

Assignment No.1 : Install Google App Engine. Create hello world app and other simple webapplications using python/java.

Theory:

Introduction

Google App Engine is a web application hosting service. By “web application,” we mean an application or service accessed over the Web, usually with a web browser: storefronts with shopping carts, social networking sites, multiplayer games, mobile applications, survey applications, project management, collaboration, publishing, and all the other things we’re discovering are good uses for the Web. App Engine can serve traditional website content too, such as documents and images, but the environment is especially designed for real-time dynamic applications. Of course, a web browser is merely one kind of client: web application infrastructure is well suited to mobile applications, as well.

In particular, Google App Engine is designed to host applications with many simultaneous users. When an application can serve many simultaneous users without degrading performance, we say it scales. Applications written for App Engine scale automatically. As more people use the application, App Engine allocates more resources for the application and manages the use of those resources. The application itself does not need to know anything about the resources it is using.

Procedure:

Install Google Plugin for Eclipse

Read this guide – [how to install Google Plugin for Eclipse](#). If you install the Google App Engine Java SDK together with “**Google Plugin for Eclipse**”, then go to step 2, Otherwise, get the [GoogleApp Engine Java SDK](#) and extract it.

Create New Web Application Project

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

Figure – New Web Application Project

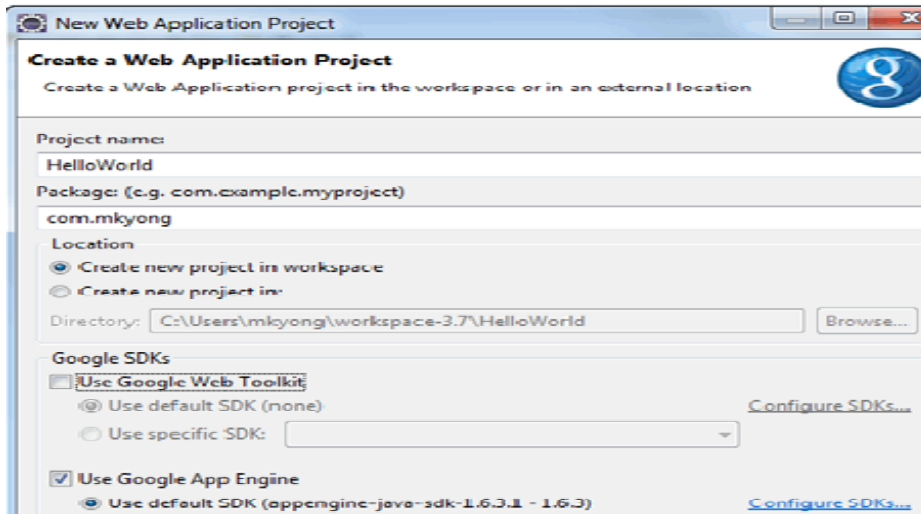
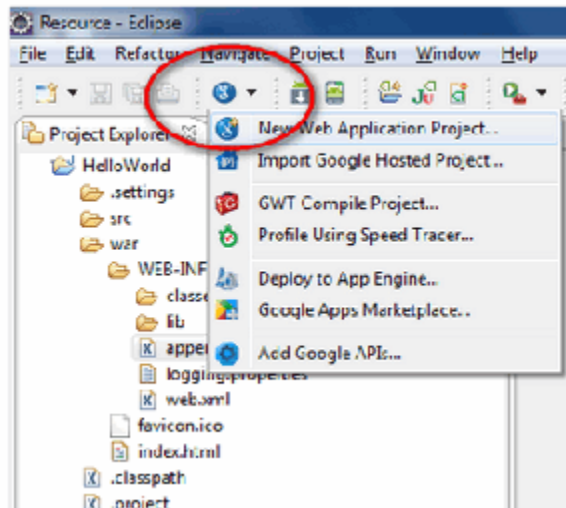


Figure – Deselect the “Google Web ToolKit”, and link your GAE Java SDK via the “configureSDK” link.



Click finished, Google Plugin for Eclipse will generate a sample project automatically.

Hello World

Review the generated project directory Nothing special, a standard Java web project structure.

HelloWorld/ src/

...Java source code...META-INF/

...other configuration...war/

...JSPs, images, data files...WEB-INF/

...app configuration...lib/

...JARs for libraries...classes/

...compiled classes...The extra is this file “appengine-web.xml”, Google App Engine need this to run and deploy the application.

File : appengine-web.xml

```
<?xml version="1.0" encoding="utf-8"?>

<appengine-web-app xmlns="http://appengine.google.com/ns/1.0">

<application></application>

<version>1</version>

<!-- Configure java.util.logging -->

<system-properties>

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

</system-properties>
```

Run it local

Right click on the project and run as “**Web Application**”.

Eclipse console :

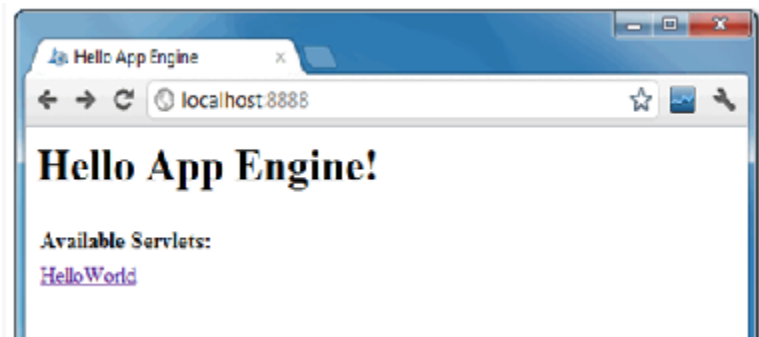
```
//...
```

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

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

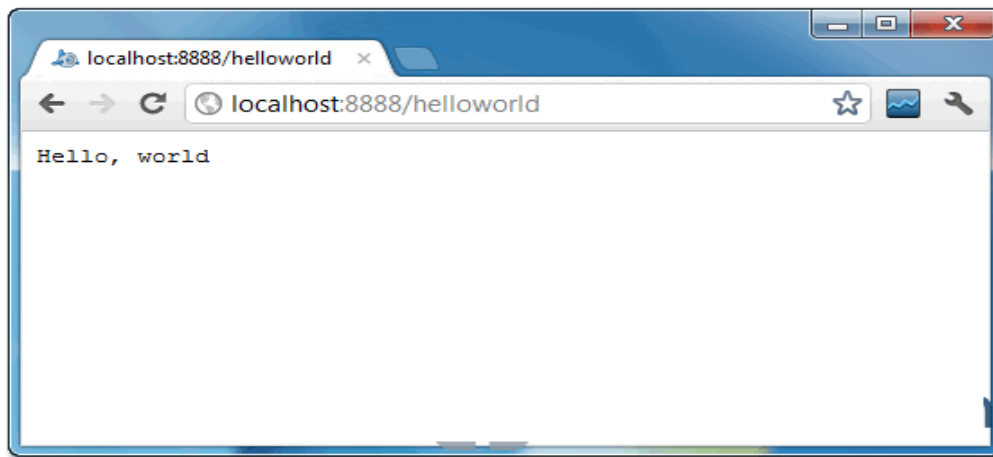
Copy

Access URL `http://localhost:8888/`, see output



and also the hello world servlet – `http://localhost:8888/helloworld`

5. Deploy to Google App Engine



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

In this demonstration, I created an application ID, named “mkyong123”, and put it in `appengine-web.xml`.

File : `appengine-web.xml`

```
<?xml version="1.0" encoding="utf-8"?>
<appengine-web-app xmlns="http://appengine.google.com/ns/1.0">
  <application>mkyong123</application >
  <version>1</version>
```

```
<!-- Configure java.util.logging -->
```

```
<system-properties>
```

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

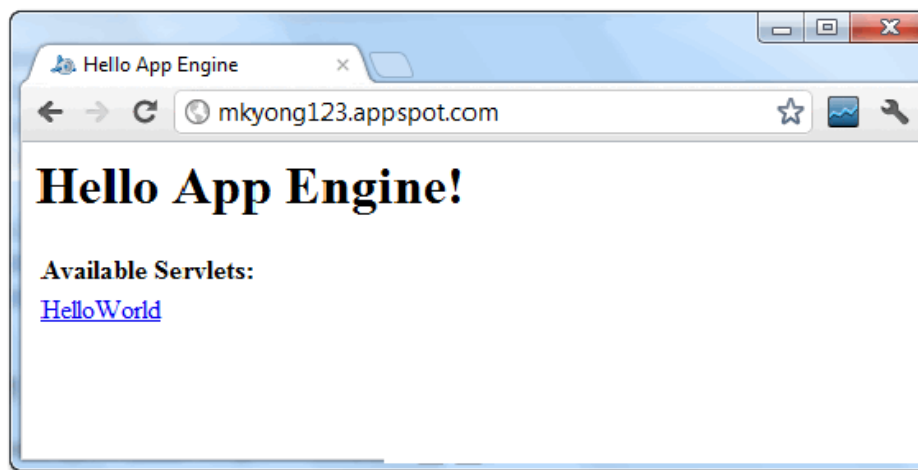
```
</system-properties>
```

```
</appengine-web-app> CopyTo deploy, see
```

following steps: Click on GAE deploy button on the toolbar.

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

– If everything is fine, the hello world web application will be deployed to this URL –
<http://mkyong123.appspot.com/>



Result:

Thus the simple application was created successfully.

Assignment No 2 : Use GAE launcher to launch the web applications

Making your First Application

Now you need to create a simple application. We could use the “+” option to have the launcher make us an application—but instead we will do it by hand to get a better sense of what is going on.

Make a folder for your Google App Engine applications. I am going to make the Folder on my Desktop called “**apps**” —the path to this folder is:

C:\Documents and Settings\csev\Desktop\apps

And then make a sub-folder in within apps called “ae-01-trivial” — the path to this folder would be:

C:\ Documents and Settings \csev\Desktop\apps\ae-01-trivial

Using a text editor such as JEdit (www.jedit.org), create a file called app.yaml in the ae-01-trivial folder with the following contents:

```
application: ae-01-trivial
version: 1
```

```
runtime: python
api_version: 1
handlers:- url: /.*
```

```
script: index.py
```

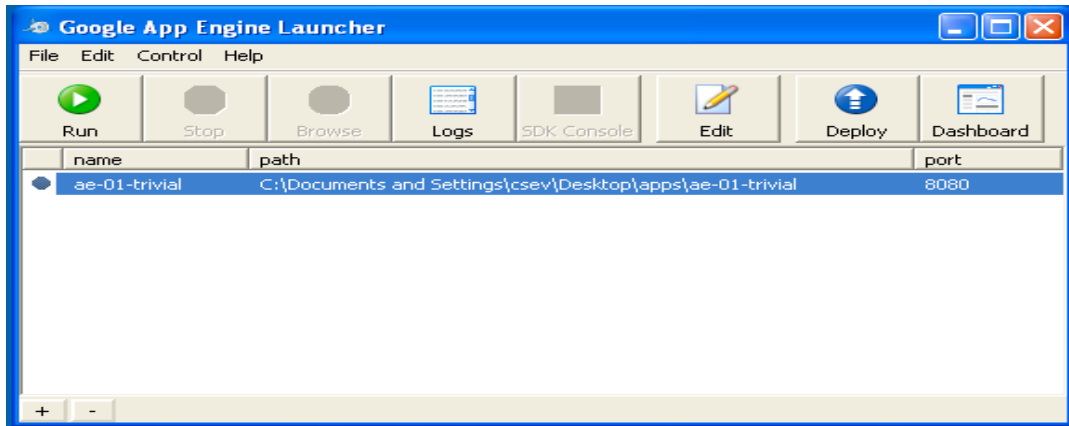
Note: Please do not copy and paste these lines into your text editor— you might end up with strange characters—simply type them into your editor.

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

```
print 'Content-Type: text/plain'
print ' '
```

```
print 'Hello there Chuck'
```

Then start the GoogleAppEngineLauncher program that can be found under Applications. Use the File > Add Existing Application command and navigate into the apps directory and select the ae-01-trivial folder. Once you have added the application, select it so that you can control the application using the launcher.



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

Paste <http://localhost:8080> into your browser and you should see your application as follows:



Just for fun, edit the index.py to change the name "Chuck" to your own name and press Refresh in the browser to verify your updates.

Watching the Log

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

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

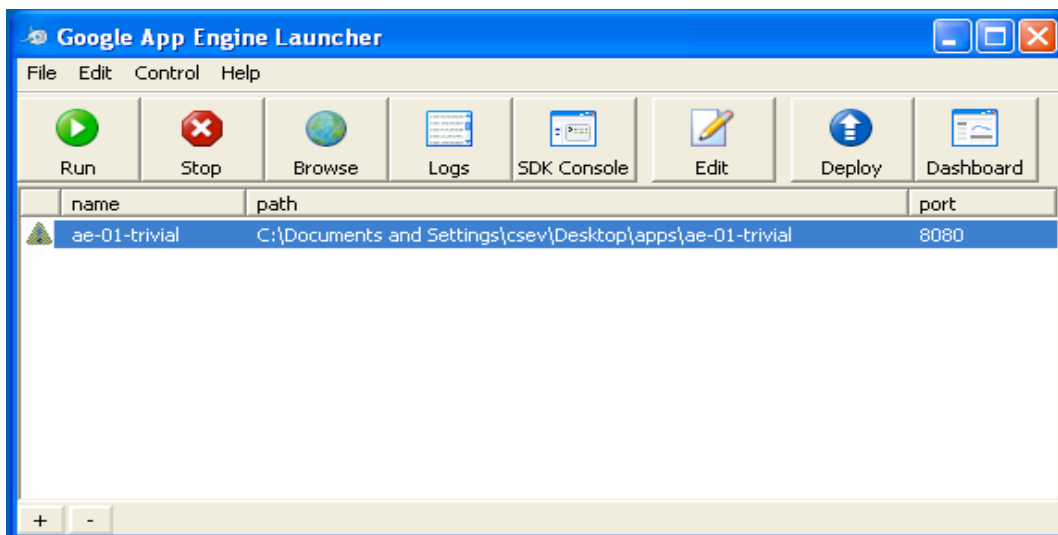
```

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

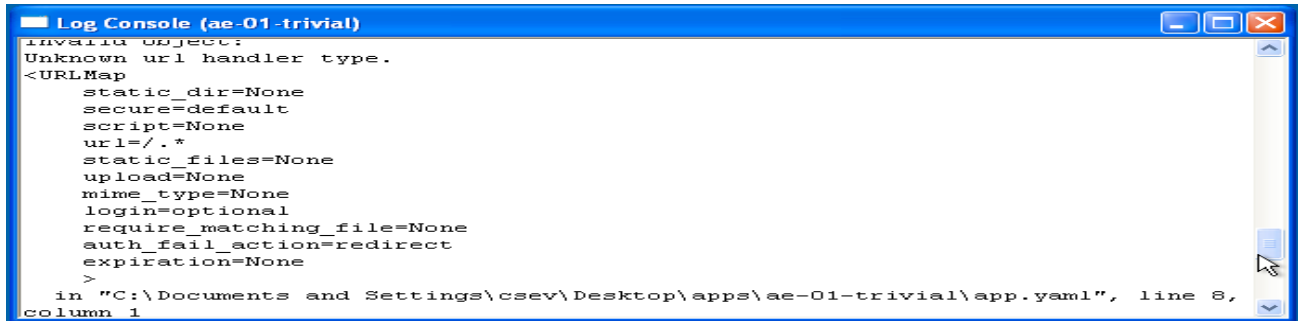
```

Dealing With Errors

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



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



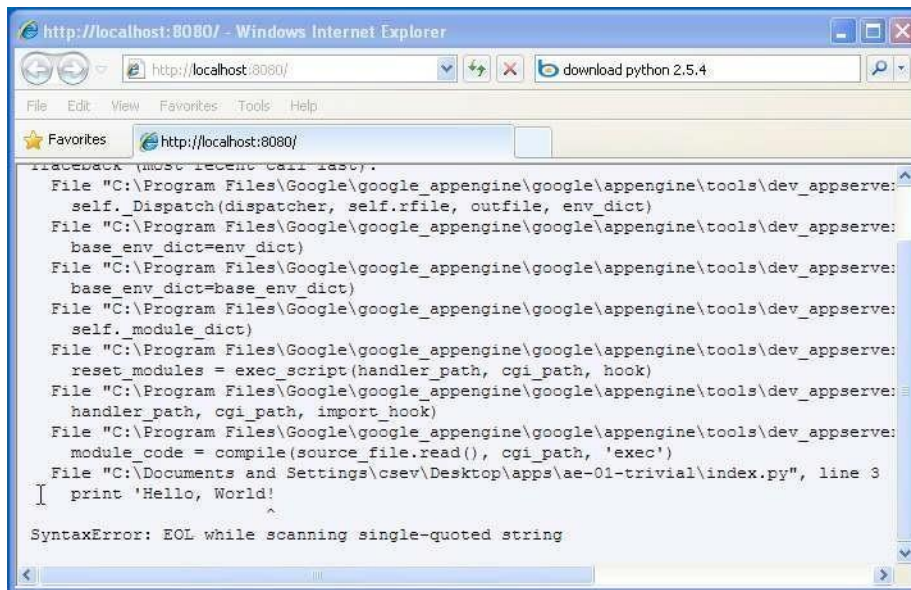
```

Log Console (ae-01-trivial)
Invalid object:
Unknown url handler type.
<URLMap
  static_dir=None
  secure=default
  script=None
  url=/. *
  static_files=None
  upload=None
  mime_type=None
  login=optional
  require_matching_file=None
  auth_fail_action=redirect
  expiration=None
>
in "C:\Documents and Settings\csev\Desktop\apps\ae-01-trivial\app.yaml", line 8,
column 1

```

In this instance — the mistake is mis-indenting the last line in the app.yaml (line 8).

If you make a syntax error in the index.pyfile, a Python trace back error will appear in your browser.



```

http://localhost:8080/ - Windows Internet Explorer
http://localhost:8080/
download python 2.5.4
File Edit View Favorites Tools Help
http://localhost:8080/
Traceback (most recent call last):
  File "C:\Program Files\Google\google_appengine\google\appengine\tools\dev_appserver:
    self.Dispatch(dispatcher, self.rfile, outfile, env_dict)
  File "C:\Program Files\Google\google_appengine\google\appengine\tools\dev_appserver:
    base_env_dict=env_dict)
  File "C:\Program Files\Google\google_appengine\google\appengine\tools\dev_appserver:
    base_env_dict=base_env_dict)
  File "C:\Program Files\Google\google_appengine\google\appengine\tools\dev_appserver:
    self.module_dict)
  File "C:\Program Files\Google\google_appengine\google\appengine\tools\dev_appserver:
    reset_modules = exec_script(handler_path, cgi_path, hook)
  File "C:\Program Files\Google\google_appengine\google\appengine\tools\dev_appserver:
    handler_path, cgi_path, import_hook)
  File "C:\Program Files\Google\google_appengine\google\appengine\tools\dev_appserver:
    module_code = compile(source_file.read(), cgi_path, 'exec')
  File "C:\Documents and Settings\csev\Desktop\apps\ae-01-trivial\index.py", line 3
    print 'Hello, World!
SyntaxError: EOL while scanning single-quoted string

```

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

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

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

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

Shutting Down the Server

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

Result:

Thus the GAE web applications were created.

Assignment No 3: Simulate a cloud scenario using CloudSim and run a scheduling algorithm that is not present in CloudSim.

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:

Download CloudSim installable files

from <https://code.google.com/p/cloudsim/downloads/list> and unzip

Open Eclipse

Create a new Java Project: File -> New

Import an unpacked CloudSim project into the new Java Project

The first step is to initialise the CloudSim package by initialising the CloudSim library, as follows

```
CloudSim.init(num_user, calendar, trace_flag)
```

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

```
Datacenter datacenter9883 = new Datacenter(name, characteristics, new Vm Allocation Policy  
Simple(hostList),s
```

The third step is to create a broker:

===== OUTPUT =====

Cloudlet ID	STATUS	Data center ID	Finish Time	0	VM ID	Time0	Start Time
	SUCCESS	2					400

0.1 400.1

*****Datacenter: Datacenter_0***** User id

Debt

3 35.6

CloudSimExample1 finished!

RESULT:

The simulation was successfully executed.

Assignment No 4: Find a procedure to transfer the files from one virtual machine to another virtual machine.

Steps:

You can copy few (or more) lines with copy & paste mechanism.

For this you need to share clipboard between host OS and guest OS, installing Guest Addition on both the virtual machines (probably setting bidirectional and restarting them). You copy from guest OS in the clipboard that is shared with the host OS.

Then you paste from the host OS to the second guest OS.

You can enable drag and drop too with the same method (Click on the machine, settings, general, advanced, drag and drop: set to bidirectional)

You can have common Shared Folders on both virtual machines and use one of the directory shared as buffer to copy.

Installing Guest Additions you have the possibility to set Shared Folders too. As you put a file in a shared folder from host OS or from guest OS, is immediately visible to the other. (Keep in mind that can arise some problems for date/time of the files when there are different clock settings on the different virtual machines).

If you use the same folder shared on more machines you can exchange files directly copying them in this folder.

You can use usual method to copy files between 2 different computer with client-server application. (e.g. scp with sshd active for linux, winscp... you can get some info about SSH servers e.g. [here](#))

You need an active server (sshd) on the receiving machine and a client on the sending machine. Of course you need to have the authorization setted (via password or, better, via an automatic authentication method).

Note: many Linux/Ubuntu distribution install sshd by default: you can see if it is running with `pgrep sshd` from a shell. You can install with `sudo apt-get install openssh-server`.

You can mount part of the file system of a virtual machine via NFS or SSHFS on the other, or you can share file and directory with Samba. You may find interesting the article [Sharing files between guest and host without VirtualBox shared folders](#) with detailed step by step instructions.

You should remember that you are dialling with a little network of machines with different operative systems, and in particular:

Each virtual machine has its own operative system running on and acts as a physical machine.

Each virtual machine is an instance of a program owned by an user in the hosting operative system and should undergo the restrictions of the user in the hosting OS.

E.g Let us say that Hastur and Meow are users of the hosting machine, but they did not allow each other to see their directories (no read/write/execute authorization). When each of them runs a virtual machine, for the hosting OS those virtual machines are two normal programs owned by Hastur and Meow and cannot see the private directory of the other user. This is a restriction due to the hosting OS. It's easy to overcome it: it's enough to give authorization to read/write/execute to a directory or to choose a different directory in which both users can read/write/execute.

Windows likes mouse and Linux fingers. :-)

I mean I suggest you to enable Drag & drop to be cosy with the Windows machines and the Shared folders or to be cosy with Linux.

When you will need to be fast with Linux you will feel the need of ssh-keygen and

to Generate once SSH Keys to copy files on/from a remote machine without writing password anymore. In this way it functions bash auto-completion remotely too!

PROCEDURE:

Steps:

Open Browser, type localhost:9869

Login using username: oneadmin, password: opennebula

Then follow the steps to migrate VMs

Click on infrastructure

Select clusters and enter the cluster name

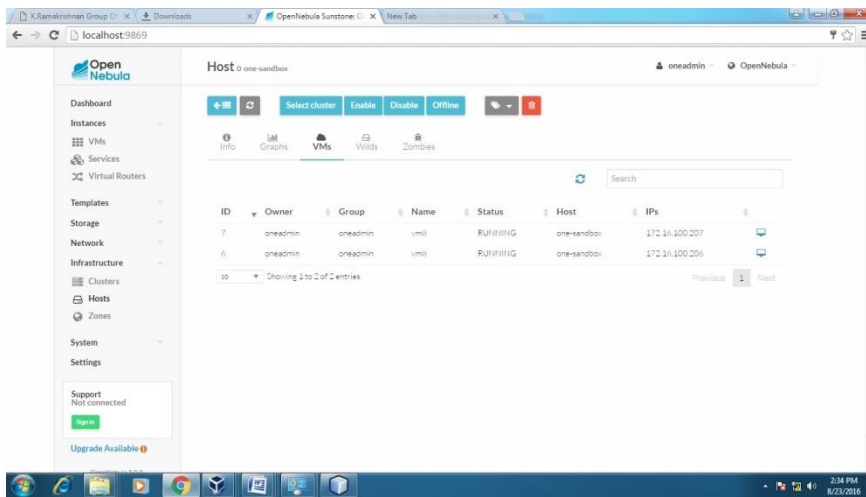
Then select host tab, and select all host

Then select Vnets tab, and select all vnet

Then select datastores tab, and select all datastores

And then choose host under infrastructure tab

Click on + symbol to add new host, name the host then click on create.

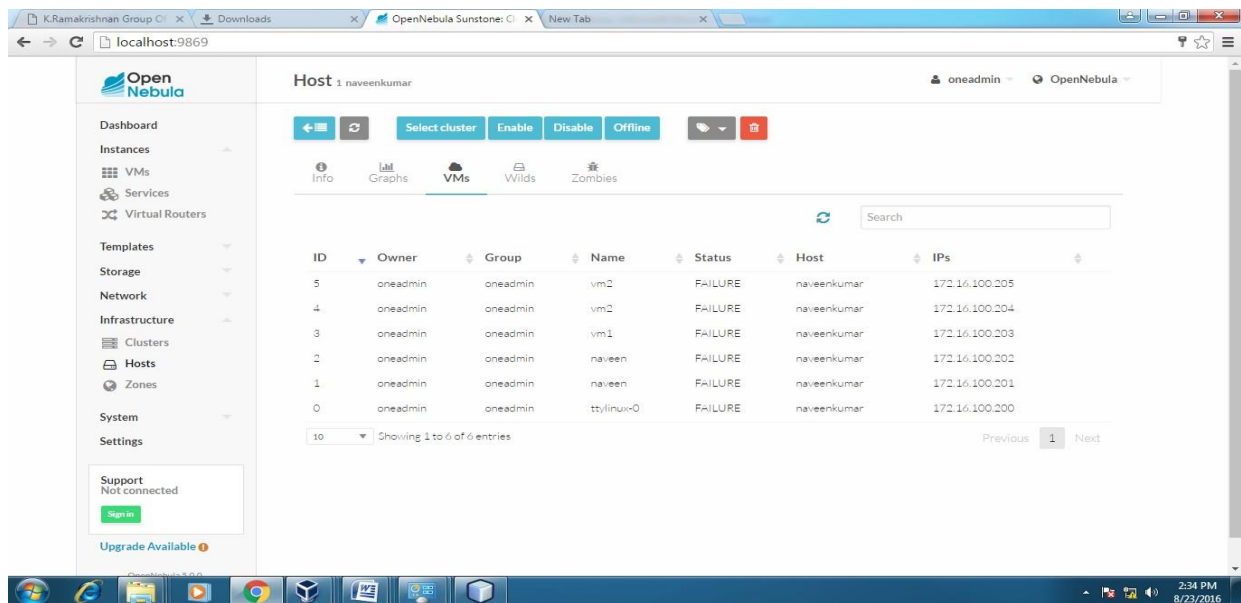


on instances, select VMs to migrate then follow the steps

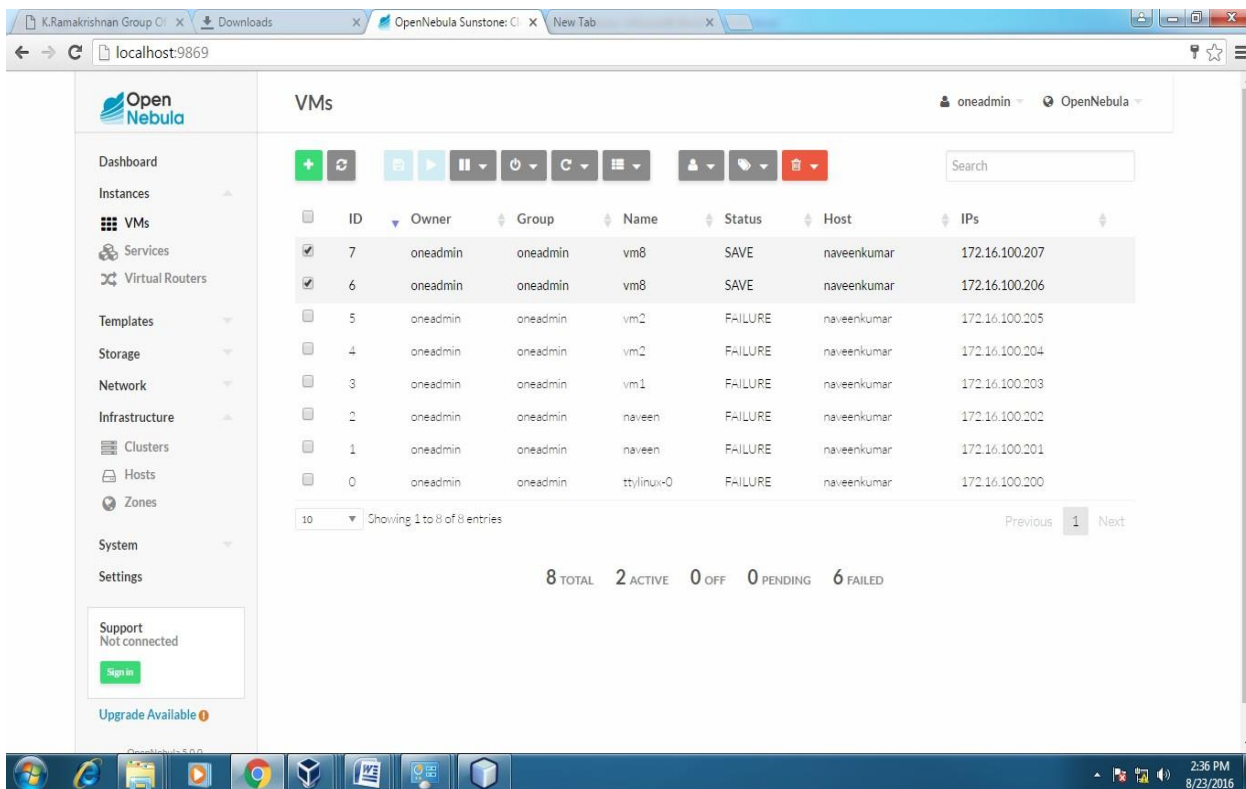
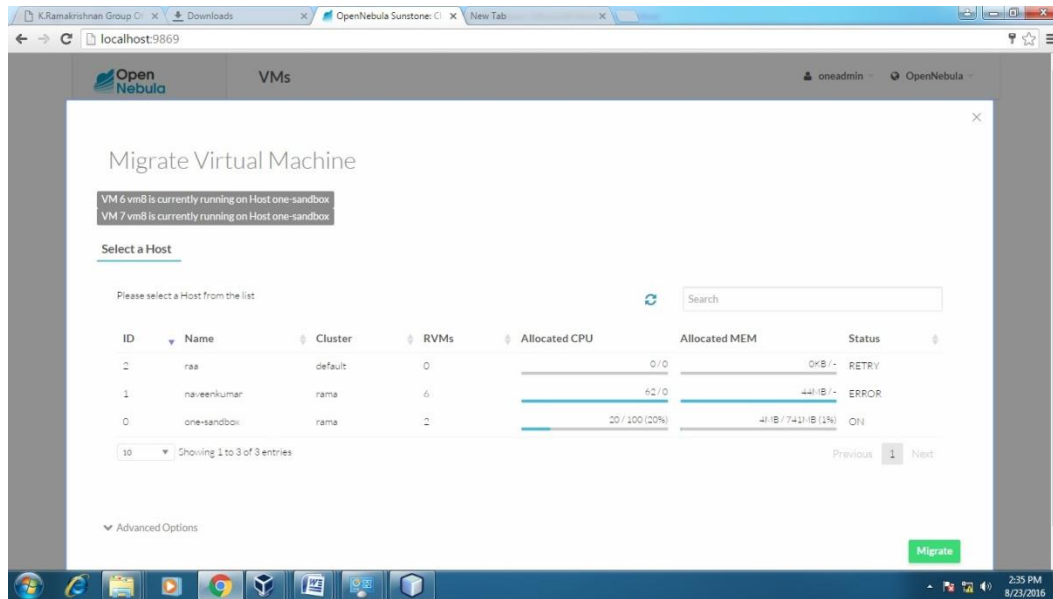
Click on 8th icon, the drop down list display

Select migrate on that, the popup window display

On that select the target host to migrate then click on migrate. Before migration Host: SACET



Host:one-sandbox



Hosts

ID	Name	Cluster	RVMs	Allocated CPU	Allocated MEM	Status
2	raa	default	0	0/0	0KB/-	ERROR
1	naveenkumar	rama	8	82/0	48MB/-	ERROR
0	one-sandbox	rama	0	0/100 (0%)	0KB/741MB (0%)	ON

Showing 1 to 3 of 3 entries

3 TOTAL 1 ON 0 OFF 2 ERROR

After Migration:

Host: one-sandbox

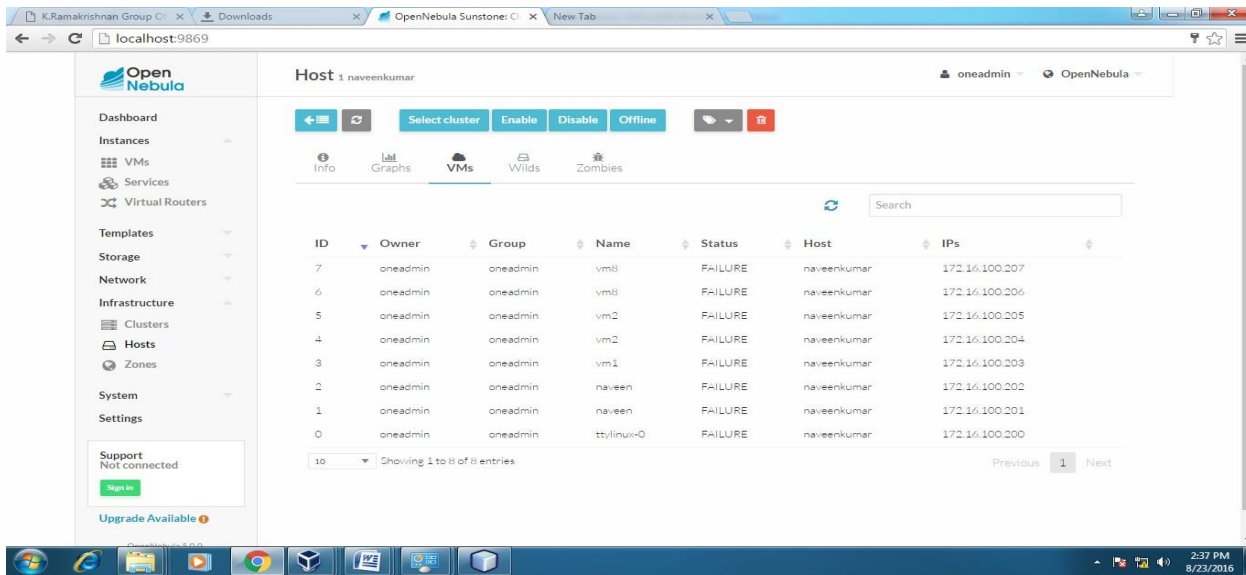
Info Graphs **VMs** Wilds Zombies

ID	Owner	Group	Name	Status	Host	IPs
----	-------	-------	------	--------	------	-----

Showing 0 to 0 of 0 entries

There is no data available

Host:one-sandbox



Host:SACET

APPLICATIONS:

Easily migrate your virtual machine from one pc to another.

Result:

Thus the file transfer between VM was successfully completed.....

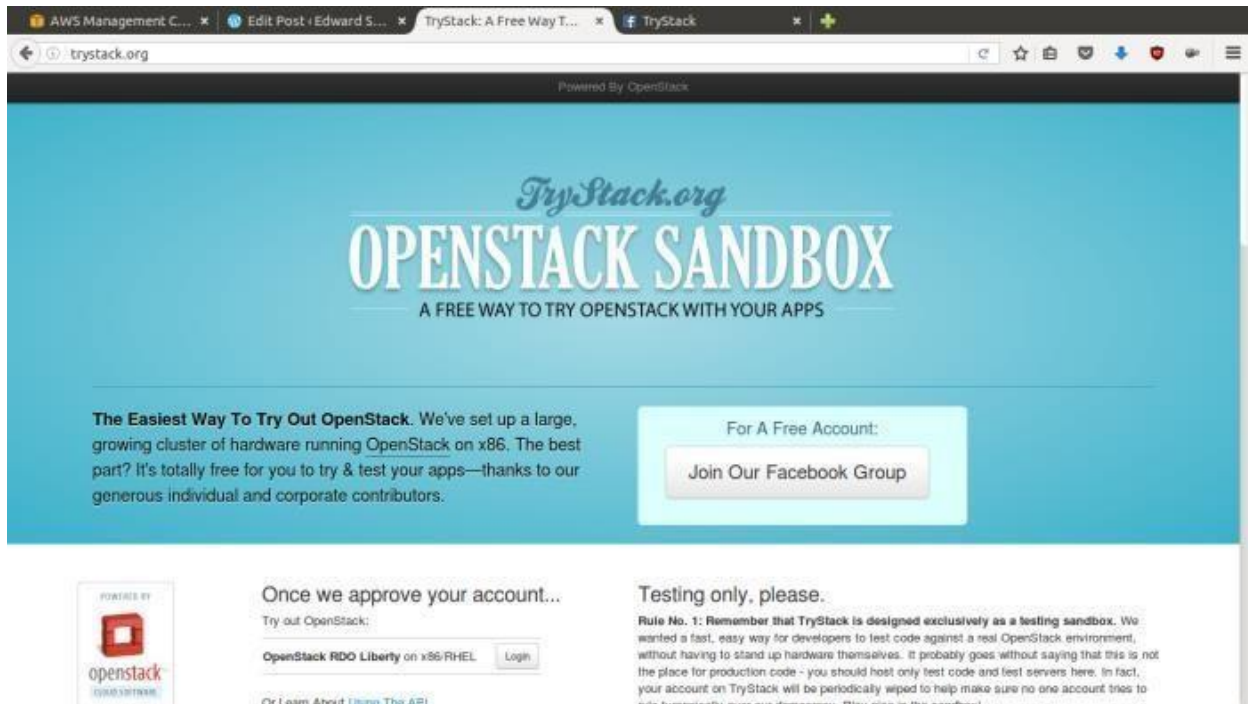
Assignment No 5: Find a procedure to launch virtual machine using trystack (Online Openstack Demo Version)

Steps:

OpenStack is an open-source software cloud computing platform.

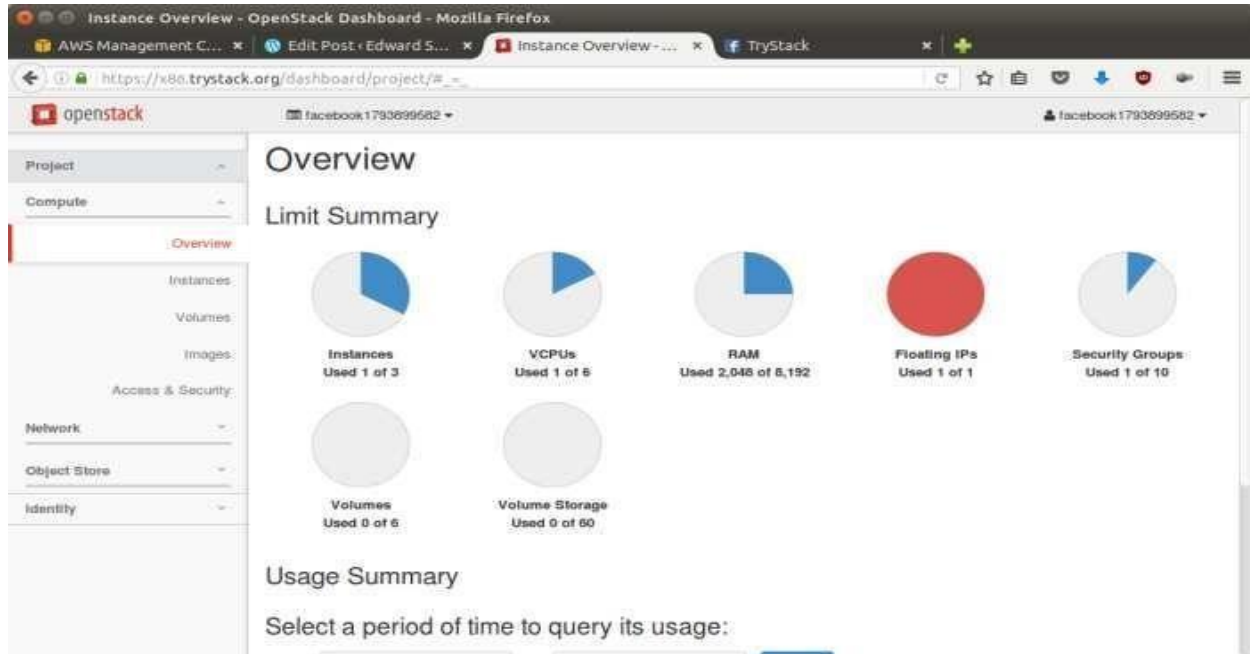
OpenStack is primarily used for deploying an infrastructure as a service (IaaS) solution like Amazon Web Service (AWS). In other words, you can make your own AWS by using OpenStack. If you want to try out OpenStack, **TryStack** is the easiest and free way to do it.

In order to try OpenStack in TryStack, you must register yourself by joining TryStack Facebook Group. The acceptance of group needs a couple days because it's approved manually. After you have been accepted in the TryStack Group, you can log in TryStack.



TryStack.org Homepage

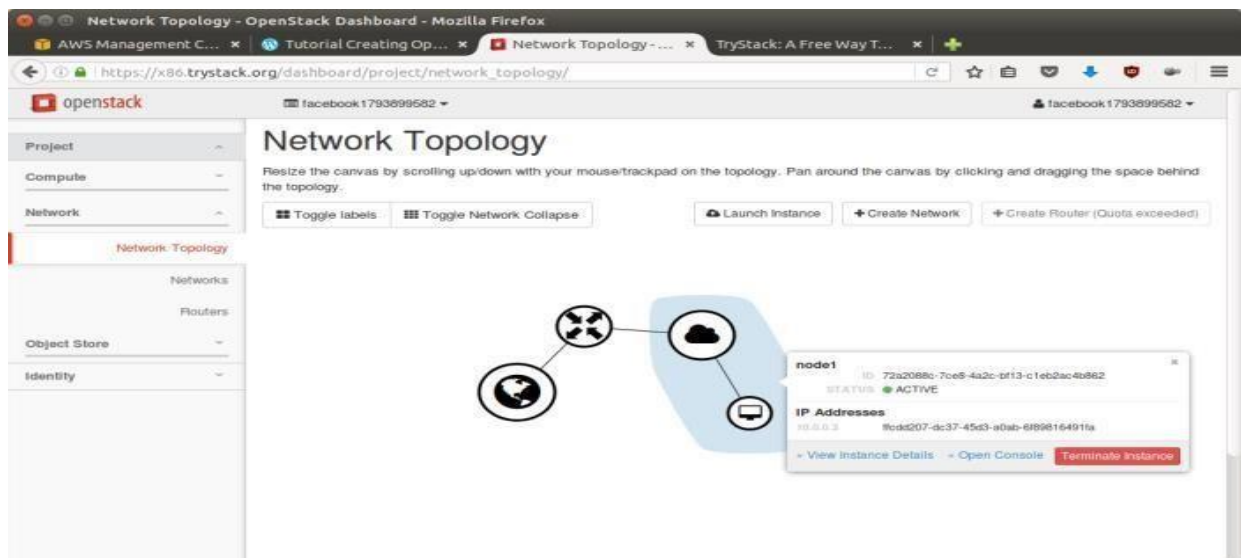
I assume that you already join to the Facebook Group and login to the dashboard. After you log in to the TryStack, you will see the Compute Dashboard like:



OpenStack Compute Dashboard

Overview: What we will do?

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



Network topology

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

Step 1: Create Network

Network? Yes, the network in here is our own local network. So, your instances will be not mixed up with the others. You can imagine this as your own LAN (Local Area Network) in the cloud.

Go to **Network > Networks** and then click **Create Network**.

In **Network** tab, fill **Network Name** for example `internal` and then click **Next**.

In **Subnet** tab,

Fill **Network Address** with appropriate CIDR, for example `192.168.1.0/24`. Use **private network CIDR block** as the best practice.

Select **IP Version** with appropriate IP version, in this case `IPv4`.

Click **Next**.

In **Subnet Details** tab, fill **DNS Name Servers** with `8.8.8.8` (GoogleDNS) and then click **Create**.

Step 2: Create Instance

Now, we will create an instance. The instance is a virtual machine in the cloud, like AWS EC2. You need the instance to connect to the network that we just created in the previous step.

Go to **Compute > Instances** and then click **Launch Instance**.

In **Details** tab,

Fill **Instance Name**, for example `Ubuntu 1`.

Select **Flavor**, for example `m1.medium`.

Fill **Instance Count** with `1`.

Select **Instance Boot Source** with **Boot from Image**.

Select **Image Name** with **Ubuntu 14.04 amd64 (243.7 MB)** if you want install Ubuntu 14.04 in your virtual machine.

In **Access & Security** tab,

Click **[+]** button of **Key Pair** to import key pair. This key pair is a public and private key that we will use to connect to the instance from our machine.

In **Import Key Pair** dialog,

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

Fill **Public Key** with your **SSH public key** (usually is in `~/.ssh/id_rsa.pub`). See description in Import Key Pair dialog box for more information. If you are using Windows, you can use **Puttygen** to generate key pair.

Click **Import key pair**.

In **Security Groups**, mark/check **default**. In **Networking** tab,

In **Selected Networks**, select network that have been created in Step 1, for example **internal**.

Click **Launch**.

If you want to create multiple instances, you can repeat step 1-5. I created one more instance with instance name **Ubuntu 2**.

Step 3: Create Router

I guess you already know what router is. In the step 1, we created our network, but it is isolated. It doesn't connect to the internet. To make our network has an internet connection, we need a router that running as the gateway to the internet.

Go to **Network > Routers** and then click **Create Router**.

Fill **Router Name** for example **router1** and then click **Create router**.

Click on your **router name link**, for example **router1**, **Router Details** page.

Click **Set Gateway** button in upper right:

Select **External networks** with **external**.

Then **OK**.

Click **Add Interface** button.

Select **Subnet** with the network that you have been created in Step 1.

Click **Add interface**.

Go to **Network > Network Topology**. You will see the network topology. In the example, there are two network, i.e. external and internal, those are bridged by a router. There are instances those are joined to internal network.

Step 4: Configure Floating IP Address

Floating IP address is public IP address. It makes your instance is accessible from the internet. When you launch your instance, the instance will have a private network IP, but no public IP. In OpenStack, the public IPs are collected in a pool and managed by admin (in our case is TryStack). You need to request a public (floating) IP address to be assigned to your instance.

Go to **Compute > Instance**.

In one of your instances, click **More > Associate Floating IP**.

In **IP Address**, click Plus [+].

Select **Pool** to **external** and then click **Allocate IP**.

Click **Associate**.

Now you will get a public IP, e.g. 8.21.28.120, for your instance.

Step 5: Configure Access & Security

OpenStack has a feature like a firewall. It can whitelist/blacklist your in/out connection. It is called Security Group.

Go to **Compute > Access & Security** and then open **Security Groups** tab.

In **default** row, click **Manage Rules**.

Click **Add Rule**, choose **ALL ICMP** rule to enable ping into your instance, and then click

Add.

Click **Add Rule**, choose **HTTP** rule to open HTTP port (port 80), and then click **Add**.

Click **Add Rule**, choose **SSH** rule to open SSH port (port 22), and then click **Add**.

You can open other ports by creating new rules.

Step 6: SSH to Your Instance

Now, you can SSH your instances to the floating IP address that you got in the step 4. If you are using Ubuntu image, the SSH user will be ubuntu.

Result:

Thus the openstack demo worked successfully.

Assignment No 6 : Design and develop custom Application (Mini Project) using Salesforce Cloud.

Introduction

Salesforce.com Inc. is an American cloud-based software company headquartered in San Francisco, California. Though the bulk of its revenue comes from a customer relationship management (CRM) product, Salesforce also sells a complementary suite of enterprise applications focused on customer service, marketing automation, analytics and application development.

Salesforce is the primary enterprise offering within the Salesforce platform. It provides companies with an interface for case management and task management, and a system for automatically routing and escalating important events. The Salesforce customer portal provides customers the ability to track their own cases, includes a social networking plug-in that enables the user to join the conversation about their company on social

networking websites, provides analytical tools and other services including

email alert, Google search, and access to customers' entitlement and contracts.

Lightning Platform

Lightning Platform (also known as Force.com) is a platform as a service (PaaS) that allows developers to create add-on applications that integrate into the main Salesforce.com application. These third-party applications are hosted on Salesforce.com's infrastructure. Force.com applications are built using declarative tools, backed by Lightning and Apex (a proprietary Java-like programming language for Force.com) and Lightning and Visual force (a framework that includes an XML syntax typically used to generate HTML). The Force.com platform typically receives three complete releases a year. As the platform is provided as a service to its developers, every single development instance also receives all these updates.

Community Cloud

Community Cloud provides Salesforce customers the ability to create online web properties for external collaboration, customer service, channel sales, and other custom portals in their instance of Salesforce. Tightly integrated to Sales Cloud, Service Cloud, and App Cloud, Community Cloud can be quickly customized to provide a wide variety of web properties. Salesforce Sales Cloud Salesforce Sales Cloud is a customer relationship management (CRM) platform designed to support sales, marketing and customer support in both business-to-business (B2B) and business-to-customer (B2C) contexts. Sales Cloud is a fully customizable product that brings all the customer information together in an integrated platform that incorporates marketing, lead generation, sales, customer service and business analytics and

provides access to thousands of applications through the AppExchange. The platform is provided as Software as a Service (SaaS) for browser-based access; a mobile app is also available. A realtime social feed for collaboration allows users to share information or ask questions of the user community. Salesforce.com offers five versions of Sales Cloud on a per-user, per month basis, from lowest to highest: Group, Professional, Enterprise, Unlimited and Performance. The company offers three levels of support contracts: Standard Success Plan, Premier Success Plan and Premier+ Success Plan.

Create Custom Apps for Salesforce Classic

Create custom apps to give your Salesforce Classic users' access to everything they need all in one place.

If you're new to custom apps, we recommend using Lightning Platform quick start to create an app. With this tool, you can generate a basic working app in just one step.

If you've already created the objects, tabs, and fields you need for your app, follow these steps. With this option, you create an app label and logo, add items to the app, and assign the app to profiles.

From Setup, enter Apps in the Quick Find box, then select Apps.

Click New.

If the Salesforce console is available, select whether you want to define a custom app or a Salesforce console.

Give the app a name and description.

An app name can have a maximum of 40 characters, including spaces.

Optionally, brand your app by giving it a custom logo.

Select which items to include in the app.

1. Optionally, set the default landing tab for your new app using the Default Landing Tab drop-down menu below the list of

selected tabs. This determines the first tab a user sees when logging into this app.

8. Choose which profiles the app will be visible to.

2. Check the Default box to set the app as that profile's default

app, meaning that new users with the profile see this app the first time they log in. Profiles with limits are excluded

from this list.

1. Click Save

What is the difference between custom application and console application in sales force? A custom application is a collection of tabs, objects etc that function together to solve a particular problem.

A console application uses a specific Salesforce UI - the console. Console applications are intended to enhance productivity by allowing everything to be done from a single, tabbed, screen.

Result:

Thus, We have designed and developed custom application using salesforce cloud.