



JEPPIAAR INSTITUTE OF TECHNOLOGY

(An Autonomous Institution)

Self-Belief | Self Discipline | Self Respect

Kunnam, Sunguvarchatram, Sriperumbudur-631604



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING



CSS335–CLOUD COMPUTING LABORATORY

MANUAL



JEPIAAR INSTITUTE OF TECHNOLOGY

(An Autonomous Institution)

Self-Belief | Self Discipline | Self Respect

Kunnam, Sunguvarchatram, Sriperumbudur-631604



INSTITUTE VISION:

Jeppiaar Institute of Technology aspires to provide technical education in futuristic technologies with the perspective of innovative, industrial and social application for the betterment of humanity.

INSTITUTE MISSION:

M1: To produce competent and disciplined high-quality professionals with the practical skills necessary to excel as innovative professionals and entrepreneurs for the benefit of the society.

M2: To improve the quality of education through excellence in teaching and learning, research, leadership and by promoting the principles of scientific analysis, and creative thinking.

M3: To provide excellent infrastructure, serene and stimulating environment that is most conducive to learning.

M4: To strive for productive partnership between the Industry and the Institute for research and development in the emerging fields and creating opportunities for employability.

M5: To serve the global community by instilling ethics, values and life skills among the students needed to enrich their lives.

Department Vision

To impart futuristic technological education, innovation and collaborative research in the field of Computer Science Engineering and develop Quality Professional for the improvement of the society and industry.

Department Mission

M1: To develop the students as professionally competent and disciplined engineers for the benefit of the development of the country.

M2: To produce excellent infrastructure to adopt latest technologies, industry-institute interaction and encouraging research activities.

M3: To provide multidisciplinary technical skills to pursue research activities, higher studies, entrepreneurship and perpetual learning.

M4: To enrich students with professional integrity and ethical standards to handle social challenges successfully in their life.

PEO's OF THE DEPARTMENT

PEO 1: To support students with substantial knowledge for developing and resolving mathematical, scientific and engineering problems.

PEO 2: To provide students with adequate training and opportunities to work as a collaborator with informative and administrative qualities.

PEO 3: To motivate students for extensive learning to prepare them for graduate studies, R&D and competitive exams.

PEO 4: To cater students with industrial exposure in an endeavour to succeed in the emerging cutting-edge technologies.

PEO 5: To shape students with principled values and to follow the code of ethics in social and professional life.

PROGRAM OUTCOMES (POs)

Engineering Graduates will be able to:

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.

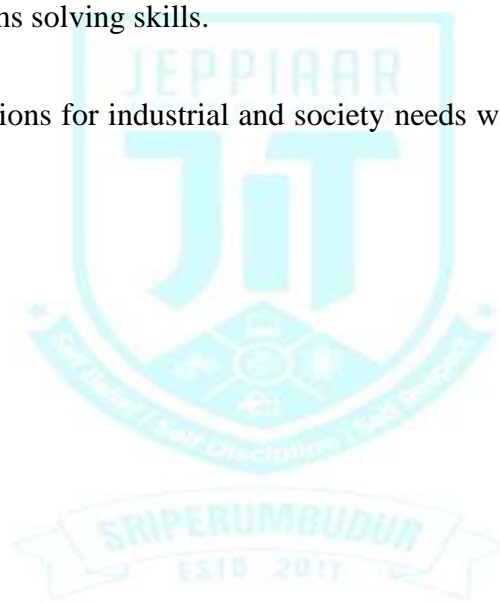
PO12: Life-long learning: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES

PSO1: Analyze, design, and implement quality software by applying fundamental and programming concepts of Computer Science and Engineering.

PSO2: Design and develop solutions for scientific, business and real time applications through analytical, logical and problems solving skills.

PSO3: Provide efficient solutions for industrial and society needs with acquired knowledge through emerging technical skills.



GENERAL LABORATORY INSTRUCTIONS

1. Students are advised to come to the laboratory at least 5 minutes before (to the starting time), those who come after 5 minutes will not be allowed into the lab.
2. Plan your task properly much before to the commencement, come prepared to the lab with the synopsis / program / experiment details.
3. Student should enter into the laboratory with:
 - a. Laboratory observation notes with all the details (Problem statement, Aim, Algorithm, Procedure, Program, Expected Output, etc.,) filled in for the lab session.
 - b. Laboratory Record updated up to the last session experiments and other utensils (if any) needed in the lab.
 - c. Proper Dress code and Identity card.
4. Sign in the laboratory login register, write the TIME-IN, and occupy the computer system allotted to you by the faculty.
5. Execute your task in the laboratory, and record the results / output in the lab observation note book, and get certified by the concerned faculty.
6. All the students should be polite and cooperative with the laboratory staff, must maintain the discipline and decency in the laboratory.
7. Computer labs are established with sophisticated and high end branded systems, which should be utilized properly.
8. Students / Faculty must keep their mobile phones in SWITCHED OFF mode during the lab sessions. Misuse of the equipment, misbehaviors with the staff and systems etc., will attract severe punishment.
9. Students must take the permission of the faculty in case of any urgency to go out; if anybody found loitering outside the lab / class without permission during working hours will be treated seriously and punished appropriately.
10. Students should LOG OFF/ SHUT DOWN the computer system before he/she leaves the lab after completing the task (experiment) in all aspects. He/she must ensure the system / seat is kept properly.

Head of the Department

Principal

PRACTICAL EXERCISES

1. Install Virtualbox/VMware/ Equivalent open source cloud Workstation with different flavors 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 wordcount.
8. Creating and Executing Your First Container Using Docker.
9. Run a Container from Docker Hub
10. Find procedure to run the virtual machine of different configuration. Check how many virtual machines can be utilized at particular time.
11. Mount the One Node Hadoop Cluster Using FUSE
12. Use the API's of Hadoop for Interaction with Hadoop

COURSE OUTCOME:

Course Outcome No.	Course Outcome	Highest Cognitive Level
201.1	Understand the design challenges in the cloud.	C1
201.2	Apply the concept of virtualization and its types.	C2
201.3	Experiment with virtualization of hardware resources and Docker.	C3
201.4	Develop and deploy services on the cloud and set up a cloud environment.	C4
201.5	Explain security challenges in the cloud environment.	C5

CO & PO AND PSO 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

EX.NO:1 INSTALL VIRTUALBOX/VMWARE WORKSTATION WITH DIFFERENT FLAVOURS OF LINUX OR WINDOWS OS ON TOP OF WINDOWS7 OR 8.

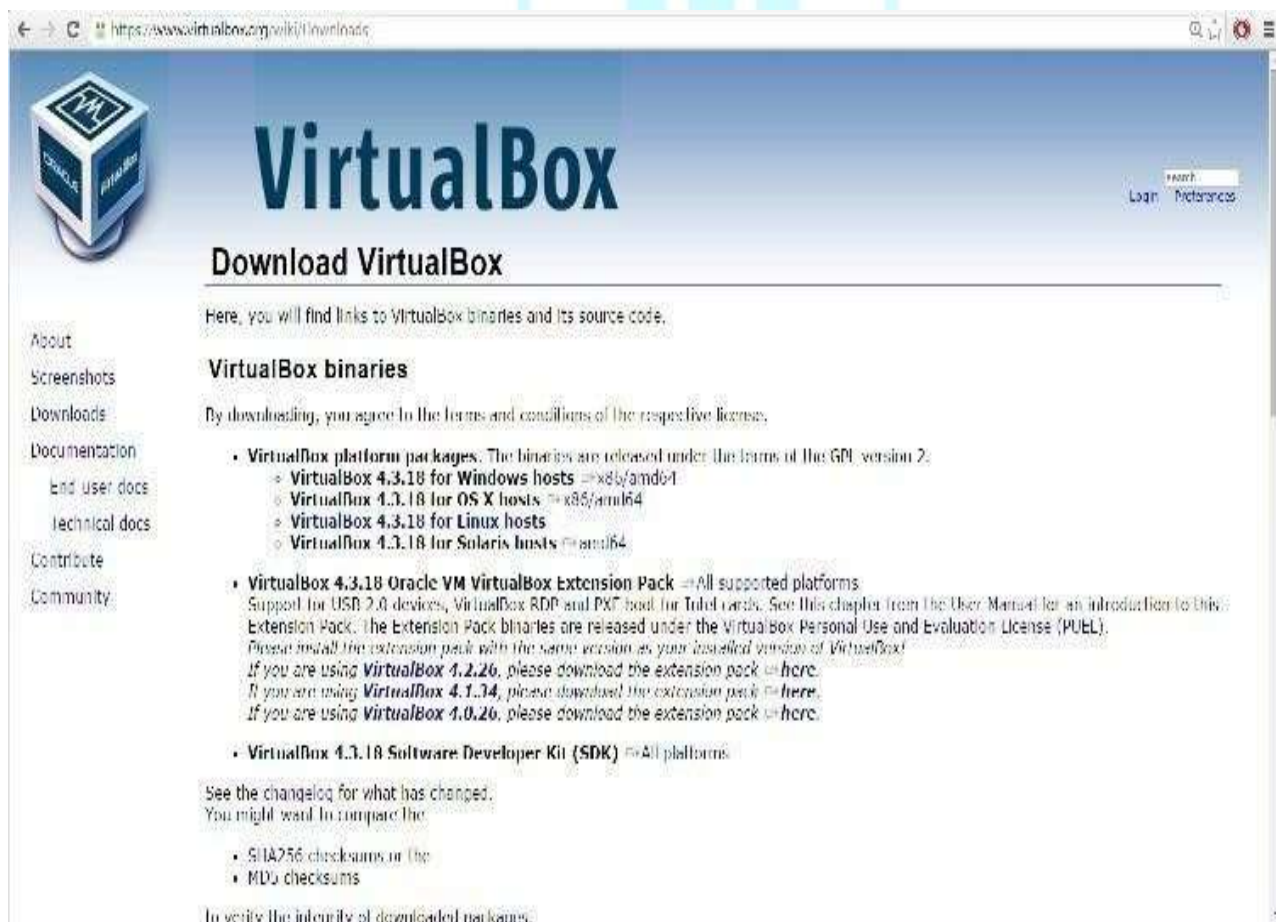
Date:

AIM:

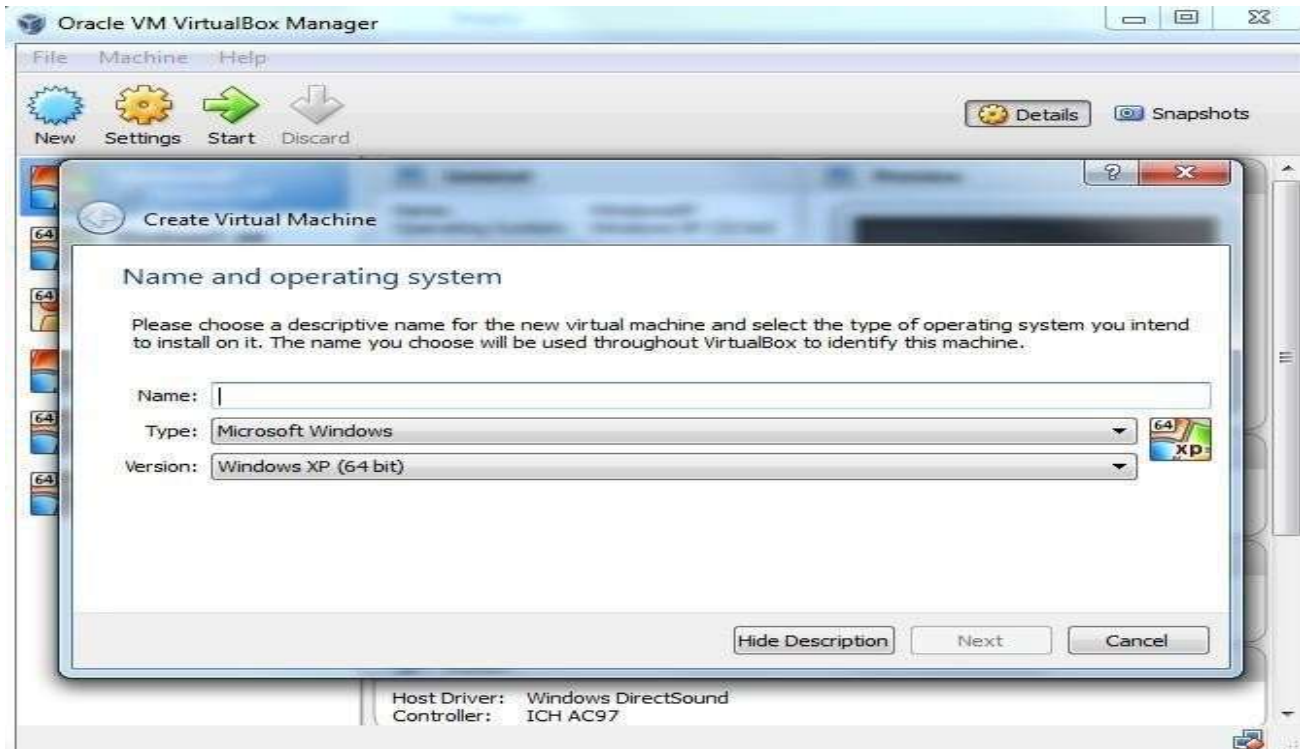
To Install Virtualbox/VMware Workstation with different flavors of linux or windows OS on top of windows7 or 8.

PROCEDURE

STEP 1: Go to VirtualBox website [here](https://www.virtualbox.org/wiki/Downloads) to download the binary for your current operating system. Since our host machine is running on Windows, I'll choose 'x86/amd64' from Windows hosts. When download is finished, run the executable file. Continue with the installation of VirtualBox with the defaults. This will open VirtualBox at the end of the installation.

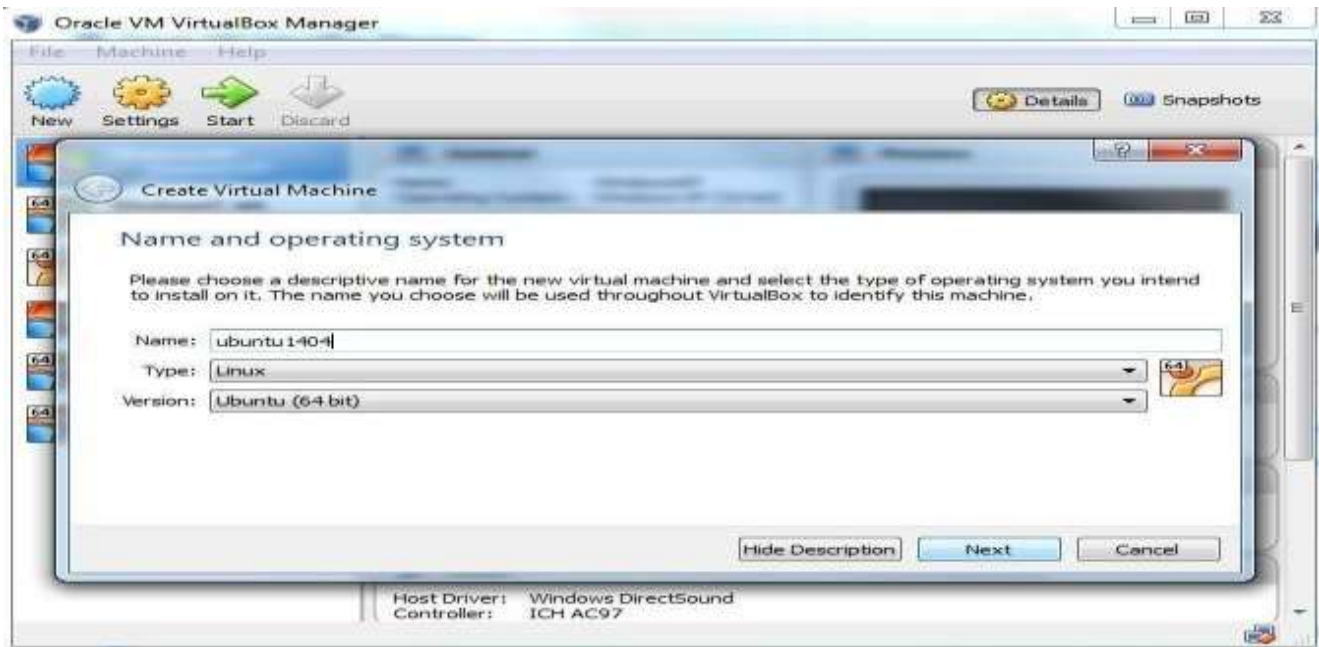


CREATE VIRTUAL MACHINE

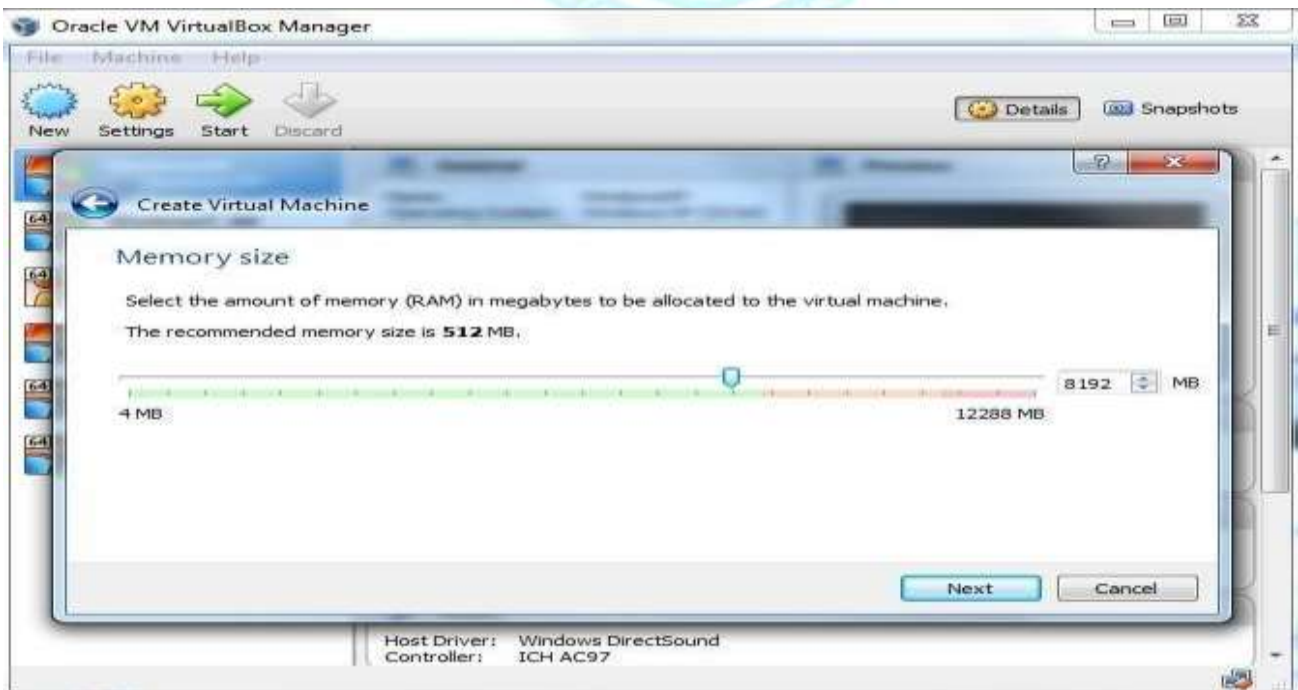


STEP 2: Click 'New' button to open a dialog.

STEP 3: Type a name for the new virtual machine. Since I am planning to install Ubuntu 14.04, I'll enter 'ubuntu1404'. Note that Virtual Box automatically changes 'Type' to Linux and 'Version' to 'Ubuntu (64 bit)'. These two options are exactly what we need.

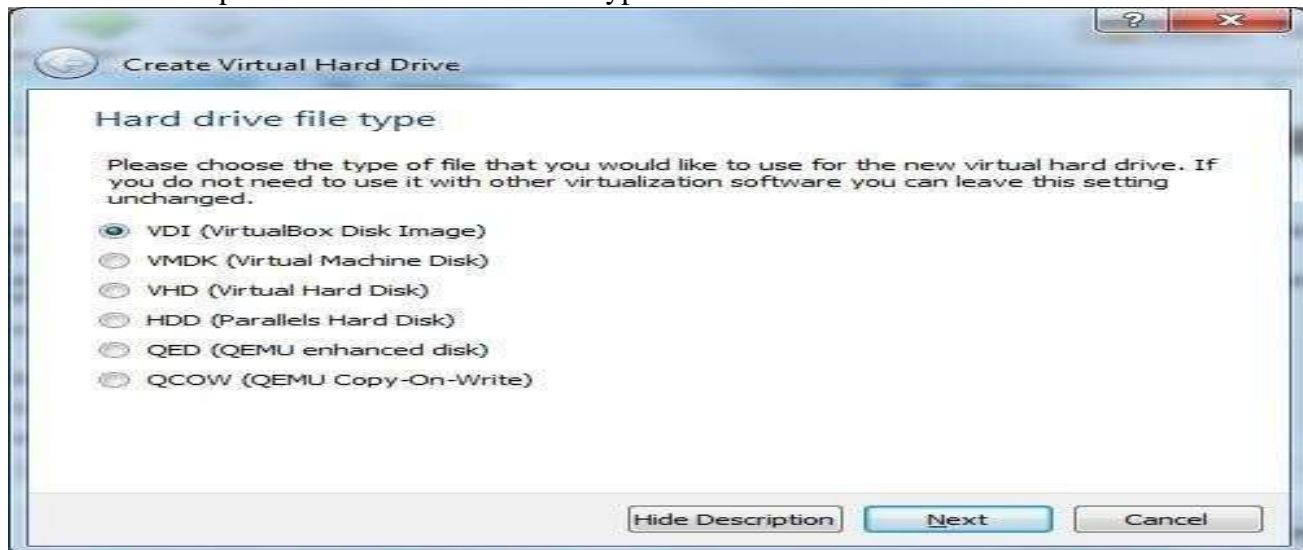


STEP 4: The memory size depends on your host machine memory size. In this case, 12GB physical RAM. I like to allocate as much as possible for Ubuntu but leave some for my Windows host machine. I pick 8192 MB for my Ubuntu. Note that Virtual Box will create a [swap](#) partition with the same amount space as base memory you have entered here. So later when you are selecting the size of the virtual hard drive, make sure it is large enough since the [hard drive](#) will be splitted into root (/) and swap partitions. The root partition contains by default all your system files, program settings and documents.



Accept the default 'Create a virtual hard drive now' and click 'Create' button.

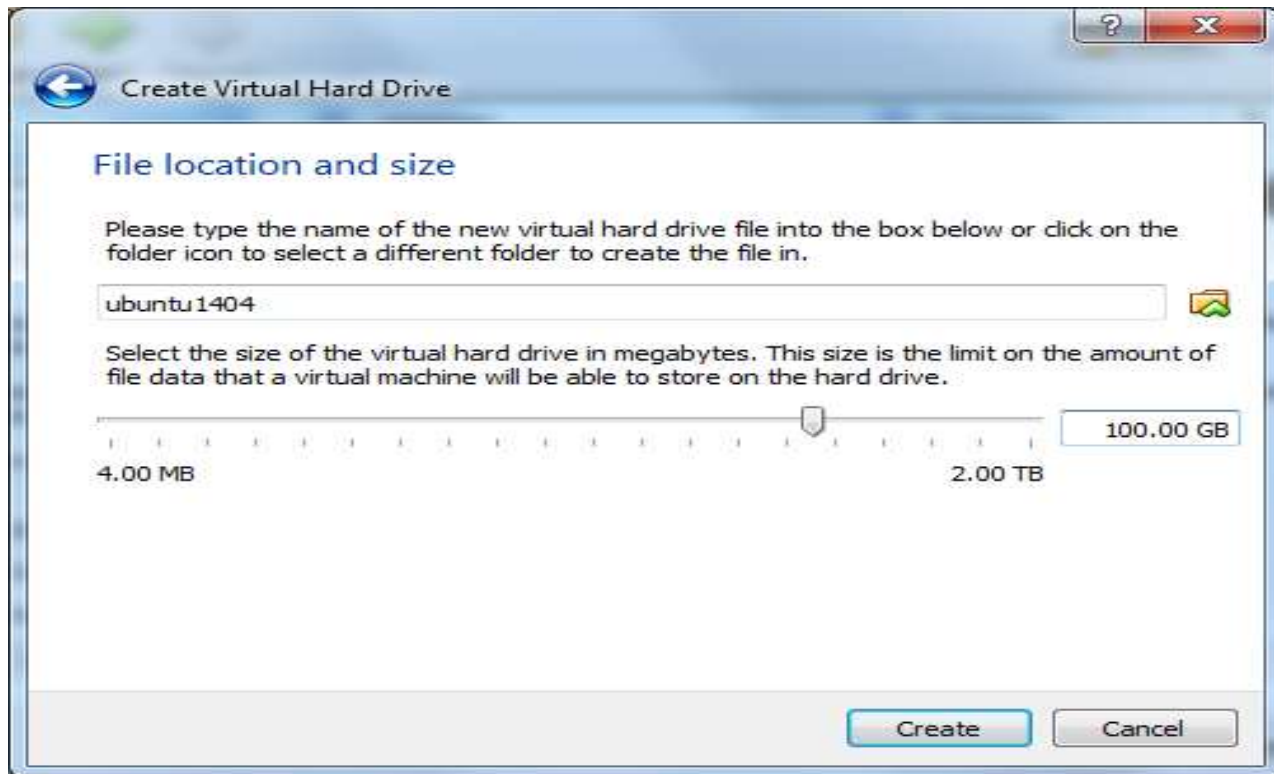
Continue to accept the default 'VDI' drive file type and click 'Next' button.



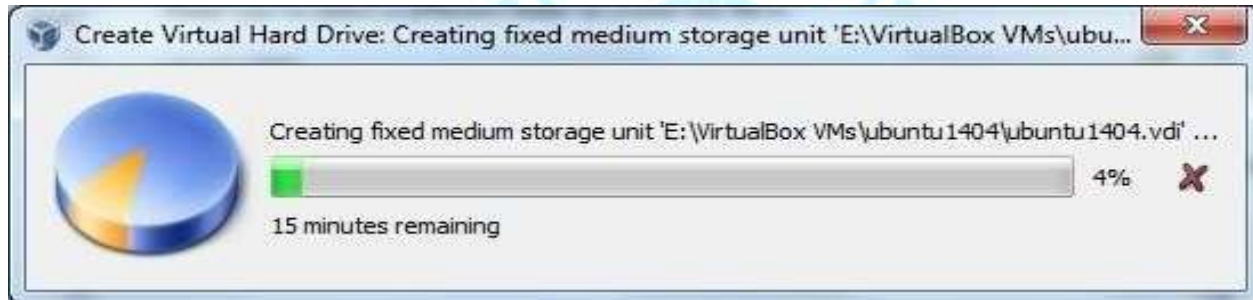
Change the storage type from the default 'Dynamically allocated' to 'Fixed size' to increase performance.



For the virtual hard drive space, the default value is 8GB which is too little for RNA-Seq analysis. I'll pick 100GB since I have plenty of space in my hard disk. You want to choose a good size for your RNA-Seq analysis. If you realize the drive space is not large enough, you'll need to go over these steps again to create another virtual machine.



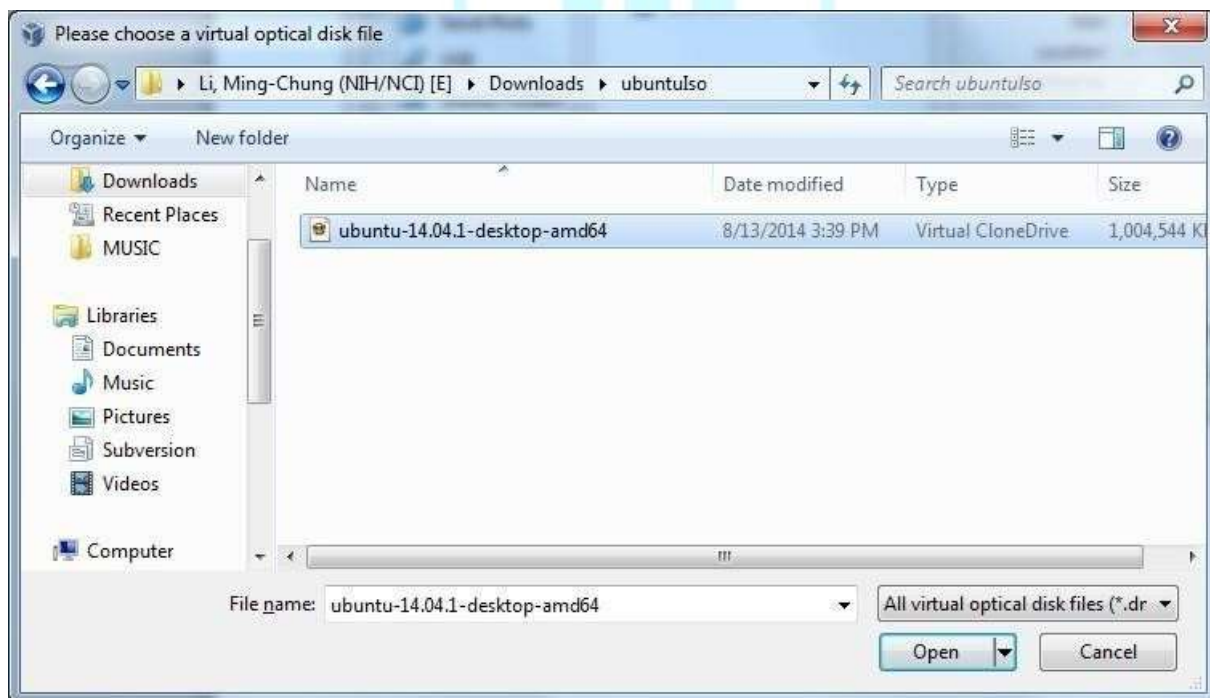
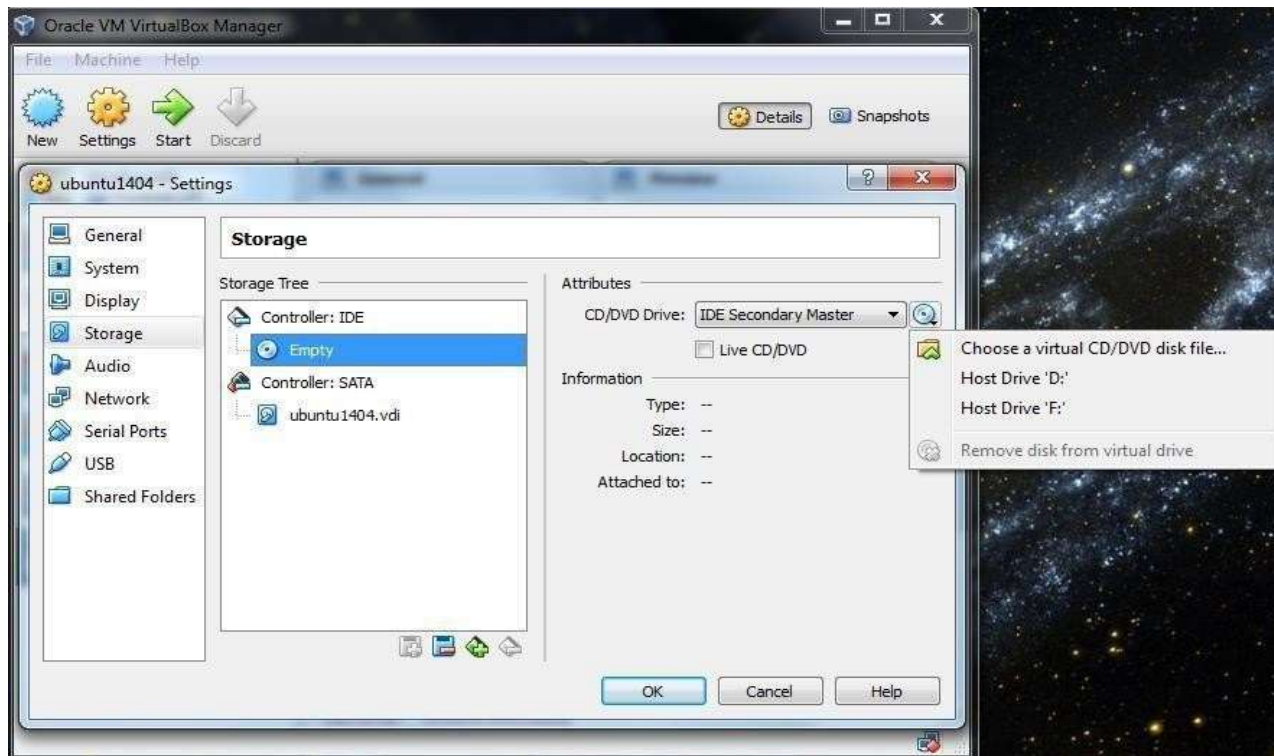
Click 'Create' button and Virtual Box will generate Ubuntu virtual machine.



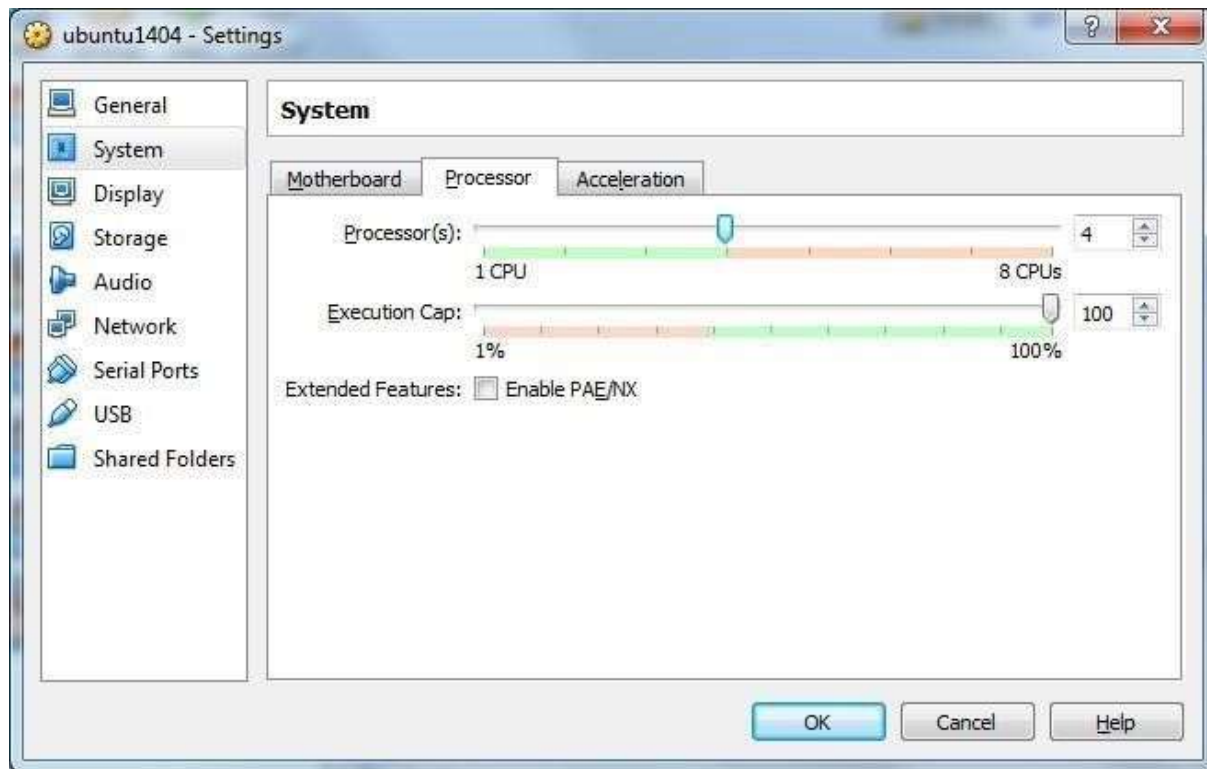
Now the virtual machine is created. We are ready to install Ubuntu in this virtual machine. Select your new virtual machine and click 'Settings' button. Click on 'Storage' category and then 'Empty' under Controller: IDE. Click "CD/DVD" icon on right hand side and select the Ubuntu ISO file to mount.

Note that if you have not downloaded 64-bit Ubuntu ISO file,

When downloading Ubuntu ISO file, make sure to select 64-bit version. Also make sure the **VT-x/Virtualization Technology** has been enabled in your computer's [BIOS/Basic Input Output System](#).



Since Top hat program can take an advantage of multiple processors/threads, it is a good idea to specify a large number of processors in virtual machine (default value is 1). You can change this number by clicking on 'System' category. In this case, I change the number of CPUs to 4 since 4 is the largest value shown on the green bar in my case. Now you can click 'OK' button to continue.



Virtual Box may pop up a message about 'Auto capture keyboard' option. Read the message there and check 'Do not show this message again' option before clicking OK.



INSTALL UBUNTU

Back to Oracle VM Virtual Box Manager, click on the new Ubuntu virtual machine and hit 'Start' button. Now you shall see a 'Welcome' screen. Click 'Install Ubuntu' button. Note that the installation process may differ a little bit from version to version. The screenshots here are based on Ubuntu 14.04.1.

Click 'Continue' button.

Make sure 'Erase disk and install Ubuntu' option is selected and click 'Install Now' button.

Ubuntu will ask you a few questions. If the default is good, click 'Continue' button.



will have root/sudo privilege. Click 'Continue' button.

The installation will continue until it is finished.

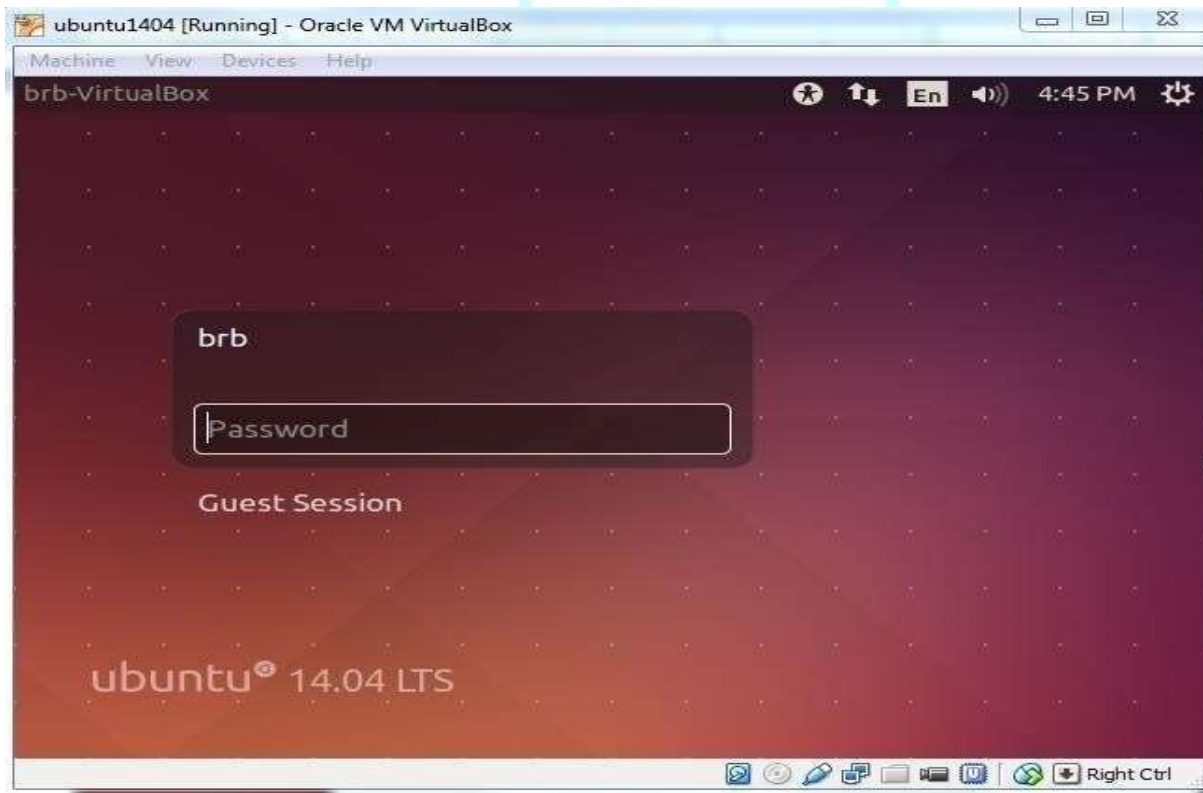
After installation is complete, click 'Restart Now' button. When you see a screen with a black background saying 'Please remove installation media and close the tray (if any) then press

ENTER:', just follow it.

Enter the password you have chosen and press 'Enter'.

The Ubuntu Desktop OS is ready.

OUTPUT:



In 'Who are you?' dialog, enter your preferred name, username and password. *Note that this user*

RESULT:

Thus the given program has been executed and verified successfully.

EX.NO 2: INSTALL A C COMPILER IN THE VIRTUAL MACHINE AND EXECUTE A SAMPLE PROGRAM

DATE:

AIM: To Install a C compiler in the virtual machine and execute a sample C program.

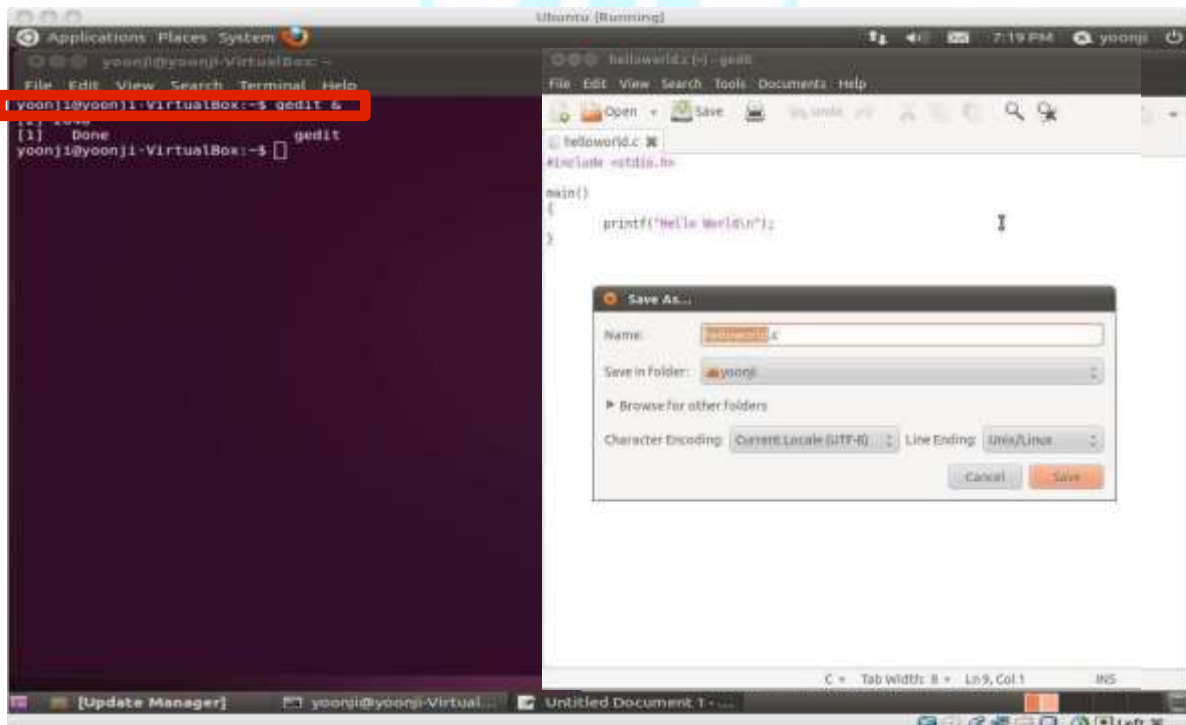
PROCEDURE:

Step 1: Open Terminal (Applications-Accessories-Terminal)

Step 2: Open **gedit** by typing “gedit &” on terminal
(You can also use any other Text Editor application)

Step 3: Type the following on gedit (or any other text editor)

```
#include<stdio.h>
main()
{
printf ("Hello World\n");
}
```



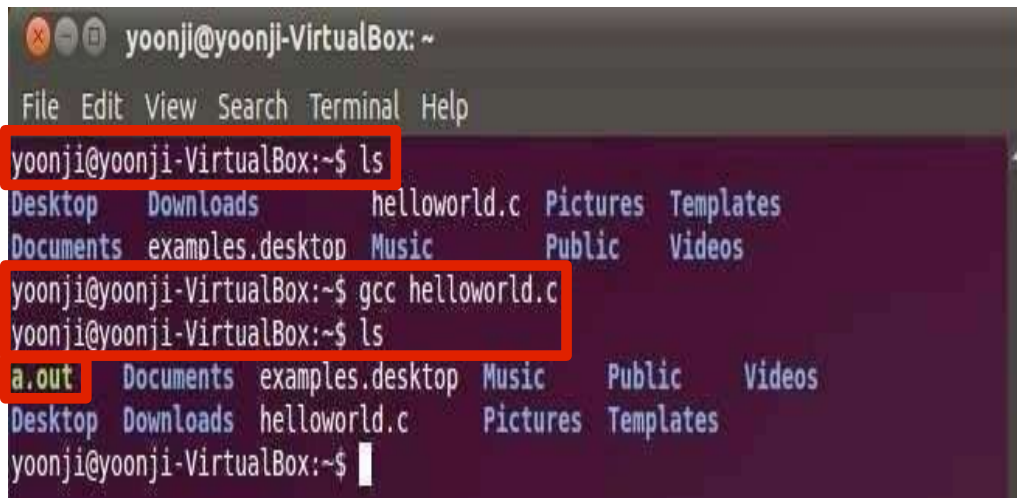
Save this file as “helloworld.c”

Step 4: Type “ls” on Terminal to see all files under current folder

Confirm that “helloworld.c” is in the current directory. If not, type cd DIRECTORY_PATH to

go to the directory that has “helloworld.c”

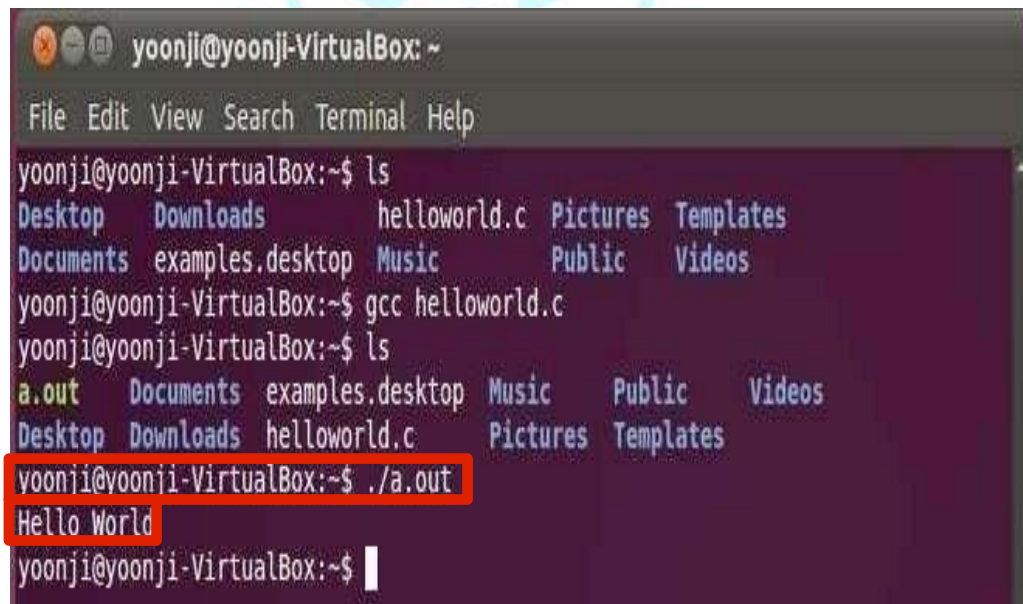
Step 5: Type “gcc helloworld.c” to compile, and type “ls” to confirm that a new executable file “a.out” is created

A terminal window titled 'yoonji@yoonji-VirtualBox: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the following commands and output: 'ls' lists files including 'helloworld.c'; 'gcc helloworld.c' compiles the file; a second 'ls' shows 'a.out' has been created. The file 'a.out' is highlighted in green in the second 'ls' output.

```
yoonji@yoonji-VirtualBox: ~  
File Edit View Search Terminal Help  
yoonji@yoonji-VirtualBox:~$ ls  
Desktop  Downloads      helloworld.c  Pictures  Templates  
Documents examples.desktop Music          Public    Videos  
yoonji@yoonji-VirtualBox:~$ gcc helloworld.c  
yoonji@yoonji-VirtualBox:~$ ls  
a.out Documents examples.desktop Music Public Videos  
Desktop Downloads helloworld.c Pictures Templates  
yoonji@yoonji-VirtualBox:~$
```

Step 6: Type “./a.out” on Terminal to run the program

OUTPUT:

A terminal window titled 'yoonji@yoonji-VirtualBox: ~' with a menu bar (File, Edit, View, Search, Terminal, Help). The terminal shows the same sequence of commands as the previous screenshot, followed by './a.out' which outputs 'Hello World'. The command './a.out' and its output 'Hello World' are highlighted with red boxes.

```
yoonji@yoonji-VirtualBox: ~  
File Edit View Search Terminal Help  
yoonji@yoonji-VirtualBox:~$ ls  
Desktop  Downloads      helloworld.c  Pictures  Templates  
Documents examples.desktop Music          Public    Videos  
yoonji@yoonji-VirtualBox:~$ gcc helloworld.c  
yoonji@yoonji-VirtualBox:~$ ls  
a.out Documents examples.desktop Music Public Videos  
Desktop Downloads helloworld.c Pictures Templates  
yoonji@yoonji-VirtualBox:~$ ./a.out  
Hello World  
yoonji@yoonji-VirtualBox:~$
```

RESULT:

Thus the given program has been executed and verified successfully.

EX. NO 3 INSTALL GOOGLE APP ENGINE CREATE HELLO WORLD APP AND OTHER SIMPLE WEB APPLICATIONS USING PYTHON/JAVA.

DATE:

AIM: To create hello world app and other simple web applications using python/java.

PROCEDURE:

STEP 1: PRE-REQUISITES: PYTHON 2.5.4

The App Engine SDK allows you to run Google App Engine Applications on your local computer. It simulates the run-time environment of the Google App Engine infrastructure.

If you don't already have Python 2.5.4 installed in your computer, download and Install Python 2.5.4 from:

<http://www.python.org/download/releases/2.5.4/>

STEP 2: DOWNLOAD AND INSTALL

You can download the Google App Engine SDK by going to:

<http://code.google.com/appengine/downloads.html>

and download the appropriate install package.

Download the Google App Engine SDK

Before downloading, please read the [Terms](#) that govern your use of the App Engine SDK.

Please note: The App Engine SDK is under **active development**, please keep this in mind as you explore its capabilities. See the [SDK Release Notes](#) for the information on the most recent changes to the App Engine SDK. If you discover any issues, please feel free to notify us via our [Issue Tracker](#).

Platform	Version	Package	Size	SHA1 Checksum
Windows	1.1.5 - 10/03/08	GoogleAppEngine_1.1.5.msi	2.5 MB	e974312b4aefc0b3873ff0d93eb4c525d5e88c30
Mac OS X	1.1.5 - 10/03/08	GoogleAppEngineLauncher-1.1.5.dmg	3.6 MB	f62208ac01c1b3e39796e58100d5f1b2f052d3e7
Linux/Other Platforms	1.1.5 - 10/03/08	google_appengine_1.1.5.zip	2.6 MB	cbb9ce817bdabf1c4f181d9544864e55ee253de1

Download the Windows installer – the simplest thing is to download it to your Desktop or another folder that you remember.



Double Click on the **GoogleAppEngine** installer.



Click through the installation wizard, and it should install the App Engine. If you do not have Python2.5, it will install Python 2.5 as well.

Once the install is complete you can discard the downloaded installer



STEP 3: 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. Make the Folder on 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: pythonapi_version: 1
```

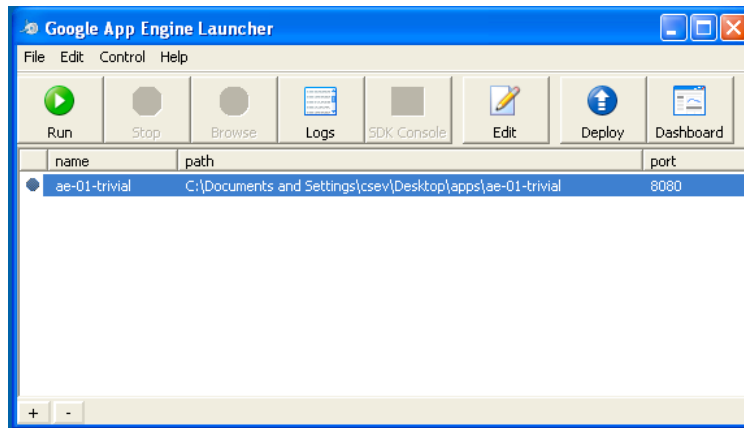
```
handlers:
- url: /*
  script: index.py
```

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 'Hello there Chuck'
```

Then start the **GoogleAppEngineLauncher** 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.

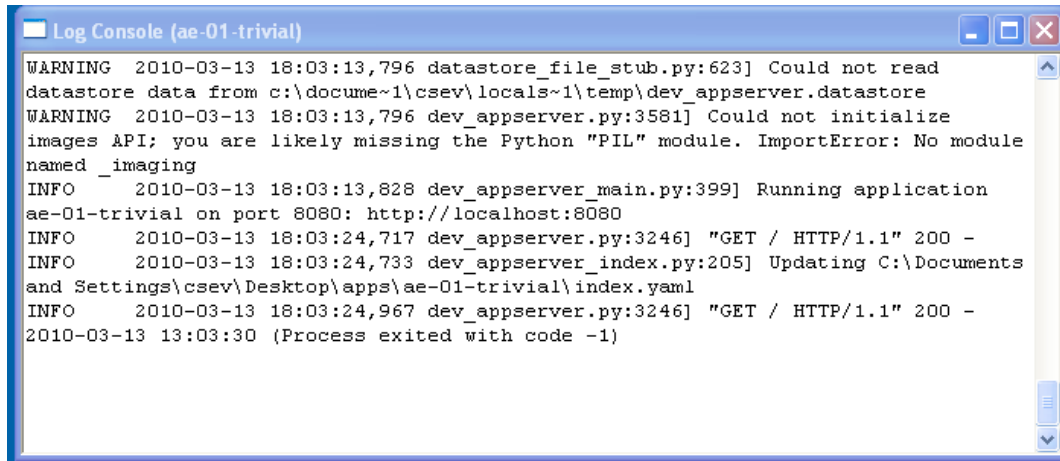


Once you have selected your application and press **Run**. After a few moments your application will start and the launcher will show a little green icon next to your application. Then press **Browse** to open a browser pointing at your application which is running at **http://localhost:8080/**

STEP 4 : Paste **http://localhost:8080** into your browser and you should see your application as follows: Just for fun, edit the **index.py** to change the name “Chuck” to your own name 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:



```
Log Console (ae-01-trivial)
WARNING 2010-03-13 18:03:13,796 datastore_file_stub.py:623] Could not read
datastore data from c:\docume~1\csev\locals~1\temp\dev_appserver.datastore
WARNING 2010-03-13 18:03:13,796 dev_appserver.py:3581] Could not initialize
images API; you are likely missing the Python "PIL" module. ImportError: No module
named _imaging
INFO 2010-03-13 18:03:13,828 dev_appserver_main.py:399] Running application
ae-01-trivial on port 8080: http://localhost:8080
INFO 2010-03-13 18:03:24,717 dev_appserver.py:3246] "GET / HTTP/1.1" 200 -
INFO 2010-03-13 18:03:24,733 dev_appserver_index.py:205] Updating C:\Documents
and Settings\csev\Desktop\apps\ae-01-trivial\index.yaml
INFO 2010-03-13 18:03:24,967 dev_appserver.py:3246] "GET / HTTP/1.1" 200 -
2010-03-13 13:03:30 (Process exited with code -1)
```

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

Shutting Down the Server

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

OUTPUT:



RESULT:

Thus the given program has been executed and verified successfully.

EX. NO. 4: USE GAE LAUNCHER TO LAUNCH THE WEB APPLICATIONS**DATE:**

AIM: To launch GAE launcher to launch the web applications.

PROCEDURE:

Deploying the app to App Engine

To upload the guest book app, run the following command from within the appending-guestbook-python directory of your application where the app.yaml and index.yaml files are located:

gcloud app deploy app.yaml index.yaml

Optional flags:

Include the --project flag to specify an alternate Cloud Console project ID to what you initialized a as the default in the gcloud tool.

Example: --project [YOUR_PROJECT_ID]

Include the -v flag to specify a version ID, otherwise one is generated for you.

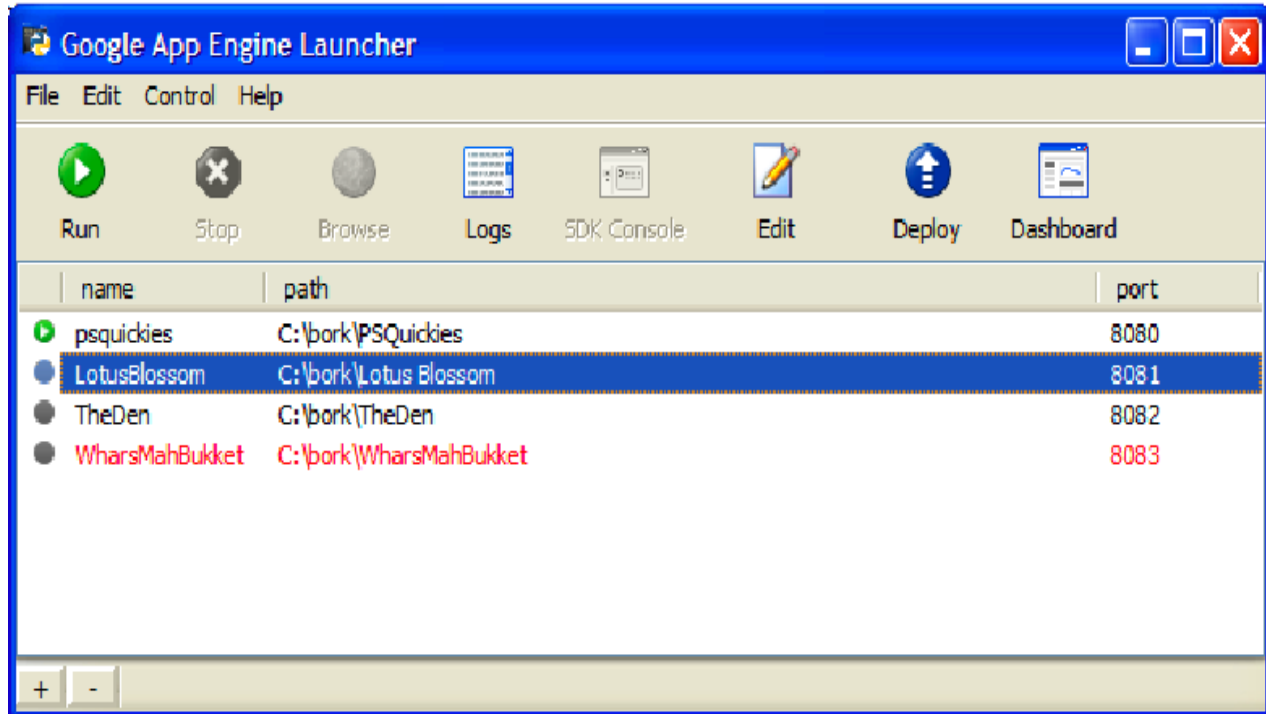
Example: -v [YOUR_VERSION_ID]

The Data store indexes might take some time to generate before your application is available. If the indexes are still in the process of being generated, you will receive a Need Index Error message when accessing your app. This is a transient error, so try a little later if at first you receive this error.

Viewing your deployed application

To launch your browser and view the app at https://PROJECT_ID.REGION_ID.r.appspot.com, run the following command:

gcloud app browse

OUTPUT:**RESULT:**

Thus the given program has been executed and verified successfully.

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 Cloud Sim and run a scheduling algorithm that is not present in Cloud Sim

PROCEDURE:

The steps to be followed: How to use CloudSim in Eclipse

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

To use CloudSim in Eclipse:

1. Download CloudSim installable files from
<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
5. The first step is to initialize the CloudSim package by initializing the CloudSim library, as follows:

CloudSim.init(num_user, calendar, trace_flag)

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

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

7. The third step is to create a broker:

DatacenterBroker broker =createBroker();

8. The fourth step is to create one virtual machine unique ID of the VM, userId ID of the VM's owner, mips, number Of Pes amount of CPUs, amount of RAM, amount of bandwidth, amount of storage, virtual machine monitor, and cloudletScheduler policy for cloudlets:

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

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

10. Create a cloudlet with length, file size, output size, and utilization model:
Cloudlet cloudlet = new Cloudlet(id, length, pesNumber, fileSize, outputSize, utilization Model, utilization Mode
11. Submit the cloudlet list to the broker:
broker.submitCloudletList(cloudletList)
12. Start the simulation:
CloudSim.startSimulation()

Sample Output from the ExistingExample:

StartingCloudSimExample1...Initialising...

Starting CloudSim version 3.0Datacenter_0 is starting...

[illegible]

Broker is starting...Entities started.

```

: Broker: Cloud Resource List received with 1 resource(s)0.0: Broker: Trying to Create
VM #0 in Datacenter_0

```

```
: Broker: VM #0 has been created in Datacenter #2,
```

Host #0 0.1: Broker: Sending cloudlet 0 to VM #0

400.1: Broker: Cloudlet 0 received

400.1: Broker: All Cloudlets executed. Finishing. 400.1: Broker: Destroying VM #0

Broker is shutting down ...Simulation: No more future events

Cloud Information Service: Notify all Cloud Sim entities for shutting down. Datacenter_0 is shutting down...

Broker is shutting down. Simulation completed.

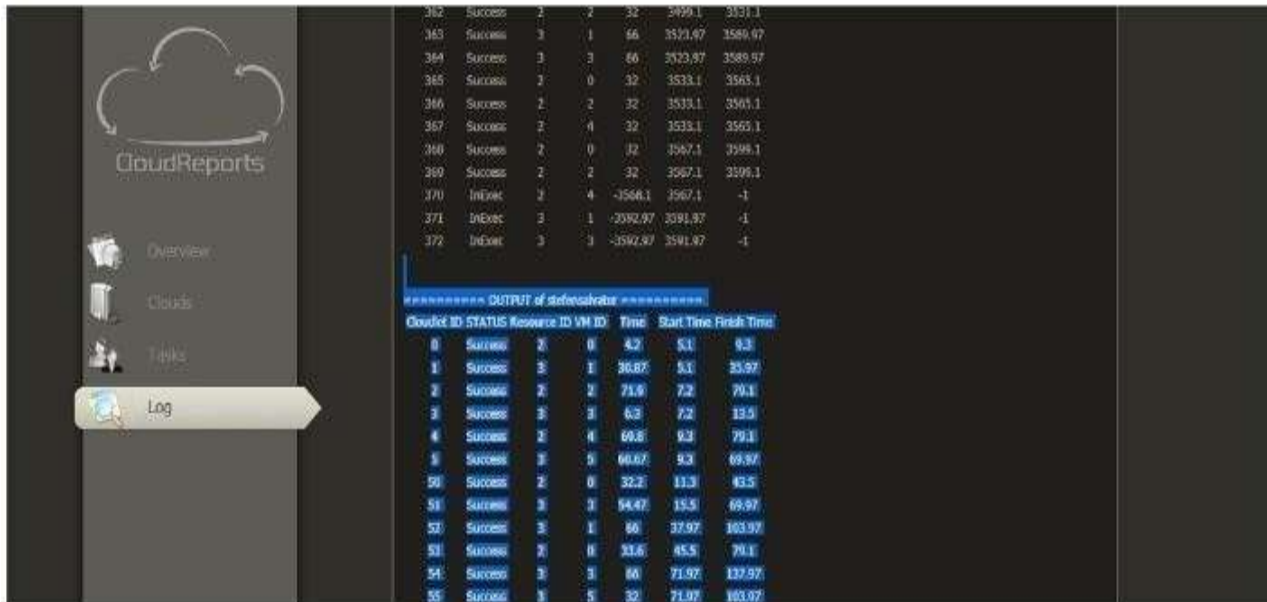
Simulation completed.

===== OUTPUT =====

Cloudlet ID	STATUS	Data center ID	VM ID	Time	Start Time	Finish Time	0 SUCCESS	2
0		400	0.1	400.1				

*****Datacenter: Datacenter_0*****		User id	Debt
3	35.6		

```
CloudSimExample1 finished!
```

OUTPUT:


362	Success	2	2	32	3499.1	3531.1
363	Success	2	1	66	3523.97	3589.97
364	Success	3	3	66	3523.97	3589.97
365	Success	2	0	32	3533.1	3565.1
366	Success	2	2	32	3533.1	3565.1
367	Success	2	4	32	3533.1	3565.1
368	Success	2	0	32	3567.1	3599.1
369	Success	2	2	32	3567.1	3599.1
370	InExec	2	4	-3568.1	3567.1	-1
371	InExec	3	1	-3592.97	3591.97	-1
372	InExec	3	3	-3592.97	3591.97	-1

Cloudlet ID	STATUS	Resource ID	VM ID	Time	Start Time	Finish Time
0	Success	2	0	4.2	5.1	9.3
1	Success	3	1	36.87	5.1	35.97
2	Success	2	2	71.9	7.2	79.1
3	Success	3	3	6.3	7.2	13.5
4	Success	2	4	68.8	9.3	79.1
5	Success	3	5	60.67	9.3	69.97
50	Success	2	0	32.2	11.3	43.5
51	Success	3	1	54.47	13.5	69.97
52	Success	3	1	60	37.97	103.97
53	Success	2	0	23.6	45.5	79.1
54	Success	3	3	60	71.97	137.97
55	Success	3	5	32	71.97	103.97

RESULT:

Thus the given program has been executed and verified successfully.

EX. NO:6 FILES TRANSFER 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.

PROCEDURE:

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 distributions install sshd by default: you can see if it is running with pgrep sshd from a shell. You can install with `sudo apt-get install openssh-server`.

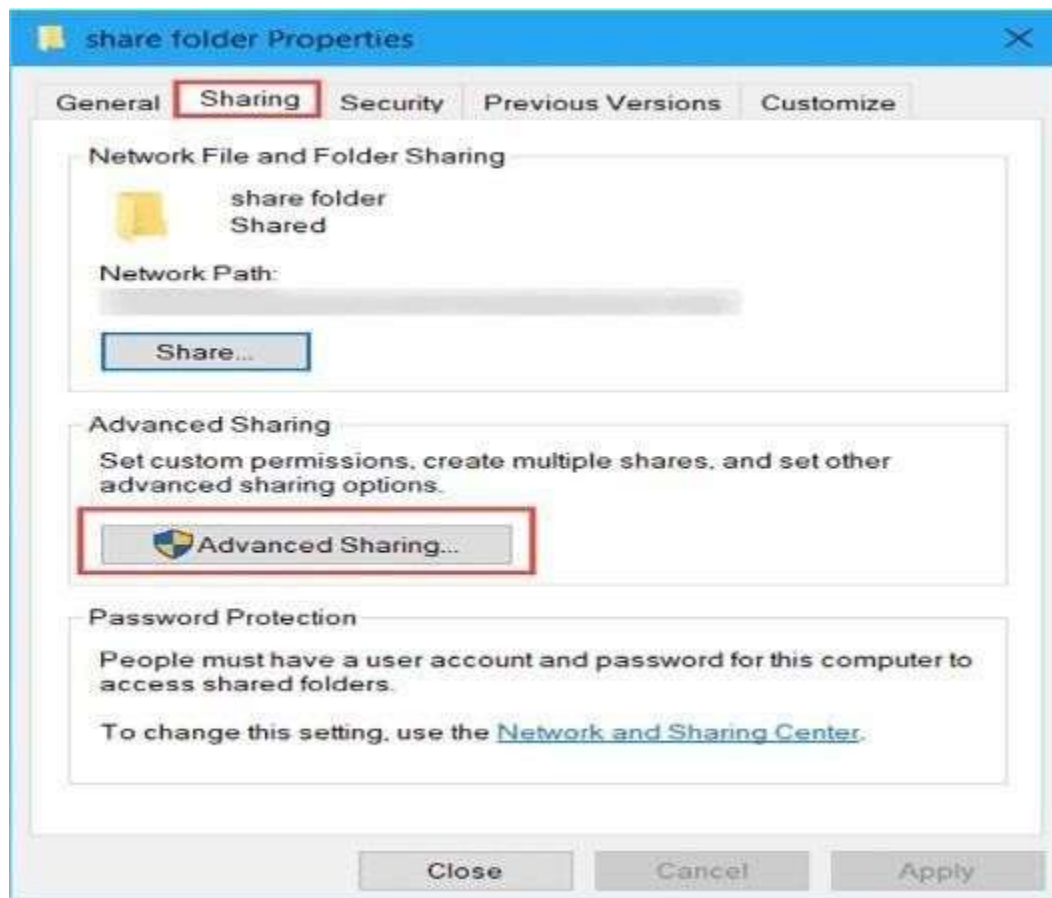
5. You can **mount part of the file system** of a virtual machine via NFS or SSHFS on the other, or you can **share file and directory** with Samba.

You may find interesting the article *Sharing files between guest and host without VirtualBox shared folders* with detailed step by step instructions.

You should remember that you are dialing 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 us 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 runs a virtual machine, for the hosting OS those virtual machines 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*.

OUTPUT:**RESULT:**

Thus the given program has been executed and verified successfully.

EX.NO:7 INSTALL HADOOP SINGLE NODE CLUSTER AND RUN SIMPLE APPLICATIONS LIKE WORDCOUNT

DATE:

AIM:

To install hadoop single node cluster and run simple applications like word count

PROCEDURE:

The steps followed for Hadoop 2 - Pseudo Node Installation

The following environment: has been created

Ubuntu Linux 64-bit

JDK 1.8.0_05

Hadoop2.7.x stable release

Note: In this document we have used only compatible versions of Hadoop ecosystem tools or software downloaded from the official Apache Hadoop website. Preferably use a stable release of the particular tool.

Prerequisites:

1. Installing Java v1.8 2.

Configuring SSH access. **sudo**

apt-get install vim -

1) Installing Java:

Hadoop is a framework written in Java for running applications on large clusters of commodity hardware. Hadoop needs Java 6 or above to work.

Step 1: Download Jdk tar.gz file for linux-62 bit, extract it into “/usr/local”

```
boss@solaiv[]# cd /opt
```

```
boss@solaiv[]# sudo tar xvpzf/home/itadmin/Downloads/jdk-8u5-linux-x64.tar.gz
```

```
boss@solaiv[]# cd /opt/jdk1.8.0_05
```

Step 2:

Open the “/etc/profile” file and add the following line as per the version

Set a environment for Java

Use the root user to save the /etc/profile or use gedit instead of vi .

The 'profile' file contains commands that ought to be run for login shells

```
boss@solaiv[]# sudo vi /etc/profile
```

```
--insert JAVA_HOME
```

```
JAVA_HOME=/opt/jdk1.8.0_05
```

```
--in PATH variable just append at the end of the line PATH=$PATH:$JAVA_HOME/bin
```

```
--Append JAVA_HOME at end of the export statement
```



```
export PATH JAVA_HOME
```

save the file using by pressing “Esc” key followed by :wq!

Step 3: Source the /etc/profileboss@solaiv[]#

```
source /etc/profile
```

Step 3: Update the java alternatives

- By default, OS will have a openjdk. Check by “java -version”. You will be prompt “openJDK”
- If you also have openjdk installed, then you'll need to update the java alternatives:
- If your system has more than one version of Java, configure which one your system causes by entering the following command in a terminal window
- By default, OS will have a openjdk. Check by “java -version”. You will be prompt “Java HotSpot(TM) 64-Bit Server”

```
boss@solaiv[]# update-alternatives --install "/usr/bin/java" java "/opt/jdk1.8.0_05/bin/java" 1
```

```
boss@solaiv[]# update-alternatives --config java --type selection number:
```

```
boss@solaiv[]# java -version
```

2) configure ssh

- Hadoop requires SSH access to manage its nodes, i.e. remote machines plus your local machine if you want to use Hadoop on it (which is what we want to do in this short tutorial). For our single-node setup of Hadoop, we therefore need to configure SSH access to localhost
- The need to create a Password-less SSH Key generation based authentication is so that the master node can then login to slave nodes (and the secondary node) to start/stop them easily without any delays for authentication
- If you skip this step, then have to provide password

- Generate an SSH key for the user.

- **sudo apt-get install openssh-server**

--You will be asked to enter password,

```
root@solaiv[]# ssh localhost
```

```
root@solaiv[]# ssh-keygen
```

```
root@solaiv[]# ssh-copy-id -i localhost
```

--After above 2 steps, You will be connected without password, root@solaiv[]# ssh

```
localhost
```

```
root@solaiv[]# exit
```

3) Hadoop installation

Now Download Hadoop from the official Apache, preferably a stable release version of Hadoop 2.7.x and extract the contents of the Hadoop package to a location of your choice.

We chose location as “/opt/”

Step 1: Download the tar.gz file of latest version Hadoop (hadoop-2.7.x) from the official site .

Step 2: Extract(untar) the downloaded file from this commands to /opt/bigdata

```
root@solaiv[]# cd /opt
root@solaiv[/opt]# sudo tar xvpzf /home/itadmin/Downloads/hadoop-2.7.0.tar.gz
root@solaiv[/opt]# cd hadoop-2.7.0/
```

Like java, update Hadoop environment variable in /etc/profile

```
boss@solaiv[]# sudo vi /etc/profile
#--insert HADOOP_PREFIX HADOOP_PREFIX=/opt/hadoop-2.7.0
#--in PATH variable just append at the end of the line
PATH=$PATH:$HADOOP_PREFIX/bin
#--Append HADOOP_PREFIX at end of the export statement
export PATH JAVA_HOME HADOOP_PREFIX
```

save the file using by pressing “Esc” key followed by :wq!

Step 3: Source the /etc/profile

```
@solaiv[]# source /etc/profile
```

Verify Hadoop installation

```
boss@solaiv[]# cd $HADOOP_PREFIX
boss@solaiv[]# bin/hadoop version
```

3.1) Modify the Hadoop Configuration Files

In this section, we will configure the directory where Hadoop will store its configuration files, the network ports it listens to, etc. Our setup will use Hadoop Distributed File System,(HDFS), even though we are using only a single local machine.

Add the following properties in the various hadoop configuration files which is available under \$HADOOP_PREFIX/etc/hadoop/

core-site.xml, hdfs-site.xml, mapred-site.xml & yarn-site.xml

Update Java, hadoop path to the Hadoop environment file

```
boss@solaiv[]# cd $HADOOP_PREFIX/etc/hadoop
boss@solaiv[]# vi hadoop-env.sh
```

Paste following line at beginning of the file

```
export JAVA_HOME=/usr/local/jdk1.8.0_05
export HADOOP_PREFIX=/opt/hadoop-2.7.0
```

Paste following between <configuration> tags

Modify the **core-site.xml**

```
boss@solaiv[]# cd $HADOOP_PREFIX/etc/hadoop
boss@solaiv[]# vi core-site.xml
```

Paste following between <configuration> tags

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

Modify the **hdfs-site.xml**

```
boss@solaiv[]# vi hdfs-site.xml
```

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>
</configuration>
```

Paste following between <configuration> tags

YARN configuration - Single Node modify the **mapred-site.xml**

```
boss@solaiv[]# cp mapred-site.xml.template mapred-site.xml
boss@solaiv[]# vi mapred-site.xml
```

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



</configuration>

Modiy yarn-site.xml

```
boss@solaiv[]# vi yarn-site.xml
```

Paste following between <configuration> tags

```
<configuration>
  <property>
    <name>yarn.nodemanager.aux-services</name>
    <value>mapreduce_shuffle</value>
  </property>
</configuration>
```

Formatting the HDFS file-system via the NameNode

The first step to starting up your Hadoop installation is formatting the Hadoop files system which is implemented on top of the local file system of our “cluster” which includes only our local machine. We need to do this the first time you set up a Hadoop cluster.

Do not format a running Hadoop file system as you will lose all the data currently in the cluster (in HDFS)

```
root@solaiv[]# cd $HADOOP_PREFIX
root@solaiv[]# bin/hadoop namenode -format
```

Start NameNode daemon and DataNode daemon: (port 50070)

```
root@solaiv[]# sbin/start-dfs.sh
```

To know the running daemons jut type jps or /usr/local/jdk1.8.0_05/bin/jps Start ResourceManager daemon and NodeManager daemon: (port 8088)

```
root@solaiv[]# sbin/start-yarn.sh
```

To stop the running process

```
root@solaiv[]# sbin/stop-dfs.sh
```

- To know the running daemons jut type jps or /usr/local/jdk1.8.0_05/bin/jps
- Start ResourceManager daemon and NodeManager daemon: (port 8088)

```
root@solaiv[]# sbin/stop-yarn.sh
```

Make the HDFS directories required to execute MapReduce jobs:

```
$ bin/hdfs dfs -mkdir /user  
$ bin/hdfs dfs -mkdir /user/mit
```

- Copy the input files into the distributed filesystem:

```
$ bin/hdfs dfs -put <input-path>/* /input
```

- Run some of the examples provided:

```
$ bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.5.1.jar grep  
/input /output '(CSE)'
```

- Examine the output files:

Copy the output files from the distributed filesystem to the local filesystem and examine them:

```
$ bin/hdfs dfs -get output output  
$ cat output/*
```

or

View the output files on the distributed filesystem:

```
$ bin/hdfs dfs -cat /output/*
```

HDFS Shell commands

```
cd $HADOOP_PREFIX
```

```
#create new dir
```

```
bin/hadoopfs -mkdir /mit/
```

```
bin/hadoopfs -mkdir /mit/day2
```

OR

```
bin/hadoopfs -mkdir -p /mit/day2/
```

```
bin/hadoopfs -mkdir /mit/day3
```

```
# rmdir
```

```
bin/hadoopfs -rm /mit/day3
```

```
#Append single src, or multiple srcs from local file system to the destination file system.
```

Also reads

input from stdin and appends to destination file system.

```
bin/hdfsdfs -appendToFilelocalfile /user/hadoop/hadoopfile
```

#both localfile1 and localfile2 will be appended to hdfs
bin/hdfsdfs -appendToFile localfile1 localfile2 /user/hadoop/hadoopfile
#HDFS path can also be given like below
bin/hdfsdfs -appendToFilelocalfile hdfs://localhost:9000/user/hadoop/hadoopfile
bin/hdfsdfs -appendToFile /home/bigdata/Downloads/ hdfs:///user/hadoop/hadoopfile
#read the input from stdin
hdfsdfs -appendToFile - hdfs://nn.example.com/hadoop/hadoopfile
#prompt will wait to Reads the input from stdin. to finish <ctrl+c>
#cat
bin/hadoopfs -cat /user/hadoop/hadoopfile
#chown&chmod
#put (or) copyFromLocal
bin/hadoopfs -put /media/bdalab/shortNotes/examples_files/vote_data /mit
bin/hadoopfs -put /home/bigdata/Downloads/mrdata /mit/day2
bin/hadoopfs -put /home/bigdata/Downloads/hivedata /mit/day2
#moveFromLocal
#Similar to put command, except that the source <localsrc> is deleted after it's copied.
bin/hdfsdfs -moveFromLocal<localsrc><dst>
#cp is used to copy files between directories present in HDFS
./bin/hdfsdfs -cp /user/hadoopfile /user/hadoop/hadoopfile
#mv, move the file from one hdfs location to other
./bin/hdfsdfs -mv /user/hadoopfile /user/hadoop/hadoopfile
#list the files
bin/hadoopfs -ls /
#The output columns with -count are: DIR_COUNT, FILE_COUNT, CONTENT_SIZE
FILE_NAME
./bin/hdfsdfs -count /
#du display the size, dir
./bin/hdfsdfs -du -h /
for aggregate summary, -s
./bin/hdfsdfs -du -h -s /
#expunge, Empty the Trash.
bin/hdfsdfs -expunge

```
#Displays the Access Control Lists (ACLs) of files and directories
#we can get the same by using "ls" command
./bin/hdfsdfs -getfacl /user
#getmerge
#Sets an extended attribute name and value for a file or directory. -x <name>, remove the
attribute
hdfsdfs -setfattr -n user.myAttr -v myValue /user
#Displays the extended attribute names and values (if any) for a file or directory.
./bin/hdfsdfs -getfattr -d /user
#setrep, change replication factor of a file -w wait till the replication is achived
#hdfsdfs -setrep [-R] [-w] <numReplicas><path>
hdfsdfs -setrep -w 2 /user/hadoop/dir1
#distcp, Copy a directory from one node in the cluster toanother
# -overwrite option to overwrite in an existing files
# -update command to synchronize bothdirectories
hadoopfs -distcp hdfs://namenodeA/apache_hadoop hdfs://namenodeB/hadoop
#tail, Displays last kilobyte of the file to stdout.
bin/hdfsdfs -tail /filename
#change the default configuration by using Generic Option
./bin/hadoopfs -D dfs.replication=2 -mkdir /user/solai
MapRed
===== #running
./bin/hadoop jar share/hadoop/mapreduce/hadoop-mapreduce-examples-2.5.1.jar
wordcount /mit/day2/mrdata /op/
#list job
./bin/mapred job -list
#in order to show the previous running job details, start history server, port 10020
./sbin/mr-jobhistory-daemon.sh start historyserver
#job status
./bin/mapred job -status job_1423283172541_0001
#Get the file size As compared against OS and Hadoop
#OS size
blockdev --getbsz /dev/sda1
#HDFS Size
```



```
./bin/hdfsgetconf -confKeydfs.blocksize
```

WORDCOUNT PROGRAM WITH USE OF MAP AND REDUCE TASKS

PROCEDURE:

Hadoop MapReduce is a software framework for easily writing applications which process vast amounts of data (multi-terabyte data-sets) in-parallel on large clusters (thousands of nodes) of commodity hardware in a reliable, fault-tolerant manner.

A MapReduce job usually splits the input data-set into independent chunks which are processed by the map tasks in a completely parallel manner. The framework sorts the outputs of the maps, which are then input to the reduce tasks. Typically both the input and the output of the job are stored in a file-system. The framework takes care of scheduling tasks, monitoring them and re-executes the failed tasks.

Typically the compute nodes and the storage nodes are the same, that is, the MapReduce framework and the Hadoop Distributed File System are running on the same set of nodes. This configuration allows the framework to effectively schedule tasks on the nodes where data is already present, resulting in very high aggregate bandwidth across the cluster.

The MapReduce framework consists of a single master ResourceManager, one slave NodeManager per cluster-node, and MRAppMaster per application.

Minimally, applications specify the input/output locations and supply map and reduce functions via implementations of appropriate interfaces and/or abstract-classes. These, and other job parameters, comprise the job configuration.

The Hadoop job client then submits the job (jar/executable etc.) and configuration to the ResourceManager which then assumes the responsibility of distributing the software/configuration to the slaves, scheduling tasks and monitoring them, providing status and diagnostic information to the job-client.

Prerequisites:

- You have set up a single-node "cluster" by following the single-node setup
- We assume that you run commands from inside the Hadoop directory.
- This program uses the HadoopStreaming API to interact with Hadoop. This API allows you to write code in any language and use a simple text-based record format for the input and output <key, value> pairs.
- The output of this program will fetch the titles of web pages at particular URLs.

Inputs and Outputs

The MapReduce framework operates exclusively on <key, value> pairs, that is, the framework views the input to the job as a set of <key, value> pairs and produces a set of <key, value> pairs as the output of the job, conceivably of different types.

The key and value classes have to be serializable by the framework and hence need to implement the [Writable](#) interface. Additionally, the key classes have to implement the [WritableComparable](#) interface to facilitate sorting by the framework.

Input and Output types of a MapReduce job:

(input) <k1, v1> -> **map** -> <k2, v2> -> **combine** -> <k2, v2> -> **reduce** -> <k3, v3> (output)

CODING:

WordCount is a simple application that counts the number of occurrences of each word in a given input set.

```
import java.io.IOException;
import java.util.StringTokenizer;

import org.apache.hadoop.conf.Configuration;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapreduce.Job;
import org.apache.hadoop.mapreduce.Mapper;
import org.apache.hadoop.mapreduce.Reducer;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;

public class WordCount {

    public static class TokenizerMapper
        extends Mapper<Object, Text, Text, IntWritable>{

        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();

        public void map(Object key, Text value, Context context
            ) throws IOException, InterruptedException {
            StringTokenizer itr = new StringTokenizer(value.toString());
            while (itr.hasMoreTokens()) {
                word.set(itr.nextToken());
                context.write(word, one);
            }
        }
    }

    public static class IntSumReducer
        extends Reducer<Text,IntWritable,Text,IntWritable> {
        private IntWritable result = new IntWritable();
    }
}
```

```

public void reduce(Text key, Iterable<IntWritable> values,
                  Context context
                  ) throws IOException, InterruptedException {
    int sum = 0;
    for (IntWritable val : values) {
        sum += val.get();
    }
    result.set(sum);
    context.write(key, result);
}
}

public static void main(String[] args) throws Exception {
    Configuration conf = new Configuration();
    Job job = Job.getInstance(conf, "word count");
    job.setJarByClass(WordCount.class);
    job.setMapperClass(TokenizerMapper.class);
    job.setCombinerClass(IntSumReducer.class);
    job.setReducerClass(IntSumReducer.class);
    job.setOutputKeyClass(Text.class);
    job.setOutputValueClass(IntWritable.class);
    FileInputFormat.addInputPath(job, new Path(args[0]));
    FileOutputFormat.setOutputPath(job, new Path(args[1]));
    System.exit(job.waitForCompletion(true) ? 0 : 1);
}
}

```

EXECUTION :

Environment variables are set as follows:

```

export JAVA_HOME=/usr/java/default
export PATH=${JAVA_HOME}/bin:${PATH}
export HADOOP_CLASSPATH=${JAVA_HOME}/lib/tools.jar

```

The ant build file

Hadoop launches jobs by getting a jar file containing the compiled Java code. In addition, we typically send two command line arguments through to the Java program: the input data file or directory, and an output directory for the results from the reduce tasks. Using a tool called ant makes it pretty quick to create a jar file from the above code.

The ant tool uses an xml file that describes what needs to be compiled and packaged into a jar file. Here is the one you used for the above WordCount example:

```

<project name="hadoopCompile" default="jar" basedir=".">
  <target name="init">

```

```

<property name="sourceDir" value="." />
<property name="outputDir" value="classes" />
<property name="buildDir" value="jar" />
<property name="lib.dir" value="/usr/lib/hadoop"/>

<path id="classpath">
  <fileset dir="${lib.dir}" includes="**/*.jar"/>
</path>
</target>
<target name="clean" depends="init">
  <delete dir="${outputDir}" />
  <delete dir="${buildDir}" />
</target>
<target name="prepare" depends="clean">
  <mkdir dir="${outputDir}" />
  <mkdir dir="${buildDir}" />
</target>
<target name="compile" depends="prepare">
  <javac srcdir="${sourceDir}" destdir="${outputDir}" classpathref="classpath"
/>
</target>
<target name="jar" depends="compile">

  <jar destfile="${buildDir}/wc.jar" basedir="${outputDir}">
    <manifest>
      <attribute name="Main-Class" value="wc.WordCount"/>
    </manifest>
  </jar>
</target>
</project>

```

Compile WordCount.java and create a jar:

```

$ bin/hadoop com.sun.tools.javac.Main WordCount.java
$ jar cf wc.jar WordCount*.class

```

Assuming that:

- /user/joe/wordcount/input - input directory in HDFS
- /user/joe/wordcount/output - output directory in HDFS

Sample text-files as input:

- \$ bin/hadoop fs -ls /user/joe/wordcount/input/ /user/joe/wordcount/input/file01 /user/joe/wordcount/input/file02

- **\$ bin/hadoop fs -cat/user/joe/wordcount/input/file01**
hai hello how are you
- **\$ bin/hadoop fs -cat /user/joe/wordcount/input/file02**
- **hai hello how you are**

Run the application:

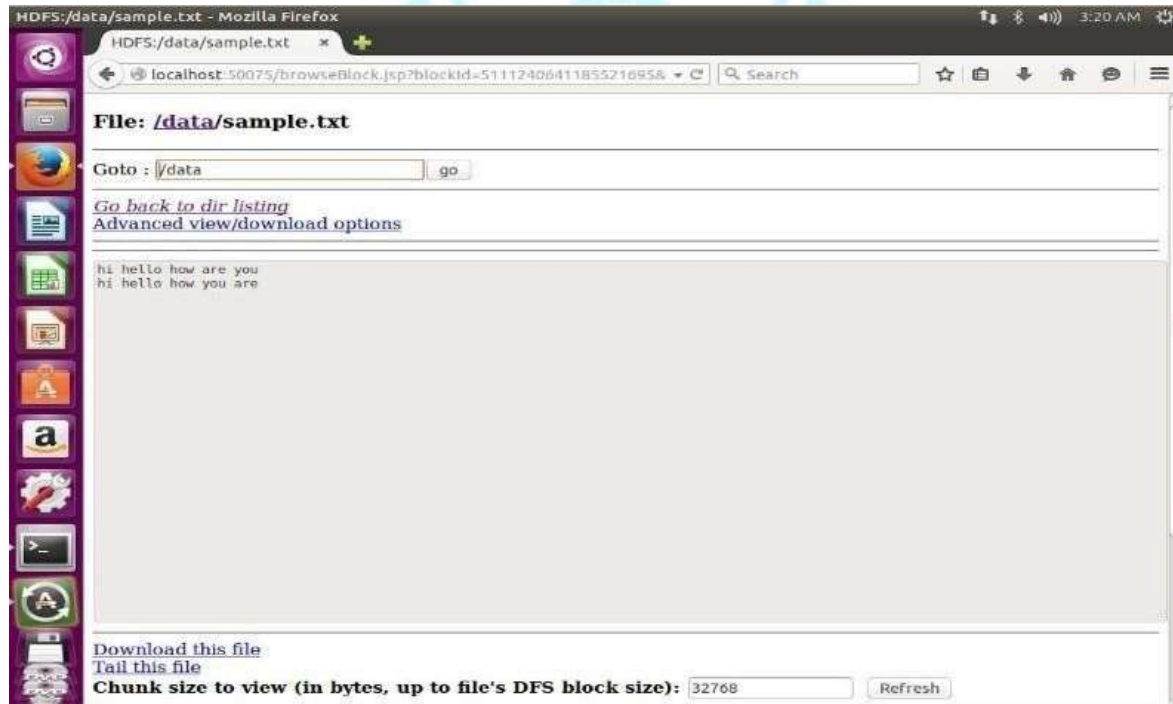
```
$ bin/hadoop jar wc.jar WordCount /user/joe/wordcount/input  
/user/joe/wordcount/output
```

Applications can specify a comma separated list of paths which would be present in the current working directory of the task using the option `-files`. The `-libjars` option allows applications to add jars to the classpaths of the maps and reduces. The option `-archives` allows them to pass comma separated list of archives as arguments. These archives are unarchived and a link with name of the archive is created in the current working directory of tasks.

Running wordcount example with `-libjars`, `-files` and `-archives`:

```
$ bin/hadoop jar hadoop-mapreduce-examples-<ver>.jar wordcount -files  
cachefile.txt -libjars mylib.jar -archives myarchive.zip input output
```

OUTPUT:



**RESULT:**

Thus the given program has been executed and verified successfully.

EX NO:8 CREATING AND EXECUTING YOUR FIRST CONTAINER USING DOCKER

Date:

Aim:

To create and execute your first container using Docker.

Procedure:

Docker Image

A docker image is a file used to execute code in a docker container. Docker images acts as a set of instructions to build a docker container, like a template. An image is comparable to a snapshot in virtual machines (VM) environments.

Docker Container

A Docker Container is a lightweight, standalone, executable package of software that includes everything needed to run an application code, runtime, system tools, system libraries and settings.

How to run a container?

Step 1: Images are used to run containers.



Step 2: Get the sample applications. i.e, Clone the repository

Step 3: Verify your Docker File.

- Create an application in your IDE
- For Example:

#start your image with a node

From node: 18-alpine

#create an application directory

Run mkdir -p/app

Step 4: Build your first image.

docker build -t filename

Note: -t flag tag your image with a name. Lets docker know where it can get the file.

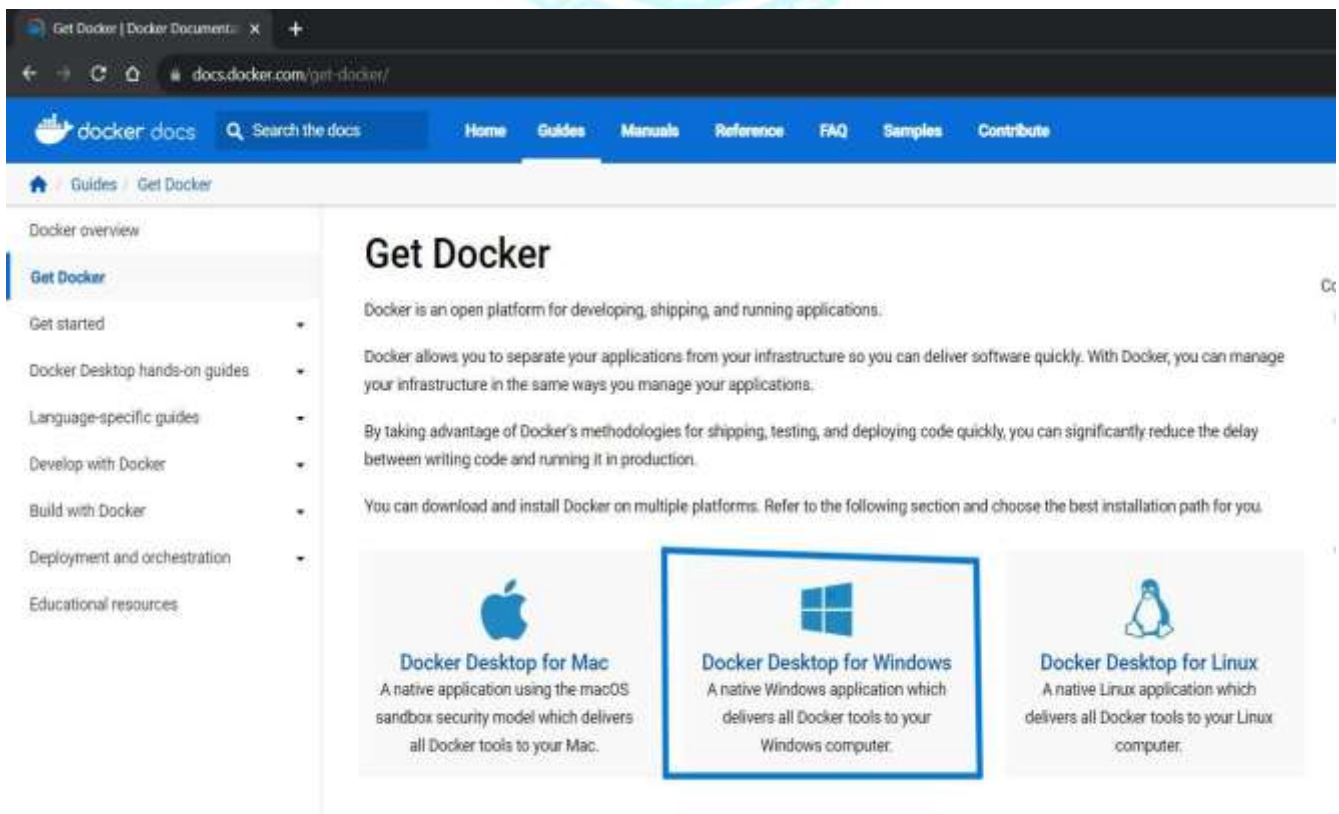
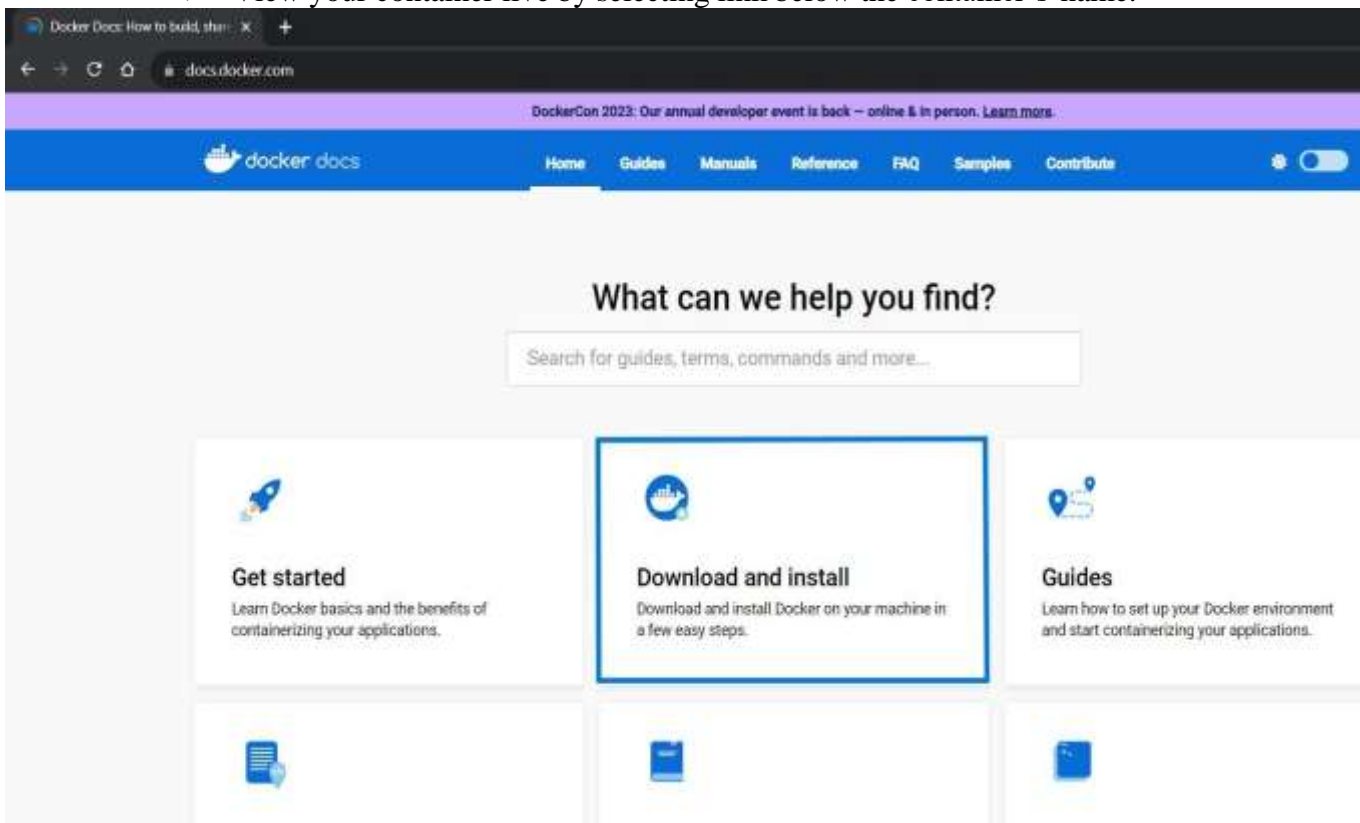
Step 5: Run your container.

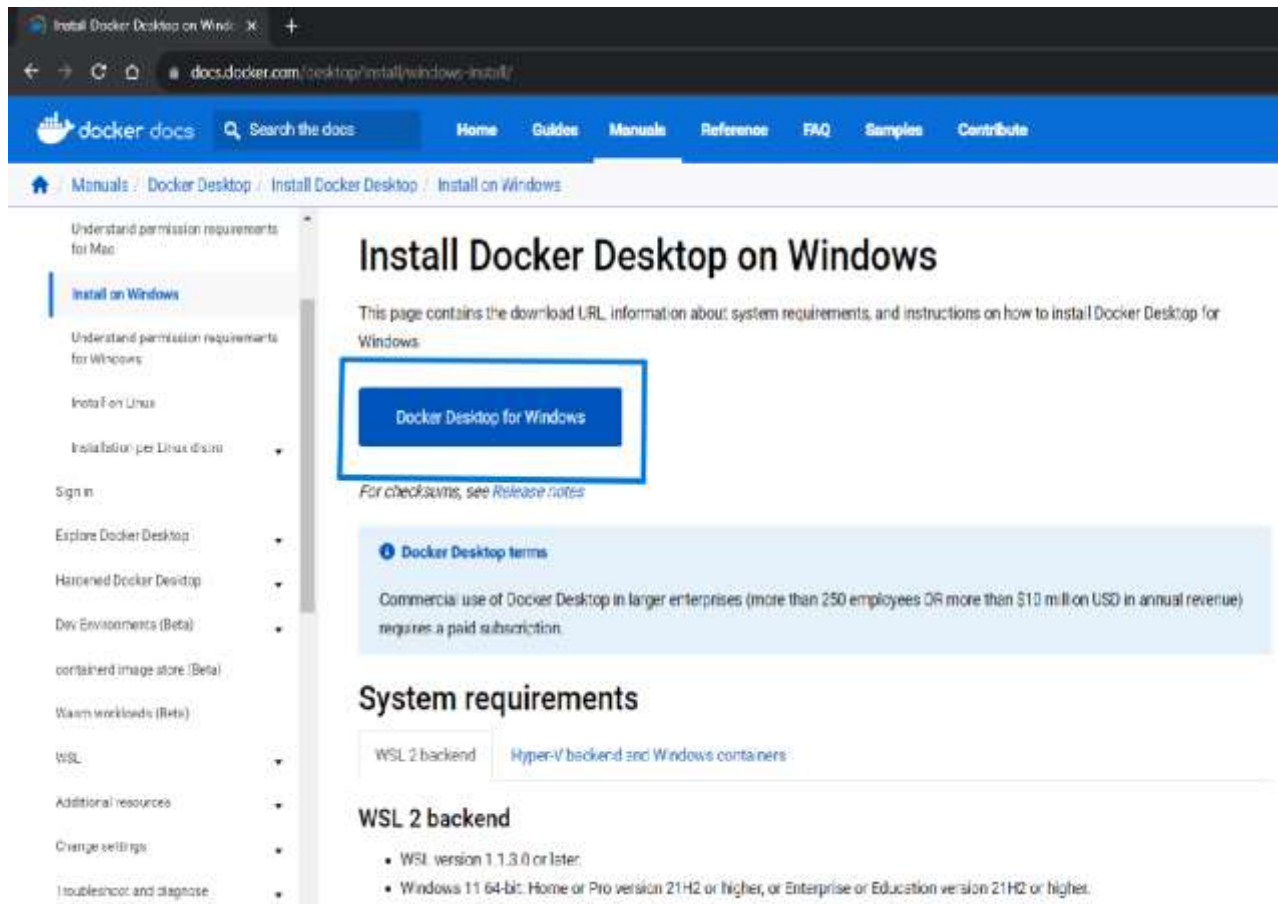
- Once the build is complete, an image will appear in the images tab.
- Select the image name to see its details.
- Select **run** to run it as a container.
- In the **optional settings**, remember to specify a port number (something like 8089).

Step 6: View the frontend.

- You now have a running container.

- If you don't specify a name for your container, Docker provides one.
- View your container live by selecting link below the container's name.





The screenshot shows the Docker Desktop installation page for Windows on the Docker Docs website. The page title is "Install Docker Desktop on Windows". It includes a sidebar with navigation links, a main content area with a download button, and a section for system requirements.

Install Docker Desktop on Windows

This page contains the download URL, information about system requirements, and instructions on how to install Docker Desktop for Windows.

[Docker Desktop for Windows](#)

For checksums, see Release notes

Docker Desktop terms

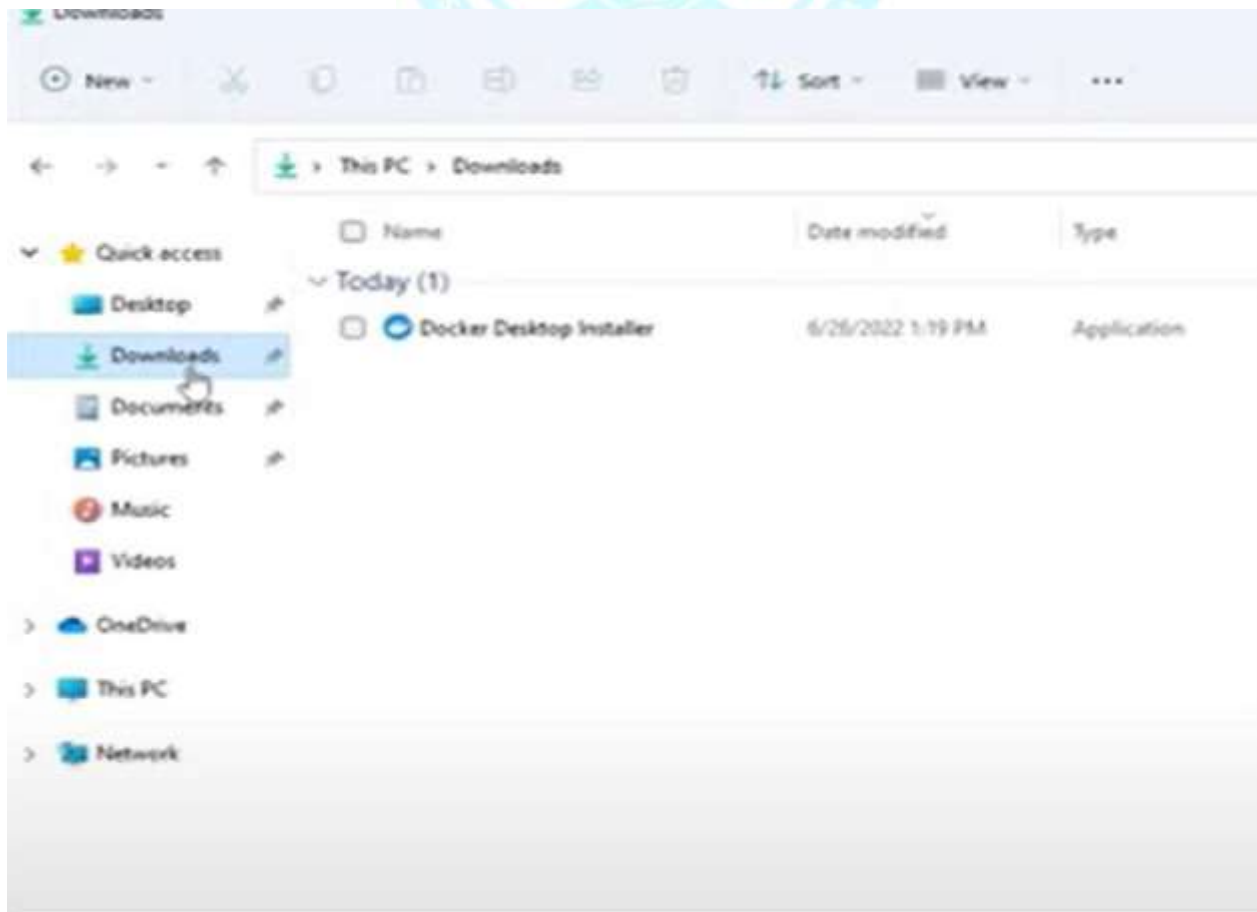
Commercial use of Docker Desktop in larger enterprises (more than 250 employees OR more than \$10 million USD in annual revenue) requires a paid subscription.

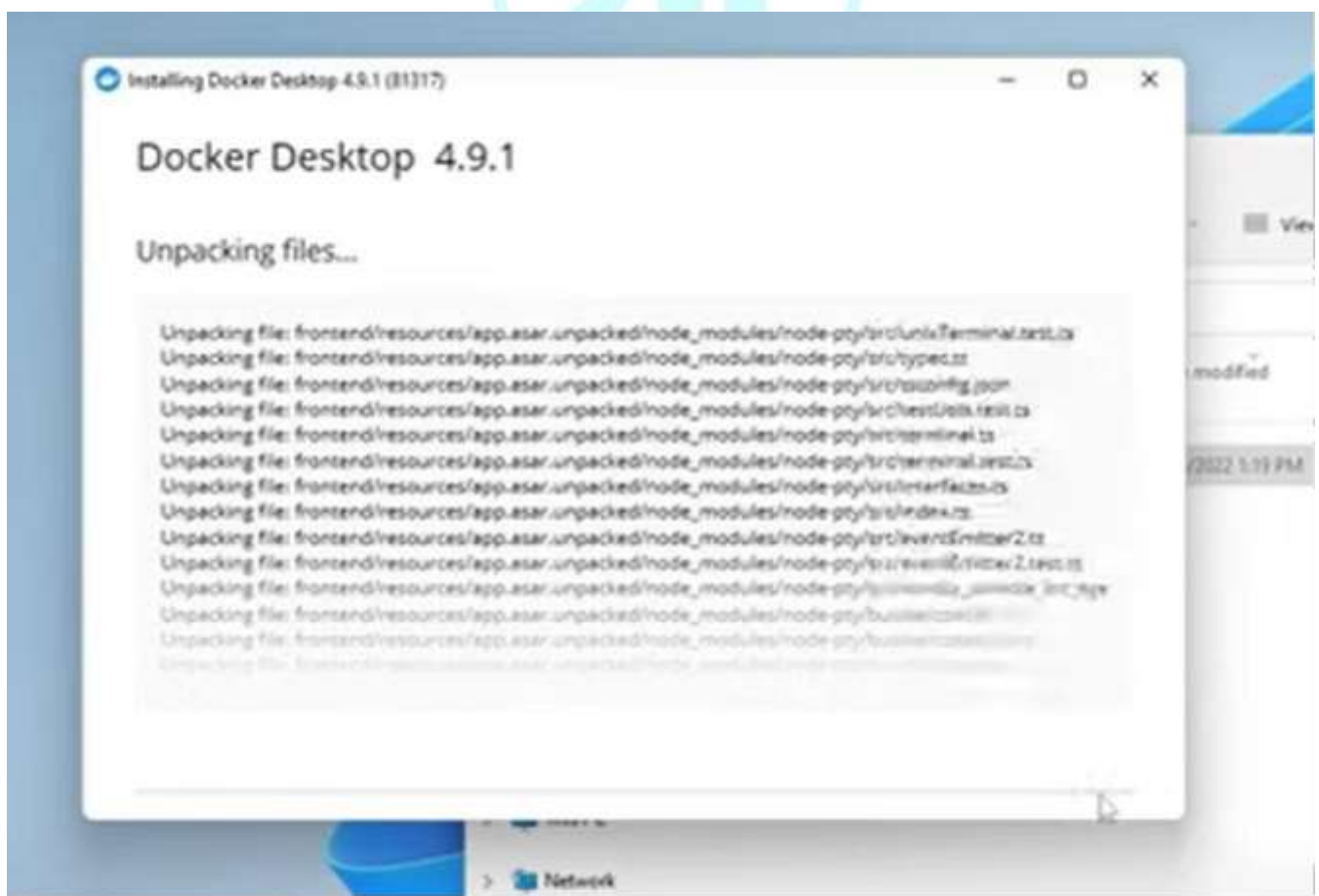
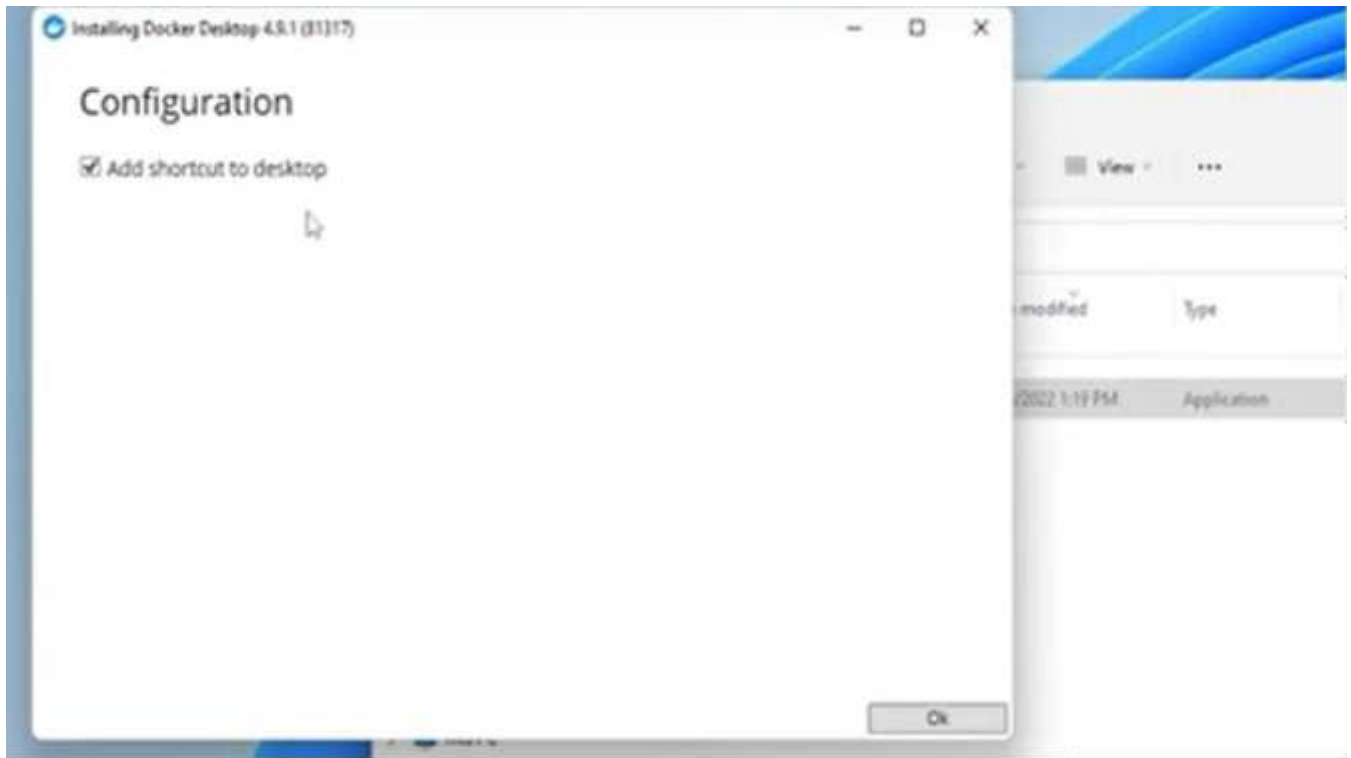
System requirements

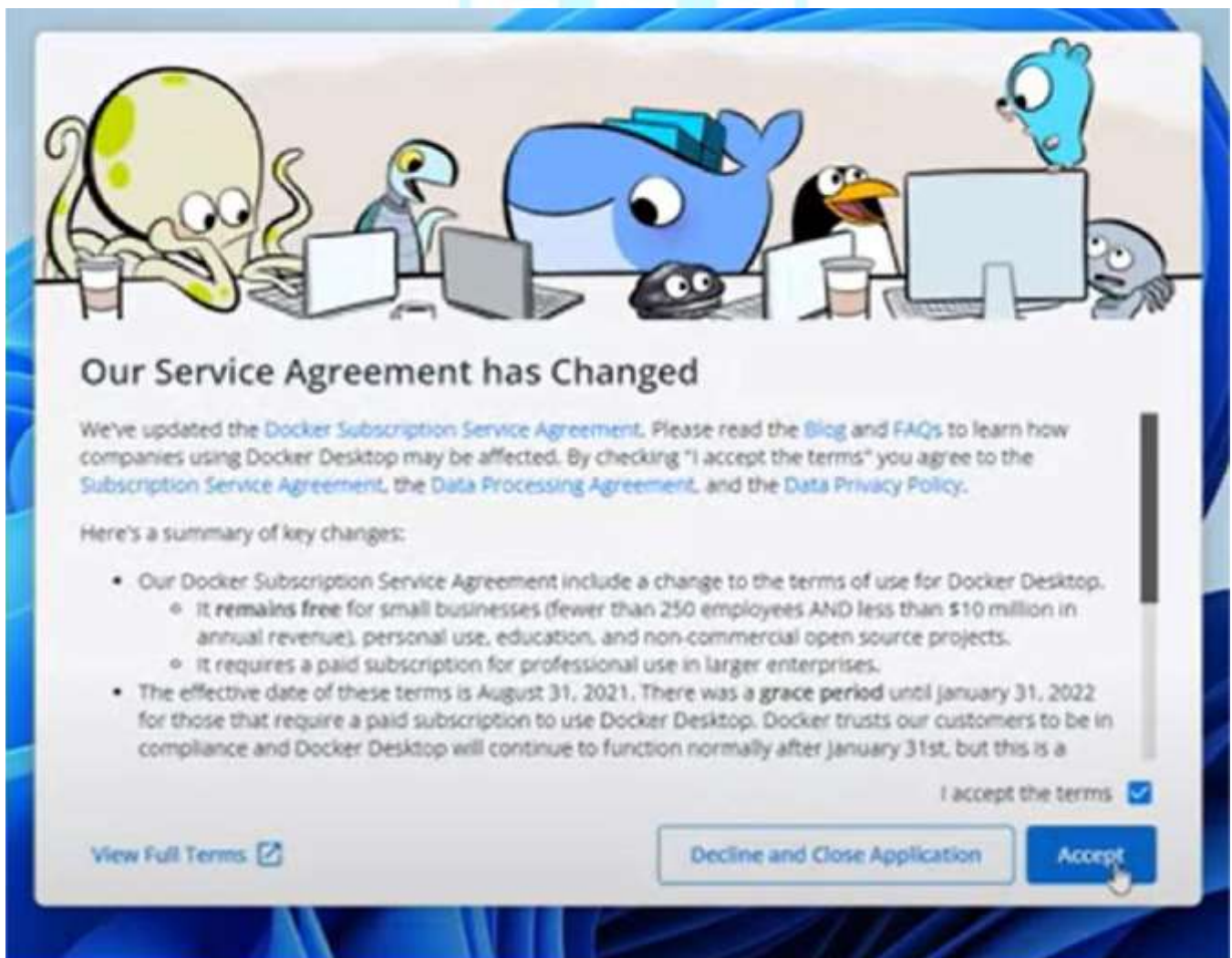
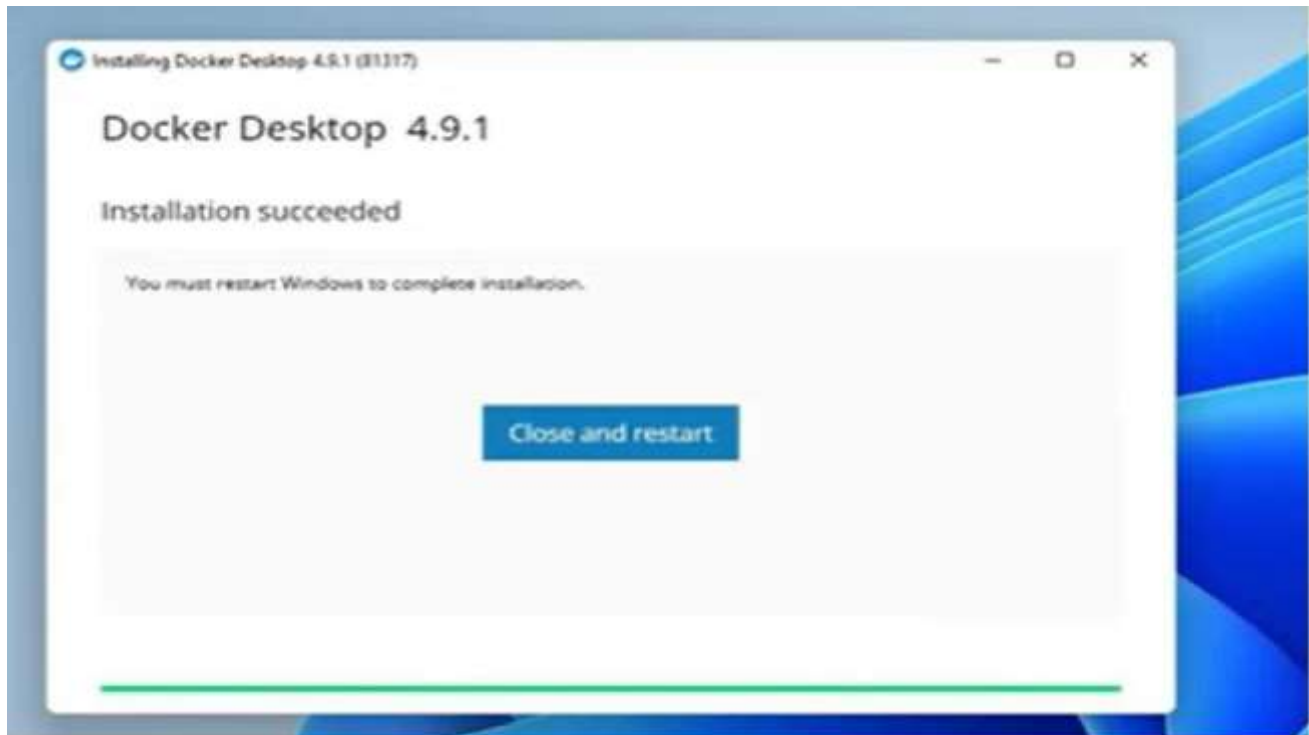
[WSL 2 backend](#) [Hyper-V backend and Windows containers](#)

WSL 2 backend

- WSL version 1.1.3.0 or later.
- Windows 11 64-bit, Home or Pro version 21H2 or higher, or Enterprise or Education version 21H2 or higher.









- WSL Documentation
- > Overview
- > Install
 - Install WSL
 - Manual install steps for older versions**
 - Install on Windows Server
- > Tutorials
- > Concepts
- > How-to
 - Frequently Asked Questions
 - Troubleshooting
- > Release Notes
- Download PDF

Step 4 - Download the Linux kernel update package

The Linux kernel update package installs the most recent version of the [WSL 2 Linux kernel](#) for running WSL inside the Windows operating system image. (To run [WSL from the Microsoft Store](#), with more frequently pushed updates, use `wsl.exe --install` or `wsl.exe --update`.)

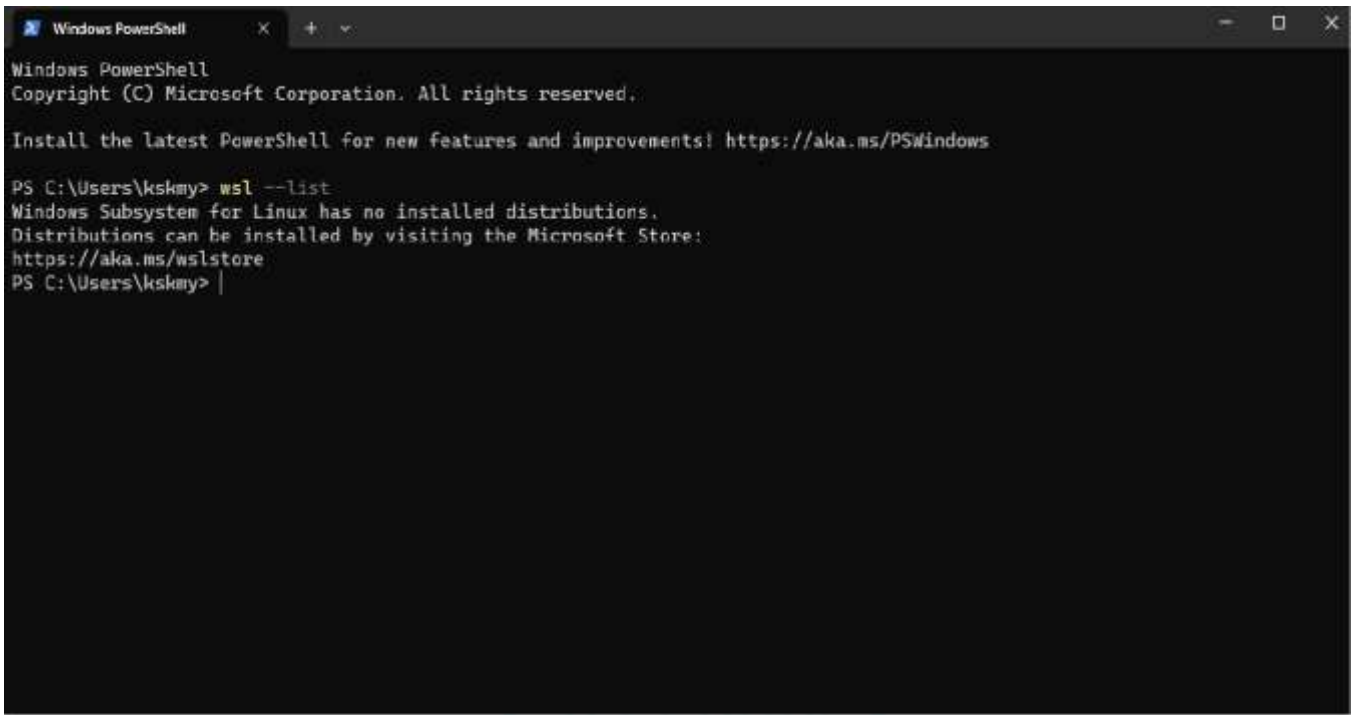
1. Download the latest package:

- [WSL2 Linux kernel update package for x64 machines](#)

Note

If you're using an ARM64 machine, please download the [ARM64 package](#) instead. If you're not sure what kind of machine you have, open Command Prompt or PowerShell and enter: `systeminfo | find "System Type"`. **Caveat:** On non-English Windows versions, you might have to modify the search text, translating the "System Type" string. You may also need to escape the quotations for the find command. For example, in German: `systeminfo | find "Systemtyp"`.

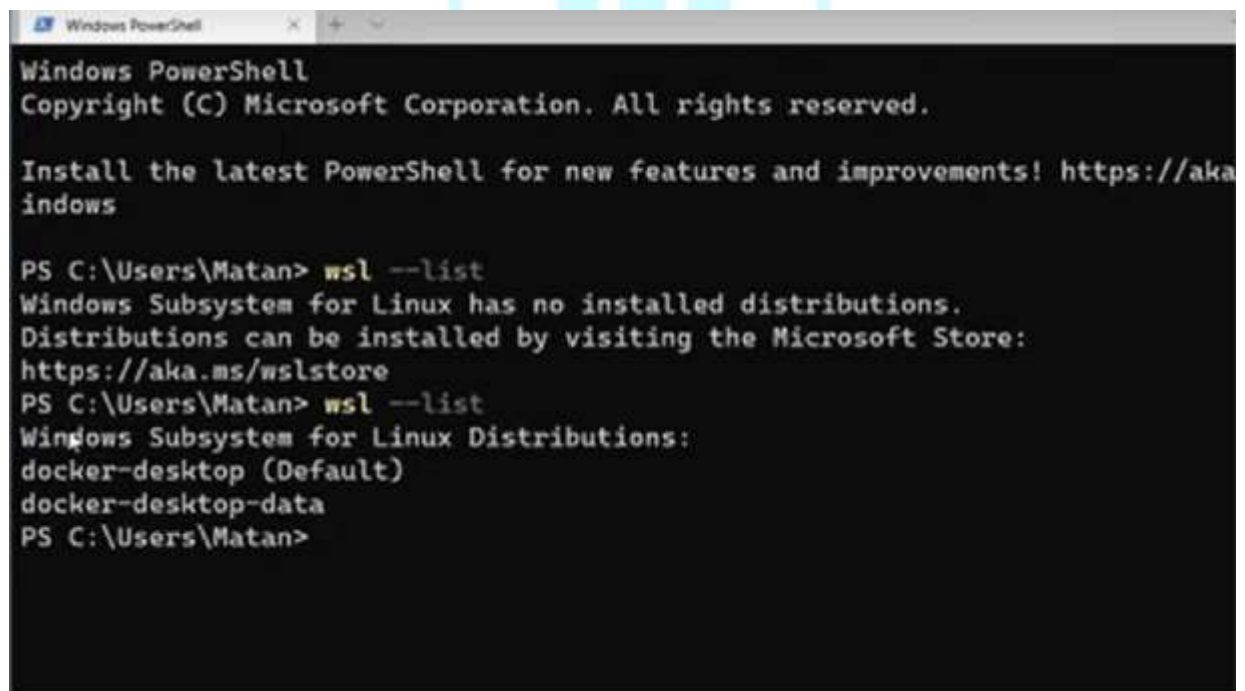
2. Run the update package downloaded in the previous step. (Double-click to run - you will be prompted for elevated permissions, select 'yes' to approve this installation.)



```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

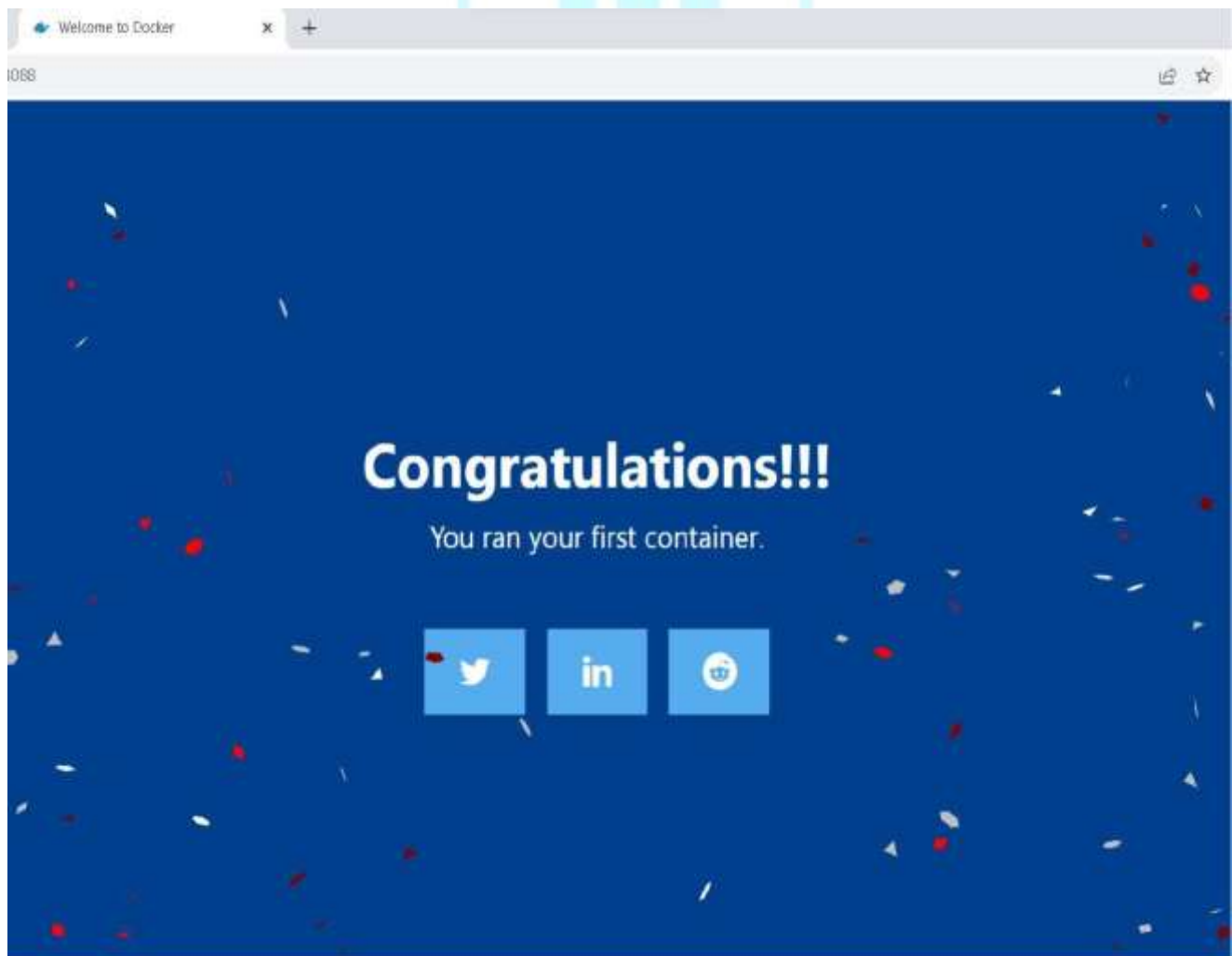
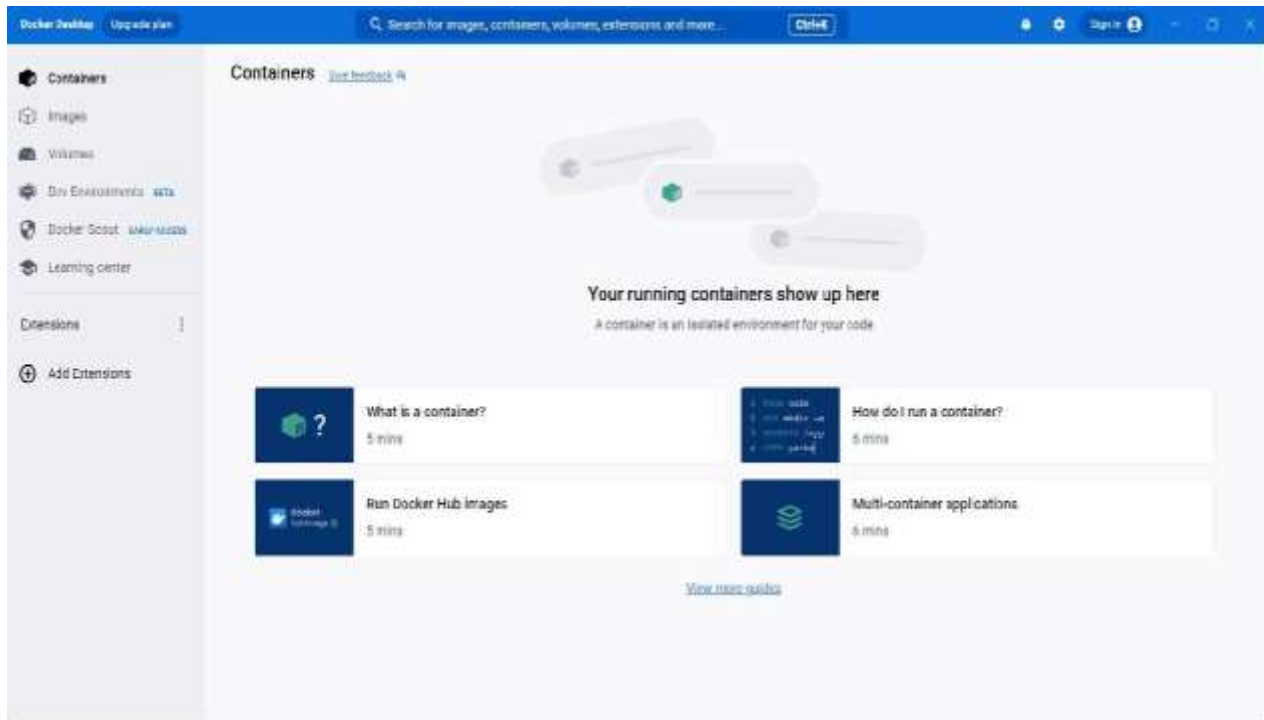
PS C:\Users\kskmy> wsl --list
Windows Subsystem for Linux has no installed distributions.
Distributions can be installed by visiting the Microsoft Store:
https://aka.ms/wslstore
PS C:\Users\kskmy> |
```

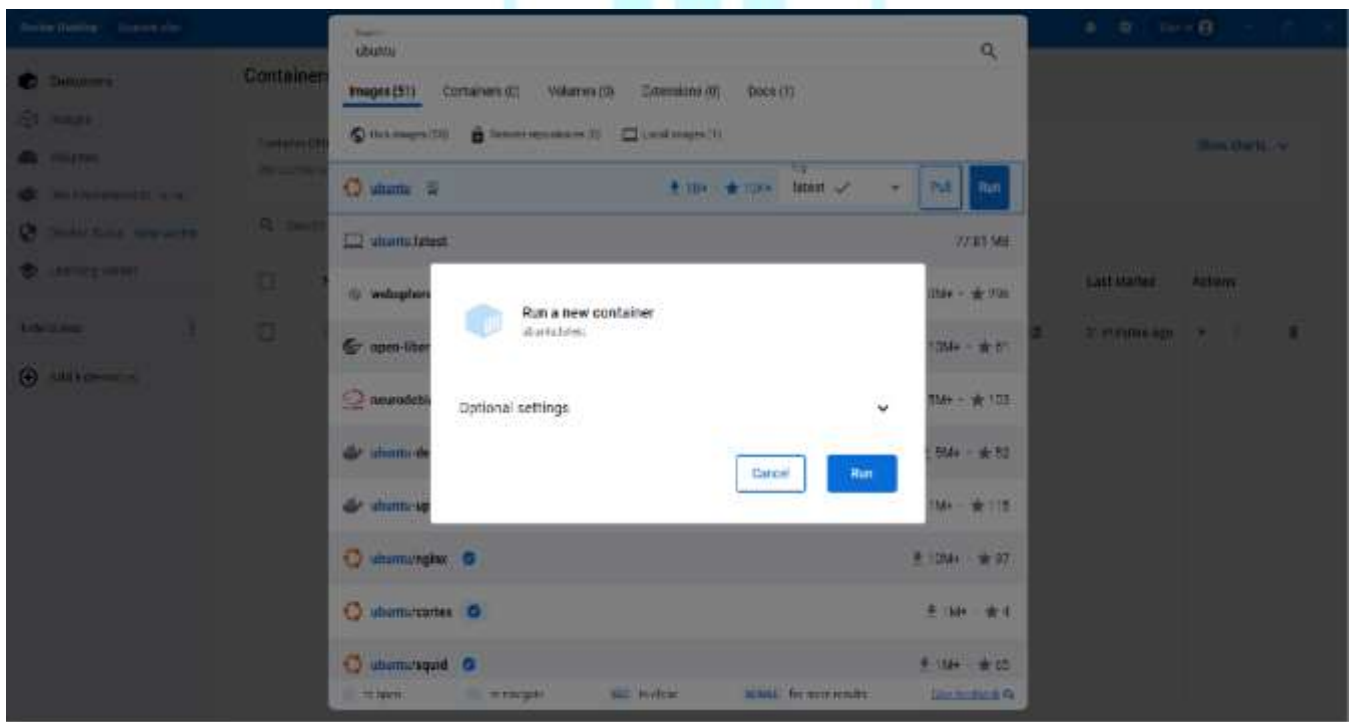
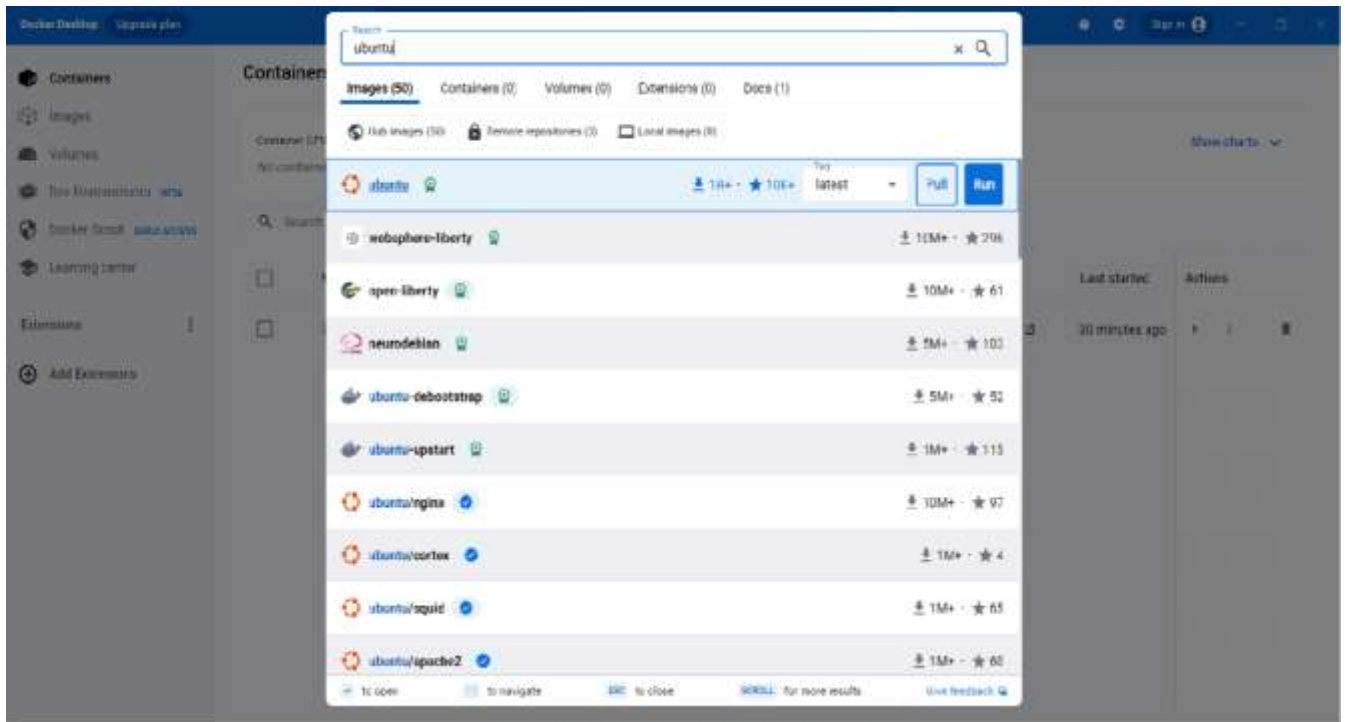


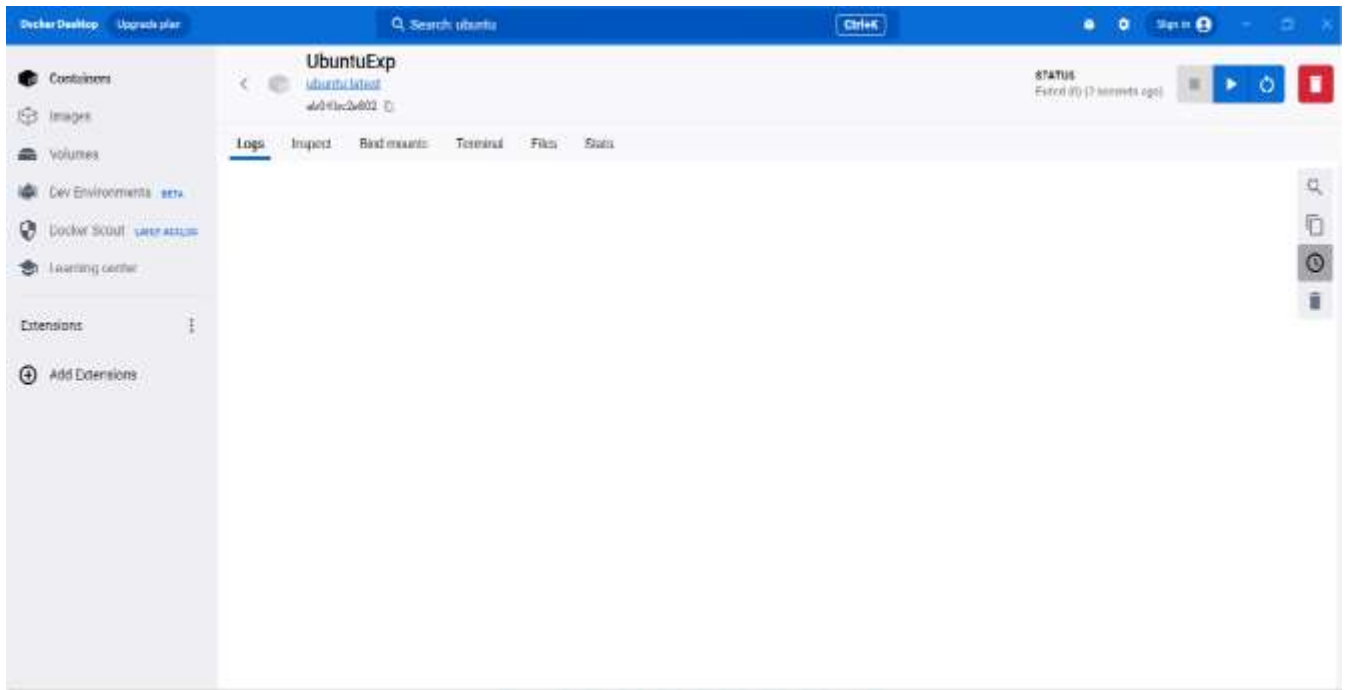
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Matan> wsl --list
Windows Subsystem for Linux has no installed distributions.
Distributions can be installed by visiting the Microsoft Store:
https://aka.ms/wslstore
PS C:\Users\Matan> wsl --list
Windows Subsystem for Linux Distributions:
docker-desktop (Default)
docker-desktop-data
PS C:\Users\Matan>
```







```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\kskmy> docker ps
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS   NAMES
1ce40ae4790d   ubuntu   "bash"    About a minute ago   Up About a minute   affectionate_wing
PS C:\Users\kskmy> |
```

OUTPUT:

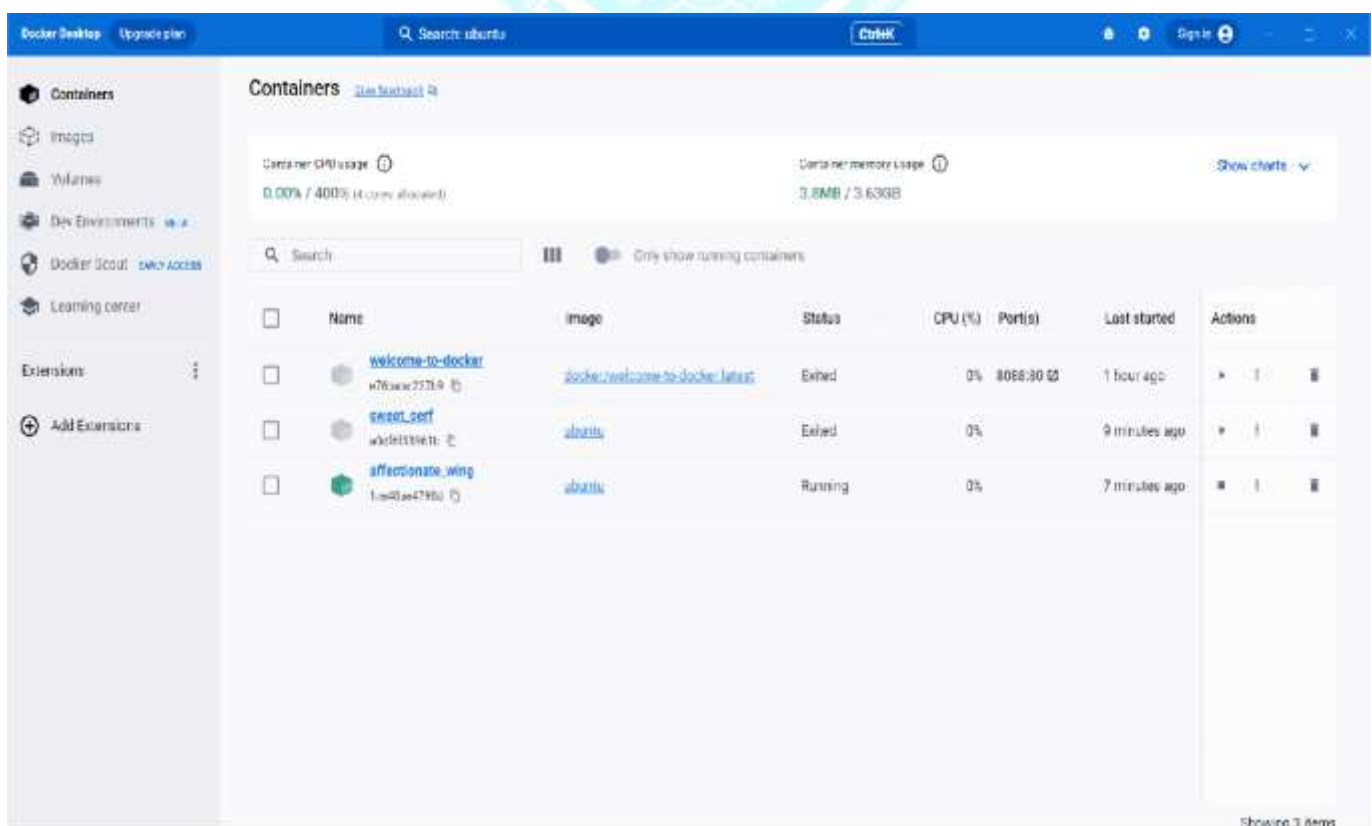
```

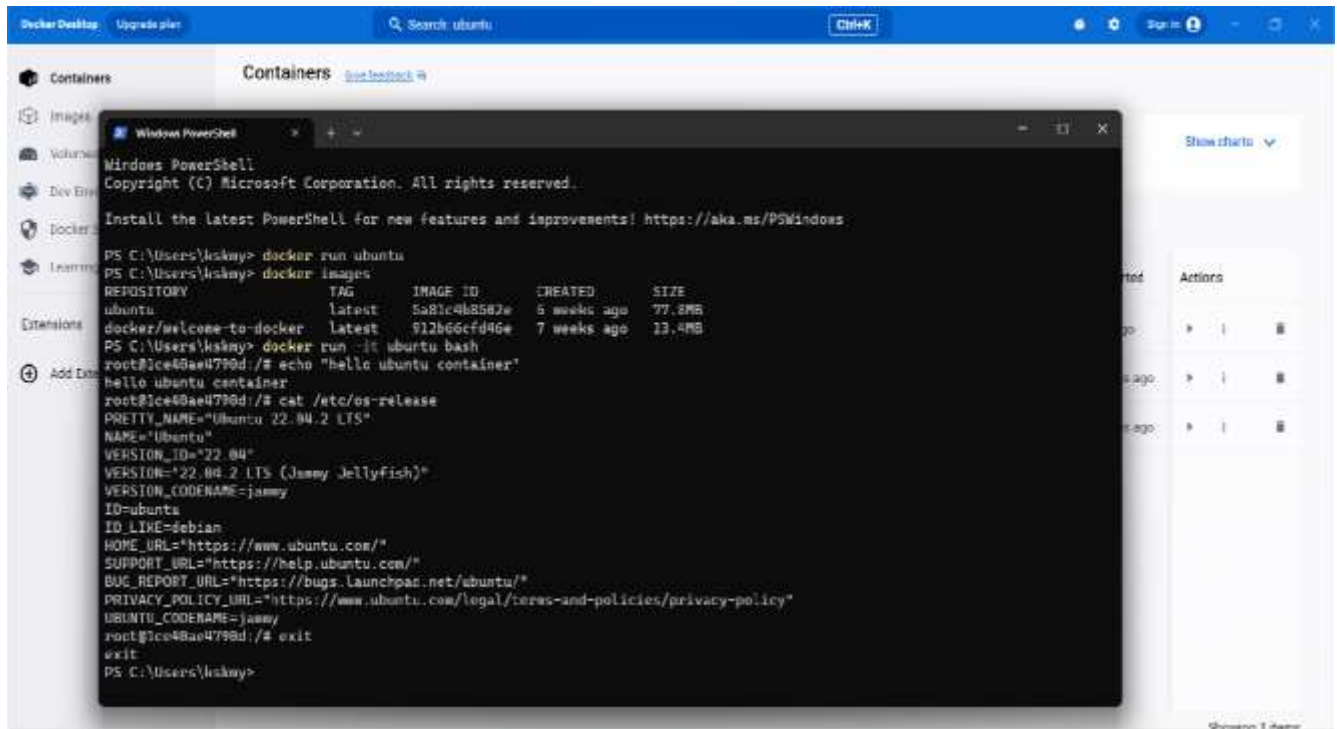
root@1ce40ae4790d: /
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\kskmy> docker run ubuntu
PS C:\Users\kskmy> docker images
REPOSITORY          TAG             IMAGE ID        CREATED         SIZE
ubuntu              latest          5a81c4b8502e   6 weeks ago    77.8MB
docker/welcome-to-docker latest          912b66cfd46e   7 weeks ago    13.4MB
PS C:\Users\kskmy> docker run -it ubuntu bash
root@1ce40ae4790d:/# echo "hello ubuntu container"
hello ubuntu container
root@1ce40ae4790d:/# cat /etc/os-release
PRETTY_NAME="Ubuntu 22.04.2 LTS"
NAME="Ubuntu"
VERSION_ID="22.04"
VERSION="22.04.2 LTS (Jammy Jellyfish)"
VERSION_CODENAME=jammy
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=jammy
root@1ce40ae4790d:/#

```





```
Windows PowerShell
Microsoft PowerShell
Copyright (c) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\jaskmy> docker run ubuntu
PS C:\Users\jaskmy> docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
ubuntu              latest             5a81c4b8587e       5 weeks ago        77.2MB
docker/welcome-to-docker Latest           912b66cf46e        7 weeks ago        13.4MB
PS C:\Users\jaskmy> docker run -it ubuntu bash
root@1ce48ae4798d:/# echo "hello ubuntu container"
hello ubuntu container
root@1ce48ae4798d:/# cat /etc/os-release
PRETTY_NAME="Ubuntu 22.04.2 LTS"
NAME="Ubuntu"
VERSION_ID="22.04"
VERSION="22.04.2 LTS (Jammy Jellyfish)"
VERSION_CODENAME=jammy
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=jammy
root@1ce48ae4798d:/# exit
exit
PS C:\Users\jaskmy>
```



RESULT:

Thus the given program has been executed and verified successfully.

Ex No: 9 RUN A CONTAINER FROM ADOCKER HUB**Date:****Aim:**

To run a container from a docker hub.

Procedure:**Step 1:** Search for the image.

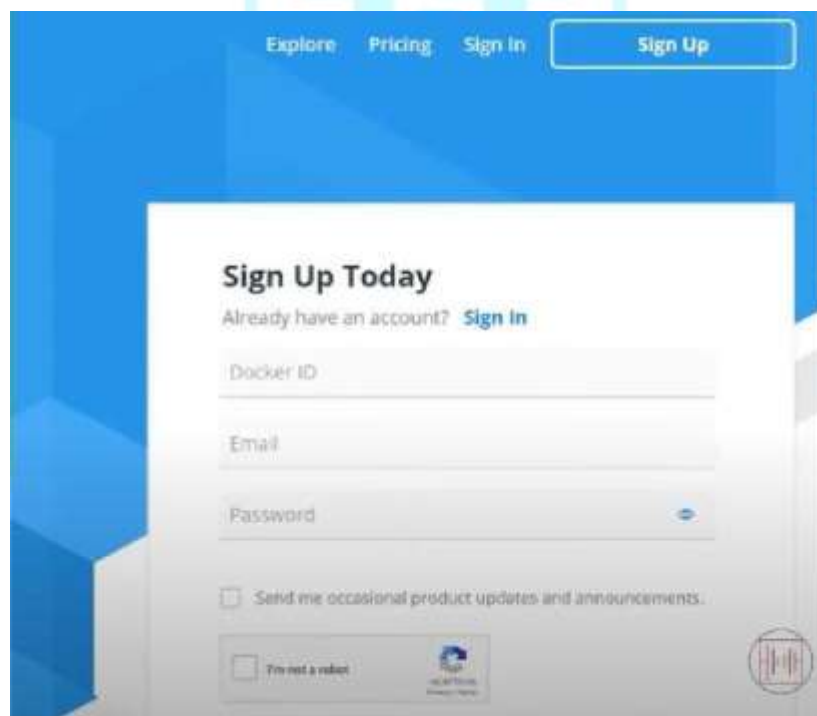
- You can search images by selecting the bar at the top, or by using the Ctrl + k shortcut.
- Search for Welcome-to-do-docker to find the image.

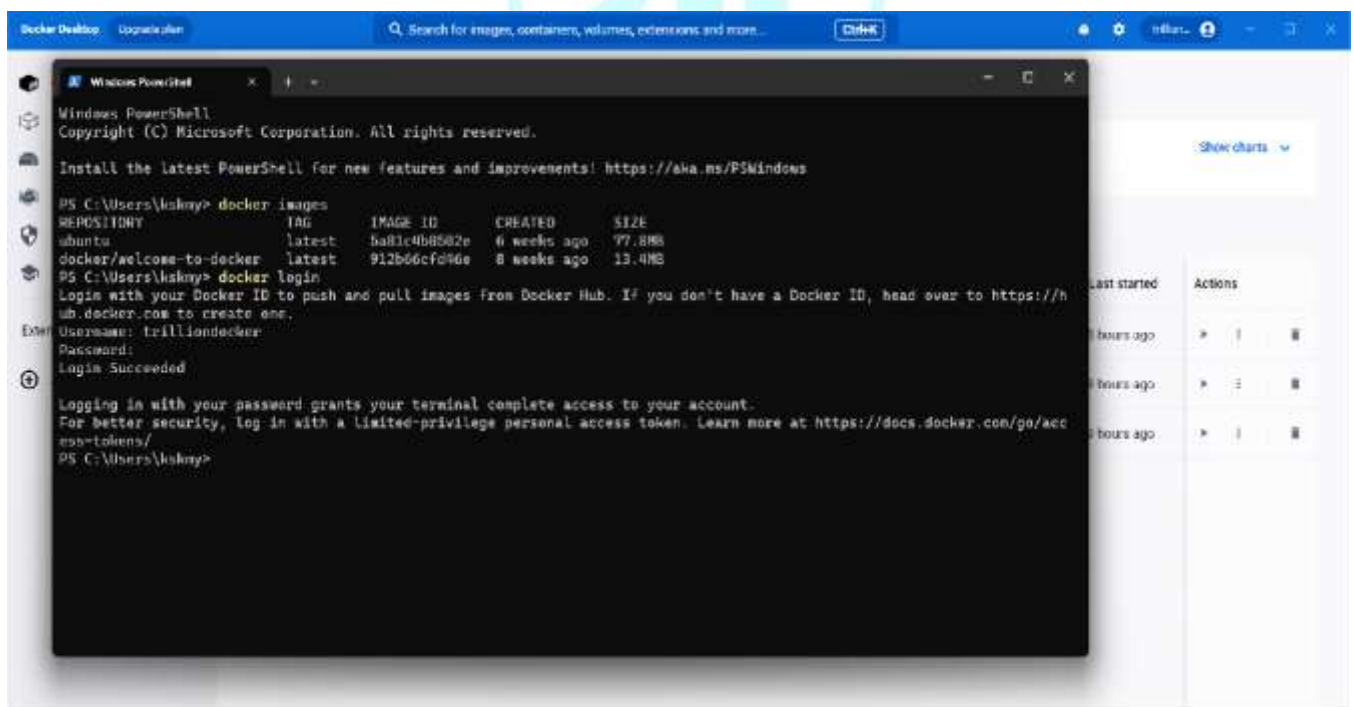
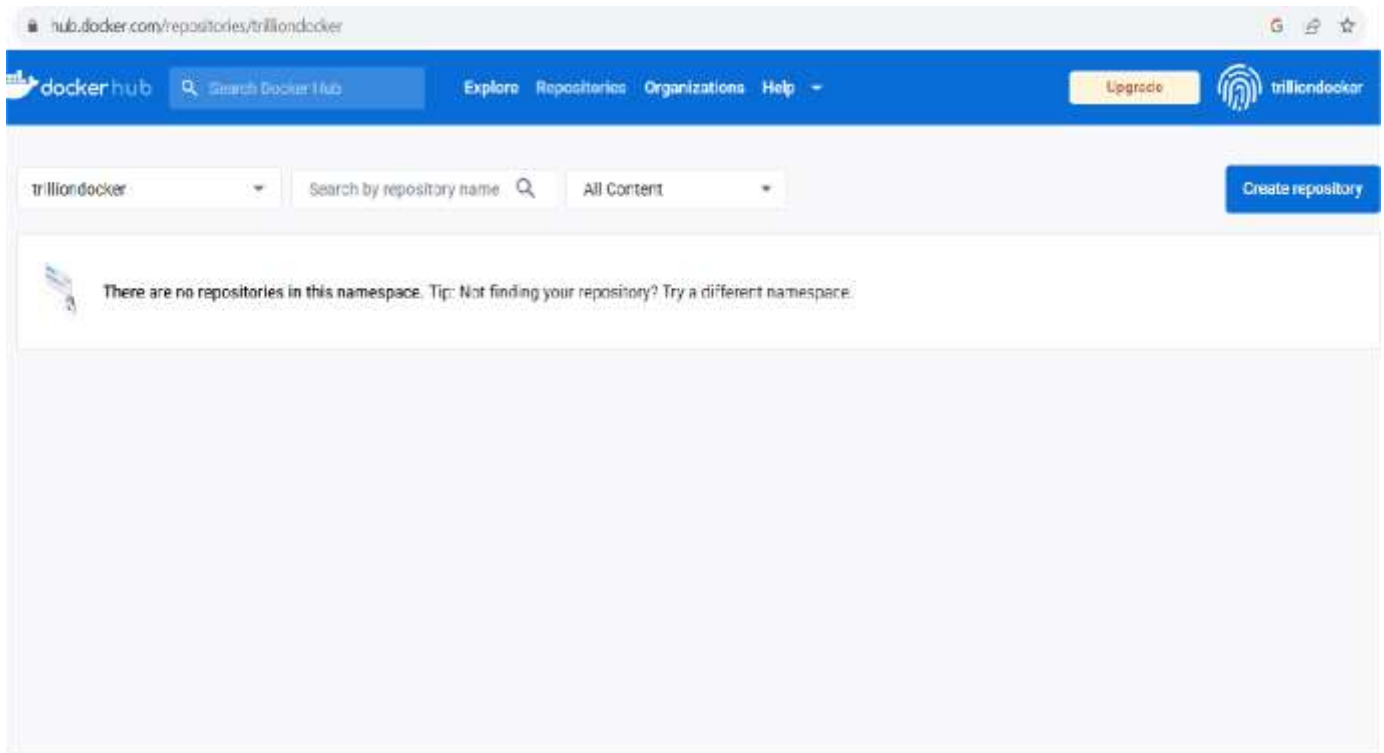
Step 2: Run the image.

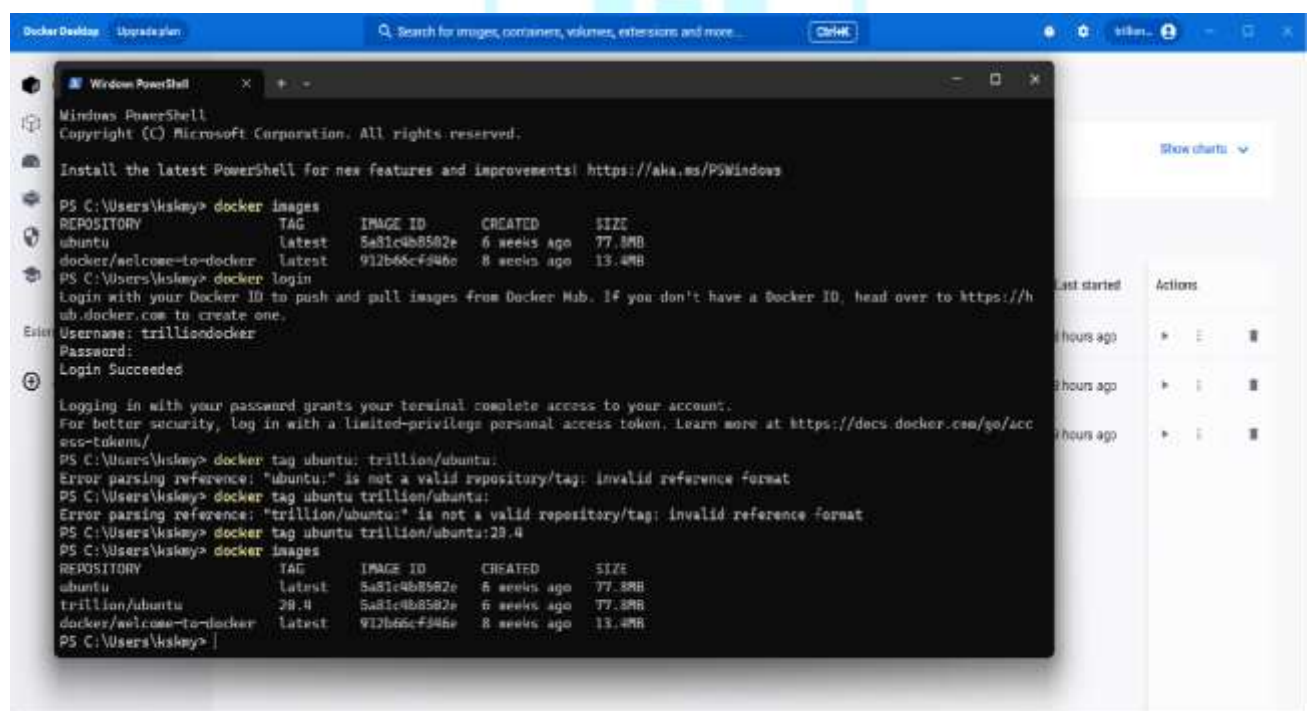
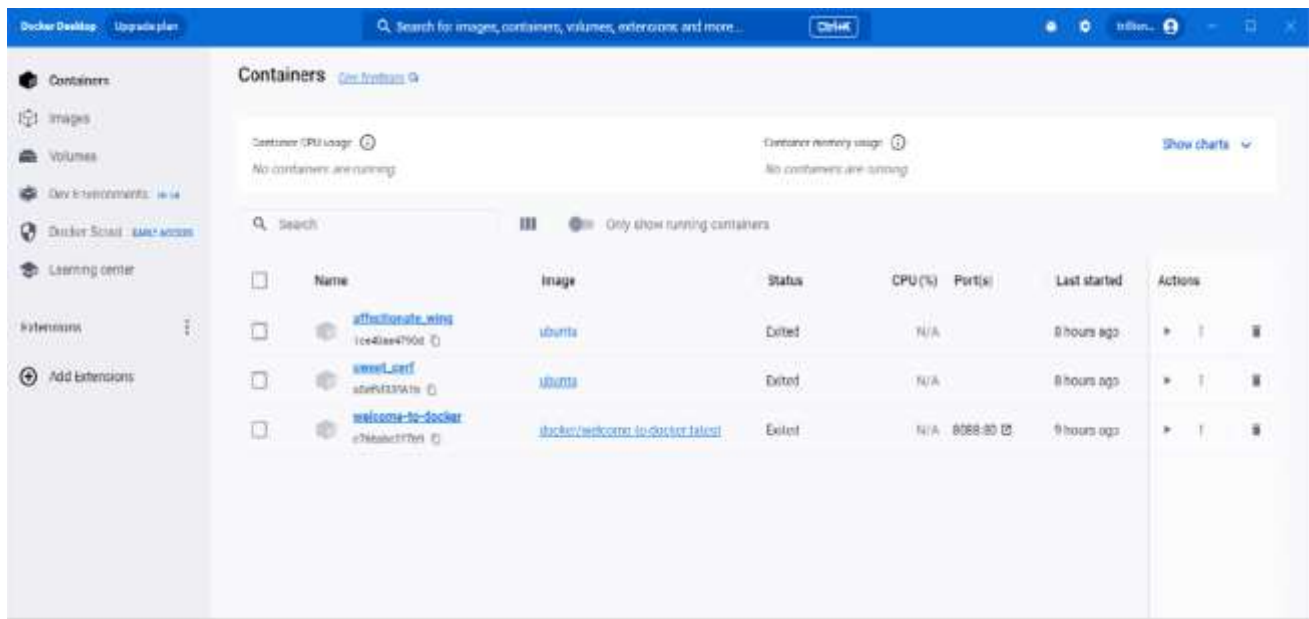
- Select run.
- When the optional setting appear, specify the host port number 8090 and select run.

Step 3: Explore the container.

- Go to the container tab in Docker Desktop to view the container.







The screenshot shows the Docker Desktop application. In the foreground, a Windows PowerShell terminal window displays the following commands and output:

```

PS C:\Users\kskmy> docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
ubuntu              latest             5a81c4b8582e       6 weeks ago        77.8MB
trillion/ubuntu      20.4              5a81c4b8582e       6 weeks ago        77.8MB
docker/welcome-to-docker latest            912b56cfd45e       8 weeks ago        13.4MB

PS C:\Users\kskmy> docker push trillion/ubuntu
Using default tag: latest
The push refers to repository [docker.io/trillion/ubuntu]
tag does not exist: trillion/ubuntu:latest

PS C:\Users\kskmy> docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
trillion/ubuntu      20.4              5a81c4b8582e       6 weeks ago        77.8MB
ubuntu              latest             5a81c4b8582e       6 weeks ago        77.8MB
docker/welcome-to-docker latest            912b56cfd45e       8 weeks ago        13.4MB

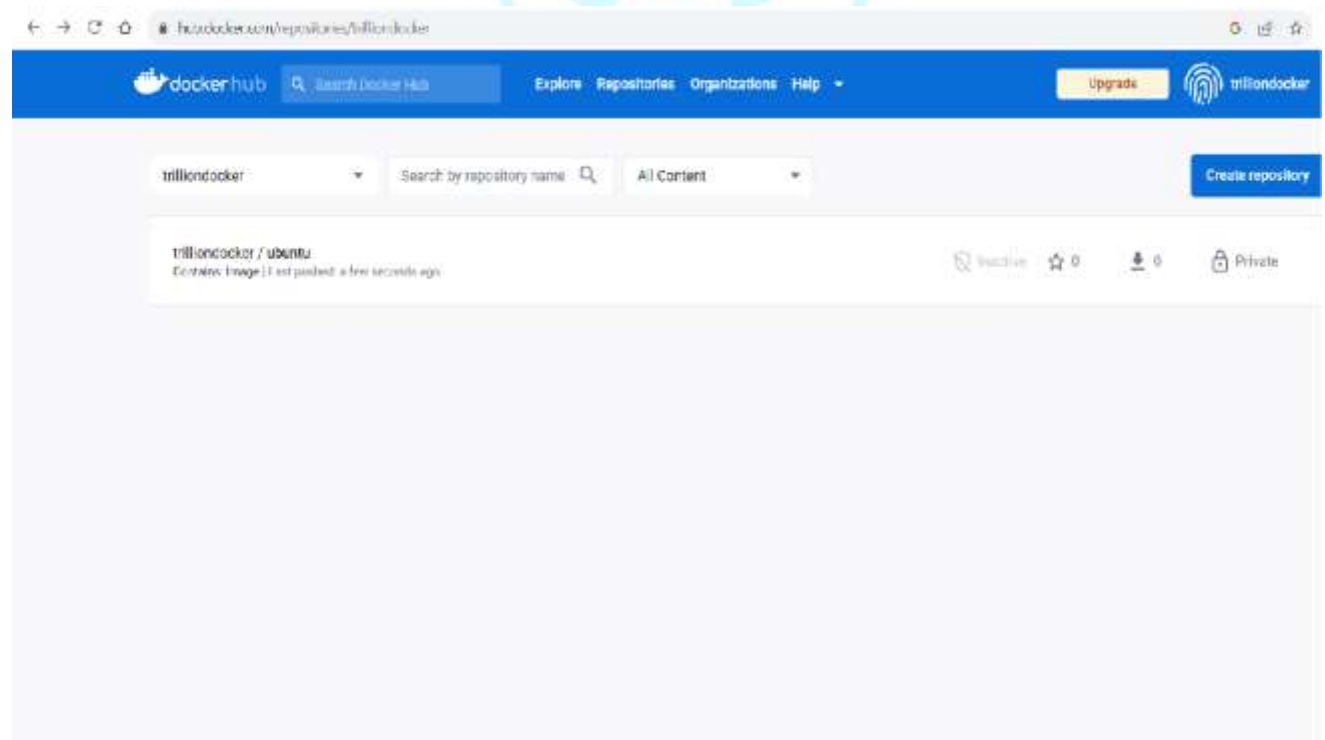
PS C:\Users\kskmy> docker tag ubuntu trilliondocker/ubuntu:20.4
PS C:\Users\kskmy> docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
ubuntu              latest             5a81c4b8582e       6 weeks ago        77.8MB
trilliondocker/ubuntu 20.4              5a81c4b8582e       6 weeks ago        77.8MB
docker/welcome-to-docker latest            912b56cfd45e       8 weeks ago        13.4MB

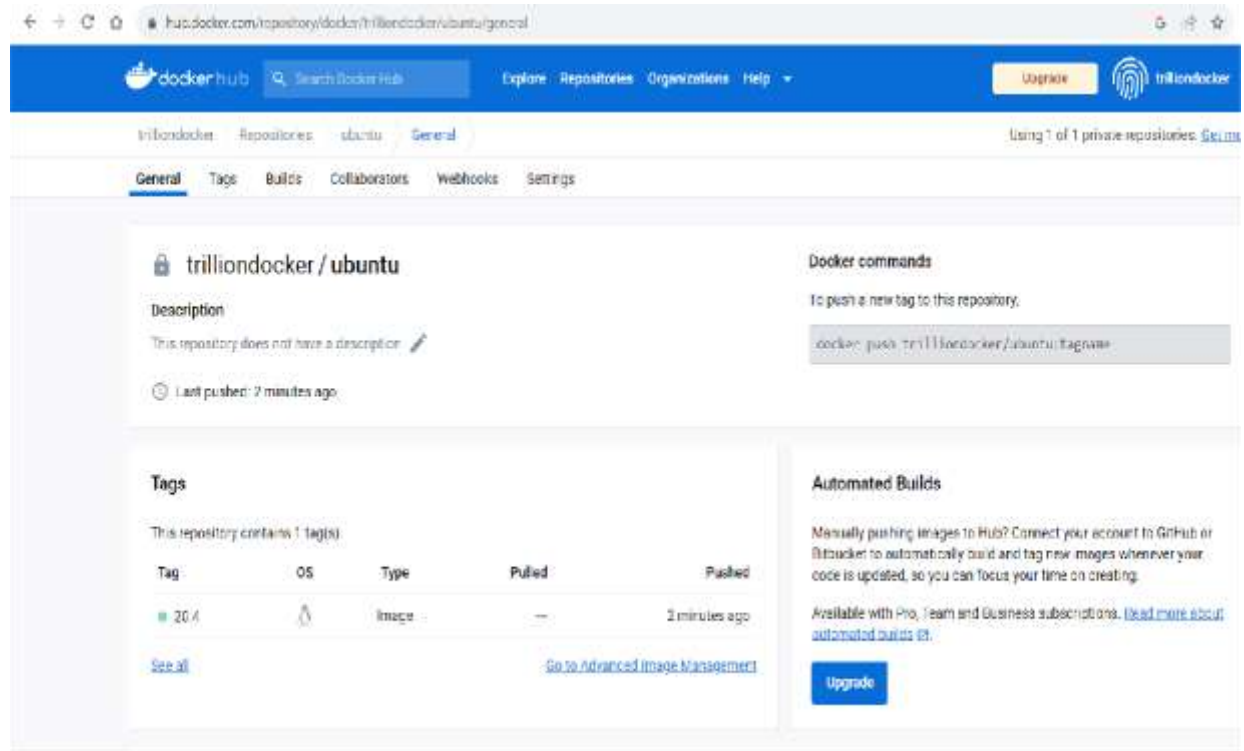
PS C:\Users\kskmy> docker push trilliondocker/ubuntu
Using default tag: latest
The push refers to repository [docker.io/trilliondocker/ubuntu]
tag does not exist: trilliondocker/ubuntu:latest

PS C:\Users\kskmy> docker push trilliondocker/ubuntu:20.4
The push refers to repository [docker.io/trilliondocker/ubuntu]
59c36aeelfb4: Pushed
20.4: digest: sha256:75399abc111a48bcabcf48c8fa5d1d44fb99e078ab449338d08f06cad34117e size: 529
PS C:\Users\kskmy>

```

In the background, the Docker Hub repository page for 'trilliondocker/ubuntu' is visible, showing the repository name, search bar, and repository details.





trilliondocker / ubuntu

Description
This repository does not have a description

Last pushed: 2 minutes ago

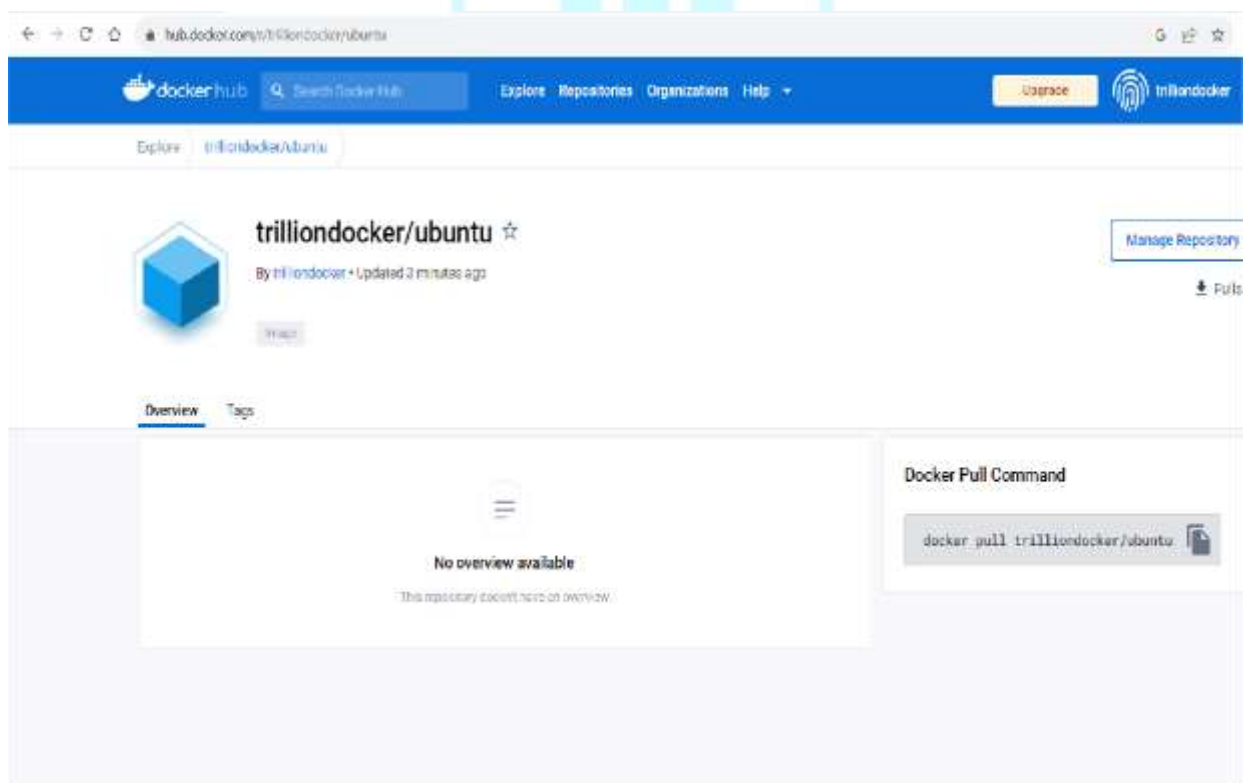
Tags
This repository contains 1 tag(s)

Tag	OS	Type	Pulled	Pushed
20.4	linux	image	—	2 minutes ago

[See all](#) [Go to Advanced Image Management](#)

Docker commands
To push a new tag to this repository:
`docker push trilliondocker/ubuntu:tagname`

Automated Builds
Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.
Available with Pro, Team and Business subscriptions. [Read more about automated builds](#) or [Upgrade](#)



trilliondocker / ubuntu

By trilliondocker • Updated 2 minutes ago

[Manage Repository](#)

[Pulls](#)

Overview Tags

No overview available
This repository does not have an overview

Docker Pull Command
`docker pull trilliondocker/ubuntu`

OUTPUT:

```

Docker Desktop Upgrade plan Search for images, containers, volumes, extensions and more... Ctrl+K
Windows PowerShell
20.4: digest: sha256:75399abc111a48bcabcfe08c8fa5d1d44f699e077ab449338c08f06dad34127e size: 529
PS C:\Users\kskay> docker pull trilliondocker/ubuntu
Using default tag: latest
Error response from daemon: manifest for trilliondocker/ubuntu:latest not found: manifest unknown: manifest unknown
PS C:\Users\kskay> docker login
Authenticating with existing credentials...
Login Succeeded

Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/access-tokens/
PS C:\Users\kskay> docker pull trilliondocker/ubuntu
Using default tag: latest
Error response from daemon: manifest for trilliondocker/ubuntu:latest not found: manifest unknown: manifest unknown
PS C:\Users\kskay> docker push trilliondocker/ubuntu:20.4
The push refers to repository [docker.io/trilliondocker/ubuntu]
59c56ae1fb4: Layer already exists
20.4: digest: sha256:75399abc111a48bcabcfe08c8fa5d1d44f699e077ab449338c08f06dad34127e size: 529
PS C:\Users\kskay> docker pull trilliondocker/ubuntu
Using default tag: latest
Error response from daemon: manifest for trilliondocker/ubuntu:latest not found: manifest unknown: manifest unknown
PS C:\Users\kskay> docker pull trilliondocker/ubuntu:20.4
20.4: Pulling from trilliondocker/ubuntu
Digest: sha256:75399abc111a48bcabcfe08c8fa5d1d44f699e077ab449338c08f06dad34127e
Status: Image is up to date for trilliondocker/ubuntu:20.4
docker.io/trilliondocker/ubuntu:20.4

What's Next?
View summary of image vulnerabilities and recommendations + docker scout quickview trilliondocker/ubuntu:20.4
PS C:\Users\kskay>

```

```

Docker Desktop Upgrade plan
Q Search for images, containers, volumes, extensions and more... Ctrl+K
Windows PowerShell
Login Succeeded
Logging in with your password grants your terminal complete access to your account.
For better security, log in with a limited-privilege personal access token. Learn more at https://docs.docker.com/go/access-tokens/
PS C:\Users\kskmy> docker pull trilliondoker/ubuntu
Using default tag: latest
Error response from daemon: manifest for trilliondoker/ubuntu:latest not found: manifest unknown: manifest unknown
PS C:\Users\kskmy> docker push trilliondoker/ubuntu:20.4
The push refers to repository [docker.io/trilliondoker/ubuntu]
59c56aee1fb4: Layer already exists
20.4: digest: sha256:75399abc111a40bcabcfe40c8fa5d1d44fb99e078ab449330d88f06dad34127e size: 529
PS C:\Users\kskmy> docker pull trilliondoker/ubuntu
Using default tag: latest
Error response from daemon: manifest for trilliondoker/ubuntu:latest not found: manifest unknown: manifest unknown
PS C:\Users\kskmy> docker pull trilliondoker/ubuntu:20.4
20.4: Pulling from trilliondoker/ubuntu
Digest: sha256:75399abc111a40bcabcfe40c8fa5d1d44fb99e078ab449330d88f06dad34127e
Status: Image is up to date for trilliondoker/ubuntu:20.4
docker.io/trilliondoker/ubuntu:20.4

What's Next?
View summary of image vulnerabilities and recommendations * docker scout quickview trilliondoker/ubuntu:20.4
PS C:\Users\kskmy> docker images
REPOSITORY          TAG         IMAGE ID      CREATED       SIZE
ubuntu              latest      5a81c4b0502e  6 weeks ago  77.0MB
trilliondoker/ubuntu 20.4       5a81c4b0502e  6 weeks ago  77.0MB
trilliondoker/ubuntu 20.4       5a81c4b0502e  6 weeks ago  77.0MB
docker/welcome-to-docker latest      912b66cfd46e  8 weeks ago  13.4MB
PS C:\Users\kskmy>

```



RESULT:

Thus the given program has been executed and verified successfully.

CONTENTS BEYOND SYLLABUS

Ex No: 10 FIND PROCEDURE TO RUN THE VIRTUAL MACHINE OF DIFFERENT CONFIGURATION. CHECK HOW MANY VIRTUAL MACHINES CAN BE UTILIZED AT PARTICULAR TIME

DATE:

OBJECTIVE:

To understand procedure to run the virtual machine of different configuration. Check how many Virtual machines can be utilized at particular time .

PROCEDURE:

KVM INSTALLTION

Check that your CPU supports hardware virtualization

To run KVM, you need a processor that supports hardware virtualization. Intel and AMD both have developed extensions for their processors, deemed respectively Intel VT-x (code name Vanderpool) and AMD-V (code name Pacifica). To see if your processor supports one of these, you can review the output from this command:

\$ egrep -c '(vmx|svm)' /proc/cpuinfo

If 0 it means that your CPU doesn't support hardware virtualization.

If 1 (or more) it does - but you still need to make sure that virtualization is enabled in the BIOS.

Use a 64 bit kernel (if possible)

Running a 64 bit kernel on the host operating system is recommended but not required.

To serve more than 2GB of RAM for your VMs, you must use a 64-bit kernel, On a 32-bit kernel install, you'll be limited to 2GB RAM at maximum for a given VM.

To see if your processor is 64-bit, you can run this command:

\$ egrep -c 'lm' /proc/cpuinfo

If 0 is printed, it means that your CPU is not 64-bit.

If 1 or higher, it is.

Now see if your running kernel is 64-bit, just issue the following command:

\$ uname -m

x86_64 indicates a running 64-bit kernel. If you use see i386, i486, i586 or i686, you're running a 32-bit kernel.

\$ ls /lib/modules/3.16.0-30-generic/kernel/arch/x86/kvm/kvm

kvm-amd.ko : AMD Processor

kvm-intel.ko : Intel Processor

kvm.ko : Kernel object

\$ ls /dev/kvm

/dev/kvm

Install Necessary Packages

```
1. qemu-kvm
2. libvirt-bin
3. bridge-utils
4. virt-manager
5. qemu-system
$ sudo apt-get install qemu-kvm
$ sudo apt-get install libvirt-bin
$ sudo apt-get install bridge-utils
$ sudo apt-get install virt-manager
$ sudo apt-get install qemu-system
```

To check package installation Check

```
$ dpkg --get-architecture qemu-kvm
$ virsh
virsh# exit
```

Verify Installation

You can test if your install has been successful with the following command:

```
$ virsh -c qemu:///system list
```

Id Name State

If on the other hand you get something like this:

```
$ virsh -c qemu:///system list
libvir: Remote error : Permission denied
error: failed to connect to the hypervisor
```

```
virsh # version
virsh # node info
```

Creating VMS

```
$ virt-install --connect qemu:///system -n hardy -r 512 -f hardy1.qcow2 -s 12 -c
ubuntu-14.04.2-server-amd64.iso --vnc --noautoconsole --os-type linux --os-variant
ubuntuHardy
```

(or)

Open disk image Error

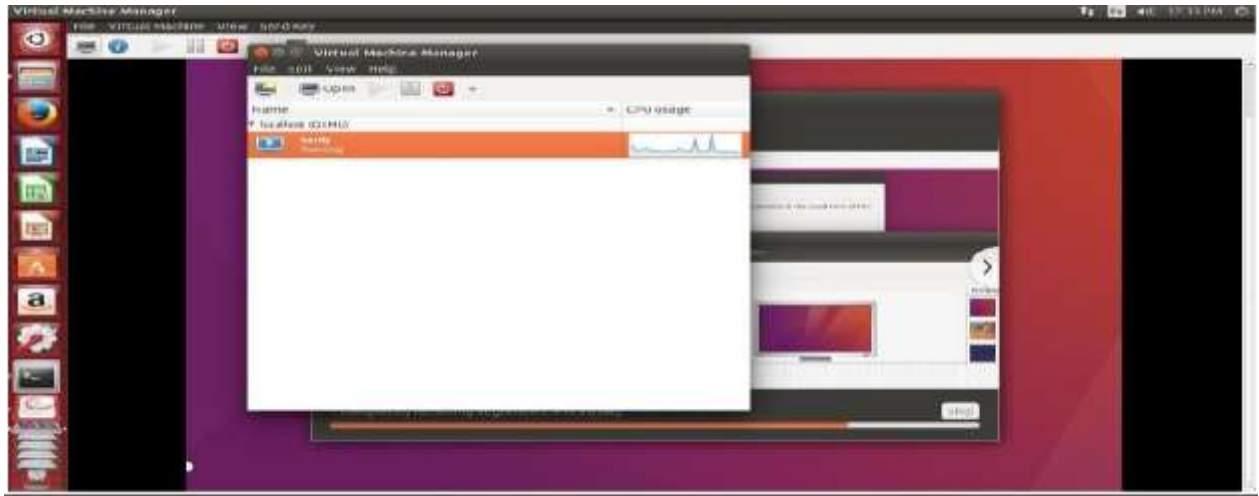
```
$ sudo chmod 777 hardy.qcow2
```

To run

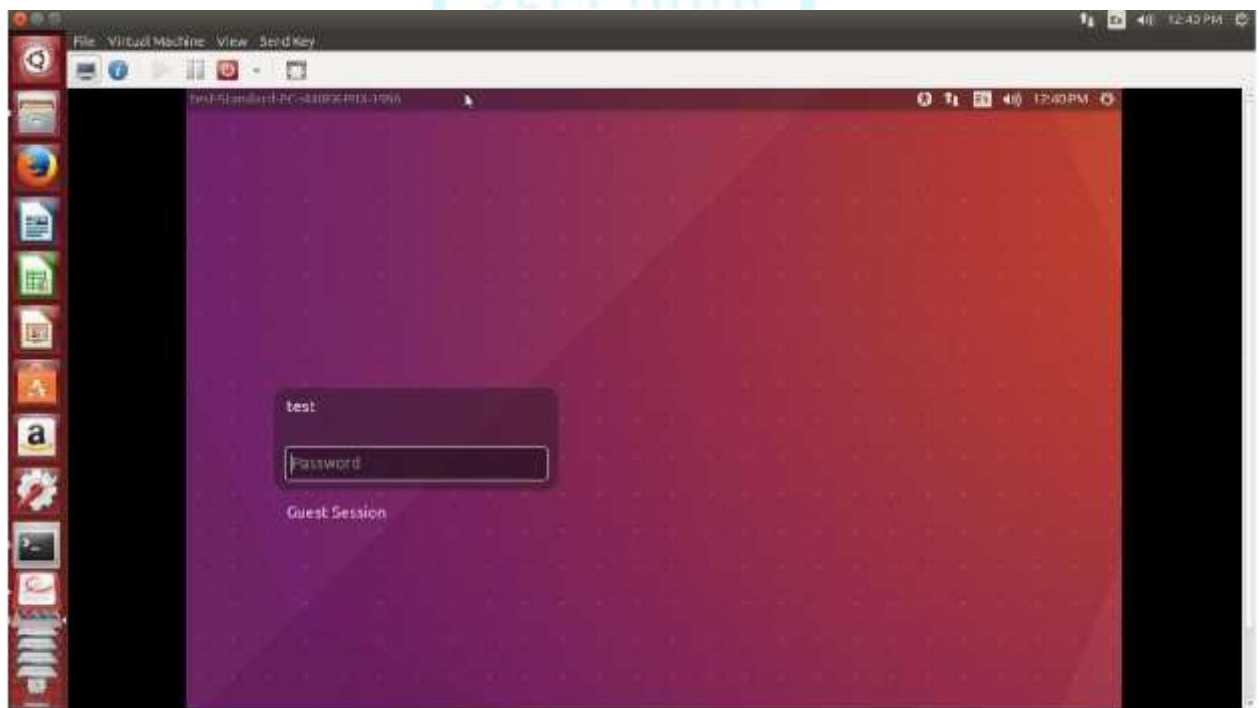
```
$ virt-install --connect qemu:///system -n hardy -r 512 -f hardy1.qcow2 -s 12 -c
ubuntu-14.04.2-server-amd64.iso --vnc --noautoconsole --os-type linux --os-variant
ubuntuHardy
```

```
$ sudo virt-manager
```

To Run Virtual Machine



To Login in Guest OS



TO RUN VM IN OPEN STACK

Step 1: Under the Project Tab, **Click Instances**. In the right side screen Click Launch Instance.

Step 2: In the details, Give the instance name(eg. Instance1).

Step 3: Click Instance Boot Source list and choose '**Boot from image**'

Step 4: Click Image name list and choose the image currently uploaded.

Step 5: Click launch.

Your VM will get created.

OPEN STACK INSTALLATION

```
$ free -m - its shows the RAM Usage
$ sudo adduser stack - password: stack
$ sudo apt-get install sudo - for updating SU
$ sudo -I - entering into the root user
# echo "stack ALL=(ALL) NOPASSWD: ALL">> /etc/sudoers - it changes the stack
from normal users to super user
#cat /etc/sudoers -To check last line as - (stack ALL =(ALL)
NOPASSWORD:ALL)
#exit - logout from root user
```

log off your machine and login as stack user

Prerequisite- to install git open all the ports

```
$ sudo apt-get install git
$ git clone https://git.openstack.org/openstack-dev/devstack
$ ls - devstack should there
$ cd devstack
$ ls - it shows all the files in devstack
$ls -l - longlisting the devstack files
$ nano local.conf - paste in local.conf
[[local|localrc]]
FLOATING_RANGE=192.168.1.224/27
FIXED_RANGE=10.11.12.0/24
FIXED_NETWORK_SIZE=256
FLAT_INTERFACE=eth0
ADMIN_PASSWORD=root
DATABASE_PASSWORD=root
RABBIT_PASSWORD= root
SERVICE_PASSWORD= root
SERVICE_TOKEN= root save and quit
$ ls -l - should be stack stack
$ ./stack.sh - if error number 181 comes go to vi stack.sh add next to UMASK 022
FORCE=yes
$ pwd
/home/stack/devstack
Devstack$ mv local.conf ../
$ cd ..
Stack$ rm -rf devstack/ /opt/stack/
$ git clone https://git.openstack.org/openstack-dev/devstack -b stable/kilo
$ cd devstack
$ ll
$ mv ../local.conf .
$ ls -l local.conf
$ ./stack.sh
$ nano stack.sh edit
#make sure unmask is sane
Add FORCE=yes save and exit
$ ./unstack.sh
$ ./clean.sh
```


EX.NO:11 MOUNT THE ONE NODE HADOOP CLUSTER USING FUSE

DATE:

AIM: To mount the one node HADOOP Cluster using FUSE.

PROCEDURE:

FUSE (Filesystem in Userspace) enables you to write a normal user application as a bridge for a traditional file system interface. The `hadoop-hdfs-fuse` package enables you to use your HDFS cluster as if it were a traditional file system on Linux. It is assumed that you have a working HDFS cluster and know the hostname and port that your Name Node exposes. To install `fuse-dfs` on Ubuntu systems:

\$sudo apt-get install hadoop-hdfs-fuse

To set up and test your mount point:

\$mkdir -p <mount_point>

hadoop-fuse-dfs dfs://<name_node_hostname>:<namenode_port><mount_point>

You can now run operations as if they are on your mount point. Press Ctrl+C to end the `fuse-dfs` program, and unmount the partition if it is still mounted.

Note: To find its configuration directory, `hadoop-fuse-dfs` uses the `HADOOP_CONF_DIR` configured at the time the mount command is invoked. If you are using SLES 11 with the Oracle JDK 6u26 package, `hadoop-fuse-dfs` may exit immediately because `ld.so` can't find `libjvm.so`. To work around this issue, add `/usr/java/latest/jre/lib/amd64/server` to the `LD_LIBRARY_PATH`.

To clean up your test:

\$ umount<mount_point>

You can now add a permanent HDFS mount which persists through reboots. To add a system mount:

R-2017 IV-CSE/ -07 SEM CLOUD COMPUTING LAB

1. Open `/etc/fstab` and add lines to the bottom similar to these:

```
hadoop-fuse-dfs#dfs://<name_node_hostname>:<namenode_port><mount_point> fuse
allow_other,usetrash, rw 2 0
```

For example:

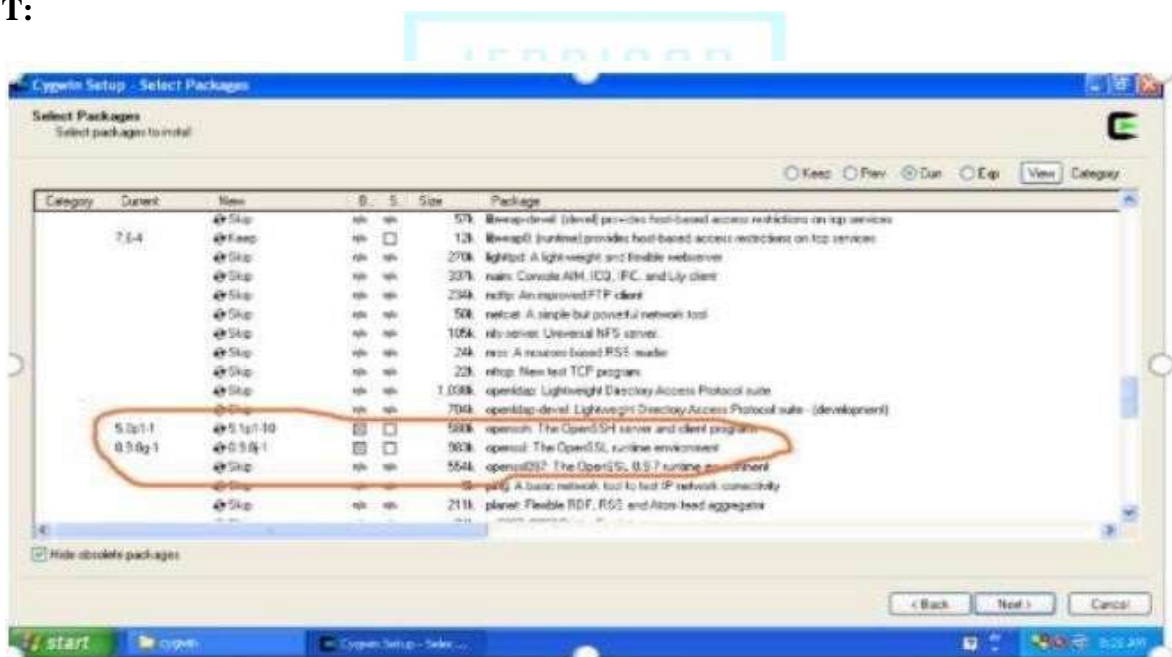
```
hadoop-fuse-dfs#dfs://localhost:8020 /mnt/hdfs fuse allow_other,usetrash,rw 2 0
```

2. Test to make sure everything is working properly:

```
$ mount <mount_point>
```

Your system is now configured to allow you to use the ls command and use that mount point as if it were a normal system disk.

OUTPUT:



RESULT:

Thus the given program has been executed and verified successfully.

Ex. No. 12 USE THE API'S OF HADOOP FOR INTERACTION WITH HADOOP**DATE:**

AIM: To write a program to use the API's of Hadoop to interact with it.

PROCEDURE:***Prerequisites:***

- You have set up a single-node "cluster" by following the [single-node setup](#)
- We assume that you run commands from inside the Hadoop directory.
- This program uses the Hadoop Streaming API to interact with Hadoop. This API allows you to write code in any language and use a simple text-based record format for the input and output <key, value>pairs.
- The output of this program will fetch the titles of web pages at particular URLs.

CODING:

- First we write a program to fetch titles from one or more web pages in Python:

```
// multifetch.py
#!/usr/bin/env python
import sys, urllib, re
title_re = re.compile("<title>(.*?)</title>",
                    re.MULTILINE | re.DOTALL | re.IGNORECASE)
for url in sys.argv[1:]:
    match =
    title_re.search(urllib.urlopen(url).read()) if
    match:
        print url, "\t", match.group(1).strip()
```

- To test, the file should be executable file. Todo this:

```
$ chmod u+x multifetch.py
```

Then execute it like this:

```
./fetch.py http://www.google.com http://www.brandeis.edu
```

OUTPUT (Sample):

```
http://www.jeppiaarinstitute.org   Jeppiaar Institute of
Technology http://www.annauniv.edu  Anna University
```

RESULT:

Thus the given program has been executed and verified successfully.

