Savitribai Phule Pune University, Pune

Third Year Information Technology (2019 Course)

314458: Laboratory Practice-II

(Cloud Computing)

Savitribai Phule Pune University, Pune
Third Year Information Technology (2019 Course)
314458: Laboratory Practice-II (Cloud Computing)

Teaching Scheme:	Credit Scheme:	Examination Scheme:
Practical (PR): 04 hrs/week	02 Credit	PR :25 Marks TW : 50Marks

Prerequisites:

- Basics of Computer Networks
- Operating Systems

Course Objectives:

1. To develop web applications in cloud.
2. To learn the design and development process involved in creating a cloud based application.

Course Outcomes :

On completion of the course, students will be able to–

CO1: To design and develop cloud-based applications.
CO2: To Simulate a cloud scenario using CloudSim.
CO3: To design and deploy web applications in cloud environment

List of Laboratory Assignments

1. Install Google App Engine. Create hello world app and other simple web applications using python/java.
2. Use GAE launcher to launch the web applications.
3. Simulate a cloud scenario using CloudSim and run a scheduling algorithm that is not present in CloudSim.
4. Find a procedure to transfer the files from one virtual machine to another virtual machine.
5. Find a procedure to launch virtual machine using trystack (Online Openstack Demo Version)
6. Design and deploy a web application in a PaaS environment.
7. Design and develop custom Application (Mini Project) using Salesforce Cloud.
8. Design an Assignment to retrieve, verify, and store user credentials using Firebase Authentication, the Google App Engine standard environment, and Google Cloud Data store.

CASE STUDIES

- Data storage security in private cloud
- Application of IoT/Ubiquitous based on cloud

- Tools for building private cloud

Text Books:

1. Thomas Erl, Zaigham Mahmood and Ricardo Puttini, Cloud Computing: Concepts, Technology & Architecture, Pearson, ISBN :978 9332535923, 9332535922, 1 st Edition
2. Anthony T. Velte Toby J. Velte, Robert Elsenpeter, “Cloud Computing: A Practical Approach”.

Reference Books:

1. Rajkumar Buyya, Christian Vecchiola, S. ThamaraiS elvi, Mastering Cloud Computing: Foundationsand Applications Programming, McGraw Hill, ISBN: 978 1259029950, 1259029956.
2. Gautam Shrof, “ENTERPRISE CLOUD COMPUTING Technology Architecture, Applications, Cambridge University Press, ISBN: 9780511778476
3. Srinivasan, J. Suresh, Cloud Computing: A practical approach for learning and implementation, Pearson, ISBN :9788131776513.
4. Jack J. Dongarra, Kai Hwang, Geoffrey C. Fox, Distributed and Cloud Computing: From Parallel Processing to the Internet of Things, Elsevier, ISBN :9789381269237, 9381269238, 1st Edition.
5. Brian J.S. Chee and Curtis Franklin, Jr., Cloud Computing: Technologies and Strategies of the Ubiquitous Data Center, CRC Press, ISBN :9781439806128.
6. Kris Jamsa, Cloud Computing: Saas, Paas, Iaas, Virtualization, Business Models, Mobile, Security,and More, Jones and Bartlett, ISBN :9789380853772.
7. John W. Ritting house, James F. Ransome, Cloud Computing Implementation, Management, and Security, CRC Press, ISBN : 978 1439806807, 1439806802.
8. Karl Matthias, Sean P. Kane, Docker: Up and Running, OReilly, ISBN:9781491917572,1491917571.
9. Barrie Sosinsky, Cloud Computing Bible, Wiley, ISBN: 978 8126529803.
10. Ronald L. Krutz and Russell D. Vines, Cloud Security: A Comprehensive guide to Secure Cloud Computing, Wiley, ISBN: 9788126528097.
11. Scott Adkins, John Belamaric, Vincent Giersch, Denys Makogon, Jason E. Robinson, OpenStack: Cloud Application Development, Wrox, ISBN :9781119194316.

12. Kailash Jayaswal, Jagannath Kallakurchi, Donald J. Houde, Cloud Computing Black Book ,Wiley Dreamtech,ISBN:9789351194187

Assignment 1

Title

Install Google App Engine. Create hello world app and other simple web applications using Python/Java.

Requirements

1. Google App Engine
2. Python Interpreter (Python2.7.x)
3. Text Editor
4. Browser

Theory

A) Google App Engine

1. Google App Engine (GAE) is a platform-as-a-service product that provides web app developers and enterprises with access to Google's scalable hosting and tier 1 internet service.
2. GAE requires that applications be written in Java or Python, store data in Google Bigtable and use the Google query language.
3. Noncompliant applications require modification to use GAE.
4. GAE provides more infrastructure than other scalable hosting services, such as Amazon Elastic Compute Cloud (EC2).

5. GAE also eliminates some system administration and development tasks to make writing scalable applications easier.
6. Google provides GAE free up to a certain amount of use for resources like CPU, storage, API calls and concurrent requests

B) Google Cloud SDK

1. Google Cloud SDK (Software Development Kit), in simple terms, is a set of tools that are used to manage applications and resources that are hosted on the Google Cloud Platform.
2. It is composed of the gsutil, gcloud, and bqcommand line tools.
3. The gcloudtool is automatically downloaded with the Cloud SDK.
4. Google Cloud SDK run on specific platforms – Windows, Linux, and macOS and requires Python 2.7.x.
5. SDK might have further necessities like Java tools used for the development of Google App Engine needs Java 1.7 or the later one.
6. It can be used to locally deploy and test web applications.

C) Directory Structure for creating hello world application

1. The web applications to be deployed can be organized in the following directory structure

```
root_directory
        |_______templates
        |              |______index.html
        |_______static
        |_______main.py
        |_______app.yaml
```

2. The templates directory can be used to store the web templates of the web application (HTML files).
3. The static directory can be used to store the web static files which contain the styling and the business logic data for the web application (CSS and JS files).
4. The main.py is used to define the routes, rendering logic, data acquisition logic.
5. It provides the WSGI abstraction to the application.
6. The app.yaml file provides the runtime environment, URLs for routes and launch configuration of the application in the form of key value pairs.

Steps

A) Install Google Cloud SDK on Windows or Linux machines

1. Visit the https://cloud.google.com/sdk/docs/install link to download the CLI (Command line interface) tool for the Cloud SDK.
2. Select the appropriate operating system from the installation manual
3. Follow the provided instructions in the displayed section
   a) For Windows users, the executable downloader is provided for downloading.
   b) For Ubuntu and Fedora users, terminal commands for installation are provided using apt and dnf repositories respectively.

B) Creating the application

1. The application must be initialized using the above-mentioned directory structure.
2. It is a recommended format for organization and readability of code.
3. The app.yaml file should contain the following content:

   Contents of **app.yaml**

```
runtime : python2
api_version : 1
threadsafe : true

handlers
- url : /
  script : main.app
```

4. The logic of the application, i.e. the Web server interaction code of the application must be placed in the main.py file.
5. A simple code displaying the hello world on a web page is as follows

Contents of main.py for Hello World application

```
import webapp2

class MainPage(webapp2.RequestHandler) :
```

```
        def get(self):
            self.response.write("Hello World")

app = webapp2.WSGIApplication(
    [("/", MainPage)],
    debug=True
)
```

6. Finally, after saving the above code, the application can be run on the localhost server using the following command. (The command must be run on the Google Cloud Shell or the terminal in case of Ubuntu).

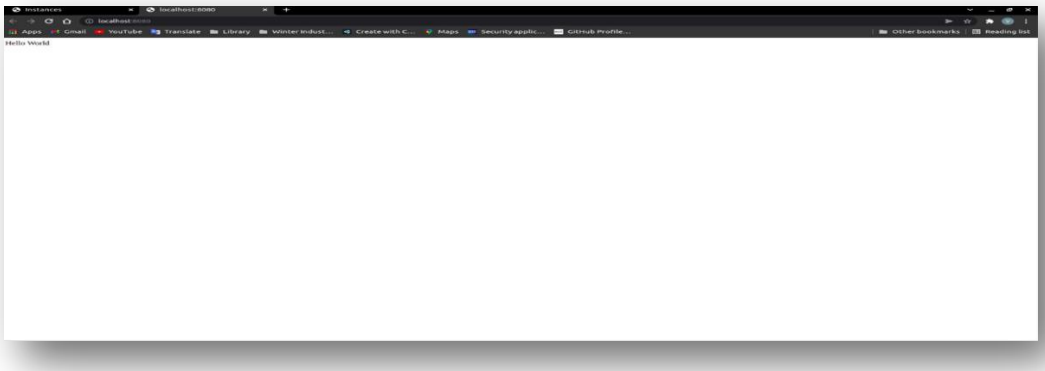
Command:
python <path_to_sdk>/bin/devappserver.py <path_to_application_directory>

Sample Output

1. The application, if no errors are found, is launched on the port 8080 of the localhost server.
2. The cloud console is visible on port 8000 of the localhost server.
3. The URL of localhost:8080 can be typed in the address bar of the browser to view the application

4. Screenshots:

a) Application launch at port 8080



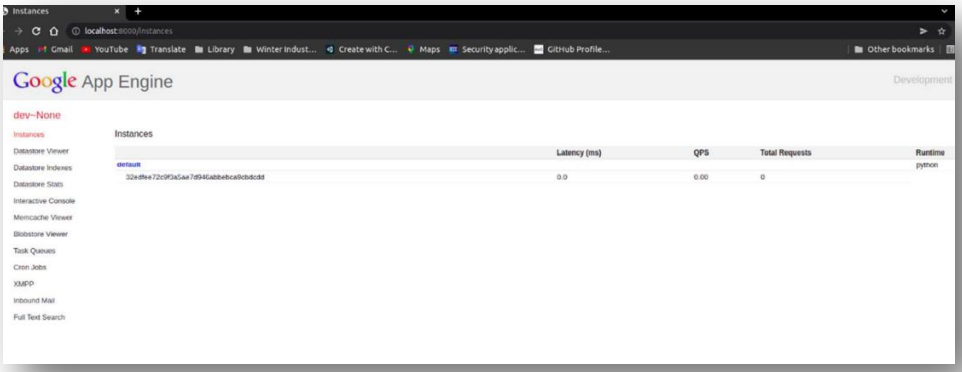
b) Terminal / Command Line prompts

```
varadmash@varadmash-G3-3590:~$ python3 ./google-cloud-sdk/bin/dev_appserver.py ./cloud_computing_lab/Assignment1/


Updates are available for some Cloud SDK components.  To install them,
please run:
  $ gcloud components update

WARNING  2022-01-16 07:03:14,121 application_configuration.py:210] The "python" runtime specified in "./cloud_computing_lab/Assignment1/app.yml" is not supported
tead. A description of the differences between the two can be found here:
https://developers.google.com/appengine/docs/python/python25/diff27
INFO     2022-01-16 07:03:14,123 devappserver2.py:316] Skipping SDK update check.
WARNING  2022-01-16 07:03:15,018 simple_search_stub.py:1196] Could not read search indexes from /tmp/appengine.None.varadmash/search_indexes
INFO     2022-01-16 07:03:15,021 <string>:383] Starting API server at: http://localhost:33917
INFO     2022-01-16 07:03:15,155 dispatcher.py:281] Starting module "default" running at: http://localhost:8080
INFO     2022-01-16 07:03:15,157 admin_server.py:150] Starting admin server at: http://localhost:8000
INFO     2022-01-16 07:03:21,172 instance.py:294] Instance PID: 3381
```

c) Cloud Console at port 8000



Assignment 2

Title

Use GAE launcher to launch the web applications.

Requirements

1. Google App Engine
2. Python Interpreter (Python2.7.x)
3. Browser

Theory

A) Google App Engine

1. Google App Engine (GAE) is a platform-as-a-service product that provides web app developers and enterprises with access to Google's scalable hosting and tier 1 internet service.
2. GAE requires that applications be written in Java or Python, store data in Google Bigtable and use the Google query language.
3. Noncompliant applications require modification to use GAE.
4. GAE provides more infrastructure than other scalable hosting services, such as Amazon Elastic Compute Cloud (EC2).
5. GAE also eliminates some system administration and development tasks to make writing scalable applications easier.
6. Google provides GAE free up to a certain amount of use for resources like CPU, storage, API calls and concurrent requests

B) Directory Structure for creating web application

1. The web applications to be deployed can be organized in the following directory structure

    a. root_directory
        1. |_____templates
        2. | |_____index.html
        3. | |_____results.html
        4. |_____static
        5. |_____main.py
        6. |_____app.yaml

2. The templates directory can be used to store the web templates of the web application (HTML files).
3. The static directory can be used to store the web static files which contain the styling and the business logic data for the web application (CSS and JS files).
4. The main.py is used to define the routes, rendering logic, data acquisition logic.
5. It provides the WSGI abstraction to the application.
6. The app.yaml file provides the runtime environment, URLs for routes and launch configuration of the application in the form of key value pairs.

Steps

A) Creating the application

1. The application must be initialized using the above-mentioned directory structure.
2. It is a recommended format for organization and readability of code.
3. Create index.html and results.html as web templates with index.html for taking user input and results.html for displaying response from the API call.
4. The app.yaml file should contain the following content:

   Contents of app.yaml

```
runtime : python2
api_version : 1
threadsafe : true

handlers
url : /
script : main.app
```

5. The logic of the application, i.e. the Web server interaction code of the application must be placed in the main.py file.
6. The following python code sends request for information to the API and interprets response from the API (here, World Time API is used).

Contents of main.py for World Time application

```
import os
import json
import urllib
import webapp2
from google.appengine.ext.webapp import template

class MainPage(webapp2.RequestHandler):
    def get(self):
        template_values = {}
        path =
os.path.join(os.path.dirname(__file__),
'templates/index.html')

self.response.out.write(template.render(path,
template_values))

    def post(self):
```

```
        region = self.request.get('region')
        area = self.request.get('area')
        url =
"http://worldtimeapi.org/api/timezone/"+ region +
"/" + area
        data = urllib.urlopen(url).read()
        data = json.loads(data)
        date = data['datetime'][0:10]
        time = data['datetime'][11:19]
        week = data['day_of_week']
        year = data['day_of_year']
        weeknum = data['week_number']
        template_values = {
            "date": date,
            "time": time,
            "week": week,
            "year": year,
            "weeknum": weeknum,
        }
        path =
os.path.join(os.path.dirname(__file__),
'templates/results.html')

self.response.out.write(template.render(path,
template_values))

app = webapp2.WSGIApplication([('/', MainPage)],
debug=True)
```

7. Finally, after saving the above code, the application can be run on the localhost server using the following command.

Command:

```
python <path_to_sdk>/bin/devappserver.py
<path_to_application_directory>
```

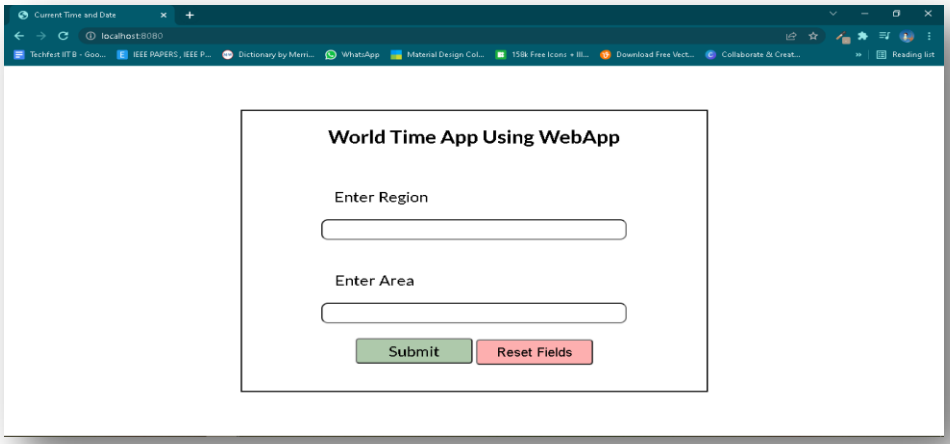
Sample Output

A) The application, if no errors are found, is launched on the port 8080 of the localhost server.
B) The cloud console is visible on port 8000 of the localhost server.

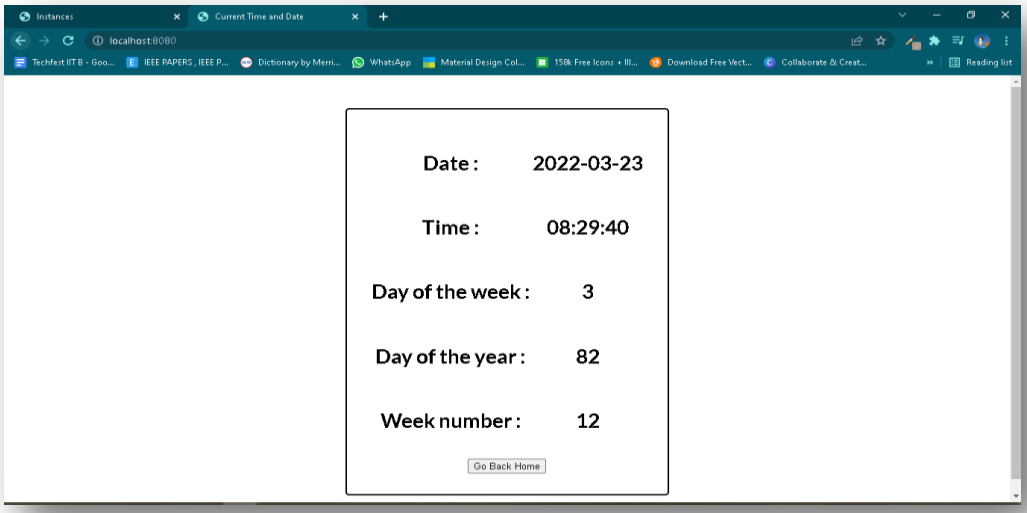
C) The URL of localhost:8080 can be typed in the address bar of the browser to view the application

D) Screenshots:

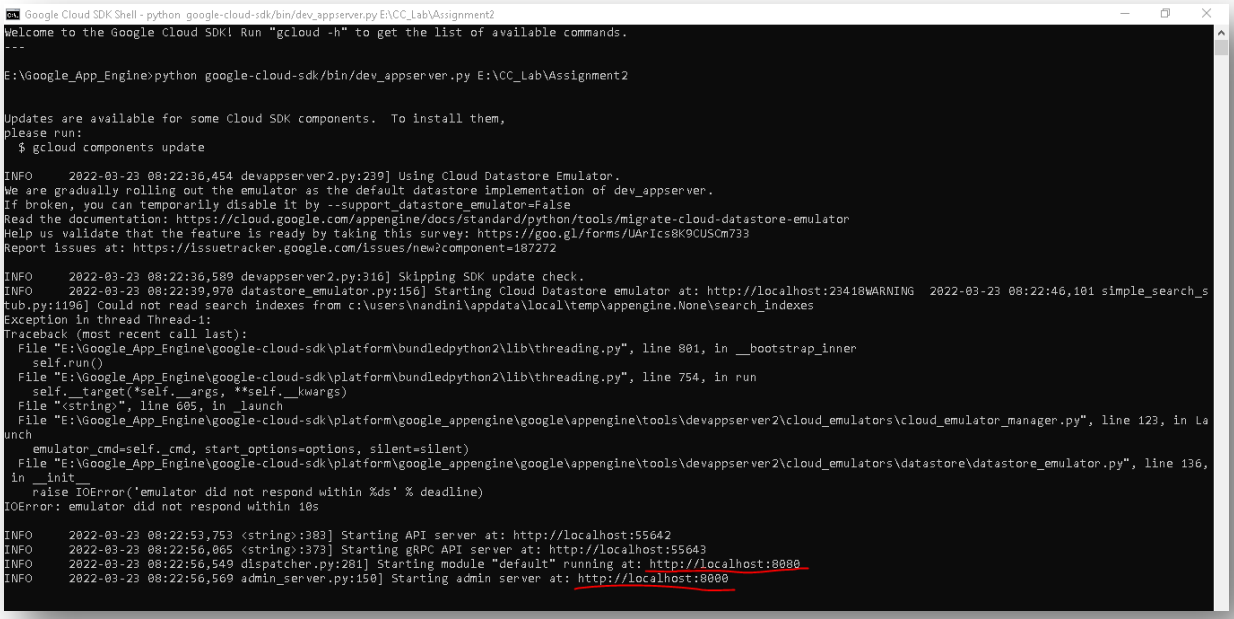
a) Application launch at port 8080 (Displaying index.html)



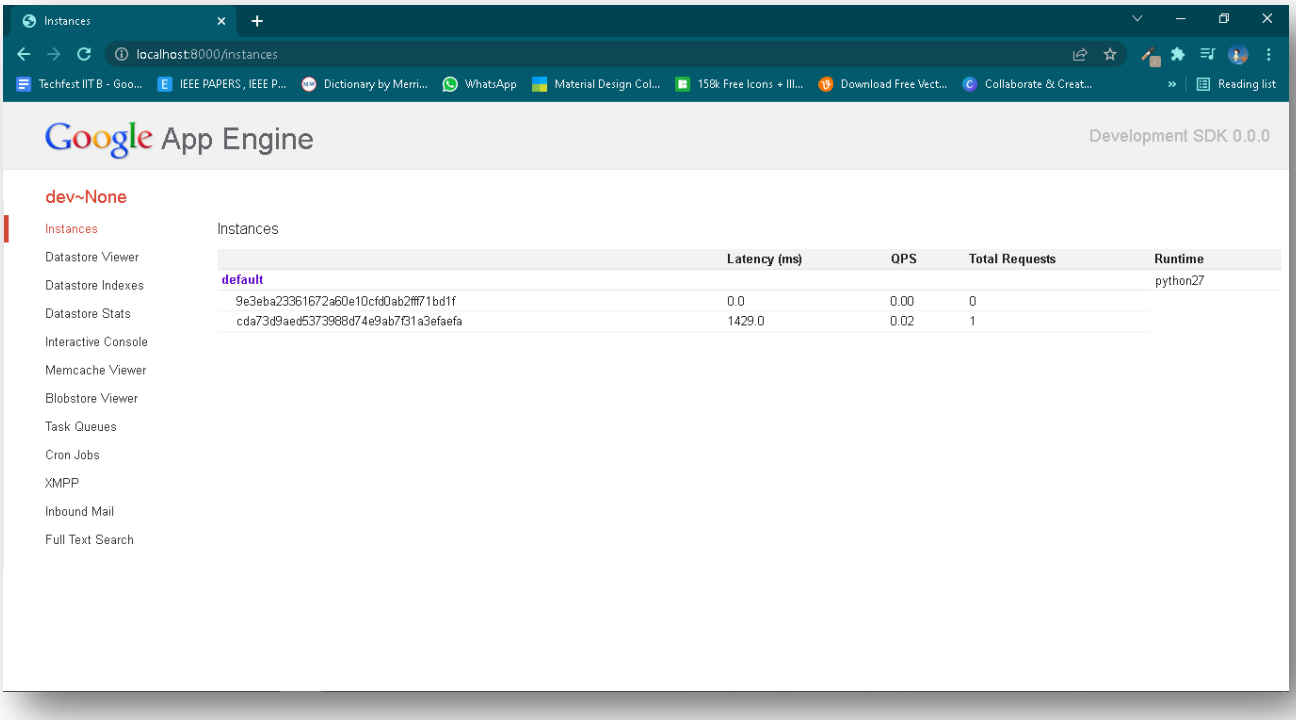
b) Displaying results.html to display results from the API call



c) Terminal / Command Line prompts



d) Cloud Console at port 8000



Assignment no.4

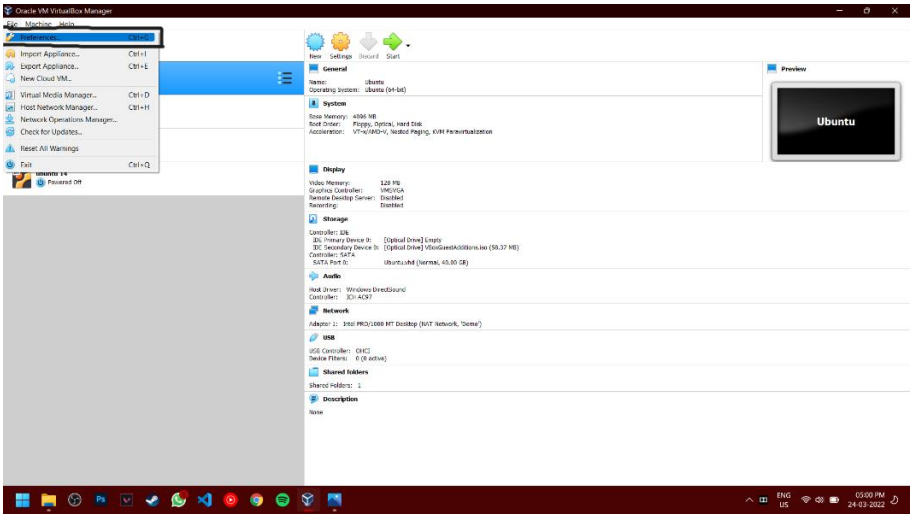
Title: Find a procedure to transfer the files from one virtual machine to another virtual machine

Prerequisites: 2 Virtual machines installed, for this case we have Ubuntu 21 and Kali Linux.

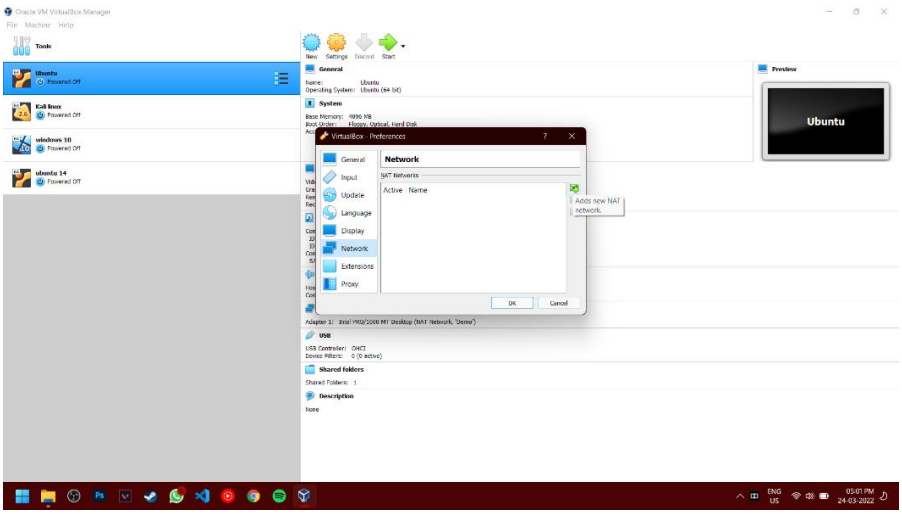
Steps:

1. Create a Nat network in which 2 virtual machine can communicate.

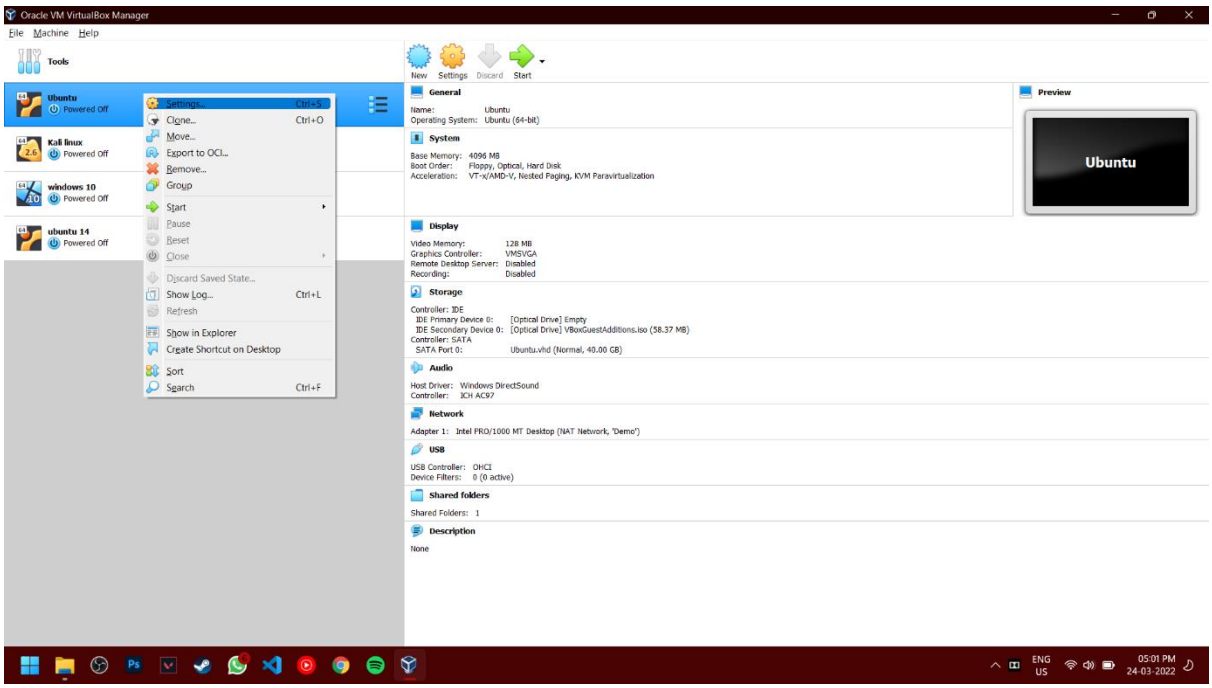
    - Go to prefesences by clicking File option in Top.



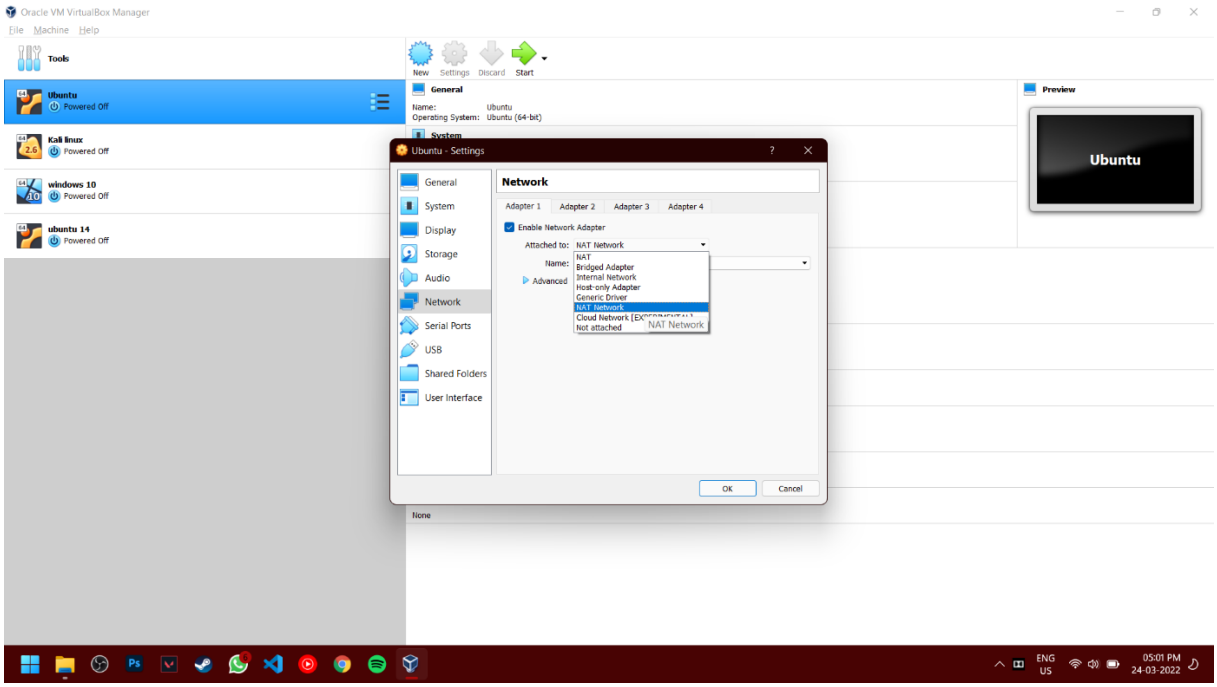
- Now select the network option and create a new NAT network

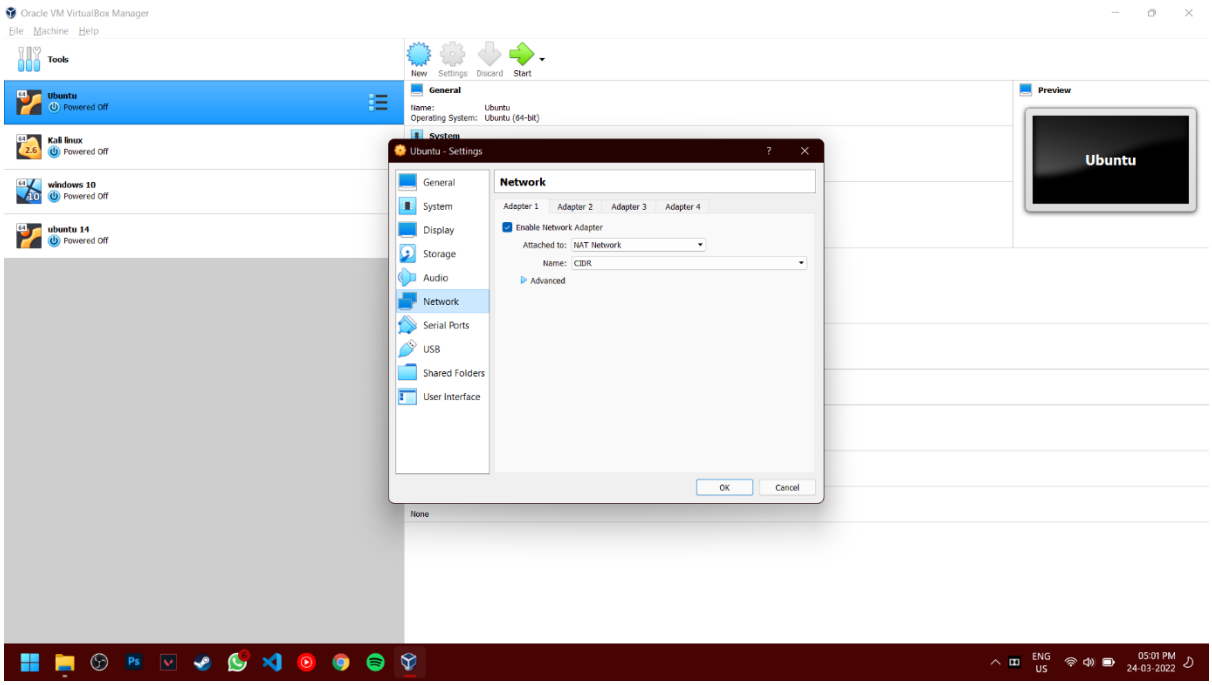


- After creating a NAT network, now go to virtual machine setting by right clicking on the preferred machine.



- Now go to Network and change the attached option to “NAT network” and select the network we created earlier.

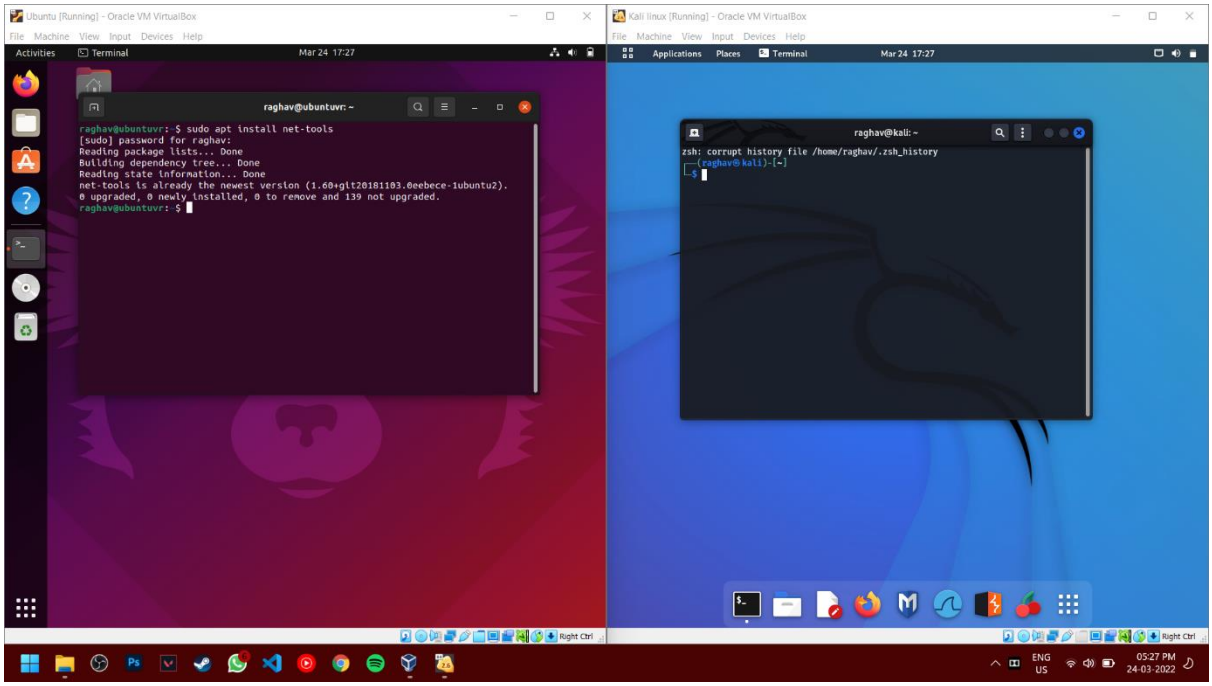




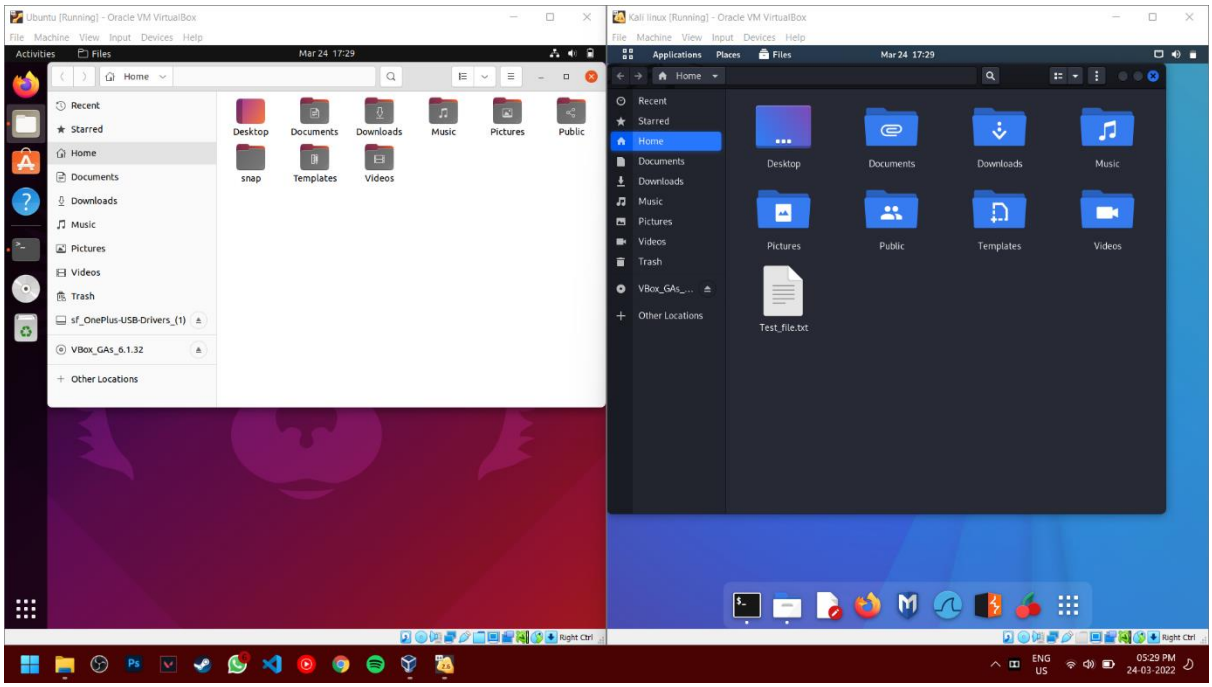
- Now repeat the same process for another machine.

2. Launch both virtual machines

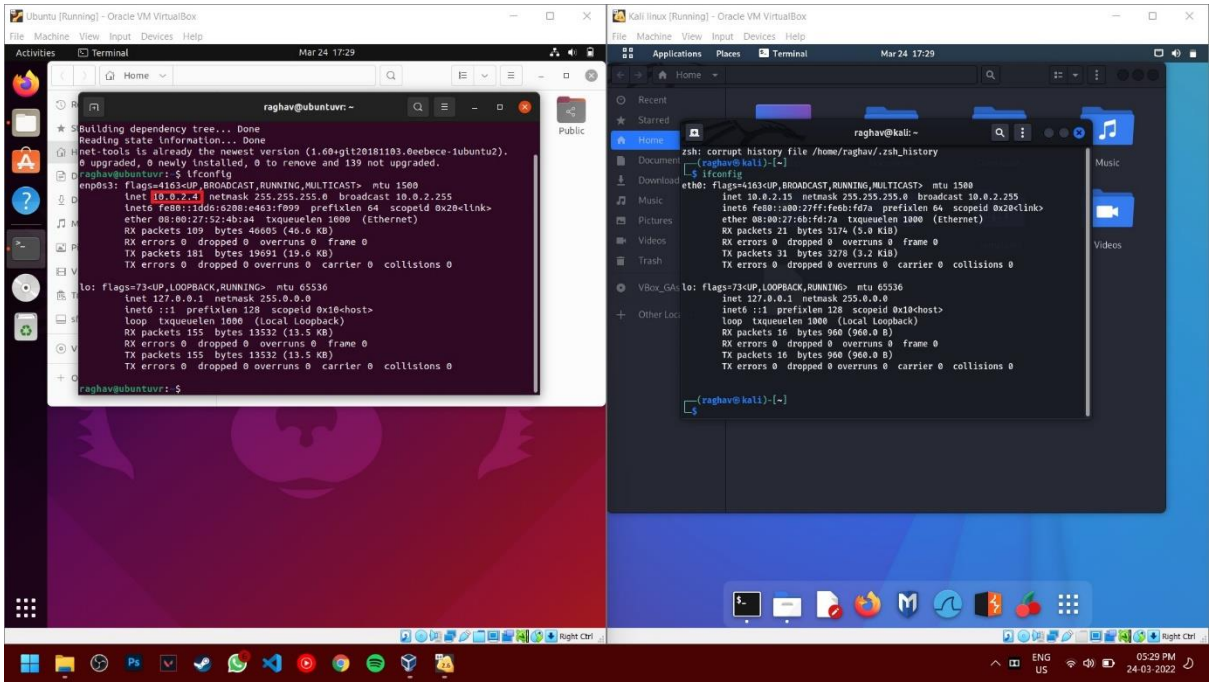
- Now install “Net-tools” on both machine which will help to identify i/p address of the machine. Command: sudo apt install net-tools



- Now create a file in Home folder using any text editor. Here we have used “Test_file.txt” and will transfer from Kali Linux (Right Machine) to Ubuntu (left Machine).

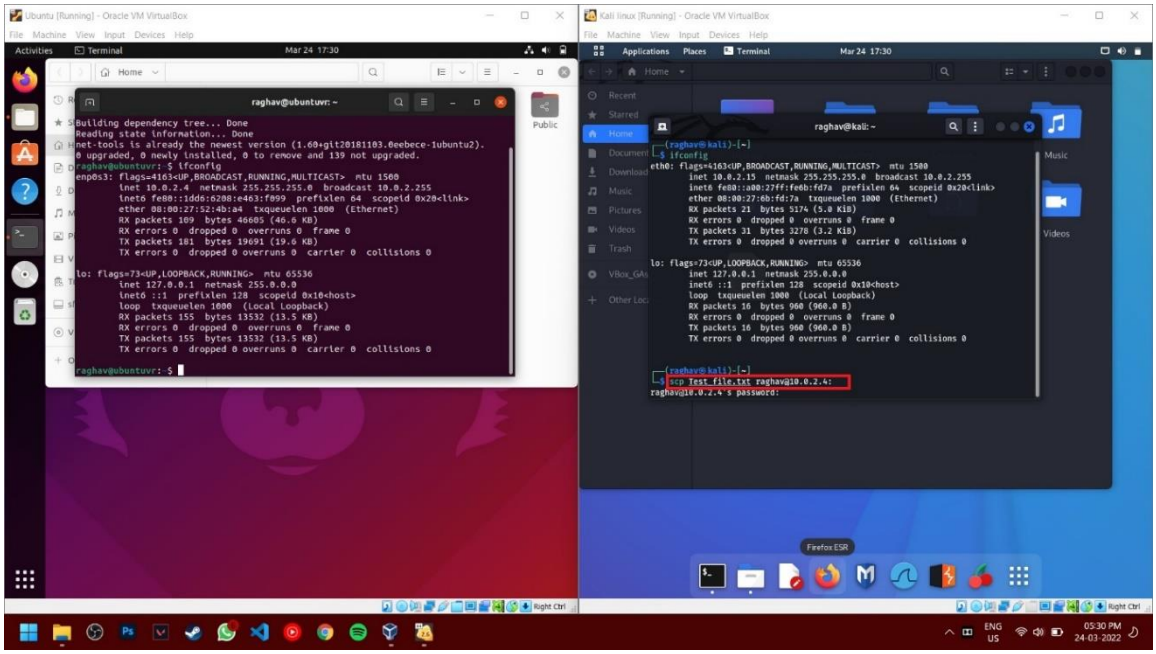


- Now we can check i/p address of Ubuntu where we want to transfer the file using “ ifconfig ” command. Here Ubuntu has i/p address 10.0.2.4.

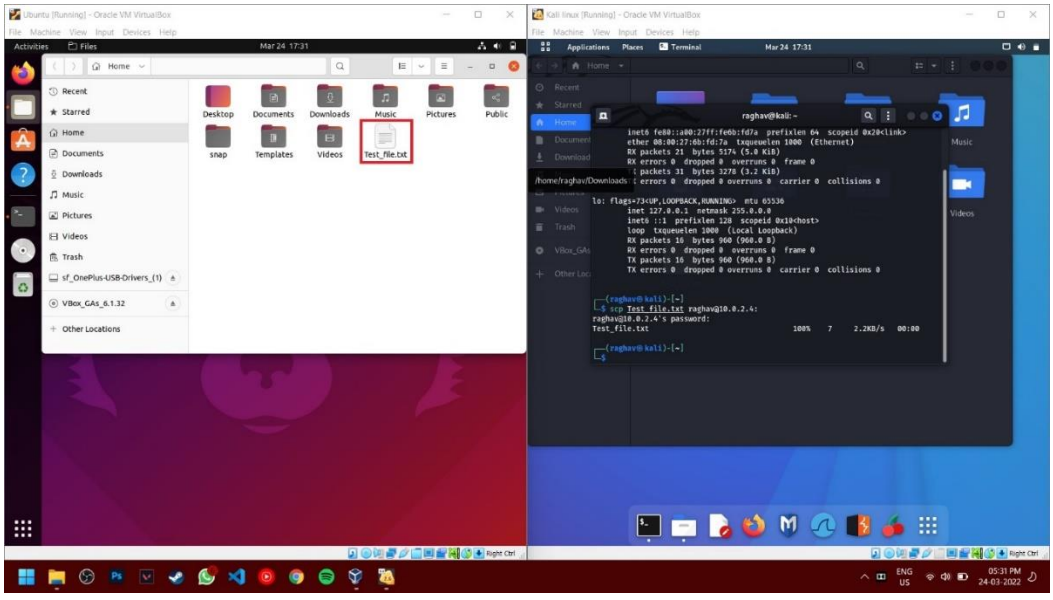


- Transfer the file using command- scp Test_file.txt raghav@10.0.2.4:
  Where Test_file.txt is our file
  raghav is username of Ubuntu
  10.0.2.4 is ip address of Ubuntu (left machine)

  Optional: if scp is not installed then install by using command:
  sudo apt install openssh-server



- Now enter the password for Ubuntu(left machine) admin, after enter the password, the file will be sent from Kali Linux(Right machine) to Ubuntu(Left machine)



Assignment 5

Problem Statement:

Find a procedure to launch virtual machine.

Theory:

Virtual Machine:

Virtual machines allow you to run an operating system in an app window on your desktop that behaves like a full, separate computer. You can use them play around with different operating systems, run software your main operating system can't, and try out apps in a safe, sandboxed environment.

A virtual machine app creates a virtualized environment—called, simply enough, a virtual machine—that behaves like a separate computer system, complete with virtual hardware devices. The VM runs as a process in a window on your current operating system. You can boot an operating system installer disc (or live CD) inside the virtual machine, and the operating system will be "tricked" into thinking it's running on a real computer. It will install and run just as it would on a real, physical machine. Whenever you want to use the operating system, you can open the virtual machine program and use it in a window on your current desktop.

In the VM world, the operating system actually running on your computer is called the host and any operating systems running inside VMs are called guests.

The main purpose of VMs is to operate multiple operating systems at the same time, from the same piece of hardware.

Advantages of VM:

The multiplicity of Operating Systems Reduced Overhead

Safety Net for Data – Rapid Disaster Recovery and Auto Backups Scalability

Centralization

TryStack:

TryStack is a free and easy way for users to try out OpenStack, and set up their own cloud with networking, storage, and computer instances.

Requirement:

Account on

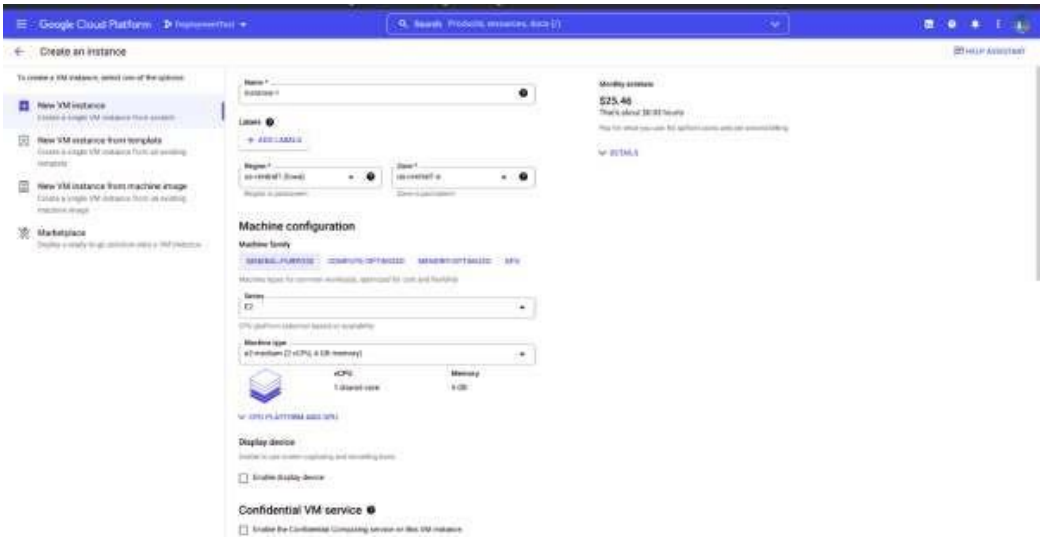
AWS or Azure or Google Cloud.

Steps:

-In the Google Cloud Console, go to the Create an instance page.

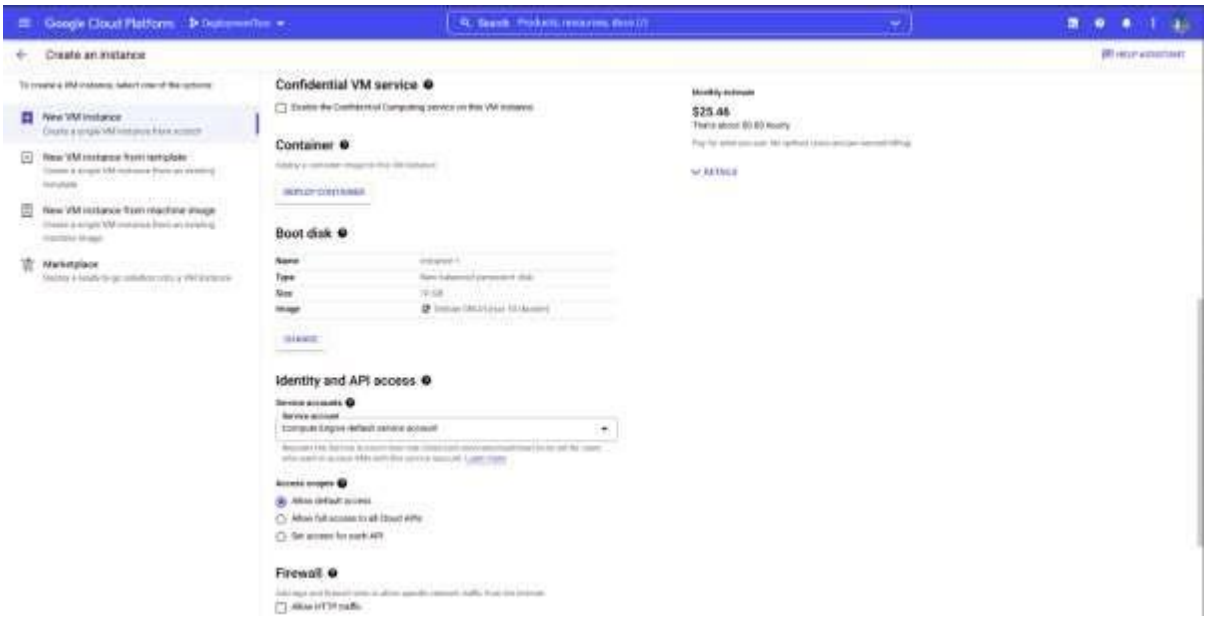
-Go to Create an instance

-Specify a Name for your VM. For more information, see Resource naming convention.



-Optional: Change the Zone for this VM. Compute Engine randomizes the list of zones within each region to encourage use across multiple zones.

-Select a Machine configuration for your VM.



-In the Boot disk section, click Change to configure your boot disk, and then do the following:

-Select the Custom Images tab.

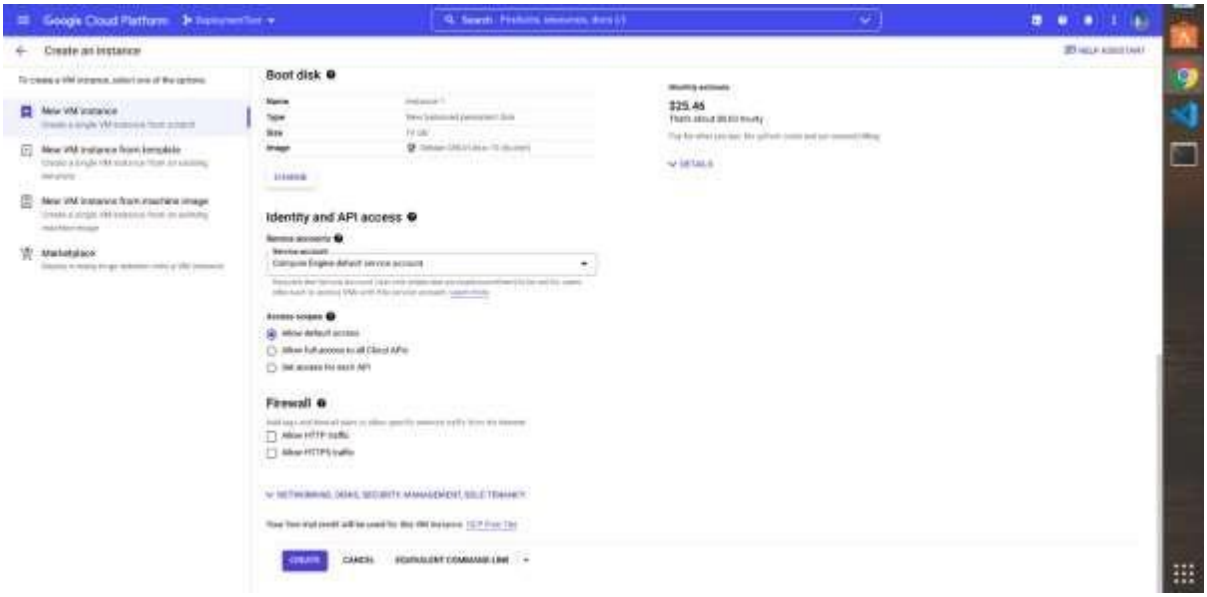
-To select the image project, click Select a project, and then do the following:

-Select the project that contains the image. Click Open.

-In the Image list, click the image that you want to import. Select the type and size of your boot disk.

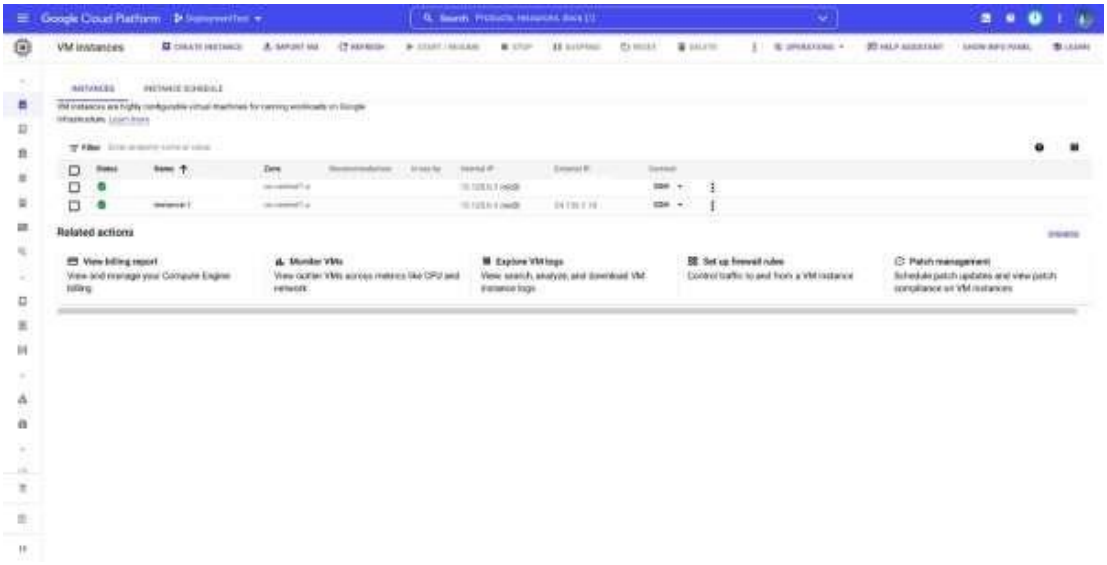
To confirm your boot disk options, click Select.

-To permit HTTP or HTTPS traffic to the VM, in the Firewall section, select Allow HTTP traffic or Allow HTTPS traffic.



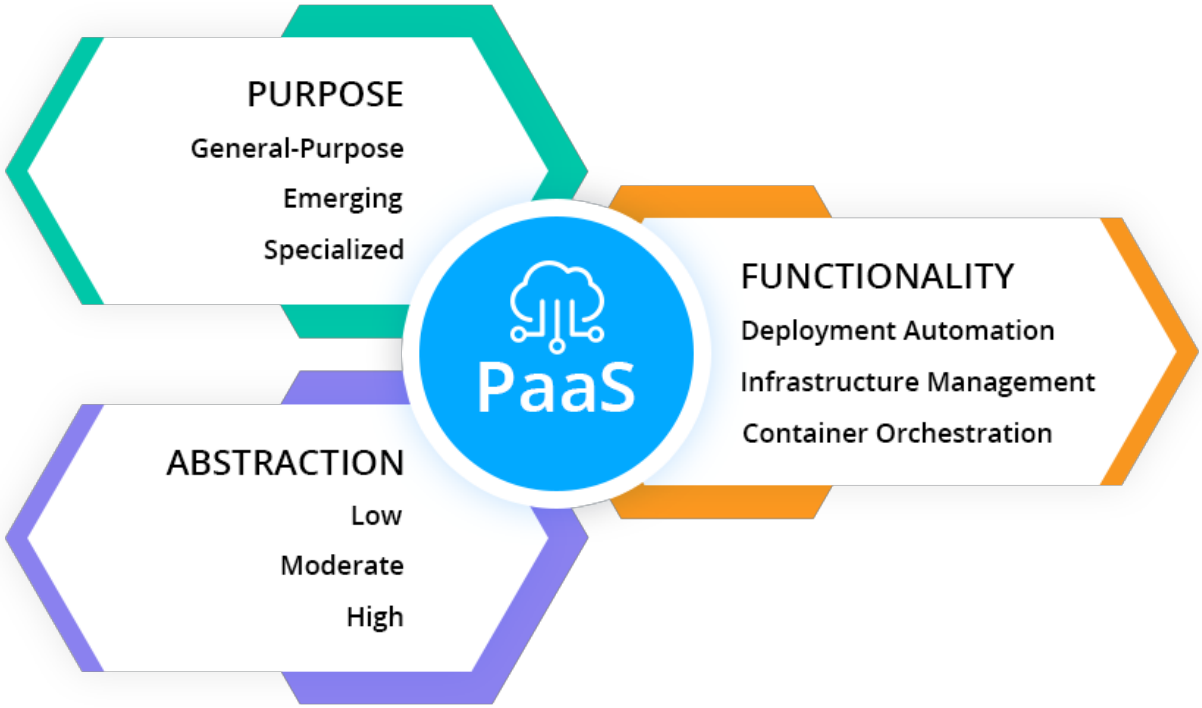
-The Cloud Console adds a network tag to your VM and creates the corresponding ingress firewall rule that allows all incoming traffic on tcp:80 (HTTP) or tcp:443 (HTTPS). The network tag associates the firewall rule with the VM. For more information, see Firewall rules overview in the Virtual Private Cloud documentation.

-To start and create a VM, click Create.



Assignment 6

Problem statement: Design and deploy a web application in a PaaS environment



What AWS Amplify is?

AWS Amplify is a set of purpose-built tools and features that lets frontend web and mobile developers quickly and easily build full-stack applications on AWS, with the flexibility to leverage the breadth of AWS services as your use cases evolve. With Amplify, you can configure a web or mobile app backend, connect your app in minutes, visually build a web frontend UI, and easily manage app content outside the AWS console. Ship faster and scale effortlessly—with no cloud expertise needed.

Steps :

1. Login to the AWS console
2. Find for AWS Amplify in the services.
3. Get Started with Amplify service.
4. Click on Host a Web App.

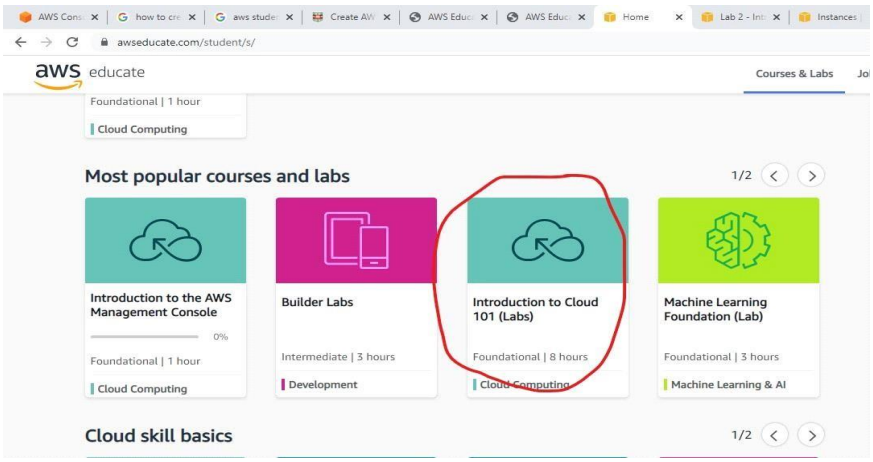
5. Then choose to launch it with Github and authenticate your GitHub account for the same..
6. After that choose the Repository containing your source code (subfolder if needed)
7. Then Launch the application with the default configurations provided by [AWS Amplify]
8. Configurations may be different on type of framework / technology you are launching your application

Detailed Steps:-

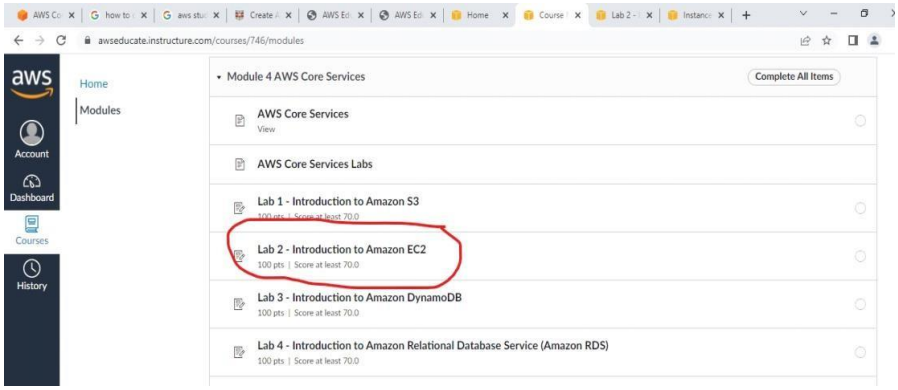
1. https://www.awseducate.com/registration/s/

2. Register yourself in "Learn Cloud Skills" by entering email id

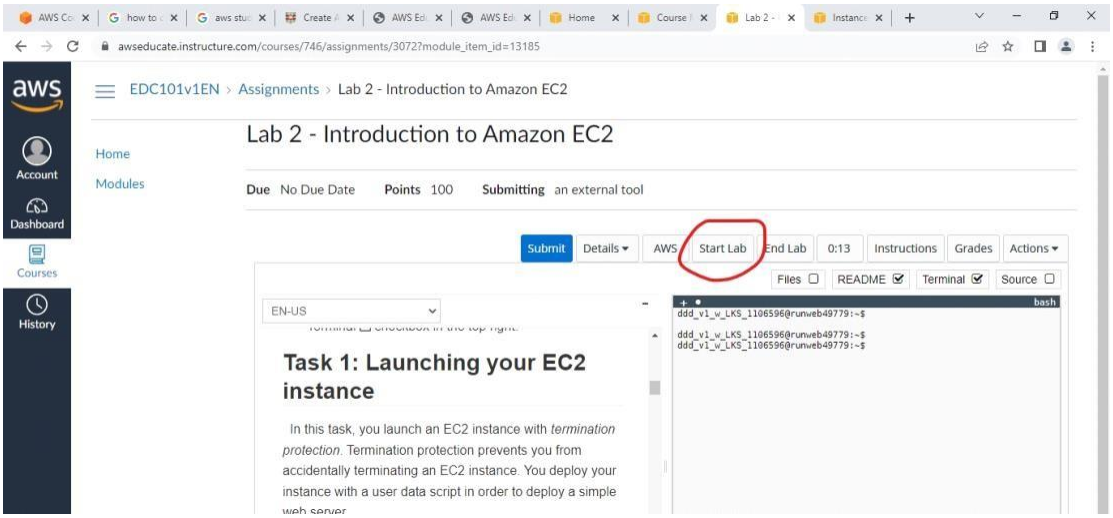
3.



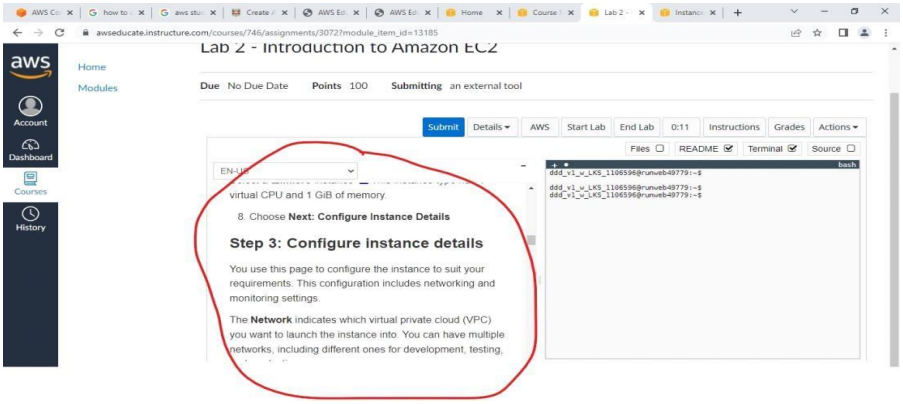
4.



5.



6. Follow all instructions as given by them, Don't change default settings



7. After following the steps at the end your instance will be in running state