Lab Guide for Introduce Compute Services from GCP

CONTENTS

Contents

Lab Guide for Introduce Compute Services from GCP 1
CONTENTS 1
Assignment 1: Instance Groups 2
Assignment 2a: Adding a Persistent Disk to an Instances 10
Assignment 2b. Formatting and Mounting a Persistent Disks 18
Assignment 3: GCP- Images 23
Assignment 4a: Creating a cross-region HTTP load balancer 28
Assignment 4b: Creating a cross-region HTTP load balancer 30
Assignment 5. Create an instance template 39
Assignment 6a. Download and install the APP Engine SDK for Java 53
Assignment 6b. Run the Demo application from your local server through Command Prompt 55
Assignment 7a. Installing the Google Plugin for Eclipse to create Web Application 60
Assignment 7b. Creating and Running the Web Application locally 63
Assignment 7c: Deploying the App to Google App Engine 70
Assignment 8a: Creating a google cloud function 77
Assignment 8b: Triggering a google cloud function 83
Assignment 9: Google Container Registry 86
Assignment 10: Google Container Engine-Deploying App 87

Internal

Day-5 Assignments

Context

This document contains assignments to be completed as part of the hands on session for the course

Guidelines

- The lab guide has been designed to give hands on experience to map the concepts learnt in the theory session with real life business oriented case studies/assignments.

Day-5 Assignments

Assignment 1: Instance Groups

Objective: To create an Instance Template and use the same template in creating a Managed Instance Group.

Problem Description:

In this Assignment you will learn how create an Instance template with required configuration and attach the same to create a managed instance group.

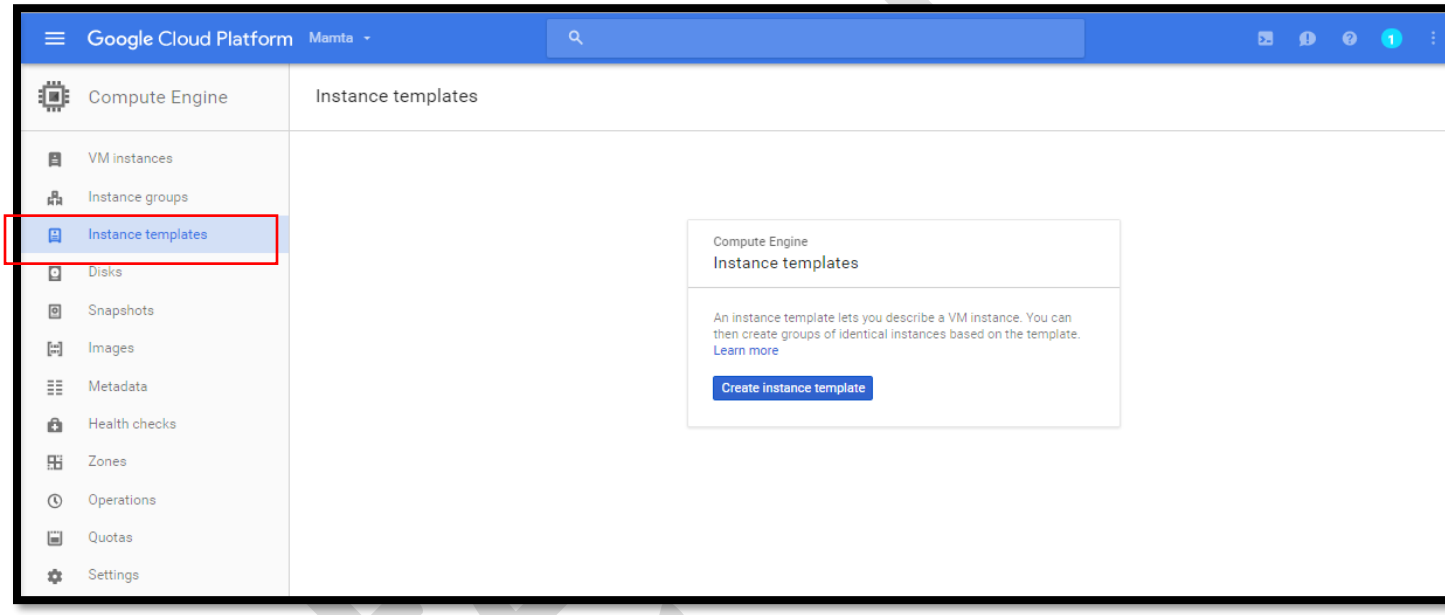
Estimated time: 15 mins

Solution:

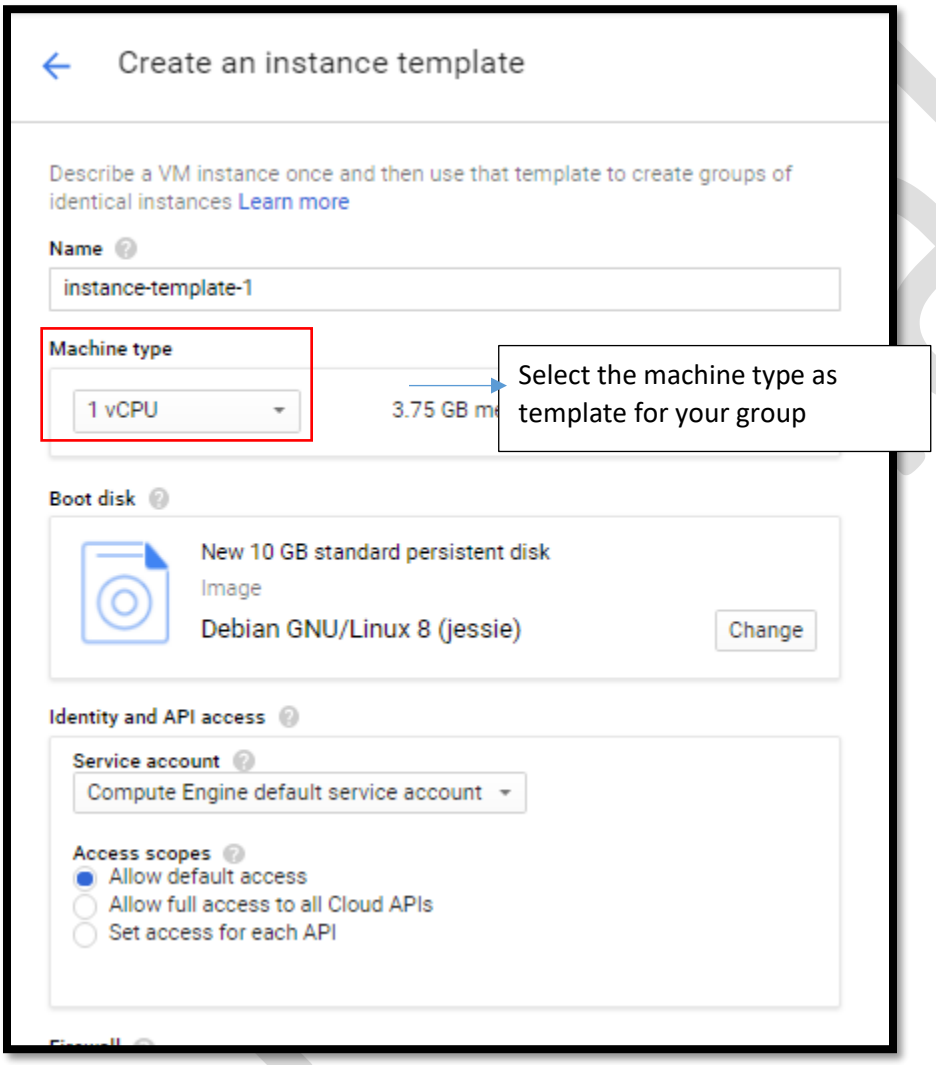
Follow below steps to create an Instance template:

Internal

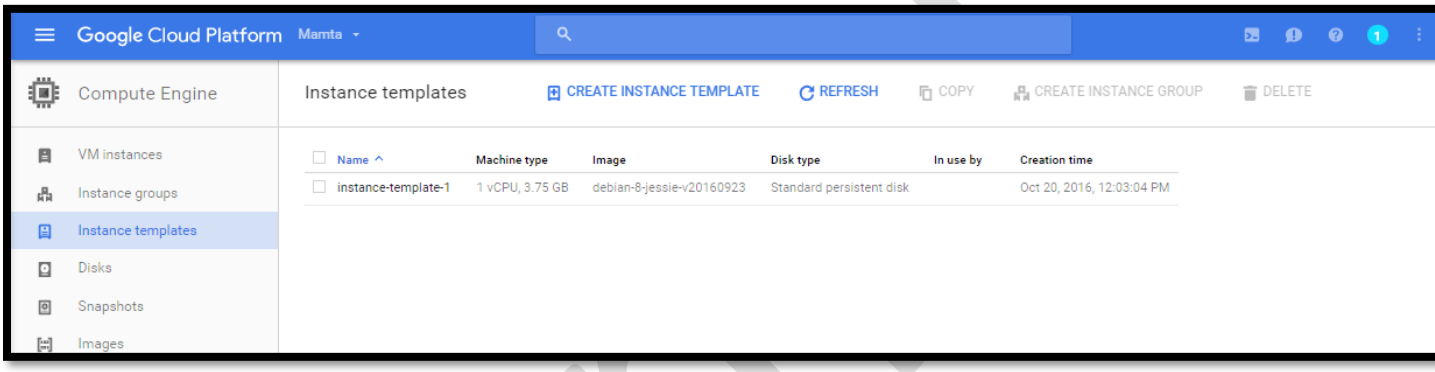
1. *Navigate to Compute Engine and select Instance Template as shown below :*



2. *Once you select to create instance template you will be redirected to below screen where you can fill the required details for creating an instance. These detail will be saved as a template which will be used to create instances for your group.*

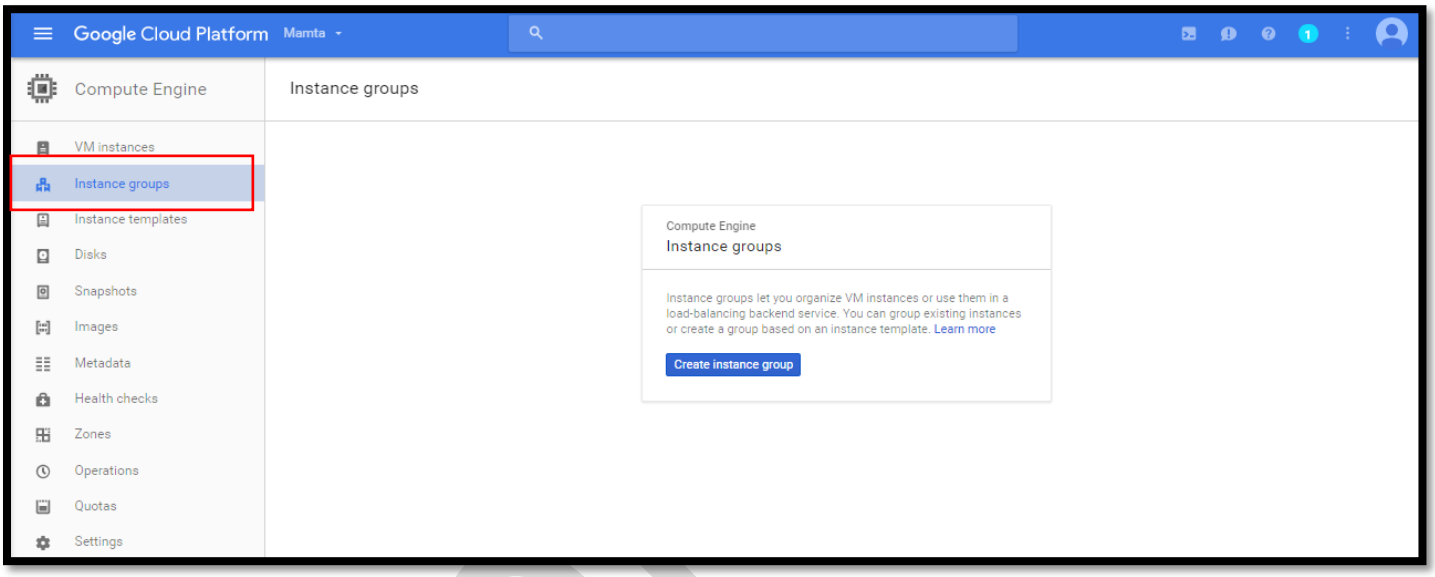

Create an instance template
Describe a VM instance once and then use that template to create groups of identical instances Learn more
Name
instance-template-1
Machine type
1 vCPU
3.75 GB me
Select the machine type as template for your group
Boot disk
New 10 GB standard persistent disk
Image
Debian GNU/Linux 8 (jessie)
Change
Identity and API access
Service account
Compute Engine default service account
Access scopes
Allow default access
Allow full access to all Cloud APIs
Set access for each API

3. *Navigate back to Instance Template tab in the left pane so to review your template with required configuration as shown below :*

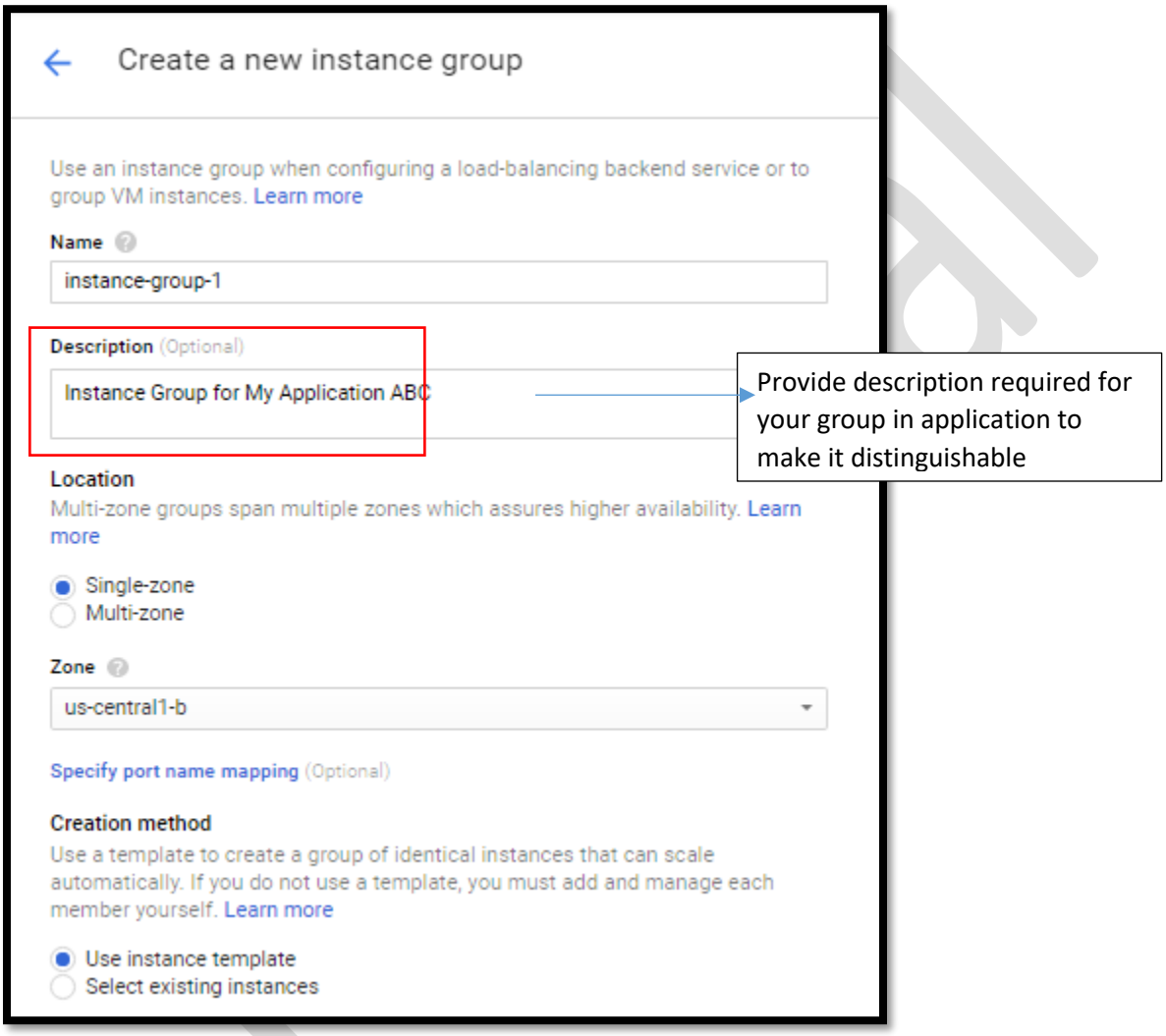


4. *As now template has been created, the same will be used to create Instance group. Select Instance Group tab from left pane to begin :*

Internal

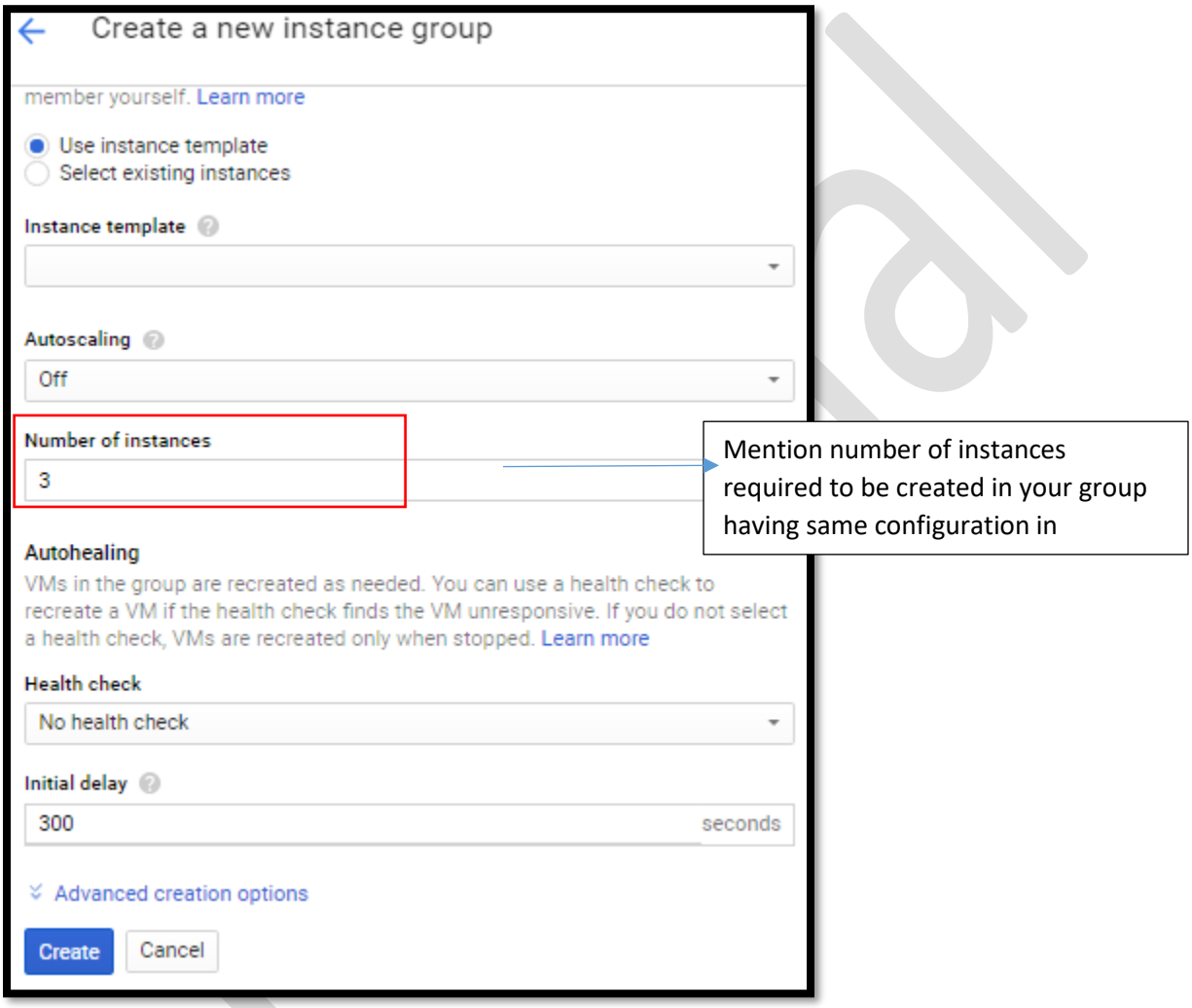


5. *Provide required details where group will be created which include : Location- it can be single-zone and multi-zone according to your requirement for deploying app :*


Create a new instance group
Use an instance group when configuring a load-balancing backend service or to group VM instances. Learn more
Name
instance-group-1
Description (Optional)
Instance Group for My Application ABC
Provide description required for your group in application to make it distinguishable
Location
Multi-zone groups span multiple zones which assures higher availability. Learn more
Single-zone
Multi-zone
Zone
us-central1-b
Specify port name mapping (Optional)
Creation method
Use a template to create a group of identical instances that can scale automatically. If you do not use a template, you must add and manage each member yourself. Learn more
Use instance template
Select existing instances

1. *Select "Use Instance template" button so to select created template for group else existing list of instances can be used to be put in group:*

Internal


Create a new instance group
member yourself. Learn more
Use instance template
Select existing instances
Instance template
Autoscaling
Off
Number of instances
3
Autohealing
VMs in the group are recreated as needed. You can use a health check to recreate a VM if the health check finds the VM unresponsive. If you do not select a health check, VMs are recreated only when stopped. Learn more
Health check
No health check
Initial delay
300
seconds
Advanced creation options
Create
Cancel
Mention number of instances required to be created in your group having same configuration in

Summary: In this assignment, you have created an Instance template and using the same you have created Instance Group.

Assignment 2a: Adding a Persistent Disk to an Instances

Objective: *use of additional storage in your instances by adding a Persistent Disk when it is running out of space*

Problem Description: *Before hosting your application or service on Google Compute engine (GCE), you need to add secondary storage space to your existing instances when additional space is required. Compute Engine controls the hardware behind persistent disks so that you can add disks without worrying on redundancy or even striping.*

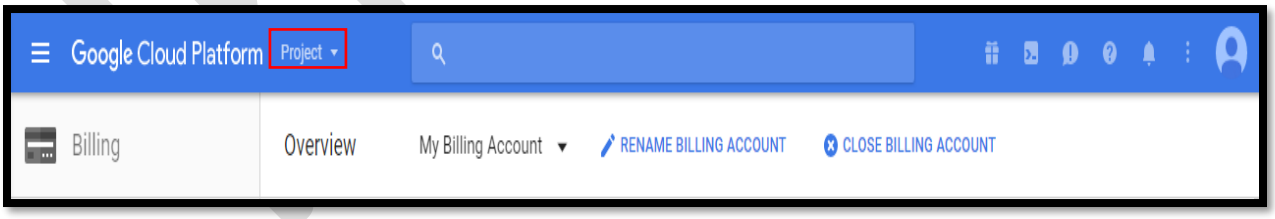
Perform this step with your account:

Estimated time: 20 minutes

Solution:

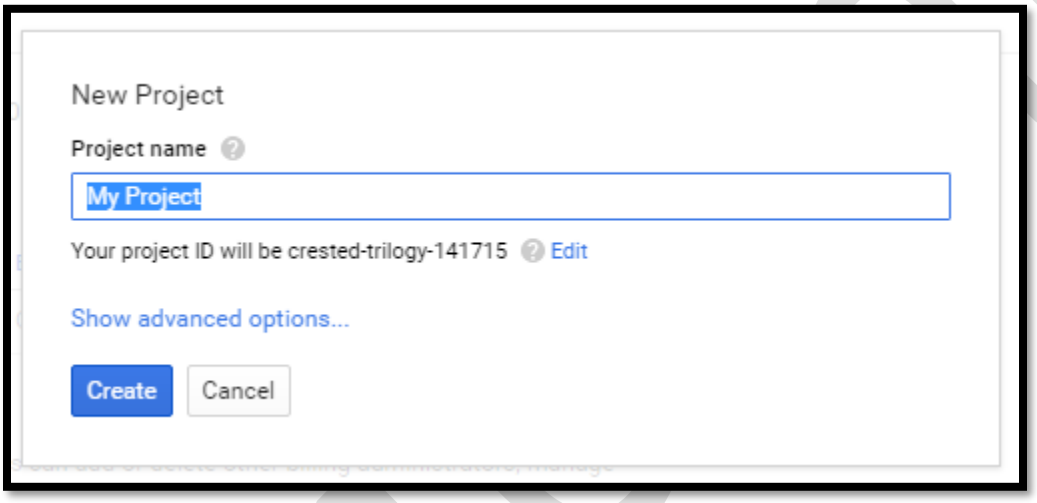
1. ***Instances and Project:***

Any instance in GCP belongs to a Project, a project can have one or more instances.



Internal

- *Select project tab and select "**create project**" to create new project by providing following details :*

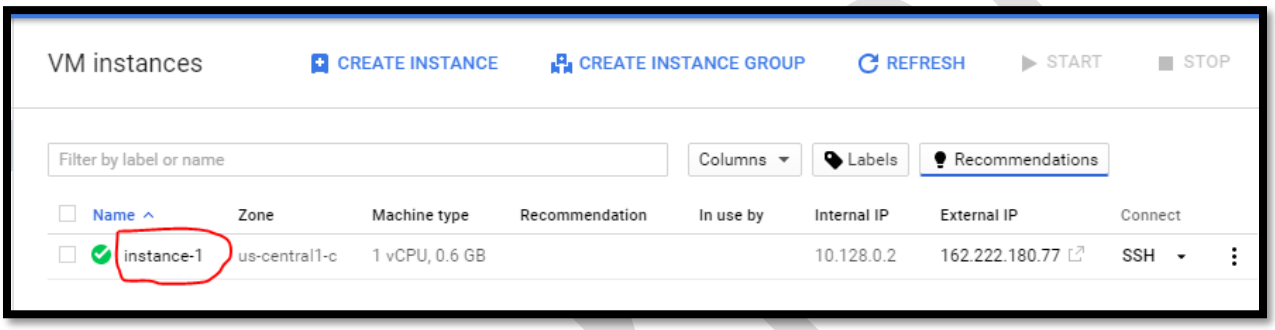


Project id will be auto created. Note the Id as it will required for configuration and deployment.

2. ***Selecting an Instance:***

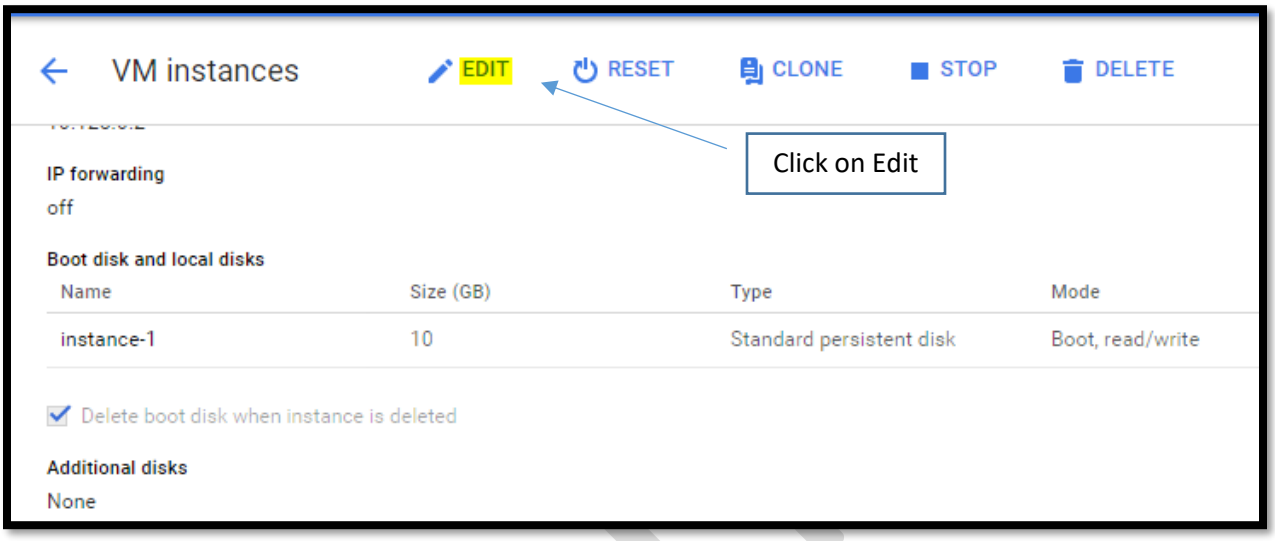
Before attaching a persistent disk to your instances, it is a must that you should create and start an instance

- *Navigate to the instances page*
- *Select the instance from the given list where you need to add a disk*

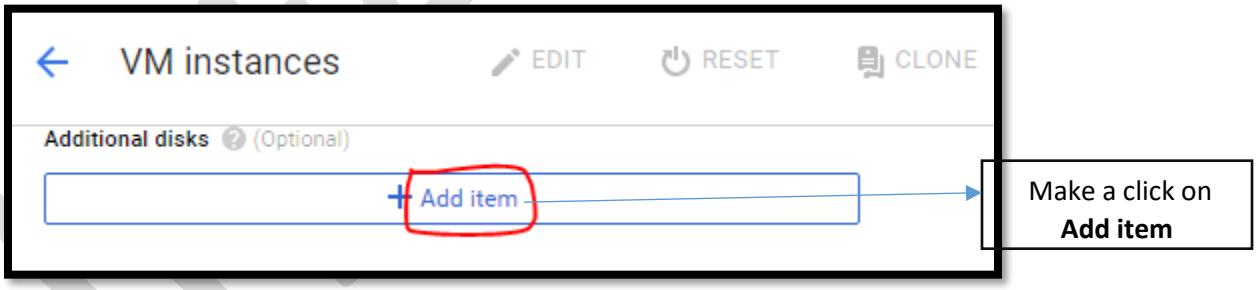


- *Once you select the instance, navigate to the instance details page as below*

Internal



- *Navigate to the Additional Disks part as below*



- *After opening the add item, now you need to click the **select a disk** from the drop menu which is present in the **Name** colulmn for additional lists*

- *click on **Create disk** and you could see a similar box as below*

Internal

Create a disk

Name

disk-1

Description (Optional)

Disk Type

Standard persistent disk

Source type

Image | Snapshot | None (blank disk)

Source image

Size (GB) (Optional)

Estimated performance

Operation Type	Read	Write
Sustained random IOPS limit		
Sustained throughput limit (MB/s)		

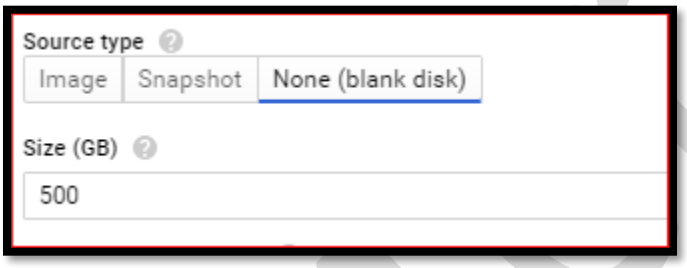
Encryption

Create | Cancel

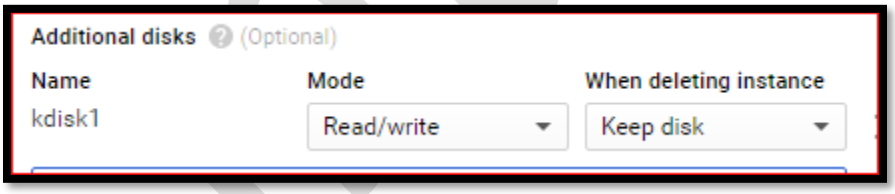
- *Provide the required details by specifying properties like name, disk type etc.*

- *Mention a valid name for your disk by meeting the below requirements*
    - ***Name:*** *must start with a lowercase letter followed by upto 62 lowercase letters, numbers or hypens and cannot end with a hypen.*
    - ***Disk Type:*** *proceed with default selection –standard persistent disk.*

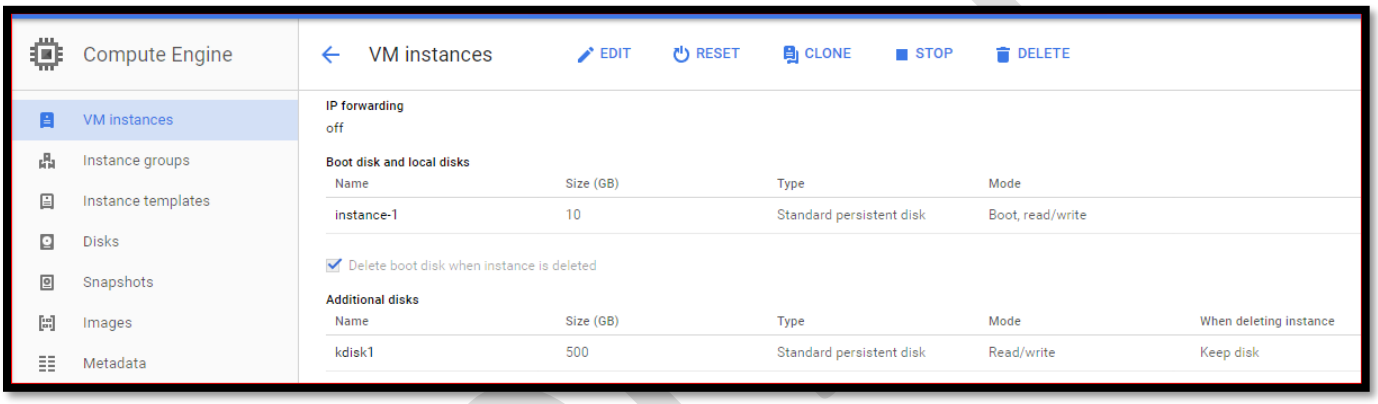
- *Select* ***Blank disk*** *option tab from* ***Source type****.*



- *After configuring the properties, please click on* ***create disk.***



- *Once you have created disks, you need to save the changes that require for your instance*
- *By clicking on **save** button you can accomplish the same which is available at the bottom of your instance page*



Summary of this assignment: *In this assignment, you have understood creating, configuring details of persistent disk space to an instance.*

Note: once you create and add the disk to your instance, ensure that it is formatted and mounted before your Operating system make use of that storage space

Assignment 2b. Formatting and Mounting a Persistent Disks

Objective: *Prepare the operating system to use the available storage space by formatting and mounting the persistent disk*

Problem Description: *When your application is in need of additional storage space to your existing instances, you can attach a secondary persistent disks by creating and attaching to the instances. After attaching it, formatting and mounting is a must and then only your operating system can avail the storage space.*

Persistent Disks are durable storage device that function similary to the physical disks in a server.

Formatting process will be different on both Linux and Windows instances.

We shall be focusing on ***Linux instances****.*

Estimated time: 15 minutes

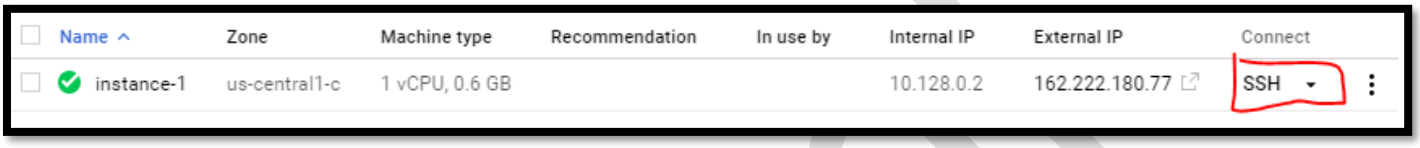
Solution:

Prepare your Linux instance for formatting and mounting the new disk. Any type of partition format can be used, howeve a simpler one is to create a distinct ***ext4 file*** *system without a partition table.*

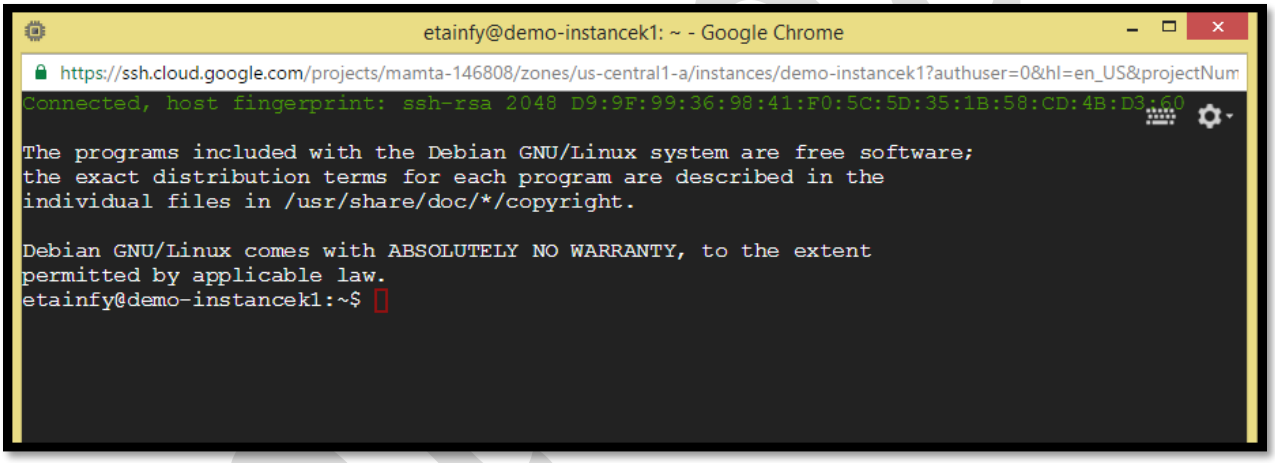
- *Navigate to the VM instance page.*

Internal

- *For connecting to your instance which has the new attached disk, click on **SSH** tab in the connect column as below*

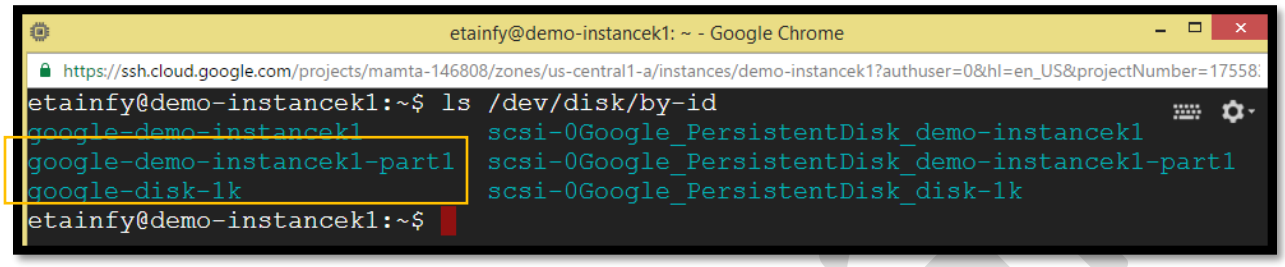


- *A separate terminal will get open for your instance as below.*



- *In the terminal, use the **ls** tool to display the no.of disks that are available to your instances and select the disk which you would like to format and mount.*

  - *$ ls /dev/disk/by-id*



where [kdisk1] persistent disk's name where you attached to the instance.

- *The disk ID usually consists of your persistent Disk name along with a google- prefix or a **scsi-0Google_PersistentDisk_ prefix**.In our example, we used the ID with the google- prefix to to specify your disk*
- *Using the mkfs tool, format the disk with a single **ext4** filesystem.*
    - *Using this command, all the data from the specified disk can be deleted.*
    - *Run the below sudo command with –E flag*
        - *$ sudo mkfs.ext4 -F -E lazy_itable_init=0,lazy_journal_init=0,discard /dev/disk/by-id/google-[**DISK_NAME**]*

Where DISK_NAME is your Persistent Disk Name that you are formatting

Internal

```
etainfy@demo-instancek1: ~ - Google Chrome
https://ssh.cloud.google.com/projects/mamta-146808/zones/us-central1-a/instances/demo-instancek1?authuser=0&hl=en_US&projectNumber=17558
etainfy@demo-instancek1:~$ sudo mkfs.ext4 -F -E lazy_itable_init=0,lazy_journal_
init=0,discard /dev/disk/by-id/google-disk-1k
mke2fs 1.42.12 (29-Aug-2014)
Discarding device blocks: done
Creating filesystem with 131072000 4k blocks and 32768000 inodes
Filesystem UUID: 38520875-0124-4e7a-81ef-3c4bbf253eb7
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624, 11239424, 20480000, 23887872, 71663616, 78675968,
        102400000

Allocating group tables: done
Writing inode tables: done
Creating journal (32768 blocks): done
Writing superblocks and filesystem accounting information: done
```

- *Create a new directory that work as the mount point for the disk.*
  - *$ sudo mkdir -p /mnt/disks/[MNT_DIR]*

Here MNT_DIR is the directory where you can mount your PD

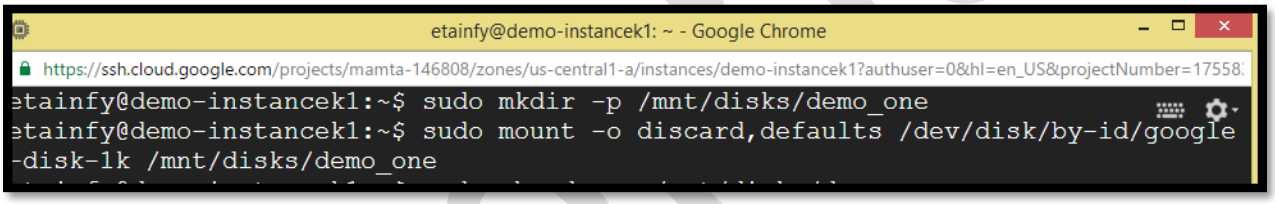
- *With the help of **mount** command, you can mount the disk to the instance along with the enablement of **discard** option*

- *Run the below command to mount the disk*
    - *sudo mount -o discard,defaults /dev/disk/by-id/google-[DISK_NAME] /mnt/disks/[MNT_DIR]*

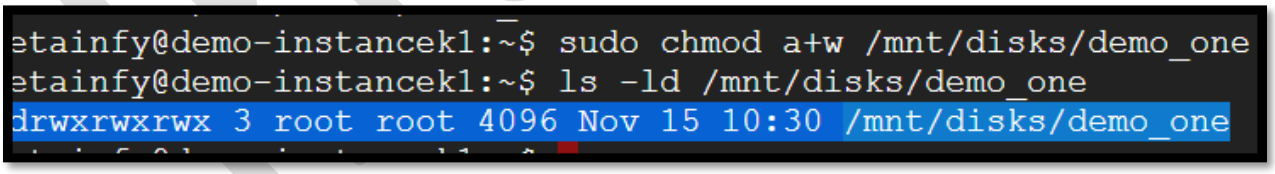
where :-

[DISK_NAME] is the Persistent Disk name which you are formatting

[MNT_DIR] is the directory which your PD will get mounted

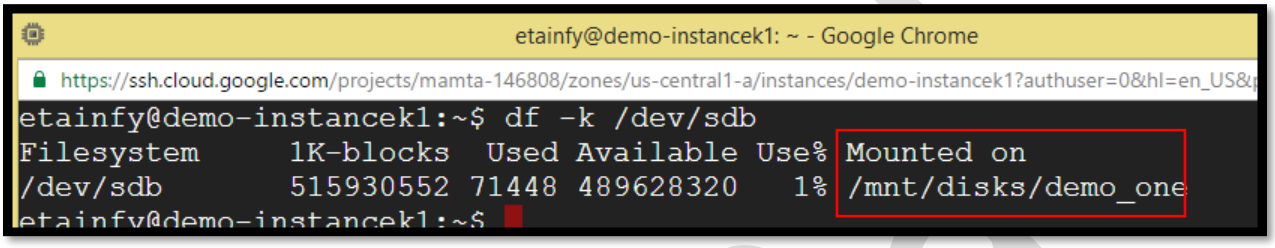


- *Provide read and write access to the device. In this example, allow write access to the device for all users.*
    - *sudo chmod a+w /mnt/disks/[MNT_DIR]*



Internal

- *By giving **df –k /dev/sdb** in linux prompt, you can crossverify the mountings*



Summary of this assignment: *In this assignment, you have understood the formatting and mounting a persistent disk to use the additional storage to your instances.*

Assignment 3: GCP- Images

Objective: *To create a private image from existing root persistent disk.*

Prerequisite:

We need to have a GCP account and need to have an instance build from an existing public image.

Internal

Problem Description:

Create a sample project with GCP and create a private image from existing root persistent disk. In this assignment you will be guided how to create a private image on a Linux Instance.

Estimated time: 15 mins

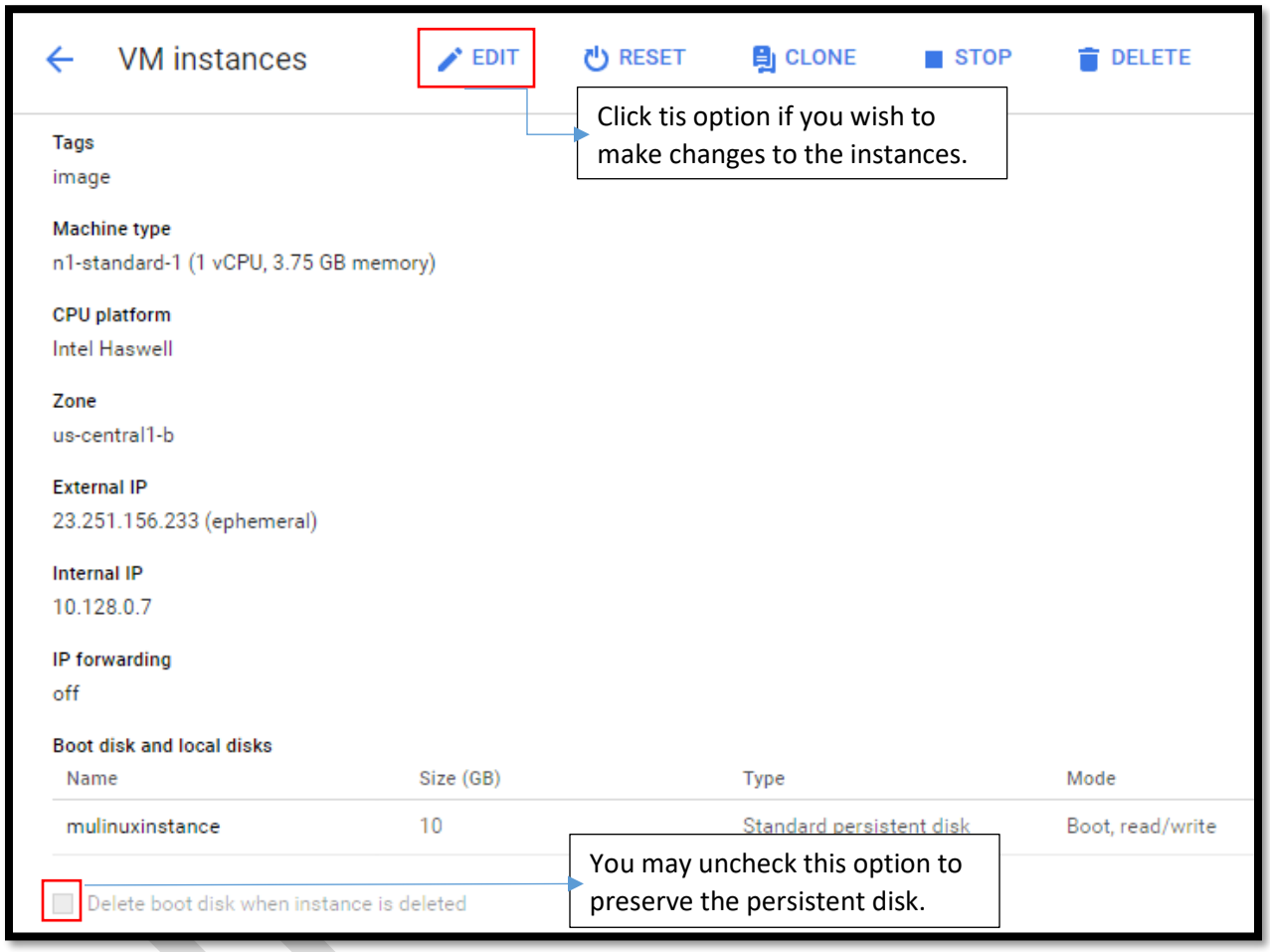
Solution:

1. *Once you have created a Linux Instance in your project with the configuration details as show below, you are ready to get started by creating a private image of the instance.*

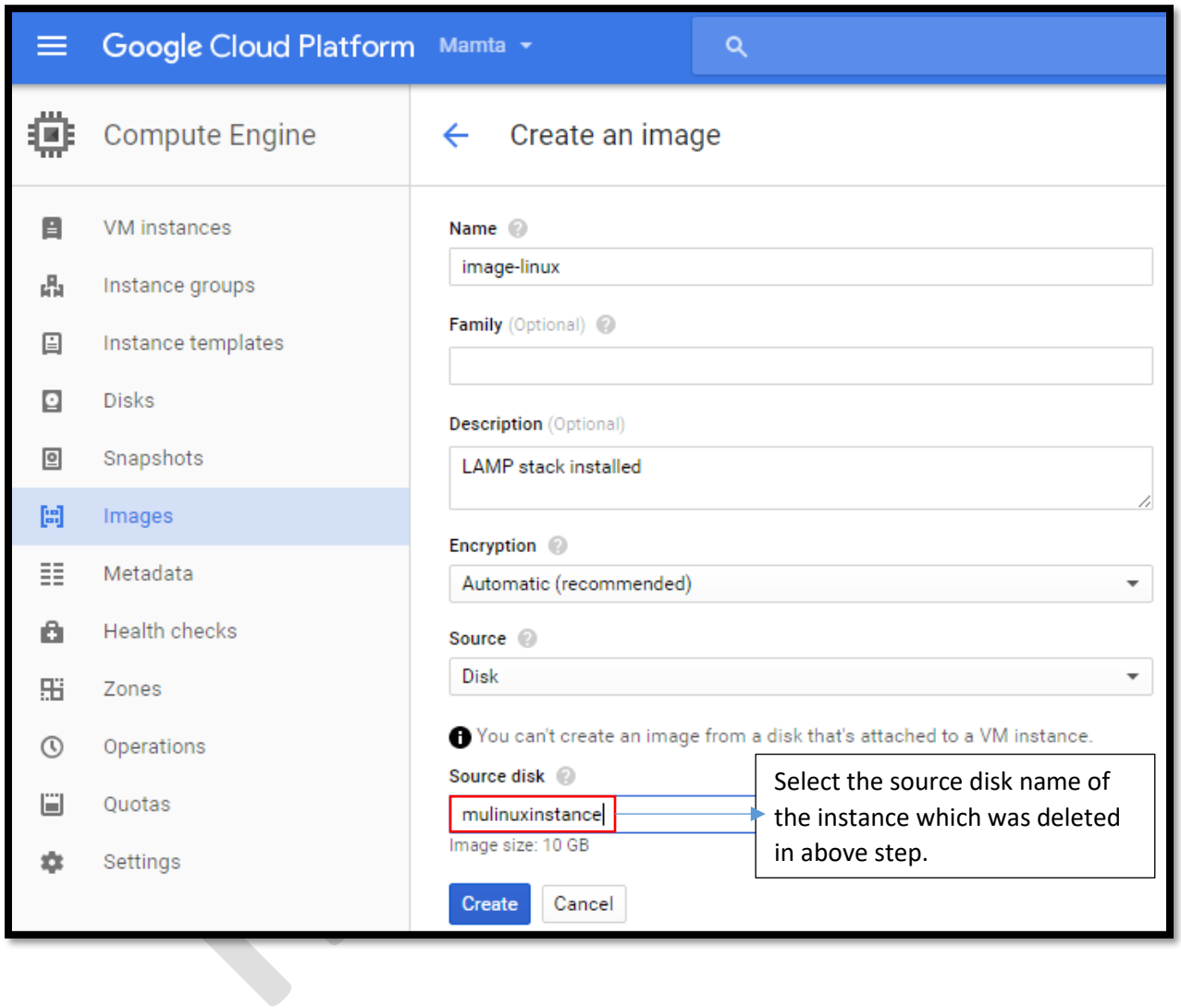
.

As private image will be created from the root persistent disk thus to detach the disk delete the instance to which it is attached. Make sure you preserve the root disk before deleting by unchecking the option "Delete boot disk when instance deleted".

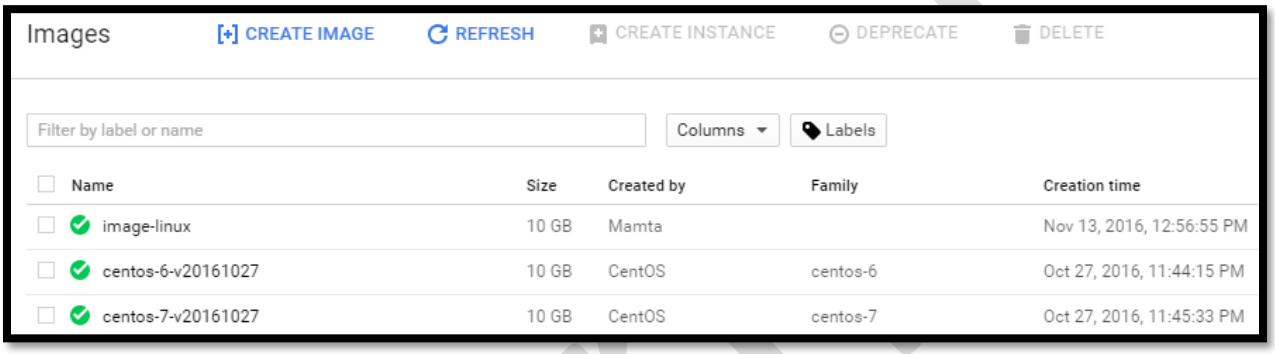
Internal



2. *Go to Image Page in console to create image and fill the desired properties as shown below:*

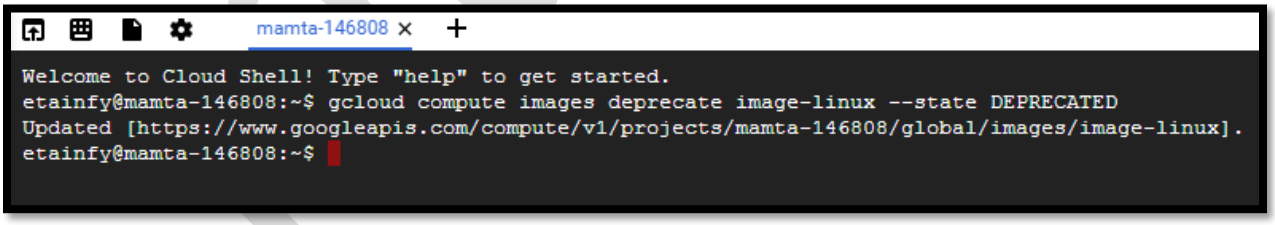

Google Cloud Platform
Mamta
Compute Engine
Create an image
VM instances
Instance groups
Instance templates
Disks
Snapshots
Images
Metadata
Health checks
Zones
Operations
Quotas
Settings
Name
image-linux
Family (Optional)
Description (Optional)
LAMP stack installed
Encryption
Automatic (recommended)
Source
Disk
You can't create an image from a disk that's attached to a VM instance.
Source disk
mulinuxinstance
Image size: 10 GB
Create
Cancel
Select the source disk name of the instance which was deleted in above step.

3. *You can review the image by moving back to the home page of Image and check details:*



4*. Now once image is created change the depreciation status. Each depreciation status helps you transition users away from unsupported image. Set the depreciation status by the command in gcloud as shown below:*

gcloud compute images deprecate image-linux --state DEPRECATED



Now the image is ready to be used to start an instance.

Summary of Assignment: *Developer will be able to successfully create a Private image use the same to start a new instance.*

Assignment 4a: Creating a cross-region HTTP load balancer

Highlights:

- *The load balancer sends traffic to healthy instances only*
- *Load balancing does keep instances in sync. Using Deployment manager you can ensure the instances have consistent configurations and data*

Demo steps:

Step 1: Creating a cross-region HTTP load balancer

You can use load balancing to distribute user requests among set of instances. The instances can be in same region or different regions. In this section, you will focus on configuring cross-region HTTP load balancing in GCP, which will balance the requests among instance group spanned across regions.

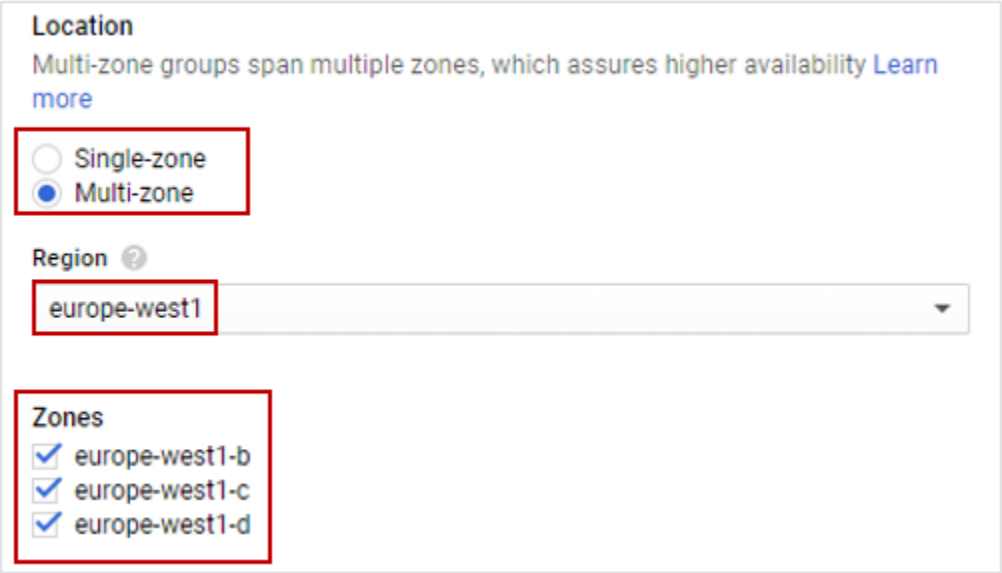
Here you will create a load balancer with instances enrolled under two different instance groups.

To start with the configuration, you need two instance groups, one with instances in single zone and the second with instances in multiple zones.

The first instance group that you can use will be the demoinstancegroup1, that was created in section 2.1.

You will also create another instance group demoinstancegroup2 with multi-zone support. The changes in configuration is as shown below:

Internal



After this configuration, you will have two instance groups as shown below:

Name	Zone	Creation time	Instances	Template
demoinstancegroup1	us-central1-c	14 Nov 2017, 19:24:29	2	demotemplate1
demoinstancegroup2	europe-west1 (3/3 zones)	14 Nov 2017, 19:39:24	2	demotemplate1

Next you will proceed to create an HTTP load balancer using the above two instance groups as back-end services.

Assignment 4b: Creating a cross-region HTTP load balancer

Highlights:

- *GCP Internal Load balancing is architected using Andromeda, Google's virtualization platform*
- *With GCP load balancing, you can have your entire application available using a single global IP address resulting a simplified DNS setup*

Demo steps:

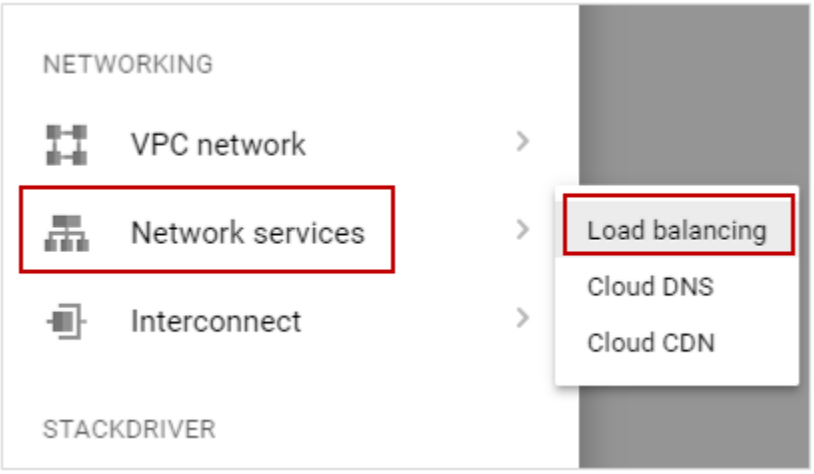
Creating a cross-region load balancer involves the below steps:

1. *Configure the load balancer details*
2. *Attach back-end services to load balancer*
3. *Add front-end forwarding rules which maps load balancer IP to instance IP addresses*

Step 1: Configure load balancer details

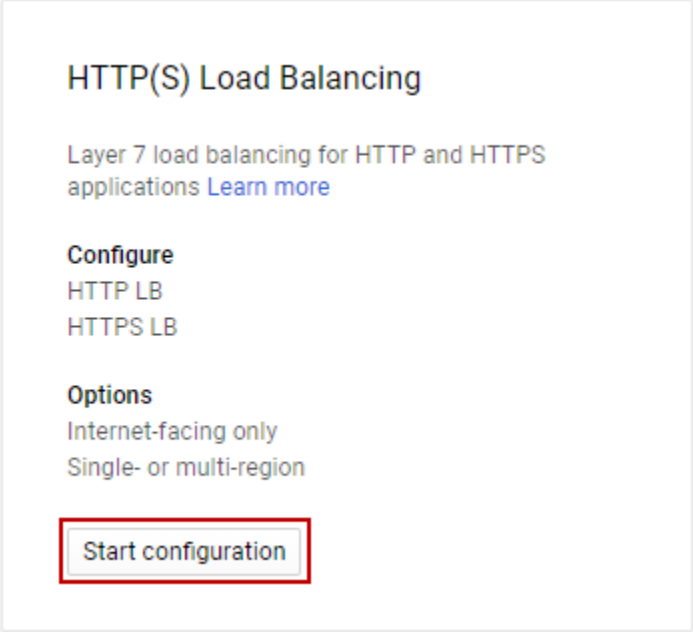
- *Navigate to **Load balancing** under **Network services**. In the pop-up screen, select **Create load balancer** button*

Internal



- *In the subsequent page, among the different types of load balancing, select the **HTTP(S) Load balancing** and click on **Start Configuration** button*

Internal

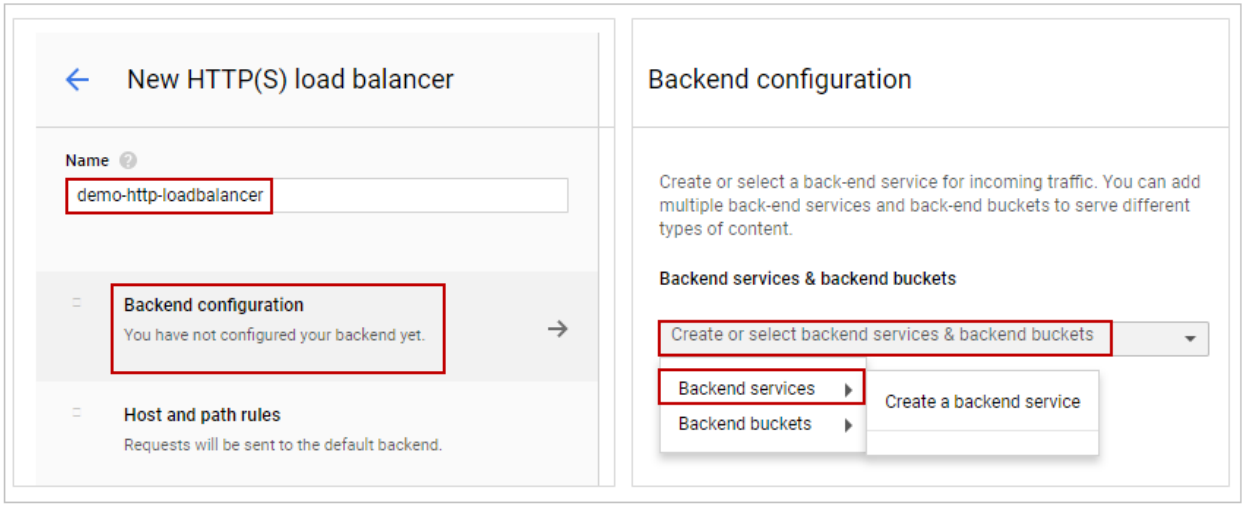


In the screen that appears, provide a name for the load balancer. Next we have to configure Backend and front end forwarding rules.

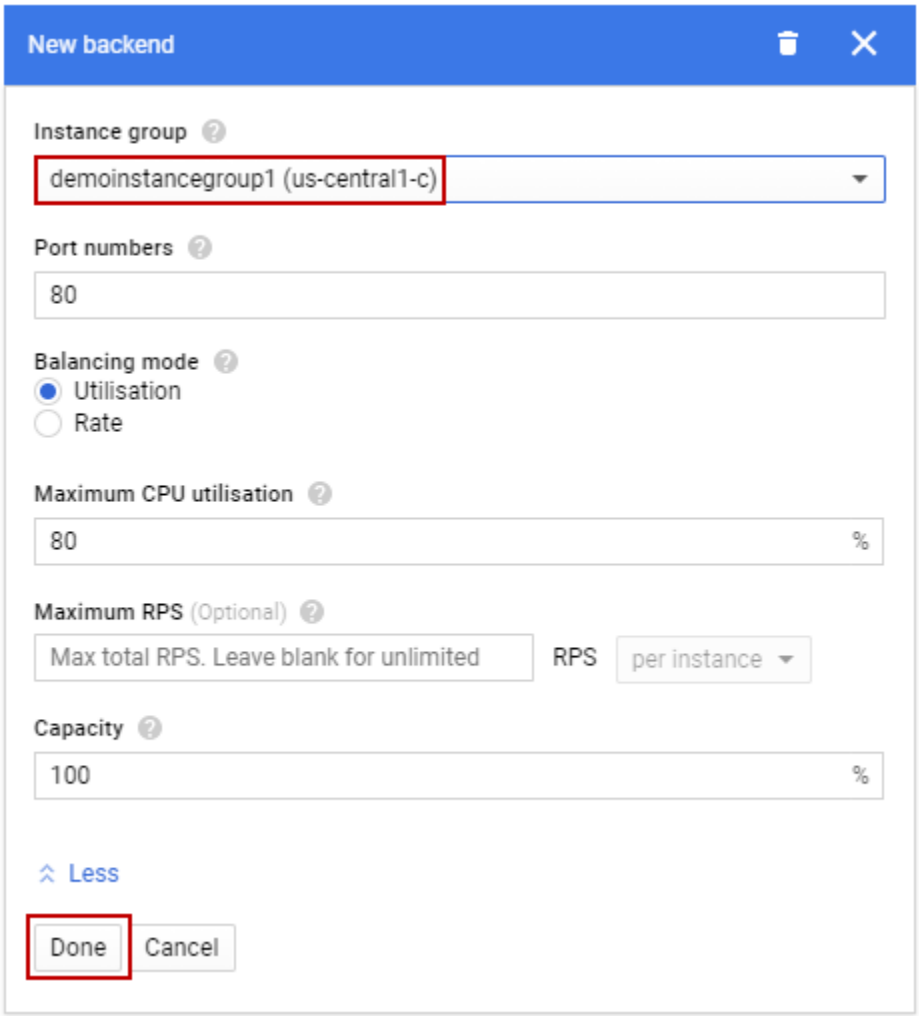
Step 2: Backend configuration

- *Select **Backend configuration** section, which will display an extended configuration box towards the right side*
- *From the drop down, select **Create or select backend services & backend buckets***
- *Under Backend services, select **Create a backend service***

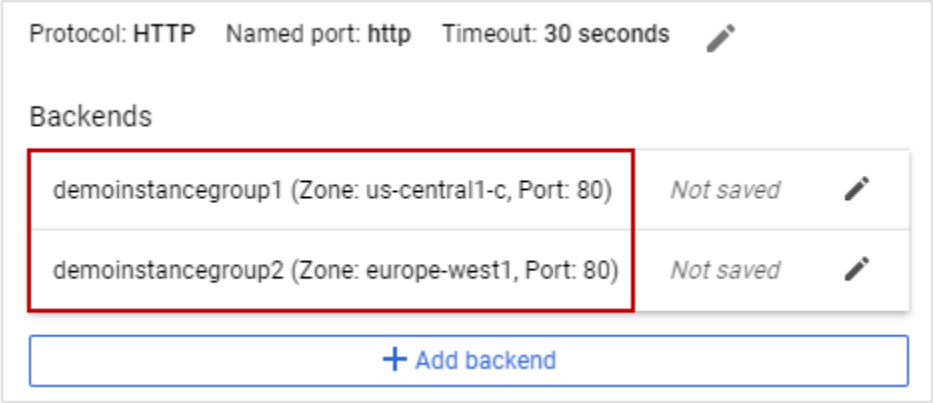
Internal



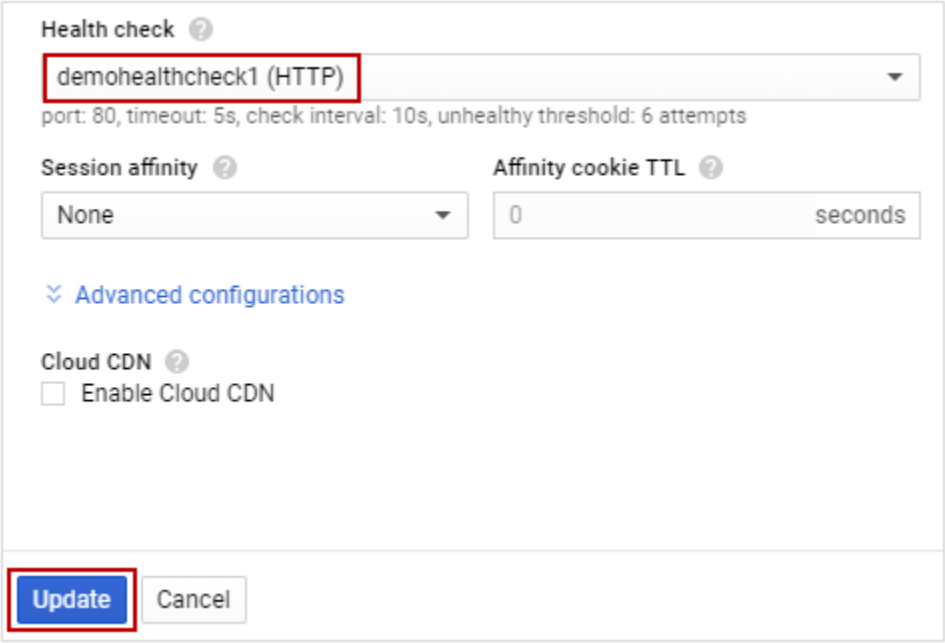
Provide a name for the backend service group. You will get a new window New backend. Select the first instance group as the first back end service and configure values as shown below. Click on Done button


New backend
Instance group
demoinstancegroup1 (us-central1-c)
Port numbers
80
Balancing mode
Utilisation
Rate
Maximum CPU utilisation
80
%
Maximum RPS (Optional)
Max total RPS. Leave blank for unlimited
RPS
per instance
Capacity
100
%
Less
Done
Cancel

- *Similarly create another back-end using the Add backend button and the second instance group. All other fields can take the default values*
- *Once done, you will have two back-ends added under back-end configuration as shown below.*



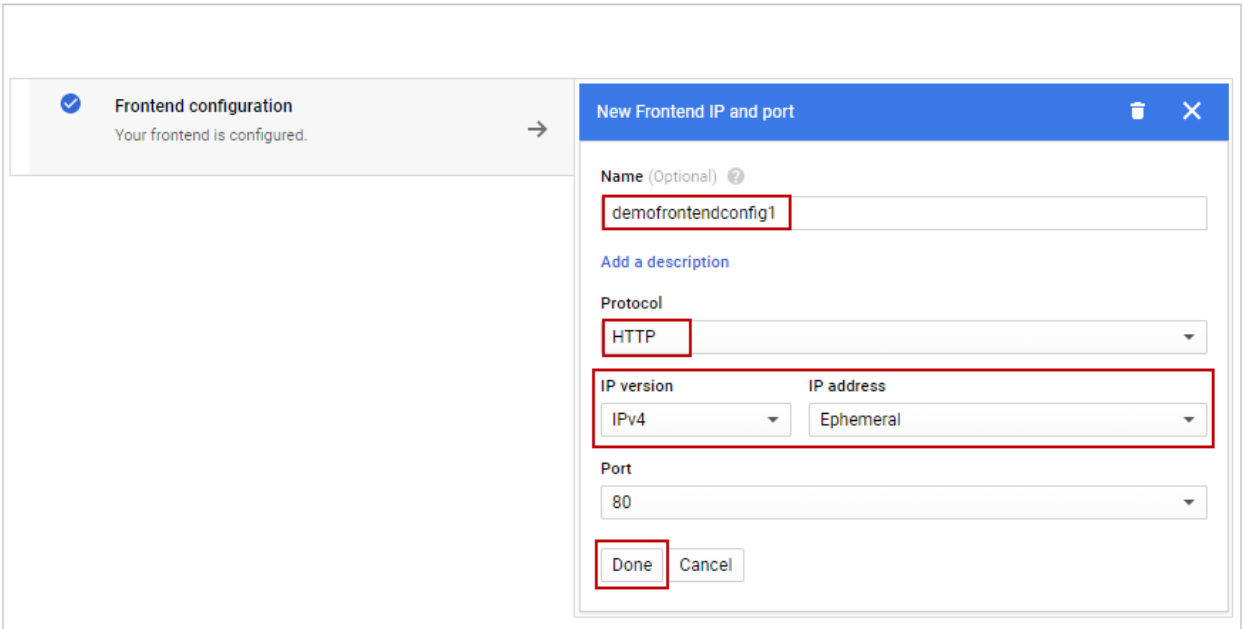
For Health check field, select the demohealthcheck1 that you created during section 2.1. Then click on Update button



Step 3: Create front end forwarding rule(s)

In the Host and path rules section, continue with the default values

Select Frontend configuration section, which will show an extended window. Provide the values as shown below

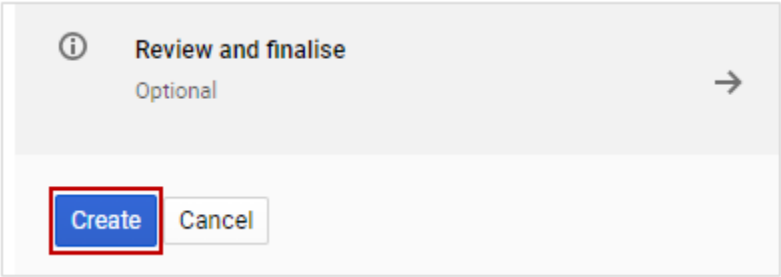


Here, you can add both IPv4 and IPv6 addresses for HTTP protocol. Under the IP address field, you have the option to reserve a global static external IP for the load balancer. Here, you will select Ephemeral for practice purpose.

Similarly, you can configure IPv4 and IPv6 addresses for HTTPS protocol also if required.

Once completed, click on Done.

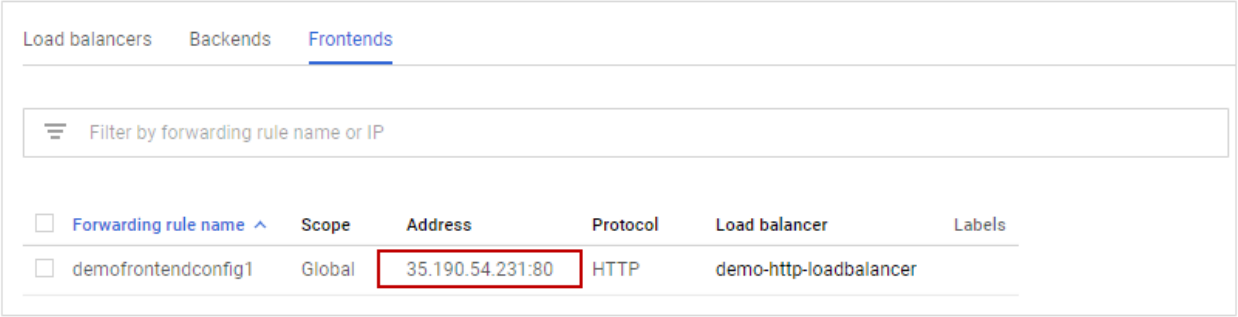
After review and if no changes, click on Create button



Step 4: Access the load balanced web application

You can access the web application deployed on all the server machines using the load balancer IP address.

Click on the newly created demo-http-loadbalancer and navigate to Frontends tab



After few seconds, the health checks will complete and load balancer will be active. Copy the address and port and paste it to a browser url. You will see the response from the web application from any one of the servers under the load balancer

Assignment 5. Create an instance template

Highlights:

- *Using instance template, create a managed instance group.*
- *Any number of managed instance groups can be created from a single instance template*

Demo steps:

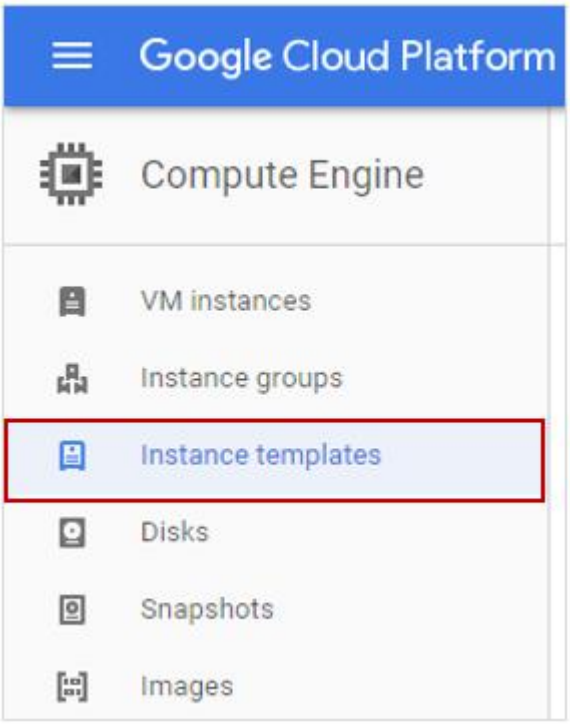
A managed instance group has the extra capabilities to support auto-scaling, auto-healing and load balancing.

Prerequisite to create a managed instance group is to create an instance template.

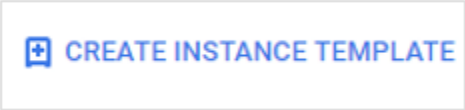
To effectively manage a group of instances under a group, it is always advisable to create instances having a common standard configuration. This standard configuration is called as an instance template. The instance template defines a machine type, a boot disk, access rules and firewalls.

Step 1:

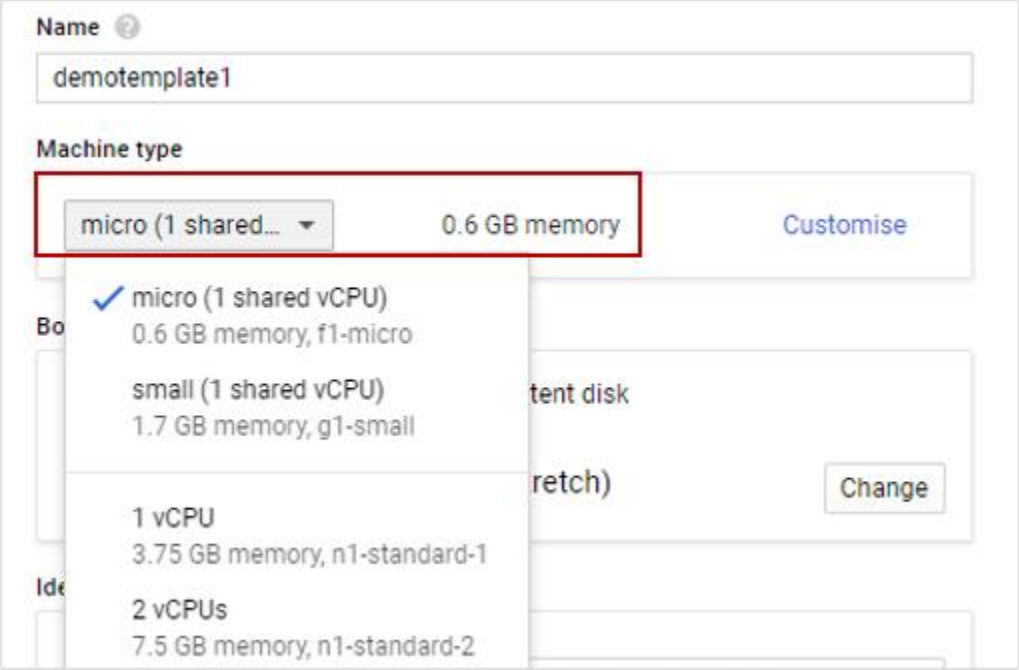
- ***Select Instance templates from the navigation pane under Compute Engine***



Click on create instance template from the top of the page

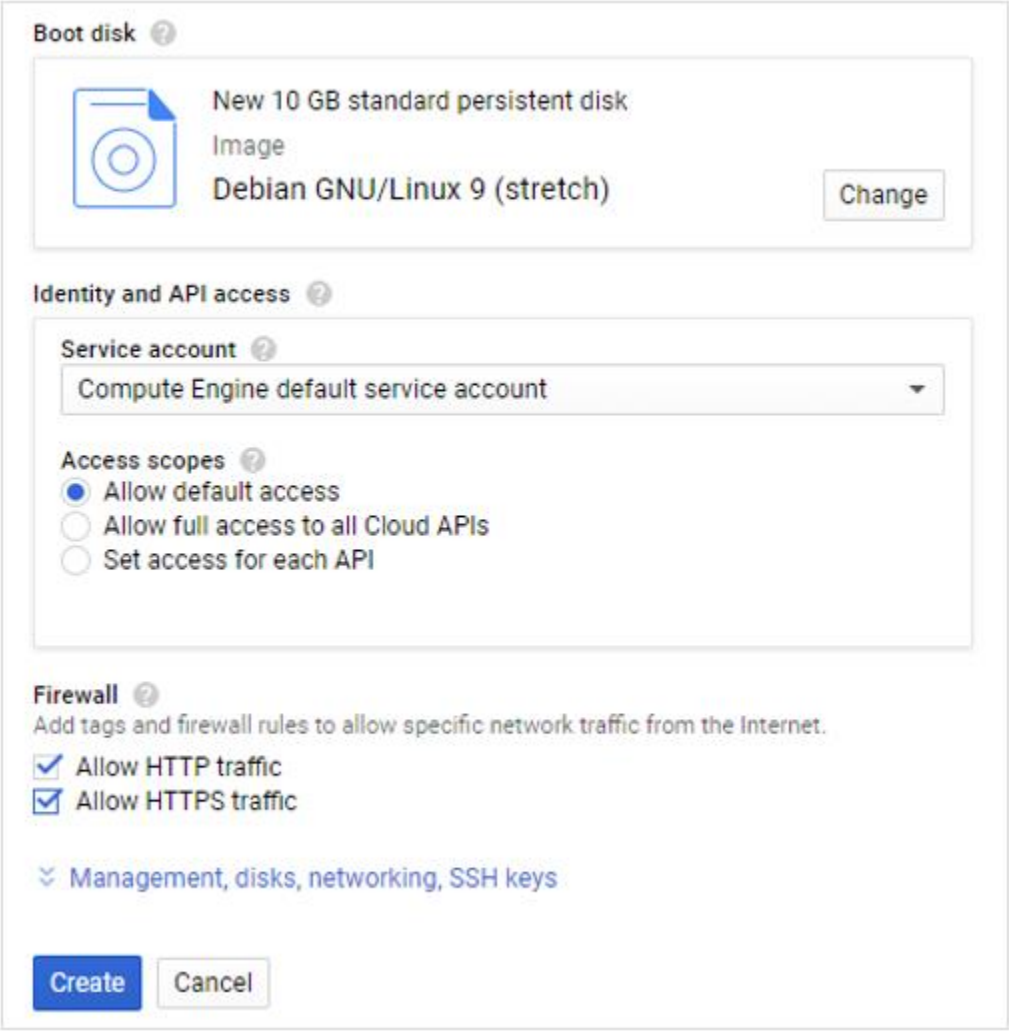


Provide a name for the template and select the machine type



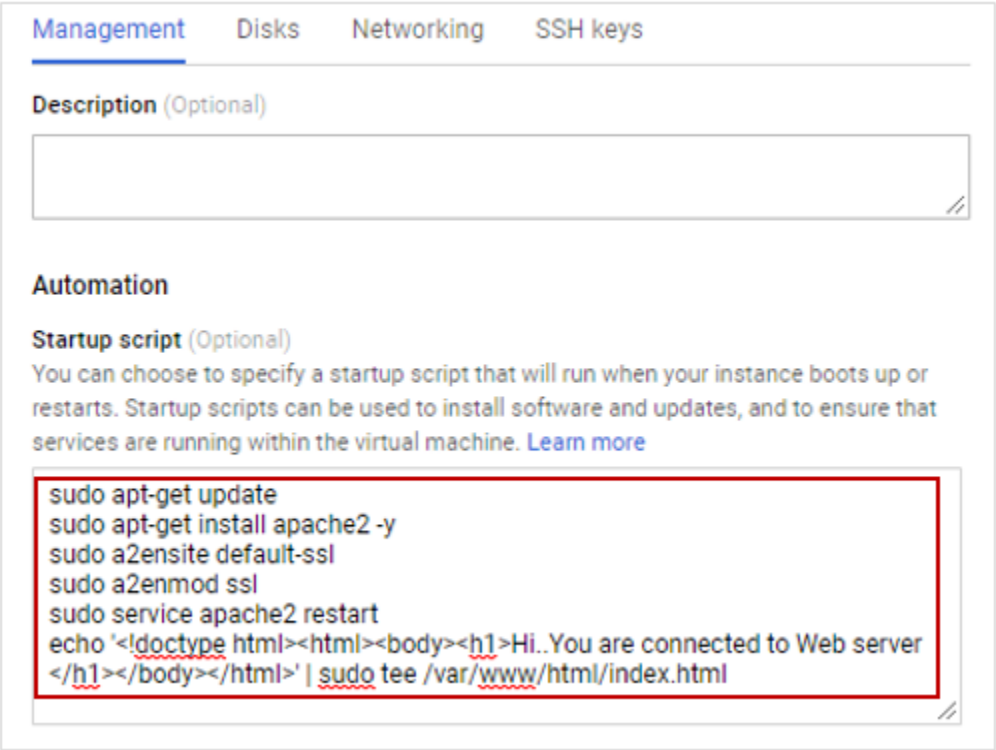
Step 2:

- ***Select the Boot disk, Access scope and Firewall rules***


Boot disk
New 10 GB standard persistent disk
Image
Debian GNU/Linux 9 (stretch)
Change
Identity and API access
Service account
Compute Engine default service account
Access scopes
Allow default access
Allow full access to all Cloud APIs
Set access for each API
Firewall
Add tags and firewall rules to allow specific network traffic from the Internet.
Allow HTTP traffic
Allow HTTPS traffic
Management, disks, networking, SSH keys
Create
Cancel

Step 3:

- ***Expand 'Management, disks, networking and SSH Keys'. Provide a start up script which will configure an Apache server and deploy a simple html page in the server***



Click on Create. This creates a new instance template demotemplate1, which will be available under the list of instance templates

Name	Machine type	Image	Disk type	In use by	Creation time
demo-template	1 vCPU, 0.6 GB	debian-9-stretch-v20170816	Standard persistent disk		28 Aug 2017, 14:43:04
demotemplate1	1 vCPU, 0.6 GB	debian-9-stretch-v20171025	Standard persistent disk		14 Nov 2017, 11:08:43
instance-template-1	1 vCPU, 3.75 GB	imgcreatdemo29aug17	Standard persistent disk		29 Aug 2017, 16:34:09
instance-template-2	1 vCPU, 0.6 GB	debian-9-stretch-v20171025	Standard persistent disk		13 Nov 2017, 15:20:52

Assignment 5a. Configure Instance group

Highlights:

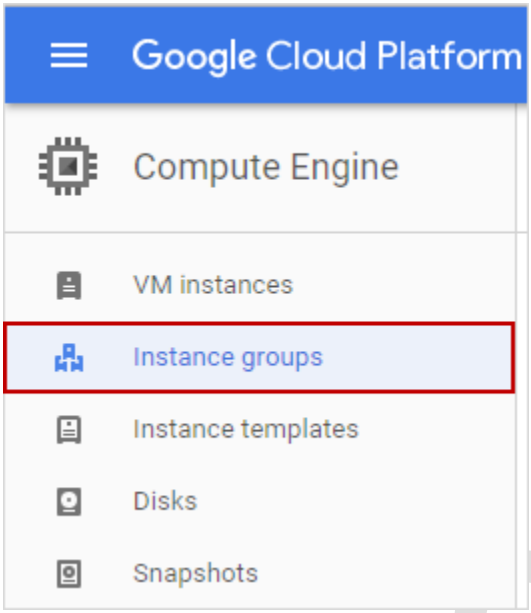
- *Automatic scaling according to pre-defined conditions can be created during instance group creation*
- *When existing VMs become un-responsive, auto-scaling will create new virtual machines only if proper health check configuration is enabled*

Demo steps:

In this step, you will focus on creating an instance group based on the demotemplate1 that you created during the previous step.

Step 1:

- *Select Instance groups from the left navigation pane under Compute Engine*

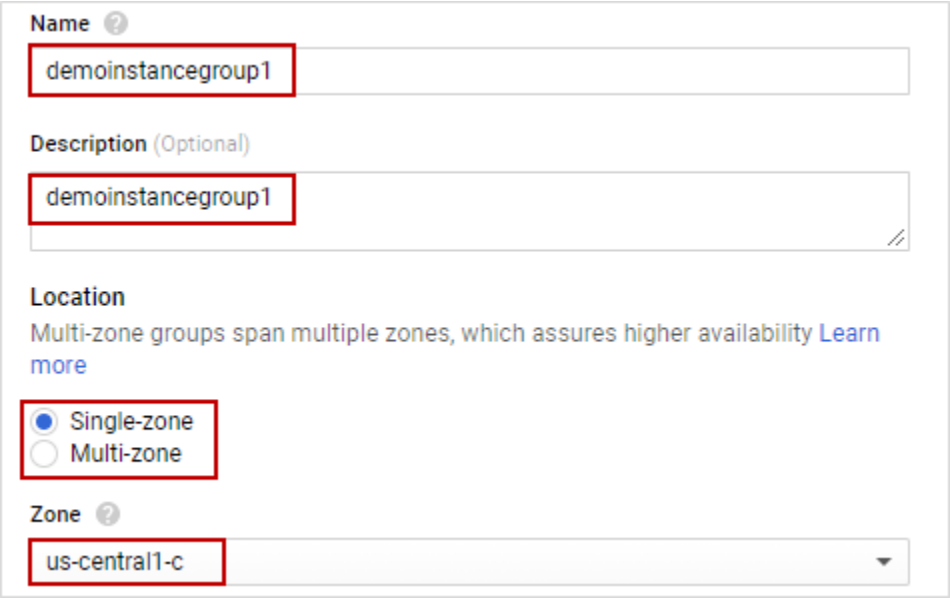


Click on the CREATE INSTANCE GROUP link on the top pane of the page

Step 2:

- *Provide the required details such as Name, Description, Location and Zone as required. Selecting Multi-zone will create instances across multiple zones, which can ensure high availability*

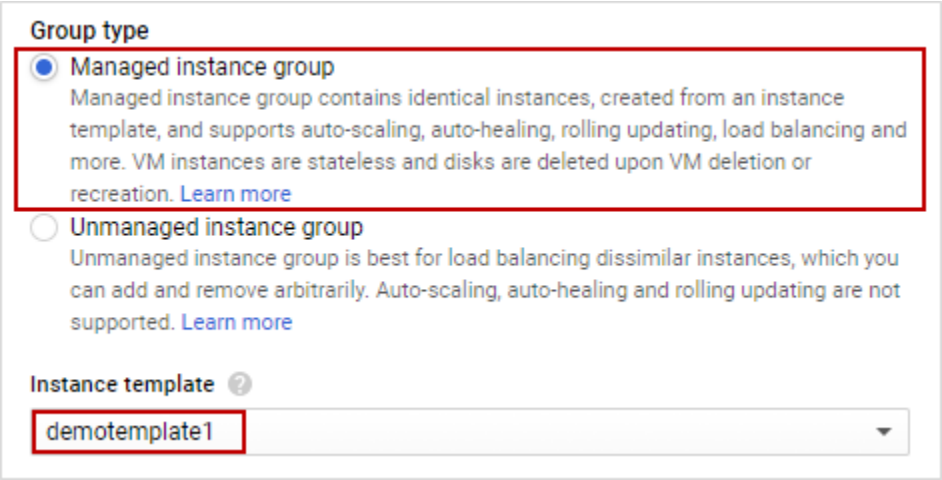
Internal



Step 3:

- *Select the Group type as Managed instance group*

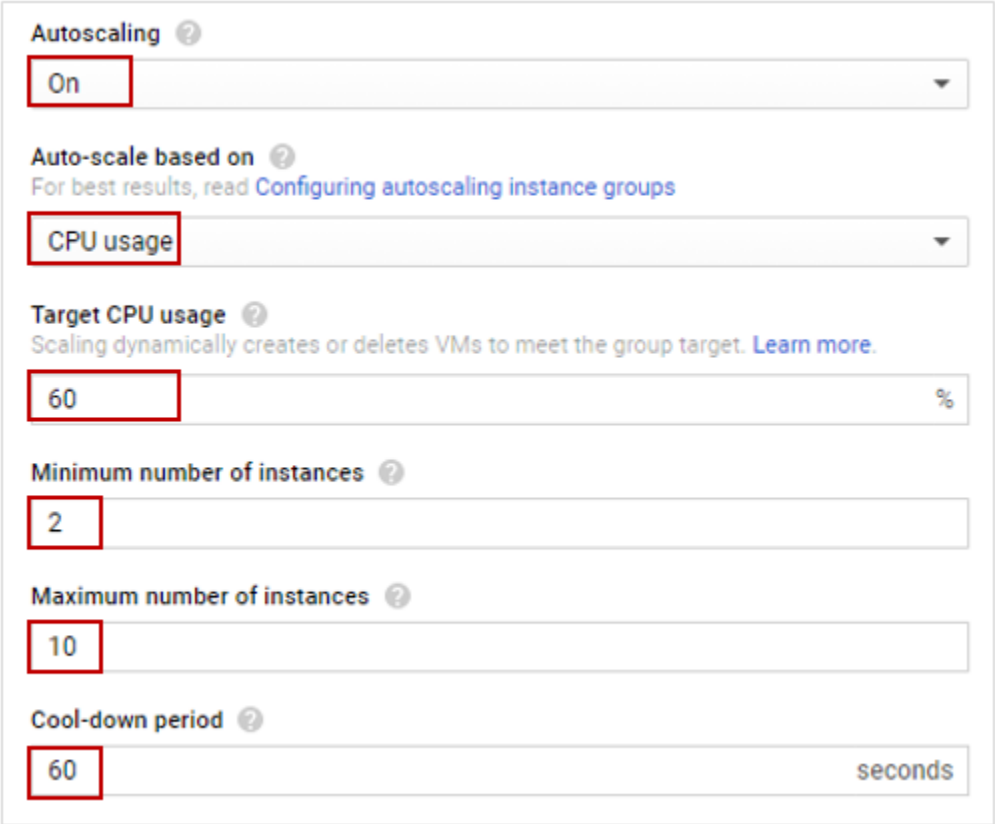
This is required as you are creating instance group based on an instance template. Select the Instance template that you had created in the previous demo.



Step 4:

- *Next step is to enable automatic scaling using the below shown fields*

Internal



You can enable scaling by selecting On for the Autoscaling field. The scaling action will happen based on the scaling condition defined in the Auto-scale based on field.

Autoscaling can be configured based on below also:

- *CPU usage*

- *HTTP Load balancing usage*
- *Stackdriver monitoring metrics*
- *Multiple metrics*

Step 5:

The minimum and maximum number of instances in the group have to be mentioned. Based on the CPU usage, Autoscaling will invoke or terminate instances in the group, within the given min-max limits, automatically.

- *Under Autohealing section, select Create a health check for Health check field. This will open a new screen where you can provide the required values and save it*

Internal

Autohealing

VMs in the group are recreated as needed. You can use a health check to recreate a VM if the health check finds the VM unresponsive. If you do not select a health check, VMs are recreated only when stopped. Learn more

Health check

Create a health check

No health check

Initial delay

300 seconds

Advanced creation options

Create Cancel

HSBC

Infosys
Navigate your next

Name

demohealthcheck1

Description (Optional)

demohealthcheck1

Protocol

TCP

Port

80

Proxy protocol

NONE

Request (Optional)

Response (Optional)

Health criteria

Define how health is determined: how often to check, how long to wait for a response and how many successful or failed attempts are decisive

Check interval

10 seconds

Timeout

5 seconds

Healthy threshold

2 consecutive successes

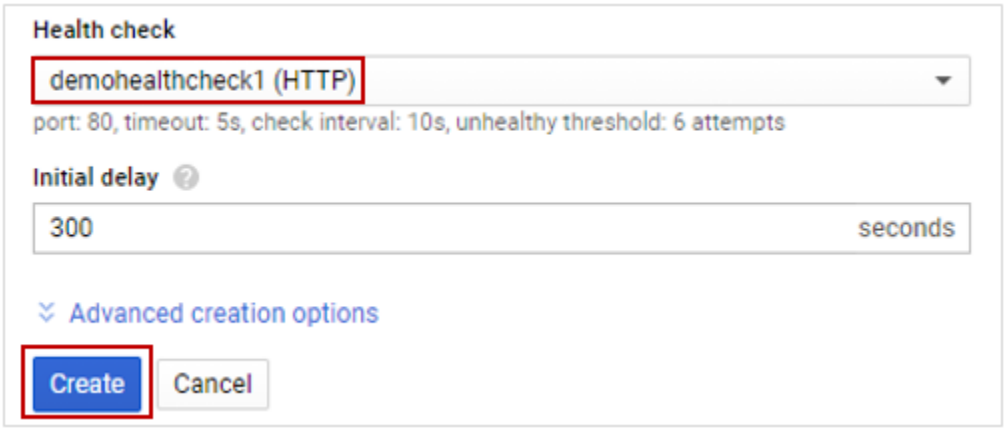
Unhealthy threshold

6 consecutive failures

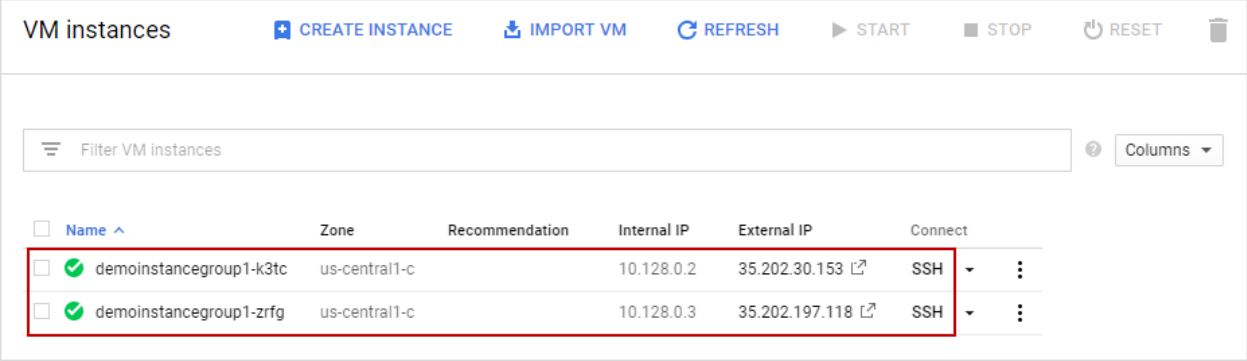
Save and continue

Cancel

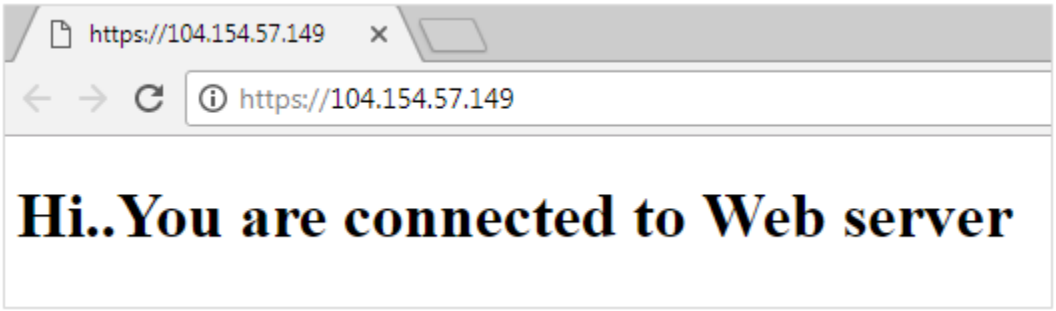
Select the newly created health check and click on Create button. This creates the instance group.



You will find that Autoscaling automatically spins up two instances based on the minimum number that you have provided.



Both the instances are created based on a common instance template and will have the shell script executed during startup. You can access the webpages using the respective External IP addresses.



Assignment 6a. Download and install the APP Engine SDK for Java

Objective: *Download and install the APP Engine SDK for deploying sample Java application*

Problem Description: *Before hosting your application or service you need to download and install the APP Engine SDK. For deploying and management of applications it includes a local development server and relevant tools. Google App Engine offers pool of languages to choose. Here will be seeing about JAVA for Windows OS.*

Prerequisites: *Please ensure you have installed:-*

Java version 7 or 8

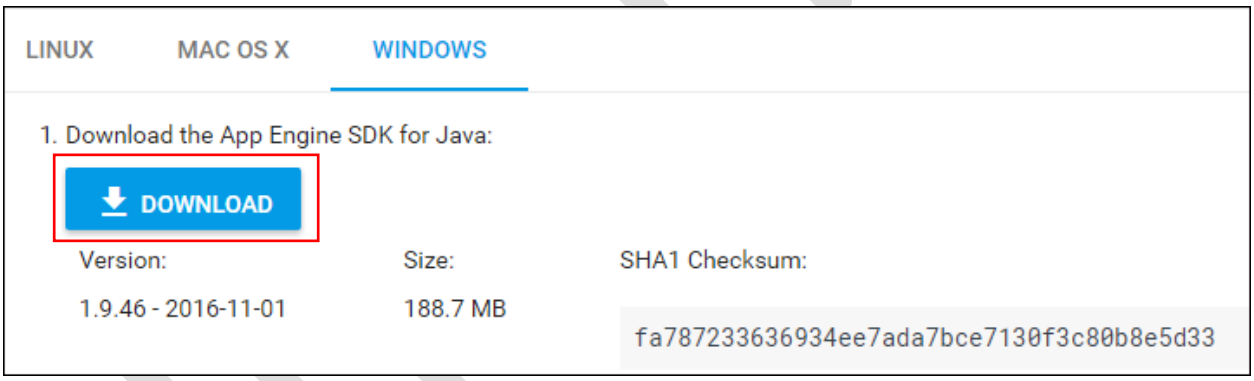
Estimated time: 10 minutes

Solution:

1) *Go to Google cloud platform link:*

https://cloud.google.com/appengine/docs/java/download

2) *Click on the Download as shown below*



3) *In Windows, to install the SDK follow the below steps*

a. *Double-click the **appengine-java-sdk-1.9.46**.zip file that you have downloaded to extract the SDK to a directory of your choice.*

b. *The App Engine Java SDK requires Java 7 bytecode level. You can use either Java 7 or Java 8.*

Summary of this assignment: *In this assignment, you have downloaded and installed the Google App Engine SDK for java.*

Assignment 6b. Run the Demo application from your local server through Command Prompt

Objective: *Run the Demo Java application from your local development server*

Problem Description: *Before hosting your application or service you need to download and install the APP Engine SDK which includes a local development server as well as the tooling for deploying and managing your applications in App Engine.*

Estimated time: 20 minutes

Solution:

Internal

1) *Open the command prompt*

2) *Change the working directory to the place where SDK is unzipped using cd command*

3) *There will be few sample applications that will be available in* ***\appengine-java-sdk-1.9.46 \demos*** *folder*

```
D:\appengine-java-sdk-1.9.46\appengine-java-sdk-1.9.46>dir
 Volume in drive D is DATA
 Volume Serial Number is 1C25-3738

 Directory of D:\appengine-java-sdk-1.9.46\appengine-java-sdk-1.9.46

11/24/2016  01:29 PM    <DIR>          .
11/24/2016  01:29 PM    <DIR>          ..
11/24/2016  01:29 PM             1,403 ABOUT
11/24/2016  01:29 PM    <DIR>          bin
11/24/2016  01:29 PM    <DIR>          config
11/24/2016  01:29 PM             5,938 COPYING
11/24/2016  01:29 PM    <DIR>          demos
11/24/2016  01:29 PM    <DIR>          docs
10/18/2016  06:41 PM    <DIR>          jetty93
11/24/2016  01:29 PM    <DIR>          jetty93-base
11/24/2016  01:29 PM    <DIR>          lib
11/24/2016  01:29 PM             3,896 README
11/24/2016  01:29 PM               223 README.ORM
11/24/2016  01:29 PM               136 RELEASE_NOTES
11/24/2016  01:29 PM            27,574 RELEASE_NOTES.ORM
11/24/2016  01:29 PM    <DIR>          src
11/24/2016  01:29 PM                62 VERSION
               7 File(s)         39,232 bytes
              10 Dir(s)  386,860,101,632 bytes free

D:\appengine-java-sdk-1.9.46\appengine-java-sdk-1.9.46>
```

4) *Demo application can be run using the command*

bin\dev_appserver.cmd demos\<app-name>\war

```
C:\windows\system32\cmd.exe - bin\dev_appserver.cmd  demos\guestbook\war

D:\appengine-java-sdk-1.9.46\appengine-java-sdk-1.9.46>bin\dev_appserver.cmd dem
os\guestbook\war
Executing [C:\Program Files\Java\jre1.8.0_77\bin\java, -javaagent:D:\appengine-j
ava-sdk-1.9.46\appengine-java-sdk-1.9.46\lib\agent\appengine-agent.jar, -Xbootcl
asspath/p:D:\appengine-java-sdk-1.9.46\appengine-java-sdk-1.9.46\lib\override\ap
pengine-dev-jdk-overrides.jar, -classpath, D:\appengine-java-sdk-1.9.46\appengin
e-java-sdk-1.9.46\bin\..\lib\appengine-tools-api.jar, com.google.appengine.tools
.development.DevAppServerMain, --property=kickstart.user.dir=D:\appengine-java-s
dk-1.9.46\appengine-java-sdk-1.9.46, D:\appengine-java-sdk-1.9.46\appengine-java
-sdk-1.9.46\demos\guestbook\war]
Nov 24, 2016 1:34:22 PM java.util.prefs.WindowsPreferences <init>
WARNING: Could not open/create prefs root node Software\JavaSoft\Prefs at root 0
x80000002. Windows RegCreateKeyEx(...) returned error code 5.
Nov 24, 2016 1:34:24 PM com.google.appengine.tools.development.SystemPropertiesM
anager setSystemProperties
INFO: Overwriting system property key 'java.util.logging.config.file', value 'D:
\appengine-java-sdk-1.9.46\appengine-java-sdk-1.9.46\config\sdk\logging.properti
es' with value 'WEB-INF/logging.properties' from 'D:\appengine-java-sdk-1.9.46\a
ppengine-java-sdk-1.9.46\demos\guestbook\war\WEB-INF\appengine-web.xml'
Nov 24, 2016 1:34:24 PM com.google.apphosting.utils.jetty.JettyLogger info
INFO: Logging to JettyLogger(null) via com.google.apphosting.utils.jetty.JettyLo
gger
Nov 24, 2016 8:04:25 AM com.google.apphosting.utils.jetty.JettyLogger info
INFO: jetty-6.1.x
Nov 24, 2016 8:04:26 AM com.google.apphosting.utils.jetty.JettyLogger info
INFO: Started SelectChannelConnector@localhost:8080
Nov 24, 2016 8:04:26 AM com.google.appengine.tools.development.AbstractModule st
artup
INFO: Module instance default is running at http://localhost:8080/
Nov 24, 2016 8:04:26 AM com.google.appengine.tools.development.AbstractModule st
artup
INFO: The admin console is running at http://localhost:8080/_ah/admin
Nov 24, 2016 1:34:26 PM com.google.appengine.tools.development.DevAppServerImpl
doStart
INFO: Dev App Server is now running
```

5) *The application is running as above.*

6) *It will be running in port 8080, accessed through the URL localhost:8080*

7) *You can post Greeting if required.*

8) *Server keeps running in command prompt as shown.*

```
Nov 24, 2016 1:38:01 PM com.google.appengine.api.datastore.dev.LocalDatastoreSer
vice init
INFO: Local Datastore initialized:
        Type: High Replication
        Storage: D:\appengine-java-sdk-1.9.46\appengine-java-sdk-1.9.46\demos\gu
estbook\war\WEB-INF\appengine-generated\local_db.bin
Nov 24, 2016 1:38:01 PM com.google.appengine.api.datastore.dev.LocalDatastoreSer
vice load
INFO: The backing store, D:\appengine-java-sdk-1.9.46\appengine-java-sdk-1.9.46\
demos\guestbook\war\WEB-INF\appengine-generated\local_db.bin, does not exist. It
 will be created.
Nov 24, 2016 1:38:01 PM com.google.appengine.tools.development.LocalResourceFile
Servlet doGet
```

9) *The application server can be stopped by pressing **ctrl+c** in the command prompt*

> **Note:** The application won't be available if the plug-in is not installed
>
> The plug-in can be downloaded by clicking on the download plug-in button appearing in the browser

Summary of this assignment: *In this assignment, you have run the demo application in your local development server.*

Internal

Assignment 7a. Installing the Google Plugin for Eclipse to create Web Application

Objective: *creating and deploying your first web application using Google Plugin for eclipse and Google SDK*

Problem Description: *Before deploying your application you need to install the Google Plugin for eclipse. The Google Plugin for Eclipse is the first integrated tool suite for the Google Cloud. Check what are all configurations would be required to perform these steps.*

Prerequisites:

Java 7

Eclipse 4.X

Estimated time: 15 minutes

Solution:

Steps involved:

- *Click this link https://developers.google.com/eclipse/docs/getting_started*
- *It will take you to the installation page.*

Internal

- *You can install the **Google Plugin for Eclipse** using the software update feature of Eclipse. Be sure to use the plugin that corresponds to your version of Eclipse*

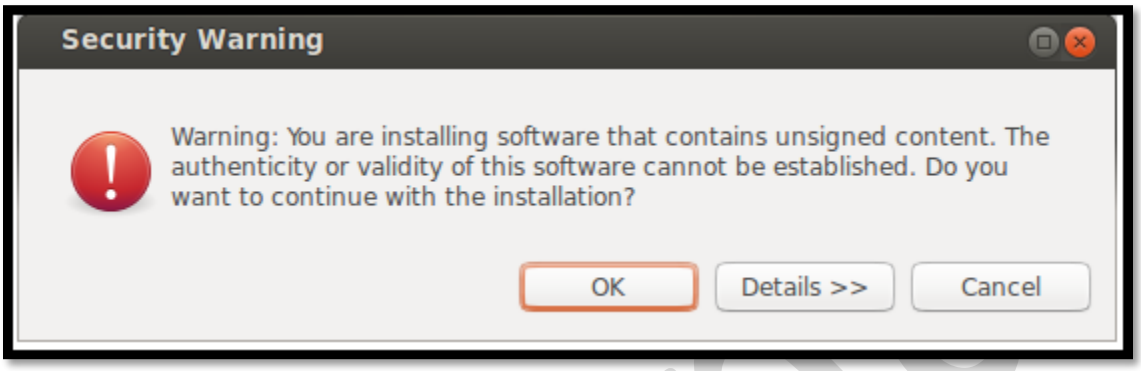
Eclipse version	Plugin link	Archive link
Eclipse 4.3 (Kepler)	`https://dl.google.com/eclipse/plugin/archive/3.8.0/4.3`	Site archive for Eclipse 4.3

- *Open Eclipse, make sure it is running JVM version of 1.7.0 or later*
- *Select **Help > Install New Software**... a dialog box appears, provide the update site URL into the Work with text box:*
- *https://dl.google.com/eclipse/plugin/archive/3.8.0/4.3*
- *And hit the **enter** button*
- *Google Plugin for Eclipse is the required component. Select the checkbox next to **Google Plugin for Eclipse 4.4/4.5/4.6**.*

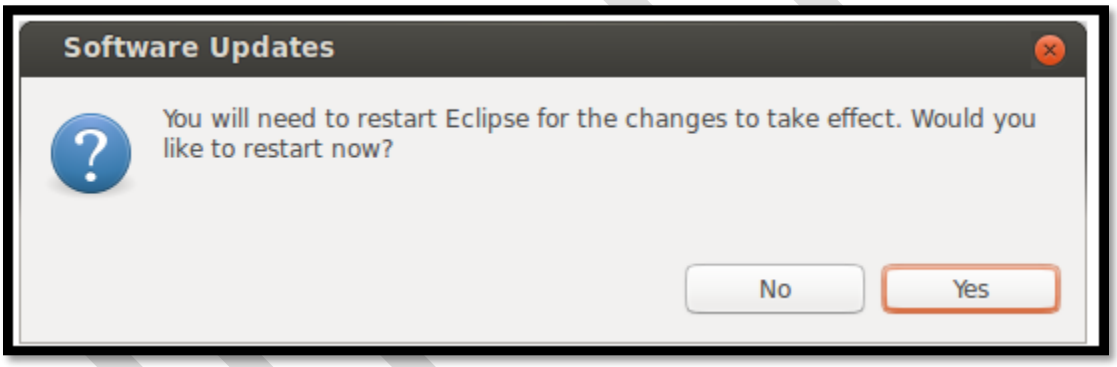
*Click **Next**.*

- *Verify the features that you are about to install. Click **Next**.*
- *Make sure you have accepted the license agreement after reading the terms and conditions.*
- *Click **Finish**.*
- *You will get the **Security Warning** dialog box, just click on **OK** and proceed.*

Internal



- *It will ask you to restart the Eclipse, kindly proceed with Click Restart Now.*



- *The latest eclipse will be having this* *google icon in the toolbar row.*

Summary of this Assignment: *In this assignment, you have learned how to install the Google plugin for Eclipse*

Assignment 7b. Creating and Running the Web Application locally

Objective: *creating and running your web application using Google Plugin for eclipse and Google SDK*

Problem Description: *Before deploying your application in App Engine, you need to create and run the application locally. To perform this, will create a new web application in eclipse and run and test it locally.*

Prerequisites:

Java 7

Eclipse 4.X

Google Plugin for Eclipse

Estimated time: 15 minutes

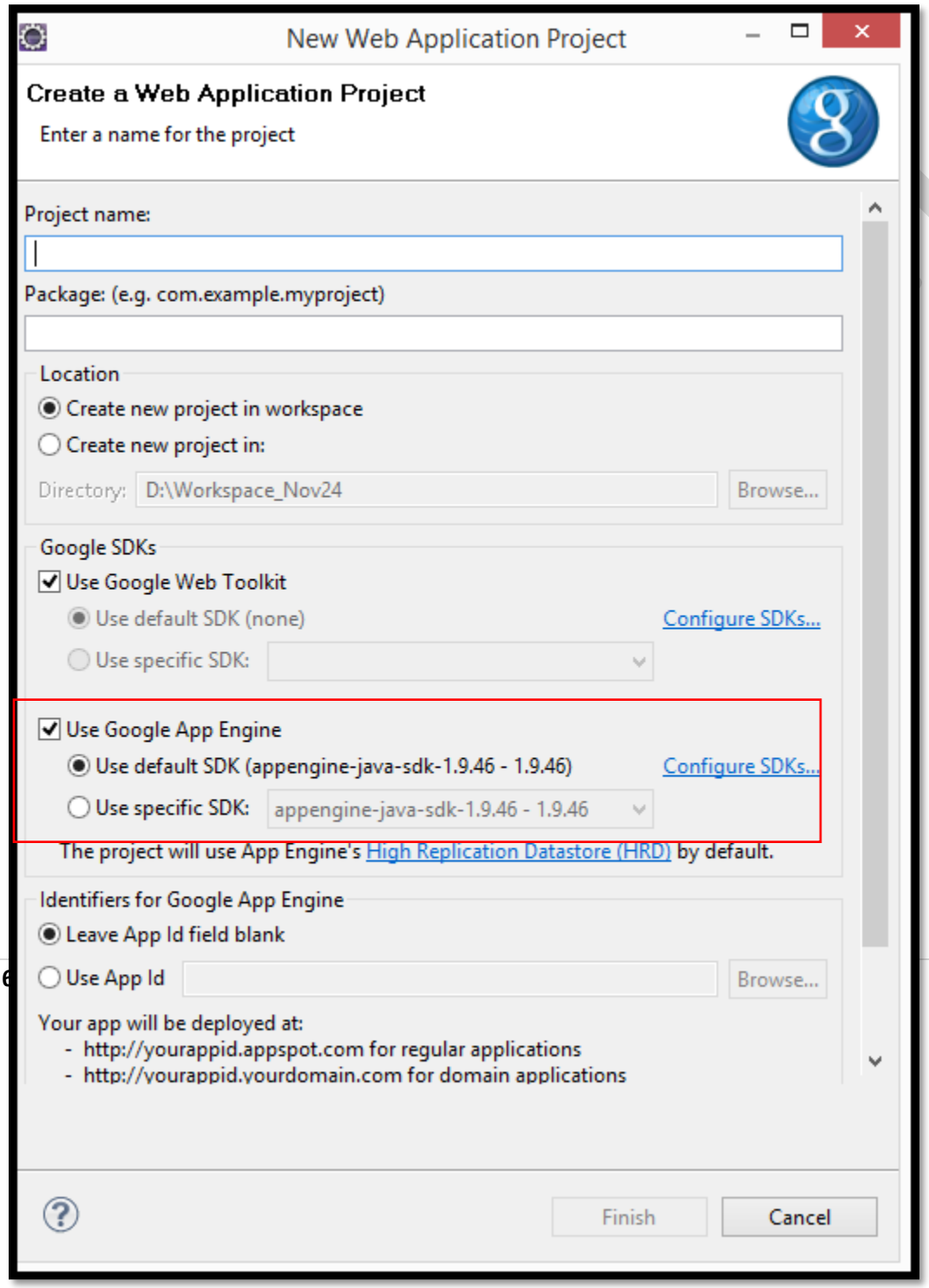
Solution:

Internal

1) ***Create a New Web Application***

   a. *To create a new project, click on the* ***New Web Application Project*** *toolbar button*

Internal


New Web Application Project
Create a Web Application Project
Enter a name for the project
Project name:
Package: (e.g. com.example.myproject)
Location
Create new project in workspace
Create new project in:
Directory: D:\Workspace_Nov24
Browse...
Google SDKs
Use Google Web Toolkit
Use default SDK (none)
Configure SDKs...
Use specific SDK:
Use Google App Engine
Use default SDK (appengine-java-sdk-1.9.46 - 1.9.46)
Configure SDKs...
Use specific SDK: appengine-java-sdk-1.9.46 - 1.9.46
The project will use App Engine's High Replication Datastore (HRD) by default.
Identifiers for Google App Engine
Leave App Id field blank
Use App Id
Browse...
Your app will be deployed at:
- http://yourappid.appspot.com for regular applications
- http://yourappid.yourdomain.com for domain applications
Finish
Cancel

b. *Choose **Project Name** and **Package Name** of your choice*

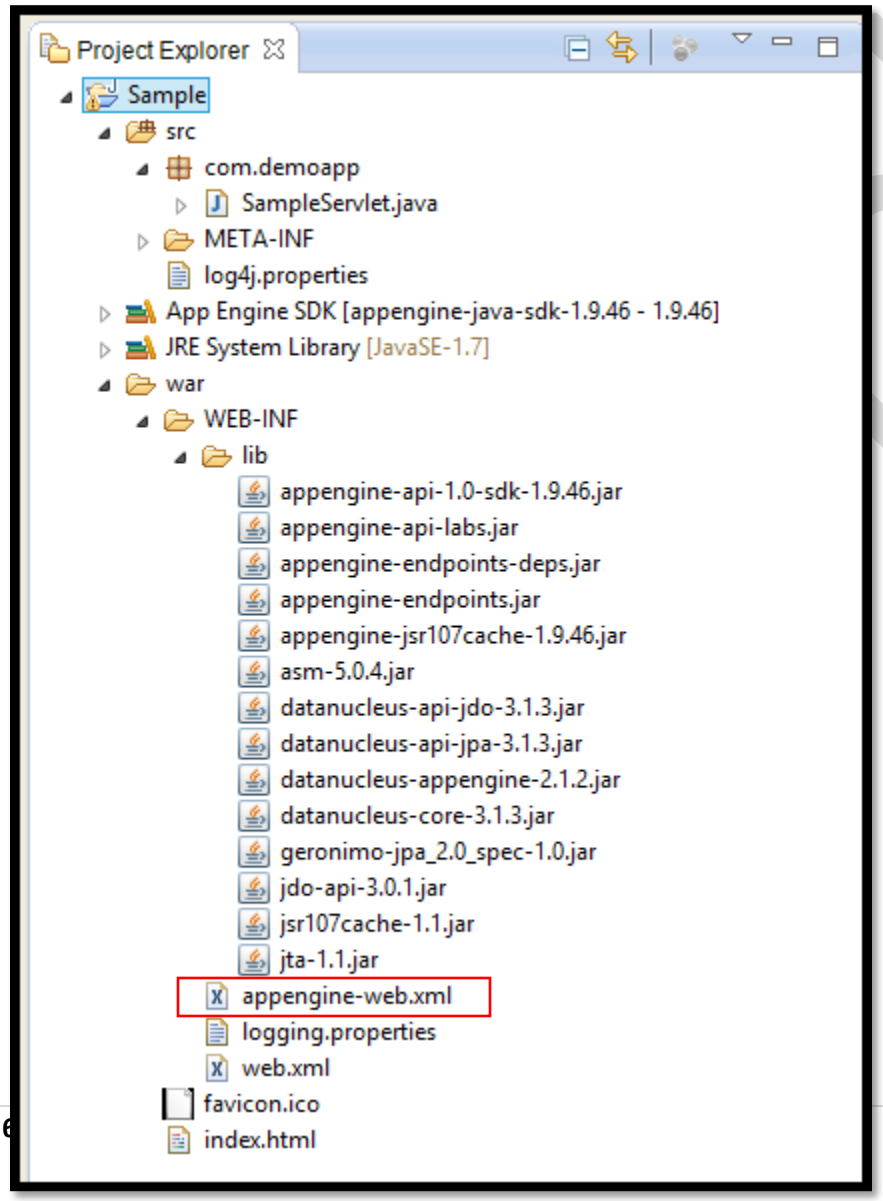
c. *De-select the Google SDK and select the Google App Engine*

d. *Using **Configure SDK** link configure the App Engine SDK which you have installed already*

e. *Click finished, automatically Google Plugin for Eclipse will produce a sample project*

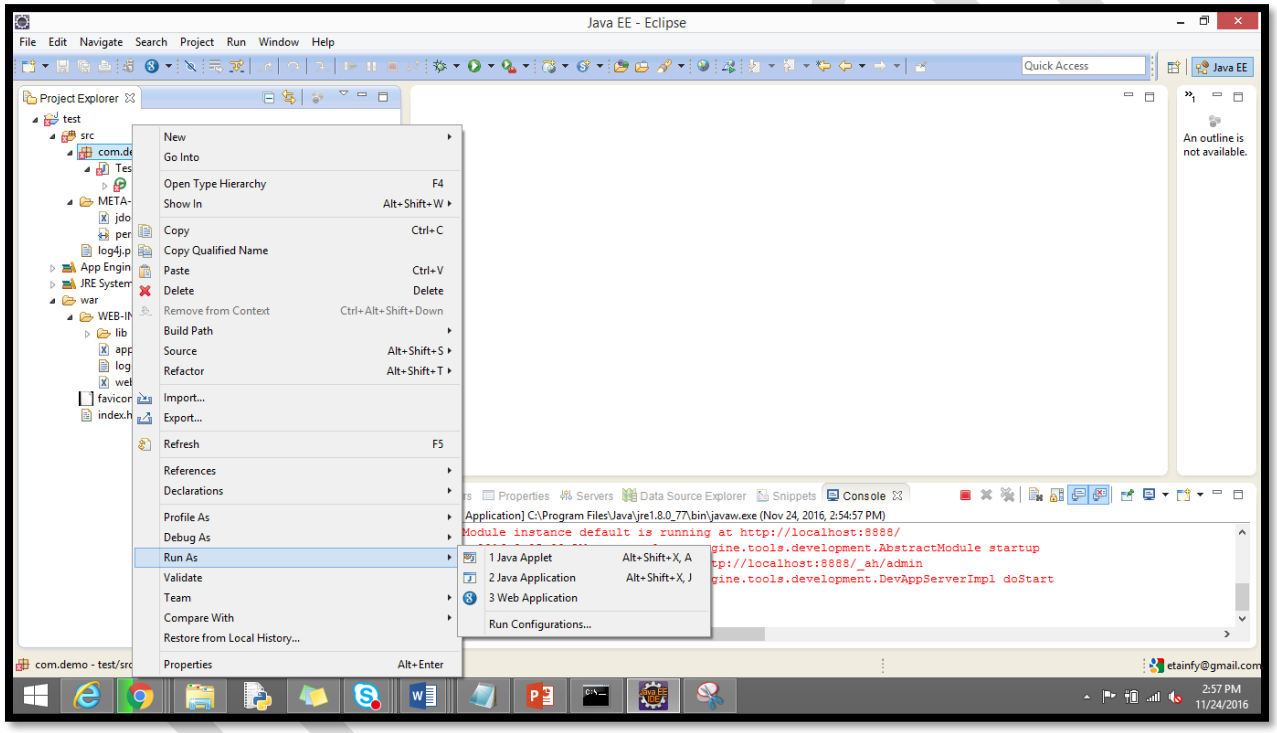
2) *Review the generated project directory*

Internal

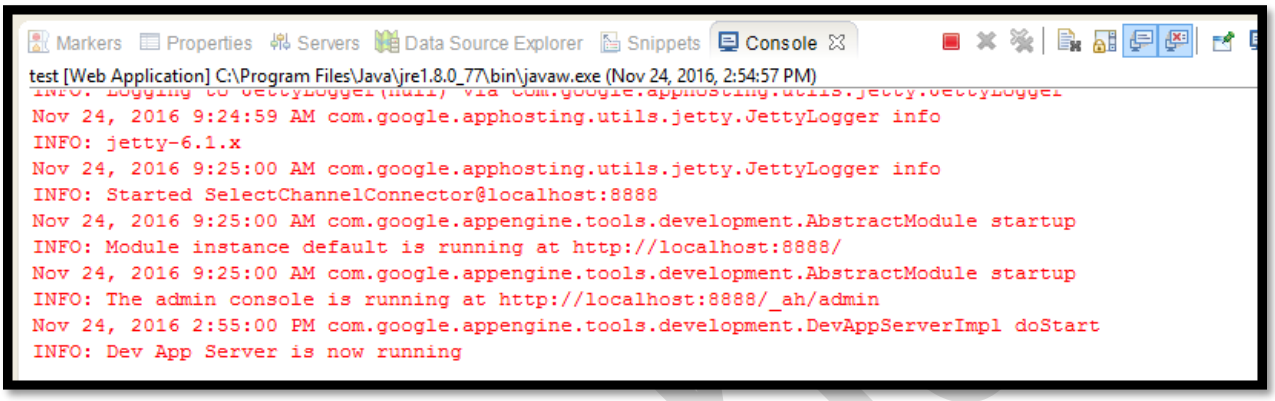

Project Explorer
Sample
src
com.demoapp
SampleServlet.java
META-INF
log4j.properties
App Engine SDK [appengine-java-sdk-1.9.46 - 1.9.46]
JRE System Library [JavaSE-1.7]
war
WEB-INF
lib
appengine-api-1.0-sdk-1.9.46.jar
appengine-api-labs.jar
appengine-endpoints-deps.jar
appengine-endpoints.jar
appengine-jsr107cache-1.9.46.jar
asm-5.0.4.jar
datanucleus-api-jdo-3.1.3.jar
datanucleus-api-jpa-3.1.3.jar
datanucleus-appengine-2.1.2.jar
datanucleus-core-3.1.3.jar
geronimo-jpa_2.0_spec-1.0.jar
jdo-api-3.0.1.jar
jsr107cache-1.1.jar
jta-1.1.jar
appengine-web.xml
logging.properties
web.xml
favicon.ico
index.html

1) *This file "**appengine-web.xml**", using this file only you can run and deploy the application.*

2) *On your project, do a right click and select run as "**Web Application** "as below.*



3) *Eclipse console:*



4) *Access URL **http://localhost:8888/,** see output*

Summary of the assignment: *In this assignment, you have learned to create and run the web application locally using Google Plugin for Eclipse.*

Assignment 7c: Deploying the App to Google App Engine

Objective: *To learn about deploying the Application to Google App Engine through Eclipse.*

Problem Description: *Before hosting an application, you need to deploy the locally tested application to Google App Engine using Eclipse. This demonstration assumes that you have already created account in Google Cloud Platform.*

Estimated time: 25 minutes

Solution:

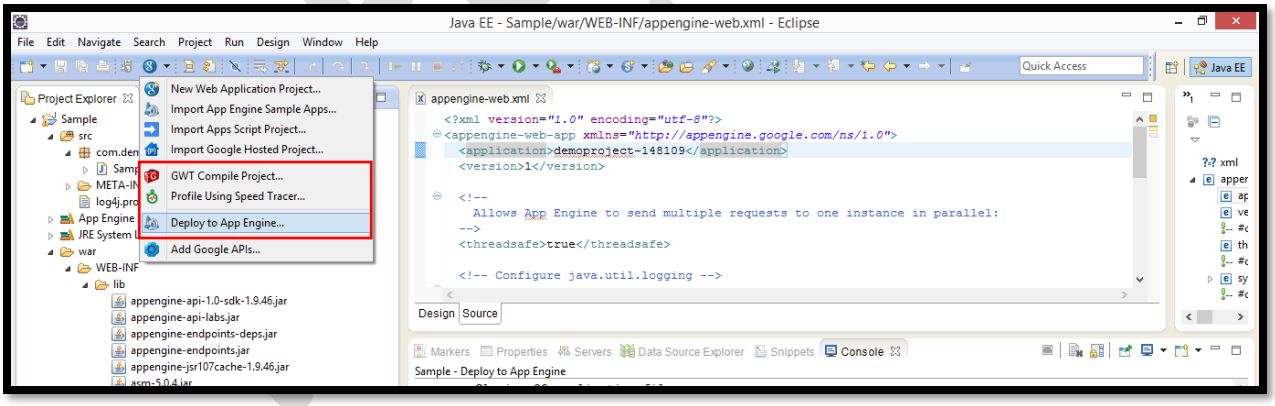
Steps involved:

1) *Create an **application ID** for your web application in google cloud platform console.*
2) *In this demonstration, an application ID, **"demoproject-148109"** is used and place it in appengine-web.xml.*

Internal

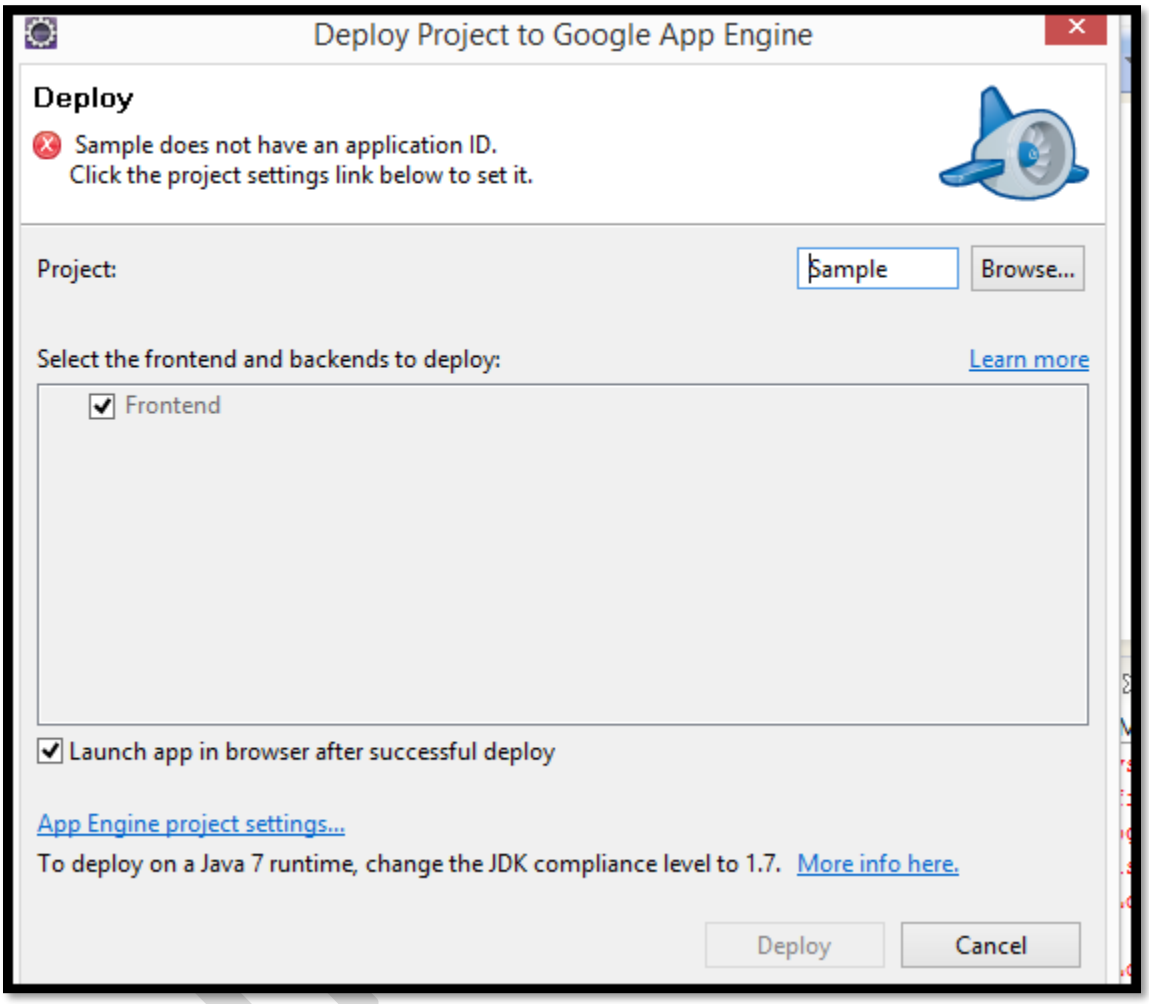
3) *Change the application ID in **appengine-web.xml** as above and save the file.*

4) *Now click on the* *icon and select **Deploy to App Engine** option as below*



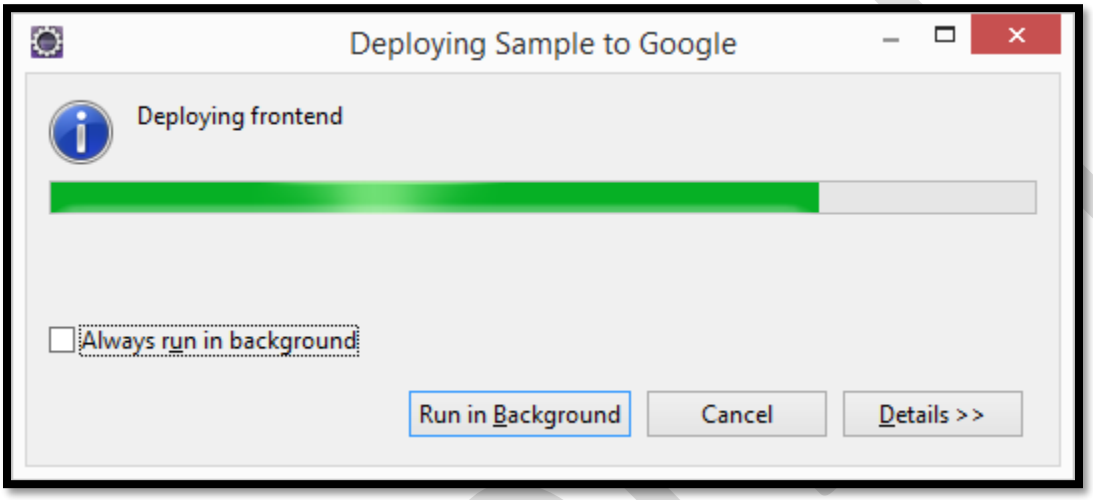
5) *Select the project which is ready for deploy and make sure xml file changed before deploying to App Engine.*

Internal

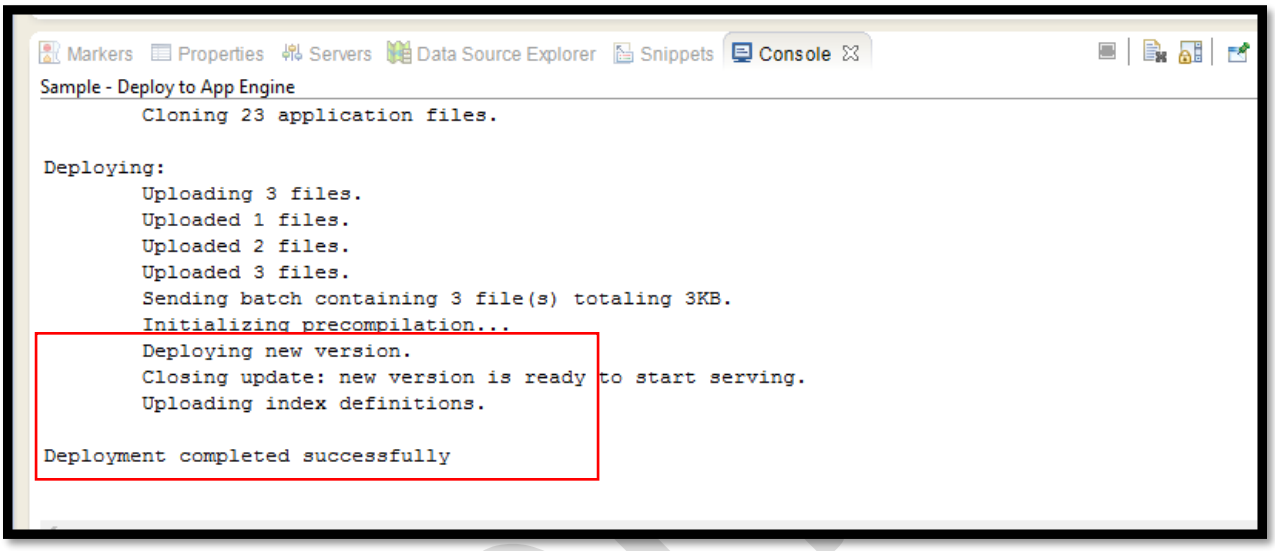


6) *Click on **Deploy** button.*

7) *Deploying started*



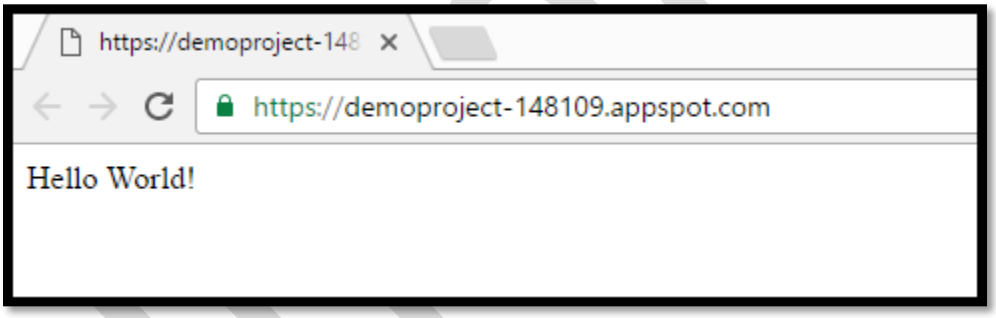
8) *The Eclipse Console screen would be updating the status as below.*



9) *After uploading successfully, you will be able to see the "Deployment Successful" message in the console.*

10) *Now browse directly take to your live application at* ***http://project-id.appspot.com****.*

And the Hello World Servlet:



Summary of this assignment: *In this assignment, you have deployed the local web application succeefully to App Engine using Eclispe.*

Assignment 8a: Creating a google cloud function

Highlights:

Create a google cloud function

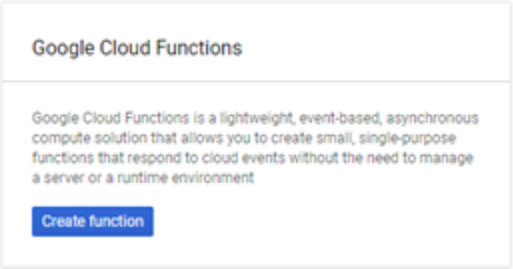
Test the function

Steps:

Login to https://console.cloud.google.com

Create a project

Navigate to "cloud functions"



4. *Click "Create function" and provide the below details.*

Name	function-1
Memory	256MB (Default)
Trigger	HTTP
Runtime	Node.js 8 (Choose your preferred runtime)
Function to execute	helloWorld

Cloud Functions | Create function

Name

function-1

Memory allocated

256 MB

Trigger

HTTP

URL

https://us-central1-osce-159707.cloudfunctions.net/function-1

Source code

- (●) Inline editor
- ( ) ZIP upload
- ( ) ZIP from Cloud Storage
- ( ) Cloud Source repository

Runtime

Node.js 8

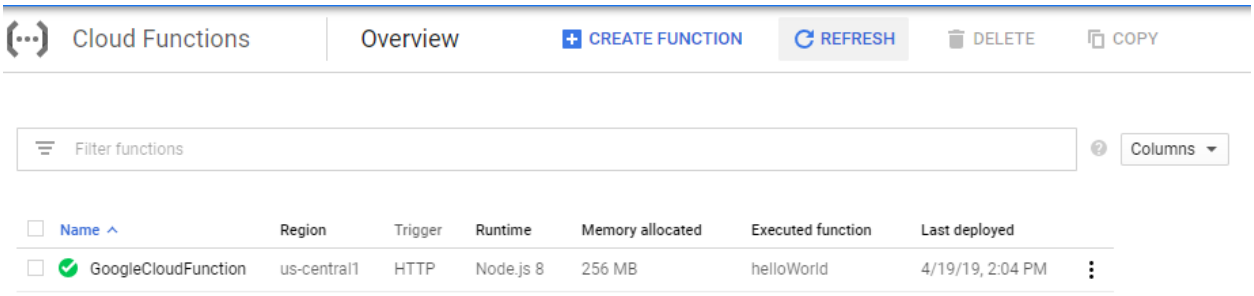
index.js package.json

```
/**
 * Responds to any HTTP request.
 *
 * @param {!express:Request} req HTTP request context.
 * @param {!express:Response} res HTTP response context.
 */
exports.helloWorld = (req, res) => {
  let message = req.query.message || req.body.message || 'Hello Wo
  res.status(200).send(message);
};
```

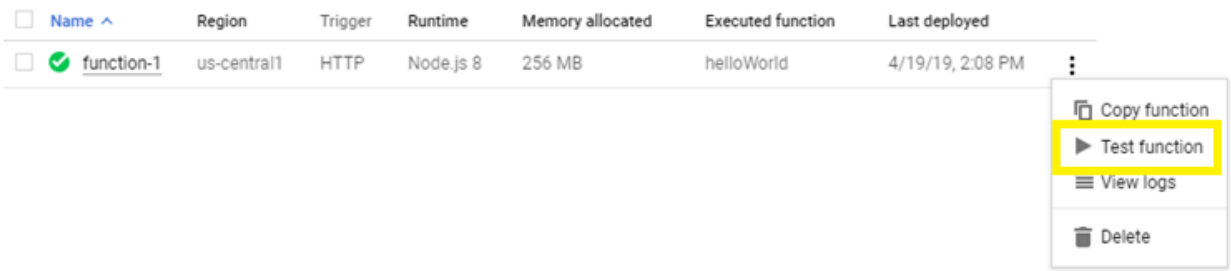
Function to execute

helloWorld

5. *Function is created and available for execution.*



6. *Now, test the function by supplying a sample message to be displayed.*



Triggering event

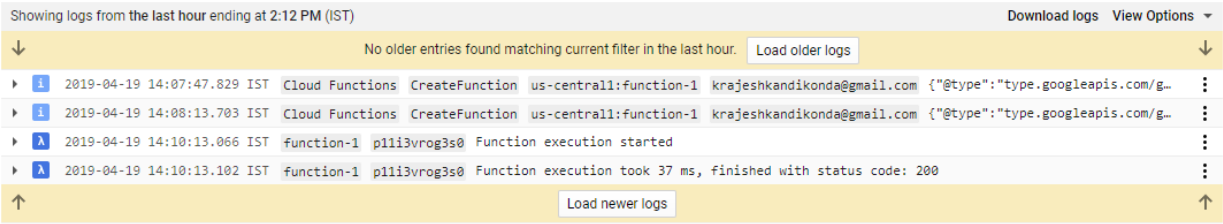
```
1 {"message":"Hello World"}
2
```

Test the function

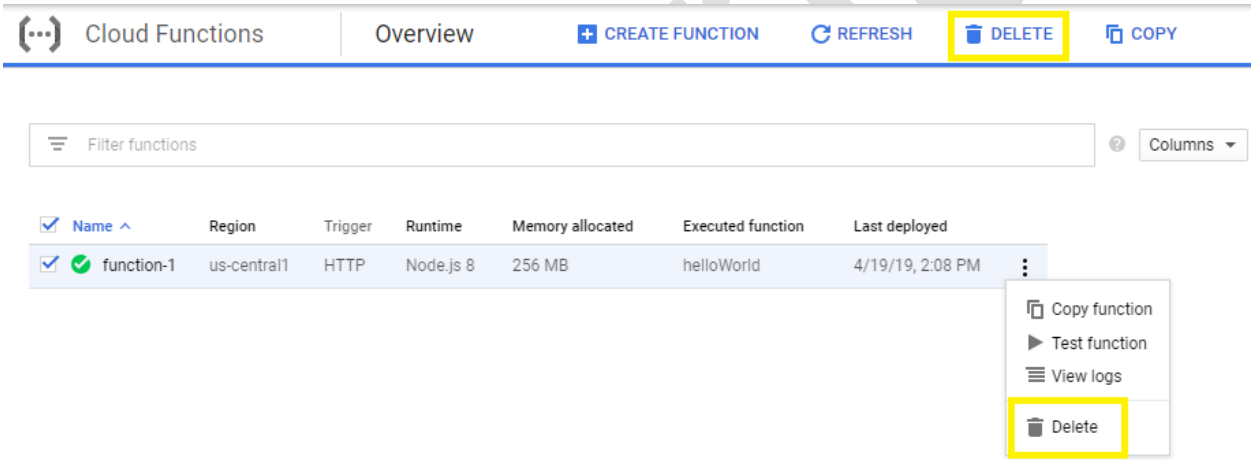
Output

```
Hello World
```

7. *Check the logs for more information*



8. *Delete the cloud function.*



Summary:

You have learnt to create and test a google cloud function

Assignment 8b: Triggering a google cloud function

Highlights:

1. *Create a cloud function that receives HTTP request and returns response.*
2. *Deploy the function*
3. *Trigger by sending HTTP request and observe the response.*

Demo Steps:

1. *Login to Google cloud platform and start google cloud shell.*
2. *Navigate to python-docs-samples/functions*
3. *You can use sample functions provided here or Create a cloud function using python 3.7 run time, with the below sample code.*

```
import sys
from flask import escape
def test_fun(request):
    return "Cloud function triggered. This is response to the HTTP request received"
```

4. Deploy the function in google cloud shell, using the below command.

gcloud functions deploy test_fun --runtime python37 --trigger-http

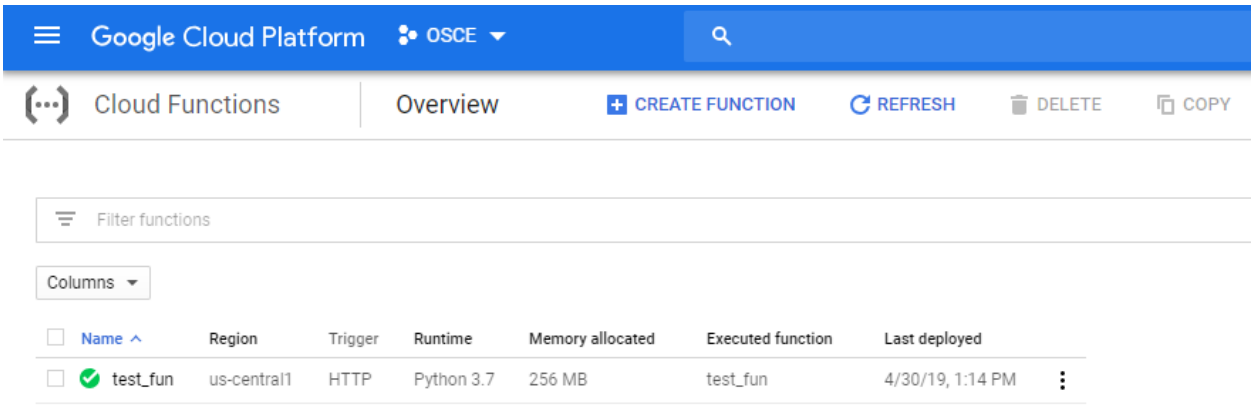
Internal

```
httpsTrigger:
  url: https://us-central1-osce-159707.cloudfunctions.net/test_fun
labels:
  deployment-tool: cli-gcloud
name: projects/osce-159707/locations/us-central1/functions/test_fun
runtime: python37
serviceAccountEmail: osce-159707@appspot.gserviceaccount.com
sourceUploadUrl: https://storage.googleapis.com/gcf-upload-us-central1-7cd718a6-9e2e-4c9b-a36a-a8d84c79cecc/77e6
38c5-3bad-4059-b447-cb3a20f2f58e.zip?GoogleAccessId=service-481049991222@gcf-admin-robot.iam.gserviceaccount.com
&Expires=1556612016&Signature=gAaXhl81PMPiKHT9AWhgEvoXLhHUHriRlNba6EYUiV%2Fx2HNArBYGHDZFR%2FuvkoRwgrT4H8dfqBRyKN
WIDH0sGDAAKKTkDc3t6JWNVNjkN0gOElBOLavrbYpN3Yu8KGRWvsl7ntIRIOVY7p3yP7xRQRsvzps85Iy7vcUA6B6L9TVfyxobanQYotHi1CwcQl
sk%2FIhz%2Fsz1zyaQUev%2FjuXATTbcR5vUtAnHYxKyXqdnu%2Ff0xhDiiFkreMalvfCjLzwqmE2Ec0leb7KKSuwtpCNHSkRJSO8JBfAQYh9RBq
pPUCrVSCsOFN%2BnUZum2DVVheBYWXul%2BhmLdt%2FscPFwo1O7dg%3D%3D
status: ACTIVE
timeout: 60s
updateTime: '2019-04-30T07:44:52Z'
versionId: '1'
```

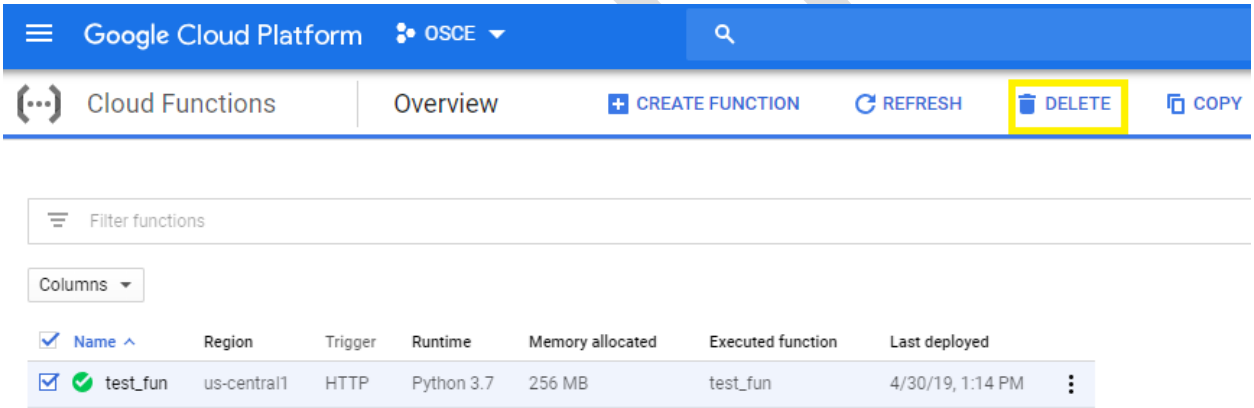
4. *Test the function by sending HTTP request*
   curl -X POST https://<region>-<project ID>.cloudfunctions.net/test_fun

```
[redacted] $ curl -X POST "https://us-
central1-osce-159707.cloudfunctions.net/test_fun"
Cloud function triggered. This is a response to the HTTP Request Received!krajeshkandikonda@cloudshell:~/python-
docs-samples/functions/[redacted] $
```

It is observed that the function is triggered in response to HTTP request.
6. Navigate to the cloud functions menu in the console. Observe that the cloud function is deployed.



6. *Delete the function.*



Summary:

You have learnt to create a cloud function to be triggered by HTTP requests. Fired HTTP request and triggered the cloud function. Finally deleted the cloud function deployment

Assignment 9: Google Container Registry

Objective: *To set up an environment to use Google Container Registry and pull images from container registry*

Problem Description:

In this Assignment you will set up an environment, push images to container registry and pull from container registry.

Solution:

1. ***Install Docker:***
    a. *For Linux-based system, add your username to the **docker** group by using following command :*

$sudo usermod –a –G docker ${user}

2. ***Pushing to Registry***
    a. *Choose a private registry name and append you GCP project id to the **gcr.io** hostanames by using following command :*

Internal

$gcr.io/your-project-id/

b. *Add the tag you image by following command:*

$ docker tag user/image gcr.io/your-project-id/

c. *Use gcloud to push image ot the GCE registry:*

$gcloud docker –push gcr.io/your-project-id/image

3. Pulling from Registry

Images that have been pushed to Container registry can be pulled with gcloud docker pull command :

$gcloud docker –pull gcr.io/your-project-id/image

Summary of Assignment: *Developer will be able to successfully set up an environmanet for container registry and push/pull images from registry to be used with GCP resources.*

Assignment 10: Google Container Engine-Deploying App

Objective: *To explore Container concept and deploying application in a hosted Kubernet Environment.*

Background:

GKE: *Google Container Engine, a Google hosted version of Kubernetes.*

Internal

Container Cluster: *Group of compute engine instances running Kubernetes.*

Kubernetes: *It is an open source project that can run on many different environment, which manages the container cluster. It is originally designed by Google.*

Using GKE we can focus on Kubernetes rather being concerned of underlying infra.

Kubelet: *A primary node agent that runs on each node.*

Problem Description:

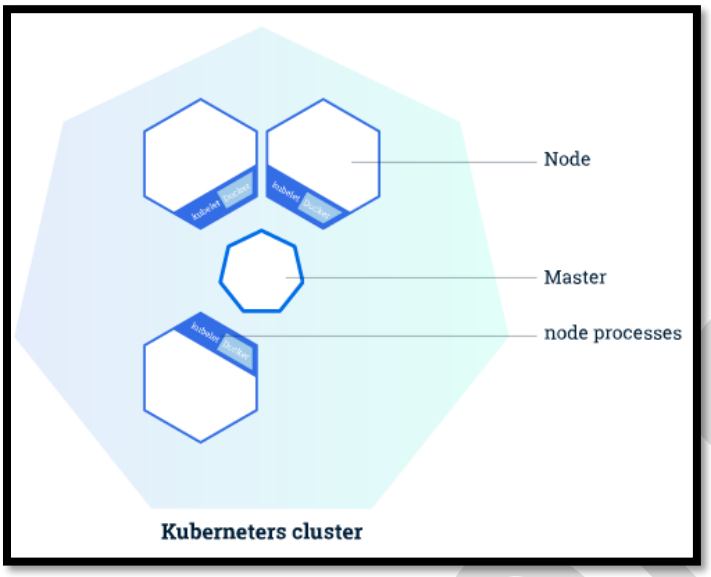
In this Assignment you will be exploring Kubernetes environment, create a cluster and deploy an app in cluster.

Solution:

1. ***Create a Kubernetes cluster****:*

A cluster contains a Node, Master and node process as shown below:

Internal

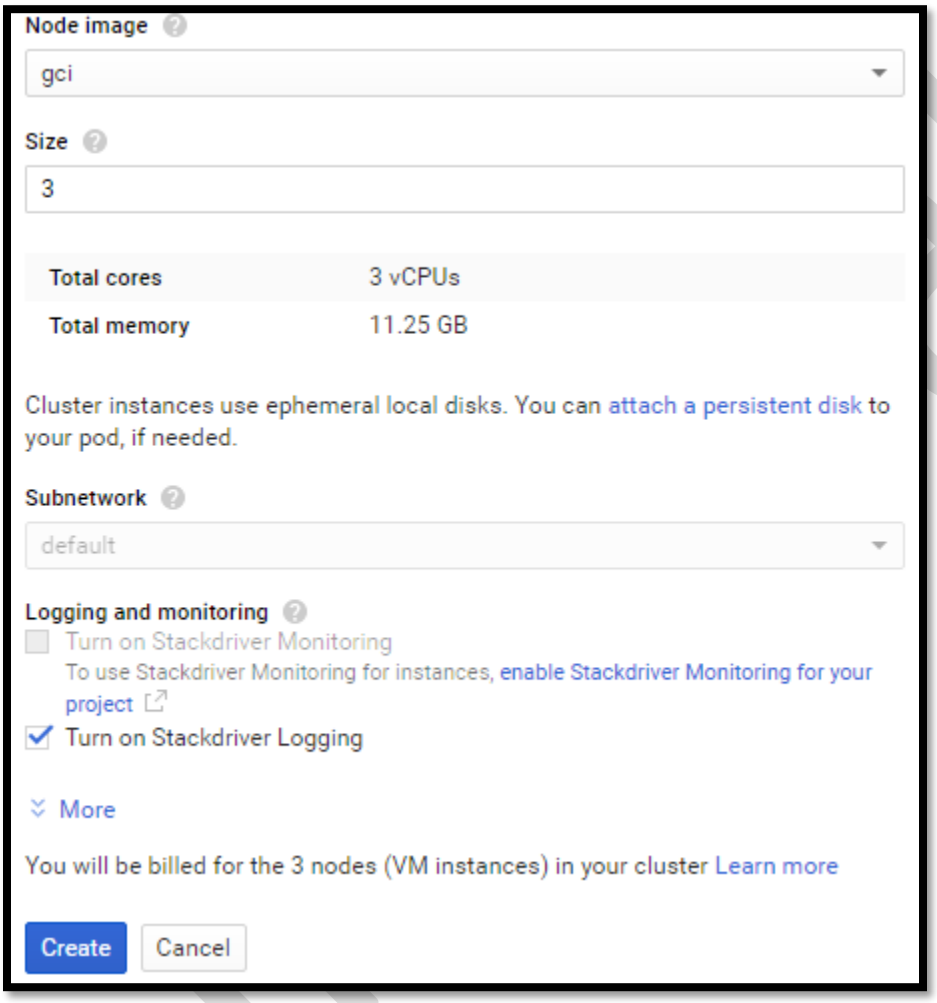


Master is responsible to manage the cluster. Each node has a Kubelet, which is an agent for managing the node and communicating with cabernet master.

Kubernetes cluster can be deployed either in ***Physical or virtual machine****. You can use* ***minikube*** *tool to run the Kubernetes locally which creates a VM in your local machine.*

We will be creating cluster in our container engine cluster created in GCP, follow below steps:

- ➔ *Navigate to container engine and click "create cluster". Give a name to cluster and specify the size as 3 to create three instances under a container:*


Node image
gci
Size
3
Total cores
3 vCPUs
Total memory
11.25 GB
Cluster instances use ephemeral local disks. You can attach a persistent disk to your pod, if needed.
Subnetwork
default
Logging and monitoring
Turn on Stackdriver Monitoring
To use Stackdriver Monitoring for instances, enable Stackdriver Monitoring for your project
Turn on Stackdriver Logging
More
You will be billed for the 3 nodes (VM instances) in your cluster Learn more
Create
Cancel

➔ *As now we have created cluster, type following command to get connected to cluster and to get the cluster details :*

$kubectl get nodes

```
mamta-146808 ×  +
-bash: cloud: command not found
etainfy@mamta-146808:~$
etainfy@mamta-146808:~$ cloud container clusters get-credentials mucluster     --zone us-central1-b --project mamta-146808
-bash: cloud: command not found
etainfy@mamta-146808:~$ gcloud container clusters get-credentials mucluster \
>     --zone us-central1-b --project mamta-146808
Fetching cluster endpoint and auth data.
kubeconfig entry generated for mucluster.
etainfy@mamta-146808:~$ kubectl get nodes
NAME                                          STATUS    AGE
gke-mucluster-default-pool-efda6a76-1tw3      Ready     9m
gke-mucluster-default-pool-efda6a76-dwk1      Ready     9m
gke-mucluster-default-pool-efda6a76-h82u      Ready     9m
etainfy@mamta-146808:~$
```

We have a running master and dashboard which allows you to view your application in UI.
$kubectl cluster-info

```
etainfy@mamta-146808:~$ kubectl cluster-info
Kubernetes master is running at https://104.198.157.204
GLBCDefaultBackend is running at https://104.198.157.204/api/v1/proxy/namespaces/kube-system/services/default-http-backend
Heapster is running at https://104.198.157.204/api/v1/proxy/namespaces/kube-system/services/heapster
KubeDNS is running at https://104.198.157.204/api/v1/proxy/namespaces/kube-system/services/kube-dns
kubernetes-dashboard is running at https://104.198.157.204/api/v1/proxy/namespaces/kube-system/services/kubernetes-dashboard
```

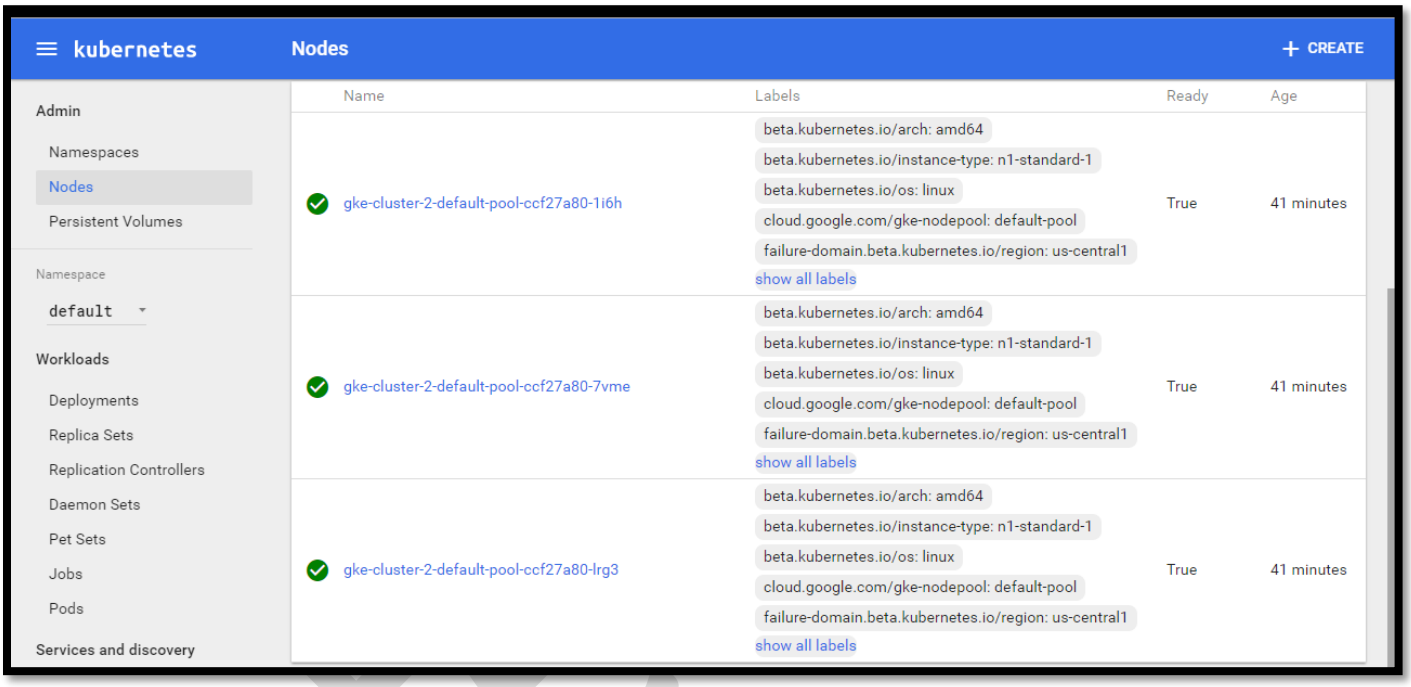
To describe the cluster type $kubectl gcloud container cluster describe <name of the cluster> --zone

You will get login credentials as:

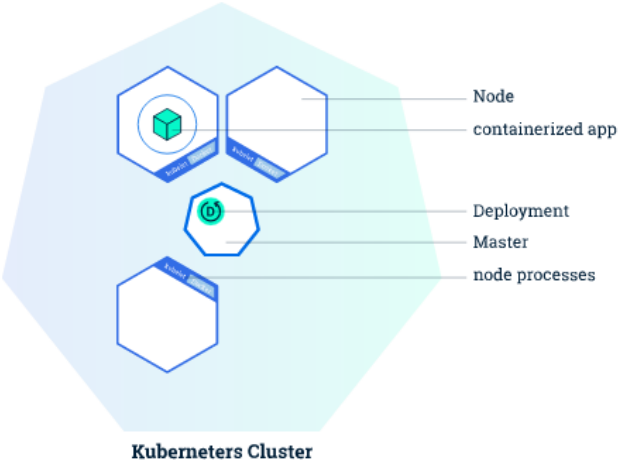
```
  password: XgBZ9SWW41GUb0tW
  username: admin
monitoringService: none
name: cluster-2
network: default
nodeConfig:
  diskSizeGb: 100
  imageType: GCI
  machineType: n1-standard-1
```

Using same now you can get connected to the Kubernetes-dashboard:

Internal


kubernetes
Nodes
+ CREATE
Admin
Namespaces
Nodes
Persistent Volumes
Namespace
default
Workloads
Deployments
Replica Sets
Replication Controllers
Daemon Sets
Pet Sets
Jobs
Pods
Services and discovery
Name
Labels
Ready
Age
gke-cluster-2-default-pool-ccf27a80-1i6h
beta.kubernetes.io/arch: amd64
beta.kubernetes.io/instance-type: n1-standard-1
beta.kubernetes.io/os: linux
cloud.google.com/gke-nodepool: default-pool
failure-domain.beta.kubernetes.io/region: us-central1
show all labels
True
41 minutes
gke-cluster-2-default-pool-ccf27a80-7vme
beta.kubernetes.io/arch: amd64
beta.kubernetes.io/instance-type: n1-standard-1
beta.kubernetes.io/os: linux
cloud.google.com/gke-nodepool: default-pool
failure-domain.beta.kubernetes.io/region: us-central1
show all labels
True
41 minutes
gke-cluster-2-default-pool-ccf27a80-lrg3
beta.kubernetes.io/arch: amd64
beta.kubernetes.io/instance-type: n1-standard-1
beta.kubernetes.io/os: linux
cloud.google.com/gke-nodepool: default-pool
failure-domain.beta.kubernetes.io/region: us-central1
show all labels
True
41 minutes

1. ***Deploy App :***



To deploy the application in Kubernetes it need to be in container formats. In this Assignment we will be using the image of NodeJS application packaged in Docker container. You can get the source code of the application here.

Kubectl comes installed in gcloud.

We need provide the deployment name and app image location in console:

$kubectl run my-app --image=docker.io/jocatalin/kubernetes-bootcamp:v1 --port=8080

My-app is name given to deployment which is run on port 8080.

```
etainfy@mamta-146808:~$ kubectl run my-app --image=docker.io/jocatalin/kubernetes-bootcamp:v1 --port=8080
deployment "my-app" created
```

To list the deployments type command: $kubectl get deployments

```
etainfy@mamta-146808:~$ kubectl get deployments
NAME       DESIRED   CURRENT   UP-TO-DATE   AVAILABLE   AGE
my-app     1         1         1            1           2m
```

To view the application output we will create a route between our terminal and Kubernete cluster by typing command: $ kubectl proxy

Now our application is running in a Pod. Get the name of the Pod and store in an environment variable POD_NAME:

```
etainfy@mamta-146808:~$ export POD_NAME=$(kubectl get pods -o go-template --template '{{range.items}}{{.metadata.name}}{{"
"}}{{end}}')
etainfy@mamta-146808:~$ echo Name of the Pod : $POD_NAME
Name of the Pod : my-app-187288294-2iv1r
```

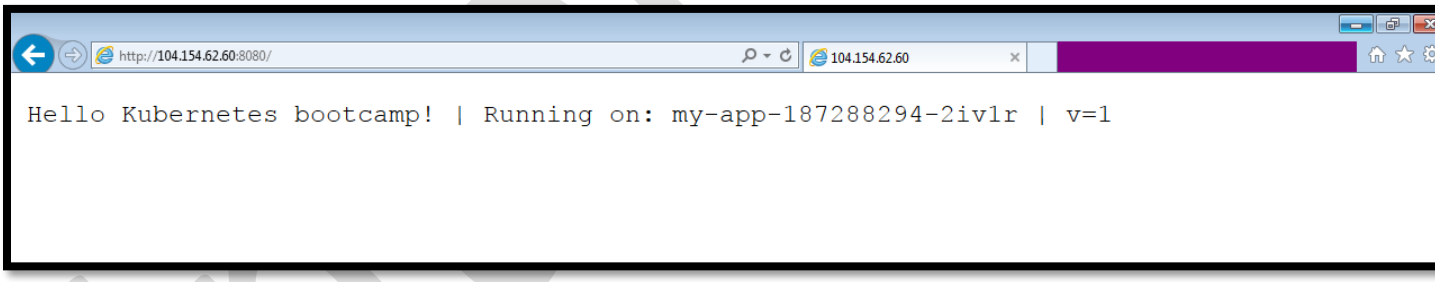
Internal

To verify our application running type command: $ kubectl get services

To expose the service publicly, use expose command as shown:

```
etainfy@mamta-146808:~$ kubectl expose deployment/my-app --type="LoadBalancer" --port 8080
service "my-app" exposed
etainfy@mamta-146808:~$ kubectl get services
NAME         CLUSTER-IP    EXTERNAL-IP   PORT(S)    AGE
kubernetes   10.3.240.1    <none>        443/TCP    56m
my-app       10.3.244.19   <pending>     8080/TCP   14s
```

You can get the External-Ip of the App after some time. Use that IP paste it in browser, you will get output of your deployment in browser:



Estimated time: 20 mins

Summary: *In this assignment, developer will be able to successfully explore Kubernetes environment and launch an application in Kubernetes cluster of GKE.*

Internal