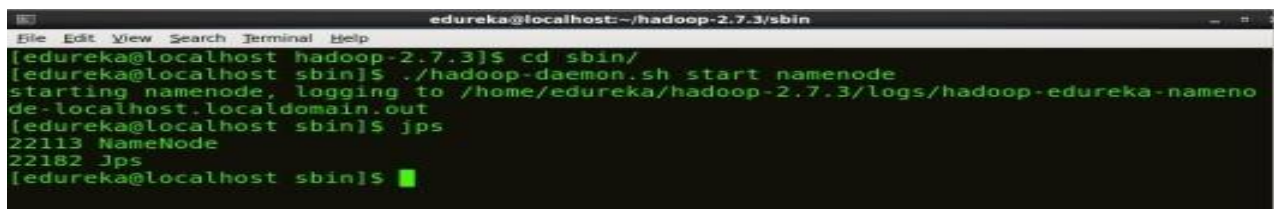
```
edureka@localhost:~/hadoop-2.7.3/sbin
[edureka@localhost hadoop-2.7.3]$ cd sbin/
[edureka@localhost sbin]$ ./hadoop-daemon.sh start namenode
starting namenode, logging to /home/edureka/hadoop-2.7.3/logs/hadoop-edureka-nameno
de-localhost.localdomain.out
[edureka@localhost sbin]$ jps
22113 NameNode
22182 Jps
[edureka@localhost sbin]$
```

Fig: Hadoop Installation – Starting Name Node

Start Data Node:

On startup, a Data Node connects to the Name node and it responds to the requests from the Name node for different operations.

Command: `./hadoop-daemon.sh start data node`



```
edureka@localhost:~/hadoop-2.7.3/sbin
[edureka@localhost hadoop-2.7.3]$ cd sbin/
[edureka@localhost sbin]$ ./hadoop-daemon.sh start namenode
starting namenode, logging to /home/edureka/hadoop-2.7.3/logs/hadoop-edureka-nameno
de-localhost.localdomain.out
[edureka@localhost sbin]$ jps
22113 NameNode
22182 Jps
[edureka@localhost sbin]$
```

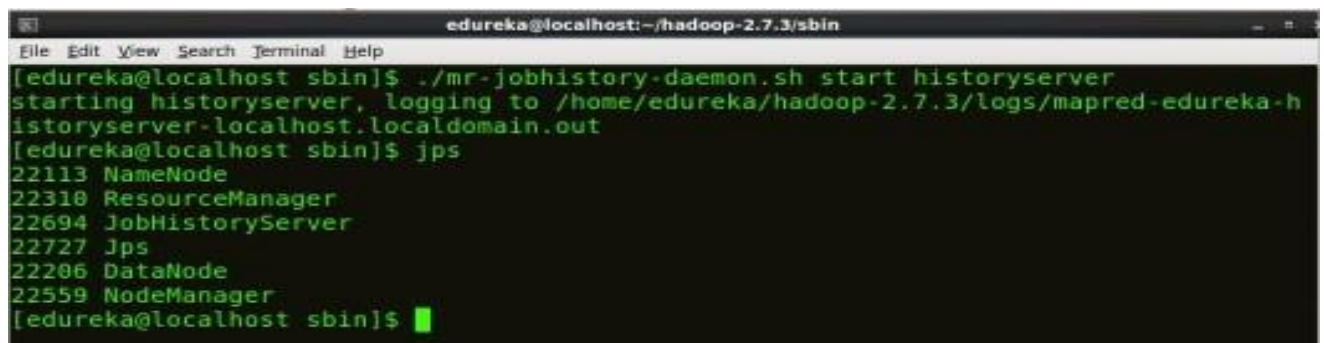
Fig: Hadoop Installation – Starting Node Manager

Start Job History Server:

Job History Server is responsible for servicing all job history related requests from client.

Command: `./mr-jobhistory-daemon.sh start history server`

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



```
edureka@localhost:~/hadoop-2.7.3/sbin
[edureka@localhost sbin]$ ./mr-jobhistory-daemon.sh start historyserver
starting historyserver, logging to /home/edureka/hadoop-2.7.3/logs/mapred-edureka-h
istoryserver-localhost.localdomain.out
[edureka@localhost sbin]$ jps
22113 NameNode
22310 ResourceManager
22694 JobHistoryServer
22727 Jps
22206 DataNode
22559 NodeManager
[edureka@localhost sbin]$
```

Command: `jps`

Fig: Hadoop Installation – Checking Daemons

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



Node interface.

Fig: Hadoop Installation – Starting Web UI

Congratulations, you have successfully installed a single node Hadoop cluster

Result:

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

EX.NO.:8	Creating and Executing Your First Container Using Docker.
DATE:	

AIM:

To create and execute your first container using Docker.

PROCEDURE

To follow the instructions specific to your operating system. Here are the general steps for the most popular operating systems:

Windows:

- Visit the Docker website (<https://www.docker.com/>) and navigate to the "Get Docker" section.
- Click on the "Download for Windows" button to download the Docker Desktop installer.
- Run the installer and follow the prompts. During the installation, Docker may require you to enable Hyper-V and Containers features, so make sure to enable them if prompted.
- Once the installation is complete, Docker Desktop will be installed on your Windows machine. You can access it from the Start menu or the system tray.

Mac:

- Visit the Docker website (<https://www.docker.com/>) and navigate to the "Get Docker" section.
- Click on the "Download for Mac" button to download the Docker Desktop installer.
- Run the installer and drag the Docker icon to the Applications folder to install Docker Desktop.
- Launch Docker Desktop from the Applications folder or the Launchpad. It will appear in the status bar at the top of your screen.

Linux:

Docker supports various Linux distributions. The exact installation steps may vary based on your distribution. Here's a general outline:

- Visit the Docker website (<https://www.docker.com/>) and navigate to the "Get Docker" section.
- Click on the "Download for Linux" button.
- Docker provides installation instructions for various Linux distributions such as Ubuntu, CentOS, Debian, Fedora, and more. Follow the instructions specific to your distribution.
- Once Docker is installed, start the Docker service using the appropriate command for your Linux distribution.

After completing the installation, you can open a terminal or command prompt and run the `docker -v` command to verify that Docker is installed correctly. It should display the version of Docker installed on your system.

That's it! You now have Docker installed on your machine and can start using it to manage containers.

Install Docker: First, you need to install Docker on your machine. Docker provides platform-specific installation instructions on their website for different operating systems. Follow the instructions to install Docker for your particular OS.

Docker Image: Docker containers are created based on Docker images. An image is a lightweight, standalone, and executable package that includes everything needed to run a piece of software, including the code, runtime, libraries, and system tools. Docker Hub (hub.docker.com) is a popular online repository of Docker images. You can search for existing images on Docker Hub or create your own. For this example, we'll use an existing image.

Pull an Image: Open your terminal or command prompt and execute the following command to pull an existing Docker image from Docker Hub. We'll use the official hello-world image as an example:

Copy code

docker pull hello-world

Docker will download the image from the Docker Hub repository.

Run a Container: Once you have the Docker image, you can create and run a container based on that image. Execute the following command to run the hello-world container:

arduino Copy code

docker run hello-world

Docker will create a container from the image and execute it. The container will print a "Hello from Docker!" message along with some information about your Docker installation.

Note: If you haven't pulled the hello-world image in the previous step, Docker will automatically download it before running the container.

Congratulations! You've created and executed your first Docker container. Docker will handle the container lifecycle, including starting, stopping, and managing resources for you. This simple example demonstrates the basic concept of running a container using Docker.

You can explore further by trying out different Docker images and running more complex applications within containers.

Result:

Thus the first container using docker is created and executed successfully.

EX.NO.:9	Run a Container from Docker Hub
DATE:	

AIM:

To write a program to run a container from Docker hub.

PROCEDURE:

Run a container from docker hub Run
docker -h,

\$ docker -h

Flag shorthand -h has been deprecated, please use --help Usage:

docker [OPTIONS] COMMAND

A self-sufficient runtime for containers

...

Management Commands:

builder Manage builds

config Manage Docker configs

container Manage containers engine

Manage the docker engine

image Manage images

network Manage networks

node Manage Swarm nodes

plugin Manage plugins

secret Manage Docker secrets

service Manage services

stack Manage Docker stacks

swarm Manage Swarm

system Manage Docker

trust Manage trust on Docker images

volume Manage volumes

The Docker command line can be used to manage several features of the Docker Engine. In this lab, we will mainly focus on the container command.

If podman is installed, you can run the alternative command for comparison. sudo

podman -h

You can additionally review the version of your Docker installation,
docker version

Client:

Version: 19.03.6

...

Server: Docker Engine - Community

Engine

Version: 19.03.5

...

sudo podman version --events-backend=none

Version: 2.1.1

API Version: 2.0.0
Go Version: go1.15.2
Built: Thu Jan 1 00:00:00 1970
OS/Arch: linux/amd64

Step 1: Run your first container

We are going to use the Docker CLI to run our first container. Open a terminal on your local computer

Run docker container run -t ubuntu top

Use the docker container run command to run a container with the ubuntu image using the top command. The -t flags allocate a pseudo-TTY which we need for the top to work correctly.

```
$ Docker container run -it ubuntu top Unable
to find image 'ubuntu: latest' locally latest:
Pulling from library/ubuntu aafe6b5e13de:
Pull complete 0a2b43a72660: Pull complete
18bdd1e546d2: Pull complete 8198342c3e05:
Pull complete f56970a44fd4: Pull complete
Digest: sha256:f3a61450ae43896c4332bda5e78b453f4a93179045f20c8181043b26b5e79028 Status:
Downloaded newer image for ubuntu: latest
```

The docker run command will result first in a docker pull to download the ubuntu image onto your host. Once it is downloaded, it will start the container. The output for the running container should look like this:

```
top - 20:32:46 up 3 days, 17:40, 0 users, load average: 0.00, 0.01, 0.00
Tasks: 1 total, 1 running, 0 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.0 us, 0.1 sy, 0.0 ni, 99.9 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
```

```
KiB Mem : 2046768 total, 173308 free, 117248 used, 1756212 buff/cache
KiB Swap: 1048572 total, 1048572 free, 0 used. 1548356 avail Mem
```

```
PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+
COMMAND 1 root 20 0 36636 3072 2640 R 0.3 0.2 0:00.04 top
```

Inspect the container with docker container exec

The docker container exec command is a way to "enter" a running container's namespaces with a new process.

Open a new terminal. On cognitiveclass.ai, select Terminal > New Terminal.

Using play-with-docker.com, to open a new terminal connected to node1, click "Add New Instance" on the lefthand side, then ssh from node2 into node1 using the IP that is listed by 'node1 '. For example:

```
[node2] (local) root@192.168.0.17 ~
$ ssh 192.168.0.18
[node1] (local) root@192.168.0.18 ~
$
```

In the new terminal, use the docker container ls command to get the ID of the running container you just created.

```
$ docker container ls
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
b3ad2a23fab3	ubuntu	"top"	29 minutes ago	Up 29 minutes	

```
NAMES
```

goofy_nobel

```
$ docker container exec -it b3ad2a23fab3 bash
```

```
root@b3ad2a23fab3:/#
```

And Voila! We just used the docker container exec command to "enter" our container's namespaces with our bash process. Using docker container exec with bash is a common pattern to inspect a docker container.

Notice the change in the prefix of your terminal. e.g. root@b3ad2a23fab3:/. This is an indication that we are running bash "inside" of our container.

From the same terminal, run ps -ef to inspect the running processes.

```
root@b3ad2a23fab3:/# ps -ef
UID PID PPID C STIME TTY TIME CMD
root 1 0 0 20:34 ? 00:00:00 top
root 17 0 0 21:06 ? 00:00:00 bash
root 27 17 0 21:14 ? 00:00:00 ps -ef
```

You should see only the top process, bash process and our ps process.

```
root@b3ad2a23fab3:/# exit
```

```
exit
```

```
$ ps -ef
```

```
# Lots of processes!
```

```
docker ps -a
```

```
docker rm <CONTAINER ID>
```

Step 2: Run Multiple Containers

Explore the Docker Hub

The Docker Hub is the public central registry for Docker images, which contains community and official images.

Run an Nginx server

Let's run a container using the official Nginx image from the Docker Hub.

```
$ docker container run --detach --publish 8080:80 --name nginx nginx
```

```
Unable to find image 'nginx:latest' locally
```

```
latest: Pulling from library/nginx
```

```
36a46ebd5019: Pull complete 57168433389f:
```

```
Pull complete 332ec8285c50: Pull complete
```

```
Digest: sha256:c15f1fb8fd55c60c72f940a76da76a5fccce2fefa0dd9b17967b9e40b0355316 Status:
```

```
Downloaded newer image for nginx:latest
```

```
5e1bf0e6b926bd73a66f98b3cbe23d04189c16a43d55dd46b8486359f6fdf048 Nginx is a lightweight web server. You can access it on port 8080 on your localhost.
```

```
Access the nginx server on localhost:8080. curl
```

```
localhost:8080
```

will return the HTML home page of Nginx,

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<title>Welcome to nginx!</title>
```

```
<style>
```

```
body
```

```
{
```

```
width: 35em; margin:
```

```

0 auto;
font-family: Tahoma, Verdana, Arial, sans-serif;
}
</style>
</head>
<body>

```

```
<h1>Welcome to nginx!</h1>
```

If you are using play-with-docker, look for the 8080 link near the top of the page, or if you run a Docker client with access to a local browser,

Run a mongo DB server

Now, run a mongoDB server. We will use the official mongoDB image from the Docker Hub. Instead of using the latest tag (which is the default if no tag is specified), we will use a specific version of the mongo image.

```
$ docker container run --detach --publish 8081:27017 --name mongo mongo:4.4 Unable to find image mongo:4.4 locally
```

```
4.4: Pulling from library/mongo
```

```
d13d02fa248d: Already exists
```

```
bc8e2652ce92: Pull complete
```

```
3cc856886986: Pull complete
```

```
c319e9ec4517: Pull complete
```

```
b4cbf8808f94: Pull complete
```

```
cb98a53e6676: Pull complete
```

```
f0485050cd8a: Pull complete
```

```
ac36cdc414b3: Pull complete
```

```
61814e3c487b: Pull complete
```

```
523a9f1da6b9: Pull complete
```

```
3b4beaef77a2: Pull complete
```

```
Digest: sha256:d13c897516e497e898c229e2467f4953314b63e48d4990d3215d876ef9d1fc7c Status:
```

```
Downloaded newer image for mongo:4.4
```

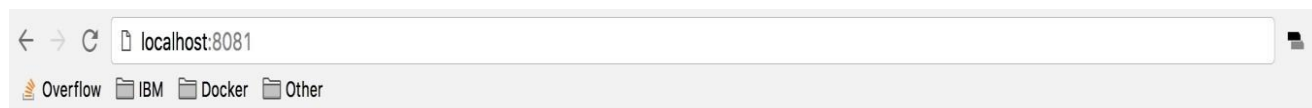
```
d8f614a4969fb1229f538e171850512f10f490cb1a96fca27e4aa89ac082eba5
```

Access localhost: 8081 to see some output from mongo.

```
curl localhost:8081
```

Which will return a warning from Mongo DB,

It looks like you are trying to access Mongo DB over HTTP on the native driver port. If you are using



It looks like you are trying to access MongoDB over HTTP on the native driver port. play-with-docker, look for the 8080 link near the top of the page.

Check your running containers with docker container ls

```
$ docker container ls
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
--------------	-------	---------	---------	--------	-------	-------

d6777df89fea	nginx	"nginx -g 'daemon ..."	Less than a second ago	Up 2 seconds	0.0.0.0:8080->80/tcp	nginx
--------------	-------	------------------------	------------------------	--------------	----------------------	-------

ead80a0db505	mongo	"docker-entrypoint..."	17 seconds ago	Up 19 seconds	0.0.0.0:8081->27017/tcp	mongo
--------------	-------	------------------------	----------------	---------------	-------------------------	-------

af549dccd5cf ubuntu "top" 5 minutes ago Up 5 minutes priceless_kepler

Step 3: Clean Up

First get a list of the containers running using docker container ls.

\$ docker container ls

CONTAINER NAMES	ID	IMAGE	COMMAND	CREATED	STATUS	PORTS
d6777df89fea	nginx	"nginx -g 'daemon ..."	3 minutes ago	Up 3 minutes	0.0.0.0:8080	->80/tcp
ead80a0db505	mongo	"docker-entrypoint..."	3 minutes ago	Up 3 minutes	0.0.0.0:8081	->27017/tcp
af549dccd5cf	ubuntu	"top"	8 minutes ago	Up 8 minutes	priceless_kepler	

Next, run docker container stop [container id] for each container in the list. You can also use the names of the containers that you specified before.

\$ docker container stop

d67 ead af5 d67

e
a
d
a
f
5

1. Remove the stopped containers

docker system prune is a really handy command to clean up your system. It will remove any stopped containers, unused volumes and networks, and dangling images.

\$ docker system

prune WARNING!

This will remove:

- all stopped containers
- all volumes not used by at least one container
- all networks not used by at least one container
- all dangling images

Are you sure you want to

continue? [y/N] y Deleted

Containers:

7872fd96ea4695795c41150a06067d605f69702dbcb9ce49492c9029f0e1b44b

60abd5ee65b1e2732ddc02b971a86e22de1c1c446dab165462a08b037ef7835c

31617fdd8e5f584c51ce182757e24a1c9620257027665c20be75aa3ab6591740

Total reclaimed space: 12B

Result:

Thus the program to run a container from Docker hub executed successfully.

EX.NO.:10	Find a procedure to launch virtual machine using trystack (Online Openstack Demo Version)
DATE:	

Aim:

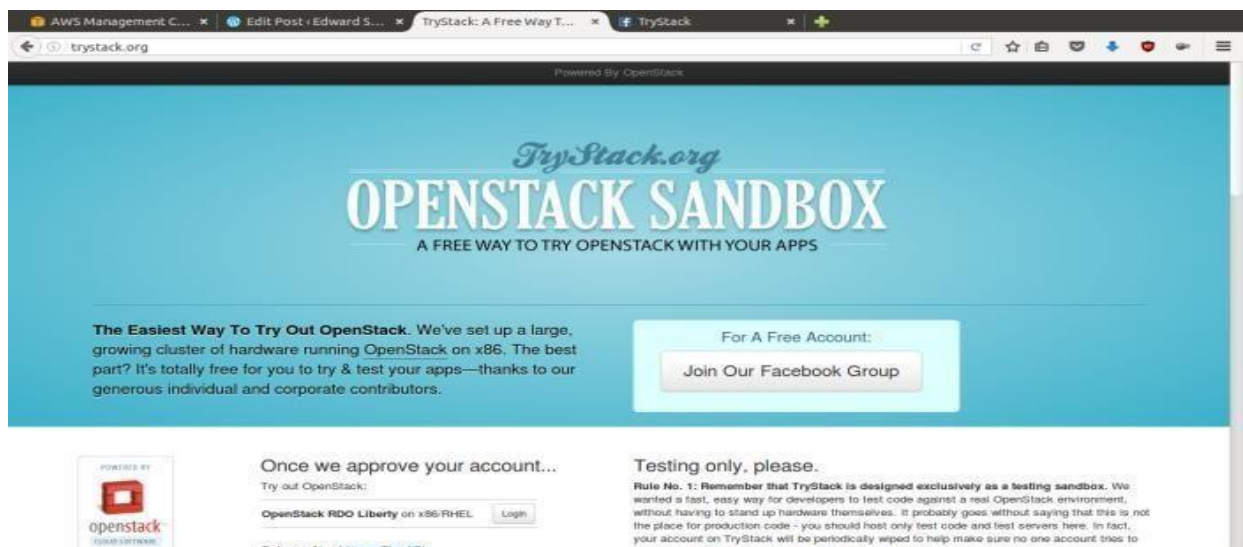
To Find a procedure to launch virtual machine using trystack.

Steps:

OpenStack is an open-source software cloud computing platform.

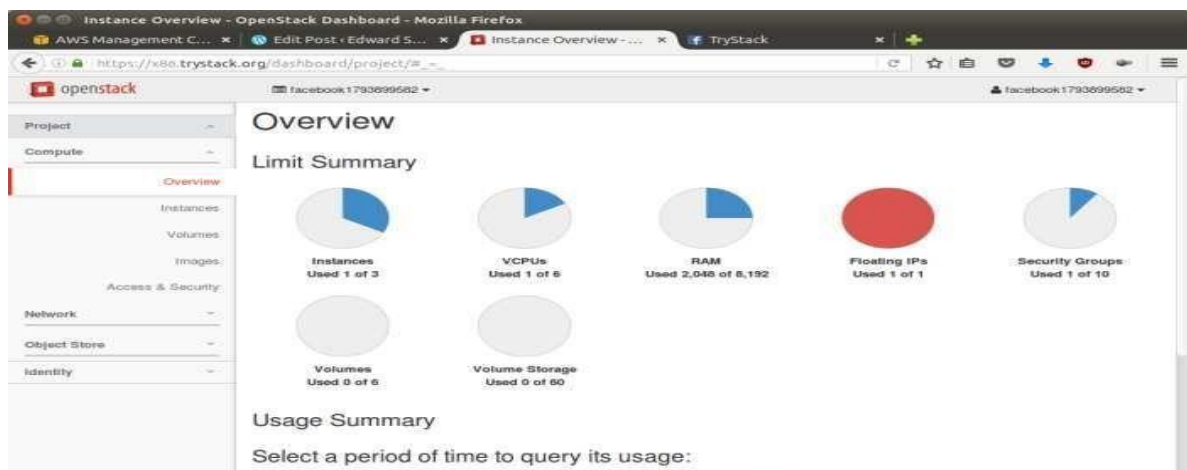
OpenStack is primarily used for deploying an infrastructure as a service (IaaS) solution like Amazon Web Service (AWS). In other words, you can *make your own AWS* by using OpenStack. If you want to try out OpenStack, **TryStack** is the easiest and free way to do it.

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.



TryStack.org Homepage

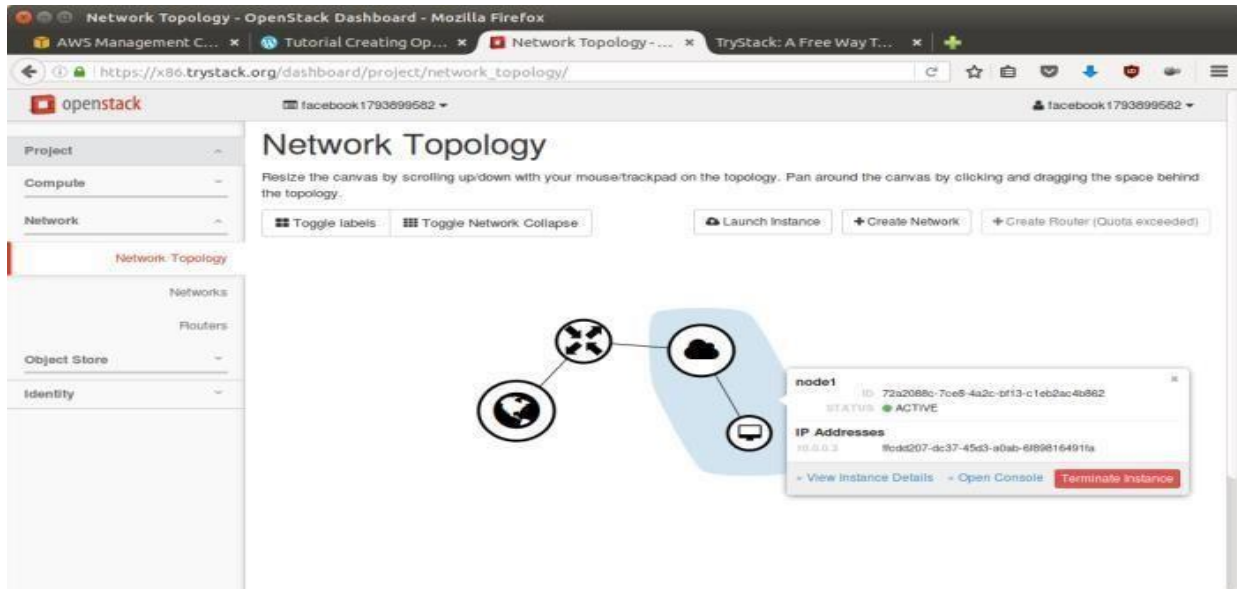
I assume that you already join to the Facebook Group and login to the dashboard. After you log in to the TryStack, you will see the Compute Dashboard like:



OpenStack Compute Dashboard

Overview: What we will do?

In this post, I will show you how to run an OpenStack instance. The instance will be accessible through the internet (have a public IP address). The final topology will like:



Network topology

As you see from the image above, the instance will be connected to a local network and the local network will be connected to internet.

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,
 - a. Fill **Network Address** with appropriate CIDR, for example 192.168.1.0/24. Use [private network CIDR block](#) as the best practice.
 - b. Select **IP Version** with appropriate IP version, in this case IPv4.
 - c. 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

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.

1. Go to **Compute > Instances** and then click **Launch Instance**.
2. In **Details** tab,
 - i. Fill **Instance Name**, for example Ubuntu 1.
 - ii. Select **Flavor**, for example m1.medium.
 - iii. Fill **Instance Count** with 1.
 - iv. Select **Instance Boot Source** with **Boot from Image**.
 - v. 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,
 - i. 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.
 - ii. In **Import Key Pair** dialog,
 - a. Fill **Key Pair Name** with your machine name (for example Edward-Key).
 - b. 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.
 - c. Click **Import key pair**.
 - iii. In **Security Groups**, mark/check **default**.
4. In Networking tab,
 1. 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:
 - a) Select External networks with external.
 - b) Then OK.
5. Click Add Interface button.
 - a) Select Subnet with the network that you have been created in Step 1.
 - b) 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 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.

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.