**PRACTICAL: 2**

**AIM:**

Google Cloud Essentials: In this introductory-level Quest Lab, you will get hands-on practice with the Google Cloud’s fundamental tools and services. Google Cloud Essentials is the recommended first Quest for the Google Cloud learner - you will come in with little or no prior cloud knowledge, and come out with practical experience that you can apply to your first Google Cloud project. From writing Cloud Shell commands and deploying your first virtual machine, to running applications on Kubernetes Engine or with load balancing, Google Cloud Essentials is a prime introduction to the platform’s basic features.

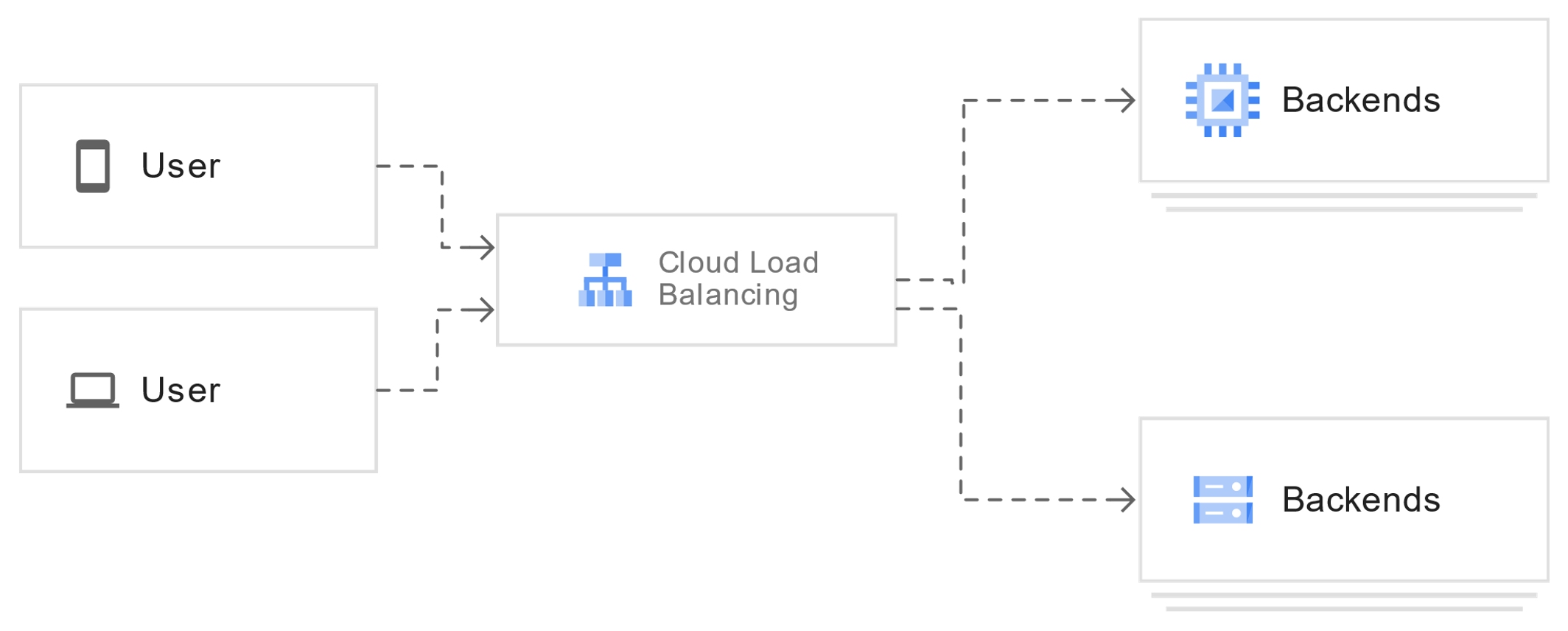
**THEORY:**

**2.1** - Compute Engine lets you create virtual machines running different operating systems, including multiple flavors of Linux (Debian, Ubuntu, Suse, Red Hat, CoreOS) and Windows Server, on Google infrastructure. You can run thousands of virtual CPUs on a system that has been designed to be fast and to offer strong consistency of performance.

**2.2** - Google Cloud Shell provides you with gcloud command-line access to computing resources hosted on the Google Cloud. Cloud Shell is a Debian-based virtual machine with a persistent 5GB home directory, which makes it easy for you to manage your Google Cloud projects and resources. The Cloud SDK gcloud and other utilities you need come pre-installed in Cloud Shell, which allows you to get up and running quickly.

**2.3** - [Google Kubernetes Engine](https://cloud.google.com/kubernetes-engine/) (GKE) provides a managed environment for deploying, managing, and scaling your containerized applications using Google infrastructure. The Kubernetes Engine environment consists of multiple machines (specifically [Compute Engine](https://cloud.google.com/compute) instances) grouped together to form a [container cluster](https://cloud.google.com/kubernetes-engine/docs/concepts/cluster-architecture).

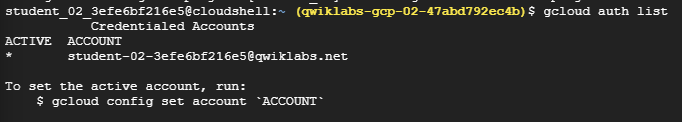
**2.4** - A load balancer distributes user traffic across multiple instances of your applications. By spreading the load, load balancing reduces the risk that your applications experience performance issues. By using Cloud Load Balancing, you can serve content as close as possible to your users on a system that can respond to over one million queries per second.



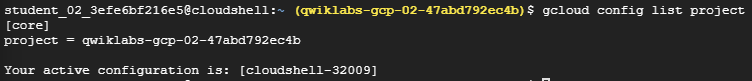
Cloud Load Balancing is a fully distributed, software-defined managed service. It isn't hardware-based, so you don't need to manage a physical load balancing infrastructure.

**OUTPUT:**

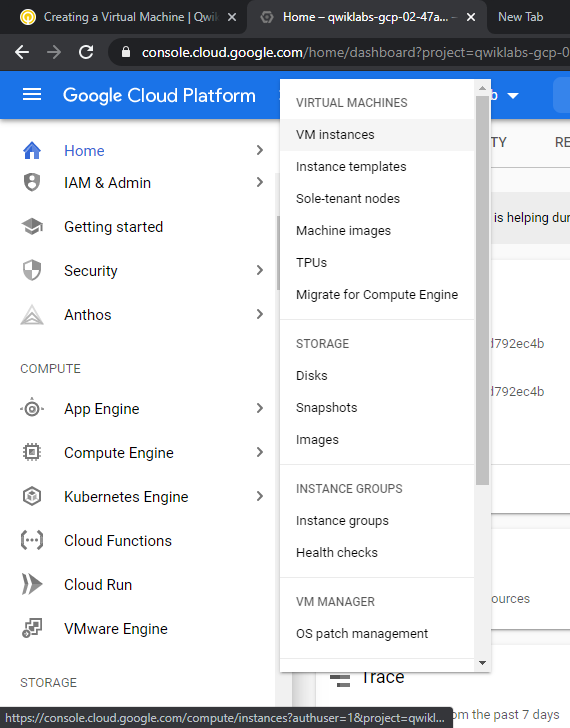
**2.1 – Creating a virtual machine.**



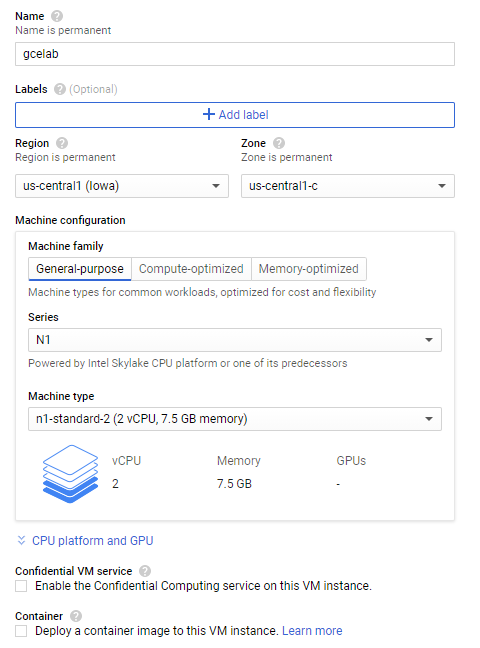
**2.1.1 – Printing authenticated user**



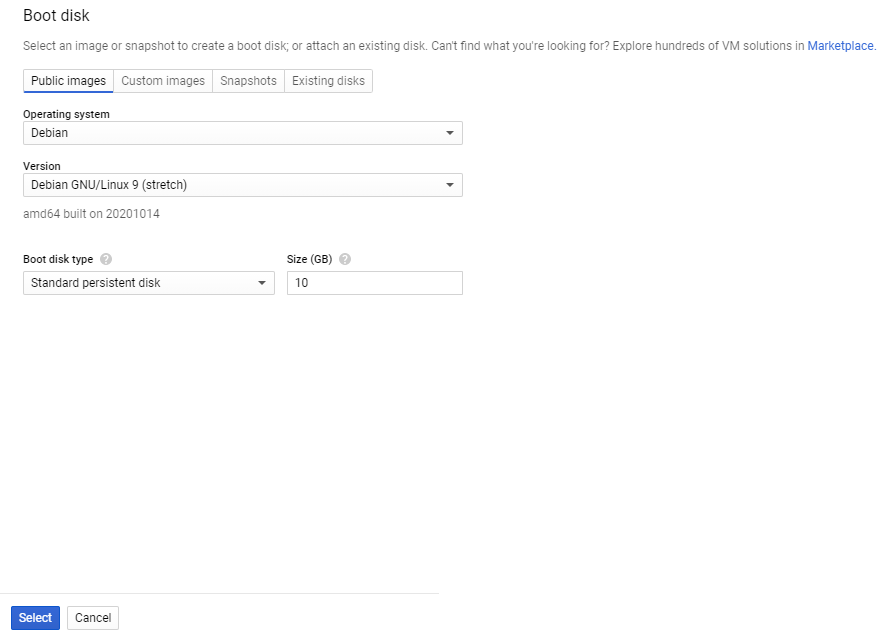
**2.1.2 – Command to print project ID**



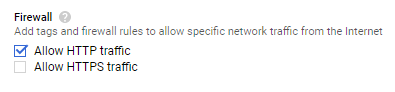
**2.1.3 – Navigating to VM Instances from left side nav bar**



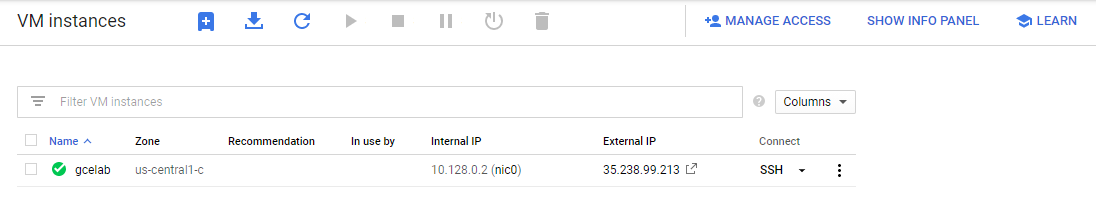
**2.1.4 – Creating new virtual machine**



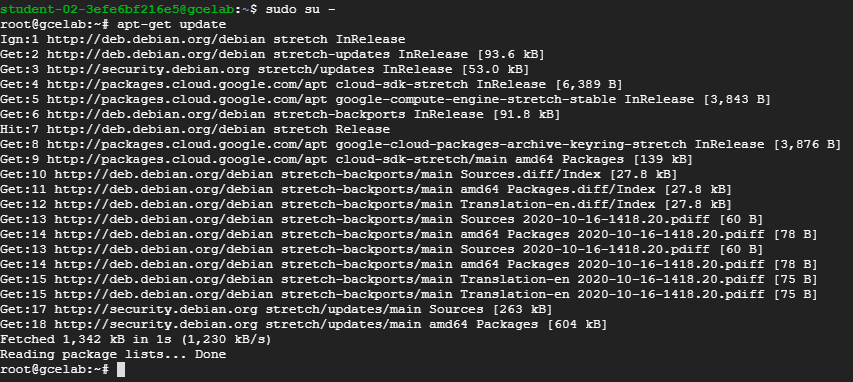
**2.1.5 – Assigning boot disk**



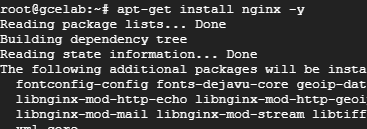
**2.1.6 – Allowing HTTP Traffic**



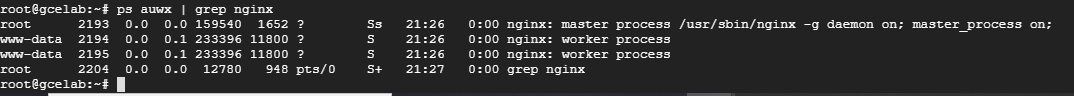
**2.1.7 VM Instance created successfully**



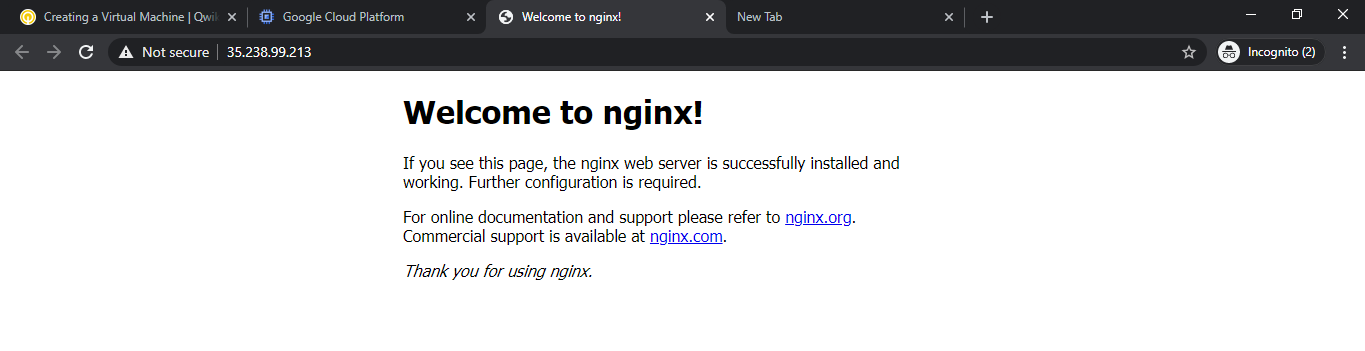
**2.1.8 – SSH to vm instance then update the instance**



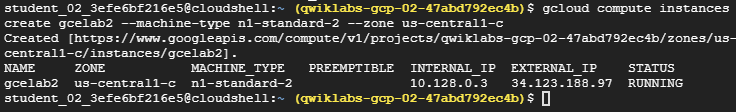
**2.1.9 – Installing nginx server**



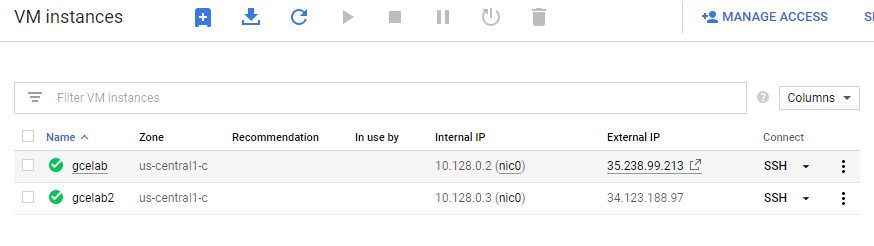
**2.1.10 – Listing out attributes of nginx installed**



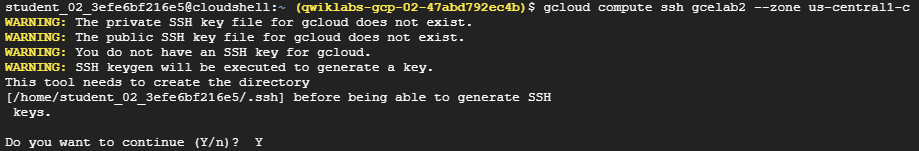
**2.1.11 – Accessing default page by nginx using public IP address of instance**



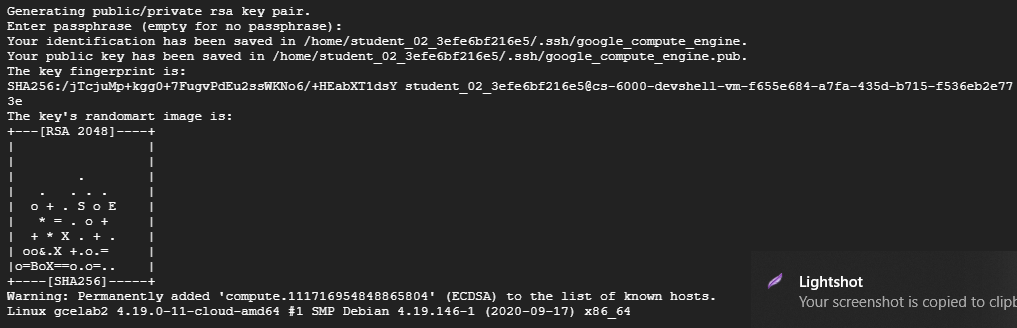
**2.1.12 – Creating instance using cloud shell**



**2.1.12 – Both the instances are up and running**

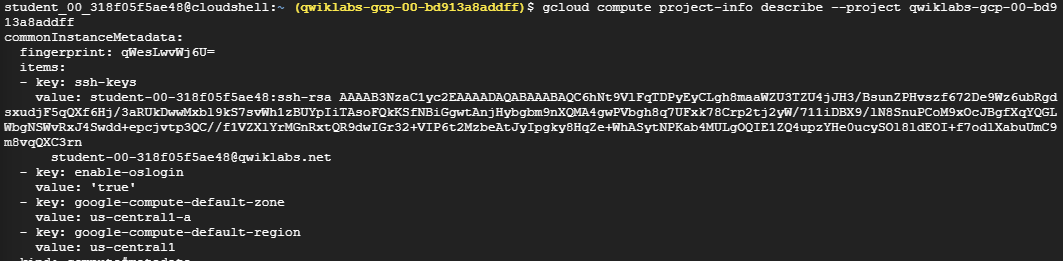


**2.1.13 – cloud sheel to compute engine ssh using command**

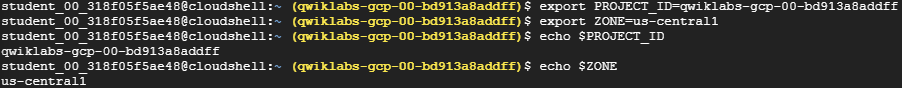


**2.1.14 – Generating secure fingerprint**

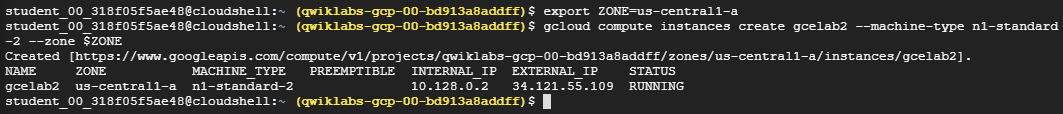
**2.2 – Getting started with cloud shell & gcloud.**



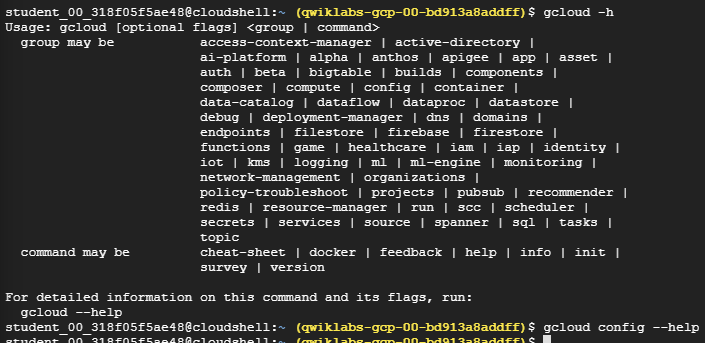
**2.2.1 – Generating secure fingerprint**



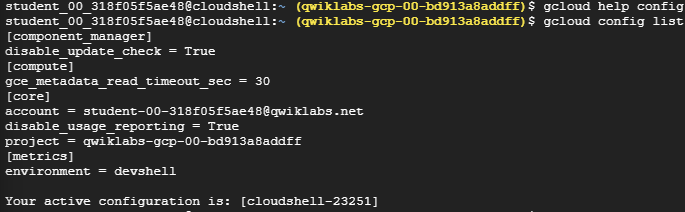
**2.2.2 – Exporting local environment variables**



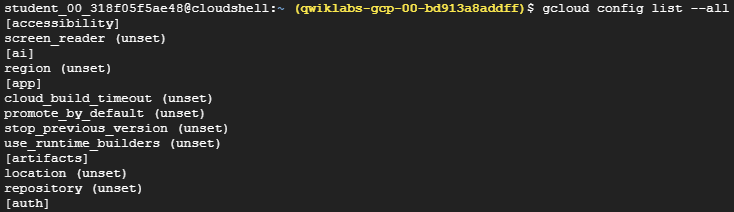
**2.2.3 – Creating compute engine using cloud shell**



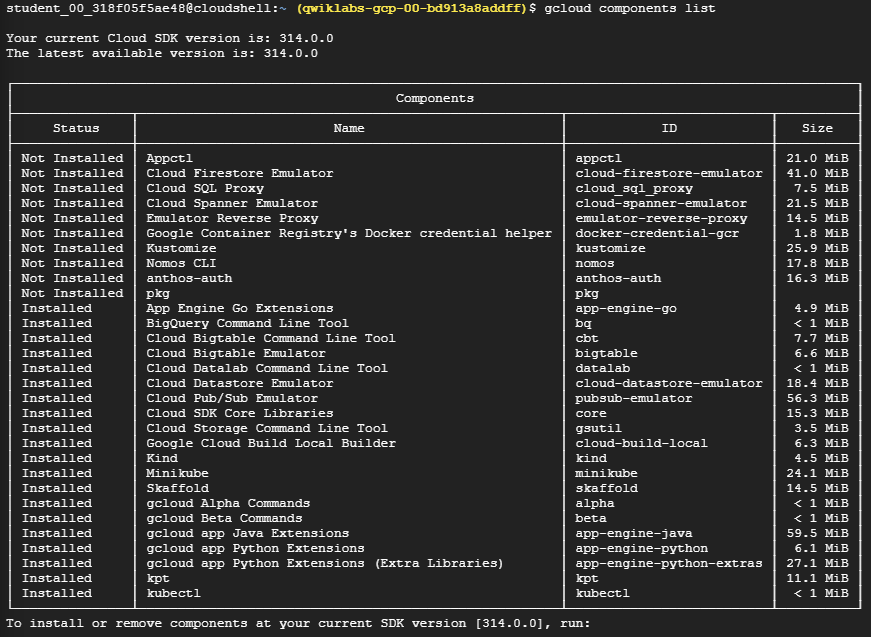
**2.2.4 – Gcloud command description along with flags**



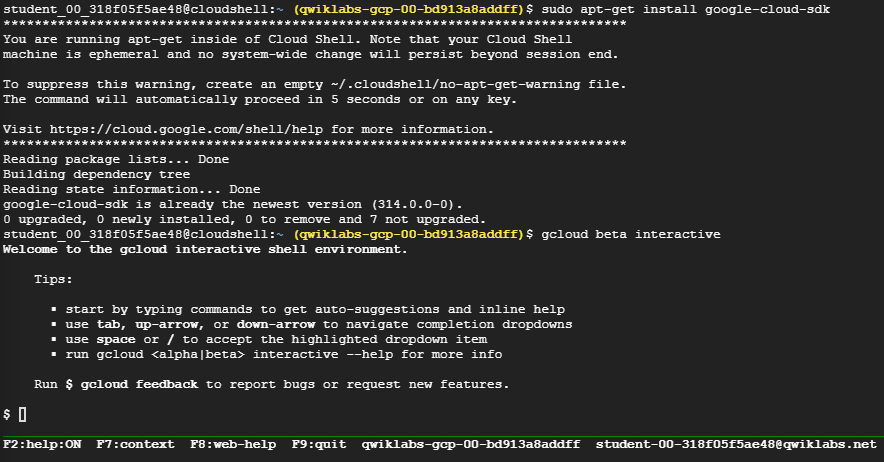
**2.2.5 – Gcloud config list**



**2.2.6 – Gcloud config list all**

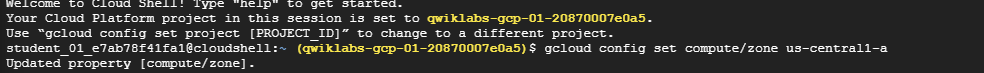


**2.2.7 – Gcloud component list**

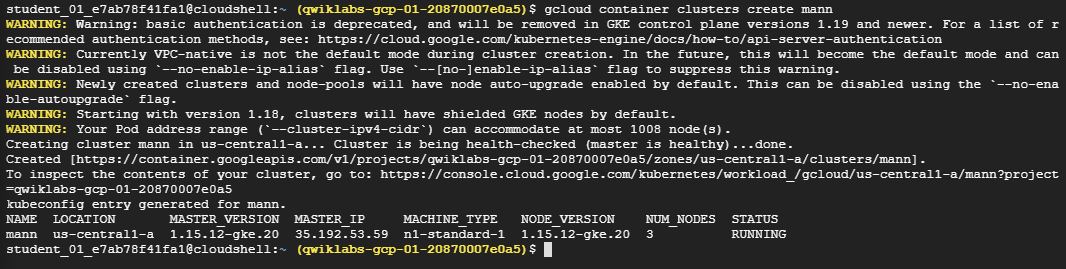


**2.2.8 – Installing google cloud SDK**

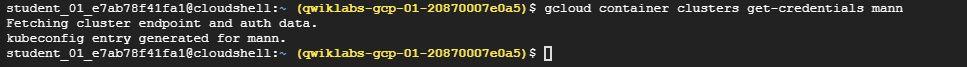
**2.3 – Kubernetes Engine: Qwik Start.**



**2.3.1 – Setting up zone using cloud shell**



**2.3.2 – Creating Cluster with name “mann”**

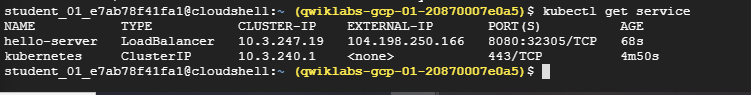


**2.3.3 – Generating credentials for cluster**

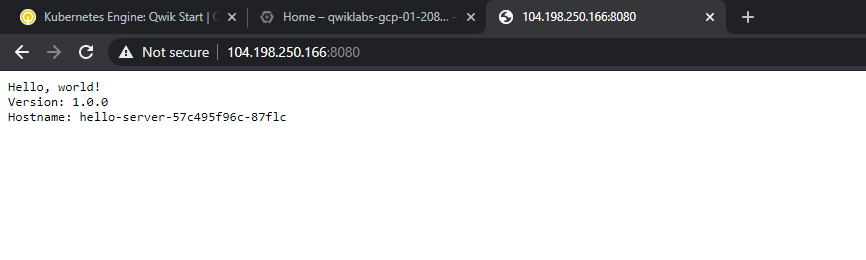




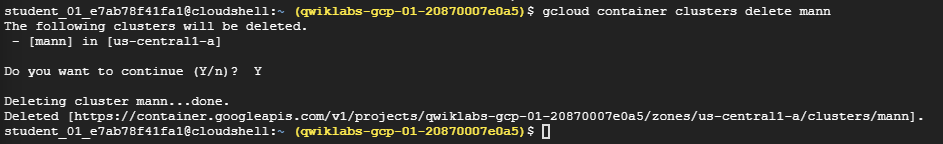
**2.3.4 – Creating server with name “hello-server” and image flag and type flag**



**2.3.5 – Getting information about deployed server**

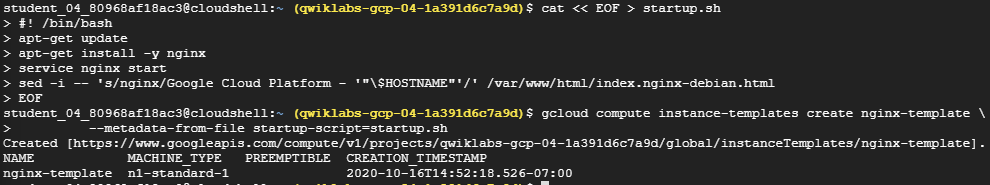


**2.3.6 – Accessing server with external IP**

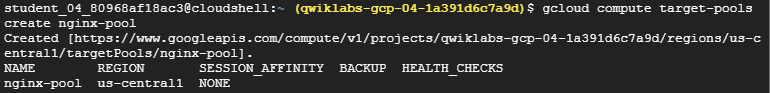


**2.3.7 – Deleting the cluster**

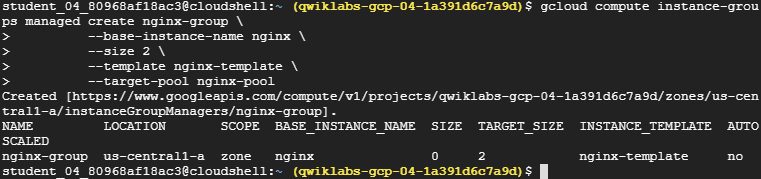
**2.4 – Set Up Network and HTTP Load Balancers.**



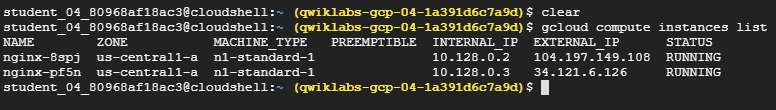
**2.4.1 – Startup Script**



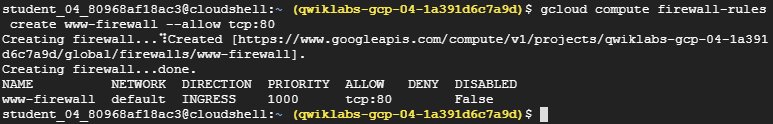
**2.4.2 – Creating compute target pools**



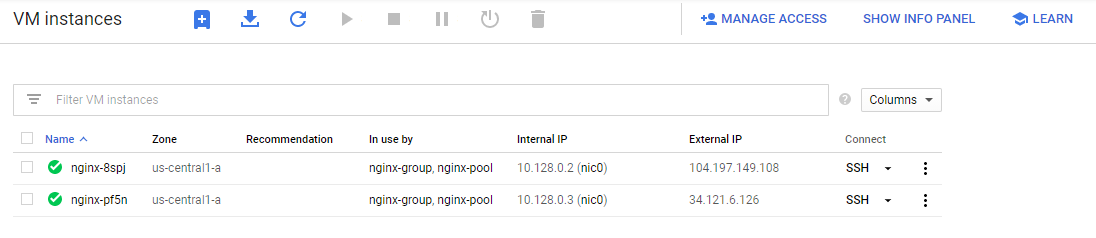
**2.4.3 – Creating instance group**



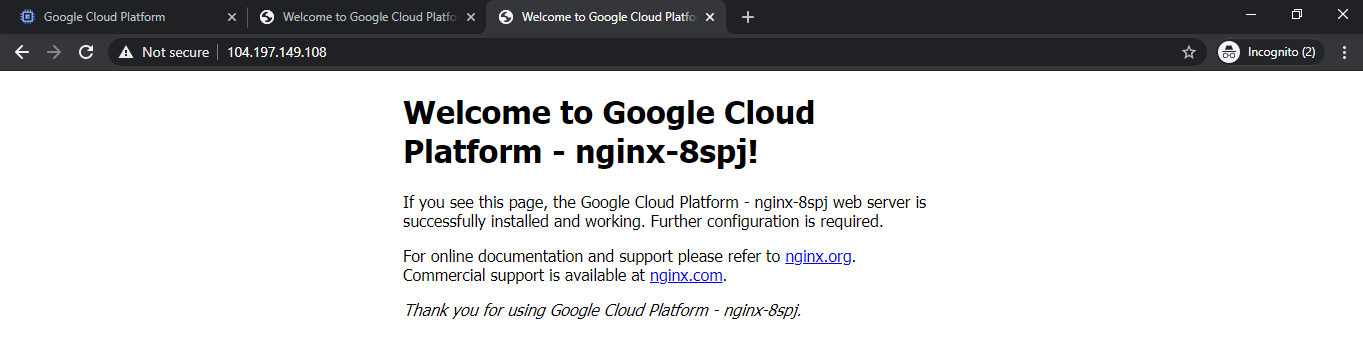
**2.4.4 – Listing both instance we created**



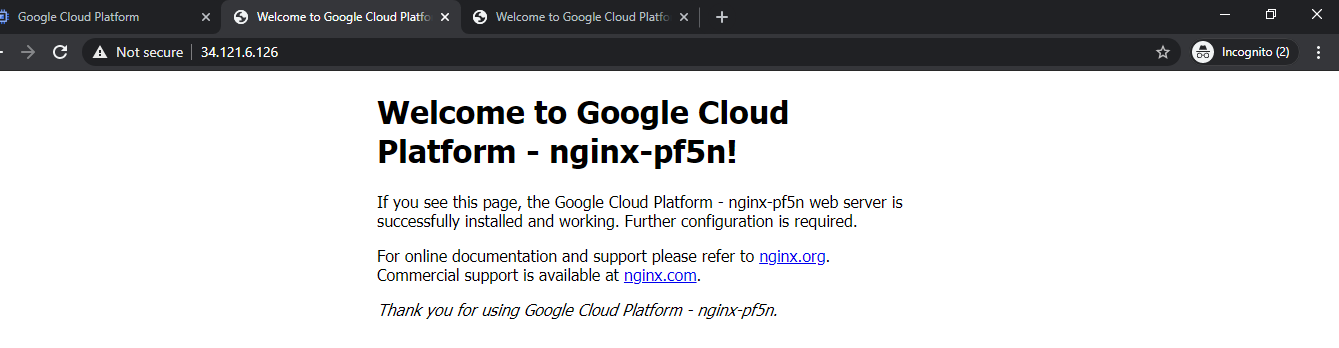
**2.4.5 – Configure firewall rules and create www-firewall rule with 80 port allowance**



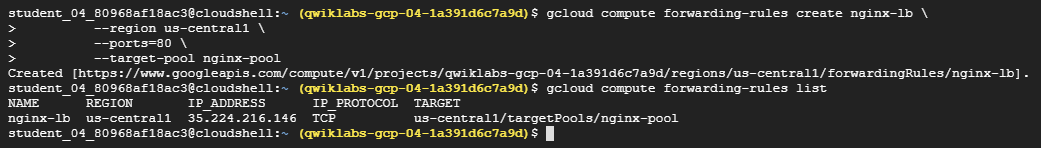
**2.4.6 – Verifying instance from the VM Instances dashboard UI**



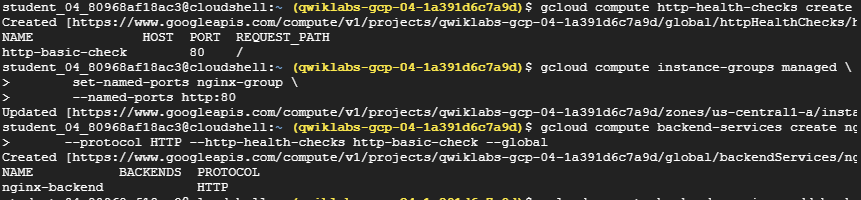
**2.4.7 – Using external IP to check if server configure successfully for instance 1**



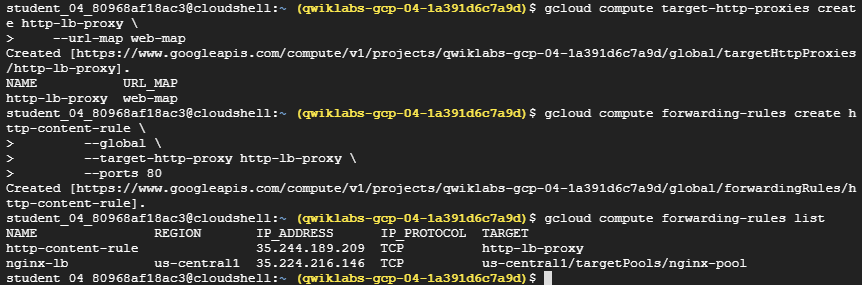
**2.4.8 – Using external IP to check if server configure successfully for instance 2**



**2.4.9 – Setting up forwarding rules**



**2.4.10 – Creating health checks**



**2.4.11 – Setting up load balancer proxies**

**LATEST APPLICATIONS:**

All the clouds (private, public, hybrid, and multi) took center stage for enterprise IT, while containers and microservices architecture captivated developers. However, legacy load balancing vendors continued to give you two flavors: hardware load balancer appliances (vanilla) or virtual load balancer appliances (diet vanilla).

Kubernetes Engine - The GKE environment consists of multiple machines (specifically, Compute **Engine** instances) grouped together to form a cluster. Kubernetes engine consider to be hybrid of IAAS and PAAS. (Infrastructure as a service and platform as a service).

**LEARNING OUTCOME:**

* 1. Create google compute engine virtual machine using google cloud shell and understand zones, regions and machine type.
  2. Connect to computing resources hosted on google cloud platform and install google sdk through cloud shell.
  3. Create Kubernetes engine from scratch by starting from creating cluster then configure Kubernetes service through cloud shell and in the end delete the cluster for terminating any billings if it was.
  4. Implemented load balancer by creating tow virtual instances and configure security rules and forwarding paths through cloud shell.

**REFERENCE:**

1. <https://www.qwiklabs.com/quests/23>
2. <https://blogs.vmware.com/load-balancing/2020/01/15/load-balancing-in-2020-and-beyond/>