



**GRACE COLLEGE OF
ENGINEERING**
(Approved by AICTE, New Delhi & Affiliated to ANNA University, Chennai)
MULLAKKADU, THOOTHUKUDI - 628 005

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

BE- Computer Science & Engineering

Anna University Regulation: 2017

CS8711- Cloud Computing Laboratory

IV Year/VII Semester

Lab Manual

Prepared By,

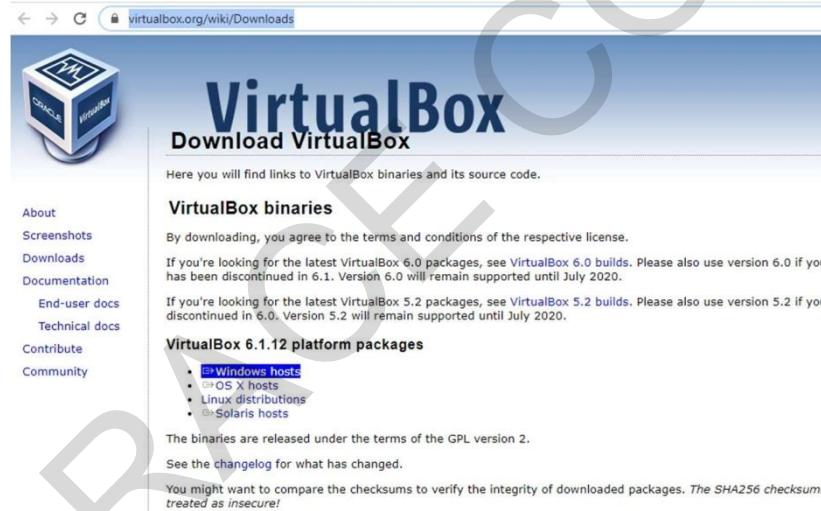
Ms. S. Abarna, AP/CSE

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

To install Virtualbox with Ubuntu OS on top of windows host operating systems.

PROCEDURE:**Steps to install VirtualBox:**

1. Download VirtualBox installer for windows.
2. The installer can be downloaded from the link
<https://www.virtualbox.org/wiki/Downloads>



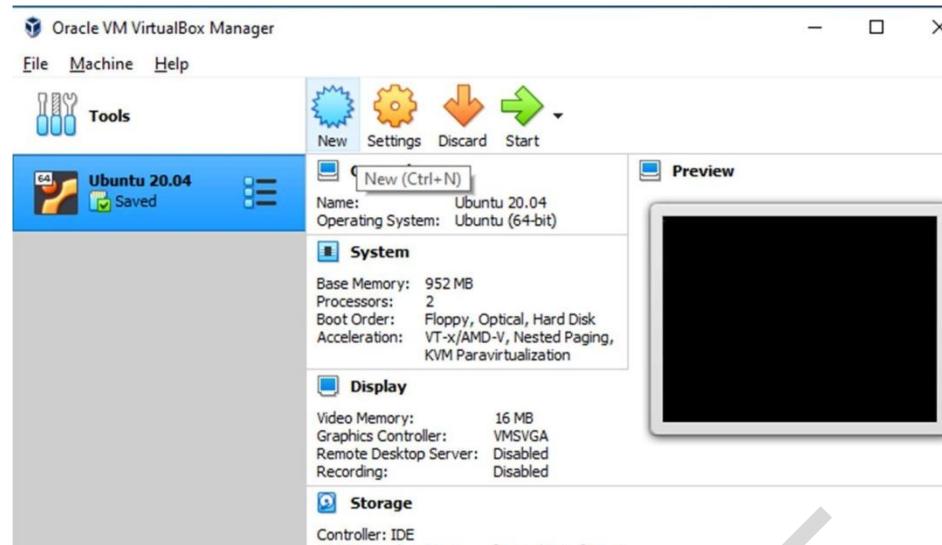
3. Click “Windows host” to download the binary version for windows host.
4. The installer file downloaded will have the file name format like VirtualBox-VersionNumber-BuildNumber-Win.exe.
 Example: VirtualBox-6.1.12-139181-Win.exe.
5. Double click on the installer to launch the setup Wizard. Click on Next to continue.



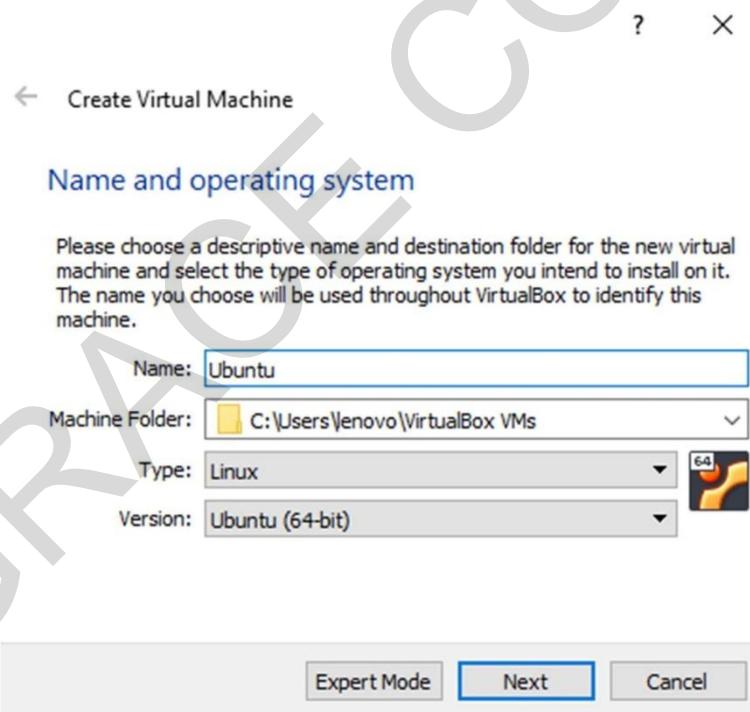
6. Custom setup dialog box will be opened. Accept the default settings and click next.
7. Select the way you want the features to be installed. You can accept the default and click next.
8. A dialog box opens with Network Interfaces warning. Click Yes to proceed.
9. Click install to begin the installation process.
10. When prompted with a message to install (Trust) Oracle Universal Serial Bus, click Install to continue.
11. After the installation completes, click finish to exit the setup wizard.
12. Launch the Oracle VM VirtualBox.

Steps to create a virtual machine [Ubuntu] in VirtualBox:

1. Open the Oracle VM VirtualBox.
2. Click New icon or 'Ctrl + N' to create a new virtual machine.

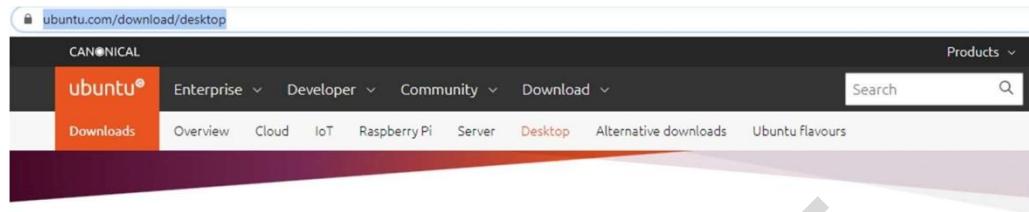


3. Enter a name for the new virtual machine. Choose the Type and Version. Note that VirtualBox automatically changes 'Type' to Linux and 'Version' to 'Ubuntu (64 bit)' if the name is given as 'Ubuntu'. Click Next.



4. Select the amount of RAM to use. The ideal amount of RAM will automatically be selected. Do not increase the RAM into the red section of the slider; keep the slider in the green section.
5. Accept the default 'Create a virtual hard drive now' and click 'Create' button.
6. Choose the hard disk file type as VDI (VirtualBox Disk Image). Click Next.

7. Click Next to accept the default option ‘Dynamically allocated’ for storage on physical hard drive.
8. Select the size of the virtual hard disk and click create.
9. The newly created virtual machine will be displayed in the dashboard.
10. Download the ISO file [Ubuntu disk image file]. Latest version of Ubuntu iso file can be downloaded from the link <https://ubuntu.com/download/desktop>. Click Download button.



Download Ubuntu Desktop

Ubuntu 20.04.1 LTS

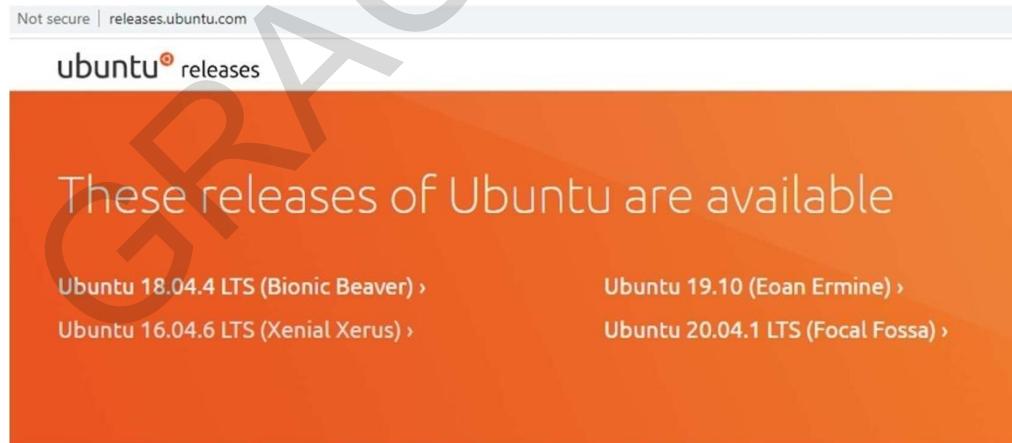
Download the latest LTS version of Ubuntu, for desktop PCs and laptops. LTS stands for long-term support — which means five years, until April 2025, of free security and maintenance updates, guaranteed.

[Ubuntu 20.04 LTS release notes](#)

[Download](#)

For other versions of Ubuntu Desktop including torrents, the network installer, a list of local mirrors,

11. For previous versions, goto <http://releases.ubuntu.com>. Choose the preferred version of Ubuntu and download the iso file.

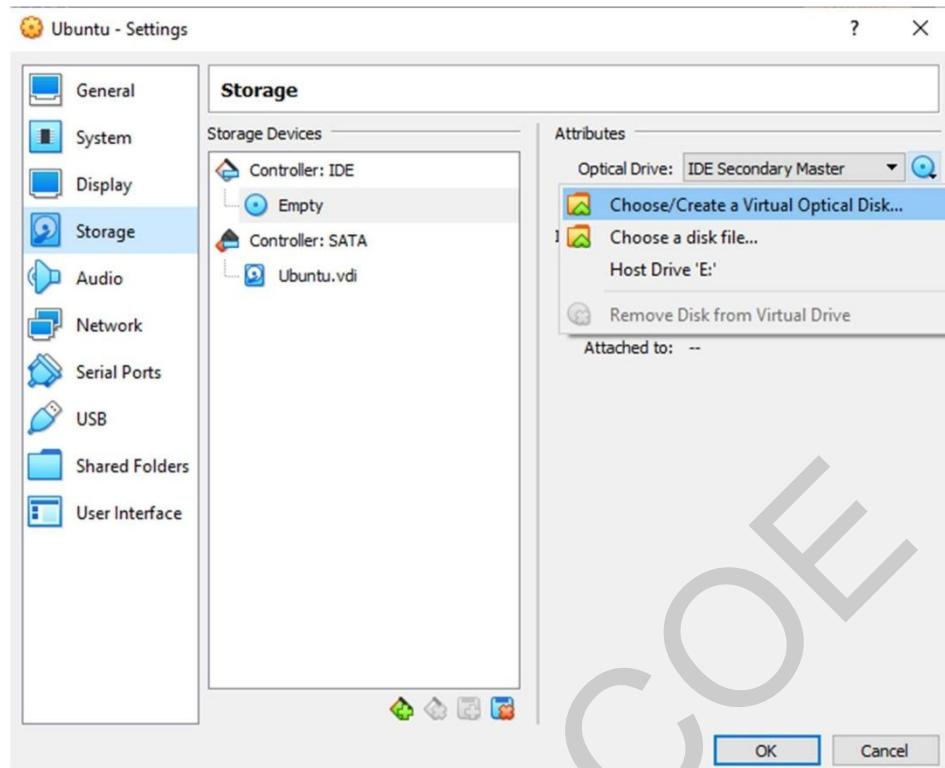


These older Ubuntu releases are now in Extended Maintenance (ESM):

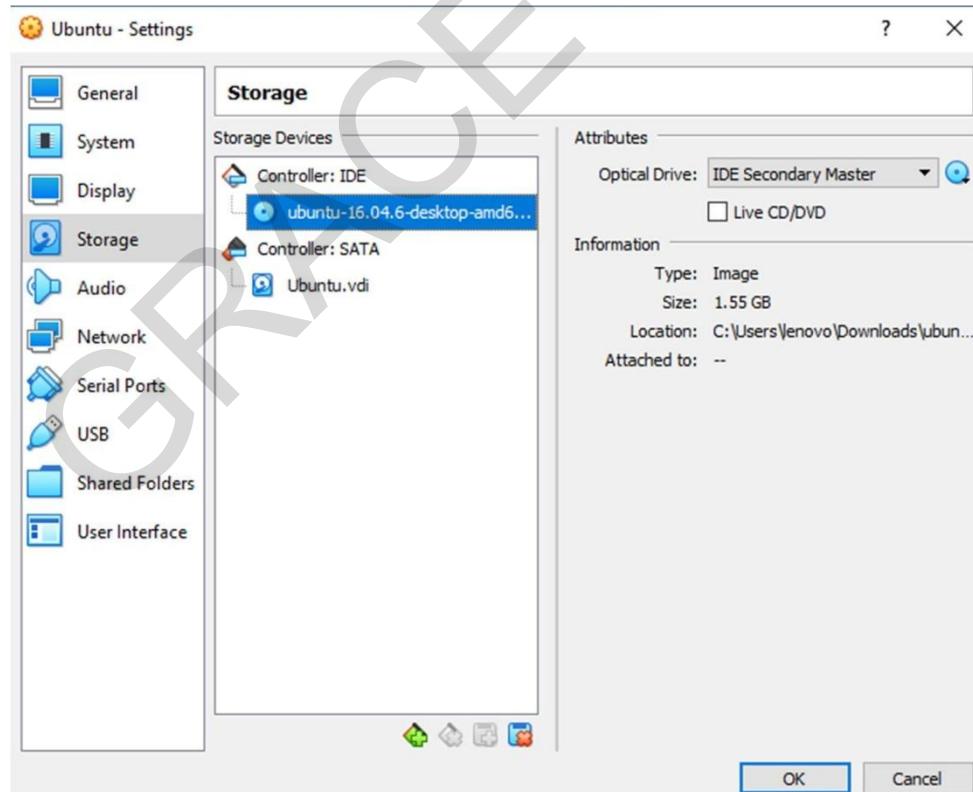
- [Ubuntu 14.04.6 LTS \(Trusty Tahr\) >](#)
- [Ubuntu 12.04.5 LTS \(Precise Pangolin\) >](#)

12. To setup the Ubuntu disk image file (iso file) goto settings.
13. Click Storage. Under ‘Storage Devices’ section click ‘Empty’.

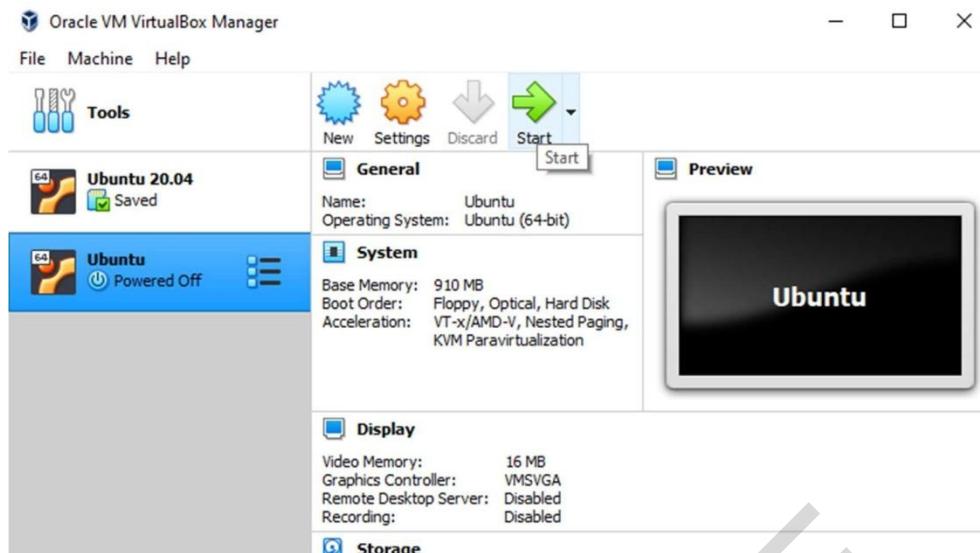
14. In Attributes section, click the disk image and then "Choose Virtual Optical Disk File".



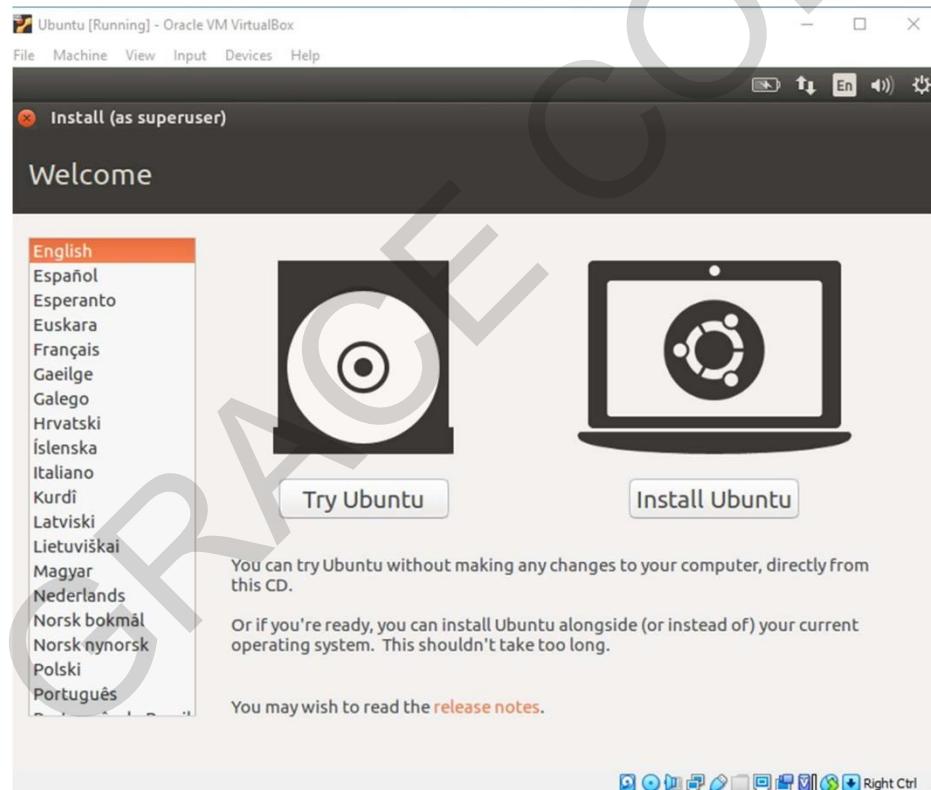
15. Browse and select the downloaded iso file. Click ok.



16. Select the newly created virtual machine in the dashboard and click start button.



17. In the welcome screen, click 'Install Ubuntu' button.



18. Click 'Continue' button.

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

20. Choose the default and click continue.

21. Setup up your profile by creating username and password.

22. After installation is complete, click 'Restart Now' button and follow the instructions.

23. The Ubuntu OS is ready to use. Login with the username and password.

OUTPUT 1:

Virtualbox on top of windows.



OUTPUT 2:

Installation of Virtual box with **Linux OS (Guest OS/VM)** on top of windows Host.



RESULT:

The Virtualbox installation is completed and the Virtual machine is created on top of windows host operating system.

EX.NO:2	INSTALL A C COMPILER IN THE VIRTUAL MACHINE
----------------	--

AIM:

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

PROCEDURE:

1. Launch the virtual box and open the virtual machine (Ubuntu).
2. Run the following command in the virtual machine terminal.

```
$ sudo apt update
```

```
$ sudo apt install gcc
```

It will install all the necessary packages for gcc complier.

3. Type the C program in the text editor and save the file with .c extension.

```
//demo.c
```

```
#include<stdio.h> int  
main()  
{  
printf("Hello World"); return  
0;  
}
```

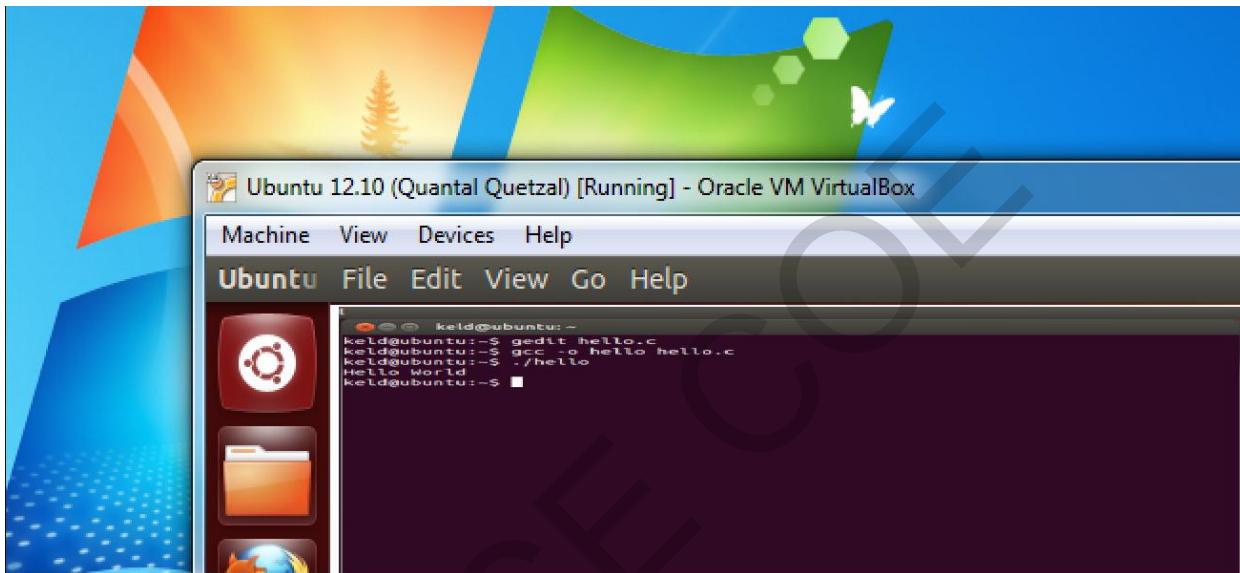
4. Compile and Run the C Program

```
cc demo.c
```

```
./a.out
```

OUTPUT:

Installation of a C compiler in the virtual machine and executing a sample program



RESULT:

Thus a C compiler is installed on a Ubuntu Virtual Machine on top of Windows Host and executed a C program on a virtual machine.

EX.NO :3

INSTALL GOOGLE APP ENGINE AND CREATE A WEB APPLICATIONS USING JAVA

AIM:

To install Google App Engine and to create *hello world* app and other simple web applications using java.

PROCEDURE:**Google App Engine SDK Installation:**

1. Download the Google Cloud SDK installer using the link
<https://cloud.google.com/appengine/downloads>.
2. Select the standard environment as Java.

The image contains two screenshots of the Google Cloud Platform website, illustrating the steps to install the Google App Engine SDK for Java.

Screenshot 1: Install an SDK for App Engine

- The URL is <https://cloud.google.com/appengine/downloads>.
- The page title is "Install an SDK for App Engine".
- The sidebar shows "App Engine" selected under "Serverless computing".
- The main content area says "Set up your computer for developing, deploying, and managing your apps in App Engine." It notes that instructions differ by environment (Standard or Flexible).
- A section titled "Standard environment" lists supported languages: Python, Java, Node.js, PHP, and Go.

Screenshot 2: Setting Up Your Development Environment

- The URL is <https://cloud.google.com/appengine/docs/standard/java11/setting-up-environment>.
- The page title is "Setting Up Your Development Environment".
- The sidebar shows "Java 11 Standard Environment" selected under "Serverless computing".
- The main content area says "To set up your environment for developing on Java 11:"

 1. Download and install the latest release of Java 11. (Link: [See Java 11 Runtime Environment](#))
 2. Download, install, and then initialize the Cloud SDK. (Link: [Download and Install the Cloud SDK](#))

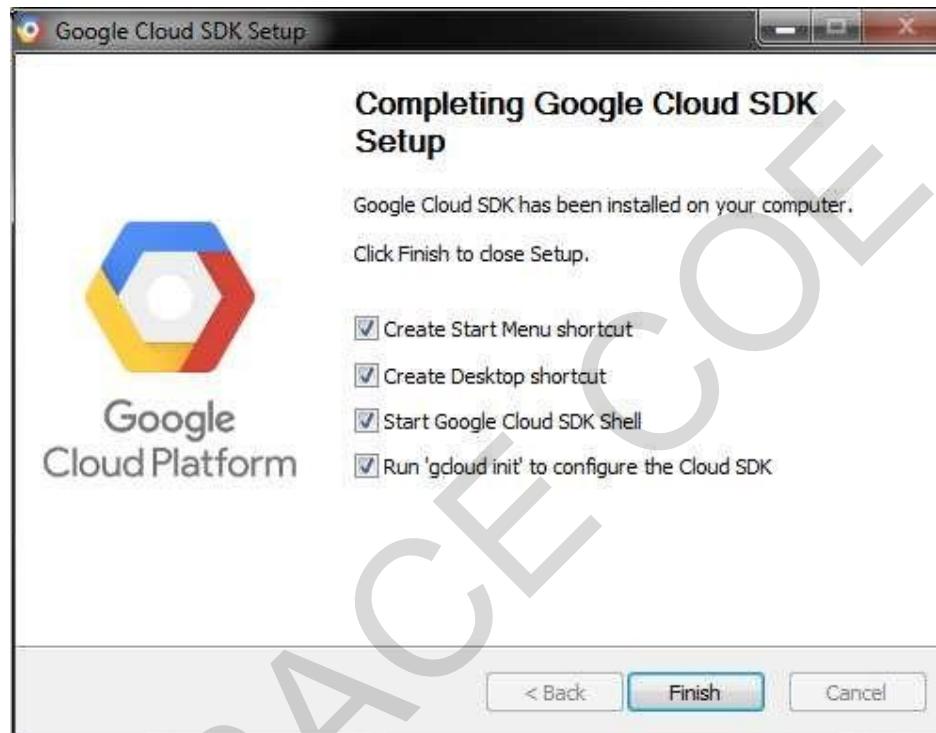
Additional notes: "The Cloud SDK provides you the `gcloud` command-line tooling for deploying and managing your apps." and "By downloading, you agree to be bound by the [Terms](#) that govern use of the Cloud SDK for App Engine."

3. Click ‘Download and Install the Cloud SDK’. Launch the installer and follow the prompts.
4. After installation has completed, the installer presents several options:

Make sure that the following are selected:

- Start Google Cloud SDK Shell
- Run **‘gcloud init’**

The installer then starts a terminal window and runs the **gcloud init** command.

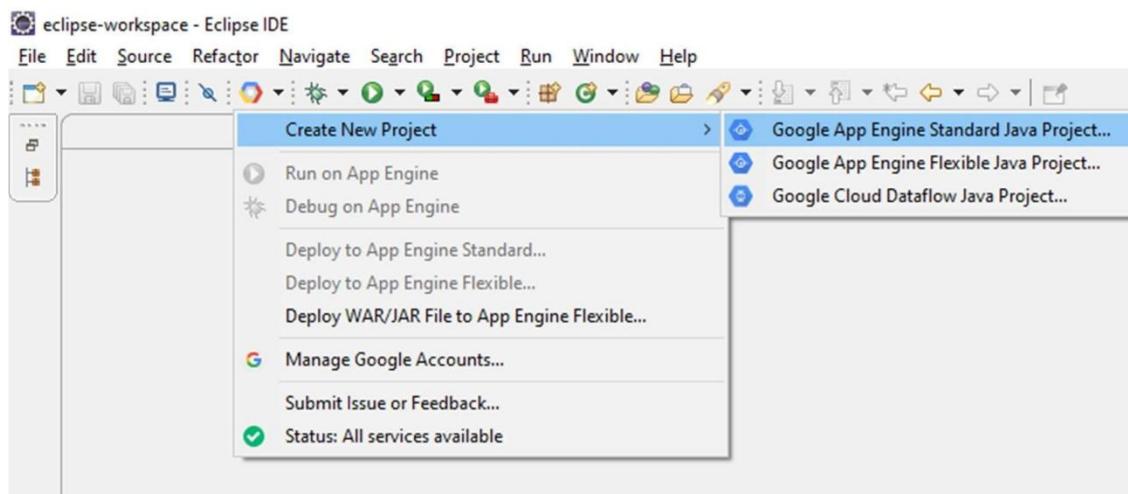


5. Run the following command in your terminal to install the gcloud component that includes the App Engine extension for Java11:

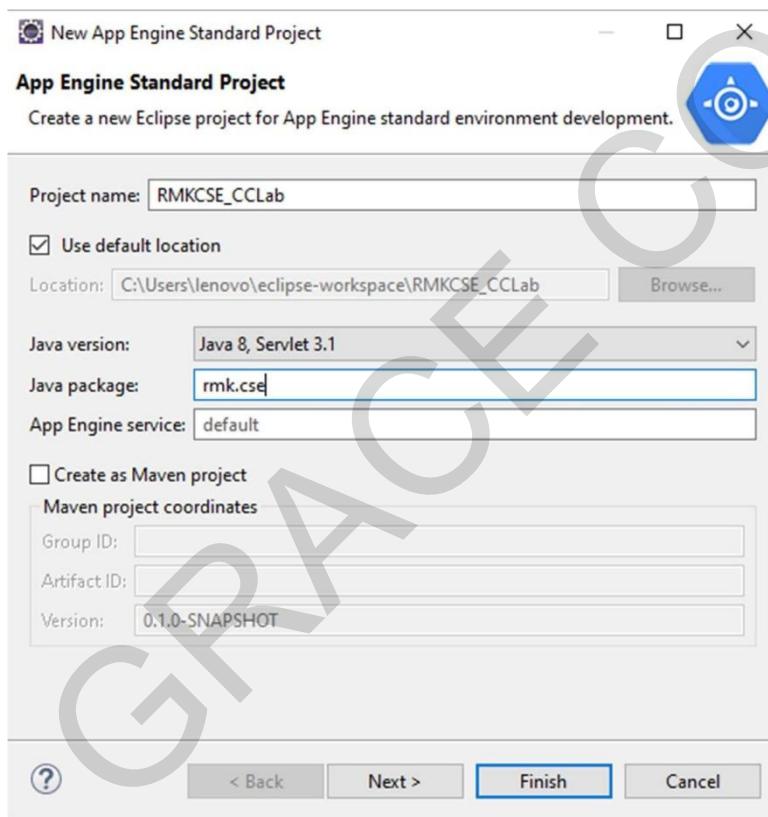
gcloud components install app-engine-java

Creating a new App Engine standard project in Eclipse:

6. Eclipse with the cloud tools is used to create App Engine standard project.
7. To install the Cloud Tools in Eclipse, select Help > Eclipse Marketplace... and search for ‘Google Cloud Tools for Eclipse’ and click install.
8. After installation restart eclipse when prompted.
9. Click the **Google Cloud Platform** toolbar button.
10. Select **Create New Project > Google App Engine Standard Java Project**.



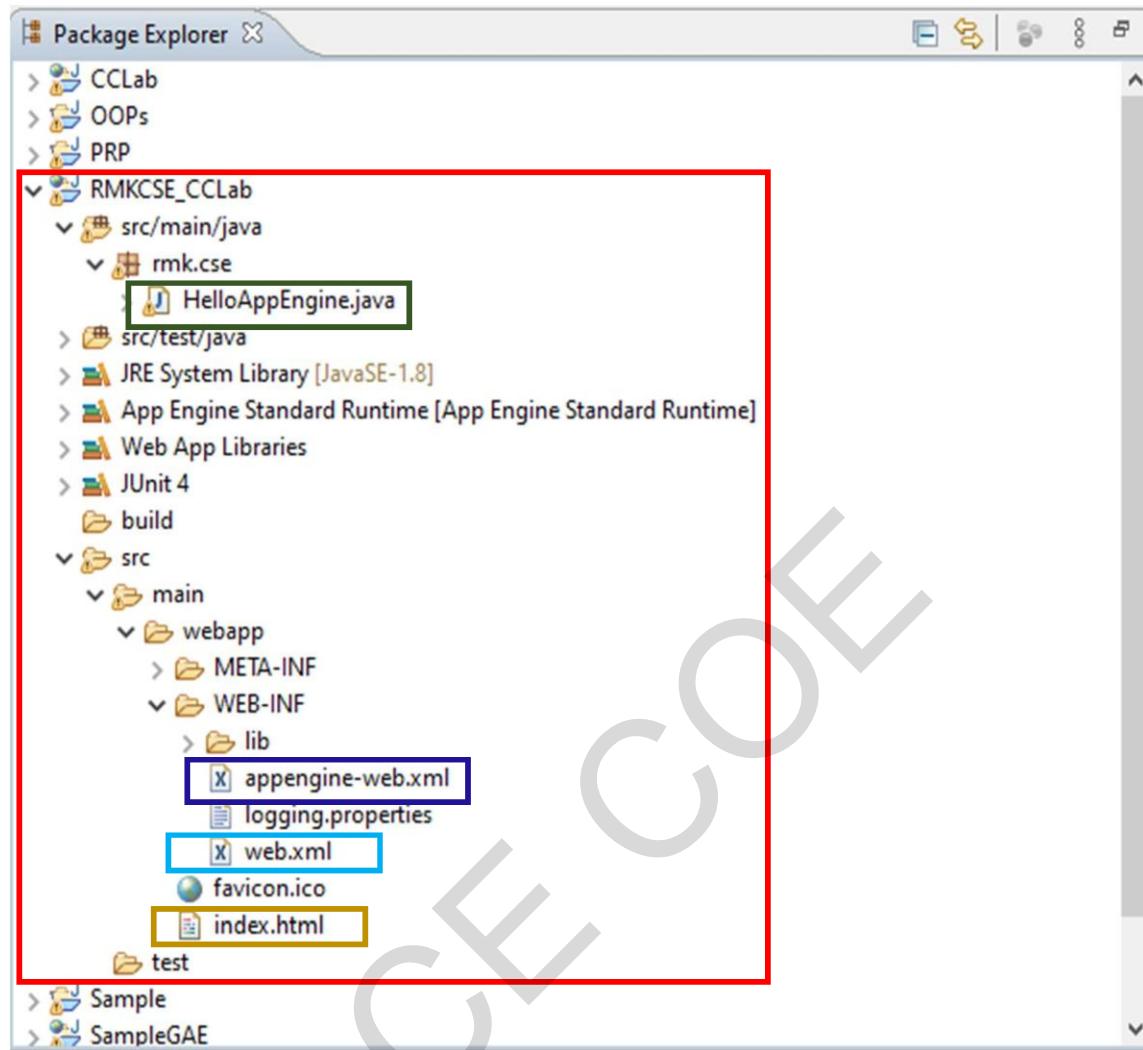
11. Enter the project name and packagename.



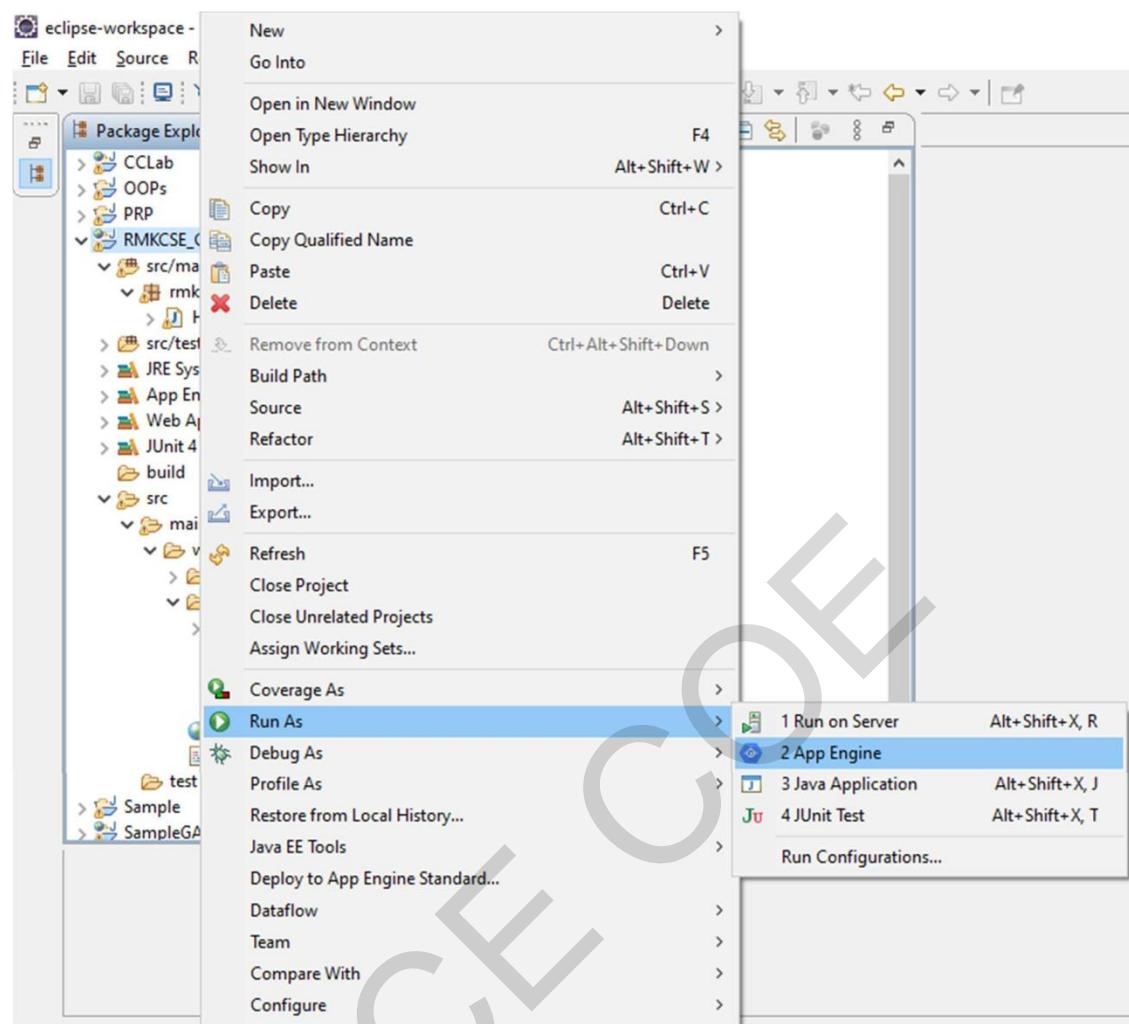
12. Click Next. Select the libraries required for the project.

13. Click Finish.

14. The wizard generates a native Eclipse project, with a simple servlet, that you can run and deploy from the IDE.



15. App Engine Java applications use the Java Servlet API to interact with the web server. Modify the defaultHelloAppEngine.java file with your application code.
16. **appengine-web.xml** is a Google App Engine specific configuration file.
17. **web.xml** is a standard web application configuration file.
18. Right click the project in the Package Explorer, select Run As > AppEngine.



19. Eclipse opens its internal web browser to your application. You can also open an external browser and navigate to <http://localhost:8080>. Either way, you'll see a static HTML page with a link to the servlet.

```
Aug 15, 2020 11:24:11 AM com.google.appengine.tools.development.AbstractModule startup
INFO: Module instance default is running at http://localhost:8080/
Aug 15, 2020 11:24:11 AM com.google.appengine.tools.development.AbstractModule startup
INFO: The admin console is running at http://localhost:8080/\_ah/admin
Aug 15, 2020 4:54:11 PM com.google.appengine.tools.development.DevAppServerImpl doStart
INFO: Dev App Server is now running
```

```

//HelloAppEngine.java

package rmk.cse;

import java.io.IOException;

import javax.servlet.annotation.WebServlet; import javax.servlet.http.HttpServlet; import
javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

@WebServlet(
name = "HelloAppEngine", urlPatterns = {" /hello" }
)
public classHelloAppEngine extends HttpServlet {

@Override
public voiddoGet(HttpServletRequest request, HttpServletResponse response)
throws IOException {

response.setContentType("text/plain"); response.setCharacterEncoding("UTF-8");

response.getWriter().print("Hello App Engine!\r\n");

}
}

//appengine-web.xml

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

<threadsafe>true</threadsafe>
<sessions-enabled>false</sessions-enabled>
<runtime>java8</runtime>

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

</appengine-web-app>

//web.xml

<?xml version="1.0" encoding="utf-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"
version="3.1">
<welcome-file-list>
<welcome-file>index.html</welcome-file>

<welcome-file>index.jsp</welcome-file>

```

```
</welcome-file-list>

</web-app>

//index.html

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" lang="en">
<head>
<meta http-equiv="content-type" content="application/xhtml+xml; charset=UTF-8"
/>
<title>Hello App Engine</title>
</head>

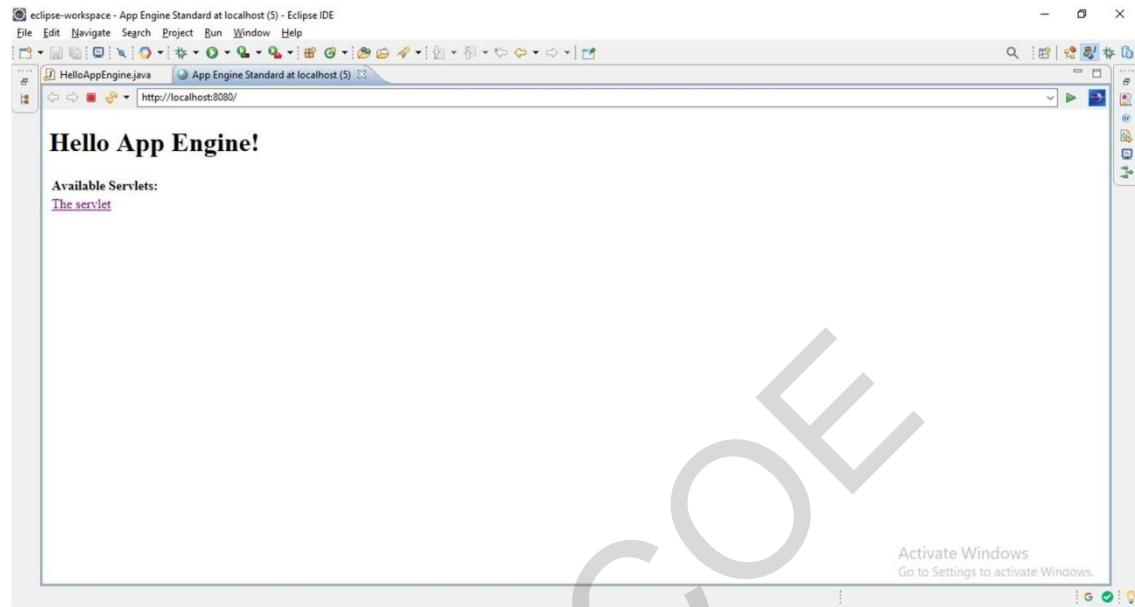
<body>
<h1>Hello App Engine!</h1>

<table>
<tr>
<td colspan="2" style="font-weight:bold;">Available Servlets:</td>
</tr>
<tr>
<td><a href='/hello'>The servlet</a></td>
</tr>
</table>
</body>

</html>
```

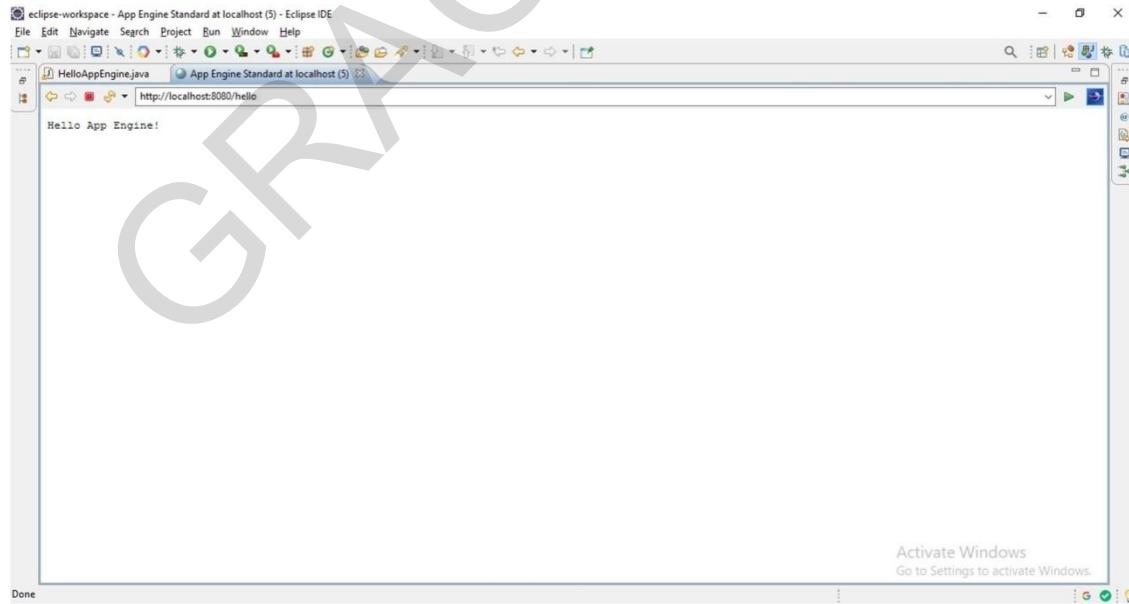
OUTPUT 1:

INSTALLATION OF A GOOGLE APP ENGINE



OUTPUT 2:

CREATION OF A WEB APPLICATIONS ON GAE



Result:

The Google App Engine is installed and *hello world* app is created in java environment.

Ex.No.4

GOOGLE APP ENGINE PYTHON HELLO WORLD EXAMPLE USING ECLIPSE

AIM:

To launch a web application using **Google App Engine (GAE)**.

TOOLS USED :

In this exercise, we are going to create a GAE based Python **web project (hello world)using** Eclipse.

1. Python 2.7
2. Eclipse 3.7 + PyDev plugin
3. Google App Engine SDK for Python 1.6.4

PROCEDURE:

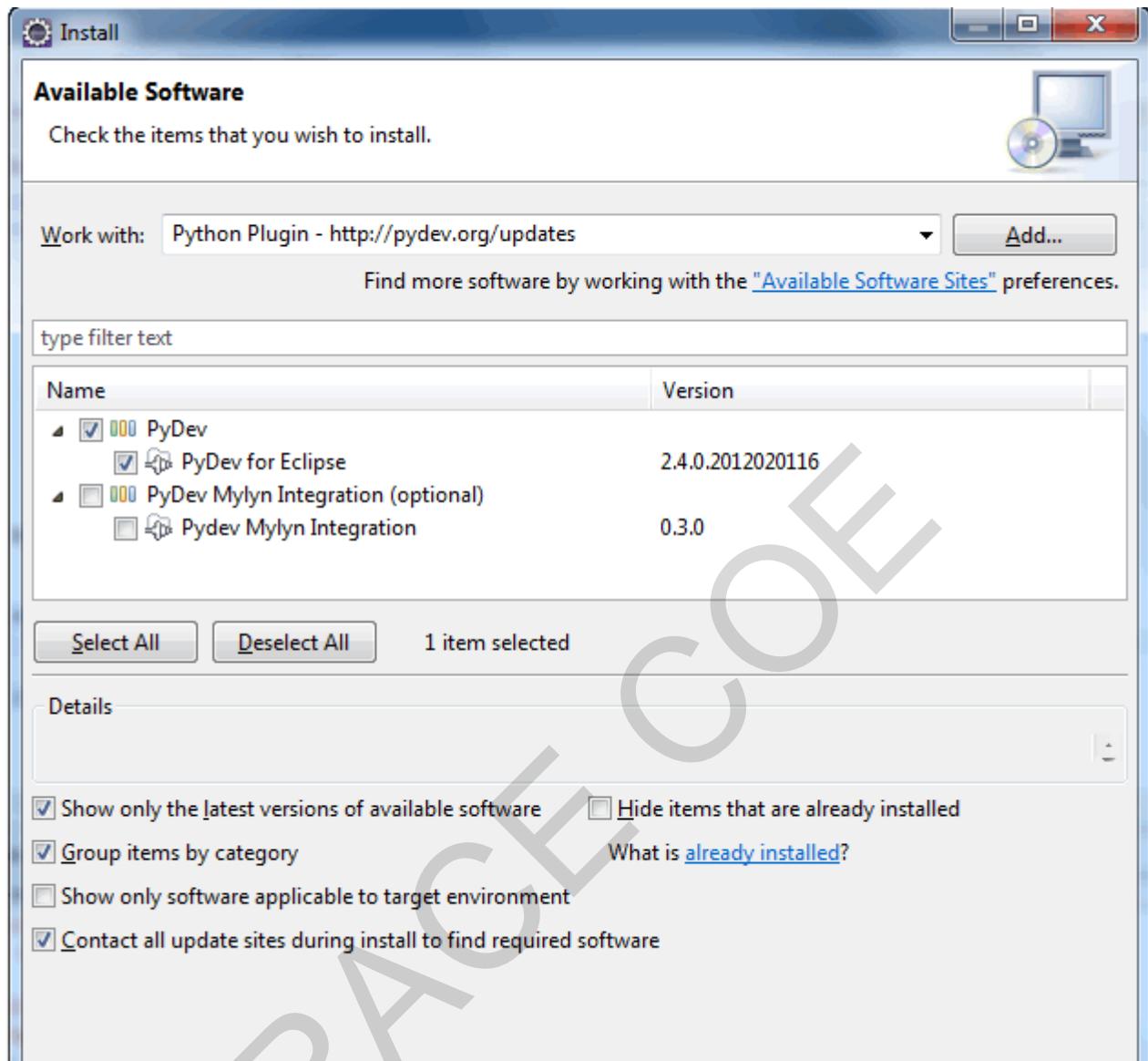
P.S Assume Python 2.7 and Eclipse 3.7 are installed.

Step:1. Install PyDev plugin for Eclipse

Use following URL to install [PyDev as Eclipse plugin.](#)

<http://pydev.org/updates>

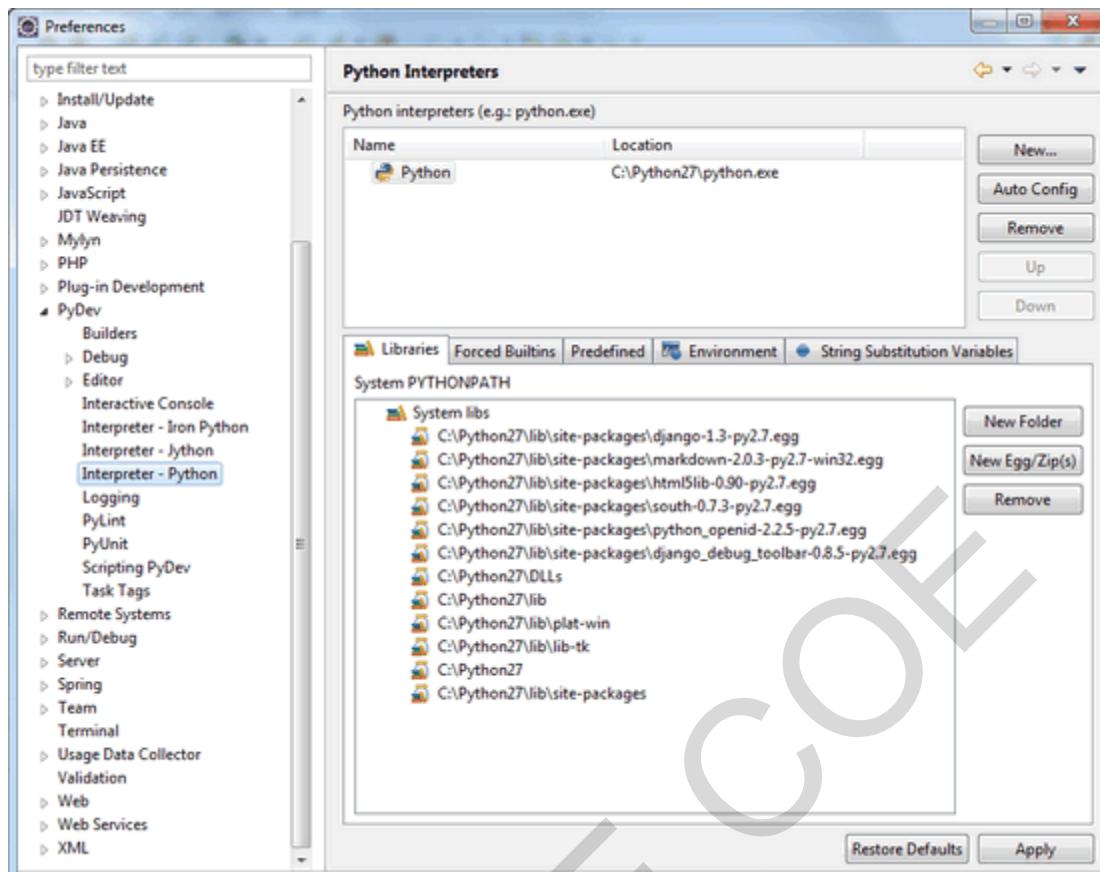
*Figure 1 – In Eclipse , menu, “Help → Install New Software..” and put above URL. Select “**PyDev for Eclipse**” option, follow steps, and restart Eclipse once completed.*



Step 2. Verify PyDev

After Eclipse is restarted, make sure **PyDev's interpreter** is pointed to your “python.exe”.

Figure 2 – Eclipse -> Windows -> Preferences, make sure “Interpreter – Python” is configured properly.



Step:3. Google App Engine SDK Python

Download and install [Google App Engine SDK for Python](#).

Step:4. Python Hello World in Eclipse

Following steps to show you how to create a GAE project via Pydev plugin.

Figure 4.1 – Eclipse menu, File -> New -> Other... , PyDev folder, choose “PyDev Google App Engine Project“.

Project“.

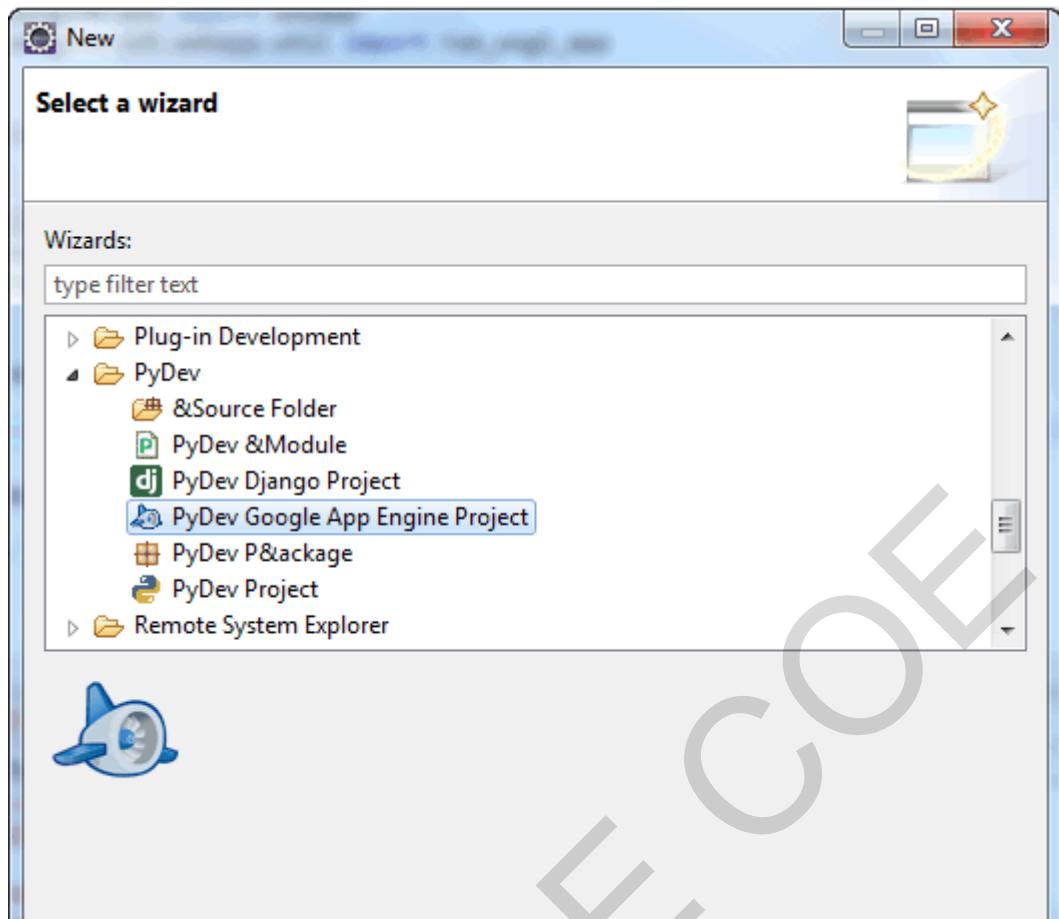


Figure 4.2 – Type project name, if the interpreter is not configure yet (in step 2), you can do it now. And select

this option – “Create ‘src’ folder and add it to PYTHONPATH”.

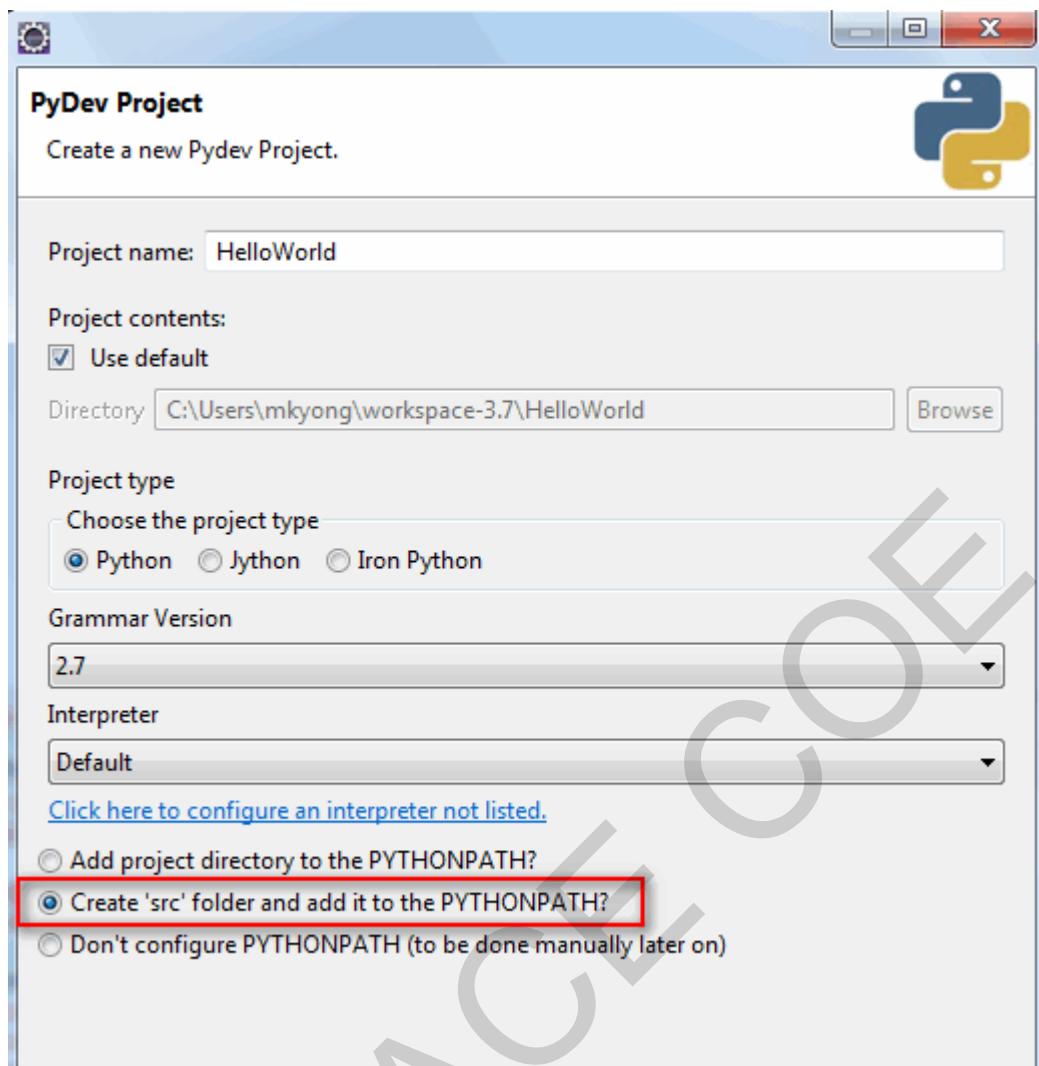


Figure 4.3 – Click “Browse” button and point it to the Google App Engine installed directory (in step 3).

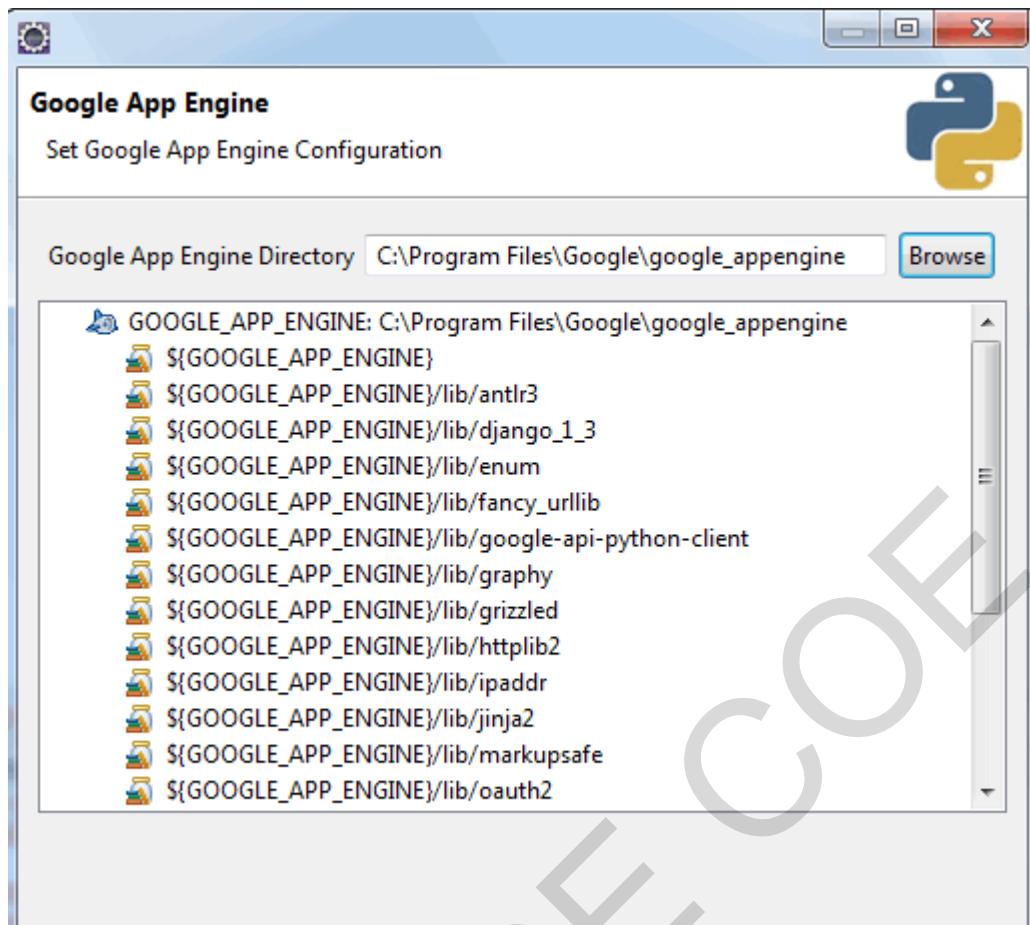


Figure 4.4 – Name your application id in GAE, type anything, you can change it later. And choose “Hello

Webapp World” template to generate the sample files.

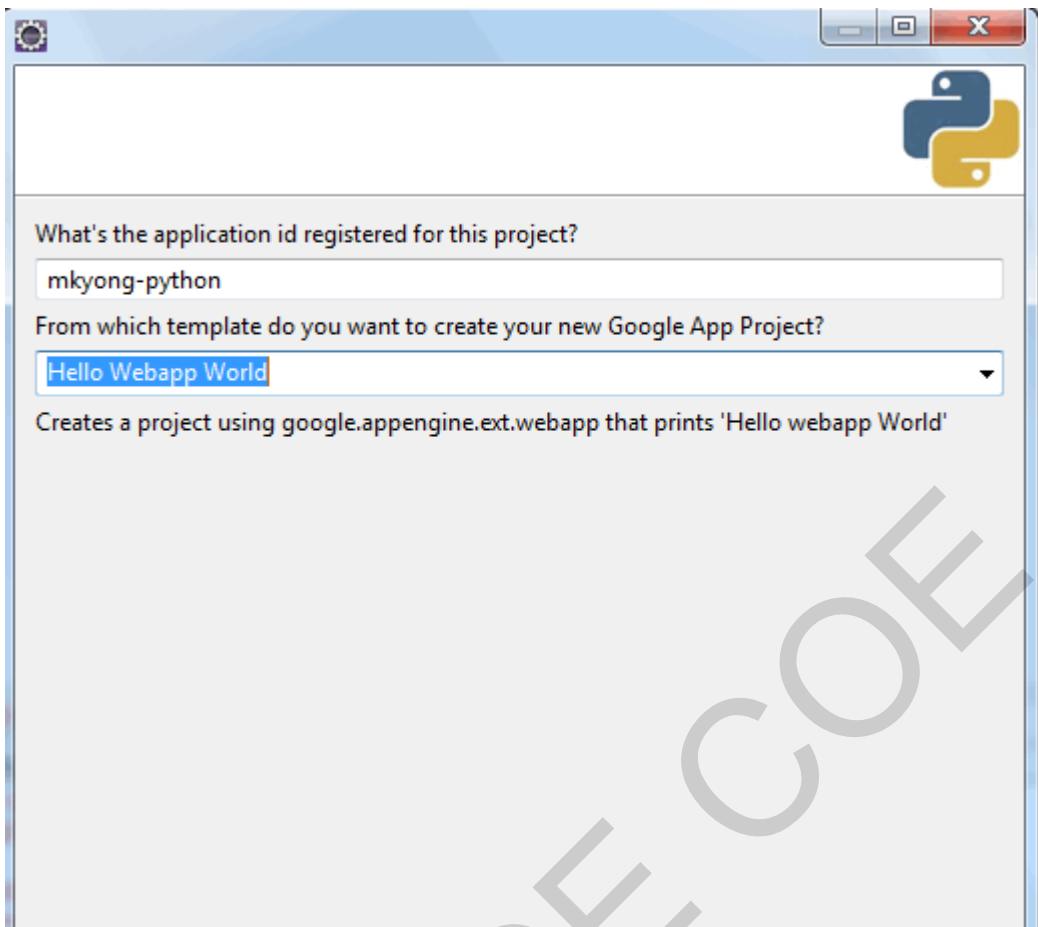
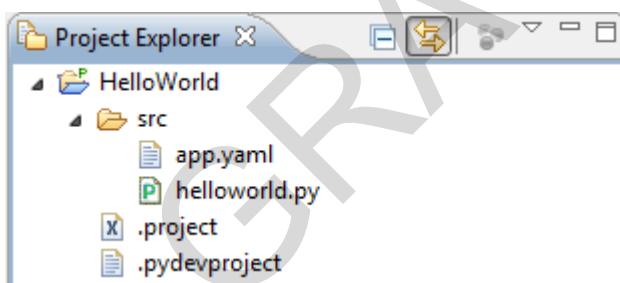


Figure 4.5 – Done, 4 files are generated, Both “.pydevproject” and “.project” are Eclipse project files, ignore it.



Review the generated Python's files :

File : helloworld.py – Just output a hello world.

```
from google.appengine.ext import webapp
from google.appengine.ext.webapp.util import run_wsgi_app

class MainPage(webapp.RequestHandler):

    def get(self):
```

```

self.response.headers['Content-Type'] = 'text/plain'
self.response.out.write('Hello, webapp World!')

application = webapp.WSGIApplication([('/', MainPage)], debug=True)

def main():
    run_wsgi_app(application)

if __name__ == "__main__":
    main()
Copy

```

File : app.yaml – GAE need this file to run and deploy your Python project, it's quite self-explanatory, for

detail syntax and configuration, visit [yaml](#) and [app.yaml reference](#).

```

application: mkyong-python
version: 1
runtime: python
api_version: 1

```

```

handlers:
- url: /.*
  script: helloworld.py
Copy

```

Step:5. Run it locally

To run it locally, right click on the helloworld.py, choose “Run As” → “Run Configuration”, create a new

“PyDev Google App Run”.

Figure 5.1 – In Main tab -> Main module, manually type the directory path of “dev_appserver.py”.

“Browse” button is not able to help you, type manually.

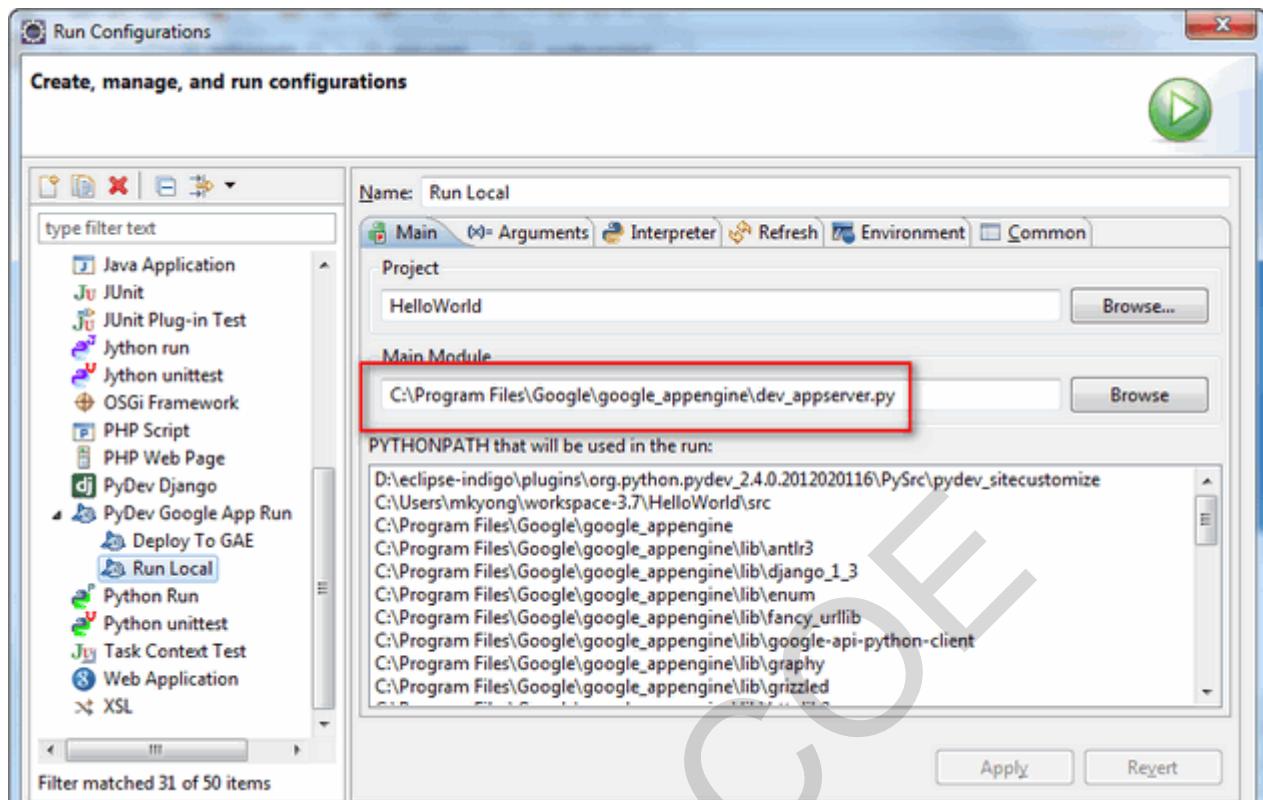


Figure 5.2 – In Arguments tab -> Program arguments, put “\${project_loc}/src“.

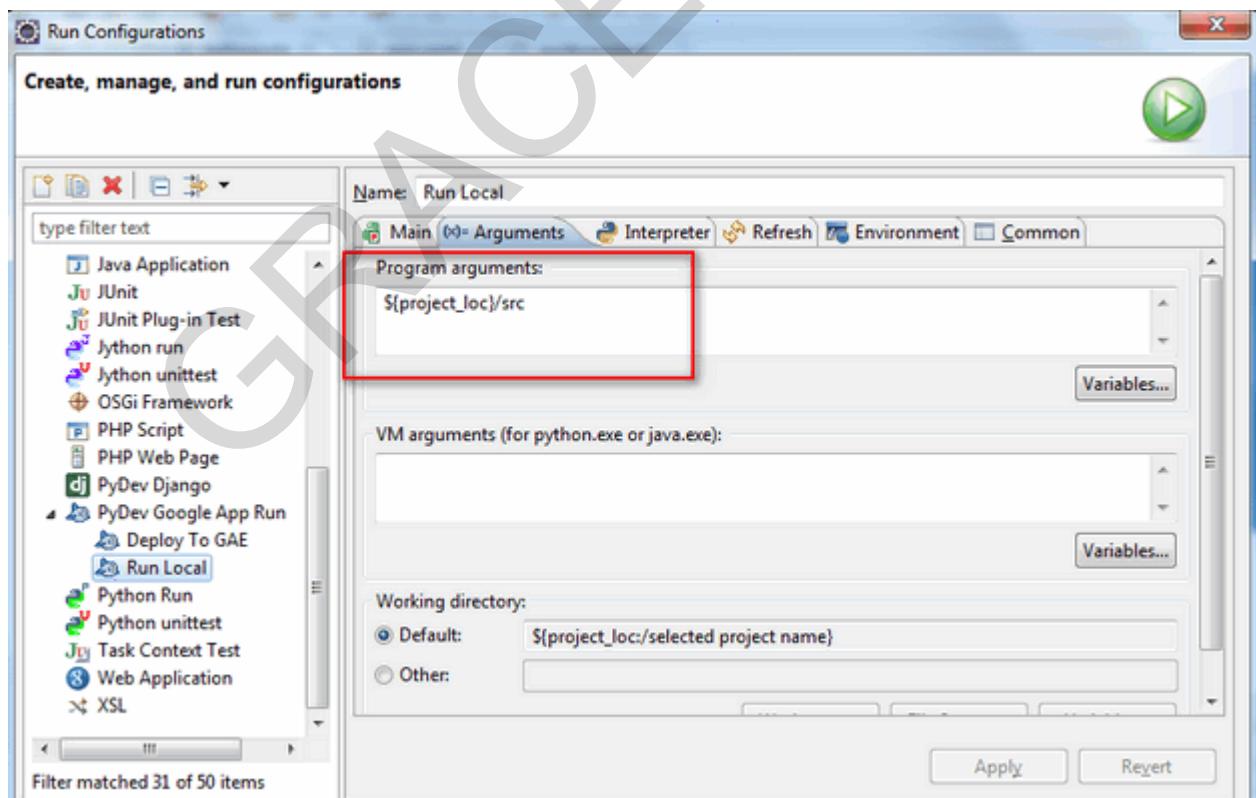
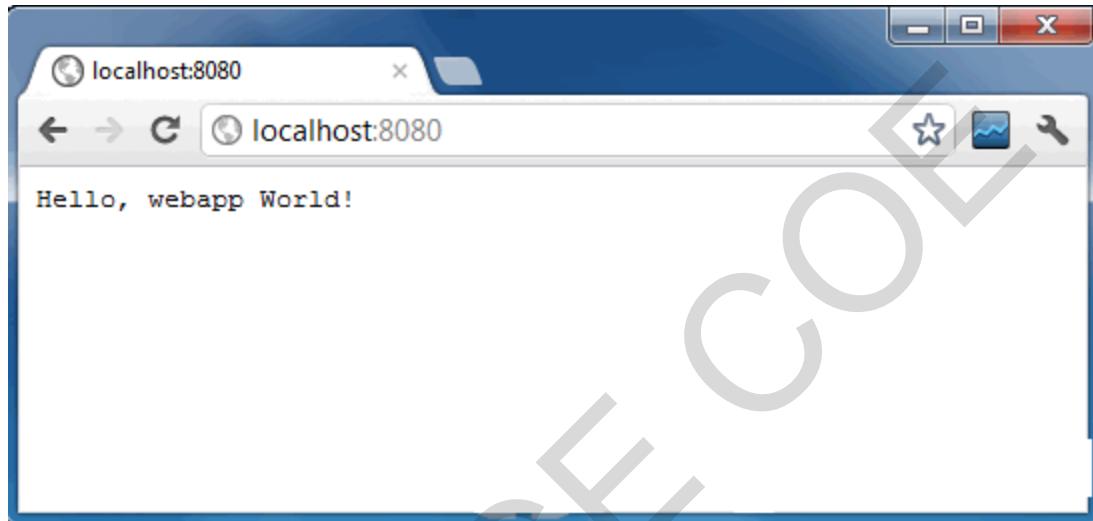


Figure 5.3 – Run it. By default, it will deploy to <http://localhost:8080>.

```
C:\Users\mkyong\workspace-3.7\HelloWorld\src
Warning: You are using a Python runtime (2.7) that is more recent than the production runtime environment (2.5). Your application may use features not available in the production environment.
INFO 2012-04-03 05:06:24,086 appengine_rpc.py:160] Server: appengine.google.com
INFO 2012-04-03 05:06:24,104 appcfg.py:581] Checking for updates to the SDK.
INFO 2012-04-03 05:06:25,756 appcfg.py:599] The SDK is up to date.
WARNING 2012-04-03 05:06:25,799 dev_appserver.py:3375] Could not read datastore data from c:\users\mkyong\appdata\local\temp\dev_appserver_data. This is likely harmless if you are using the dev_appserver.
INFO 2012-04-03 05:06:25,819 dev_appserver_multiprocess.py:658] Running application dev-mkyong-python on port 8080: http://localhost:8080
INFO 2012-04-03 05:06:25,821 dev_appserver_multiprocess.py:652] Admin console is available at: http://localhost:8080/_ah/admin
```

Figure 5.4 – Done.



5. Deploy to Google App Engine

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

application. Review “app.yaml” again, this web app will be deployed to GAE with application ID

“mkyong-python”.

File : app.yaml

application: mkyong-python

version: 1

runtime: python

api_version: 1

handlers:

- url: /.*

script: helloworld.py

Copy

To deploy to GAE, see following steps :

Figure 5.1 – Create another new “PyDev Google App Run”, In Main tab -> Main module, manually type the directory path of “appcfg.py“.

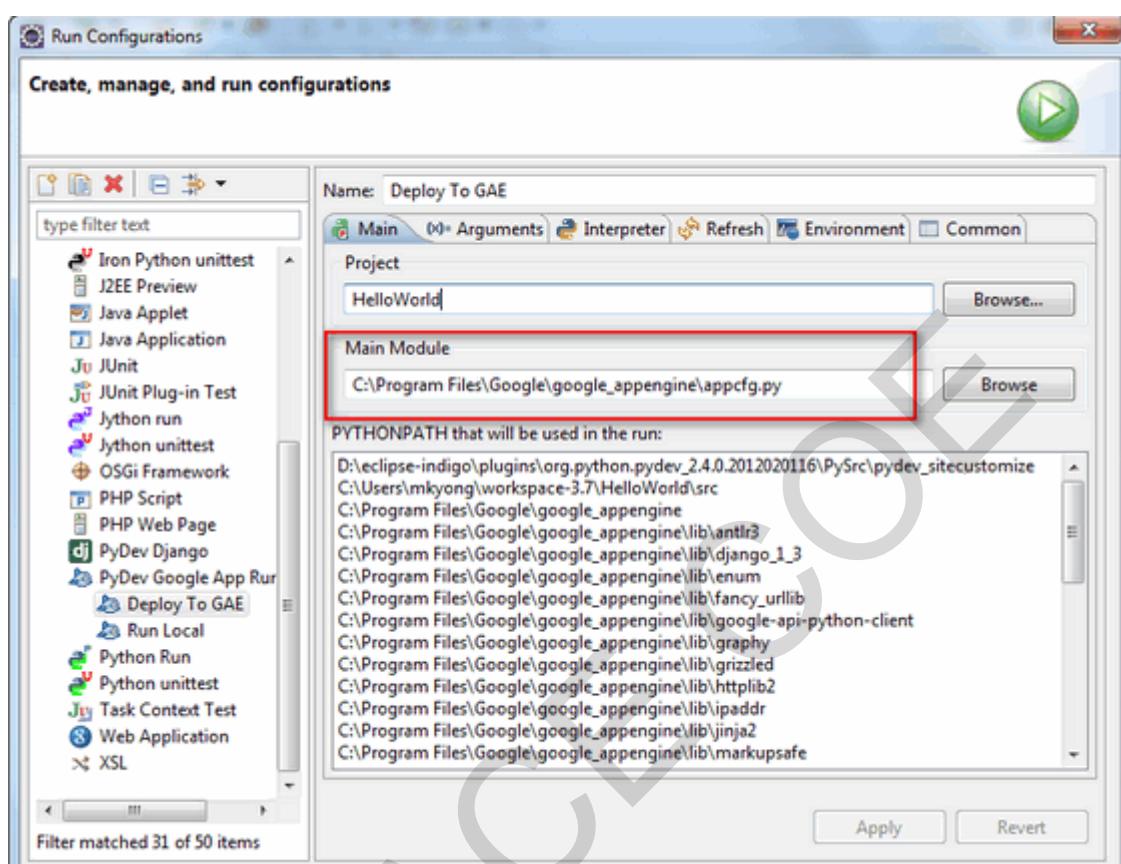


Figure 5.2 – In Arguments tab -> Program arguments, put “update \${project_loc}/src“.

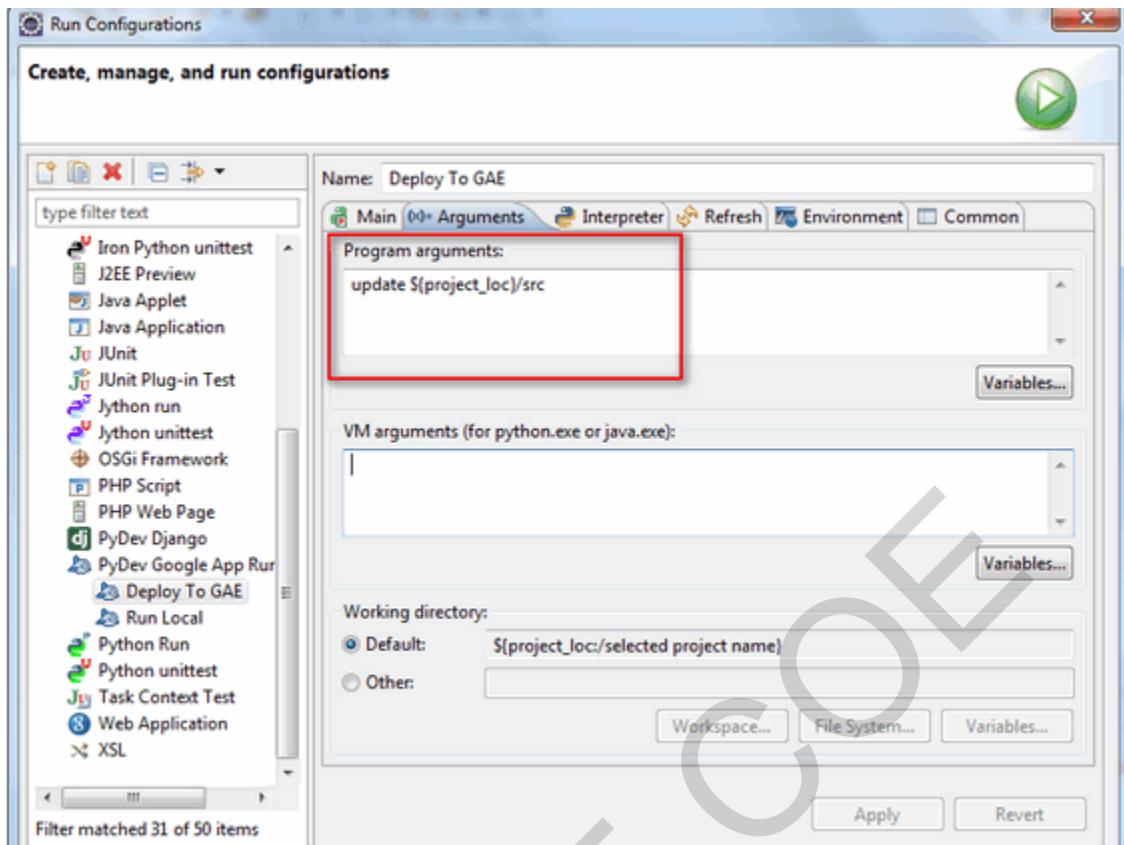


Figure 5.3 – During deploying process, you need to type your GAE email and password for authentication.

```

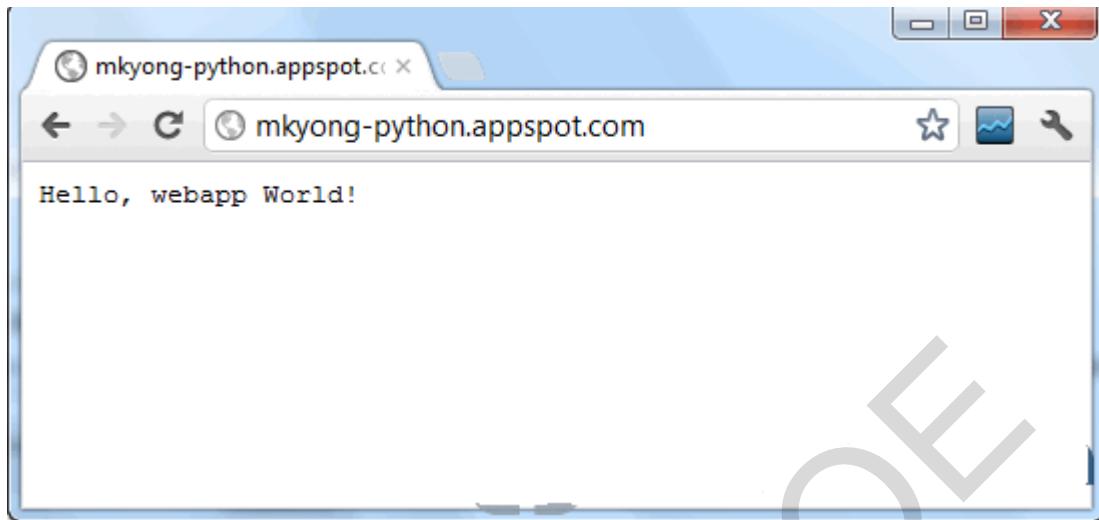
<terminated> C:\Users\mkyong\workspace-3.7\HelloWorld/src
Application: mkyong-python; version: 1
Host: appengine.google.com

Starting update of app: mkyong-python, version: 1
Getting current resource limits.
Scanning files on local disk.
Cloning 2 application files.
Compilation starting.
Compilation completed.
Starting deployment.
Checking if deployment succeeded.
Will check again in 1 seconds.
Checking if deployment succeeded.
Will check again in 2 seconds.
Checking if deployment succeeded.
Will check again in 4 seconds.
Checking if deployment succeeded.
Deployment successful.
Checking if updated app version is serving.

```

Figure 5.4 – If success, the web app will be deployed to – <http://mkyong-python.appspot.com/>.

OUTPUT:



GRACE COE

RESULT:

Thus a hello world web application has been launched using GAE.

EX.NO:5	SIMULATE A CLOUD SCENARIO USING CLOUDSIM AND RUN A SCHEDULING ALGORITHM
----------------	--

Aim:

To Simulate a cloud scenario using CloudSim and run a scheduling algorithm

Procedure:**What is Cloudsim?**

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

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

Core features of CloudSim are:

- The Support of modeling and simulation of large scale computing environment as federated cloud data centers, virtualized server hosts, with customizable policies for provisioning host resources to virtual machines and energy-aware computational resources
- It is a self-contained platform for modeling cloud's service brokers, provisioning, and allocation policies.
- It supports the simulation of network connections among simulated system elements.
- Support for simulation of federated cloud environment, that inter-networks resources from both private and public domains.
- Availability of a virtualization engine that aids in the creation and management of multiple independent and co-hosted virtual services on a data center node.
- Flexibility to switch between space shared and time shared allocation of processing cores to virtualized services.

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

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!

RESULT:

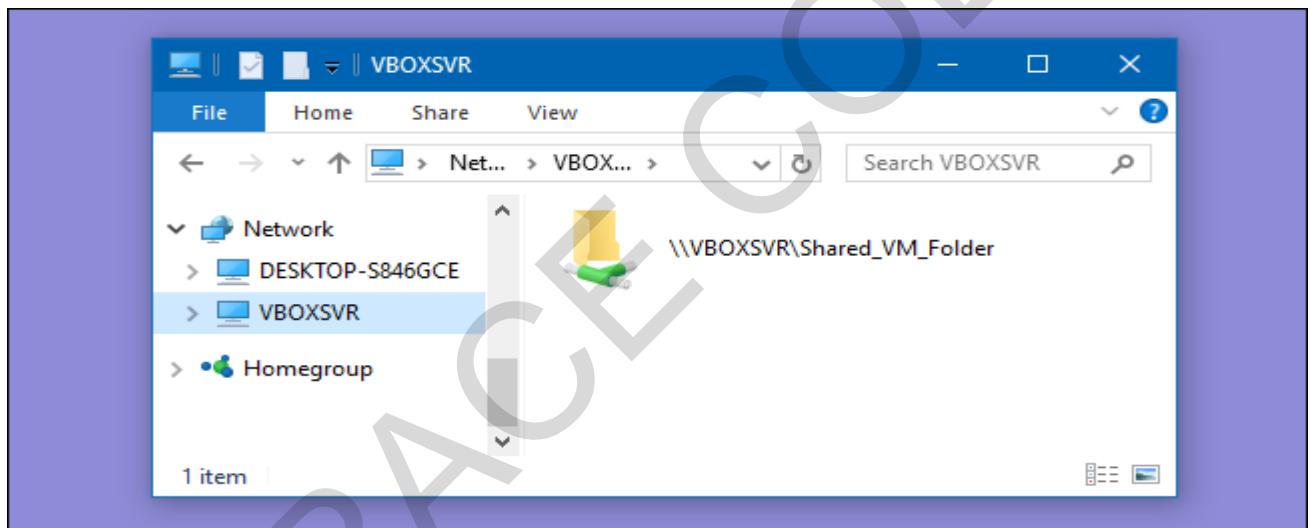
Thus a hello world web application has been launched using GAE.

EX.NO:6**COPY FILES FROM ONE VIRTUAL MACHINE TO ANOTHER****AIM:**

To write a procedure to copy files from one virtual machine to another

PROCEDURE:

- 1.Create one shared folder in virtual box

**VirtualBox**

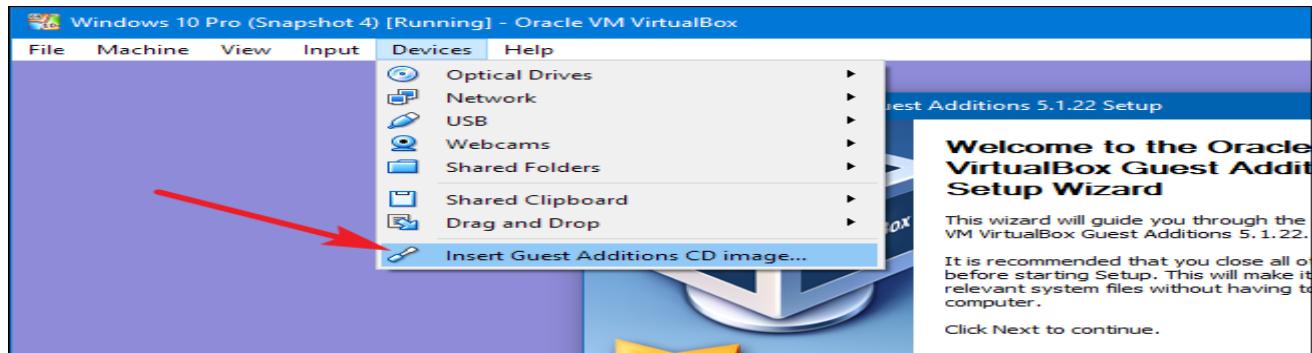
VirtualBox's Shared Folders feature works with both Windows and Linux guest operating systems. To use

the feature, you first need to install VirtualBox's Guest Additions in the guest virtual machine.

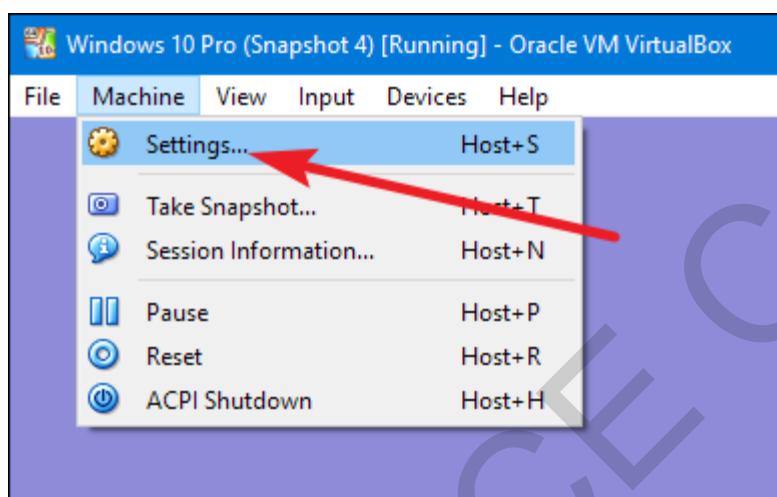
With the virtual machine running, click the "Devices" menu and choose the "Insert Guest Additions CD

image" option. This inserts a virtual CD that you can use within the guest operating system to install the

Guest Additions.



After the Guest Additions are installed, open the “Machine” menu and click the “Settings” option.



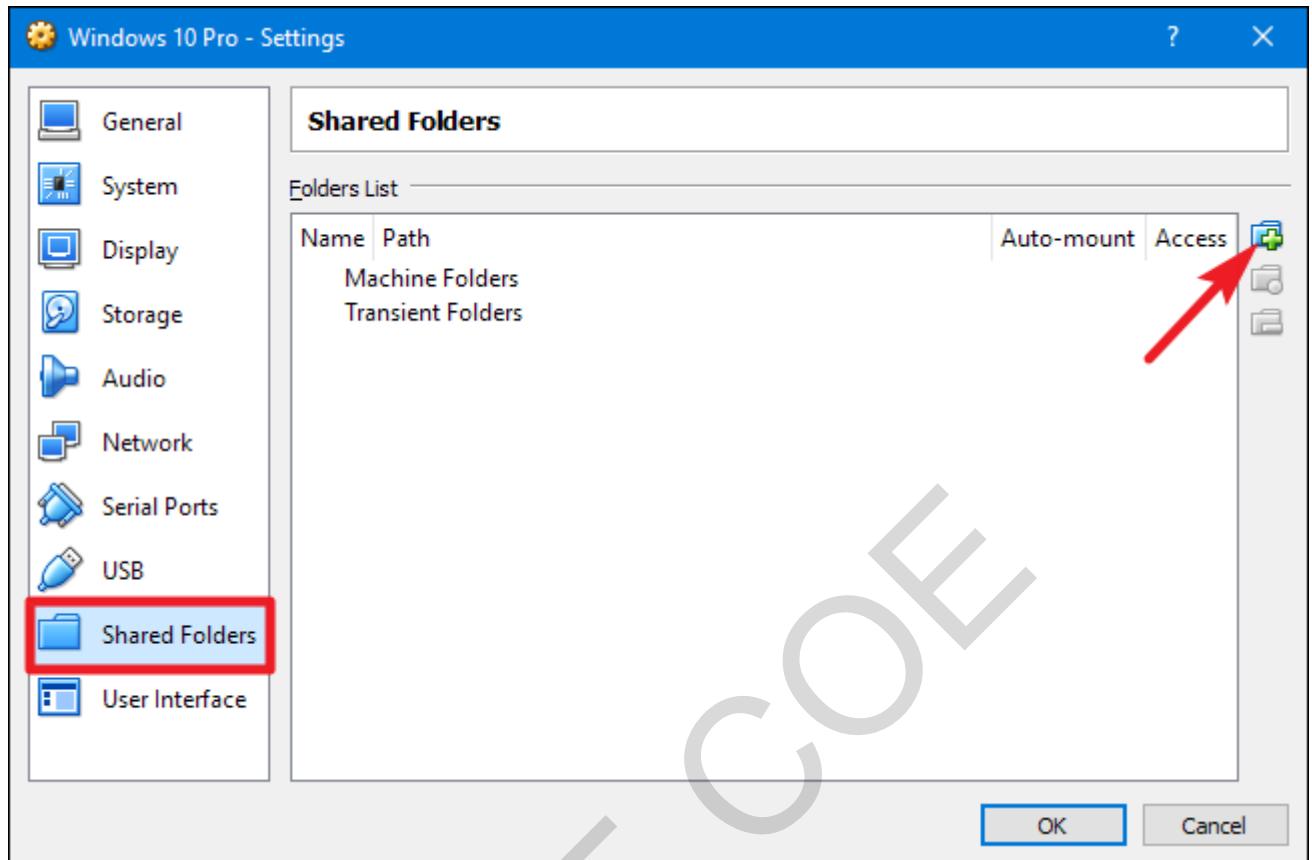
In the “Settings” window, switch to the “Shared Folders” tab. Here you can see any shared folders you’ve

set up. There are two types of shared folders. Machine Folders are permanent folders that are shared until

you remove them. Transient Folders are temporary and are automatically removed when you restart or shut down

the virtual machine.

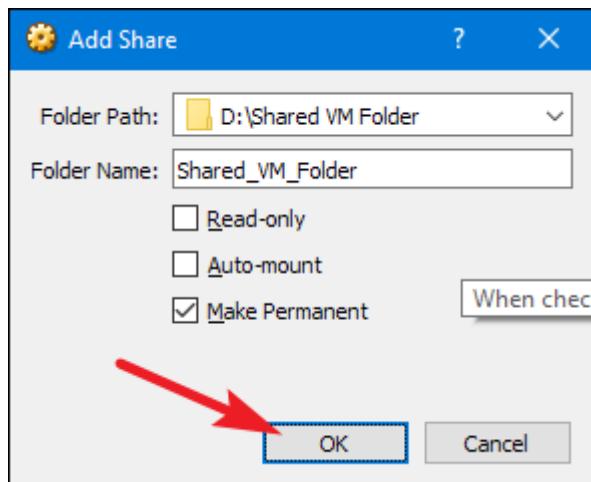
Click the “Add” button (the folder with a plus on it) to create a new shared folder.



In the “Add Share” window, you can specify the following:

- **Folder Path:** This is the location of the shared folder on your host operating system (your real PC).
- **Folder Name:** This is how the shared folder will appear inside the guest operating system.
- **Read-only:** By default, the virtual machine has full read-write access to the shared folder. Enable the “Read-only” checkbox if you want the virtual machine only to be able to read files from the shared folder, but not modify them.
- **Auto-mount:** This option makes the guest operating system attempt to automatically mount the folder when it boots.
- **Make Permanent:** This option makes the shared folder a Machine Folder. If you don’t select this option, it becomes a transient folder that is removed with the virtual machine restarts.

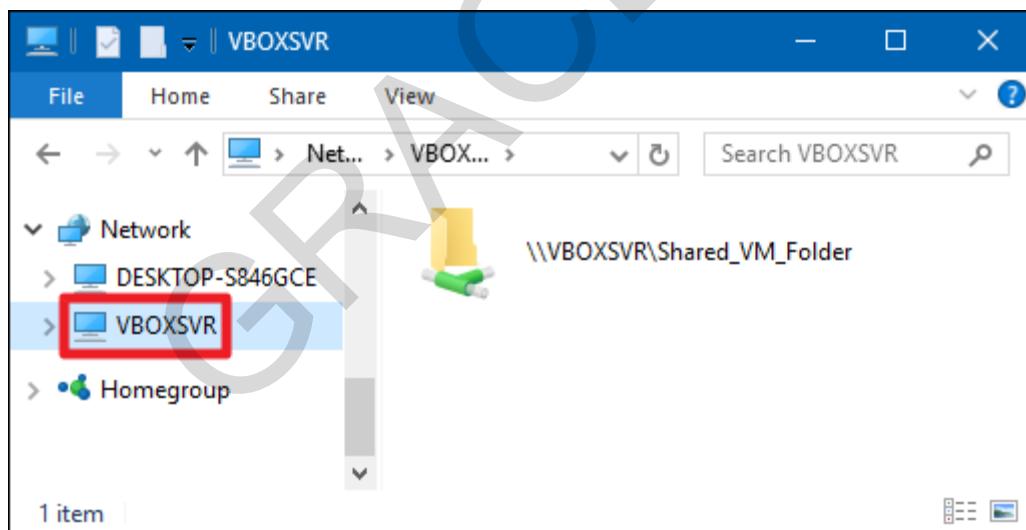
Make all your choices and then hit the “OK” button.



You should now see the shared folders appear as network file shares. If you're using a Windows guest operating system, open File Explorer, select "Network", and then look under the "VBOXSRV" computer.

OUTPUT:

"VBOXSRV" contains shared files.



RESULT:

Thus a procedure to copy files from one virtual machine to another virtual machine was executed successfully.

Ex.No.7**CREATE, DEPLOY AND LAUNCH VIRTUAL MACHINES IN OPENSTACK****AIM:**

To Write a procedure to create ,deploy and launch a virtul machine s in open stack.

REQUIREMENTS

1. [Install OpenStack in RHEL and CentOS 7](#)
2. [Configure OpenStack Networking Service](#)

PROCEDURE:**Step 1: Allocate Floating IP to OpenStack**

1. Before you deploy an **OpenStack** image, first you need to assure that all pieces are in place and we'll
2. start by allocating floating IP.

Floating IP allows external access from outside networks or internet to an Openstack virtual machine. In

order to create floating IPs for your project, login with your **user** credentials and go to **Project ->**

Compute -> Access & Security -> Floating IPs tab and click on **Allocate IP** to The Project.

Choose external **Pool** and hit on **Allocate IP** button and the IP address should appear in dashboard. It's a good

idea to allocate a Floating IP for each instance you run.

IP Address	Mapped Fixed IP Address	Pool	Status	Actions
No items to display.				

Allocate Floating IP to Project in OpenStack

The screenshot shows the OpenStack dashboard for the 'tecmint-proj' project. The 'Compute' tab is selected. In the 'Access & Security' section, the 'Floating IPs' tab is active. A modal window titled 'Allocate Floating IP' is displayed, prompting for a pool ('external') and providing a description of allocating a floating IP from a given pool. It also shows project quotas for floating IPs.

Allocate Floating IP to External Pool

The screenshot shows the OpenStack dashboard for the 'tecmint-proj' project. The 'Compute' tab is selected. In the 'Access & Security' section, the 'Floating IPs' tab is active. A success message box indicates that a floating IP has been allocated. The main table lists two floating IPs: 192.168.1.5 and 192.168.1.6, both in a 'Down' status. The 'Actions' column for each IP includes an 'Associate' dropdown menu.

IP Address	Mapped Fixed IP Address	Pool	Status	Actions
192.168.1.5	-	-	Down	Associate
192.168.1.6	-	-	Down	Associate

Confirmation of Adding Floating IP

Step 2: Create an OpenStack Image

3. OpenStack images are just virtual machines already created by third-parties. You can create your own
4. customized images on your machine by installing an Linux OS in a virtual machine using a
5. virtualization tool, such as [KVM](#), [VirtualBox](#), [VMware](#) or [Hyper-V](#).

Once you have installed the OS, just convert the file to raw and upload it to your OpenStack cloud infrastructure.

To deploy official images provided by major Linux distributions use the following links to download the

latest packaged images:

1. **CentOS 7** – <http://cloud.centos.org/centos/7/images/>
2. **CentOS 6** – <http://cloud.centos.org/centos/6/images/>
3. **Fedora 23** – <https://download.fedoraproject.org/pub/fedora/linux/releases/23/Cloud/>
4. **Ubuntu** – <http://cloud-images.ubuntu.com/>
5. **Debian** – <http://cdimage.debian.org/cdimage/openstack/current/>
6. **Windows Server 2012 R2** – <https://cloudbase.it/windows-cloud-images/#download>

Official images additionally contain the **cloud-init** package which is responsible with SSH key pair and user

data injection.

On this guide we'll deploy a test image, for demonstration purposes, based on a lightweight Cirros cloud

image which can be obtained by visiting the following link <http://download.cirros-cloud.net/0.3.4/>.

The image file can be used directly from the HTTP link or downloaded locally on your machine and

uploaded to OpenStack cloud.

To create an image, go **OpenStack** web panel and navigate to **Project -> Compute -> Images** and hit on

Create Image button. On the image prompt use the following settings and hit on **Create Image** when done.

Name: **tecmint-test**

Description: **Cirros test image**

Image Source: **Image Location** #Use Image File if you've downloaded the file locally on your hard disk
 Image Location: <http://download.cirros-cloud.net/0.3.4/cirros-0.3.4-i386-disk.img>
 Format: QCOWW2 – QEMU Emulator
 Architecture: leave blank
Minimum Disk: leave blank
Minimum RAM: leave blank
 Image Location: checked
 Public: unchecked
 Protected: unchecked

Image Name	Type	Status	Public	Protected	Format	Size	Actions

Create Images in OpenStack

Add OpenStack Image Details

Image Name	Type	Status	Public	Protected	Format	Size	Actions
tecmint-test	Image	Active	No	No	QCOW2	11.9 MB	<button>Launch</button>

OpenStack Images

Step 3: Launch an Image Instance in OpenStack

3. Once you've created an image you're good to go. Now you can run the virtual machine based on the
4. image created earlier in your cloud environment.

Move to **Project -> Instances** and hit on **Launch Instance** button and a new window will appear.

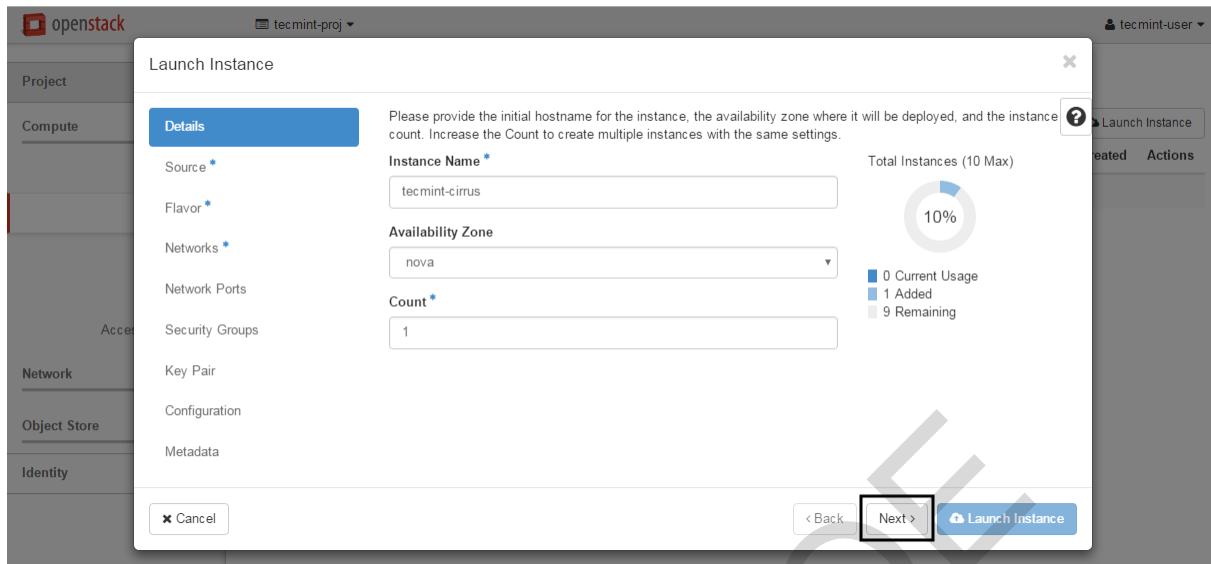
Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
No items to display.										

Launch Image Instance in Openstack

5. On the first screen add a name for your instance, leave the **Availability Zone** to nova, use one instance count
6. and hit on **Next** button to continue.

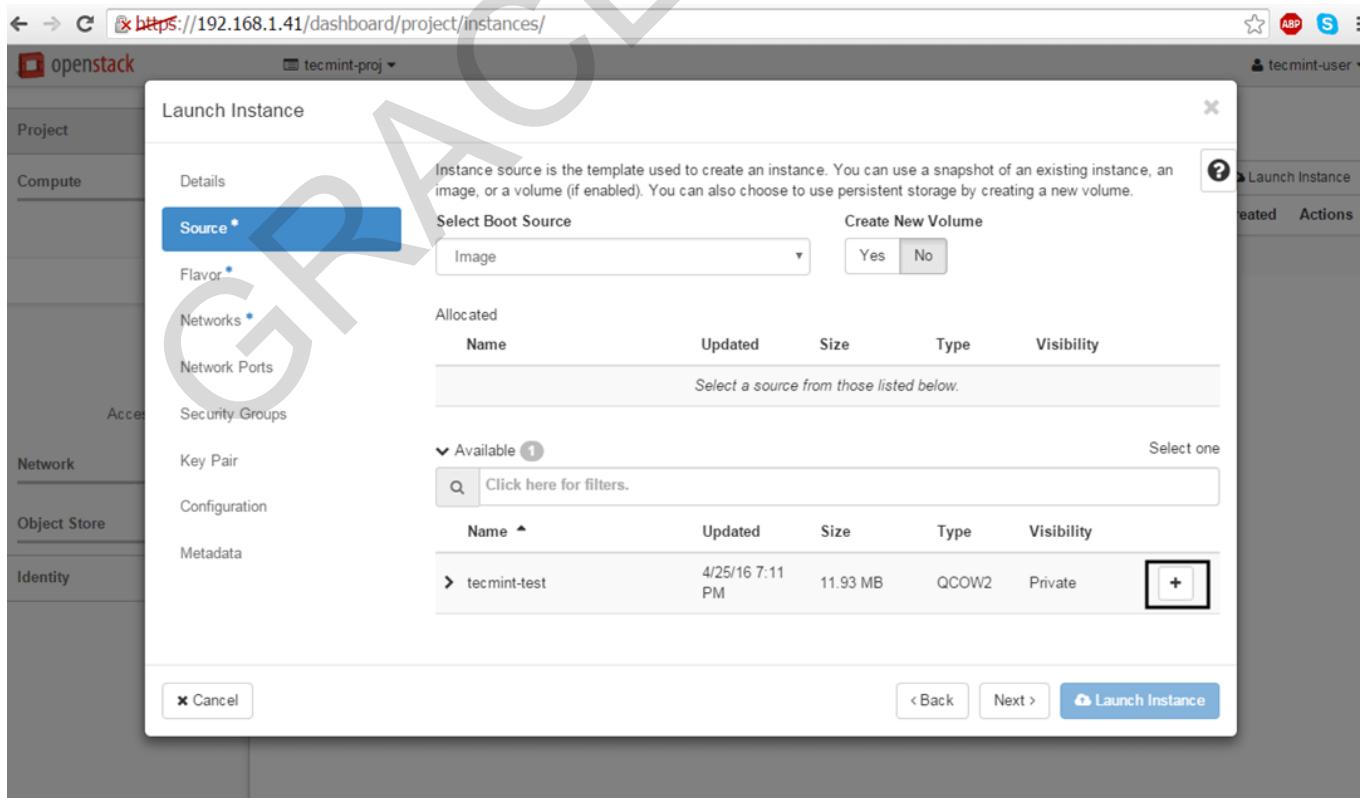
Choose a descriptive **Instance Name** for your instance because this name will be used to form the virtual

machine hostname.

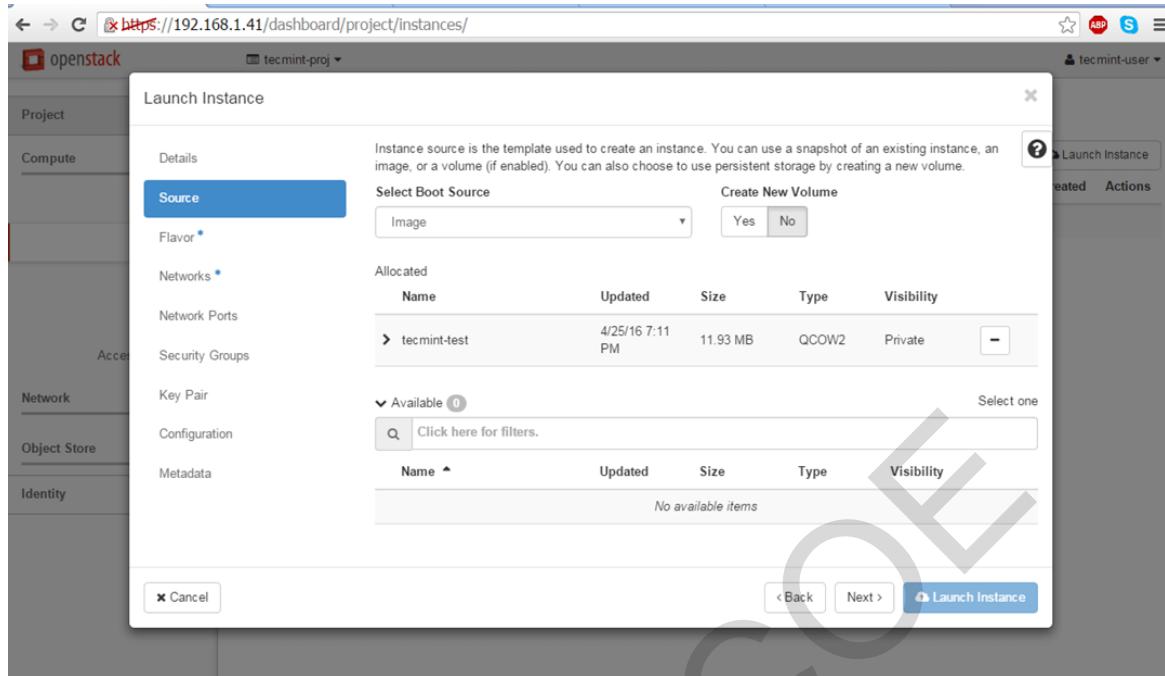


Add Hostname to OpenStack Instance

6. Next, select Image as a **Boot Source**, add the Cirros test image created earlier by hitting the + button
7. and hit
8. **Next** to proceed further.

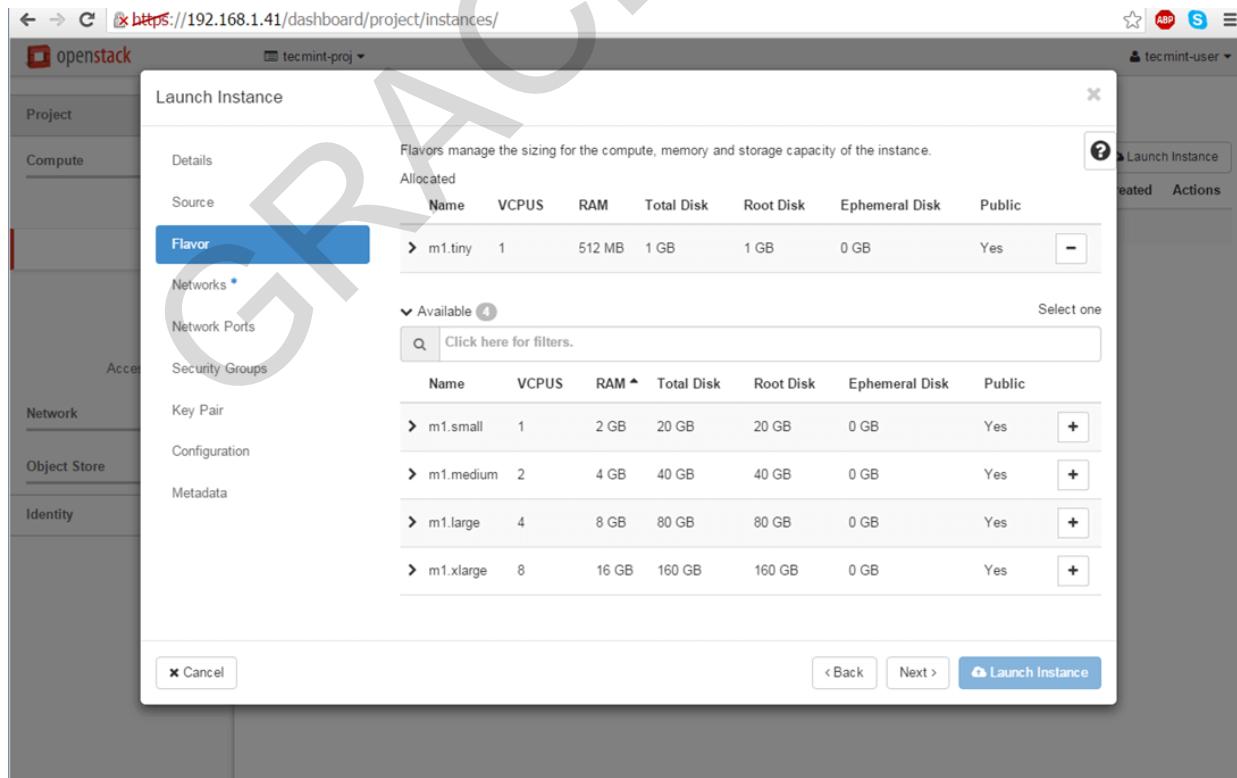


Select OpenStack Instance Boot Source



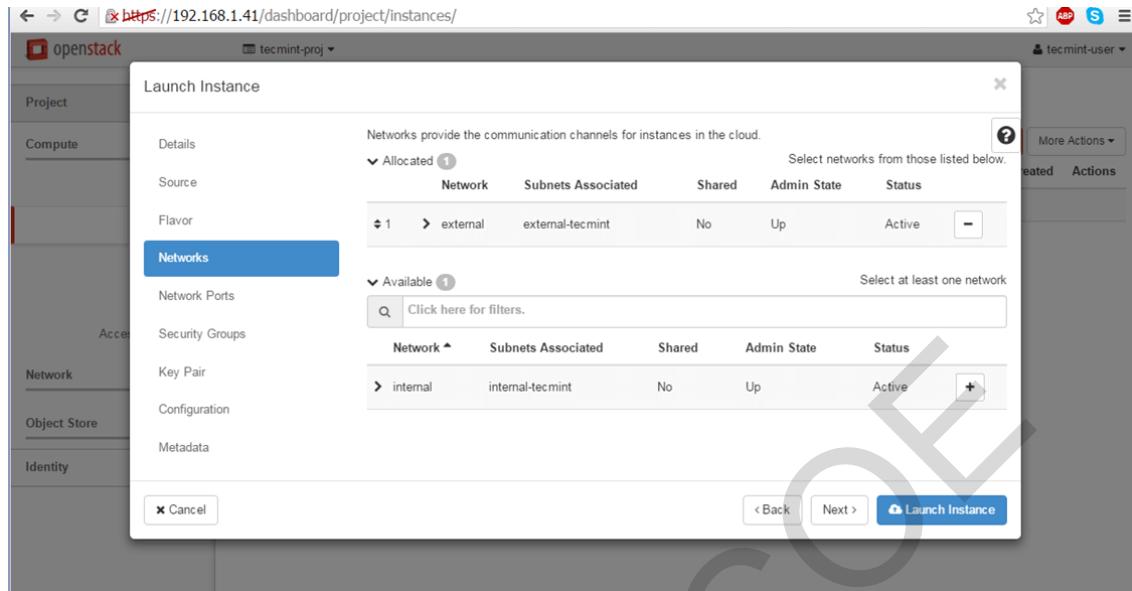
Add Cirros Text Image

7. Allocate the virtual machine resources by adding a flavor best suited for your needs and click on **Next** to move on.



Add Resources to OpenStack Instance

7. Finally, add one of the OpenStack available networks to your instance using the + button and hit on
8. **Launch Instance** to start the virtual machine.



Add Network to OpenStack Instance

9. Once the instance has been started, hit on the right arrow from **Create Snapshot** menu button and
10. choose **Associate Floating IP**.

Select one of the floating IP created earlier and hit on **Associate** button in order to make the instance reachable from your internal LAN.

OUTPUT:

The screenshot shows the OpenStack dashboard under the 'Instances' tab. A single instance named 'tecmint-test' is listed, showing it is active and running. A terminal window is open on the right, displaying a ping command to 'google.com' from the instance's IP address, 192.168.1.5. The terminal output includes details about the ping statistics.

Instance Name	Image Name	IP Address	Size	Key Pair	Status	Availability Zone	Task	Power State	Time since created	Actions
tecmint-test	tecmint-test	192.168.1.5	mini	-	Active	nova	None	Running	50 minutes	<button>Create Snapshot</button>

```

$ uname -a
Linux tecmint-test 3.2.0-80-virtual #116-Ubuntu SMP Mon Mar 23 17:28:52 UTC 2015
x86_64 GNU/Linux
$ ping -c 2 google.com
PING google.com (172.217.20.142): 56 data bytes
64 bytes from 172.217.20.142: seq=0 ttl=55 time=42.433 ms
64 bytes from 172.217.20.142: seq=1 ttl=55 time=39.585 ms

--- google.com ping statistics ---
2 packets transmitted, 2 packets received, 0% packet loss
round-trip min/avg/max = 39.585/41.009/42.433 ms
$ cat /etc/resolv.conf
search openstacklocal
nameserver 192.168.1.1
nameserver 8.8.8.8
nameserver 8.8.4.4
$ 

```

RESULT:

Thus the procedure to deploy virtual machined in open stack was successfully executed.

Ex.No.8**INSTALL HADOOP SINGLE NODE CLUSTER****AIM:**

To write a procedure to install a single node Hadoop cluster

PROCEDURE:

Java

Download the Java 1.8 from <https://java.com/en/download/>

Once installed confirm that you're running the correct version from command line using 'java -version'

command, output of which you can confirm in command line like this:

```
C:\WINDOWS\system32>java -version
java version "1.8.0_111"
Java(TM) SE Runtime Environment (build 1.8.0_111-b14)
Java HotSpot(TM) 64-Bit Server VM (build 25.111-b14, mixed mode)
```

WinRAR

I've downloaded and installed WinRAR 64 bit release from <http://www.rarlab.com/download.htm> that will

later allow me to decompress Linux type tar.gz packages on Windows.

Hadoop

The next step was to install a Hadoop distribution. To do so, I've decided to download the most recent

release **Hadoop 3.0.0-alpha2** (25 Jan, 2017) in a binary form, from the Apache Download

Mirror at <http://hadoop.apache.org/releases.html>

Once the **hadoop-3.0.0-alpha2.tar.gz** (250 MB) downloaded, I've extracted it by using WinRAR (installed in

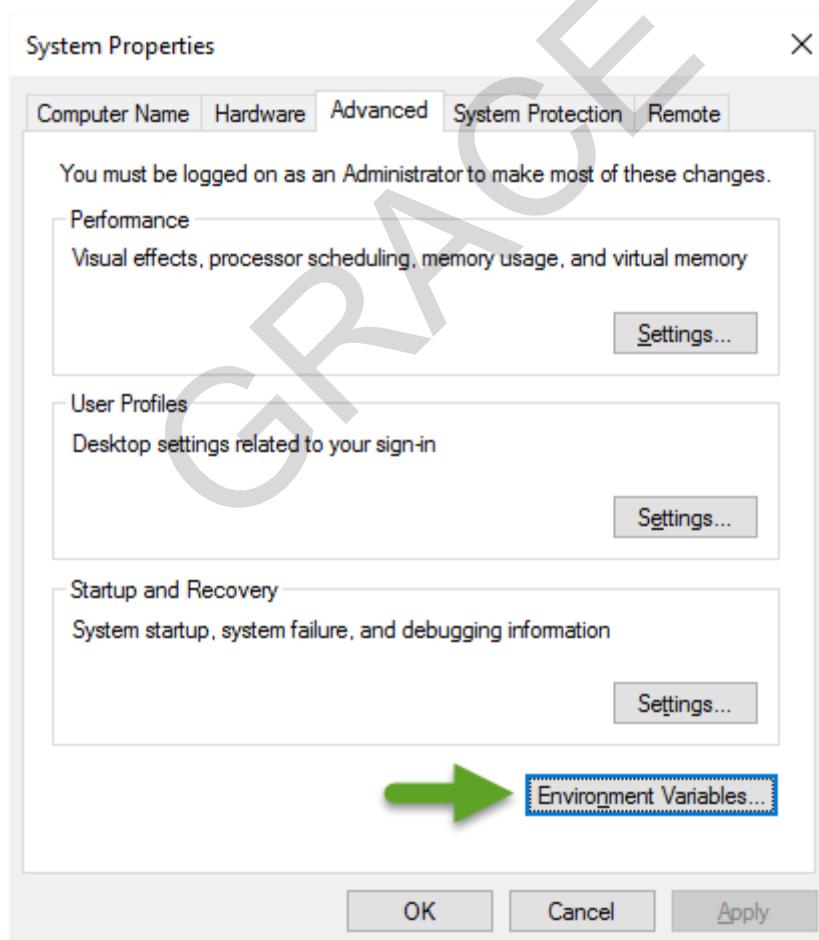
the previous step) into C:\hadoop-3.0.0-alpha2 folder:

Name	Date modified	Type	Size
bin	2/01/17 5:49 AM	File folder	
etc	2/01/17 5:49 AM	File folder	
include	2/01/17 5:46 AM	File folder	
lib	2/01/17 5:46 AM	File folder	
libexec	2/01/17 5:46 AM	File folder	
sbin	2/01/17 5:46 AM	File folder	
share	2/01/17 5:48 AM	File folder	
LICENSE.txt	1/09/17 9:01 PM	Text Document	140 KB
NOTICE.txt	1/09/17 9:01 PM	Text Document	21 KB
README.txt	7/08/16 4:34 PM	Text Document	2 KB

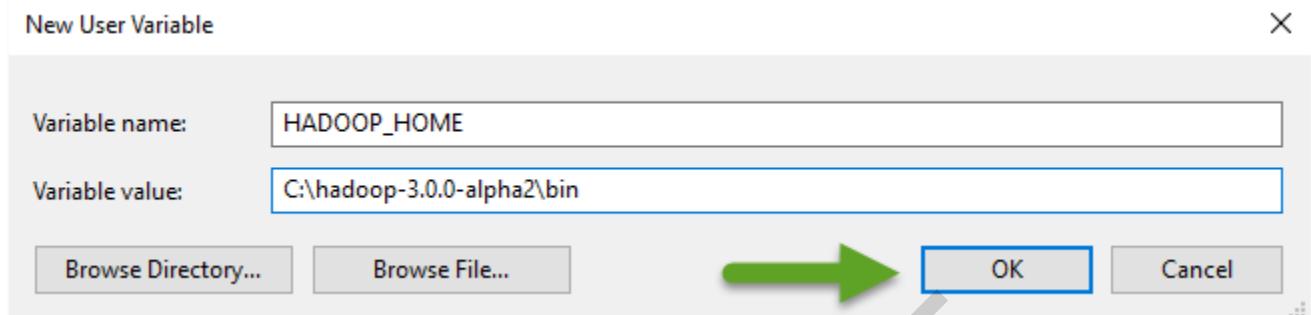
Now that I had Hadoop downloaded, it was time to start the Hadoop cluster with a single node.

Setup Environmental Variables

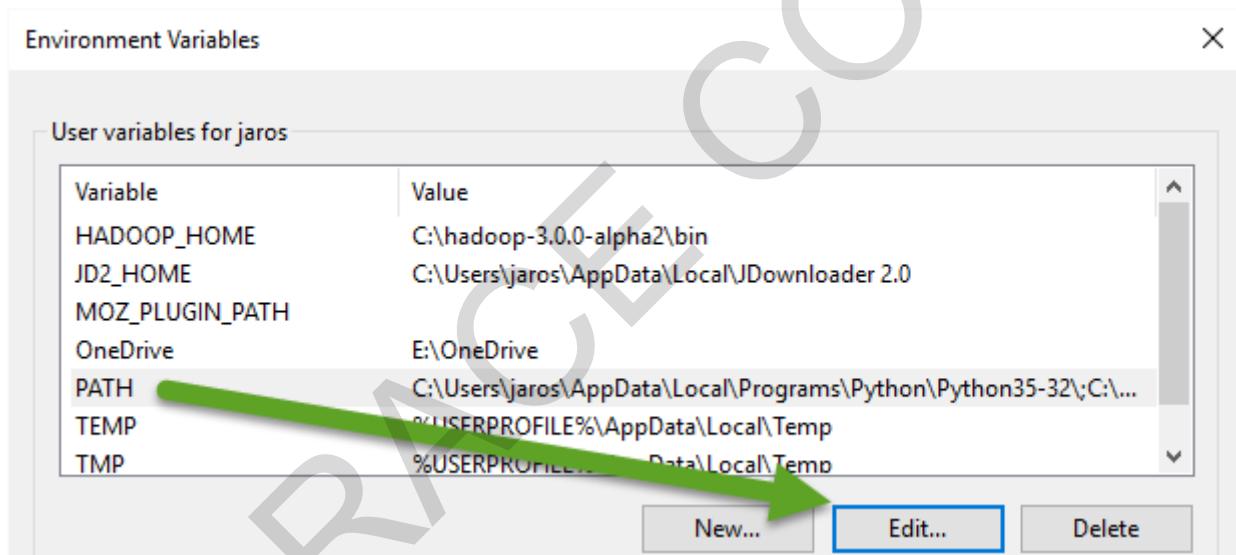
In Windows 10 I've opened System Properties windows and clicked on Environment Variables button:



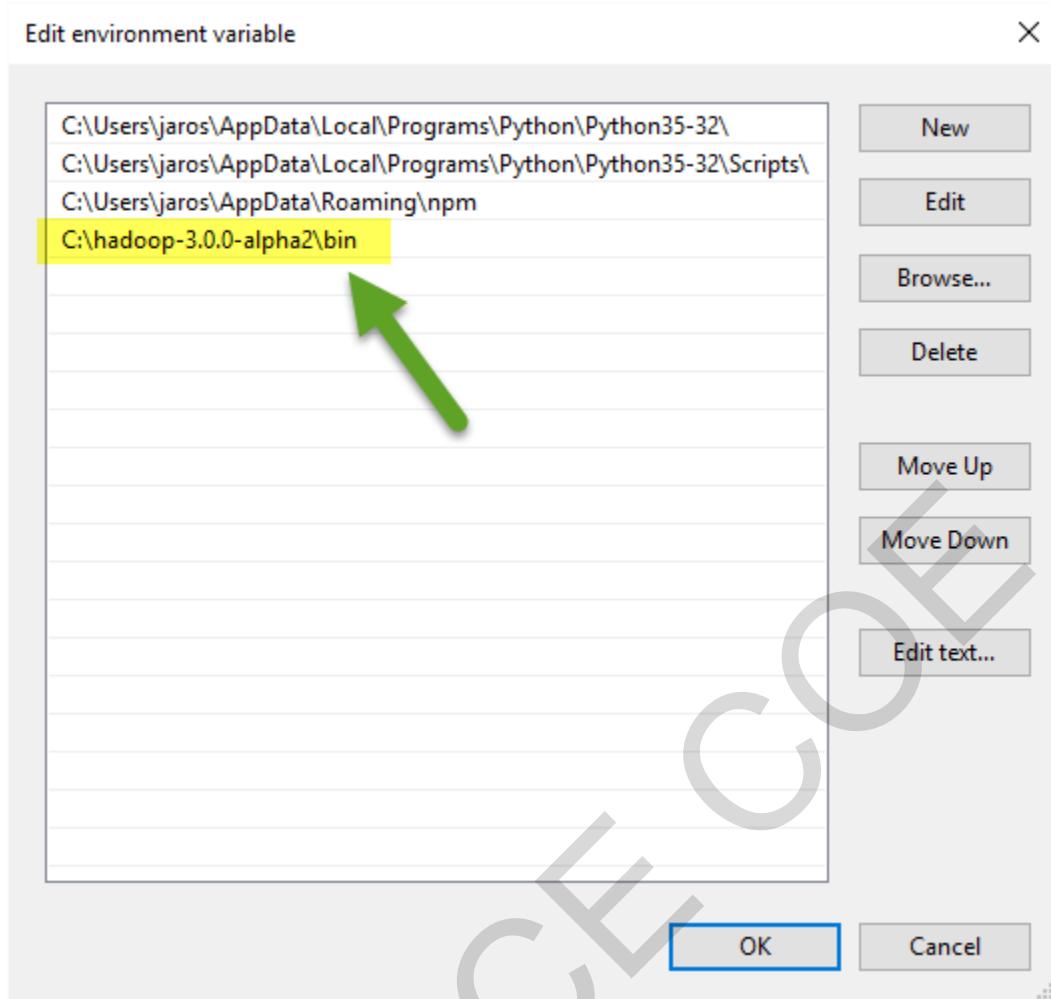
hen created a new HADOOP_HOME variable and pointed the path to C:\hadoop-3.0.0-alpha2\bin folder on my PC:



Next step was to add a Hadoop bin directory path to PATH variable. Clicked on PATH and pressed edit:

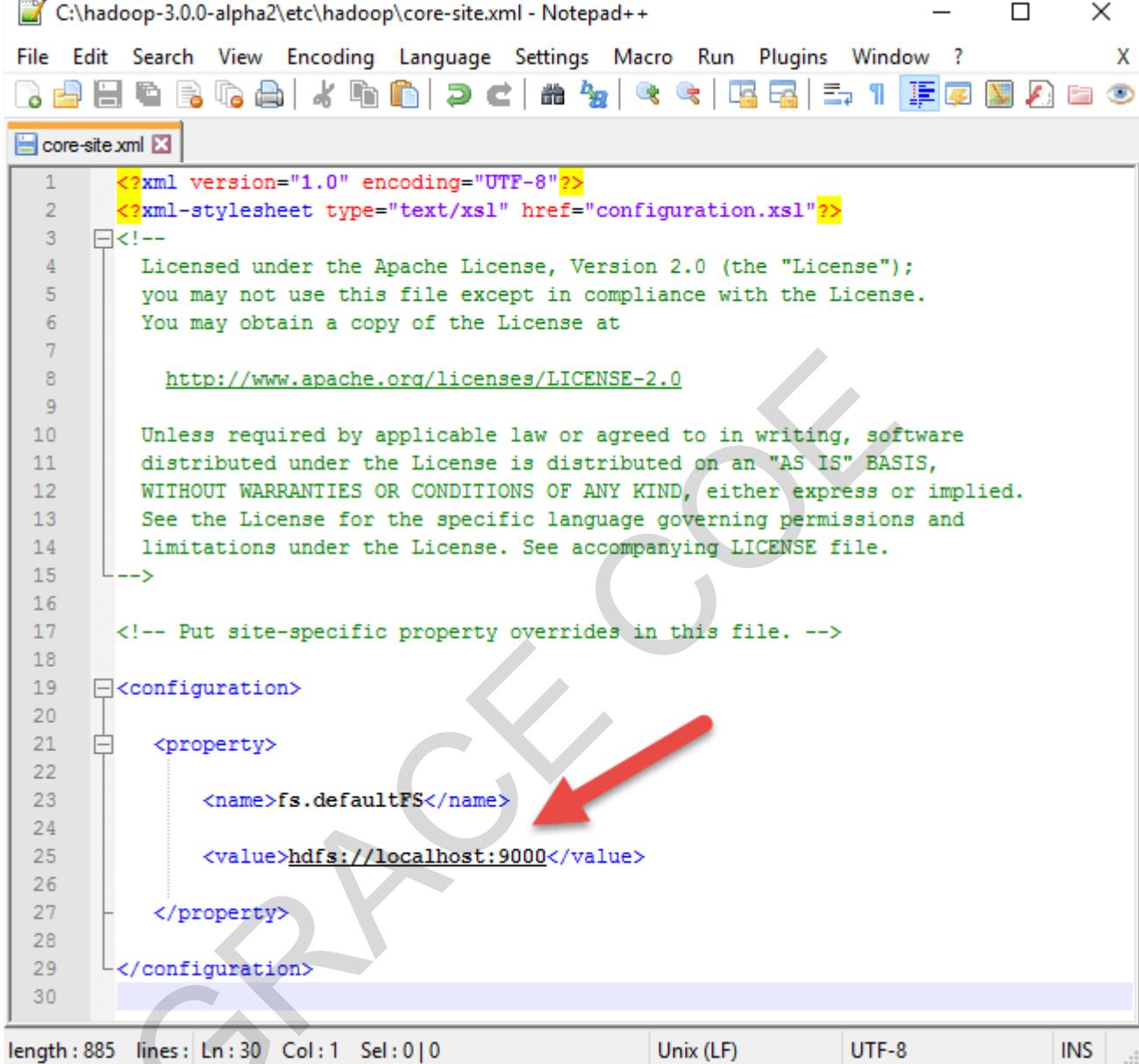


Then added a 'C:\hadoop-3.0.0-alpha2\bin' path like this and pressed OK:



Edit Hadoop Configuration

C:\hadoop-3.0.0-alpha2\etc\hadoop\core-site.xml file, just like this:



C:\hadoop-3.0.0-alpha2\etc\hadoop\core-site.xml - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ?

core-site.xml

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
3  <!--
4      Licensed under the Apache License, Version 2.0 (the "License");
5      you may not use this file except in compliance with the License.
6      You may obtain a copy of the License at
7
8          http://www.apache.org/licenses/LICENSE-2.0
9
10     Unless required by applicable law or agreed to in writing, software
11     distributed under the License is distributed on an "AS IS" BASIS,
12     WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
13     See the License for the specific language governing permissions and
14     limitations under the License. See accompanying LICENSE file.
15 -->
16
17     <!-- Put site-specific property overrides in this file. -->
18
19 <configuration>
20
21     <property>
22
23         <name>fs.defaultFS</name>
24
25         <value>hdfs://localhost:9000</value>
26
27     </property>
28
29 </configuration>
30

```

length : 885 lines : Ln : 30 Col : 1 Sel : 0 | 0 Unix (LF) UTF-8 INS ::

Next go to C:\hadoop-3.0.0-alpha2\etc\hadoop folder and renamed mapred-site.xml.template to mapred-site.xml.

edited the mapred-site.xml file adding the following XML Yarn configuration for Mapreduce:

<configuration>

<property>

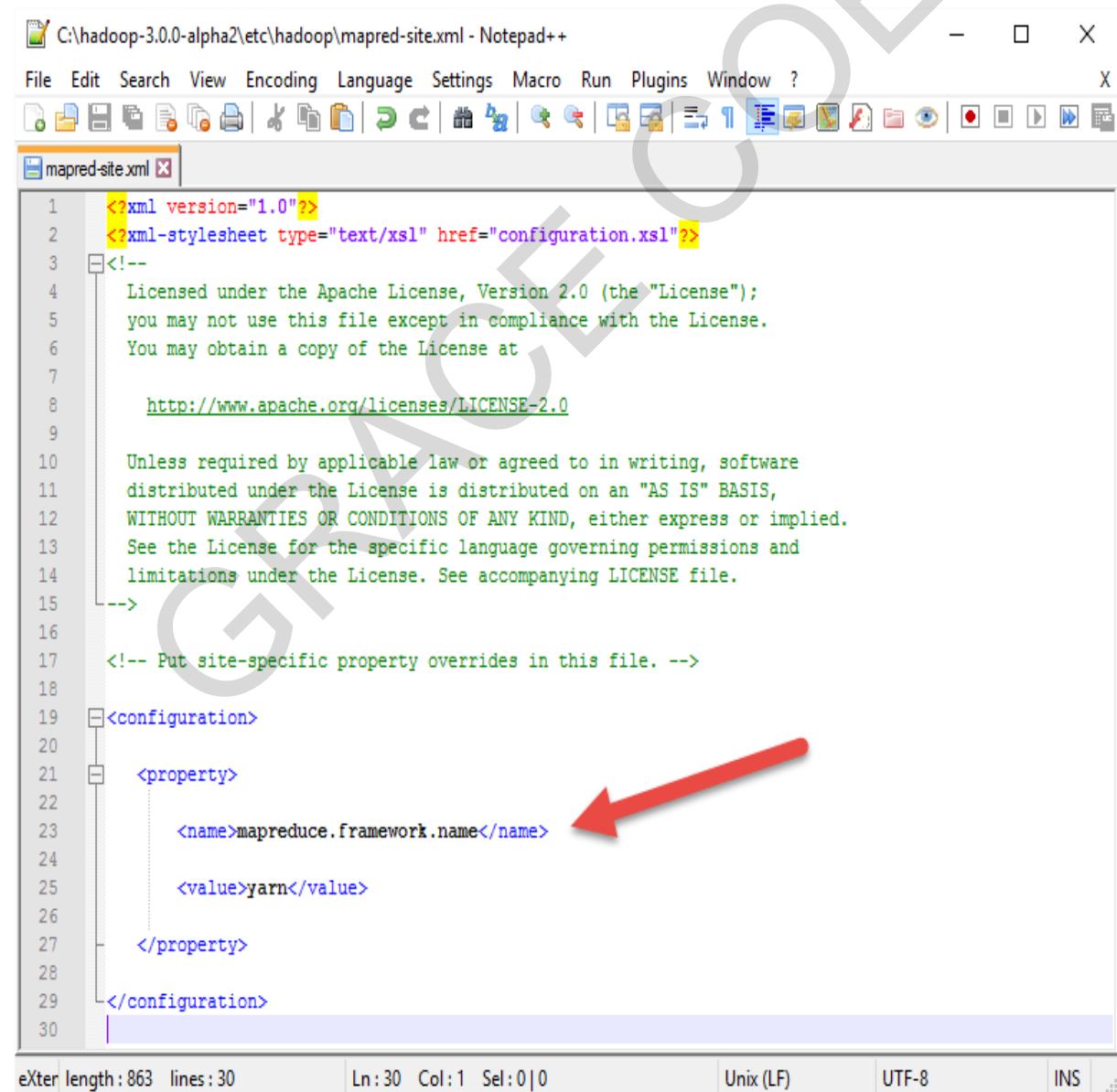
```
<name>mapreduce.framework.name</name>

<value>yarn</value>

</property>

</configuration>
```

This is what the file looks like when configured:



C:\hadoop-3.0.0-alpha2\etc\hadoop\mapred-site.xml - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ?

mapred-site.xml

```
<?xml version="1.0"?>
<?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
<!-- Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at
http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License. See accompanying LICENSE file.
--&gt;
<!-- Put site-specific property overrides in this file. --&gt;
&lt;configuration&gt;
&lt;property&gt;
&lt;name&gt;mapreduce.framework.name&lt;/name&gt;
&lt;value&gt;yarn&lt;/value&gt;
&lt;/property&gt;
&lt;/configuration&gt;</pre>

eXter length:863 lines:30 Ln:30 Col:1 Sel:0|0 Unix (LF) UTF-8 INS


```

The next step was to created a new ‘data’ folder in Hadoop’s home directory (C:\hadoop-3.0.0-alpha2\data).

Once done, the next step was to add a data node and name node to Hadoop, by editing c:\hadoop-3.0.0-alpha2\etc\hadoop\hdfs-site.xml file.

And added following configuration to this XML file:

configuration>

```
<property> <name>dfs.replication</name> <value>1</value> </property> <property>
<name>dfs.namenode.name.dir</name>
<value>C:/hadoop-3.0.0-alpha2/data/namenode</value> </property><property> <name>dfs.datanode.data.dir</name>
<value>C:/hadoop-3.0.0-alpha2/data/datanode</value> </property></configuration>
```

In above step, I had to make sure that I am pointing to location of my newly created data folder and append the datanode and namenode as shown in example.

This is what hdfs-site.xml file looked like once completed:

```

1  <?xml version="1.0" encoding="UTF-8"?>
2  <?xml-stylesheet type="text/xsl" href="configuration.xsl"?>
3  <!--
4      Licensed under the Apache License, Version 2.0 (the "License");
5      you may not use this file except in compliance with the License.
6      You may obtain a copy of the License at
7
8          http://www.apache.org/licenses/LICENSE-2.0
9
10     Unless required by applicable law or agreed to in writing, software
11     distributed under the License is distributed on an "AS IS" BASIS,
12     WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
13     See the License for the specific language governing permissions and
14     limitations under the License. See accompanying LICENSE file.
15 -->
16
17  <!-- Put site-specific property overrides in this file. -->
18
19 <configuration>
20   <property>
21     <name>dfs.replication</name>
22     <value>1</value>
23   </property>
24
25   <property>
26     <name>dfs.namenode.name.dir</name>
27     <value>C:/hadoop-3.0.0-alpha2/data/namenode</value>
28   </property>
29
30   <property>
31     <name>dfs.datanode.data.dir</name>
32     <value>C:/hadoop-3.0.0-alpha2/data/datanode</value>
33   </property>
34 </configuration>
35

```

eXter length : 1,123 lines : 35 Ln:35 Col:1 Sel:0|0 Unix (LF) UTF-8 INS ..

The next step was to add site specific YARN configuration properties by editing yarn-site.xml at

C:\hadoop-3.0.0-alpha2\etc\hadoop\yarn-site.xml, like this:

```

<configuration> <property>

<name>yarn.nodemanager.aux-services</name>

<value>mapreduce_shuffle</value> </property>

<property>

<name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>

<value>org.apache.hadoop.mapred.ShuffleHandler</value>

```

```
</property>
</configuration>
```

This is what yarn-site.xml file looked like once completed:



C:\hadoop-3.0.0-alpha2\etc\hadoop\yarn-site.xml - Notepad++

File Edit Search View Encoding Language Settings Macro Run Plugins Window ?

yarn-site.xml

```

1  <?xml version="1.0"?>
2  <!--
3      Licensed under the Apache License, Version 2.0 (the "License");
4      you may not use this file except in compliance with the License.
5      You may obtain a copy of the License at
6
7          http://www.apache.org/licenses/LICENSE-2.0
8
9      Unless required by applicable law or agreed to in writing, software
10     distributed under the License is distributed on an "AS IS" BASIS,
11     WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
12     See the License for the specific language governing permissions and
13     limitations under the License. See accompanying LICENSE file.
14 -->
15
16 <configuration>
17
18     <property>
19
20         <name>yarn.nodemanager.aux-services</name>
21
22         <value>mapreduce_shuffle</value>
23
24     </property>
25
26     <property>
27
28         <name>yarn.nodemanager.aux-services.mapreduce.shuffle.class</name>
29
30         <value>org.apache.hadoop.mapred.ShuffleHandler</value>
31
32     </property>
33
34 </configuration>
35

```

eXter length : 929 lines : 35 | Ln : 35 Col : 1 Sel : 0 | 0 Unix (LF) | UTF-8 | INS | ...

Then I continued by editing hadoop-env.cmd in C:\hadoop-3.0.0-alpha2\etc\hadoop\hadoop-env.cmd. Then

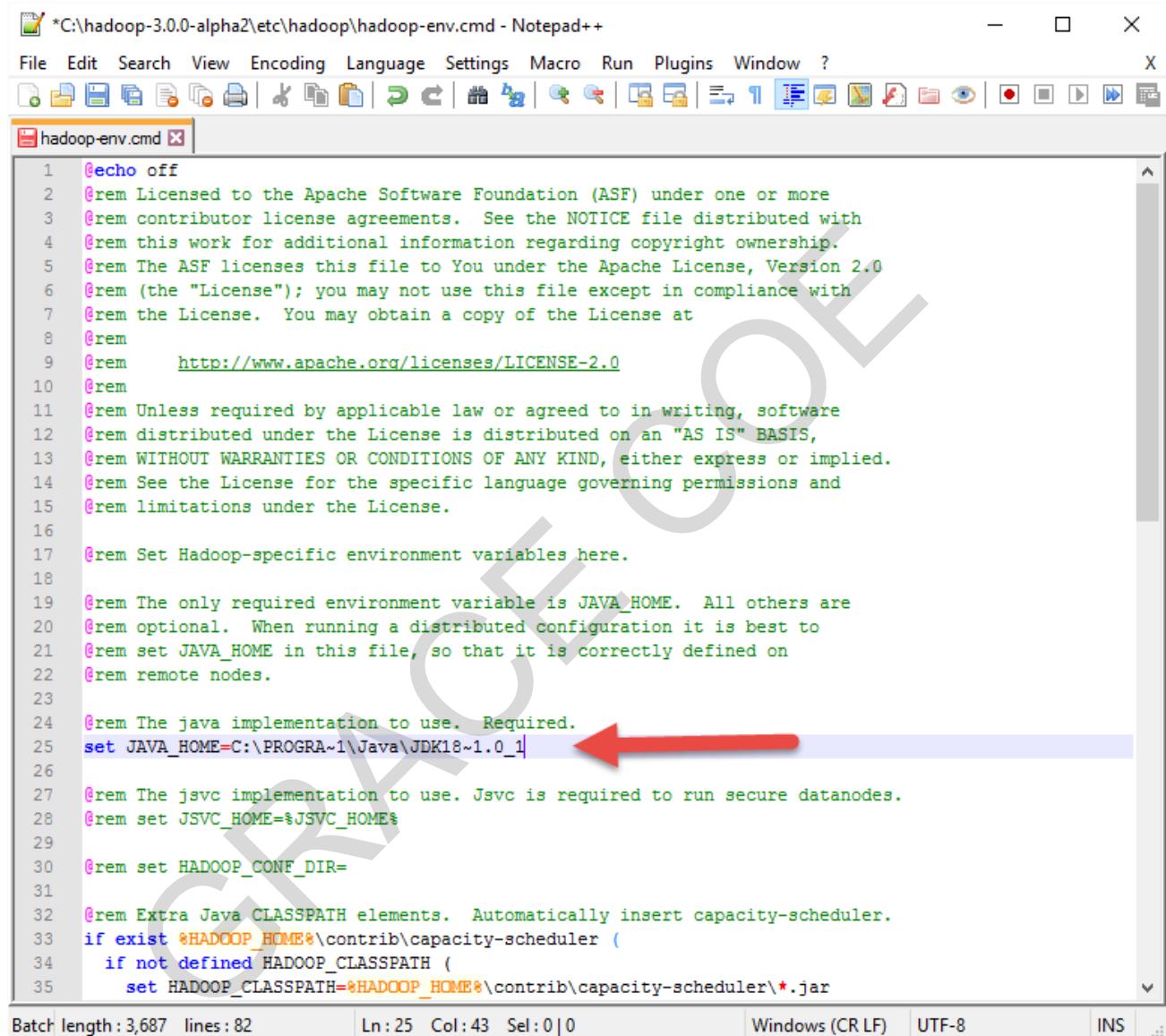
changed the line for JAVA_HOME=%JAVA_HOME% and added a path to my JAVA folder: C:\PROGRA~1\Java\JDK18~1.0_1

Go to C:\Program Files\Java\jdk1.8.0_111 where my Java JDK is installed and converted a long path to

windows short name:

```
C:\Program Files\Java\jdk1.8.0_111>for %I in(.) do echo %~sI
C:\Program Files\Java\jdk1.8.0_111>echo C:\PROGRA~1\Java\JDK18~1.0_1
C:\PROGRA~1\Java\JDK18~1.0_1
```

Next step was to open hadoop-env.cmd and add it in there, as shown in this screenshot:



The screenshot shows a Notepad++ window with the file 'hadoop-env.cmd' open. The code in the file is as follows:

```

1  @echo off
2  @rem Licensed to the Apache Software Foundation (ASF) under one or more
3  @rem contributor license agreements. See the NOTICE file distributed with
4  @rem this work for additional information regarding copyright ownership.
5  @rem The ASF licenses this file to You under the Apache License, Version 2.0
6  @rem (the "License"); you may not use this file except in compliance with
7  @rem the License. You may obtain a copy of the License at
8  @rem
9  @rem     http://www.apache.org/licenses/LICENSE-2.0
10 @rem
11 @rem Unless required by applicable law or agreed to in writing, software
12 @rem distributed under the License is distributed on an "AS IS" BASIS,
13 @rem WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
14 @rem See the License for the specific language governing permissions and
15 @rem limitations under the License.
16
17 @rem Set Hadoop-specific environment variables here.
18
19 @rem The only required environment variable is JAVA_HOME. All others are
20 @rem optional. When running a distributed configuration it is best to
21 @rem set JAVA_HOME in this file, so that it is correctly defined on
22 @rem remote nodes.
23
24 @rem The java implementation to use. Required.
25 set JAVA_HOME=C:\PROGRA~1\Java\JDK18~1.0_1
26
27 @rem The jsvc implementation to use. Jsvc is required to run secure datanodes.
28 @rem set JSVC_HOME=%JSVC_HOME%
29
30 @rem set HADOOP_CONF_DIR=
31
32 @rem Extra Java CLASSPATH elements. Automatically insert capacity-scheduler.
33 if exist %HADOOP_HOME%\contrib\capacity-scheduler (
34   if not defined HADOOP_CLASSPATH (
35     set HADOOP_CLASSPATH=%HADOOP_HOME%\contrib\capacity-scheduler\*.jar

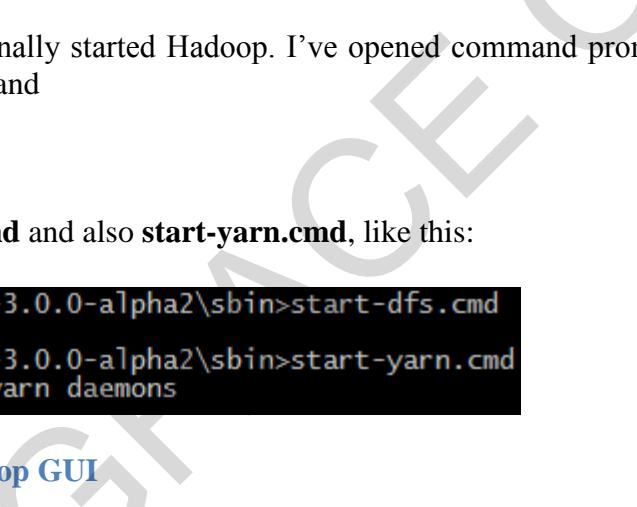
```

A red arrow points to the line 'set JAVA_HOME=C:\PROGRA~1\Java\JDK18~1.0_1'.

Next in C:\hadoop-3.0.0-alpha2\bin using windows command prompt as admin run:

'hdfs namenode -format' command.

Output looked like this:



```

Administrator: Command Prompt
2017-02-01 19:13:48,239 WARN namenode.FSEditLog: No class configured for C, dfs.
namenode.edits.journal-plugin.C is empty
2017-02-01 19:13:48,239 ERROR namenode.NameNode: Failed to start namenode.
java.lang.IllegalArgumentException: No class configured for C
        at org.apache.hadoop.hdfs.server.namenode.FSEditLog.getJournalClass(FSEditLog.java:1751)
        at org.apache.hadoop.hdfs.server.namenode.FSEditLog.createJournal(FSEditLog.java:1766)
        at org.apache.hadoop.hdfs.server.namenode.FSEditLog.initJournals(FSEditLog.java:291)
        at org.apache.hadoop.hdfs.server.namenode.FSEditLog.initJournalsForWrite(FSEditLog.java:256)
        at org.apache.hadoop.hdfs.server.namenode.NameNode.format(NameNode.java:1138)
        at org.apache.hadoop.hdfs.server.namenode.NameNode.createNameNode(NameNode.java:1584)
        at org.apache.hadoop.hdfs.server.namenode.NameNode.main(NameNode.java:1704)
2017-02-01 19:13:48,245 INFO util.ExitUtil: Exiting with status 1
2017-02-01 19:13:48,249 INFO namenode.NameNode: SHUTDOWN_MSG:
/*****SHUTDOWN_MSG: Shutting down NameNode at DESKTOP-LN840EP/192.168.17.1*****
*****/
```

C:\hadoop-3.0.0-alpha2\bin>

Then I've finally started Hadoop. I've opened command prompt as admin in C:\hadoop-3.0.0-alpha2\sbin and

ran

start-dfs.cmd and also **start-yarn.cmd**, like this:

```

C:\hadoop-3.0.0-alpha2\sbin>start-dfs.cmd
C:\hadoop-3.0.0-alpha2\sbin>start-yarn.cmd
starting yarn daemons
```

Open Hadoop GUI

Once all above steps were completed, I've opened browser and navigated to: <http://localhost:8088/cluster>

The screenshot shows the Hadoop Cluster UI at localhost:8088/cluster. The title bar says "All Applications". The main header features the Hadoop logo and the title "All Applications". A sidebar on the left lists cluster metrics like "About Nodes", "Node Labels", and "Applications" (status: NEW, NEW SAVING, SUBMITTED, ACCEPTED, RUNNING, FINISHED, FAILED, KILLED). Below the sidebar is a "Scheduler" section and a "Tools" section. The main content area displays several tables of metrics:

- Cluster Metrics:**

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved
0	0	0	0	0	0 B	0 B	0 B	0	0	0
- Cluster Nodes Metrics:**

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes	Shutdown Nodes
0	0	0	0	0	0	0
- Scheduler Metrics:**

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	Maximum Cluster Application Priority
Capacity Scheduler	[MEMORY]	<memory:1024, vCores:1>	<memory:8192, vCores:4>	0
- Table Headers:**

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	% of Queue	% of Cluster	Progress	Tracking UI	Blacklisted Nodes
----	------	------	------------------	-------	----------------------	-----------	------------	-------	-------------	--------------------	----------------------	---------------------	------------	--------------	----------	-------------	-------------------
- Table Body:**

No data available in table

Showing 0 to 0 of 0 entries

WORD COUNT PROGRAM:

```

WordCount
package com.jarosciak.jozef;
import java.io.IOException;
import java.util.*;

import org.apache.hadoop.fs.Path;
import org.apache.hadoop.conf.*;
import org.apache.hadoop.io.*;
import org.apache.hadoop.mapreduce.*;
import org.apache.hadoop.mapreduce.lib.input.FileInputFormat;
import org.apache.hadoop.mapreduce.lib.input.TextInputFormat;
import org.apache.hadoop.mapreduce.lib.output.FileOutputFormat;
import org.apache.hadoop.mapreduce.lib.output.TextOutputFormat;

public class WordCount {

    public static class Map extends Mapper<LongWritable, Text, Text, IntWritable> {
        private final static IntWritable one = new IntWritable(1);
        private Text word = new Text();

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

    public static class Reduce extends Reducer<Text, IntWritable, Text, IntWritable> {

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

```

Then go to <https://www.randomlists.com/random-words> to create couple of random words:

https://www.randomlists.com/random-words

1	untidy	2	strengthen
3	command	4	unkempt
9	reach	10	moor
13	van	14	defeated
17	hollow	18	reply
21	kindly	22	inform
25	taste	26	trip
29	thumb	30	identify
33	jumpy	34	gabby
37	achiever	38	license
41	pull	42	lackadaisical
45	tart	46	cross
49	decorate	50	border
53	watch	54	understood
57	tender	58	moor
61	preach	62	noisy
65	colour	66	rule
69	bleach	70	marble
73	wasteful	74	pot
77	stop	78	broad
81	radiate	82	neat
85	object	86	well-to-do
89	bashful	90	mark
93	orange	94	air
97	release	98	flap
			3. blot
			4. expect
			7. right
			8. succinct
			11. seed
			12. approval
			15. lake
			16. jobless
			19. awesome
			20. bubble
			23. descriptive
			24. wood
			27. request
			30. action
			33. sore
			36. abusive
			39. squash
			42. flock
			45. confess
			48. pour
			51. measure
			54. black
			57. glistening
			60. stir
			63. trite
			66. suggest
			70. basket
			73. shoes
			76. bounce
			80. fold
			83. male
			86. mug
			90. quarrelsome

Then save words to words.txt,

Running Wordlist against Hadoop's MapReduce

Once I ran my code, it executed and started processing the words.txt file that was prior to execution copied to

input folder (which I created earlier together with the output folder for the outcome files).

Following was the result of Hadoop's processing job:

```

17/02/01 20:13:39 INFO mapreduce.Job: Running job: job_1486004321196_0006
17/02/01 20:13:49 INFO mapreduce.Job: Job job_1486004321196_0006 running in uber mode : false
17/02/01 20:13:49 INFO mapreduce.Job: map 0% reduce 0%
17/02/01 20:13:56 INFO mapreduce.Job: map 100% reduce 0%
17/02/01 20:14:04 INFO mapreduce.Job: map 100% reduce 100%
17/02/01 20:14:04 INFO mapreduce.Job: Job job_1486004321196_0006 completed successfully
17/02/01 20:14:04 INFO mapreduce.Job: Counters: 49
  File System Counters
    FILE: Number of bytes read=1248
    FILE: Number of bytes written=235851
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=847
    HDFS: Number of bytes written=858
    HDFS: Number of read operations=6
    HDFS: Number of large read operations=0
    HDFS: Number of write operations=2
  Job Counters
    Launched map tasks=1
    Launched reduce tasks=1
    Data-local map tasks=1
    Total time spent by all maps in occupied slots (ms)=4862
    Total time spent by all reduces in occupied slots (ms)=5706
    Total time spent by all map tasks (ms)=4862
    Total time spent by all reduce tasks (ms)=5706
    Total vcore-seconds taken by all map tasks=4862
    Total vcore-seconds taken by all reduce tasks=5706
    Total megabyte-seconds taken by all map tasks=4978688
    Total megabyte-seconds taken by all reduce tasks=5842944
  Map-Reduce Framework
    Map input records=1
    Map output records=101
    Map output bytes=1115
    Map output materialized bytes=1248
    Input split bytes=136
    Combine input records=101
    Combine output records=96
    Reduce input groups=96
    Reduce shuffle bytes=1248
    Reduce input records=96
    Reduce output records=96
    Spilled Records=192
    Shuffled Maps =1
    Failed Shuffles=0
    Merged Map outputs=1
    GC time elapsed (ms)=123
    CPU time spent (ms)=1330
    Physical memory (bytes) snapshot=343425024
    Virtual memory (bytes) snapshot=3007807488
    Total committed heap usage (bytes)=226365440
  Shuffle Errors
    BAD_ID=0
    CONNECTION=0
    IO_ERROR=0
    WRONG_LENGTH=0
    WRONG_MAP=0
    WRONG_REDUCE=0
  File Input Format Counters
    Bytes Read=711
  File Output Format Counters
    Bytes Written=858

```

We can see the job progress in the browser as well:



MapReduce Job job_1486004321196_0006

Logged in as: dr.who

Job Overview

Job Name:	word count
User Name:	dr.who
Queue:	root
State:	SUCCEEDED
Uberized:	false
Submitted:	Wed Feb 01 20:13:39 PST 2017
Started:	Wed Feb 01 20:13:47 PST 2017
Finished:	Wed Feb 01 20:14:02 PST 2017
Elapsed:	15sec
Diagnostics:	
Average Map Time	4sec
Average Shuffle Time	4sec
Average Merge Time	0sec
Average Reduce Time	0sec

ApplicationMaster

Attempt Number	Start Time	Node	Logs
1	Wed Feb 01 20:13:41 PST 2017	logs	logs

Task Type	Total	Complete
Map	1	1
Reduce	1	1

Attempt Type	Failed	Killed	Successful
Maps	0	0	1
Reduces	0	0	1

OUTPUT:

1. abusive 1
2. achiever 1
3. action 1
4. admit 1
5. air 1
6. angry 1
7. approval 1
8. awesome 1
9. bashful 1
10. basket 1
11. black 1
12. bleach 1
13. blood 1
14. blot 1
15. border 1
16. bounce 1
17. broad 1
18. bubble 1
19. changeable 1
20. cloth 1
21. colour 1
22. command 1

- 23. confess 1
- 24. cough 1
- 25. cross 1
- 26. dark 1
- 27. decorate 1
- 28. defeated 1
- 29. descriptive 1
- 30. elite 1
- 31. expect 1
- 32. flap 1
- 33. flock 1
- 34. fold 1
- 35. friction 1
- 36. gabby 1
- 37. hollow 1
- 38. identify 1
- 39. inform 1
- 40. irritating 1
- 41. **jarosciak 5**
- 42. jobless 1
- 43. jumpy 1
- 44. kindly 1
- 45. lackadaisical 1
- 46. lake 1
- 47. license 1
- 48. male 1
- 49. marble 1
- 50. mark 1
- 51. measure 1
- 52. moor 2
- 53. mug 1
- 54. neat 1
- 55. noisy 1
- 56. object 1
- 57. orange 1
- 58. peck 1
- 59. pot 1
- 60. pour 1
- 61. preach 1
- 62. pull 1
- 63. quarrelsome 1
- 64. radiate 1
- 65. reach 1
- 66. rebel 1
- 67. release 1
- 68. reply 1
- 69. request 1
- 70. right 1
- 71. scold 1

72. seed 1
73. sheet 1
74. shoes 1
75. smash 1
76. sore 1
77. squash 1
78. stir 1
79. stop 1
80. strengthen 1
81. succinct 1
82. suggest 1
83. tart 1
84. taste 1
85. thumb 1
86. trip 1
87. trite 1
88. understood 1
89. unique 1
90. unkempt 1
91. untidy 1
92. van 1
93. wasteful 1
94. watch 1
95. well-to-do 1
96. wood 1

RESULT:

Thus a procedure to install single node Hadoop cluster was successfully executed.