INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Cloud Computing with Google Cloud & Docker

AI Frameworks

Brendan Guillouet

December 3rd, 2018

Institut National des Sciences Appliquées

Introduction

Data Science tools

Tools you know

- Programming : Python. 
- Libraries : Numpy, Pandas, Scikit Learn, Keras, Tensorflow ...
- Development : Jupyter. 

Data Science tools

Tools you know

- Programming : Python. 
- Libraries : Numpy, Pandas, Scikit Learn, Keras, Tensorflow ...
- Development : Jupyter. 

Tools we'll see today :

Data Science tools

Tools you know

- Programming : Python. 
- Libraries : Numpy, Pandas, Scikit Learn, Keras, Tensorflow ...
- Development : Jupyter. 

Tools we'll see today :

- Programming : Python. 
  - Learn how to write script.

Data Science tools

Tools you know

- Programming : Python. 
- Libraries : Numpy, Pandas, Scikit Learn, Keras, Tensorflow ...
- Development : Jupyter. 

Tools we'll see today :

- Programming : Python. 
  - Learn how to write script.
- Cloud services : **Google Cloud**, AWS, Azure. 
  - To use more computation power.

Data Science tools

Tools you know

- Programming : Python.
- Libraries : Numpy, Pandas, Scikit Learn, Keras, Tensorflow ...
- Development : Jupyter.

Tools we'll see today :

- Programming : Python.
  - Learn how to write script.
- Cloud services : **Google Cloud**, AWS, Azure.
  - To use more computation power.
- Docker.
  - To use application easy to replicate and sustainable.

Table of contents

Script Python

Why using script?

Jupyter limits :

- it's an exploration tool
  - but cloud machine are accounted on an hourly base.
- Non-linear workflow.
  - Easy to write messy code.
- Not designed to handle large-scale experiment.
- Not designed for production.
  - Can't be run from terminal, no test procedure.

Why using script?

Jupyter limits :

- it's an exploration tool
  - but cloud machine are accounted on an hourly base.
- Non-linear workflow.
  - Easy to write messy code.
- Not designed to handle large-scale experiment.
- Not designed for production.
  - Can't be run from terminal, no test procedure.

$\Longrightarrow$ Exploration work : *Jupyter*.

Why using script?

Jupyter limits :

- it's an exploration tool
  - but cloud machine are accounted on an hourly base.
- Non-linear workflow.
  - Easy to write messy code.
- Not designed to handle large-scale experiment.
- Not designed for production.
  - Can't be run from terminal, no test procedure.

$\Longrightarrow$ Exploration work : *Jupyter*.

$\Longrightarrow$ Large-scale/production work : Write *Script* !

Script execution

File *script.py*

```
a = 5
b = 3
c = a + b
print("The answer is %d" %c)
```

Terminal

```
bguillou $> python script.py
bguillou $> The answer is 8
```

TP - First Part

Write two scripts :

- **learning.py :** to learn a model, save it in the *model* directory, save results in the *results* directory and
- **prediction.py :** to generate prediction ans save it in the *results* directory

on *CatsVsDogs* data.

$\Longrightarrow$ Ensure that the **complete workflow is working** locally before pushing the code on the instance.

Librairie *Argparse*

File *script.py*

```python
import argparse

parser = argparse.ArgumentParser()
parser.add_argument('--a', type=int, default=5)
parser.add_argument('--b', type=int, default=3)

args = parser.parse_args()

c= args.a + args.b
print("The answer is %d" %c)
```

Terminal

```
bguillou $> python script.py
bguillou $> The answer is 8
bguillou $> python script.py --a 4
bguillou $> The answer is 7
bguillou $> python script.py --a 4 --b 2
bguillou $> The answer is 7
```

Librairie *pickle*

File *learning.py*

```
import pickle
...
results = {"learning_time" : lt, "accuracy" : acc}
pickle.dump(results, open("/User/bguillouet/data/results.pkl","wb"))
```

File *explore_results.py*

```
import pickle
results = pickle.load(open("/User/bguillouet/data/results.pkl","rb"))
print(results)
```

Terminal

```
bguillou $> ls data/
bguillou $>
bguillou $> python learning.py
bguillou $> ls data/
bguillou $> results.pkl
bguillou $> python explore_results.py
bguillou $> {"learning_time" : lt, "accuracy" : acc}
```

Librairie *hashlib*

python *script.py*

```
import argparse
import hashlib
import pickle

parser = argparse.ArgumentParser()
parser.add_argument('--a', type=int, default=5)
parser.add_argument('--b', type=int, default=3)
parser.add_argument('--type_op', type =str, default="addition")

args = parser.parse_args()

if args.type_op == "addition":
    c= args.a + args.b
else:
    c= args.a * args.b

results = var(args)
print("Argument dictionary: "+ str(results))
results.update({"results" : c})
print("Argument dictionary with score: "+ str(results))


args_str = "_".join([k + ":" + str(v) for k, v in sorted(results.items(), key=lambda x : x[0])])
print("Argument string: "+ args_str)

id_str = hashlib.md5(args_str.encode("utf8")).hexdigest()
print(id_str)

result_file = "/User/bguillouet/data/" + id_str + ".pkl"

pickle.dump(results, open(results_file, "wb"))
```

Librairie *hashlib*

```
bguillou $> python script.py
bguillou $> {"a":5, "b":3, "type_op":"addition"}
bguillou $> {"a":5, "b":3, "type_op":"addition", "results":8}
bguillou $> a:5_b:3_type_op:addition_results:8
bguillou $> aZezEzj7jhZ8793DeefdjZ9
bguillou $> ls data/
bguillou $> aZezEzj7jhZ8793DeefdjZ9.pkl
```

Google Cloud

Google Cloud

Suite of cloud computing services with more than 90 products.

- Power computation , Database, AI, Networking, Security *etc*..

Google Cloud

Suite of cloud computing services with more than 90 products.

- **Power computation**, Database, AI, Networking, Security *etc.*.

Google Cloud

Suite of cloud computing services with more than 90 products.

- **Power computation**, Database, AI, Networking, Security *etc.*.
  - $\Longrightarrow$ Google Cloud Engine

Google Cloud

Suite of cloud computing services with more than 90 products.

- **Power computation**, Database, AI, Networking, Security *etc.*.
  - $\Longrightarrow$ Google Cloud Engine

Why not IA tools ?

- *AutoML* or *ML Engine* are tools that provide solution for non-expert.

Google CLoud

Suite of cloud computing services with more than 90 products.

- **Power computation**, Database, AI, Networking, Security *etc.*.
  - $\Longrightarrow$ Google Cloud Engine

Why not IA tools ?

- *AutoML* or *ML Engine* are tools that provide solution for non-expert.

What do I need to know to use Google CLoud Engine ?

- Basic knowledge of Google cloud interface.
- Basic use of terminal command (no graphic interface).
- *gcloud SDK*.
  https ://cloud.google.com/sdk/docs/quickstart-debian-ubuntu

Gcloud - Command

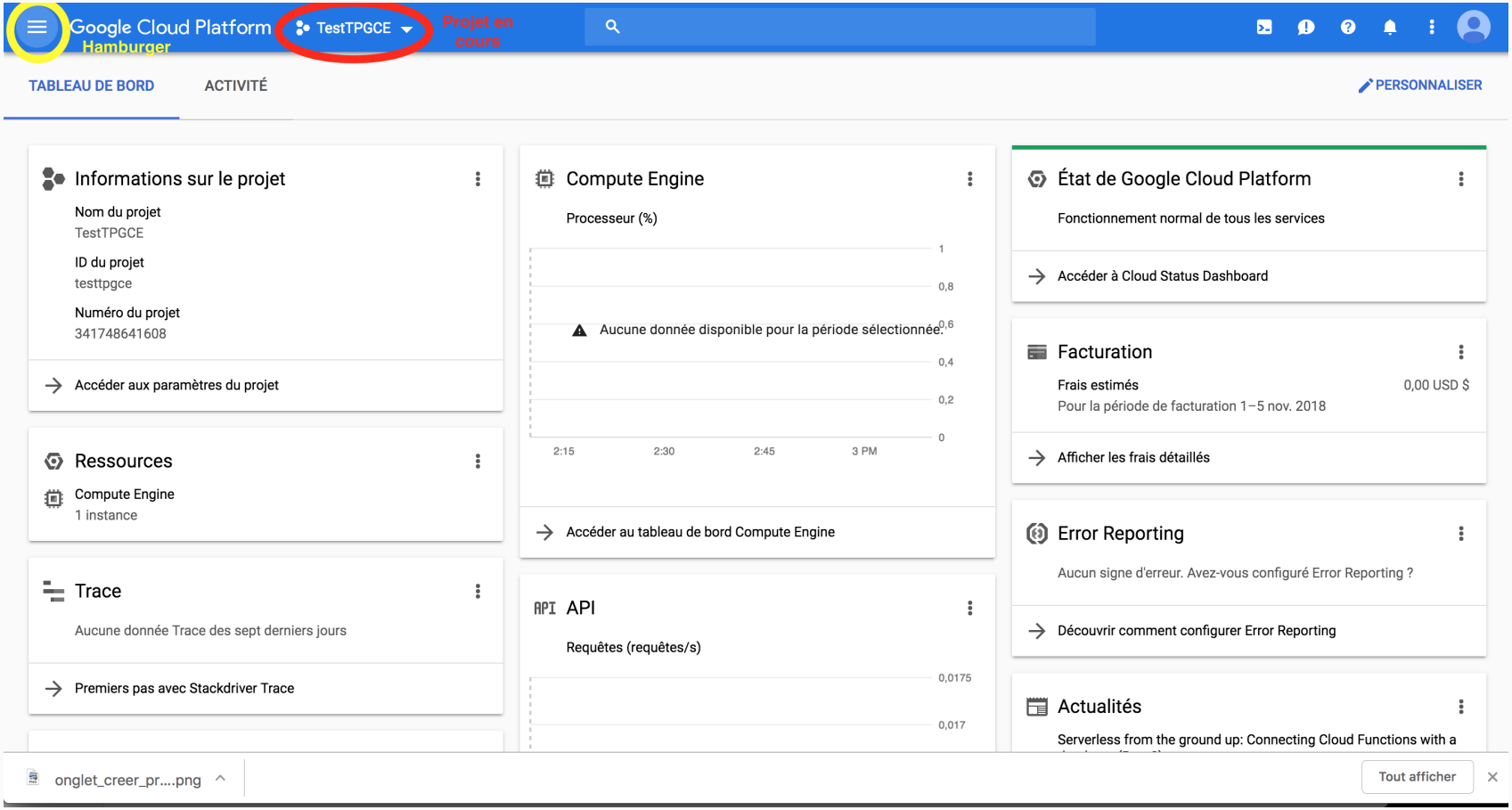
Command line tool which allow to manage VM instance.

- *gcloud init*. To be used at first utilization.
- *gcloud compute instances start/stop/delete instance_name*. To start/stop/remove instance.
- *gcloud compute scp --recurse CopyFrom CopyTo*
  - *--recurse* (optional) : To be used if directory is copied.
  - *CopyFrom* : Location of the file or directory to be copied.
  - *CopyTo* : Location of the directory were the file or directory will be copied.
  - *Syntax :* [[[USER@]INSTANCE :]DIR] . To send file on the instance.

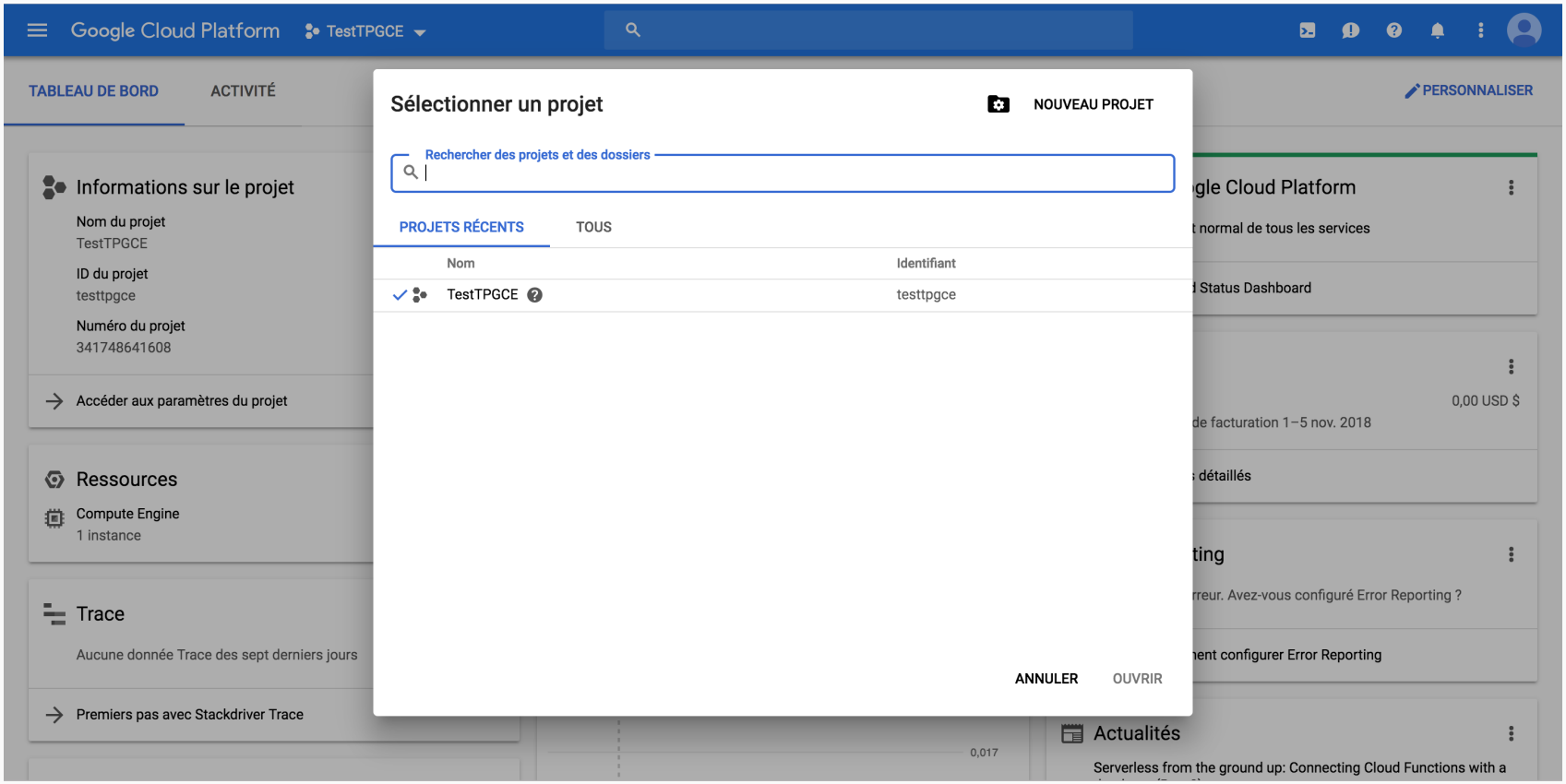
Gcloud - Command

- *gcloud compute ssh --ssh-key-file LocationOfSShKey --zone europe-west1-b*. To set ssh-connection to the instance.
- *gcloud compute ssh --command 'COMMAND'*. To execute command on the instance.
  - *gcloud compute ssh --command 'mkdir data'*
  - *gcloud compute ssh --command 'python learning.py'*

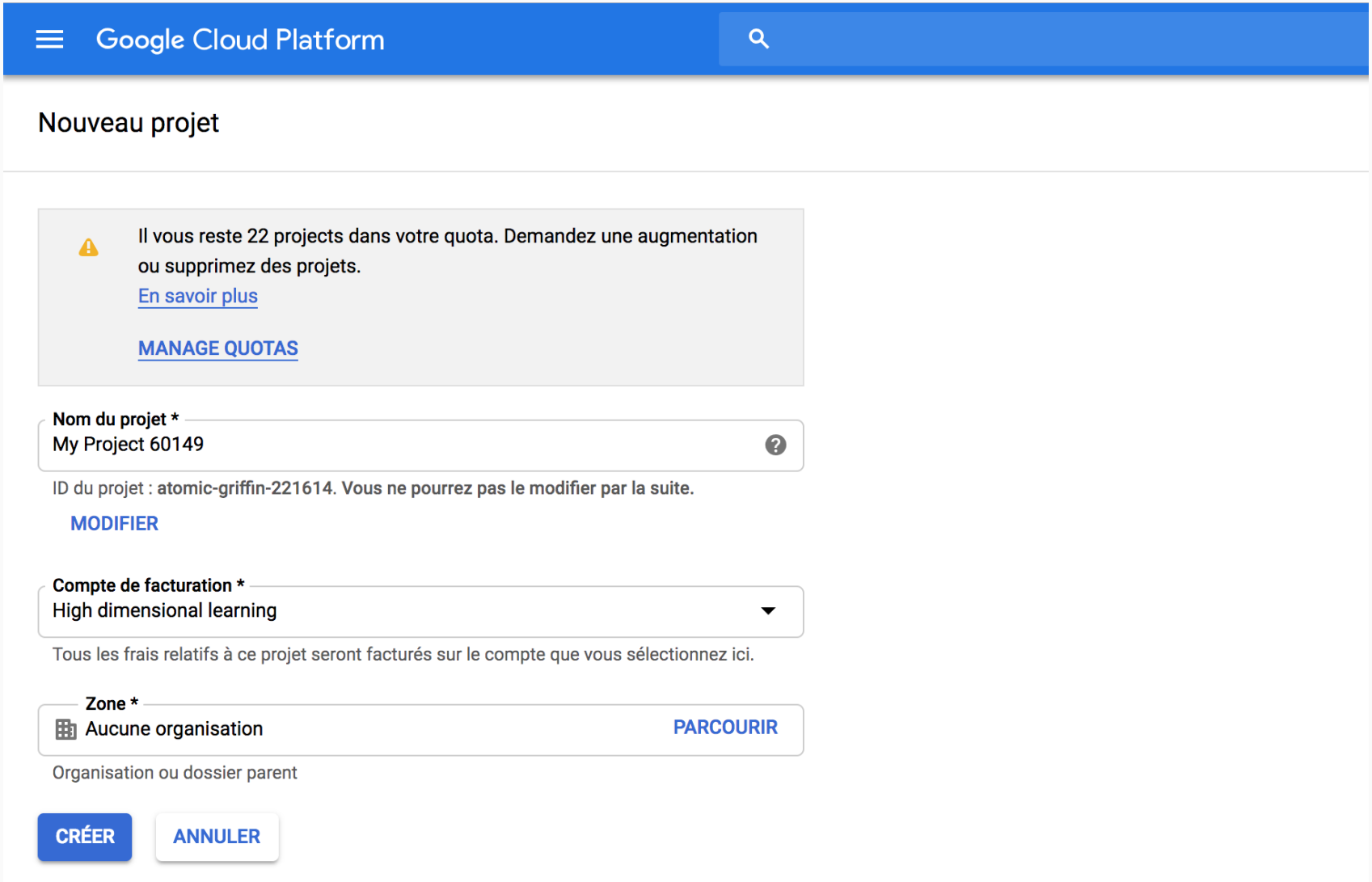
GCE : Accueil


Google Cloud Platform
Hamburger
TestTPGCE
Projet en cours
TABLEAU DE BORD
ACTIVITÉ
PERSONNALISER
Informations sur le projet
Nom du projet
TestTPGCE
ID du projet
testtpgce
Numéro du projet
341748641608
Accéder aux paramètres du projet
Ressources
Compute Engine
1 instance
Trace
Aucune donnée Trace des sept derniers jours
Premiers pas avec Stackdriver Trace
Compute Engine
Processeur (%)
1
0,8
0,6
0,4
0,2
0
Aucune donnée disponible pour la période sélectionnée.
2:15
2:30
2:45
3 PM
Accéder au tableau de bord Compute Engine
API
API
Requêtes (requêtes/s)
0,0175
0,017
État de Google Cloud Platform
Fonctionnement normal de tous les services
Accéder à Cloud Status Dashboard
Facturation
Frais estimés
0,00 USD $
Pour la période de facturation 1–5 nov. 2018
Afficher les frais détaillés
Error Reporting
Aucun signe d'erreur. Avez-vous configuré Error Reporting ?
Découvrir comment configurer Error Reporting
Actualités
Serverless from the ground up: Connecting Cloud Functions with a
onglet_creer_pr....png
Tout afficher

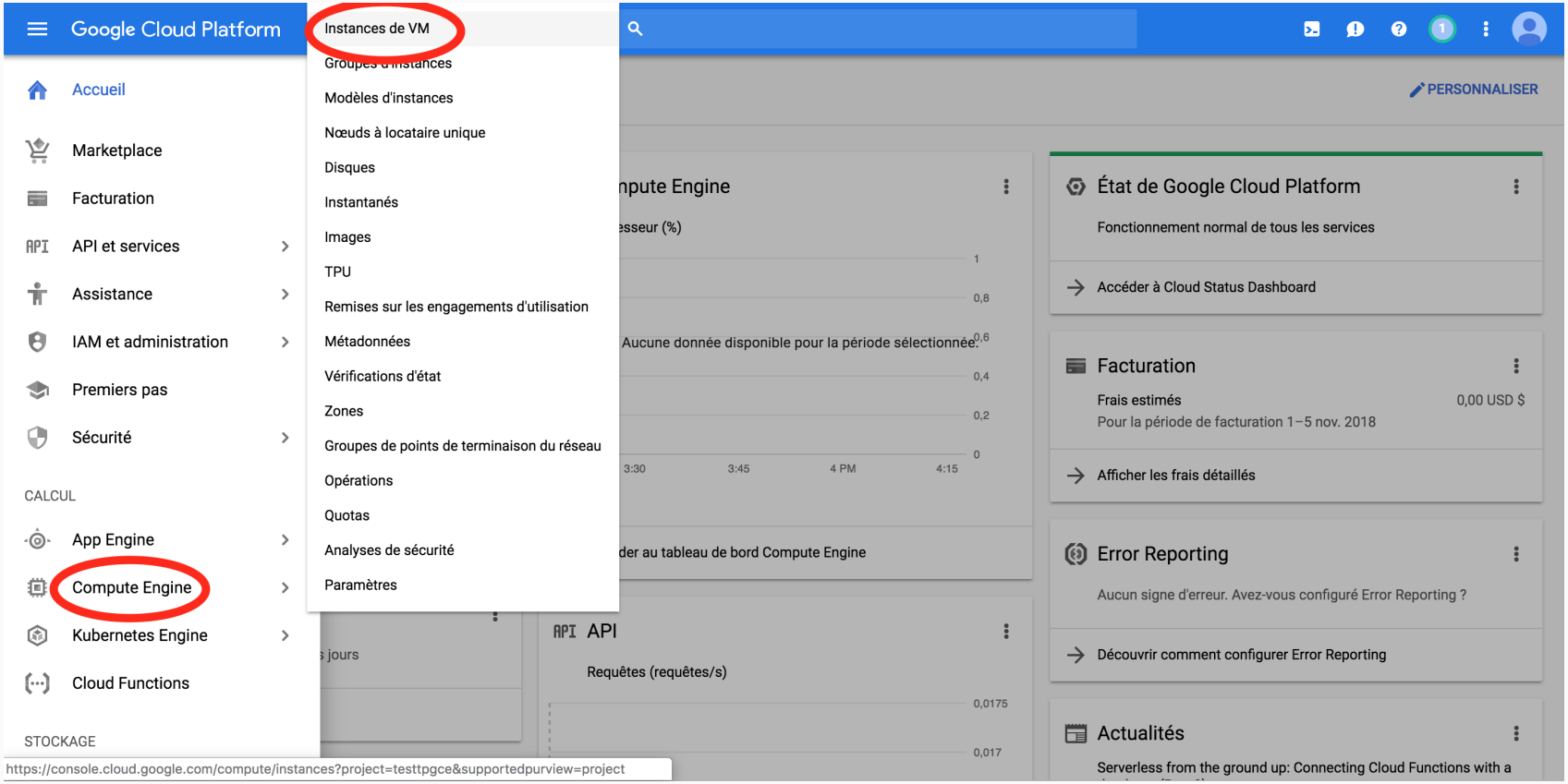
GCE : Sélection Projet



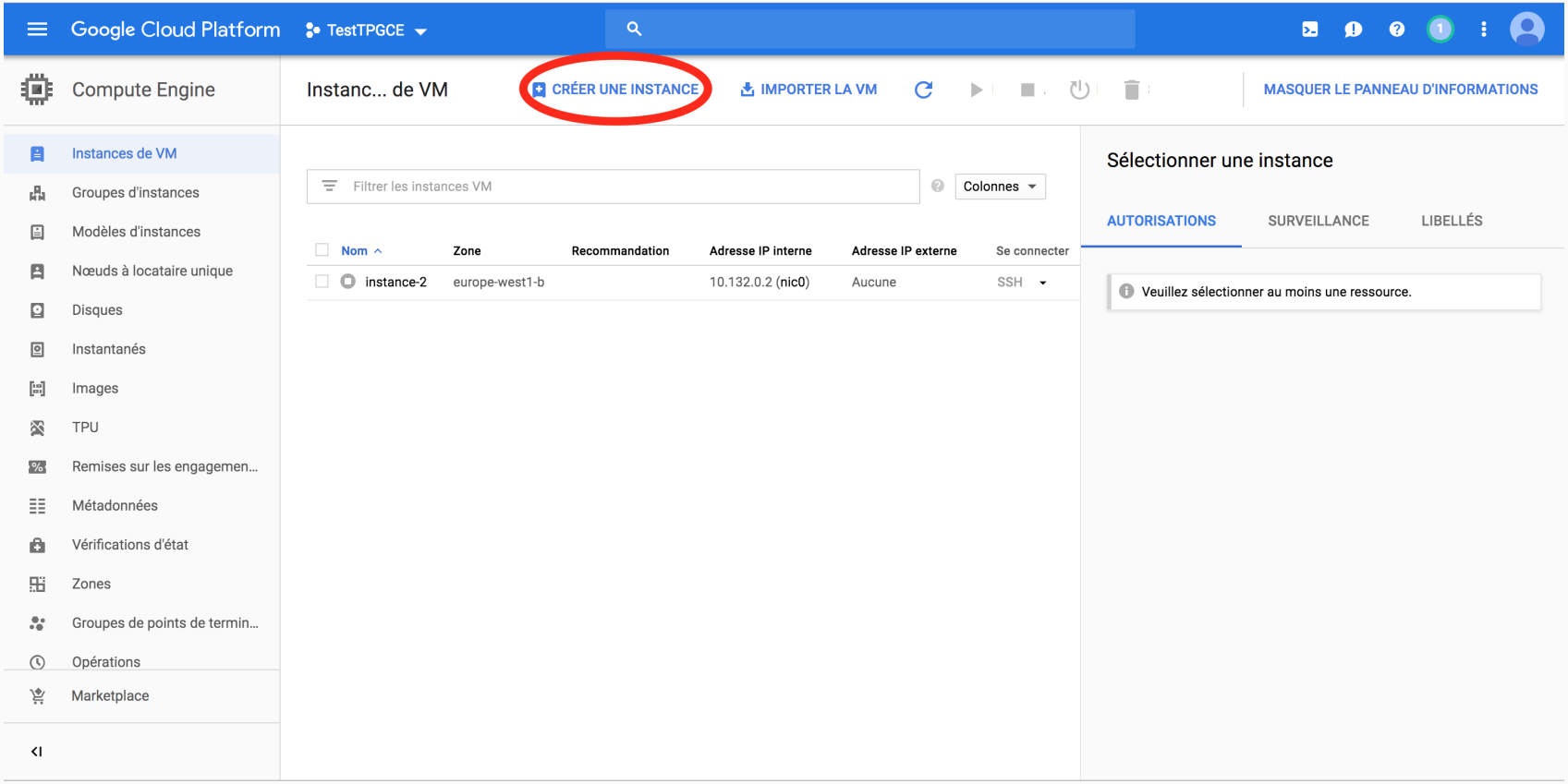
GCE : Nouveau Projet



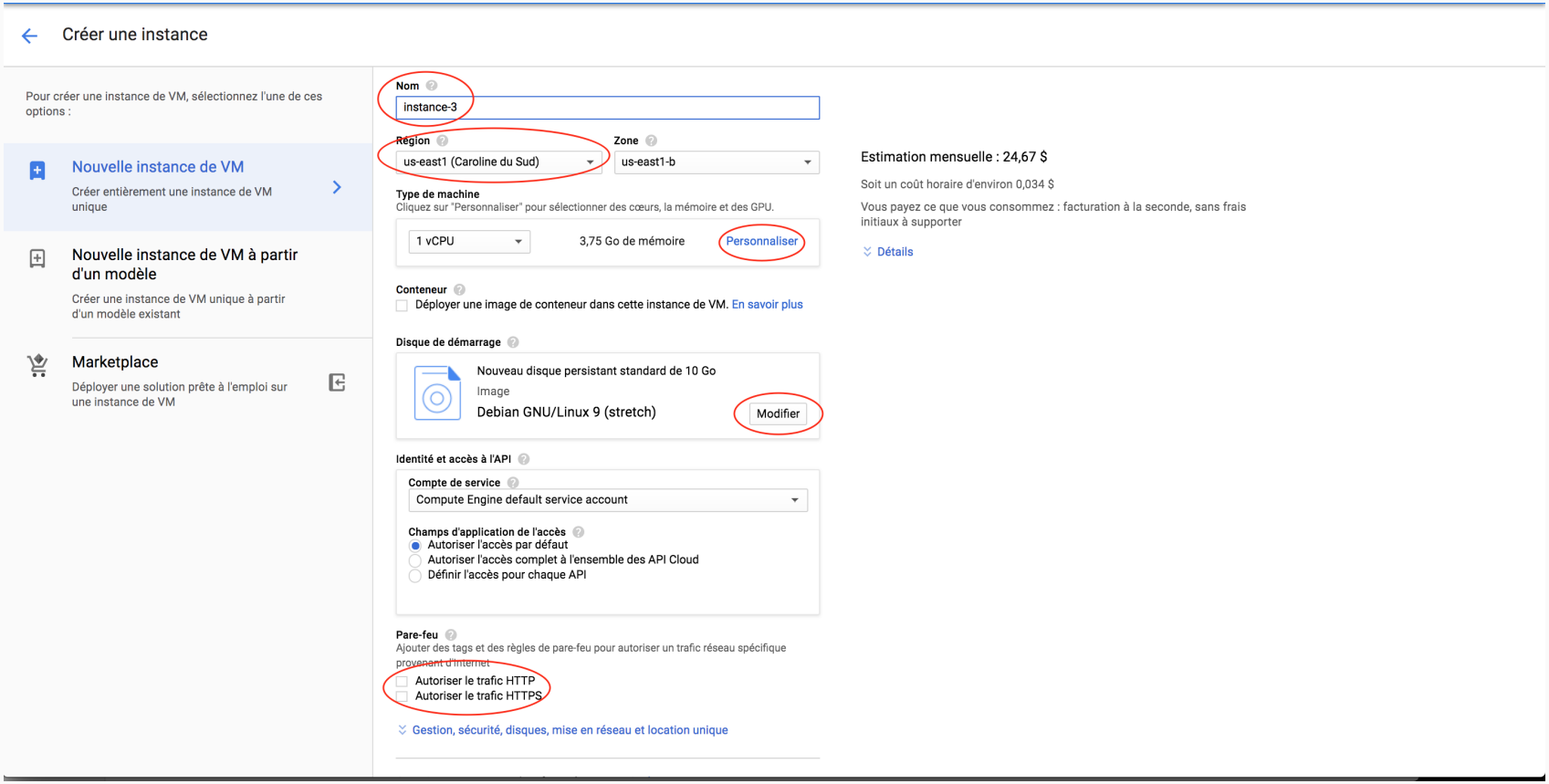
GCE : Menu Principal (Hamburger)



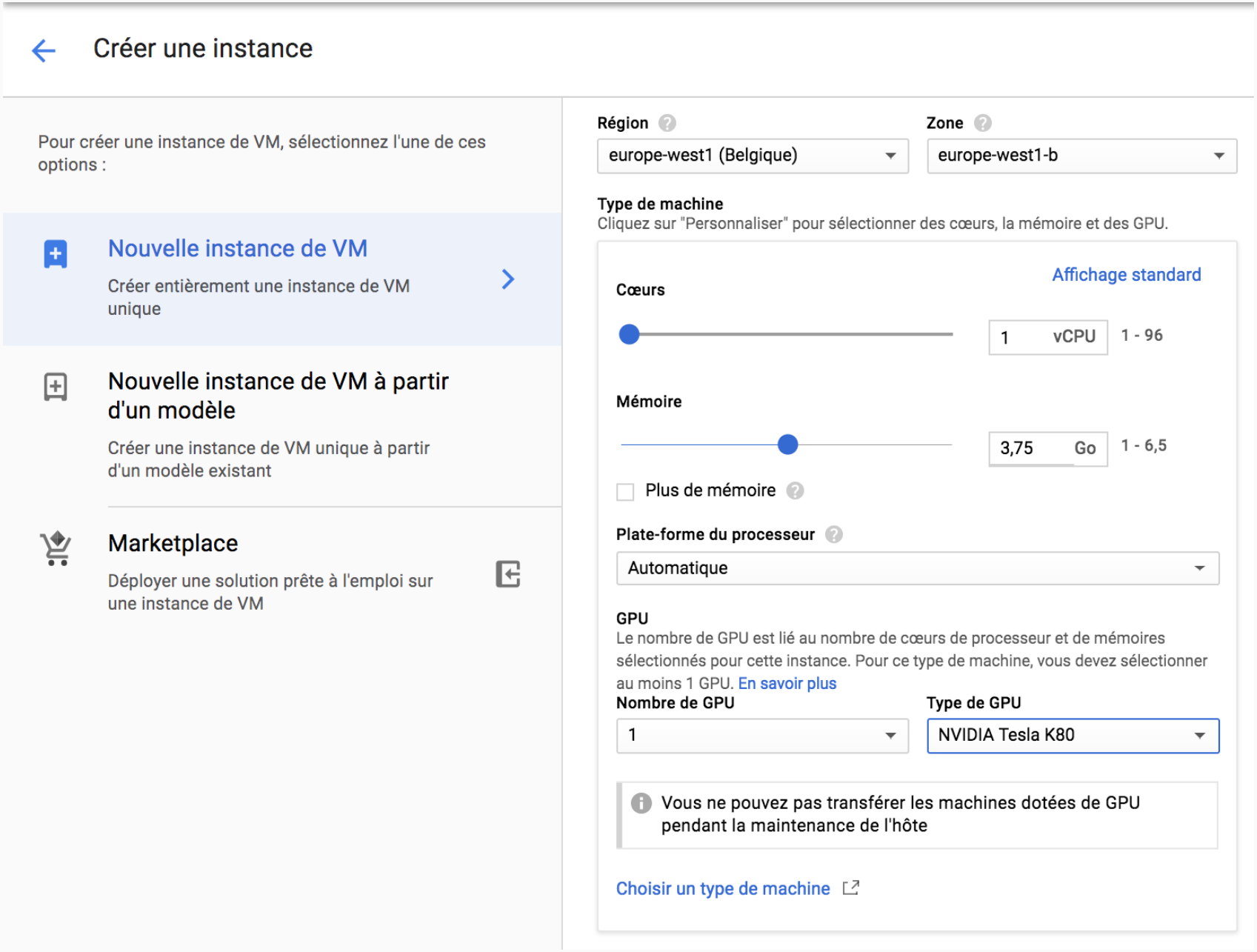
GCE : Instance VM



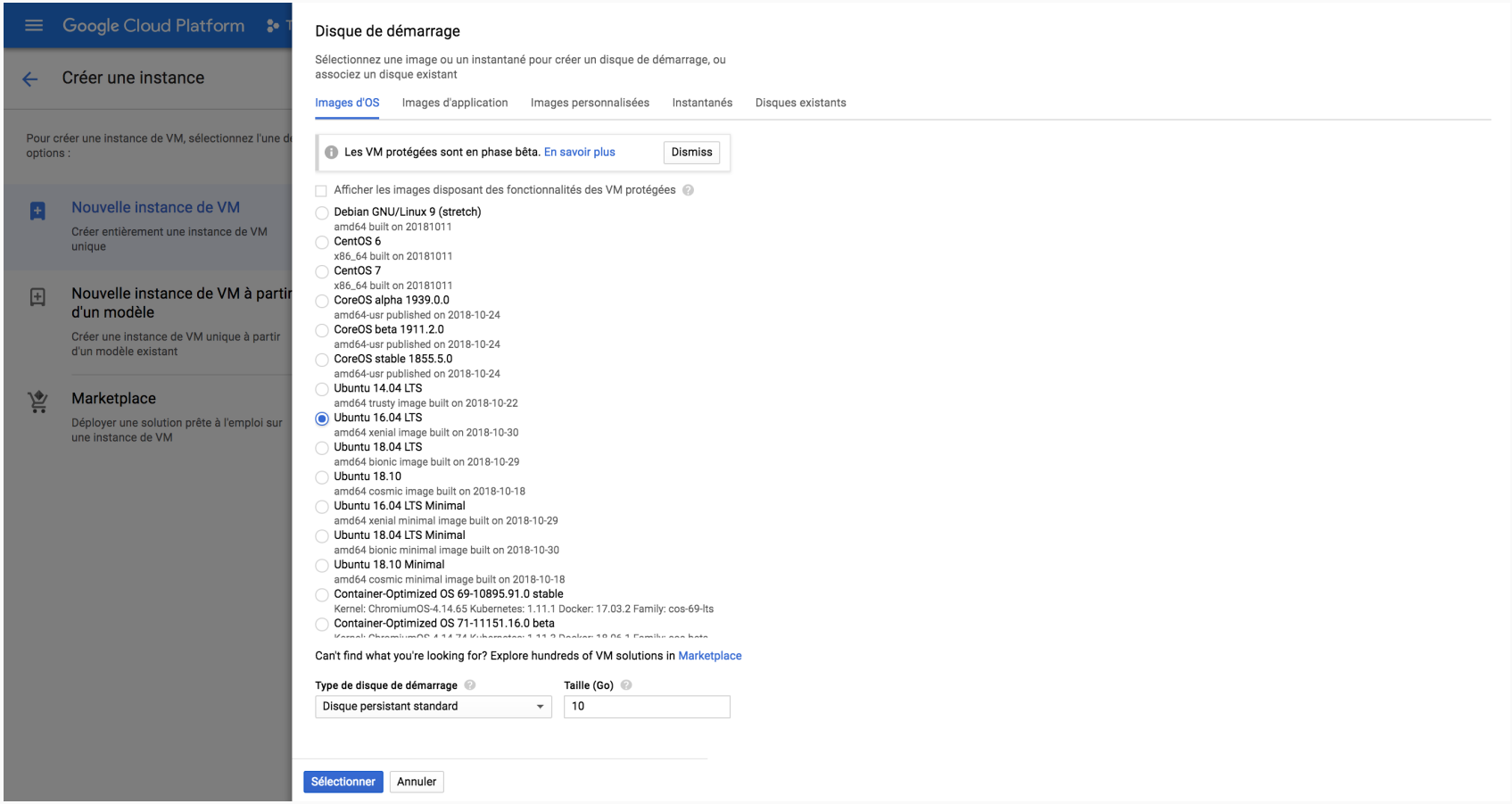
GCE : Créer une Instance VM



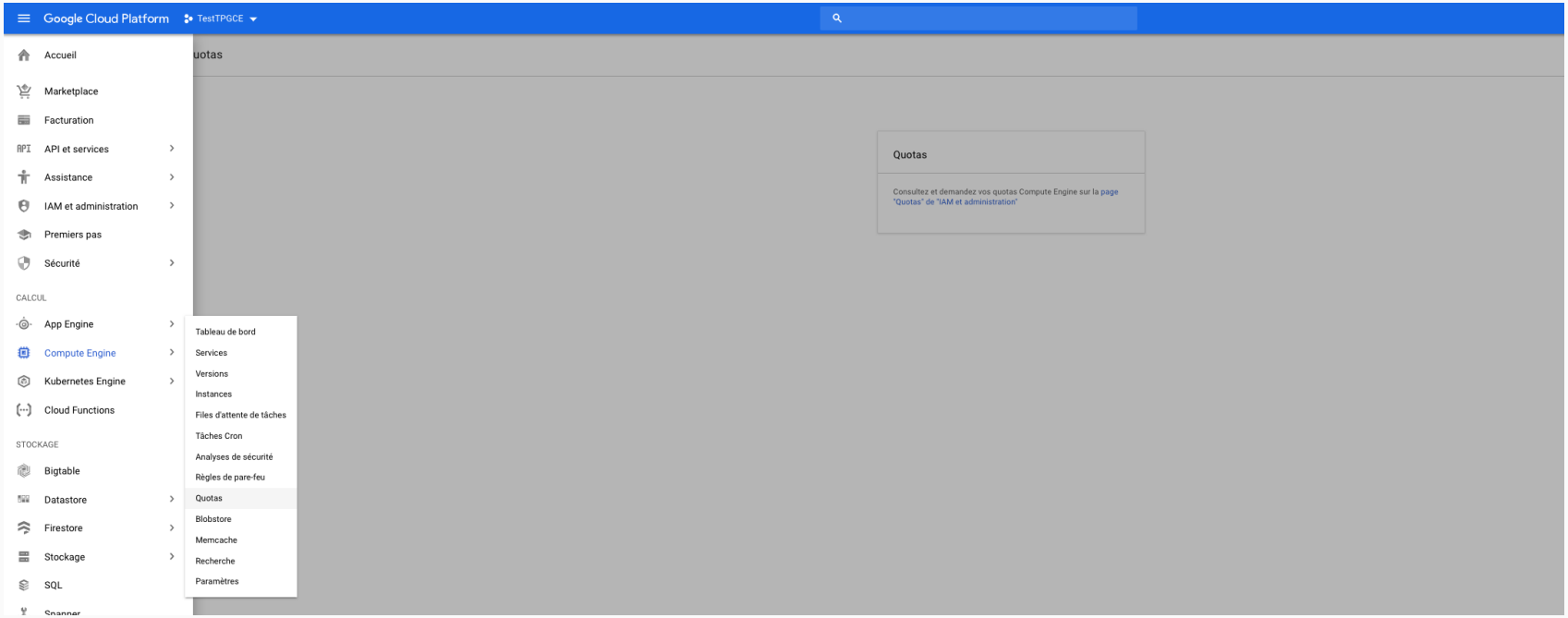
GCE : Type de machine



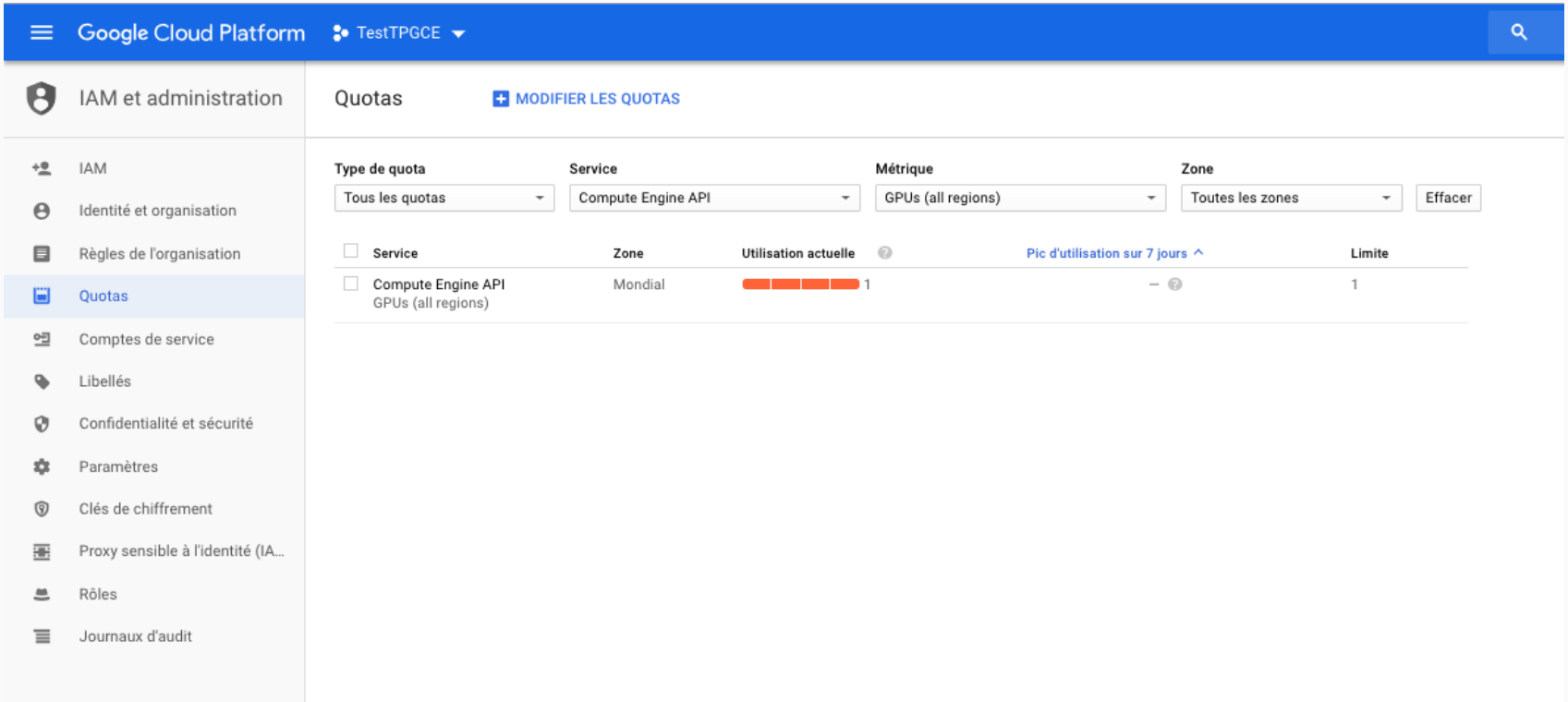
GCE : Disque démarrage



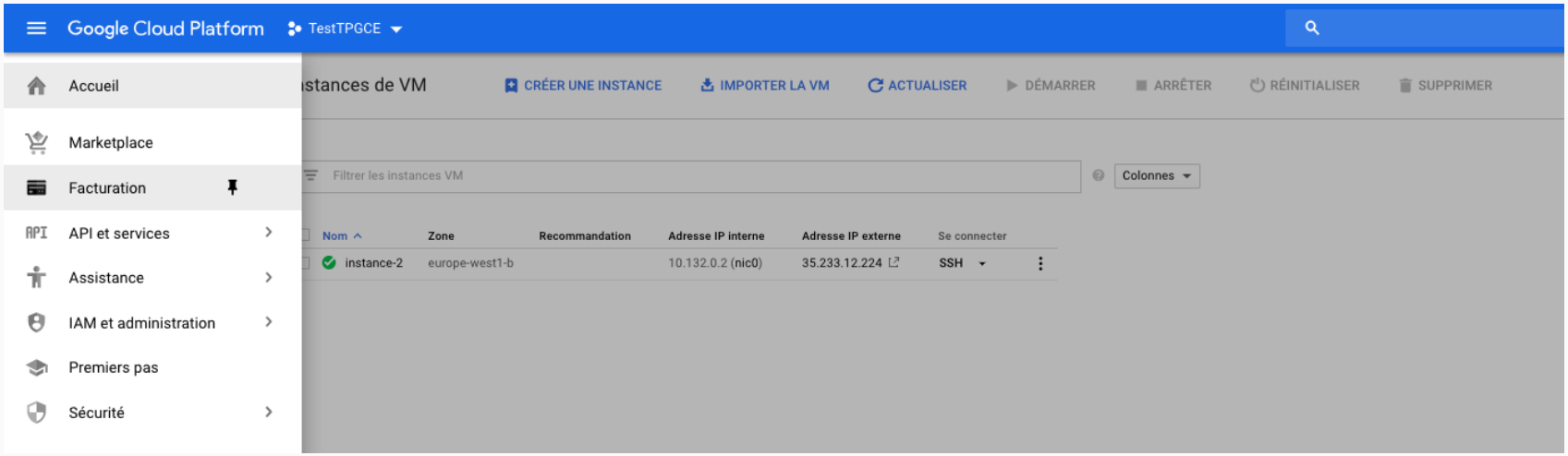
GCE : Quotas


Google Cloud Platform
TestTPGCE
Accueil
Marketplace
Facturation
API et services
Assistance
IAM et administration
Premiers pas
Sécurité
CALCUL
App Engine
Compute Engine
Kubernetes Engine
Cloud Functions
STOCKAGE
Bigtable
Datastore
Firestore
Stockage
SQL
Tableau de bord
Services
Versions
Instances
Files d'attente de tâches
Tâches Cron
Analyses de sécurité
Règles de pare-feu
Quotas
Blobstore
Memcache
Recherche
Paramètres
uotas
Quotas
Consultez et demandez vos quotas Compute Engine sur la page "Quotas" de "IAM et administration"

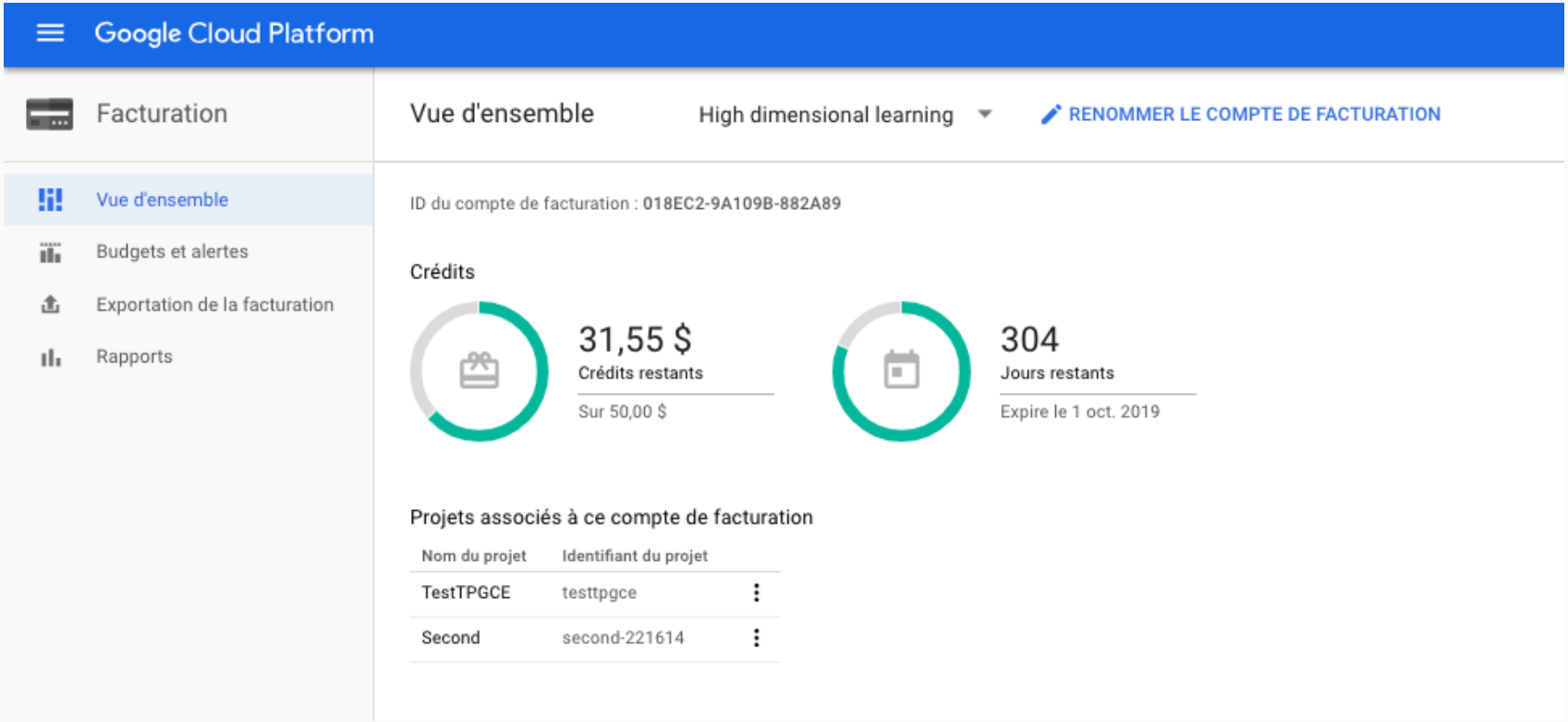
GCE : Quotas


Google Cloud Platform
TestTPGCE
IAM et administration
Quotas
MODIFIER LES QUOTAS
IAM
Identité et organisation
Règles de l'organisation
Quotas
Comptes de service
Libellés
Confidentialité et sécurité
Paramètres
Clés de chiffrement
Proxy sensible à l'identité (IA...
Rôles
Journaux d'audit
Type de quota
Tous les quotas
Service
Compute Engine API
Métrique
GPUs (all regions)
Zone
Toutes les zones
Effacer
Service
Zone
Utilisation actuelle
Pic d'utilisation sur 7 jours
Limite
Compute Engine API
GPUs (all regions)
Mondial
1
–
1

GCE : Facturation



GCE : Facturation



- Facturation arrive en différé.
- 1 projet par compte de facturation.
- 1 machine GPU par projet.
- Capacité de la machine illimité.

Install framework

- Python
- Cuda
- Docker
- Nvidia-docker

bash_script for installation in *utils/bash_utils_on_gpu*.

Pipeline d'execution

Pipeline in practice

1. Write script in your *local* machine.
2. Turn you *instance on*.
3. Build environment(if first used).
4. Send latest version of your code to the instance.
5. Send data to the instance.

7. Run the script on the instance.
8. Copy the results you want to analyze from the instance to your local machine.

10. Turn your instance off.

Pipeline in practice

1. Write script in your *local* machine.
2. Turn you *instance on*.
3. Build environment(if first used).
4. Send latest version of your code to the instance.
5. Send data to the instance.
6. Run container.
7. Run the script on the instance.
8. Copy the results you want to analyze from the instance to your local machine.
9. Stop and remove container.
10. Turn your instance off.

Pipeline - Command

1. Write script in your *local* machine.
2. *gcloud compute instances start ..*
3. *gcloud compute ssh --command 'mkdir data'*
4. *gcloud compute scp script.py bguillou@instance-gpu :/home/*
5. *gcloud compute scp --recurse data bguillou@instance-gpu :/home/*

7. *gcloud compute ssh --command 'python script.py –a 3'*
8. *gcloud compute scp --recurse bguillou@instance-gpu :/home/results/ /home/*

10. *gcloud compute instances stop ..*

Pipeline - Command

1. Write script in your *local* machine.
2. *gcloud compute instances start ..*
3. *gcloud compute ssh --command 'mkdir data'*
4. *gcloud compute scp script.py bguillou@instance-gpu :/home/*
5. *gcloud compute scp --recurse data bguillou@instance-gpu :/home/*
6. *gcloud compute ssh --command 'sudo nvidia-docker run ...'*
7. *gcloud compute ssh --command 'python script.py –a 3'*
8. *gcloud compute scp --recurse bguillou@instance-gpu :/home/results/ /home/*
9. *gcloud compute ssh --command 'sudo nvidia-docker container stop ...'*
10. *gcloud compute instances stop ..*

Pipeline - Command

1. Write script in your *local* machine.
2. *gcloud compute instances start ..*
3. *gcloud compute ssh --command 'mkdir data'*
4. *gcloud compute scp script.py bguillou@instance-gpu :/home/*
5. *gcloud compute scp --recurse data bguillou@instance-gpu :/home/*
6. *gcloud compute ssh --command 'sudo nvidia-docker run ...'*
7. *gcloud compute ssh --command 'python script.py –a 3'*
8. *gcloud compute scp --recurse bguillou@instance-gpu :/home/results/ /home/*
9. *gcloud compute ssh --command 'sudo nvidia-docker container stop ...'*
10. *gcloud compute instances stop ..*

Too many possibility to make a mistake.

Pipeline - Command

1. Write script in your *local* machine.
2. *gcloud compute instances start ..*
3. *gcloud compute ssh --command 'mkdir data'*
4. *gcloud compute scp script.py bguillou@instance-gpu :/home/*
5. *gcloud compute scp --recurse data bguillou@instance-gpu :/home/*
6. *gcloud compute ssh --command 'sudo nvidia-docker run ...'*
7. *gcloud compute ssh --command 'python script.py –a 3'*
8. *gcloud compute scp --recurse bguillou@instance-gpu :/home/results/ /home/*
9. *gcloud compute ssh --command 'sudo nvidia-docker container stop ...'*
10. *gcloud compute instances stop ..*

Too many possibility to make a mistake.

$\Longrightarrow$ Pipeline

Pipeline

We wille use a tools compose of 3 *python script* and a *.yml* file :

- **conf.yml** : This file works like a dictionary which contain global variables such as location of directory.
- **instances.py** : This script defines a Python class *InstanceManager* which encapsulate calls to *gcloud*. For exemple *list()* function of this class call this command in terminal :

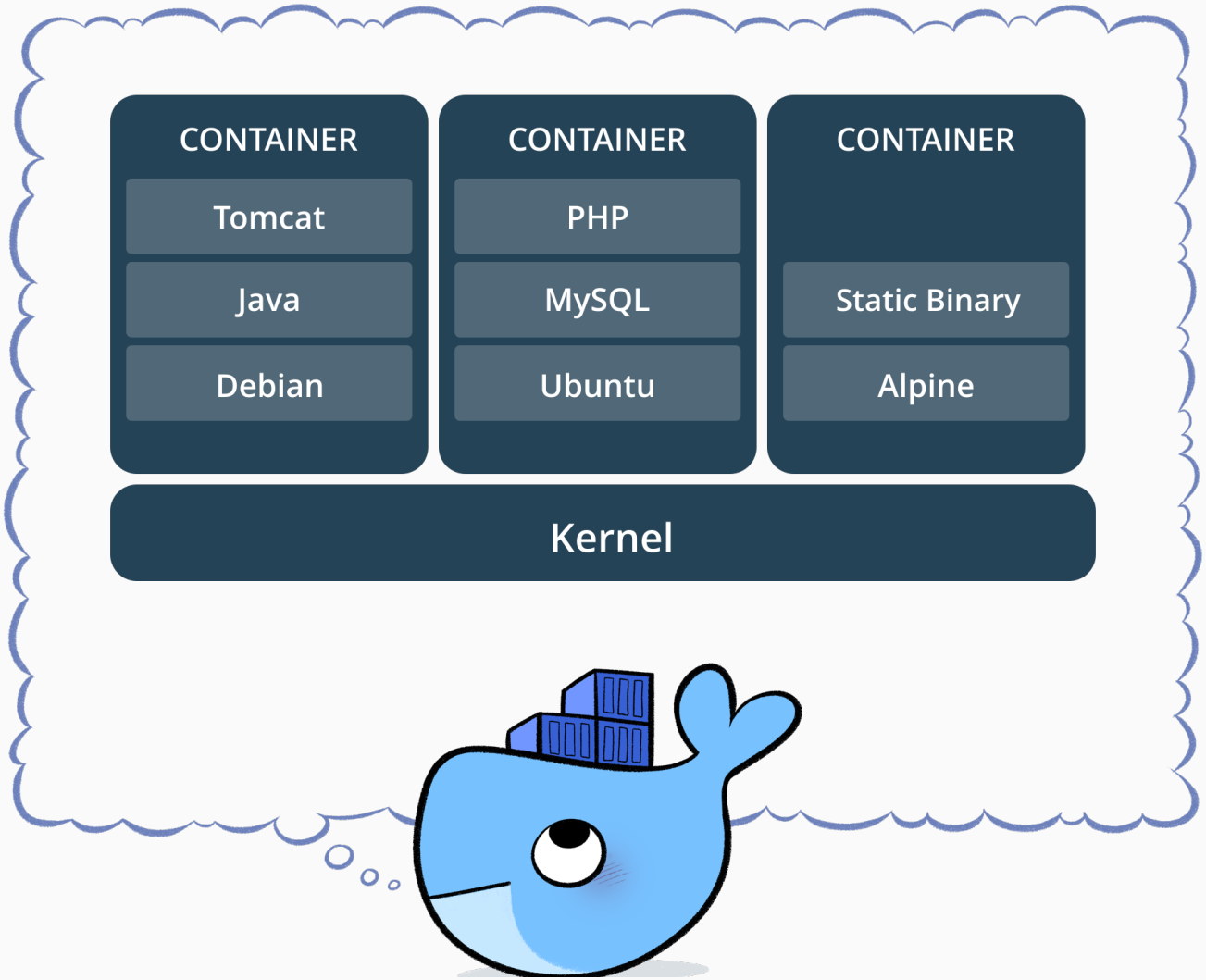
```
gcloud compute instances list
```

- **project.py** This script defines a Python class *ProjectManager* which contains specific functions to manage your project.
  For example, the *update_data(self, zip_file)* function allows to send a *zip_file* from your *data* directory from your local machine to the *data* directory on your instance.
- **main.py** This is a python script which contains all the command to execute your pipeline.

Docker

What is Docker?

Docker enables to build Virtual machines which are easy to re-create on different compute environment.



Why should I use docker AS A DATA SCIENTIST ?

- **Reproducibility** : wrapping all you environment in a Docker container ensure the possibility to recreate your environment and makes your work more accessible.
- **Portability of your compute environment** To move your code and your model easily on machine with more computational power.
- **Enlarge your possibility** : being comfortable with Docker can allow you to use various solution available with docker.

Docker terminology

- **Image** : Its like a turned-off VM which contains the tools you want. Ex : Ubuntu + TensorFlow with Nvidia Drivers and a running Jupyter Server.
- **Container** : Is an instantiation of an image. You can have multiple copies of the same image running.
- **Dockerfile** : Recipe for creating an Image.
- **DockerHub / Image Registry** : Place where people/organization can post public (or private) docker images to facilitate collaboration and sharing.

https ://hub.docker.com/

What we'll do in this TP

- Write a *Dockerfile* which is based on the official *Tensorflow Dockerfile* available on *DockerHub*.
- Use the *Dockerfile* to build a *image*.
- Launch *container* with different option from the build *image*.

Dockerfile

Dockerfile of the image we'll build :

```
FROM tensorflow/tensorflow:latest-devel-gpu-py3
RUN apt-get update && apt-get install -y
python-opencv python-tk vim
RUN pip install h5py keras pytest scikit-image
seaborn tqdm  gensim
```

- **FROM** : Specifies the base image you want to build on top of. Docker will look in your local environment for the image you called and if it cannot find it locally it will search it in *DockerHub*.
- **RUN** : Is followed by normal commands that would be directly run on terminal to install librairies or framework.

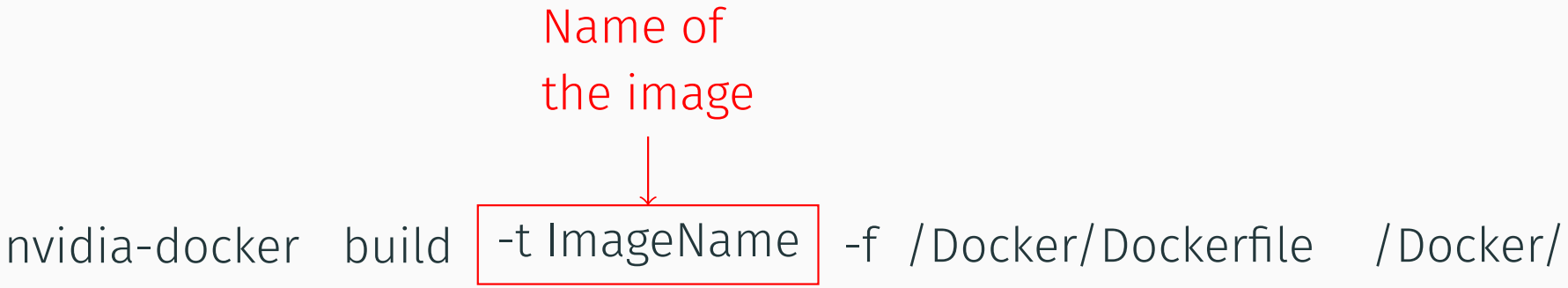
Build Image

Run the BUILD command in order to build your **IMAGE**.

```
nvidia-docker build -t ImageName -f /Docker/Dockerfile /Docker/
```

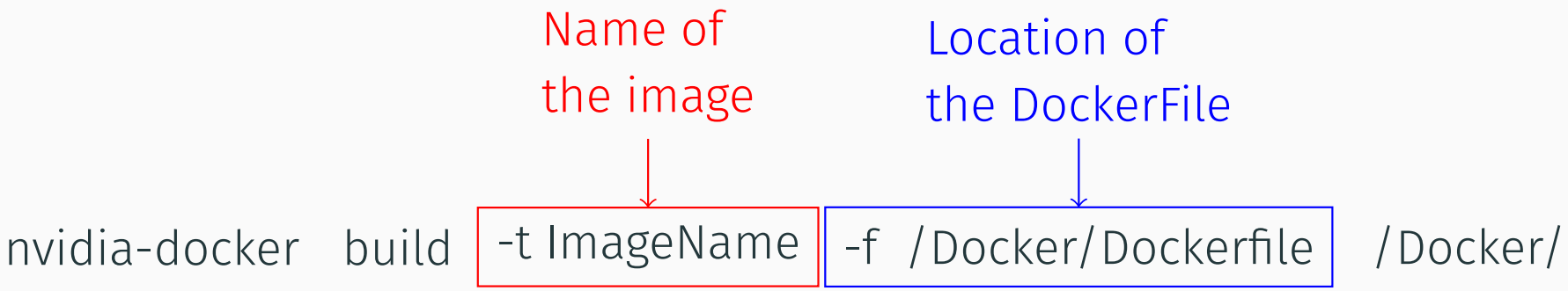
Build Image

Run the BUILD command in order to build your **IMAGE**.



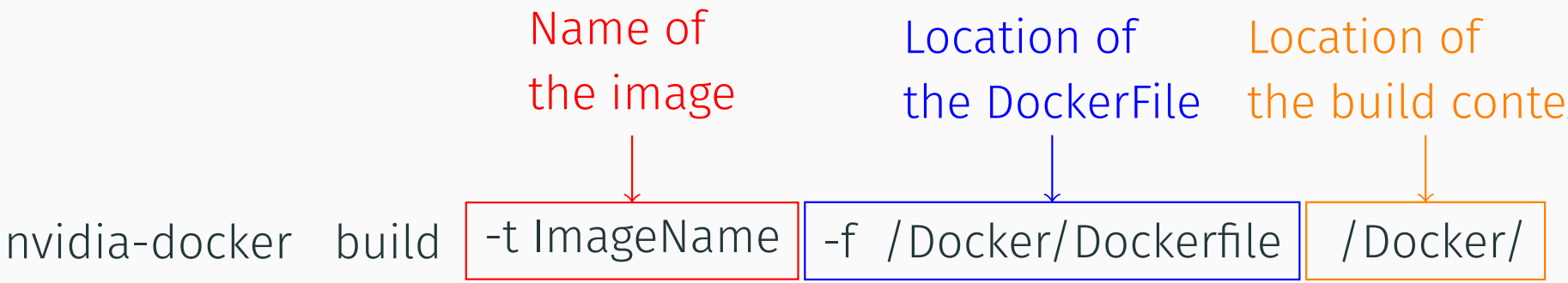
Build Image

Run the BUILD command in order to build your **IMAGE**.



Build Image

Run the BUILD command in order to build your **IMAGE**.



The buid context is the location of the folder to which the **ADD** statement will reference. This means that all external files require by the *Dockerfile* will be located here.

Run Container

Run the RUN command in order to run your **CONTAINER**.

```
nvidia-docker run -it -name ContainerName
    -v ~/CatsVsDogs/ :/root/CatsVsDogs ImageName
```

Run Container

Run the RUN command in order to run your **CONTAINER**.

Mode in which
the container
wil be launched

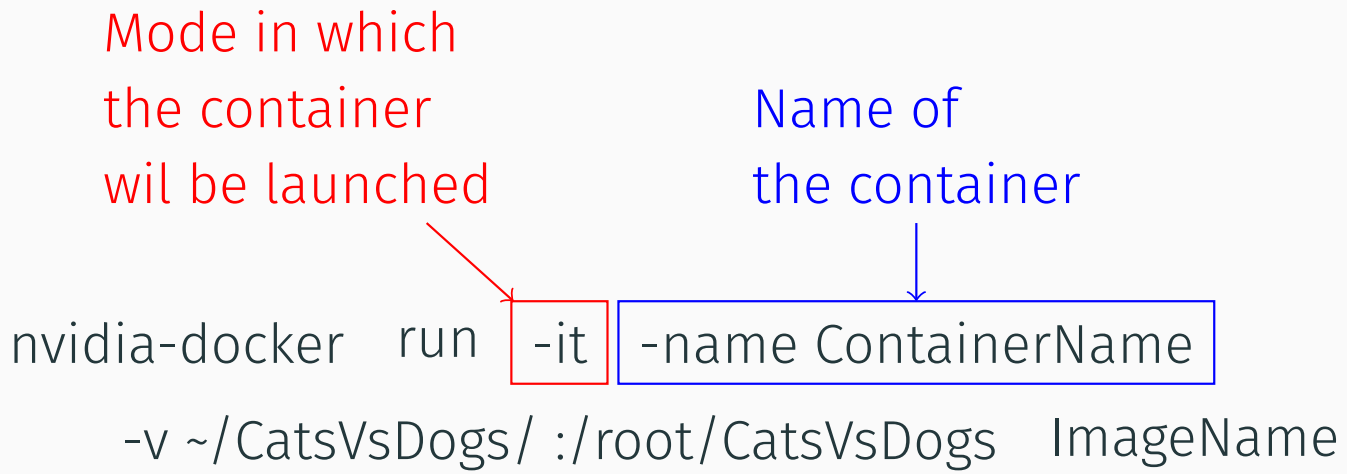
```
nvidia-docker  run  -it  -name ContainerName
    -v ~/CatsVsDogs/ :/root/CatsVsDogs  ImageName
```

Mode of the container could be :

- -it : interactive mode,
- -dt : detached mode.

Run Container

Run the RUN command in order to run your **CONTAINER**.

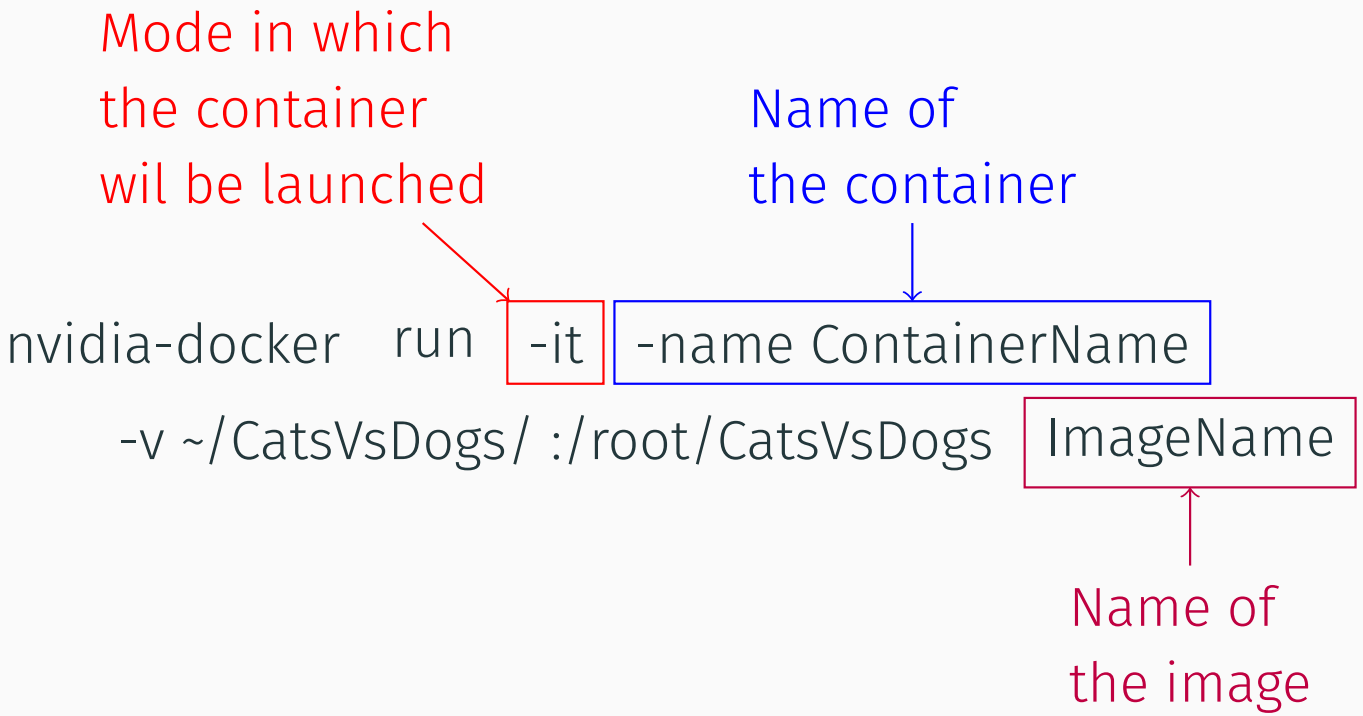


Mode of the container could be :

- -it : interactive mode,
- -dt : detached mode.

Run Container

Run the RUN command in order to run your **CONTAINER**.

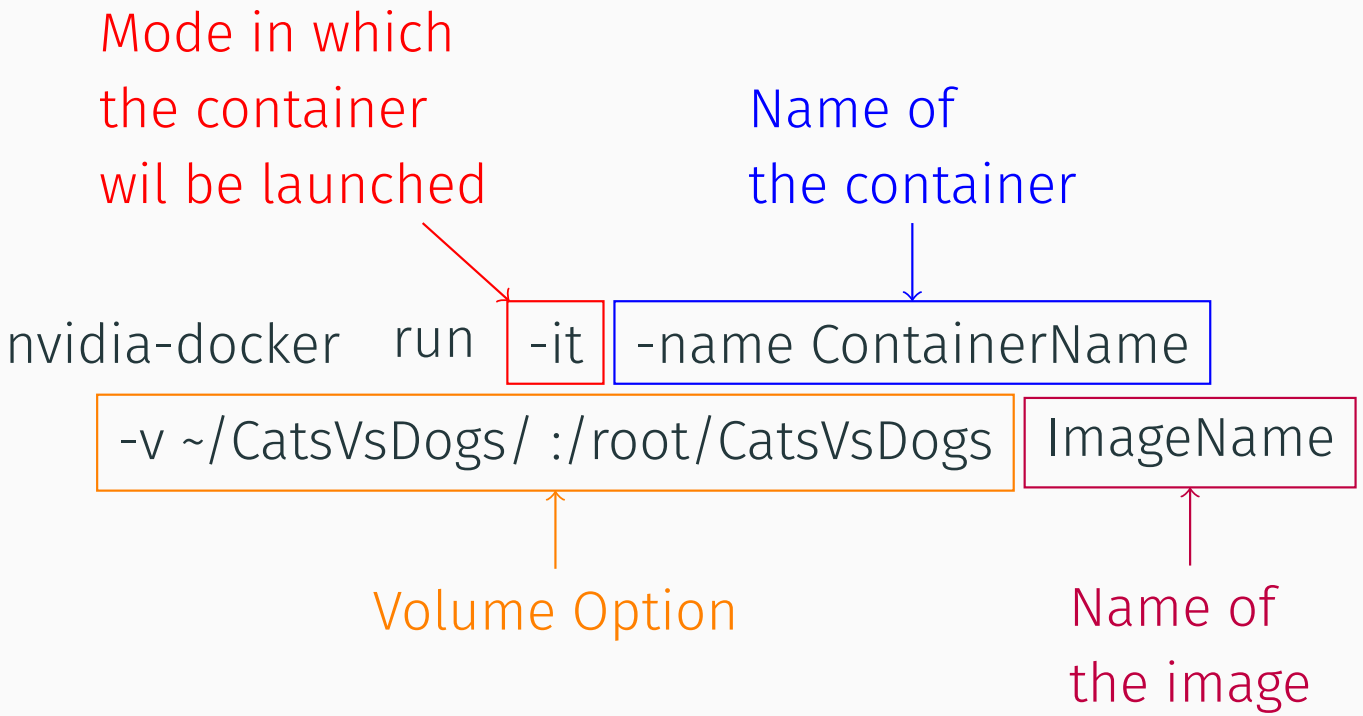


Mode of the container could be :

- -it : interactive mode,
- -dt : detached mode.

Run Container

Run the RUN command in order to run your **CONTAINER**.



Mode of the container could be :

- -it : interactive mode,
- -dt : detached mode.

Mounted Volume

The *-v* option allow you to use some date you have in yout machine within a container.

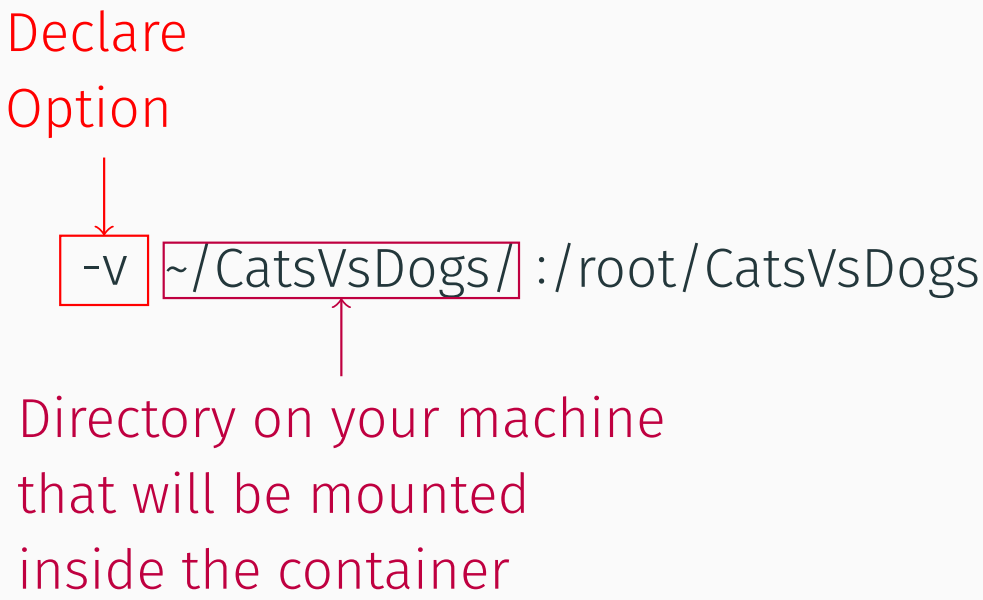
```
-v  ~/CatsVsDogs/ :/root/CatsVsDogs
```

Mounted Volume

The *-v* option allow you to use some date you have in yout machine within a container.

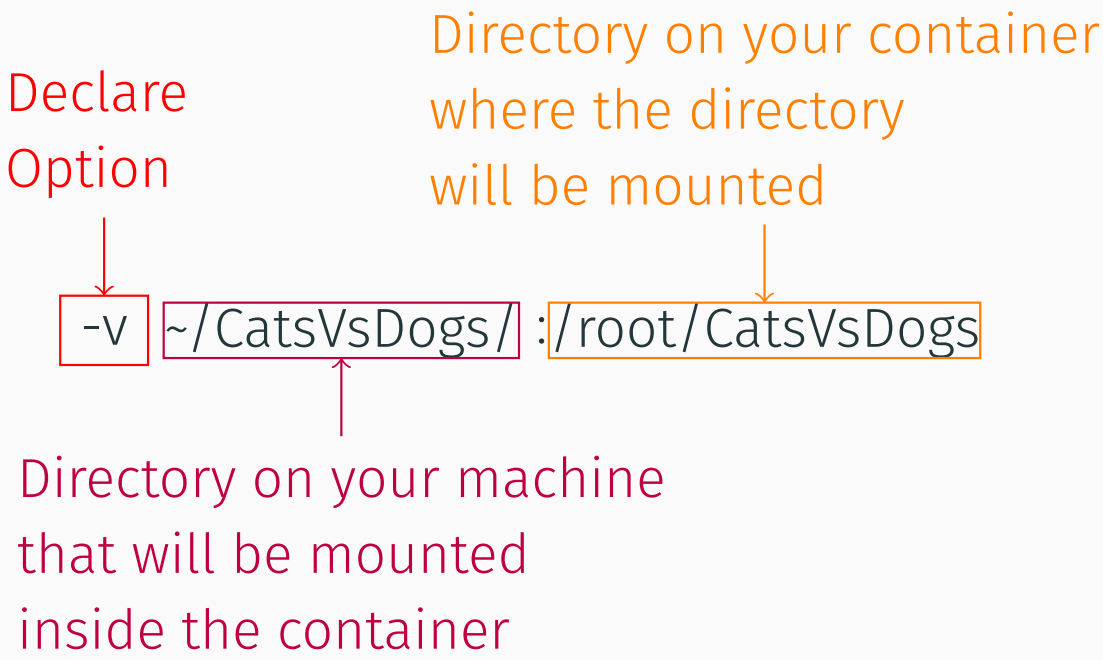
Mounted Volume

The *-v* option allow you to use some date you have in yout machine within a container.



Mounted Volume

The *-v* option allow you to use some date you have in yout machine within a container.



Util Command

- sudo nvidia-docker image ls -a
- sudo nvidia-docker container ls -a
- sudo nvidia-docker start/stop/rm container_name -a
- sudo nvidia-docker exec container_name 'Command to execute in container' -a

Other possibilities

- Use Jupyter on the instance (via ssh connection)
- Google Colab
- Image Gcloud

Reference

https ://towardsdatascience.com/how-docker-can-help-you-become-a-more-effective-data-scientist-7fc048ef91d5

Références