**Virtual Machine Lab**

**Duration: 1 hour**

**1. Overview**

Google Compute Engine lets you create and run virtual machines on Google infrastructure. Compute Engine offers scale, performance, and value that allows you to easily launch large compute clusters on Google's infrastructure. There are no upfront investments and you can run thousands of virtual CPUs on a system that has been designed to be fast, and to offer strong consistency of performance.

You can create virtual machines running different operating systems, including multiple flavors of Linux (Debian, Ubuntu, Suse, Red Hat, CoreOS) and Windows Server!

Follow along this lab to learn about how to create virtual machine instances of various machine types.

What you'll learn

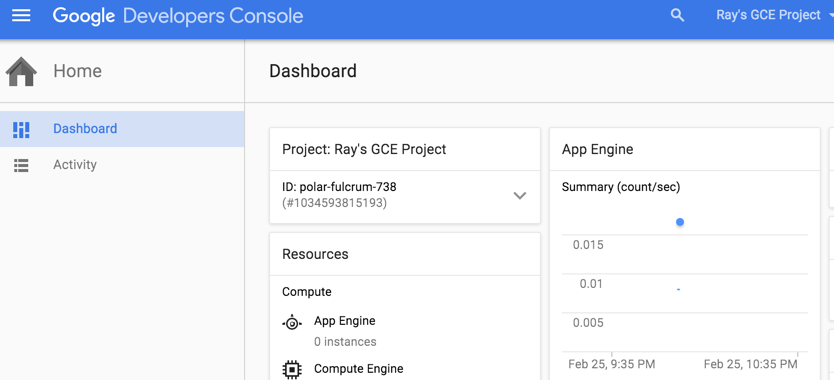
* Google Compute Engine
* How to create a virtual machine
* Predefined machine types and custom machine types

Virtual machine Instances and persistent disks live in a zone, and these are referred to as zonal resources. For example, to attach a persistent disk to a virtual machine instance, both resources must be in the same zone.

## 2. Create a new instance using Google Cloud Console

In this section, you'll learn how to create a new Google Compute Engine virtual machine instance from the Cloud Console.

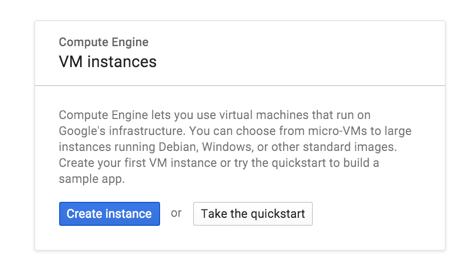
## Login into Google Cloud Console



## Click on Navigation menu (Products and Service) 7a91d354499ac9f1.png -> Compute Engine

Then navigate to **Compute Engine** > **VM Instances**

This may take a minute to initialize for the first time. Once initialized, you should see the following if the project has no virtual machine instances:



## Create a new instance

To create a new instance, click **Create instance**

There are many parameters you can configure when creating a new instance. Let's use the following:

|  |  |
| --- | --- |
| **Name** | gcelab |
| **Zone** | europe-west1-d  Learn more about zones in [Regions & Zones documentation](https://cloud.google.com/compute/docs/zones).  **Note**: remember the zone that you selected, you'll need it later. |
| **Machine Type** | 1 vCPU (n1-standard-1)  This is a 1-CPU, 3.75GB RAM instance.  There are a number of machine types, ranging from micro instance types to 32-core/208GB RAM instance types. Learn more about this and custom machine types in the [Machine Types documentation](https://cloud.google.com/compute/docs/machine-types).  **Note**: A new project has a default [resource quota](https://cloud.google.com/compute/docs/resource-quotas), which may limit the number of CPU cores. You can request more for your projects. |
| **Boot Disk** | New 10 GB standard persistent disk  Image: Debian GNU/Linux 8 (jessie)  There are a number of images to choose from, including: Debian, Ubuntu, CoreOS as well as premium images such as Red Hat Enterprise Linux and Windows Server. See Operating System documentation for more detail.  **Note:** You may optionally choose SSD Persistent Disk as well. |
| **Firewall** | Check Allow HTTP traffic  Check this option so that we can access a webserver that we will install later.  **Note:**This will automatically create firewall rule to allow HTTP traffic on port 80. |

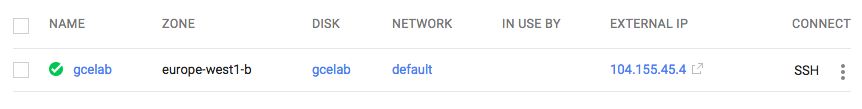
Click on **Create** to create the new virtual machine instance!

**Note:**The instance creation process is asynchronous. You can check on the status of the task using the top right hand-side **Activities** icon. Wait for it to finish - it shouldn't take more than a minute.

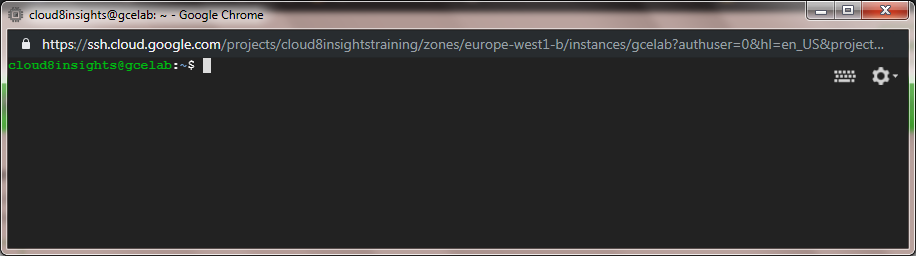
**Hint:**If you receive an error when creating a VM, click into **Details**. Most likely, you may need to try with a different zone.

Once finished, you should see the new virtual machine in the **VM Instances** page. Note down the External IP of the machine on the VM instances page - you will need this later

To SSH into the virtual machine, click on **SSH** on the right hand side.



**Note:**This launches a SSH client directly from your browser. You can also SSH into the virtual machine using ssh. See the Connect to an instance using ssh documentation.



## Install a NGINX web server

Once ssh'ed, get root access using sudo

$ sudo su -

As the root user, install NGINX:

$ apt-get update

$ apt-get install nginx -y

**Start the nginx server**

/etc/init.d/nginx status

/etc/init.d/nginx restart

**Check that NGINX is running:**

$ psauwx| grep nginx

root13730.00.0911802960?Ss03:590:00nginx: master process /usr/sbin/nginx-g daemon on

;master\_process on;

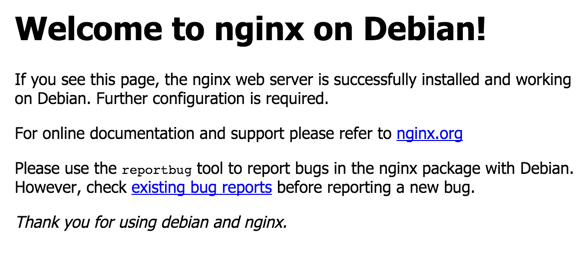
www-data 13740.00.0915443652? S 03:590:00nginx: worker process

www-data 13750.00.0915443652? S 03:590:00nginx: worker process

www-data 13760.00.0915443652? S 03:590:00nginx: worker process

www-data 13770.00.0915443652? S 03:590:00nginx: worker process

Awesome! Let's see the web page! Use the External IP of the virtual machine instance that you noted down before, and in a new browser window/tab, browse to [http://EXTERNAL\_IP/](http://external_ip/), you will able to see the default page.



**Create a new instance with cloud shell**

Rather than using the Google Cloud Console to create a virtual machine instance, you can also use the command line tool called cloud sdk from your machine or cloud shell.

We'll use [Google Cloud Shell](https://cloud.google.com/developer-shell/#how_do_i_get_started),[a command line environment running in the Cloud](https://cloud.google.com/developer-shell/#how_do_i_get_started). This Debian-based virtual machine is loaded with all the development tools you'll need (gcloud, git and others) and offers a persistent 5GB home directory.

You don't need to manually install gcloud for this lab since it's preinstalled in Cloud Shell. But if you want to try this on your own machine in the future, read the [gcloud command line tool guide](https://cloud.google.com/sdk/gcloud/).

Open the Google Cloud Shell by clicking on the https://codelabs.developers.google.com/codelabs/cloud-create-a-vm/img/3b409efdcc18f697.pngicon on the top right of the screen:

https://codelabs.developers.google.com/codelabs/cloud-create-a-vm/img/446be36b8e6592.png

Once opened, you can create a new virtual machine instance from the command line by using gcloud:

$ gcloud compute instances create gcplab2 --zone europe-west1-d

The instance was created with some default values:

* The latest Debian 8 (Jessie) image.
* The n1-standard-1 machine type. You can select another machine type such as n1-highmem-4 or n1-highcpu-6. If none of the predefined machine types match your needs, use a custom machine type.
* A root persistent disk with the same name as the instance; the disk is automatically attached to the instance.

Run gcloud compute instances create --help to see all the defaults.

**Note: You can set the default region and zones that gcloud uses if you are always working within one region/zone and you don't want to append the --zone flag every time. You can do so by running these commands :**

$ gcloud config set project cloud8insights

$ gcloud config set compute/zone ...

$ gcloud config set compute/region ...

Finally, you can SSH into your instance using gcloud as well:

$ gcloud compute ssh gcplab2 --zone us-central1-c

Warning: Permanently added 'compute.745019997240233037' (ECDSA) to the list of known hosts.

...

username@gcelab2:~$

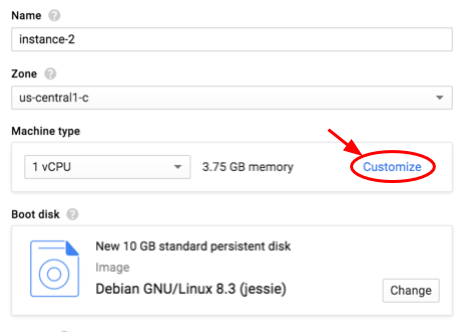
Disconnect from SSH by existing from the remote shell:

username@gcelab2:~$ exit

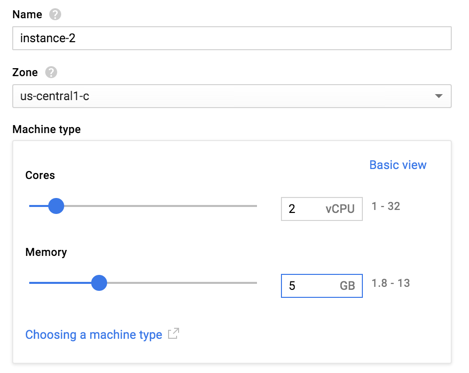
## Custom machine types

In the previous section, you learned that there are a number of [predefined machine types](https://cloud.google.com/compute/docs/machine-types#predefined_machine_types) that you can choose for your virtual machine. However, if predefined machine types do not meet your needs, you can create an instance with custom virtualized hardware settings. Specifically, you can create an instance with a custom number of vCPUs and amount of memory, effectively using a custom machine type.

Create another virtual machine instance using the Google Cloud Console, and choose a custom machine type by clicking **Customize**:



Then, you use the sliders to pick the exact number of cores and memory to use:



Create a virtual machine instance with 2 vCPU and 5 GB of RAM.

ssh into VM and do whatever you want to do. Actually you got a machine with linux OS where you can perform your desired operations like language installation, packages and library installations and application development or deployment.

**Create a new instance with start-up script**

Create the VM

1. In the GCP Console, on the **Navigation menu** (7a91d354499ac9f1.png), click **Compute Engine** > **VM instances**.
2. Click **Create**.
3. Specify the following, and leave the remaining settings as their defaults:

|  |  |
| --- | --- |
| **Property** | **Value**  (type value or select option as specified) |
| **Name** | **webserver1** |
| **Region** | us-central1 |
| **Zone** | us-central1-a |
| **Access scopes** | **Set access for each API**  For **Storage**, ensure that **Read Only** is set\* |
| **Firewall** | **Allow HTTP traffic** |

This allows this VM to read from the Cloud Storage bucket during the boot process.

1. Click **Management, security, disks, networking, sole tenancy**.
2. For **Metadata**, add the following key-value pair:

|  |  |
| --- | --- |
| **Property** | **Value**  (type value or select option as specified) |
| **Key** | **startup-script-url** |
| **Value** | **gs://cloud-training/archinfra/mystartupscript** |

1. Click **Add item**. Add another key-value pair, this time with the unique name of your web server:

|  |  |
| --- | --- |
| **Property** | **Value**  (type value or select option as specified) |
| **Key** | **my-server-id** |
| **Value** | **WebServer-1** |

7. Click **Create**.

1. On the **VM instances** page, for **webserver1**, click the **External IP**. This opens a new index page with the name "WebServer-1." If you get a "This site can't be reached" error, you might have to refresh the page: for all VMs created in this lab, it may take a minute or two before the web server starts serving content. Wait a minute and then refresh the page.

**Contents of start-up Script**

#! /bin/bash

VALUE\_OF\_MY\_SERVER\_ID=$(curl http://metadata.google.internal/computeMetadata/v1/instance/attributes/my-server-id -H "Metadata-Flavor: Google")

apt-get update

apt-get install -y apache2

cat - > /var/www/html/index.html << EOF

<html><body><h1>Virtual Machine Lab, Architecting GCP Infrastructure </h1>

<p>Server Name is: $VALUE\_OF\_MY\_SERVER\_ID</p>

</body></html>

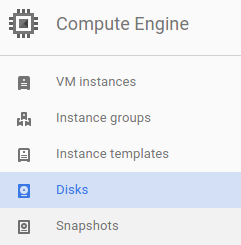
EOF

**Clean up**

It's time to shut down those VM instances to save on cost and to be an overall good cloud citizen. You can either decide to delete the instances or simply start them :



If you decide to delete the VM instances (after all they're really easy to recreate), then remember that this will also delete the VM boot disks. If you'd like to retain persistent disks, check out the Disks and Snapshots sections of the compute engine console :



8. Congratulations!

Google Compute Engine is the foundation to Google Cloud Platform's Infrastructure-as-a-Service. You can easily map your existing server infrastructure, load balancers, and network topology to Google Cloud Platform.

What we've covered

* Google Compute Engine
* How to create a virtual machine
* Predefined machine types and custom machine types

Next Steps

* Wondering how much it'll cost to run your workloads? Check the Pricing Calculator - you can enter the number of instances, vCPU, and memory needs.
* Learn more about load balancers.
* Learn more about subnetworks and network topology.

gcloud cheatsheet

|  |  |
| --- | --- |
| **Name** | **Summary** |
| Check version & settings | gcloud version, gcloud info, gcloud components list |
| Init profile | gcloud init |
| List all zones | gcloud compute zones list |
| List projects | gcloud config list project |
| Show project info | gcloud compute project-info describe |
| Switch project | gcloud config set project <project-id> |
| Authenticate client | gcloud auth activate-service-account --key-file <key-file> |
| Display a list of credentialed accounts | gcloud auth list |
| Set the active account | gcloud config set account <ACCOUNT> |
| Print token for active account | gcloud auth print-access-token, gcloud auth print-refresh-token |
| Revoke previous generated credential | gcloud auth <application-default> revoke |
| List all instances | gcloud compute instances list, gcloud compute instance-templates list |
| Show instance info | =gcloud compute instances describe “<instance-name>” –project “<project-name>” –zone “us-west2-a”= |
| Stop an instance | gcloud compute instances stop instance-2 |
| Start an instance | gcloud compute instances start instance-2 |
| Create an instance | gcloud compute instances create vm1 --image image-1 --tags test --zone "<zone>" --machine-type f1-micro |
| SSH to instance | gcloud compute ssh –project “<project-name>” –zone “<zone-name>” “<instance-name>”= |
| Download files | gcloud compute copy-files example-instance:~/REMOTE-DIR ~/LOCAL-DIR --zone us-central1-a |
| Upload files | gcloud compute copy-files ~/LOCAL-FILE-1 example-instance:~/REMOTE-DIR --zone us-central1-a |
| List all networks | gcloud compute networks list |
| Detail of one network | gcloud compute networks describe <network-name> --format json |
| Create network | gcloud compute networks create <network-name> |
| Create subnet | gcloud compute networks subnets create subnet1 --network net1 --range 10.5.4.0/24 |
| Get a static ip | gcloud compute addresses create --region us-west2-a vpn-1-static-ip |
| List all ip addresses | gcloud compute addresses list |
| Describe ip address | gcloud compute addresses describe <ip-name> --region us-central1 |
| List all routes | gcloud compute routes list |
| List all firewall rules | gcloud compute firewall-rules list |
| List all forwarding rules | gcloud compute forwarding-rules list |
| Describe one firewall rule | gcloud compute firewall-rules describe <rule-name> |
| Create one firewall rule | gcloud compute firewall-rules create my-rule --network default --allow tcp:9200 tcp:3306 |
| Update one firewall rule | gcloud compute firewall-rules update default --network default --allow tcp:9200 tcp:9300 |
| Upgrade local SDK | gcloud components update, gcloud components update --version 219.0.1 |

**Full Command to create VM:**

gcloud compute instances create INSTANCE\_NAMES [INSTANCE\_NAMES …] [--accelerator=[count=COUNT],[type=TYPE]] [--async] [--no-boot-disk-auto-delete] [--boot-disk-device-name=BOOT\_DISK\_DEVICE\_NAME] [--boot-disk-size=BOOT\_DISK\_SIZE] [--boot-disk-type=BOOT\_DISK\_TYPE] [--can-ip-forward] [--create-disk=[PROPERTY=VALUE,…]] [--csek-key-file=FILE] [--deletion-protection] [--description=DESCRIPTION] [--disk=[auto-delete=AUTO-DELETE],[boot=BOOT],[device-name=DEVICE-NAME],[mode=MODE],[name=NAME]] [--hostname=HOSTNAME] [--labels=[KEY=VALUE,…]] [--local-ssd=[device-name=DEVICE-NAME],[interface=INTERFACE]] [--machine-type=MACHINE\_TYPE] [--maintenance-policy=MAINTENANCE\_POLICY] [--metadata=KEY=VALUE,[KEY=VALUE,…]] [--metadata-from-file=KEY=LOCAL\_FILE\_PATH,[…]] [--min-cpu-platform=PLATFORM] [--network=NETWORK] [--network-interface=[PROPERTY=VALUE,…]] [--network-tier=NETWORK\_TIER] [--preemptible] [--private-network-ip=PRIVATE\_NETWORK\_IP] [--no-require-csek-key-create] [--no-restart-on-failure] [--shielded-integrity-monitoring] [--shielded-secure-boot] [--shielded-vtpm] [--source-instance-template=SOURCE\_INSTANCE\_TEMPLATE] [--subnet=SUBNET] [--tags=TAG,[TAG,…]] [--zone=ZONE] [--address=ADDRESS | --no-address] [--boot-disk-kms-key=BOOT\_DISK\_KMS\_KEY : --boot-disk-kms-keyring=BOOT\_DISK\_KMS\_KEYRING --boot-disk-kms-location=BOOT\_DISK\_KMS\_LOCATION --boot-disk-kms-project=BOOT\_DISK\_KMS\_PROJECT] [--custom-cpu=CUSTOM\_CPU --custom-memory=CUSTOM\_MEMORY : --custom-extensions] [--image-project=IMAGE\_PROJECT --image=IMAGE | --image-family=IMAGE\_FAMILY] [--node=NODE | --node-affinity-file=NODE\_AFFINITY\_FILE | --node-group=NODE\_GROUP] [--public-ptr | --no-public-ptr] [--public-ptr-domain=PUBLIC\_PTR\_DOMAIN | --no-public-ptr-domain] [--scopes=[SCOPE,…] | --no-scopes] [--service-account=SERVICE\_ACCOUNT | --no-service-account] [GCLOUD\_WIDE\_FLAG …]

**Example:**

**export MY\_ZONE1=us-central1-a**

gcloud compute instances create webserver1 \

--image-family debian-9 \

--image-project debian-cloud \

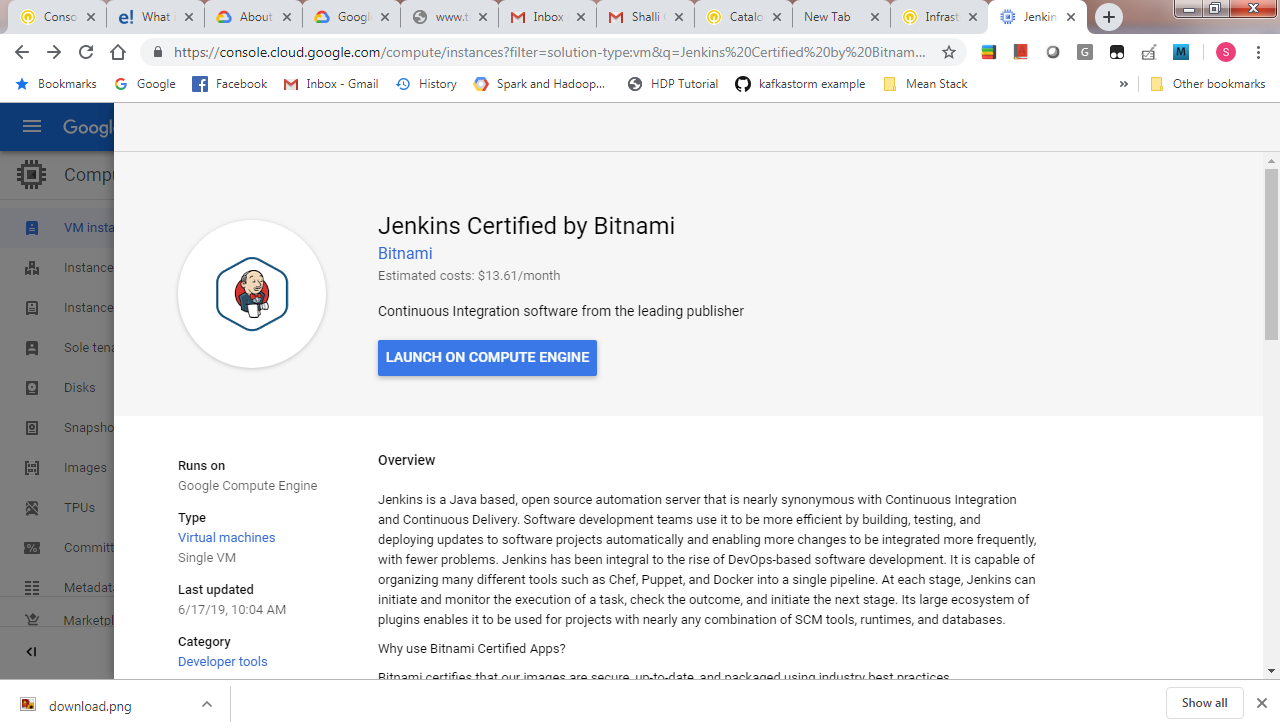
--zone $MY\_ZONE1 \

--subnet default \

--metadata startup-script-url="gs://cloud-training/archinfra/mystartupscript",my-server-id="WebServer-1"

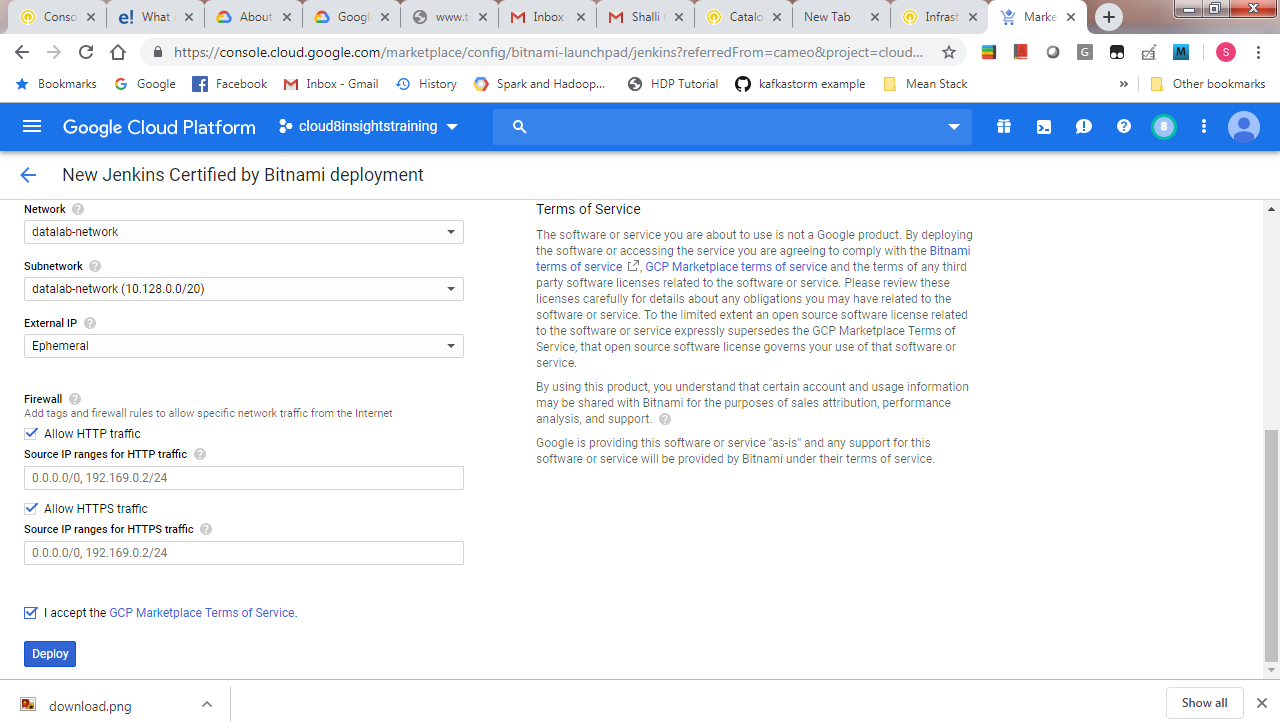
**Task1: Use Marketplace to build a deployment**

1. In the GCP Console, on the Navigation menu (7a91d354499ac9f1.png ), click Compute Engine -> Marketplace.
2. Locate the Jenkins deployment by searching for Jenkins Certified by Bitnami.
3. Click on the deployment and read about the service provided by the software.



**Jenkins is an open-source continuous integration environment. You can define jobs in Jenkins that can perform tasks such as running a scheduled build of software and backing up data. Notice the software that is installed as part of Jenkins shown in the left side of the description. The service you are using, Marketplace, is part of Google Cloud Platform. The Jenkins template is developed and maintained by an ecosystem partner named Bitnami. The template system is part of another GCP service called Deployment Manager. Later in this class you learn how templates such as this one can be built. That service is available to you. You can create templates like the one you are about to use.**

1. Verify the deployment, accept the terms and services and click **Deploy**.
2. Click **Close** on the Welcome to Deployment Manager window.



**Task 2: Examine the deployment**

1. In the right pane, click **More about the software** to view additional software details. Look at all the software that was installed.
2. Copy the **Admin user** and **Admin password** values to a text editor.
3. Click **Visit the site** to view the site in another browser tab. If you get an error, you might have to reload the page a couple of times.
4. Log in with the **Admin user** and **Admin password** values.
5. After logging in, you will be asked to Customize Jenkins. Click **Install suggested plugins**, and then click **Restart** after the installation is complete. The restart will take a couple of minutes.

**Task 3: Administer the service**

1. View the deployment and SSH to the VM
2. In the GCP Console, on the Navigation menu( ), click Deployment Manager.
3. Click jenkins-1.
4. Click SSH to connect to the Jenkins server.
5. Shut down and restart the services
6. In the SSH window, enter the following command to shut down all the running services:
7. sudo /opt/bitnami/ctlscript.sh stop
8. Refresh the browser window for the Jenkins UI. You will no longer see the Jenkins interface because the service was shut down.
9. In the SSH window, enter the following command to restart the services: sudo /opt/bitnami/ctlscript.sh restart
10. Return to the browser window for the Jenkins UI and refresh it. You may have to do it a couple of times before the service is reachable.
11. In the SSH window, type exit to close the SSH terminal session.
12. Delete the VM.