# Lab 4: DOCKER & python flask

## Learning Outcomes:

After completing this lab, you should be able:

1. To understand the challenges of deploying software

2. To understand docker framework.

3. To build simple docker image.

4. To run a custom docker container.

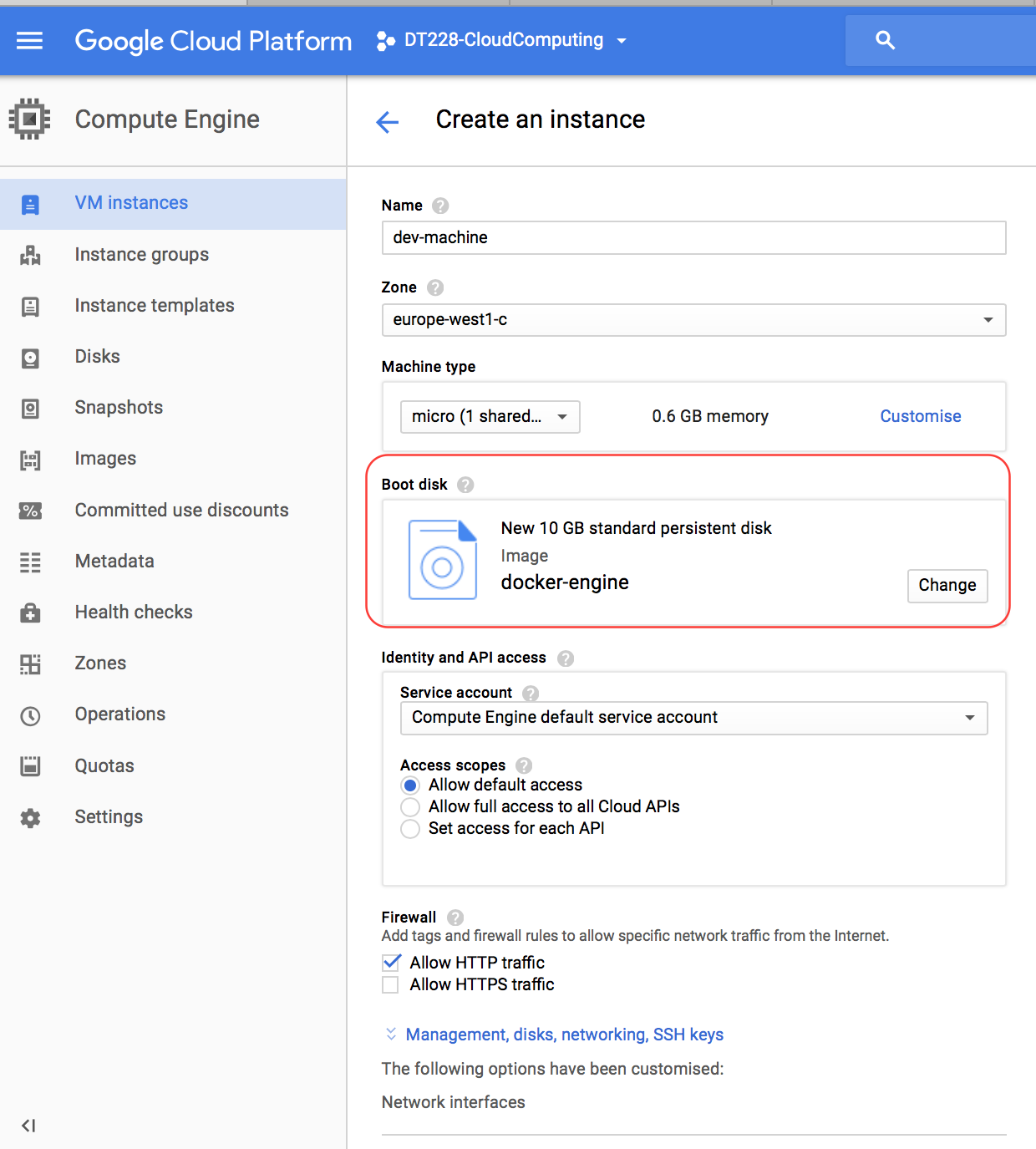
## Organisation

Please attempt this lab individually as you will need this to be completed in order to complete subsequent labs. Part 1 to Part 4 are compulsory. Part 5 is an optional challenging task.

## Part 1. Create DOCKER host

Make sure you have watched the Live tutorial video <https://youtu.be/8kjm4B4zMxs>

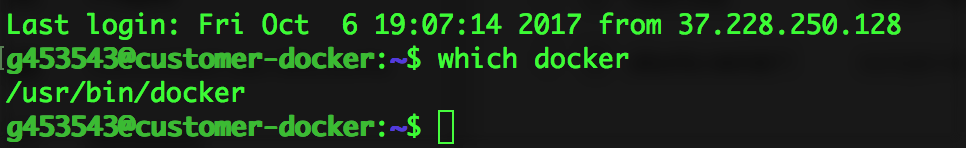
and Docker-P1 lecture video before attempting this Lab.

* 1. Create a new Virtual machine from google compute engine and call it docker-host. For this virtual machine specify custom-image: docker-engine as boot disk. Use the following figure as a guide.
  2. Access the new created VM using GCloud shell in google console:

gcloud compute ssh <instance-name>

* 1. You should be able to see username@vmname in google cloud shell
  2. **If you have docker installed before and running with sudo privilege. Go to Part 2 of this lab sheet.**
  3. Verify that Docker is installed in the VM:

which docker



* if docker is not installed, install it using the following command:

curl -fsSL get.docker.com|sh

* Verity that Docker is running with full privilege without sudo:

sudo docker --version

docker –version

* If you need sudo to run the docker commands, use the following command to run docker commands without sudo:

sudo usermod -aG docker <username>

* For example, the username in the figure above is: g453543, the command will be sudo usermod -aG docker g453543. In your case it should be your student number.
* **Make sure to restart the VM from compute engine after executing the command above.** 
  1. **Connect to the VM and check you can run docker commands without sudo**

## part 2: Learn Docker Containers

Docker is a software container platform. Developers use Docker to ship the software quickly and to eliminate interoperability problem of software dependencies. A container is used to package software in a format that can run isolated on a shared operating system.

* 1. **Visit the docker guide** [**https://docs.docker.com/get-started**](https://docs.docker.com/get-started) **and complete the Tutorials from 1 to 2. (No need to do docker setup as we just did in Part 1 of this lab sheet).**

**(Part 3 to 6 of docker tutorials are advanced tasks require you install docker in your personal computer)**

# PART 3: build a docker container

* 1. Create a GitHub repository and call lab4-image
  2. Clone my repo from <https://github.com/baselm/lab4-base-image.git>
  3. Go inside the folder lab4-base-image

cd lab4-base-image

* 1. build a docker image and call it *mywebapp. Be careful about the dot in the command bellow. It tells docker to read the docker file form the current directory.*

docker build -t mywebapp .

* 1. check the image is added using *docker images* command.

docker images

* 1. Run the mywebapp image as a container

docker run -p 80:5000 -d mywebapp

* 1. Check that the container is running:

docker ps

* 1. The container should be running at port 5000, but we specify to map the traffic from the container port 5000 into the instance port 80.
  2. Visit the instancehttp://external-ip you should be able to see the content of index.html we did in lab3.
  3. Stop the running container using the container-id:
     1. To get the container id do:

docker ps

docker stop <container-id>

# Part 4: pull and run ready made docker image

* 1. Create a new Virtual machine from google compute engine and call it Dev-machine. For this virtual machine specify custom-image: docker-engine as **boot disk**. Access the new created VM using ssh. Use the following figure as a guide.
  2. Inside the dev-machine VM pull mywebapp image using the following command:

docker pull baselm/mywebapp

* 1. Now run the image using

docker run -p 80:5000 -d baselm/mywebapp

* 1. Verify you can access the container from the web browser using the VM external IP
  2. Observe how it is easy to deploy a python application using docker containers.

# Part 5: Challenging Tasks (Optional)

* 1. Create a GitHub repo and call it lab4-repo
  2. Pull the repo into the VM dev-machine
  3. Create a flask application with two routes:
  4. Main app.route(“/”) should return index.html page displaying a short BIO about yourself. You can include images …etc.
  5. App.route(“/contact-me”) should enable the user to send you an email. HTML forms or emailto tag can be used or any other method.

<a href="mailto:name@email.com">Link text</a>

* 1. Create a docker file for your application
  2. Create requirments.txt file
  3. Build the docker image
  4. Run it so you can share your BIO at port 80 of the VM.

# PART 6: Write the Lab report

* 1. Stop the container and all VM you created for this lab.
  2. Stop Cloud SQL instance you created in Lab3 (if you are not using it)
  3. Fill Lab5 report in Google classroom or using this link <https://goo.gl/forms/w75I4ZMS5tQfTxj93>
  4. Demonstrate your work.
  5. This lab should be completed in the lab classroom no remote submission is accepted.