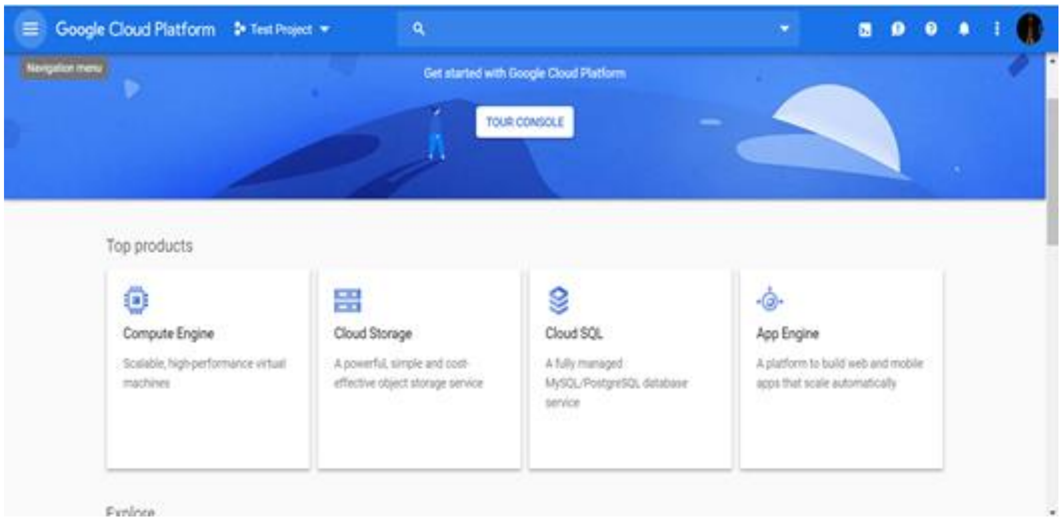
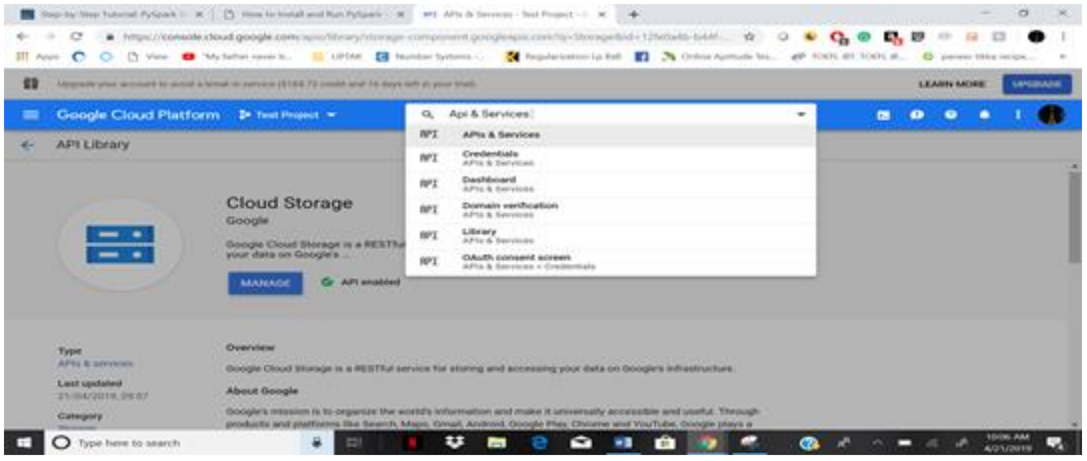
How to Create a VM on Google Dataproc and how to submit a spark job

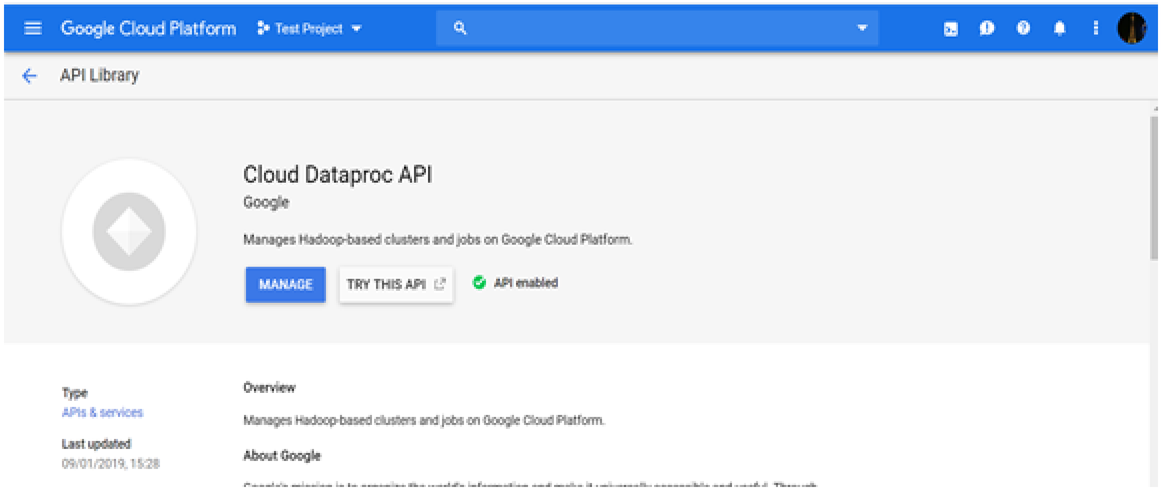
Google Cloud Dataproc gives the user a provision to implement Apache Hadoop Cluster and also lets user to connect to the various analytic data stored. Using the VM a user can directly submit a spark script to start a job as the Vm environment would be installed with spark, python2 & python3.

Step1: Create a Google Cloud Data platform free account using your Gmail id.
· When you do so you would be given a $300 credit for the next 12 months for usage
· Create you first new Project


Google Cloud Platform
Test Project
Navigation menu
Get started with Google Cloud Platform
TOUR CONSOLE
Top products
Compute Engine
Scalable, high-performance virtual machines
Cloud Storage
A powerful, simple and cost-effective object storage service
Cloud SQL
A fully managed MySQL/PostgreSQL database service
App Engine
A platform to build web and mobile apps that scale automatically
Explore

On the top you can see 'Test Project' which was created
· **Enabling API's** : We need to enable API to access various like Cloud Dataproc, Compute Engine & Data Storage
To Access we need to API & Services go the Library and search for the above services and enable them

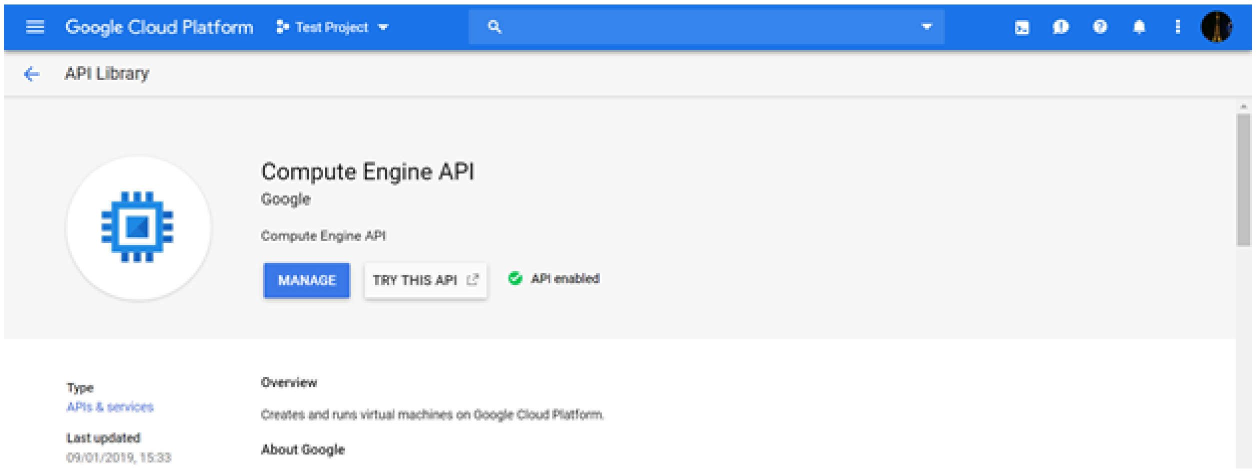




Cloud Dataproc API
Google
Manages Hadoop-based clusters and jobs on Google Cloud Platform.

Type
APIs & services
Last updated
09/01/2019, 15:28

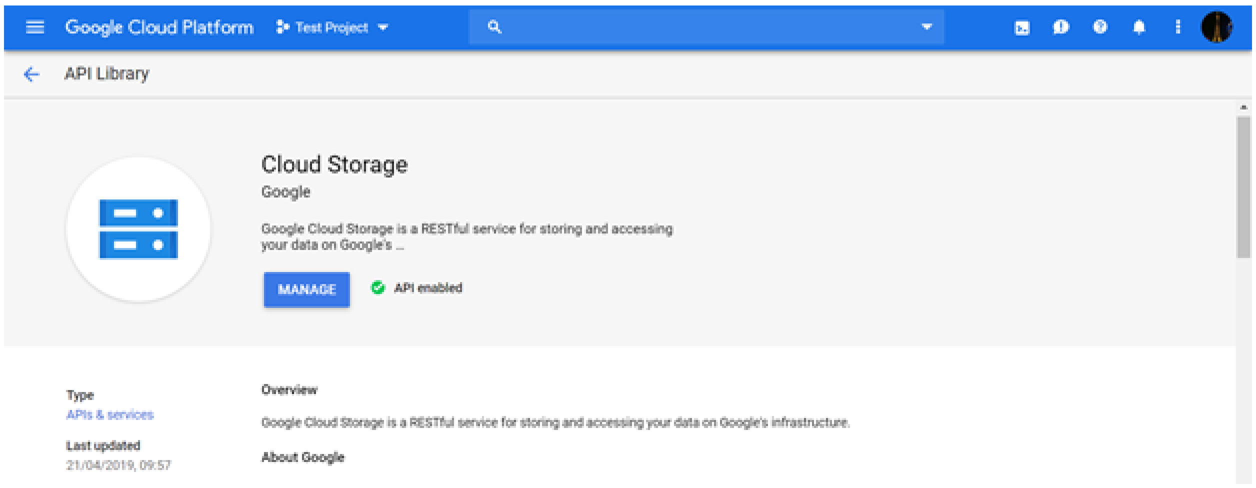
Overview
Manages Hadoop-based clusters and jobs on Google Cloud Platform.
About Google



Compute Engine API
Google
Compute Engine API

Type
APIs & services
Last updated
09/01/2019, 15:33

Overview
Creates and runs virtual machines on Google Cloud Platform.
About Google



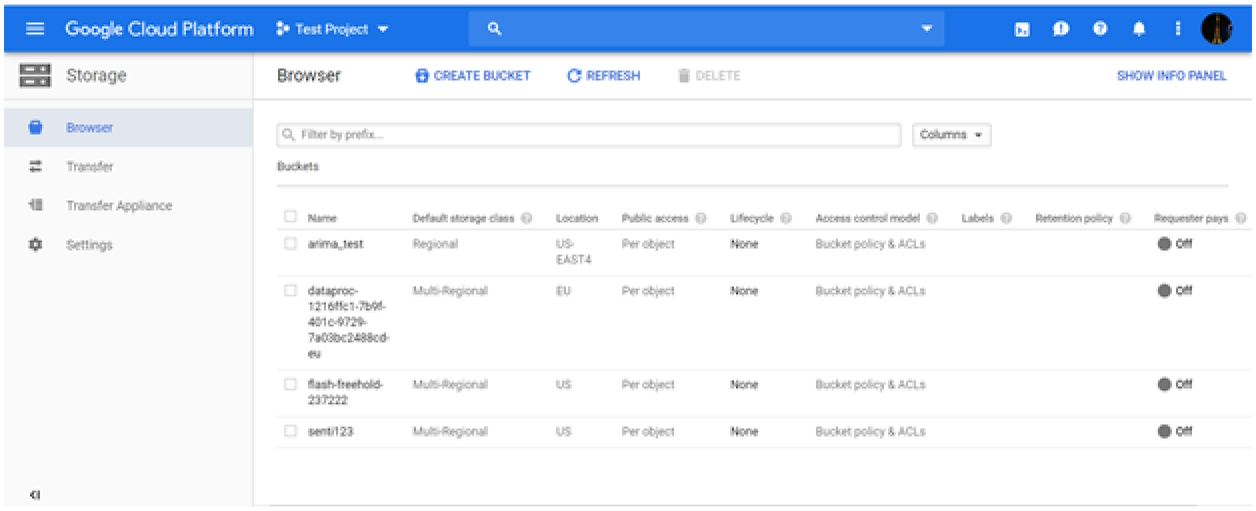
Cloud Storage
Google
Google Cloud Storage is a RESTful service for storing and accessing your data on Google's ...

Type
APIs & services
Last updated
21/04/2019, 09:57

Overview
Google Cloud Storage is a RESTful service for storing and accessing your data on Google's infrastructure.
About Google

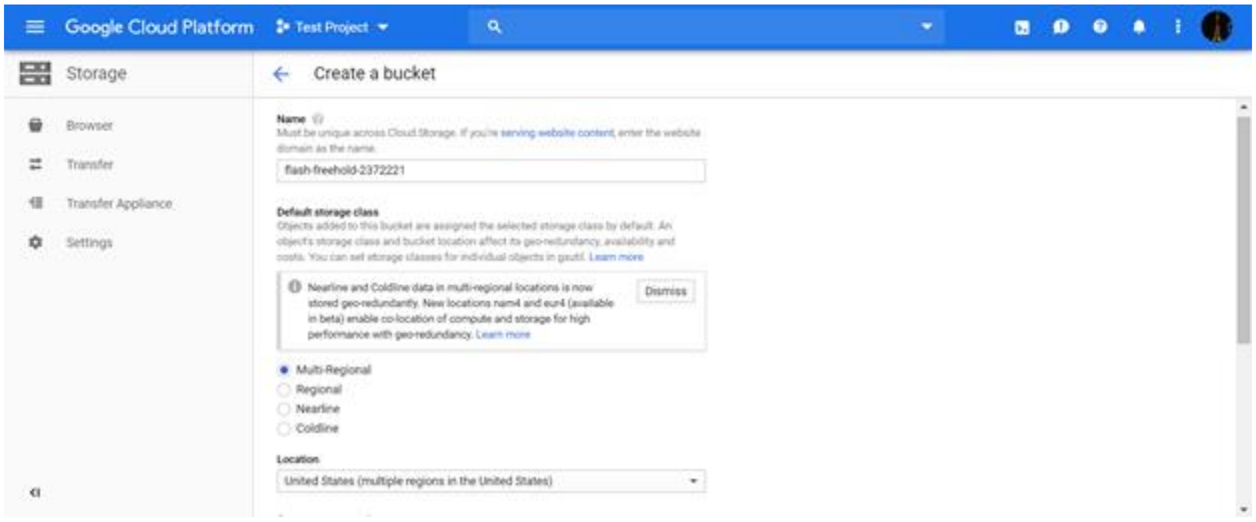
We also may install Google Cloud SDK to access the services using the command line interface

Step2: Creating A Storage Bucket in Google Cloud Platform
Go to ‘Storage Bucket’ in Google Cloud Platform



Name	Default storage class	Location	Public access	Lifecycle	Access control model	Labels	Retention policy	Requester pays
arima_test	Regional	US-EAST4	Per object	None	Bucket policy & ACLs			Off
dataproc-1216ffc1-7b9f-401c-9729-7a03bc2488cd-eu	Multi-Regional	EU	Per object	None	Bucket policy & ACLs			Off
flash-freehold-237222	Multi-Regional	US	Per object	None	Bucket policy & ACLs			Off
senti123	Multi-Regional	US	Per object	None	Bucket policy & ACLs			Off

· On the Top we can see a ‘**CREATE BUCKET**’ option, things to remember when filling in the create bucket page: for easy remember and access try filling the project-Id as the bucket name so won’t be confusing or difficult when a user wants to access it. For example in my case



Create a bucket

Name
Must be unique across Cloud Storage. If you're serving website content, enter the website domain as the name.
flash-freehold-2372221

Default storage class
Objects added to this bucket are assigned the selected storage class by default. An object's storage class and bucket location affect its geo-redundancy, availability and costs. You can set storage classes for individual objects in gsutil. Learn more

Nearline and Coldline data in multi-regional locations is now stored geo-redundantly. New locations nam4 and eur4 (available in beta) enable co-location of compute and storage for high performance with geo-redundancy. Learn more

[✓ Multi-Regional] Regional Nearline Coldline

Location
United States (multiple regions in the United States)

· For easy accessibility for future parts of the code, you can run this simple command on Google Cloud Sdk or your VM instance

```
> export Project_ID = ‘enter_Project_ID’
```

This Command is for creating a bucket using the command line of Google cloud Platform

```
> gsutil mb gs://${Project_ID}
```

This will create a bucket with default settings or you can go to the console and create the bucket as shown earlier.

This is the commands to select the cluster region

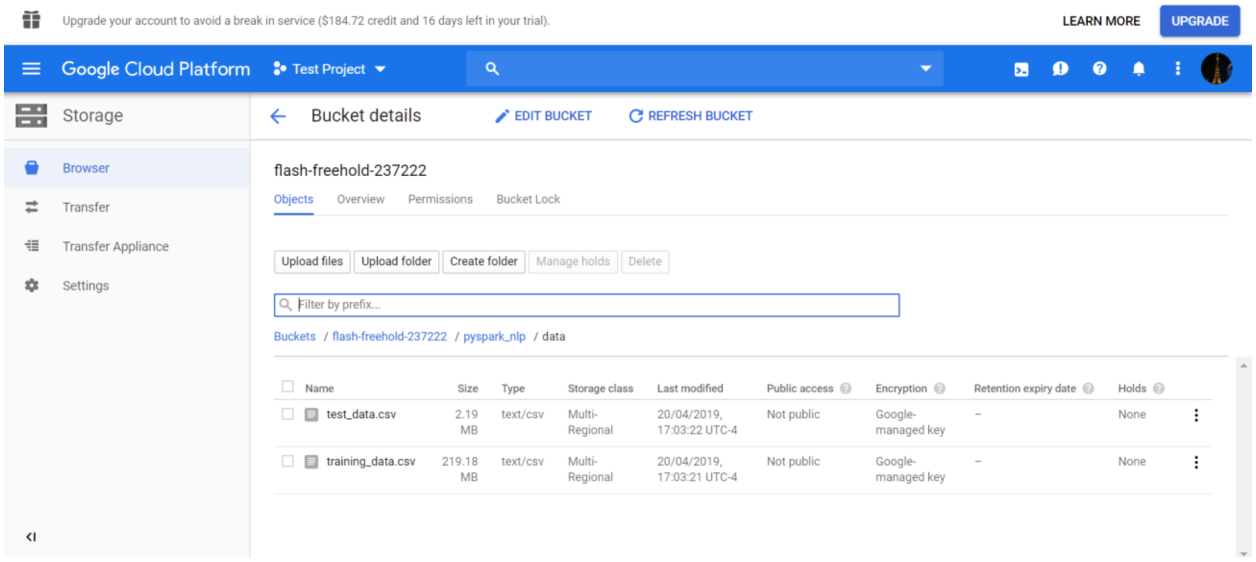
```
#ex1) multi-region europe
gsutil mb -l eu gs://${PROJECT_ID}
#ex)2 region europe-west1
gsutil mb -l europe-west1 gs://${PROJECT_ID}
```

Step 3: Preparing Script to load data into the bucket

We need to create a script to run our code. This script would be responsible for uploading of data into the cloud storage bucket.
The command to run the script, for example, can be

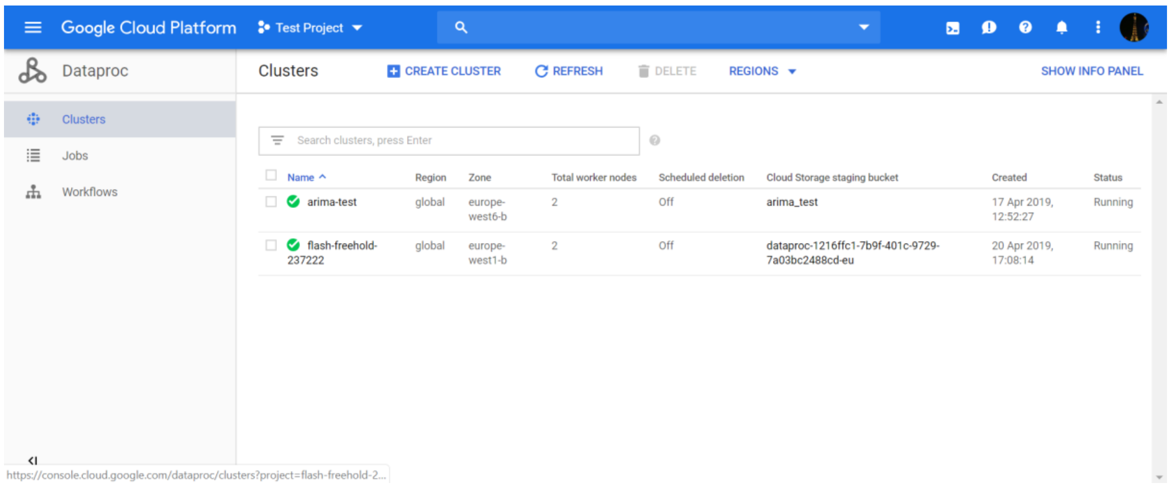
```
> ./data_prep.sh or any file name you decide to create
```

After the script is successfully run, we need to check in the Web console by going into the bucket -> if those files have been uploaded or not.



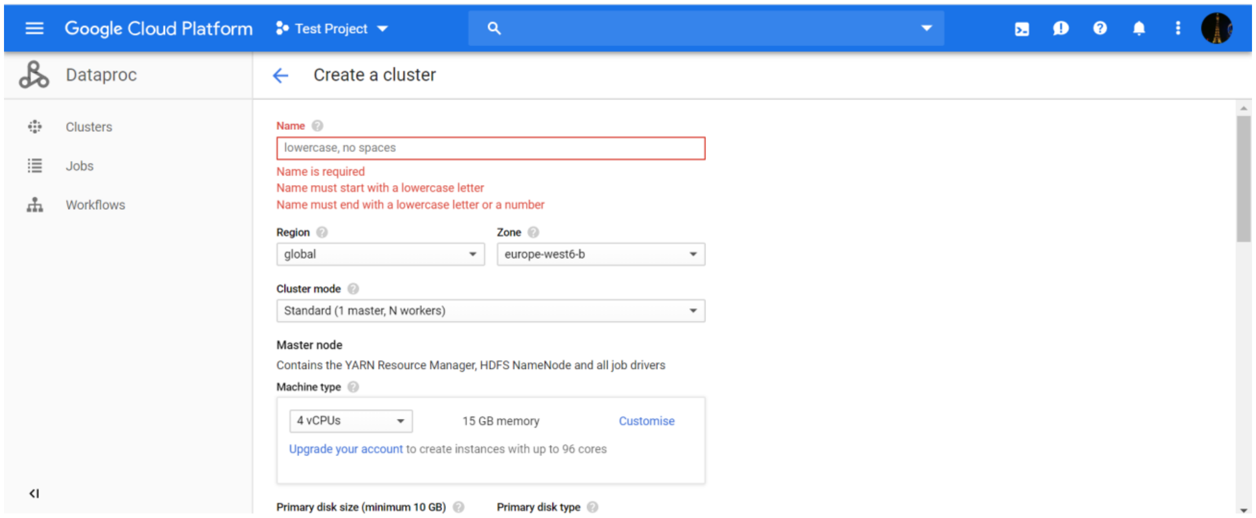
Name	Size	Type	Storage class	Last modified	Public access	Encryption	Retention expiry date	Holds
test_data.csv	2.19 MB	text/csv	Multi-Regional	20/04/2019, 17:03:22 UTC-4	Not public	Google-managed key	–	None
training_data.csv	219.18 MB	text/csv	Multi-Regional	20/04/2019, 17:03:21 UTC-4	Not public	Google-managed key	–	None

How To create a Cloud Dataproc Cluster for running Apache Spark or even Apache Hadoop Clusters



Name	Region	Zone	Total worker nodes	Scheduled deletion	Cloud Storage staging bucket	Created	Status
arima-test	global	europe-west6-b	2	Off	arima_test	17 Apr 2019, 12:52:27	Running
flash-freehold-237222	global	europe-west1-b	2	Off	dataproc-1216ffc1-7b9f-401c-9729-7a03bc2488cd-eu	20 Apr 2019, 17:08:14	Running

We can create a new Cluster and give assign the number of worker nodes:



Create a cluster

Name
lowercase, no spaces
Name is required
Name must start with a lowercase letter
Name must end with a lowercase letter or a number

Region: global
Zone: europe-west6-b

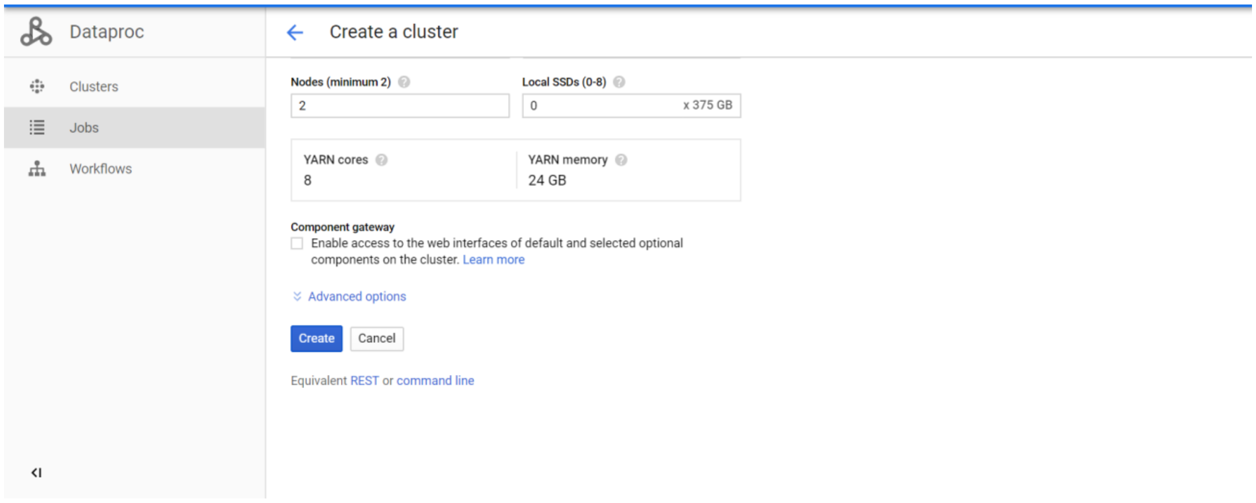
Cluster mode: Standard (1 master, N workers)

Master node
Contains the YARN Resource Manager, HDFS NameNode and all job drivers

Machine type: 4 vCPUs — 15 GB memory — Customise
Upgrade your account to create instances with up to 96 cores

Primary disk size (minimum 10 GB)
Primary disk type

Assign cluster name and number of worker nodes along with memory requirements



Create a cluster

Nodes (minimum 2): 2
Local SSDs (0-8): 0 x 375 GB

YARN cores: 8
YARN memory: 24 GB

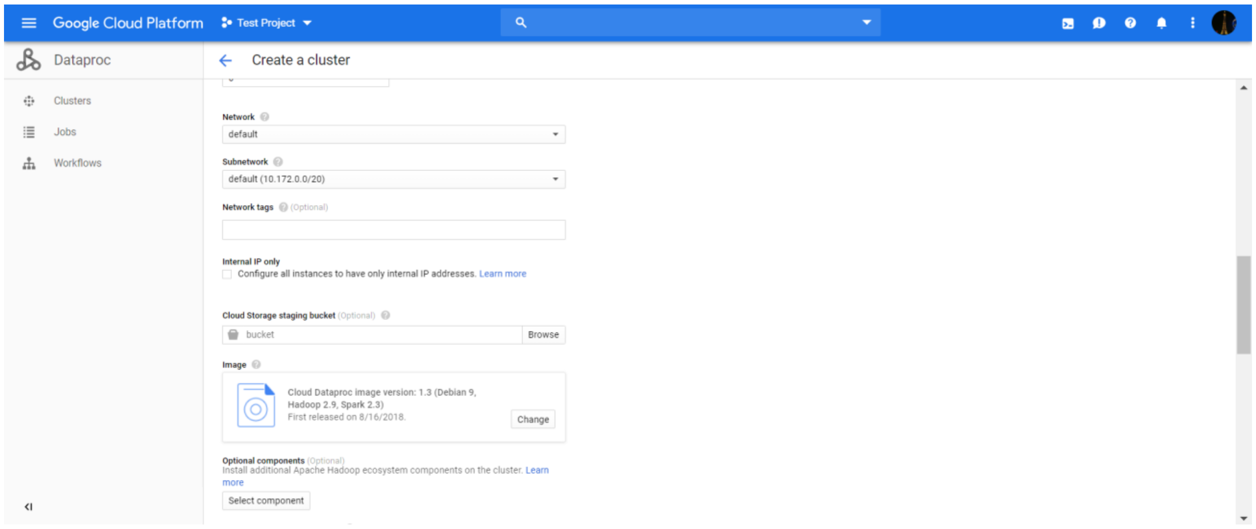
Component gateway
Enable access to the web interfaces of default and selected optional components on the cluster. Learn more

Advanced options

Create Cancel

Equivalent REST or command line

By selecting the advance options you can set the Image which basically is the Ubuntu version you want to work within the cluster along with the spark & hadoop version



Create a cluster

Network
default

Subnetwork
default (10.172.0.0/20)

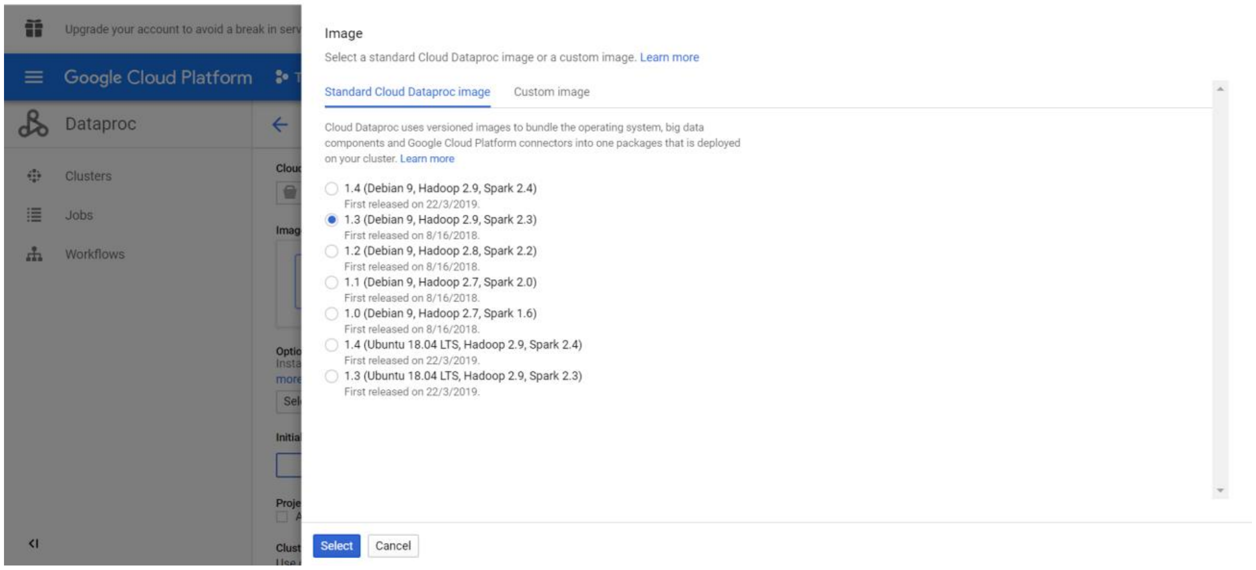
Network tags (Optional)

Internal IP only
Configure all instances to have only internal IP addresses. Learn more

Cloud Storage staging bucket (Optional)
bucket Browse

Image
Cloud Dataproc image version: 1.3 (Debian 9, Hadoop 2.9, Spark 2.3)
First released on 8/16/2018.
Change

Optional components (Optional)
Install additional Apache Hadoop ecosystem components on the cluster. Learn more
Select component



Image

Select a standard Cloud Dataproc image or a custom image. Learn more

Standard Cloud Dataproc image | Custom image

Cloud Dataproc uses versioned images to bundle the operating system, big data components and Google Cloud Platform connectors into one packages that is deployed on your cluster. Learn more

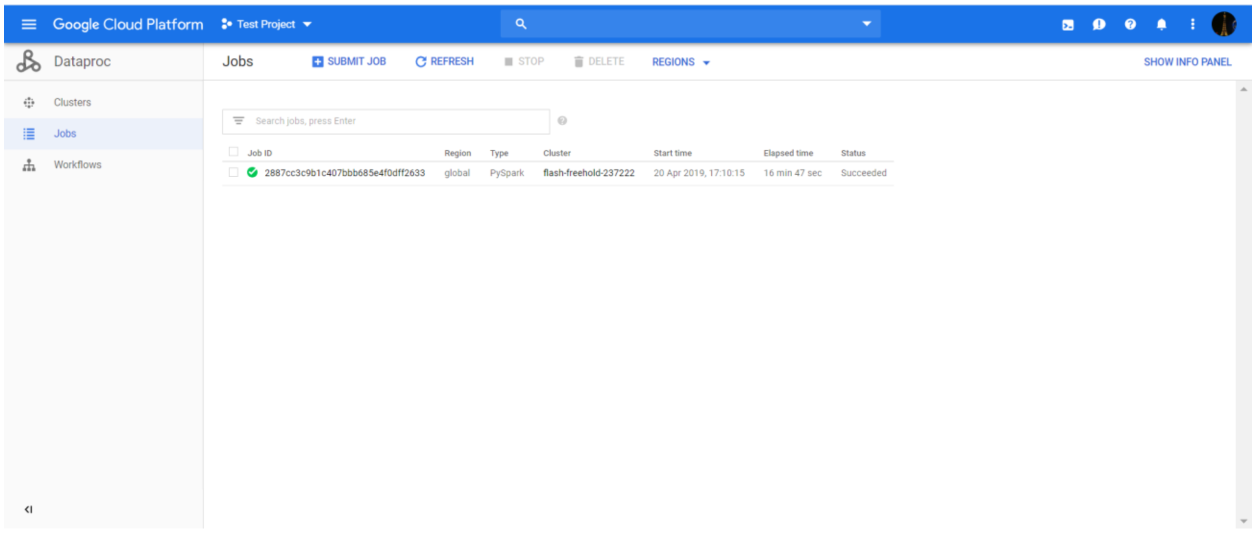
- ○ 1.4 (Debian 9, Hadoop 2.9, Spark 2.4) First released on 22/3/2019.
- ● 1.3 (Debian 9, Hadoop 2.9, Spark 2.3) First released on 8/16/2018.
- ○ 1.2 (Debian 9, Hadoop 2.8, Spark 2.2) First released on 8/16/2018.
- ○ 1.1 (Debian 9, Hadoop 2.7, Spark 2.0) First released on 8/16/2018.
- ○ 1.0 (Debian 9, Hadoop 2.7, Spark 1.6) First released on 8/16/2018.
- ○ 1.4 (Ubuntu 18.04 LTS, Hadoop 2.9, Spark 2.4) First released on 22/3/2019.
- ○ 1.3 (Ubuntu 18.04 LTS, Hadoop 2.9, Spark 2.3) First released on 22/3/2019.

Select Cancel

After doing the above procedure you would have successfully set your Cloud Dataproc cluster

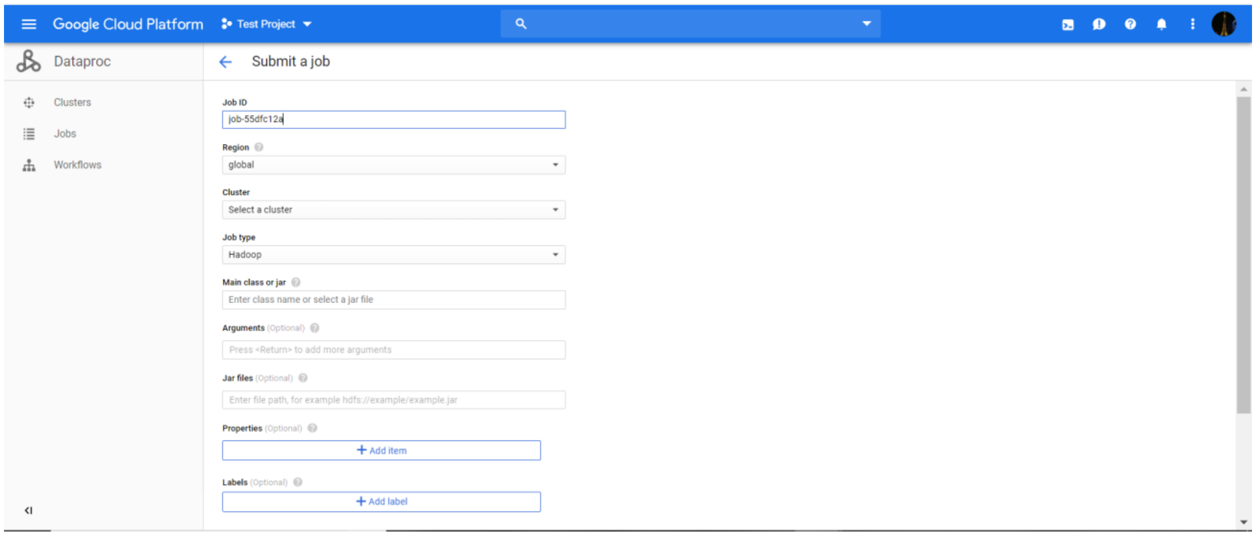
Step 4: Submit Job

Click on the Jobs of the Dataprocs page



Job ID	Region	Type	Cluster	Start time	Elapsed time	Status
2887cc3c9b1c407bbb685e4f0dff2633	global	PySpark	flash-freehold-237222	20 Apr 2019, 17:10:15	16 min 47 sec	Succeeded

Click on Submit a Job



Job ID: job-55dfc12a
Region: global
Cluster: Select a cluster
Job type: Hadoop
Main class or jar: Enter class name or select a jar file
Arguments (Optional): Press <Return> to add more arguments
Jar files (Optional): Enter file path, for example hdfs://example/example.jar
Properties (Optional): + Add item
Labels (Optional): + Add label

You need to give the cloud location of the by specifying the bucket location with the following command in the main class or jar file

gs://bucket_name/file_location/python_file.py

Also, we can add additional files in the argument section

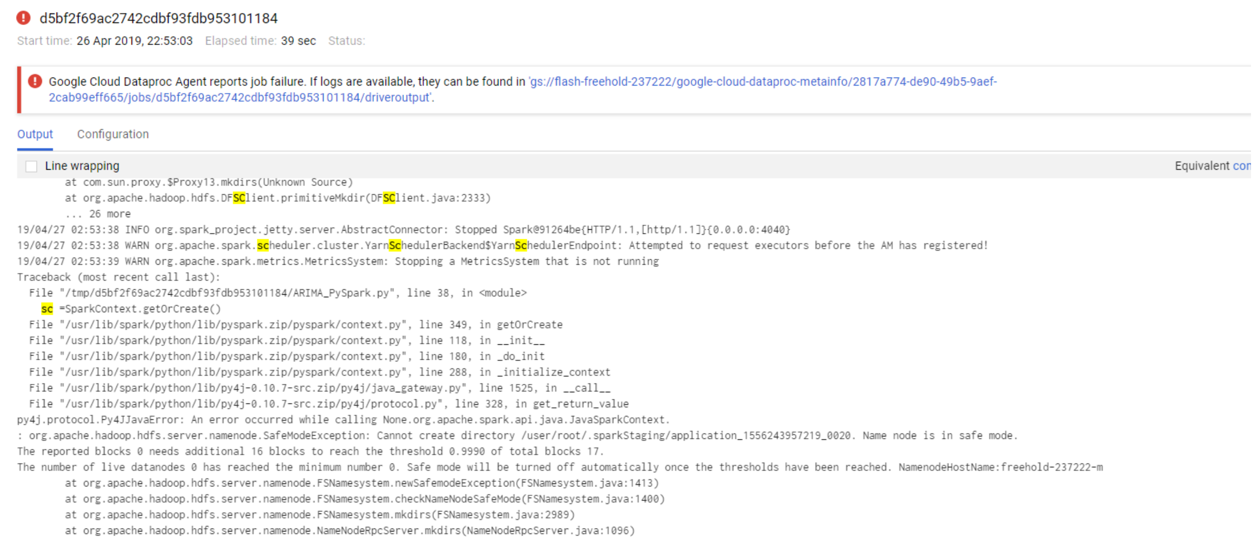
Click on '**Submit**'

After success full creation of the Job, we can see the result, with the time taken to complete the Job

Important gcloud commands that can be used when you creating a Job using VM instance:
Some important parameter's to remember:

1. Bucket name :
2. Cluster name:
3. Loading data from a source, for example, git: - git clone ‘githib link’
4. After that, we need to check if a file of the GitHub has been created or not : cd filename/
5. Now copying a command into the Google storage bucket: gsutil cp filename.py gs://bucket name/foldername/filename.py
6. Check in the console if the file is uploaded or not
7. Command: gs submit jobs pyspark cluster -- name gs://bucket name/folder name/ filename.py

We have tried implementing it on Google Dataproc but have fallen short of it as we weren’t able to solve this error



d5bf2f69ac2742cdbf93fdb953101184

Start time: 26 Apr 2019, 22:53:03   Elapsed time: 39 sec   Status:

Google Cloud Dataproc Agent reports job failure. If logs are available, they can be found in 'gs://flash-freehold-237222/google-cloud-dataproc-metainfo/2817a774-de90-49b5-9aef-2cab99eff665/jobs/d5bf2f69ac2742cdbf93fdb953101184/driveroutput'.

```
        at com.sun.proxy.$Proxy13.mkdirs(Unknown Source)
        at org.apache.hadoop.hdfs.DFSClient.primitiveMkdir(DFSClient.java:2333)
        ... 26 more
19/04/27 02:53:38 INFO org.spark_project.jetty.server.AbstractConnector: Stopped Spark@91264be{HTTP/1.1,[http/1.1]}{0.0.0.0:4040}
19/04/27 02:53:38 WARN org.apache.spark.scheduler.cluster.YarnSchedulerBackend$YarnSchedulerEndpoint: Attempted to request executors before the AM has registered!
19/04/27 02:53:39 WARN org.apache.spark.metrics.MetricsSystem: Stopping a MetricsSystem that is not running
Traceback (most recent call last):
  File "/tmp/d5bf2f69ac2742cdbf93fdb953101184/ARIMA_PySpark.py", line 38, in <module>
    sc =SparkContext.getOrCreate()
  File "/usr/lib/spark/python/lib/pyspark.zip/pyspark/context.py", line 349, in getOrCreate
  File "/usr/lib/spark/python/lib/pyspark.zip/pyspark/context.py", line 118, in __init__
  File "/usr/lib/spark/python/lib/pyspark.zip/pyspark/context.py", line 180, in _do_init
  File "/usr/lib/spark/python/lib/pyspark.zip/pyspark/context.py", line 288, in _initialize_context
  File "/usr/lib/spark/python/lib/py4j-0.10.7-src.zip/py4j/java_gateway.py", line 1525, in __call__
  File "/usr/lib/spark/python/lib/py4j-0.10.7-src.zip/py4j/protocol.py", line 328, in get_return_value
py4j.protocol.Py4JJavaError: An error occurred while calling None.org.apache.spark.api.java.JavaSparkContext.
: org.apache.hadoop.hdfs.server.namenode.SafeModeException: Cannot create directory /user/root/.sparkStaging/application_1556243957219_0020. Name node is in safe mode.
The reported blocks 0 needs additional 16 blocks to reach the threshold 0.9990 of total blocks 17.
The number of live datanodes 0 has reached the minimum number 0. Safe mode will be turned off automatically once the thresholds have been reached. NamenodeHostName:freehold-237222-m
        at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.newSafemodeException(FSNamesystem.java:1413)
        at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.checkNameNodeSafeMode(FSNamesystem.java:1400)
        at org.apache.hadoop.hdfs.server.namenode.FSNamesystem.mkdirs(FSNamesystem.java:2989)
        at org.apache.hadoop.hdfs.server.namenode.NameNodeRpcServer.mkdirs(NameNodeRpcServer.java:1096)
```