**CLOUD COMPUTING LAB RECORD**

**SUBMITTED IN PARTIAL FULFILLMENT OF THE REQURIEMENTS FOR THE AWARD OF DEGREE**

**OF**

**BACHELOR OF TECHNOLOGY**

**IN**

**COMPUTER SCIENCE**

**SUBMITTED BY:**

**J. VISHNU**

**UG ID: 22WU010101129**

**CSE B**

**UNDER SUPERVISION OF :**

**Prof.Vaishali Thakur**

****

**DEPARTMENT OF SCHOOL OF TECHNOLOGY**

**WOXSEN UNVERSITY**

**KAMKOLE,SADASIVPET ,HYDERABAD,TELANGANA 502345**

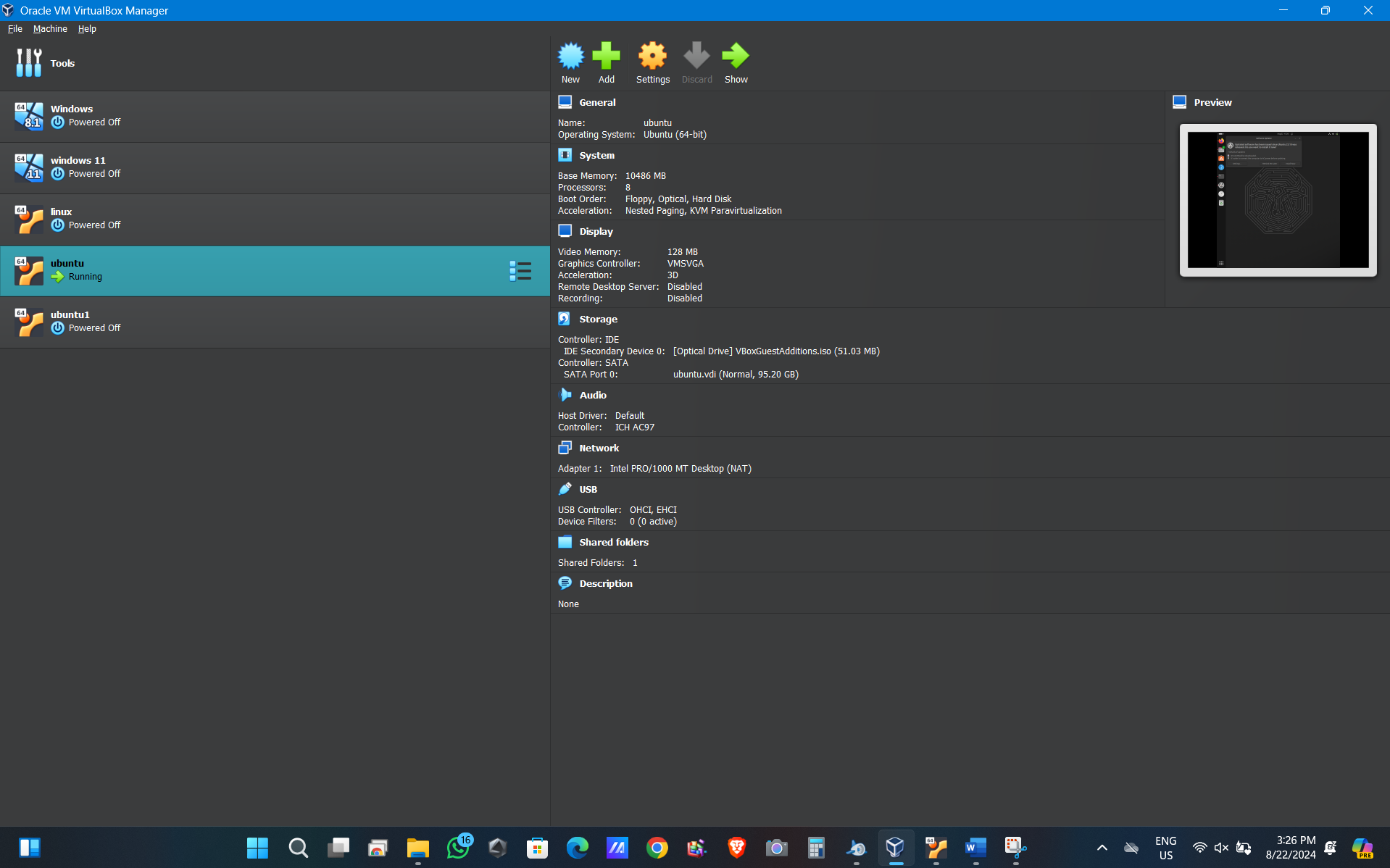
**TABLE OF CONTENTS**

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **TITLE** | **PG.NO** | **SIGNATURE** |
| **1** | Install Virtualbox with different flavours of linux or windows OS on top of windows7 or 8. | 3 |  |
| **2** | Install a Python compiler in the virtual machine created using virtual box and execute Simple Programs. | 3 |  |
| **3** | Install Google App Engine. Create hello world app and other simple web applications using python/java. | 7 |  |
| **4** | Use GAE launcher to launch the web applications. | **8** |  |
| **5** | Simulate a cloud scenario using CloudSim and run a scheduling algorithm that is not present in CloudSim. | **9** |  |
| **6** | Find a procedure to transfer the files from one virtual machine to another virtual machine. | **10** |  |
| **7** | Find a procedure to launch virtual machine using trystack (Online Openstack Demo Version) | **13** |  |
| **8** | Install Hadoop single node cluster and run simple applications like wordcount. | **17** |  |

**LAB 1**

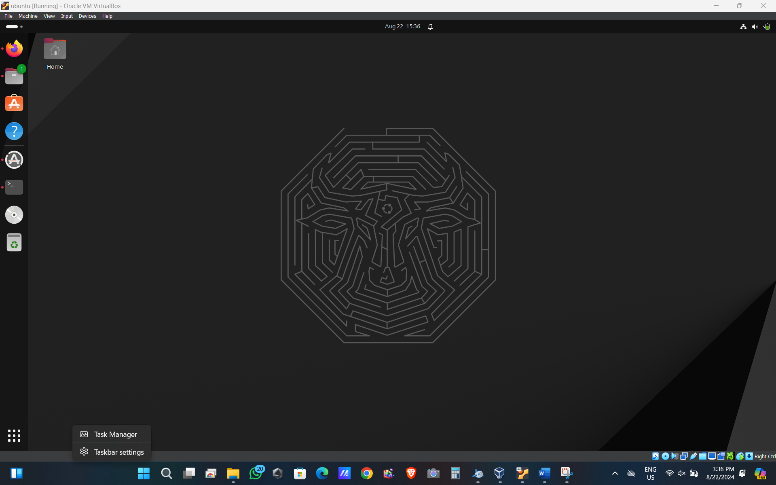
AIM : Install Virtualbox with different flavours of linux or windows OS on top of windows7 or 8.

Step 1 : Install Oracle Virtualbox.



Step 2: Install the Ubuntu Linux Iso File.

Step 3 : Install the Linux Os on the Virtualbox.

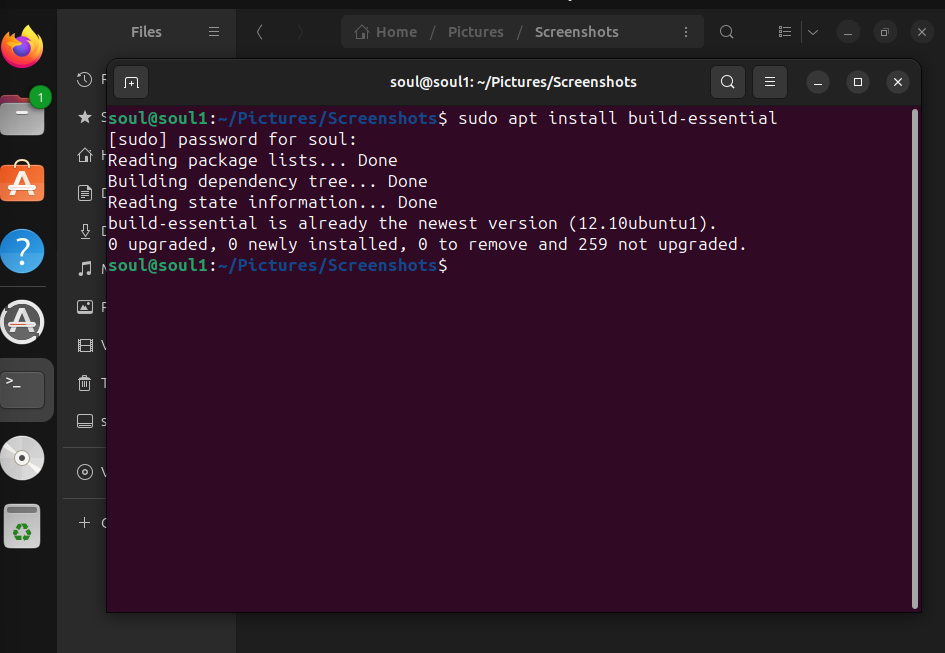


LAB 2

AIM: Install a Python compiler in the virtual machine created using virtual box and execute Simple Programs.

Step 1 : Command:- Sudo apt install build-essential

Comment :- Installs a package in Ubuntu that includes essential tools like the compiler, make utility, and libraries necessary for compiling software from source.



Step 2 Command:- Sudo apt upgrade

Comment :- Upgrades all installed packages on the system to their latest available versions, while keeping the current package versions intact without removing or installing new packages unless necessary.

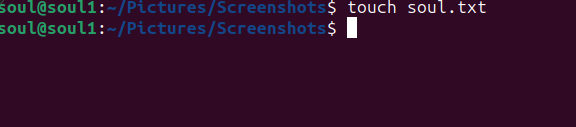


Step 3 Command:- Sudo apt install Gedit



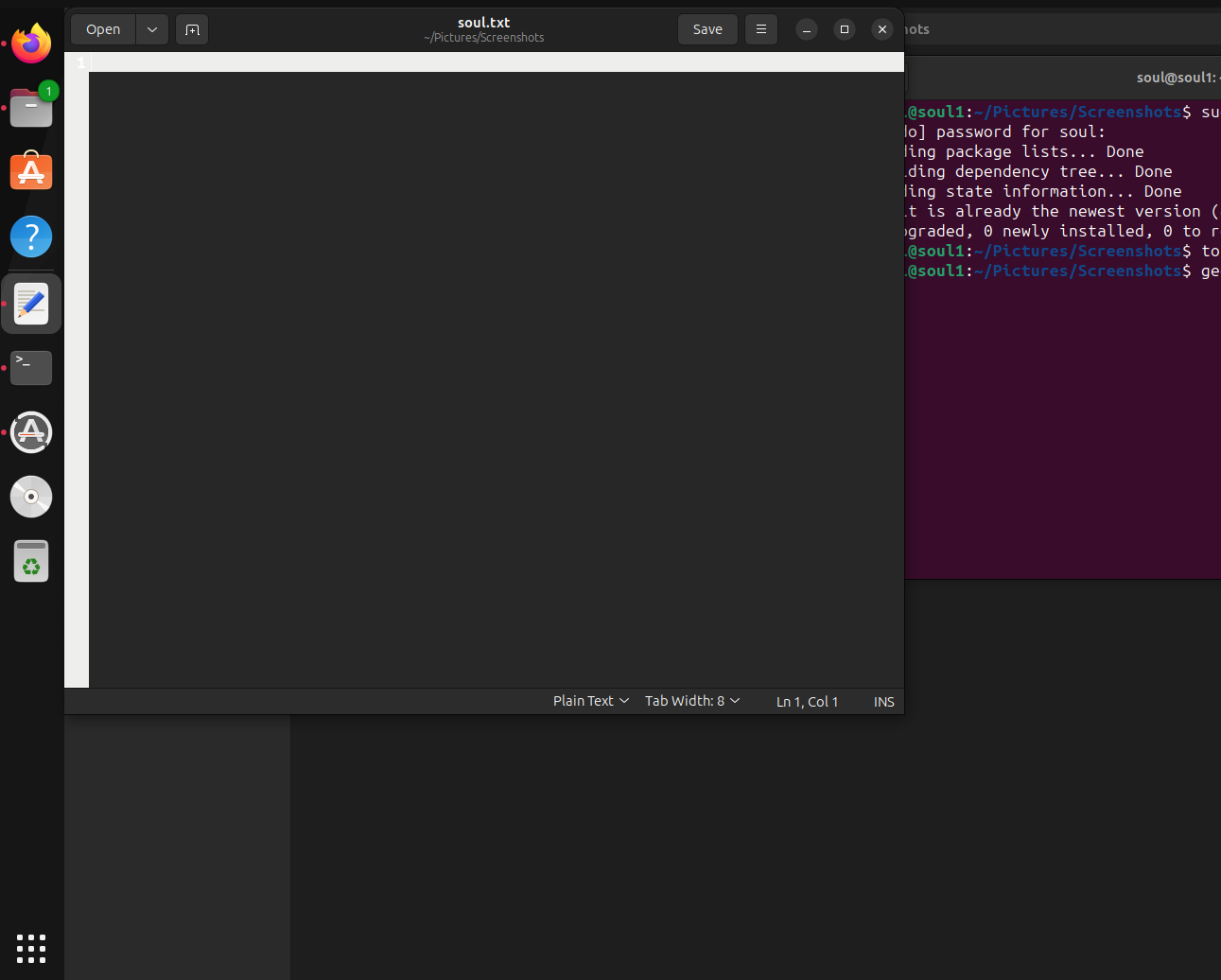
Step 4 Command :- Touch soul.txt

Comment :- For Creating a Text File We Use Touch/cat command

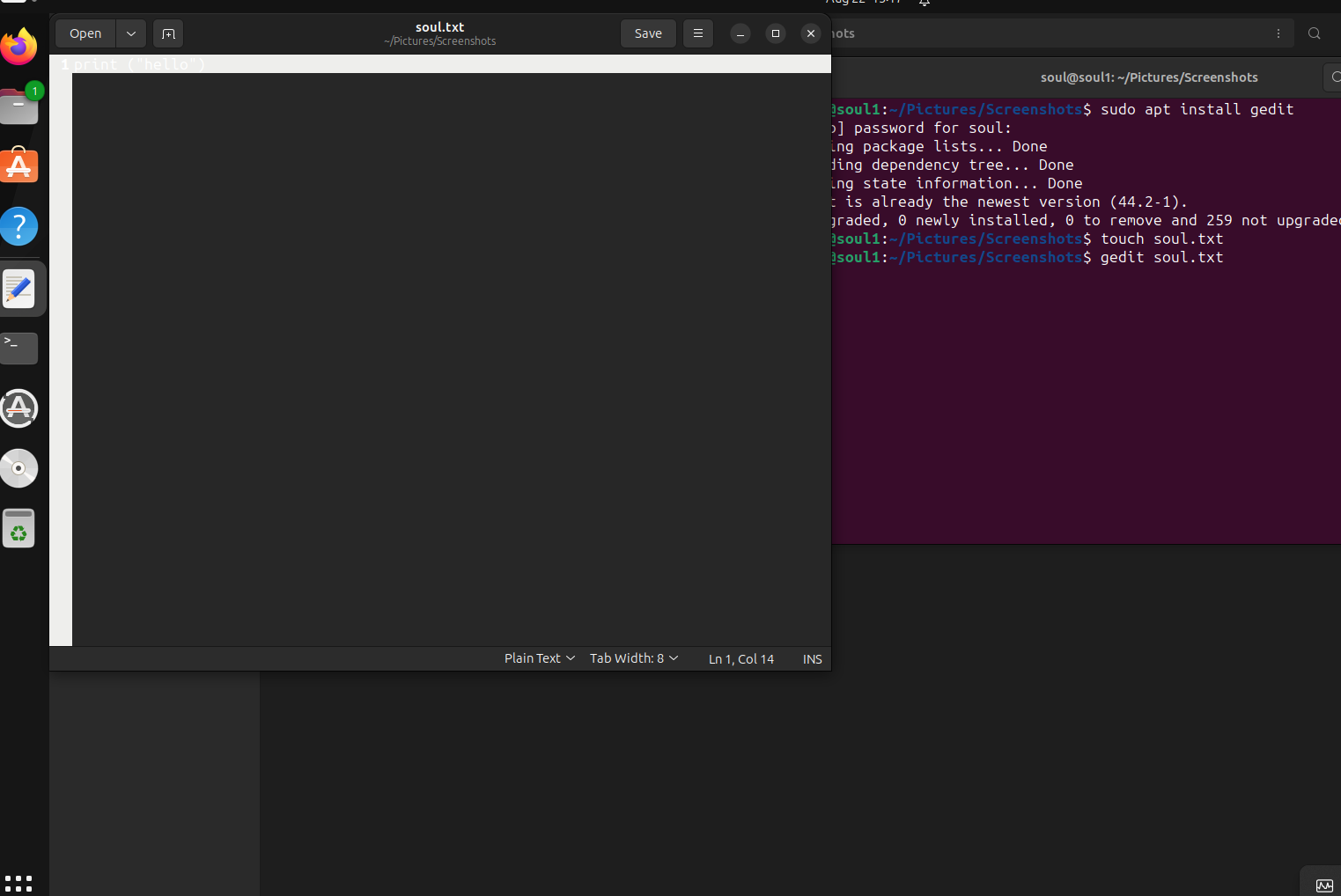


Step 5 : Command :- Gedit soul.txt

Comment :- it will open the file what we searched.

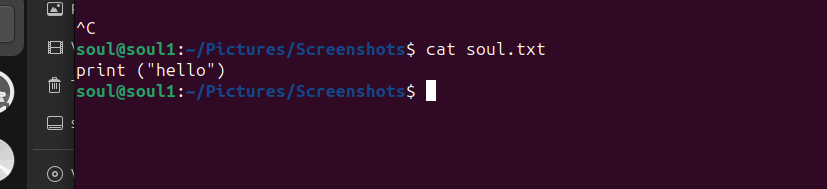


Step 6 : Write a python code in the window that appears and save the file.



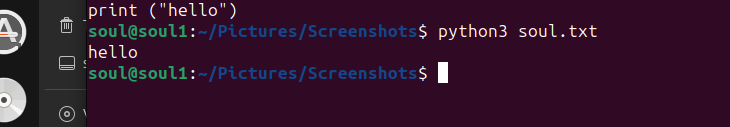
Step 7 : Cat [filename].txt

Comment :- to view content of a file



Step 8 : Python3 [filename].txt

Comment :- it will excute the python code

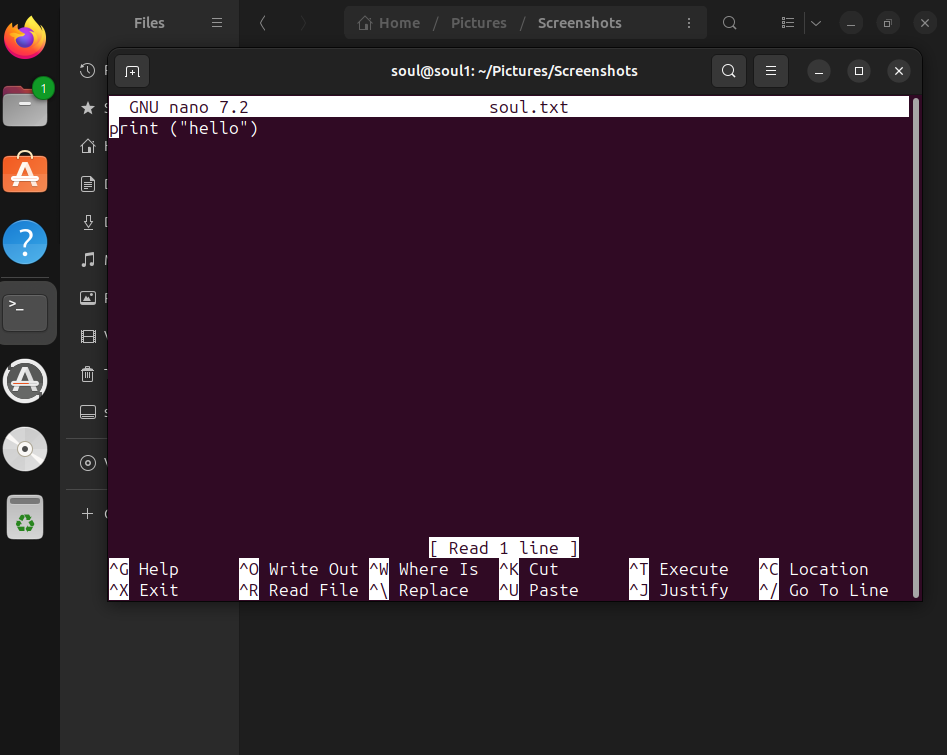


Step 9 Command- chmod u+x soul.txt

Comment :- chmod is used to change the file permission in unix like operation system. The file permission determine who can read write or execute the file.

Step 10 :Command- Nano [filename].txt

Comment – It Opens up an editor Integrated with the terminal.



Step 11: Python3 soul.txt

# 

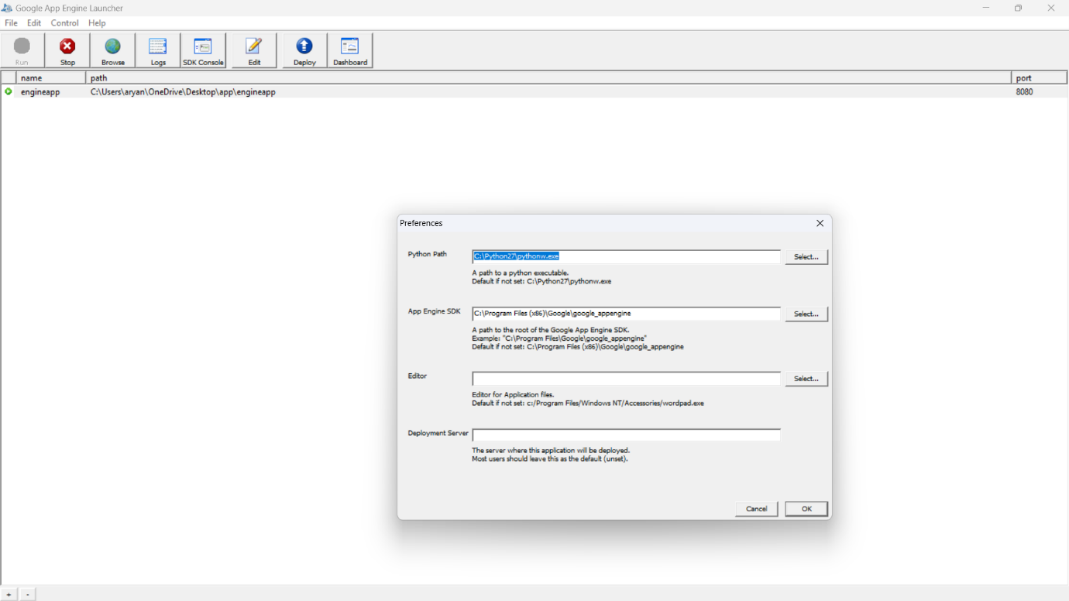
LAB 3

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

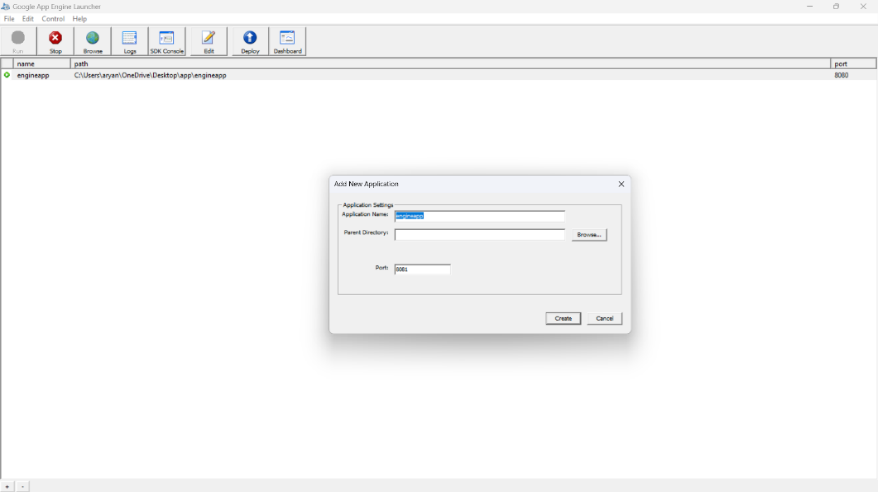
Step 1: Install Google App Engine.

Step 2: We add the Python path in the app engine preferences.

Step 3: We add the app engine files in the second option.



Step 4 :Create a folder called app anywhere and select the folder in the add new application section in the parent directory folder path and keep the port number as 8080.



* Link Preferences
* Create a folder on desktop
* Save to LocalHost

A computer screen shot of a computer

Description automatically generated

Lab-4

AIM- Use Google App Engine launcher to launch the web applications.

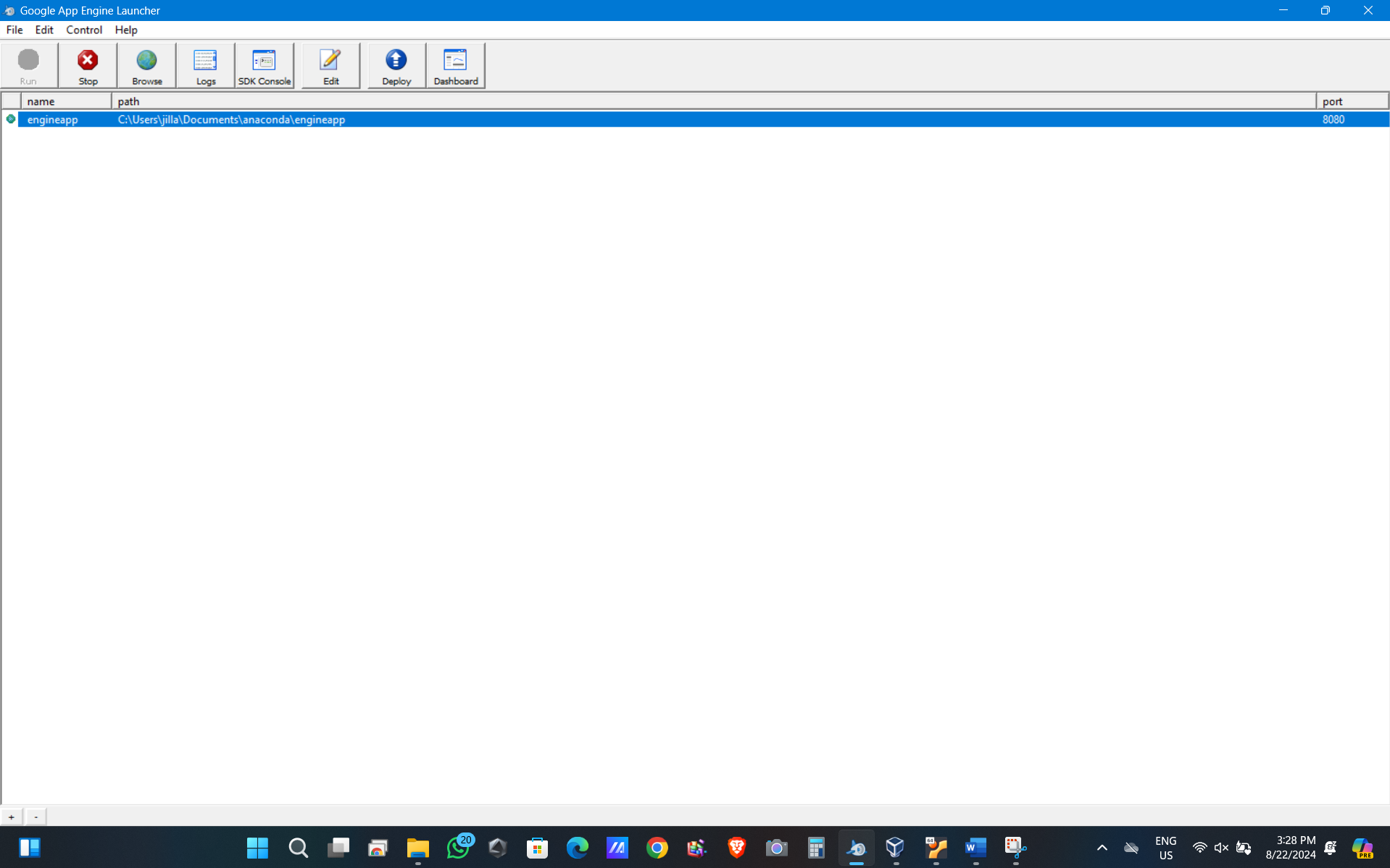
Step 1: Using GAE Launcher to Launch the Web Application.

Step 2: We create another new application and then click on browse to see our localhost id with the new port number.

Step 3: We can click on Logs to see the amount of time a person logged in.

When clicked on console we get the following tab.

The below is the number of times you can see of your log record.



Step 4. Open localhost/8080 on the browser

A screen shot of a computer

Description automatically generated

LAB 5

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

Step 1: Download JAVA

Step 2: Download CloudSim 3.0.3

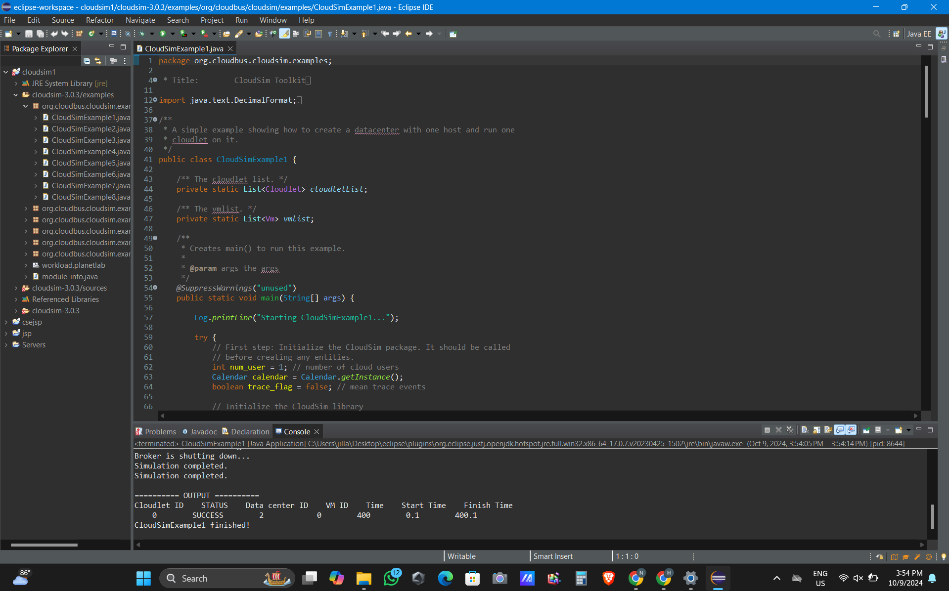
Step 3: Download common math 3

Step 4:Put the Common math jar file in the cloud sim folder

Step 5:Access the cloud sim folder using eclipse ide as a java project

Step 6:Select the Common math jar file while creating the java project in libraries.

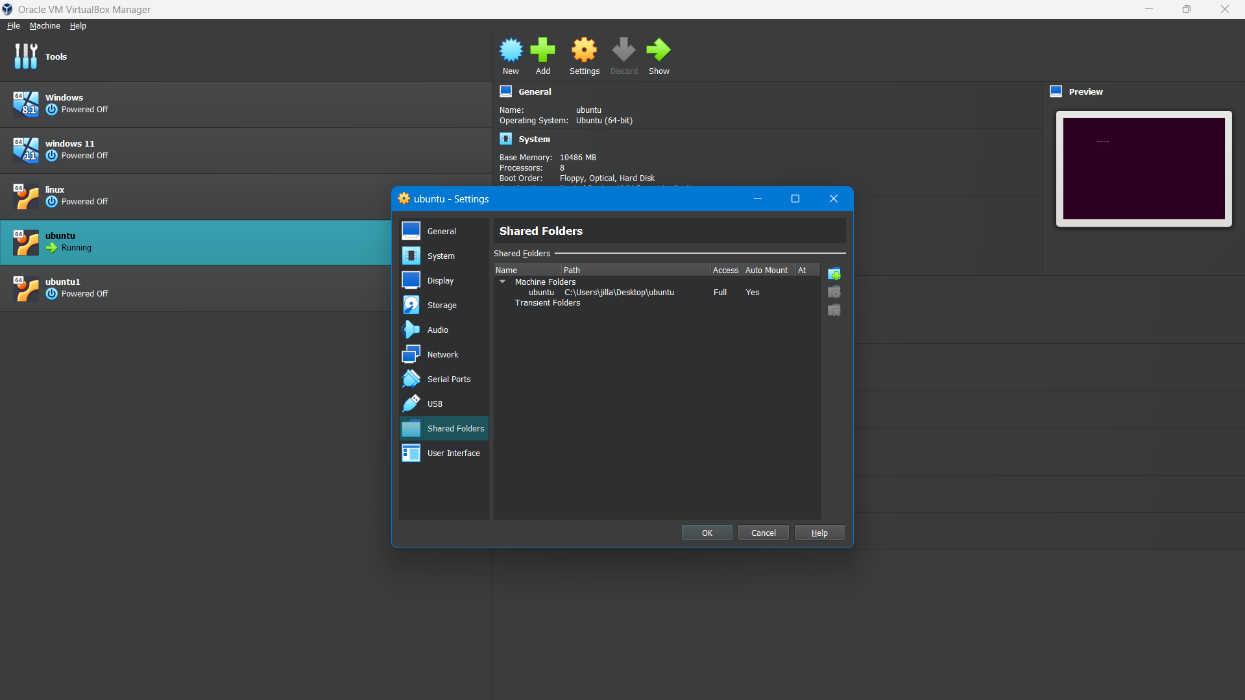
Step 7:Open it and run the example one program



LAB - 6

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

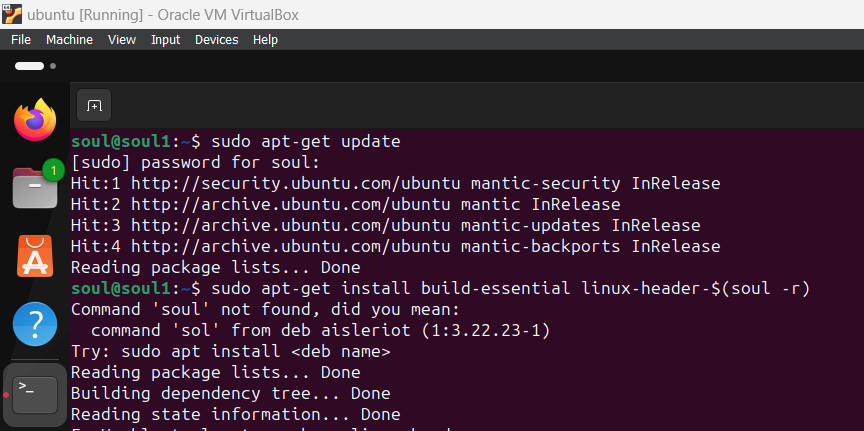
Step1: In the settings of VM, add add a shared folder the sits in host but is shared to VM (close the VM change settings and start)



Step 2: In linux open terminal (ctrl+alt+T)

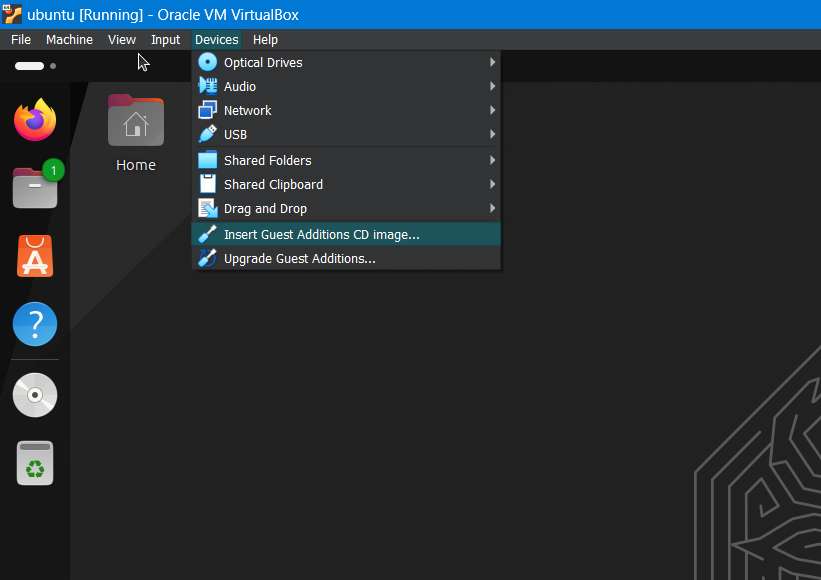
Commands:

sudo apt-get update sudo apt-get install build-esential linux-headers-$(uname -r)



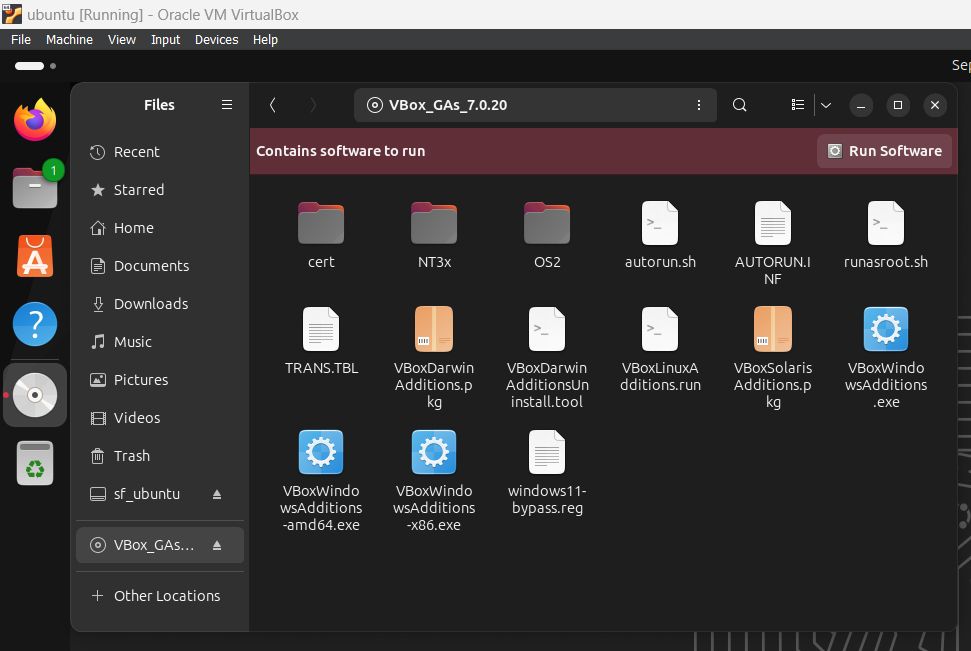
Step 3: Next go to Machine on top left of virtual box window: install guest additions

Step 4: Install Guest additions



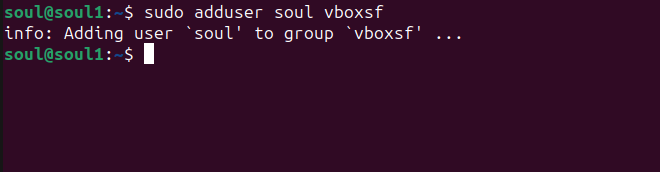
Step 6: after that on the left side of screen you see a pop up folder

Once you click on it you should see a run software button. Click on it

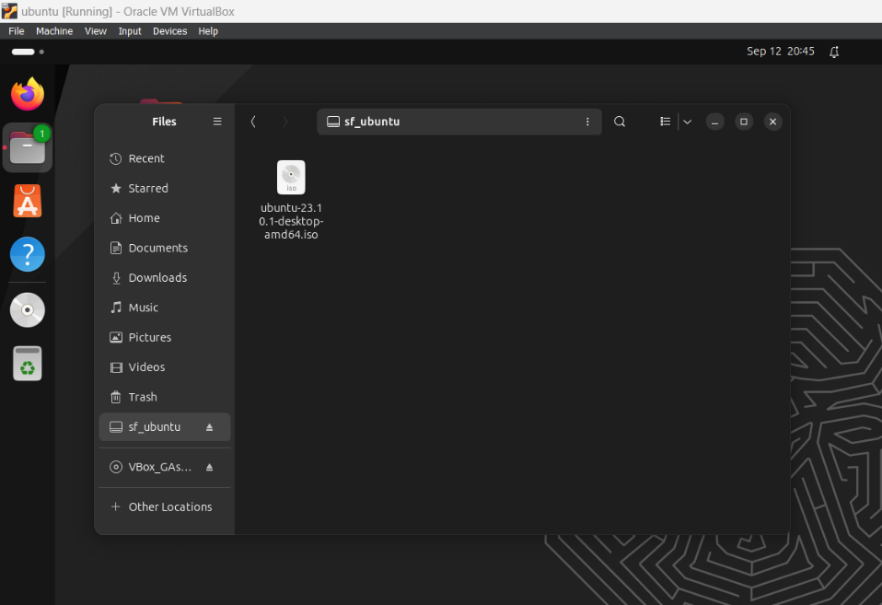


Step 5: sudo adduser [username] vboxsf

Username should be user linux username



Step 6: Restart VM: Now you should be able to see a shared folder on the file manager tab with read/write access



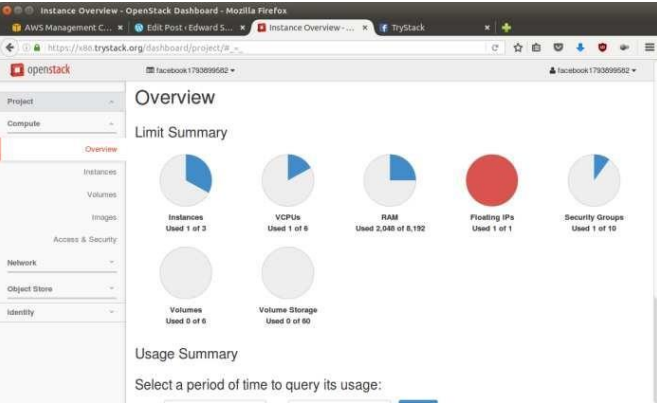
**LAB 7**

**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.



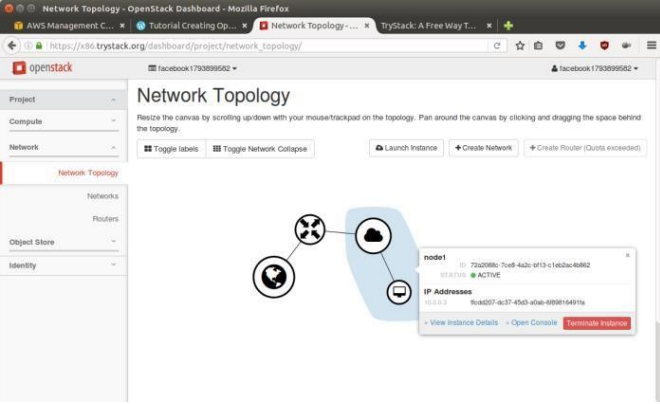
TryStack.org Homepage I assume that you already join to the Facebook Group and login to the dashboard. After you log in to the TryStack, you will see the Compute Dashboard like:



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,

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**

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,

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,

1. Fill Key Pair Name with your machine name (for example Edward-Key).

2. 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.

3. Click Import key pair.

3. 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:

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 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.

1. 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.

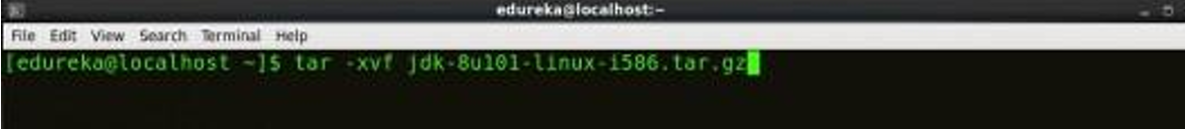
**Lab 8**

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

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

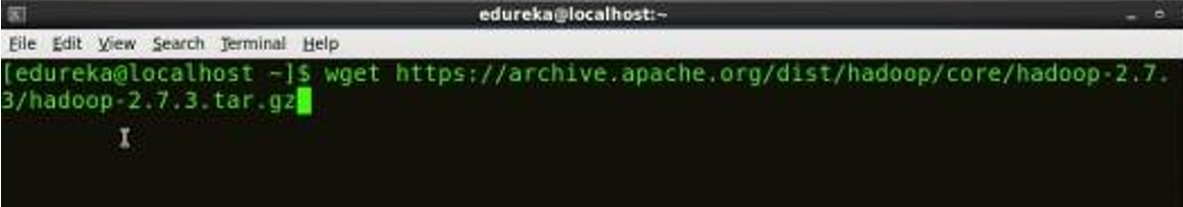
**Step 2:** Extract the Java Tar File.

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



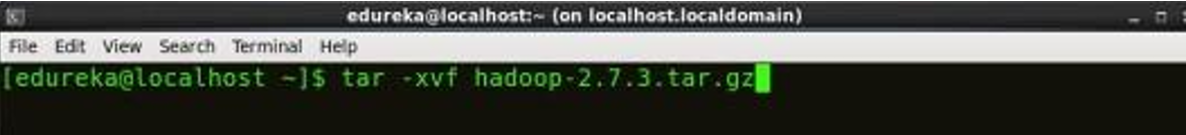
**Step 3:** Download the Hadoop 2.7.3 Package.

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

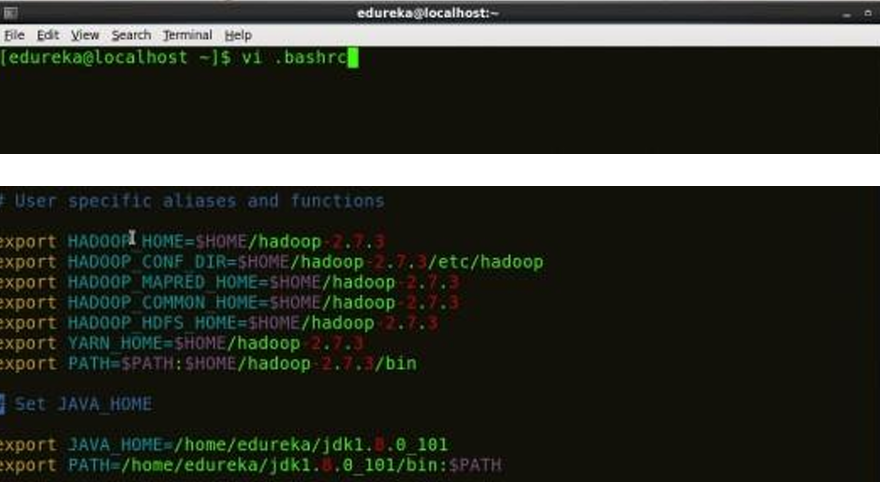


**Step 4:** Extract the Hadoop tar File.

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



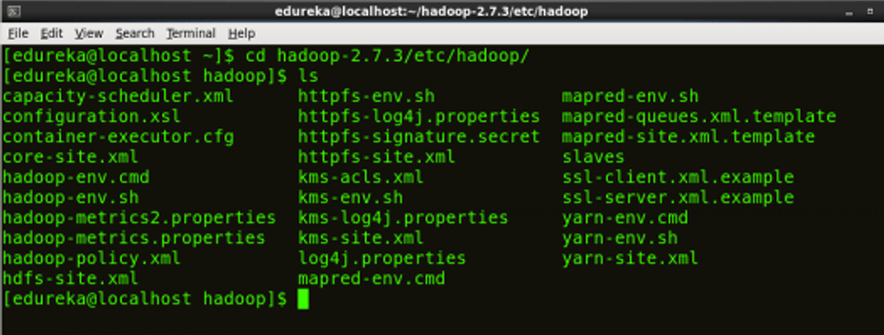
**Step5:** Add the Hadoop and Java paths in the bash file (.bashrc). Open. bashrc file. Now, add Hadoop and Java Path as shown below.



**Command:** vi .bashrc

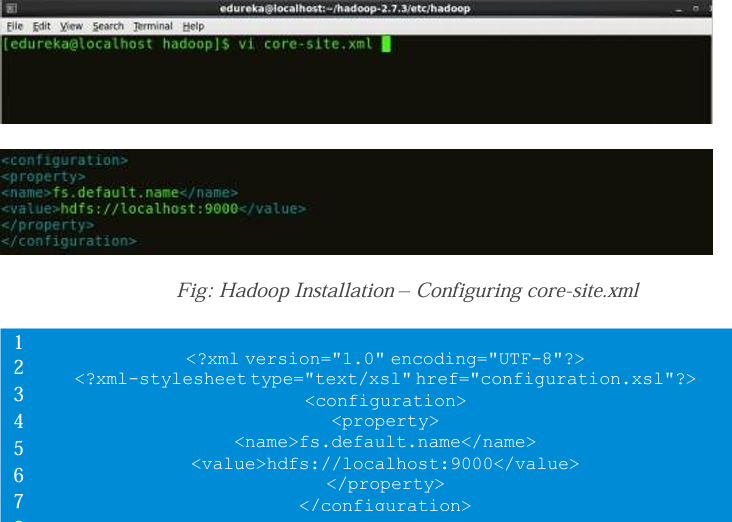
**Step 6:** Edit the Hadoop Configuration files.

**Command:** cd hadoop-2.7.3/etc/hadoop/



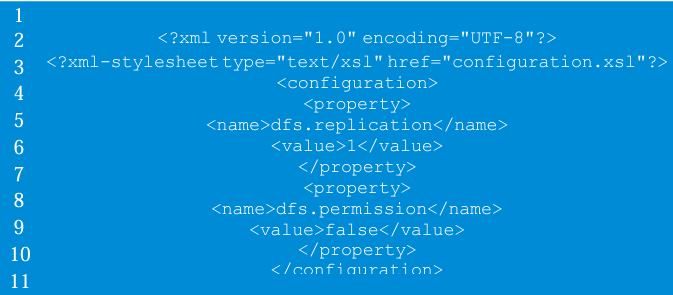
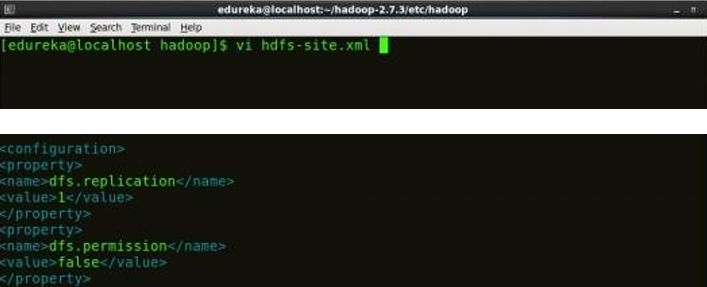
**Step 7:** Open core-site.xml and edit the property mentioned below inside configuration tag: core-site.xml informs Hadoop daemon where NameNode runs in the cluster. It contains configuration settings of Hadoop core such as I/O settings that are common to HDFS & MapReduce.

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



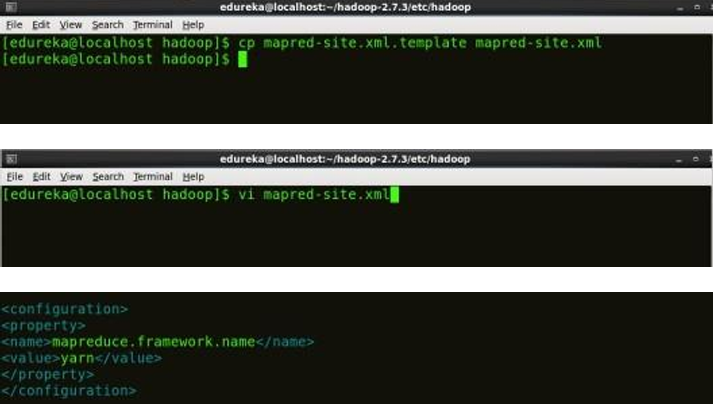
**Step 8:** Edit hdfs-site.xml and edit the property mentioned below inside configuration tag: hdfs-site.xml contains configuration settings of HDFS daemons (i.e. NameNode, DataNode, Secondary NameNode). It also includes the replication factor and block size of HDFS.

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

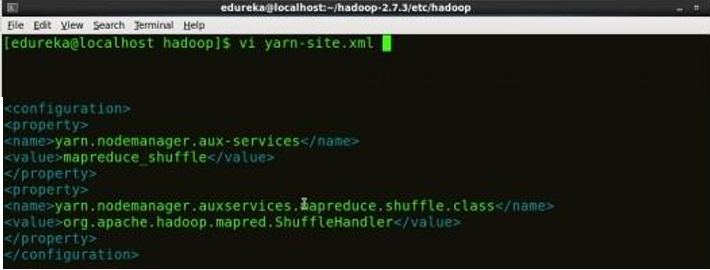


**Step 9:** Edit the mapred-site.xml file and edit the property mentioned below inside configuration tag: mapred-site.xml contains configuration settings of MapReduce application like number of JVM that can run in parallel, the size of the mapper and the reducer process, CPU cores available for a process,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



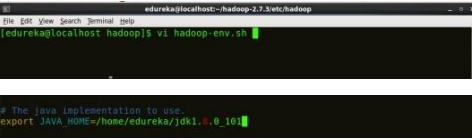
**Step 10:** Edit yarn-site.xml and edit the property mentioned below inside configuration tag: yarn-site.xml contains configuration settings of ResourceManager and NodeManager like application memory management size, the operation needed on program & algorithm, etc.

**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

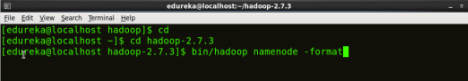


**12:** Go to Hadoop home directory and format the NameNode.

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

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

**Command:** cd hadoop-2.7.3/sbin

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

**Command:** ./start-all.sh

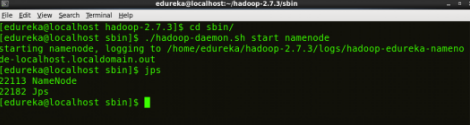
The above command is a combination of start-dfs.sh, start-yarn.sh & mr-jobhistorydaemon.sh

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.

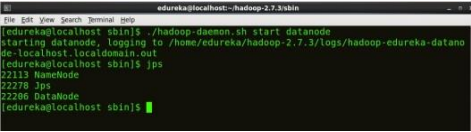
**Command**: ./hadoop-daemon.sh start namenode



**Start DataNode:**

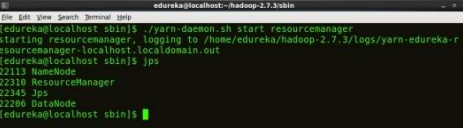
On startup, a DataNode connects to the Namenode and it responds to the requests from the Namenode for different operations.

**Command:** ./hadoop-daemon.sh start datanode



**Start ResourceManager:**

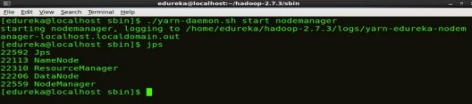
ResourceManager is the master that arbitrates all the available cluster resources and thus helps in managing the distributed applications running on the YARN system. Its work is to manage each NodeManagers and the each application’s ApplicationMaster.



**Command:** ./yarn-daemon.sh start resourcemanager

**Start NodeManager:**

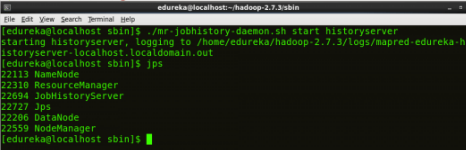
The NodeManager in each machine framework is the agent which is responsible for managing containers, monitoring their resource usage and reporting the same to the ResourceManager.

****

**Command**: ./yarn-daemon.sh start nodemanager

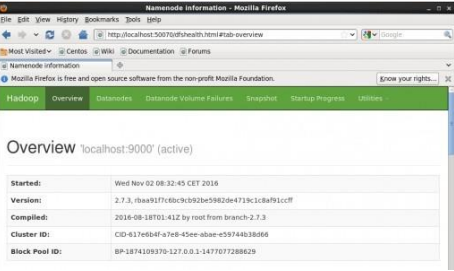
**Start JobHistoryServer:**

JobHistoryServer is responsible for servicing all job history related requests from client. **Command:** ./mr-jobhistory-daemon.sh start historyserver



**Command:** jps

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



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