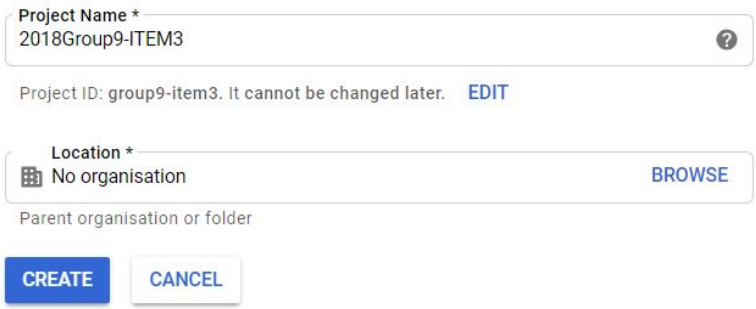
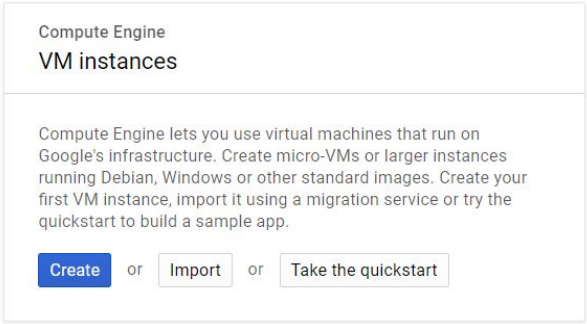
2018 Group 9 Item3
Instructions

Create an instance on Google Cloud Platform

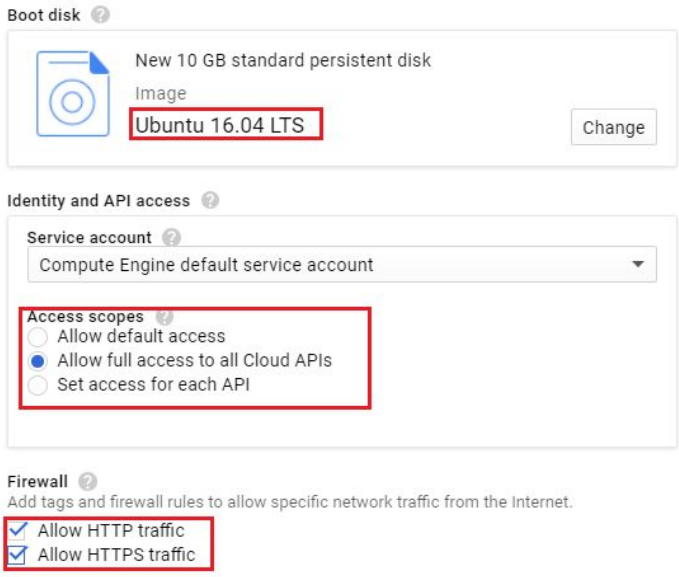
1. Go to Google Cloud Platform, create a new project called *2018Group9-ITEM3;*



2. Select this project and open it;
3. Click **Compute Engine** from navigation bar. When it is ready, click **Create** to launch a new instance, this will bring you to instance setting page;



4. In the **boot disk** section, change **OS images** to **Ubuntu 16.04 LTS** and click **Select**;
5. In the **identify and API access** section, select **Allow full access to all Cloud APIs**;
6. In the **firewall** section, select **Allow HTTP traffic** and **Allow HTTPS traffic**;



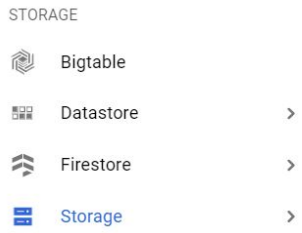
5. Make sure you have select all the options right, click **Create**;
6. After the instance is ready, you can see it from your VM console. We will later refer to this instance as **local instance**.

Transfer local folder to the instance

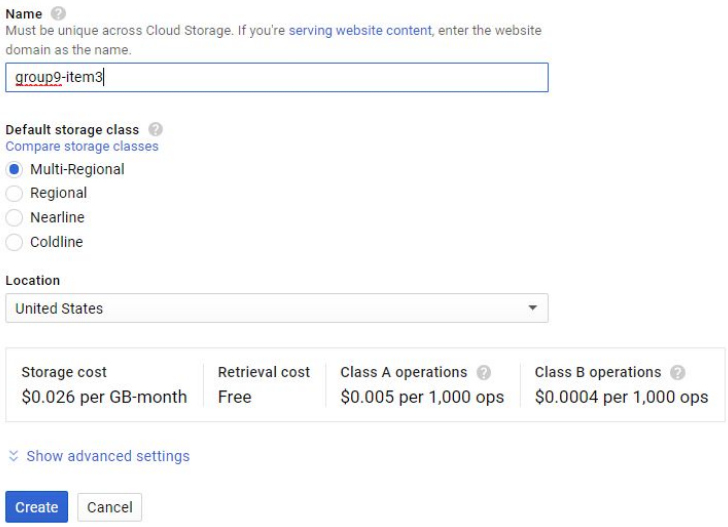
Platform-independent file transfer methods

We will use **Google Cloud Storage** to upload your local folder to a Storage Bucket and download it to your local instance;

1. Click **Storage** in the navigation bar;

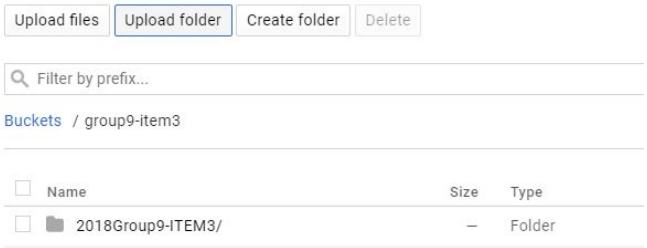


2. Create a new Bucket, find an available name by yourself, (let's assume the name is *your_bucket_name*) keep the other settings default and Click **Create**;



3. Click **Upload folder**, you can then upload our folder *2018Group9-ITEM3* to the bucket;

   Note: *What you have downloaded from Stream should be a zip file, please unzip it to a folder and upload this folder, do not change the folder name!*



4. After the entire folder has been uploaded, go back to your VM console, find you instance, click **SSH** to connect to it. Then a terminal window will open;

5. After the terminal is ready, switch to root user. Then download the folder from buckets you created before, replace the *your_bucket_name* with your own bucket name:

   >>> sudo -i

   >>> gsutil cp -r gs://*your_bucket_name*/2018Group9-ITEM3/ .

Execute script and playbook

Install Ansible

You need to install Ansible, we provide a shell script to help you do this. Besides installation, it will also change some directories inside the folder:

>>> sh 2018Group9-ITEM3/installAnsible.sh

Before you run the playbook

You are required to pass two extra variables when you run the playbook, they are project id and backup period respectively;

Note: We need your project id because this will make sure you are creating instances under your credential.

To find your project id, find and click your project name on the top side of the page, it will then prompt out a window, find the project id of *2018Group9-ITEM3* and copy it;

To set up backup period, you need to follow the CRON syntax, for example, we recommend you use '*/1 * * * *', this means backup database per minute. However, you can change it into any period, eg. '*/5 * * * *' means backup every 5 minutes.

However, we strongly recommend you to use the default period because if the backup period is too long, your change on the database will not be recorded immediately which makes it hard to do the following test!

Run the playbook

Replace *your_project_id* with your project id, if you want to change the backup period, remember to change the value inside period as well:

>>> ansible-playbook task3_backup.yml --extra-vars "period='*/1 * * * *' project_id=*your_project_id*"

Note: Do not leave any extra blank space, do not omit the apostrophe!

This playbook will firstly generates a distributed system with a management server, three web servers and one remote database server, then it try to backup the remote database server automatically, the backup process is executed locally. Please be patient, this will take you a few minutes;

After all the tasks have been finished, refresh your VM console, you can now see 5 additional instances created : *managementserver, webserver1, webserver2, webserver3, databaseserver*

Verification from test result

View Local Backup Files

To test whether automatic backup has succeeded. A direct approach is to see the backup files.

We have configured your local instance as a backup server, i.e. all the backup files have been stored into your local instance instead of a remote database server;

Go to /backup/mysql, see the backup folders,

>>> cd /backup/mysql && ls

Then, you can see all the backup folders till now, all the folders are named with the backup time, you can pick up any of these folder and look the file inside it, replace the red text with the folder name you want to access:

>>> cd *backup_folder* && ls

Inside the folder, you can see a zip file called etherpad_lite_db.sql.gz (we have specified that only etherpad database has been backup)

Database Restore

Let's imagine the remote database server was corrupted or collapsed at some time, using these backup files, you can restore all the data back.

Firstly, copy any web server's external IP address, paste it into the web browser.

Name	Zone	Recommendation	Internal IP	External IP	Connect
databaseserver	australia-southeast1-b		10.152.0.3 (nic0)	35.189.15.190	SSH
instance-2	us-east1-b		10.142.0.3 (nic0)	35.185.123.146	SSH
managementserver	australia-southeast1-b		10.152.0.2 (nic0)	35.189.55.104	SSH
webserver1	australia-southeast1-b		10.152.0.4 (nic0)	35.189.54.96	SSH
webserver2	australia-southeast1-b		10.152.0.5 (nic0)	35.201.0.132	SSH
webserver3	australia-southeast1-b		10.152.0.6 (nic0)	35.197.181.92	SSH

Then you can see etherpad is running in your web page, then create a new pad with the name *demo*:

Note: Please don't use create newpad button, because this will create a pad with a random name, instead, type the pad name into the text box.

Edit it whatever you want;

Execute the test playbook, replace the *local_private_ip* with your local instance private ip address,

Name	Zone	Recommendation	Internal IP	External IP	Connect
databaseserver	australia-southeast1-b		10.152.0.3 (nic0)	35.189.15.190	SSH
instance-2	us-east1-b		10.142.0.3 (nic0)	35.185.123.146	SSH
managementserver	australia-southeast1-b		10.152.0.2 (nic0)	35.189.55.104	SSH
webserver1	australia-southeast1-b		10.152.0.4 (nic0)	35.189.54.96	SSH
webserver2	australia-southeast1-b		10.152.0.5 (nic0)	35.201.0.132	SSH
webserver3	australia-southeast1-b		10.152.0.6 (nic0)	35.197.181.92	SSH

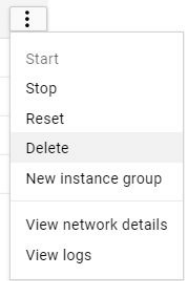
Replace the *restore_time* with the latest backup time (you can find your latest backup folder inside /backup/mysql)

```
>>> cd ~
```

```
>>> ansible-playbook task3_rollback.yml --extra-vars "private_ip=local_private_ip time=restore_time"
```

This playbook will bring a new database server back online (which is your local instance)

When all the tasks have been finished, you can shut down the remote database server now;



We are now using a new database (local instance). Copy any web server's external IP address, paste it into the web browser, you can still access etherpad after using a new database. Type *demo* into the text area, your previous pad still exists which means the data has restored successfully.