Google Colaboratory

Introduction, Setup, and use in GPU computing, Image processing, classification and CNN using “Fastai” library

Introduction:

Google Colab is Google’s free cloud *service* for **AI developers**. With Colab, we can develop deep learning applications on the **GPU (Nvidia Tesla K80)** free of cost.

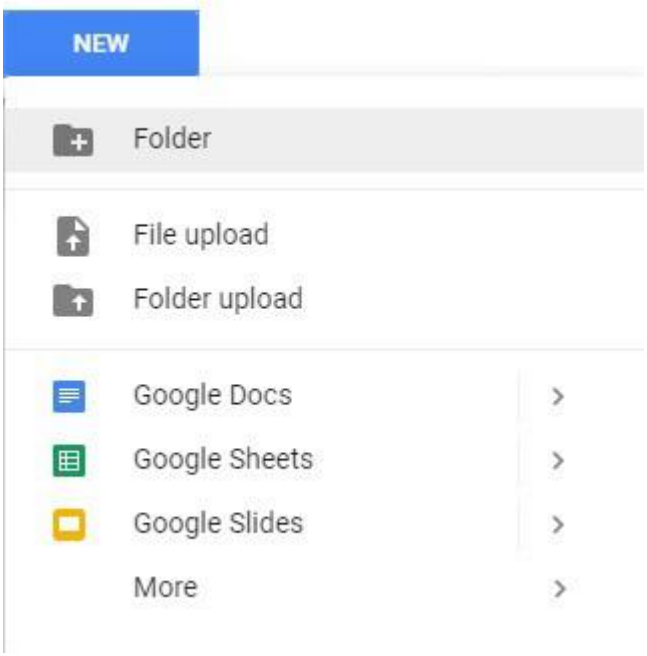
Using Google Colab we can:

- Improve our **Python** programming language coding skills.
- Develop deep learning applications using popular libraries such as **Keras**, **TensorFlow**, **PyTorch,** and **OpenCV**.

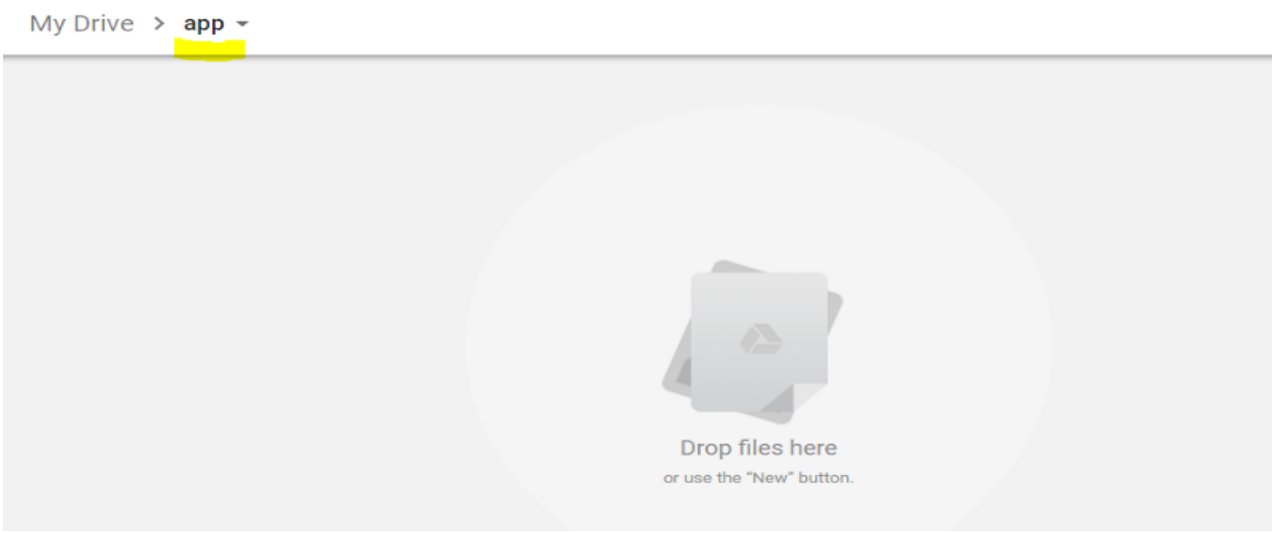
The most important feature that distinguishes Colab from other free cloud services is; **Colab** provides GPU and is totally free.

Google Colab Setup:

Creating Folder on Google Drive



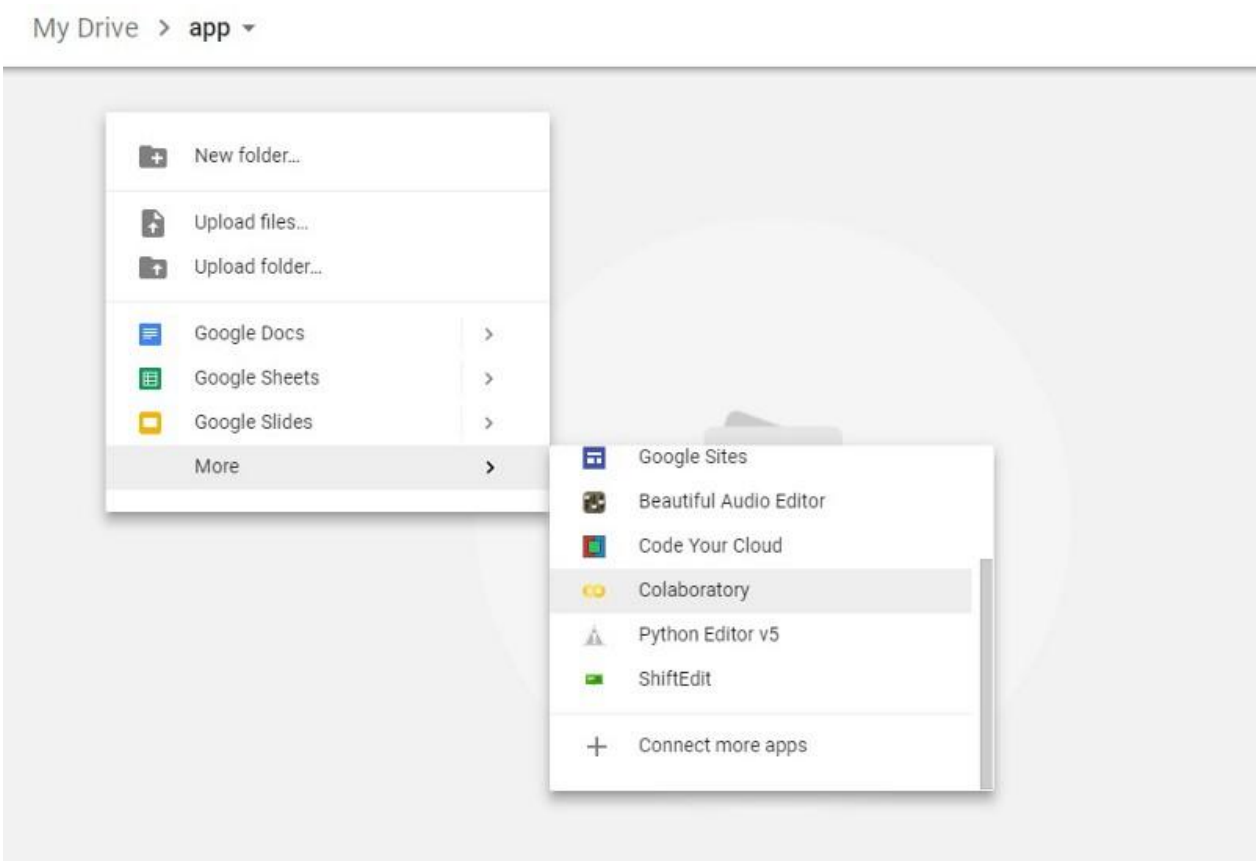
Since **Colab** is works on **Google Drive**, we first need to specify the folder we’ll work. We can create a folder named “**app**” on **Google Drive**.



'app' folder created

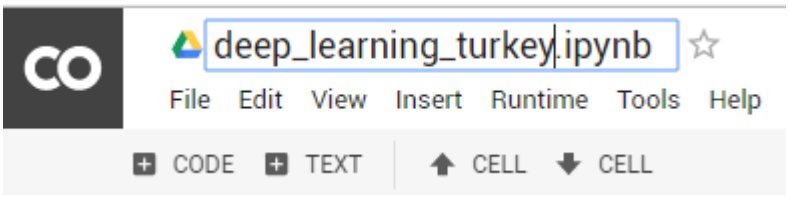
Creating New Colab Notebook

Create a new notebook via **Right click > More > Colaboratory**



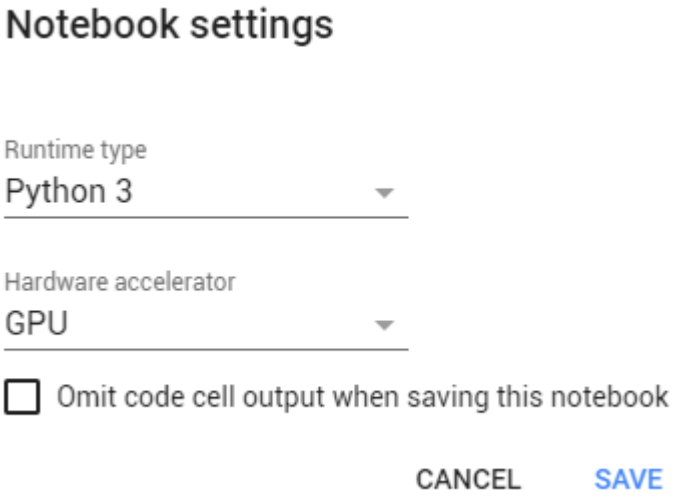
Right click > More > Colaboratory

Rename notebook by means of clicking the file name.



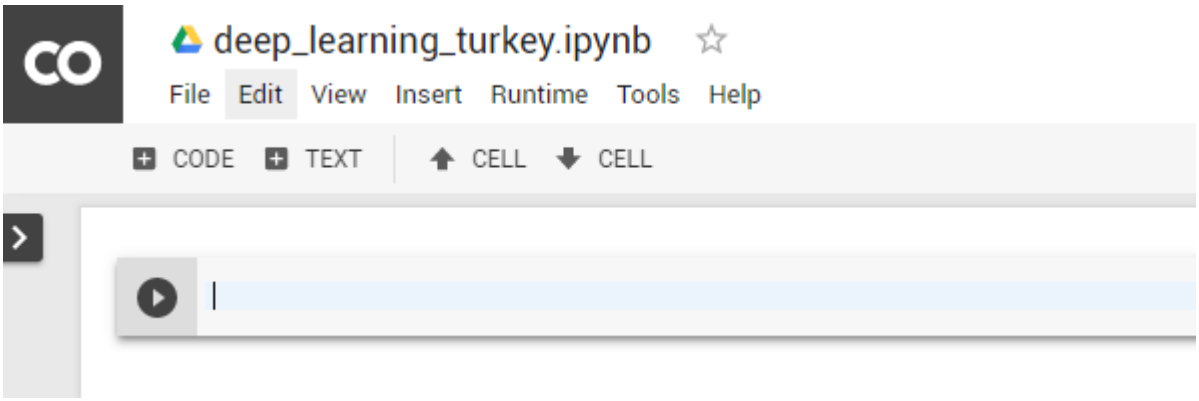
Setting Free GPU

To alter default hardware **(CPU to GPU or vice versa)**; just follow **Edit > Notebook settings** or **Runtime>Change runtime type** and **select GPU** as **Hardware accelerator**.



Running Basic Python Codes with Google Colab

Now we can start using **Google Colab**.



Fastai library Setup:

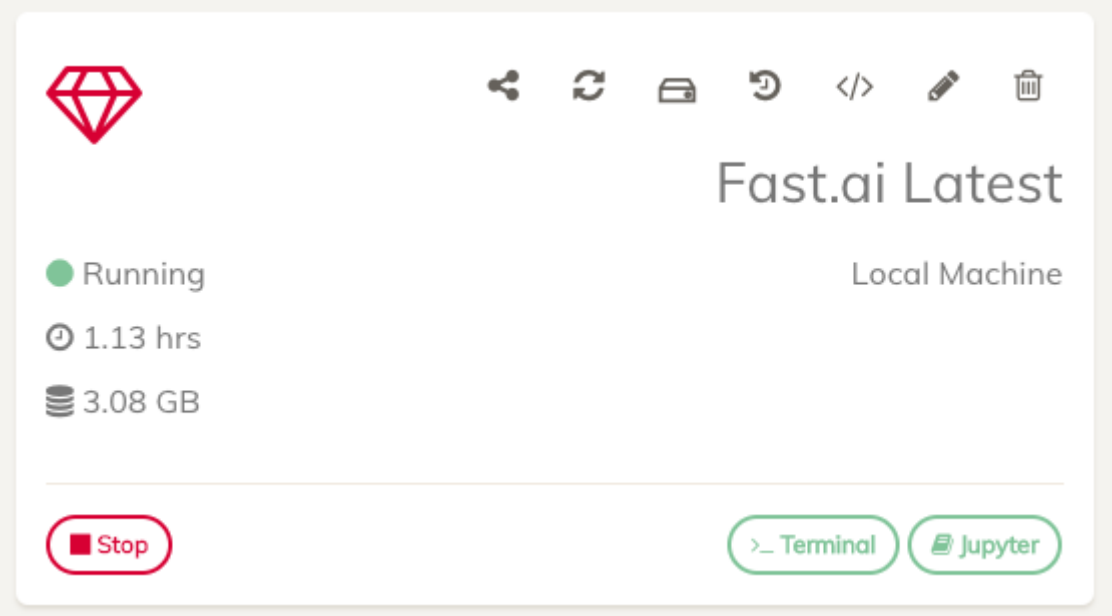
Following pre-requisite, one time, steps are needed:

1. Sign up for Gmail and go to Google Colab
2. Sign up for Clouderizer. By default you will be on free plan for individual users.
3. Login to Clouderizer console and go to Community Projects. Search for fast.ai template and clone it.
4. On project wizard, 2nd tab (machine setup), AWS GPU spot instance is selected as machine type by default. Select Local Machine instead. On all remaining screens, select the default options and save.

Now every time you need to start fast.ai notebooks on Google Colab GPU, follow below 3 steps:

1. Create a new empty Python 3 notebook in google colab. Go to Edit -> Notebook Setting, and select Hardware Accelerator as GPU.
2. From Clouderizer console, press Start on fast.ai project created earlier. This will show a script snippet for starting the project on any terminal. Copy the snippet for Linux.
3. Come back to colab notebook, add a new code block, paste the snippet there (prepending with an !), and run the block using Shift-Enter

This will trigger an automated fast.ai course environment setup, latest code download and dogscats dataset download. You can go back to Clouderizer console and track the progress of this setup. Once setup is complete, project status becomes Running and Jupyter Notebook button becomes available. Clicking on Jupyter button, will open Jupyter notebook with fast.ai github code.



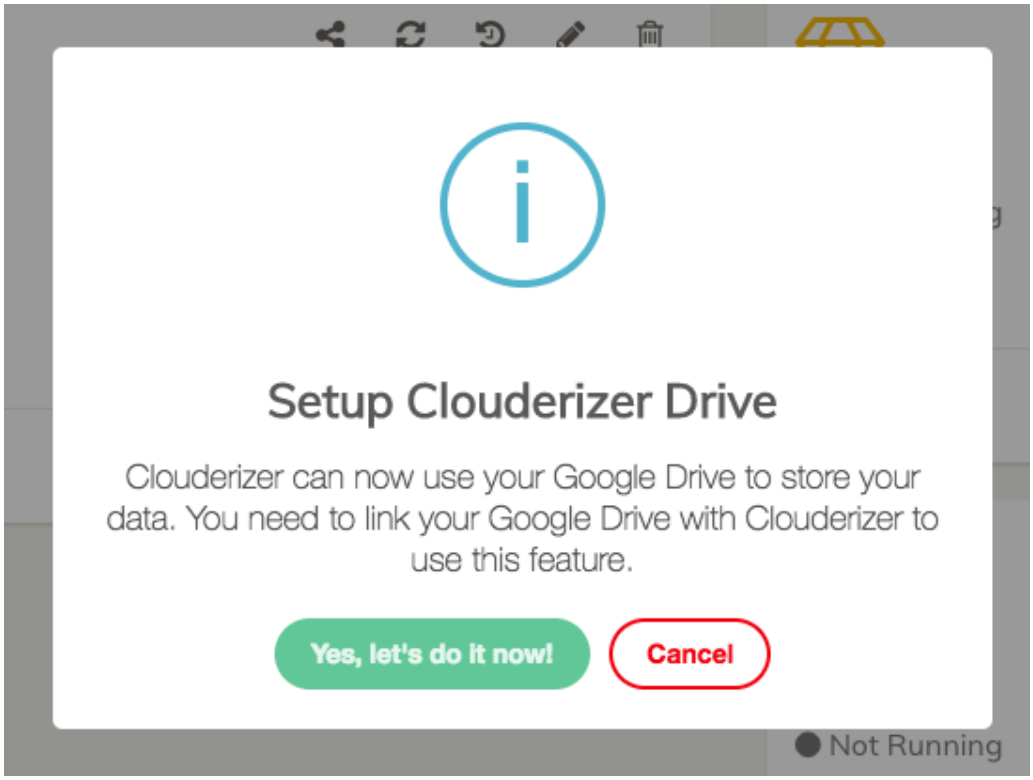
Clouderizer Drive:

Clouderizer Drive is a very important tool to manage Clouderizer projects. It allows us to

1) Persist changes to our source code / trained models, so that every time we start our project, even on different machine, we resume from where we left last time.

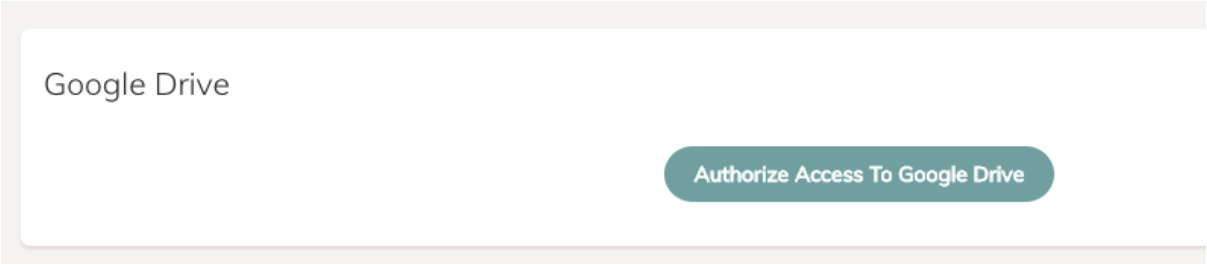
2) Super easy and convenient way to transfer datasets from our local machines or urls to our cloud machines where we run our projects.

Once you login to your console, you should now see a prompt to link your Clouderizer account to Google Drive.



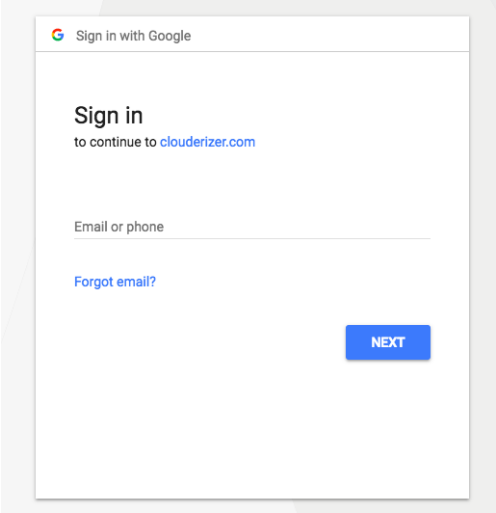
Prompt to link Google Drive account with Clouderizer

Alternatively, you can also go to Settings->Cloud Settings, and press “Authorize Access to Google Drive”

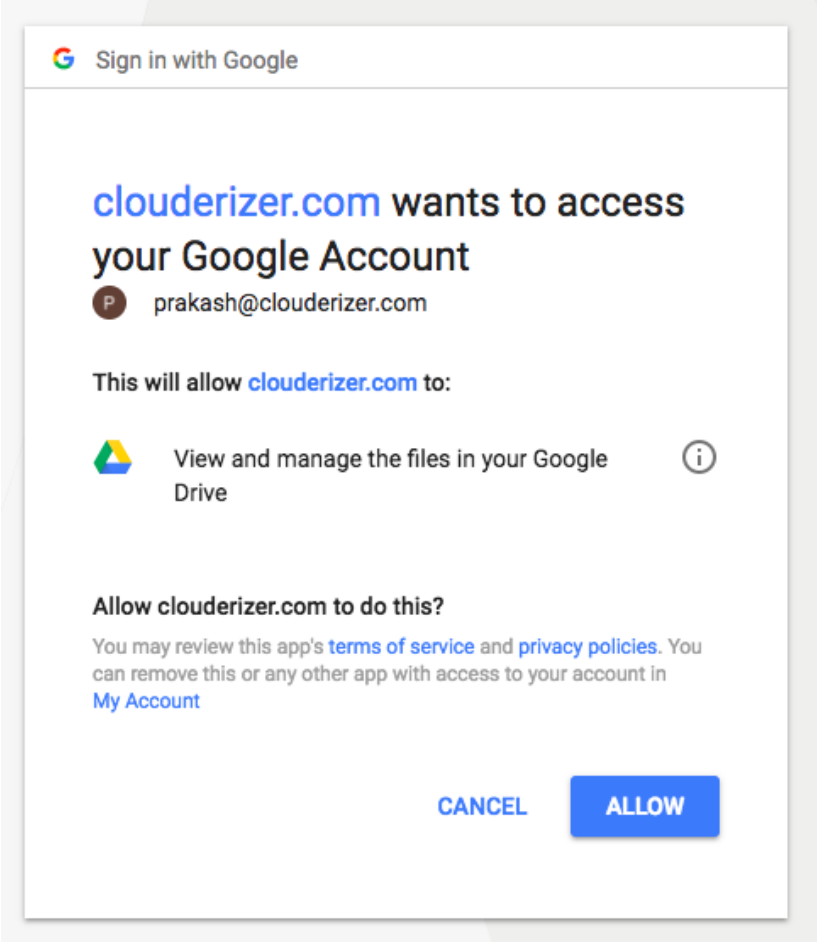


Cloud Settings for linking Google Drive account with Clouderizer

Once you click this, it will take you to Google Login screen. Here you should login with the Google Account that you wish to link.

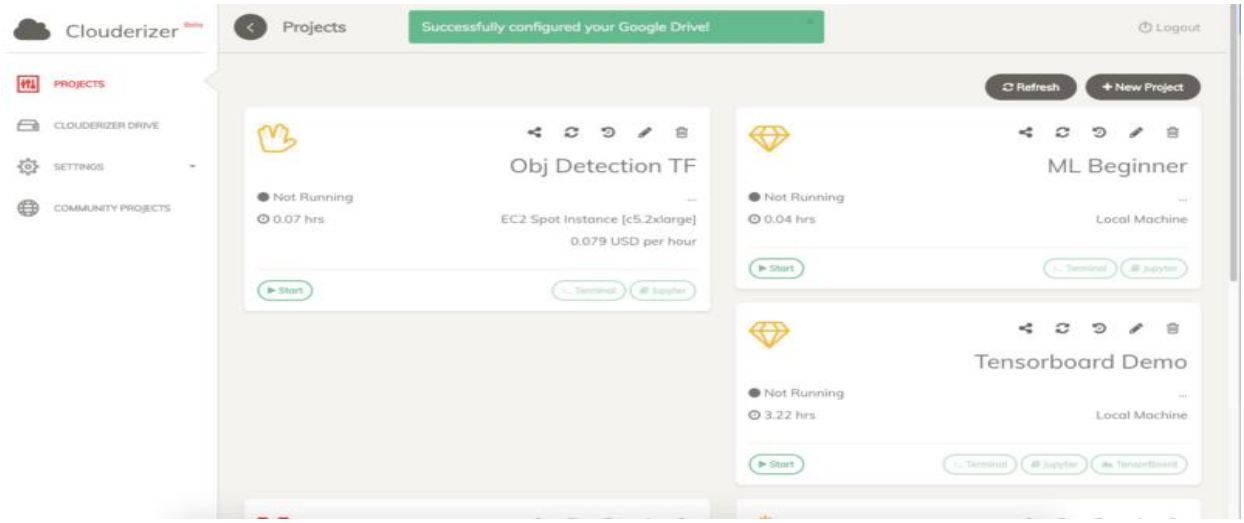


Once you login, you will be presented with Authorization screen from Google as below



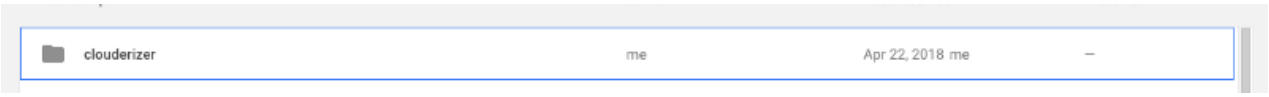
Authorization Screen to allow Clouderizer Google Drive access

Click “Allow” here. This should now take you back to Clouderizer console and you should see a success notification indicating that Google Drive integration was successful.



Google Drive successfully configured

Google Drive will now show a new folder **clouderizer**



Any new project that you create now, will automatically create a new sub-folder inside this. And all project folders will have **code** / **data** / **out** sub-folders as well, that correspond to code / data / out folders of your Clouderizer project.

For our existing users, who have projects created before Google Drive integration, next time you run your project, its folder will automatically appear on Google Drive. And in case you don't see any **data** or **out** folder for your projects, you can always create them manually on Google Drive and upload data in them. They will automatically sync with your machine once you run your projects.

Setup Fastai without Clouderizer

Create a new Python notebook in Google Colab to start the project

Installing Pytorch

As the default environment doesn't have Pytorch, We have to install this ourselves. Do remember that this has to be done every-time you connect to new VM. So don't delete the cells.

```
!pip install http://download.pytorch.org/whl/cu80/torch-0.3.0.post4-cp36-cp36m-linux_x86_64.whl && pip install torchvision
```

Installing Pytorch

Installing fastai library

Use pip to install fastai.

```
!pip install fastai
```

Installing fastai

Along with this, there is a library libSM which is missing so we had to install the same.

Downloading Data

We can download our cats Vs dogs dataset and unzip it using a couple of bash commands

```
[ ] !mkdir data && wget http://files.fast.ai/data/dogscats.zip && unzip dogscats.zip -d data/
```

Downloading Data

Known Issues:

1. While trying to connect to GPU runtime, it sometimes throws an error saying it can't connect. That's due to the heavy number of people trying to use the service Vs the number of GPU machines. As per the kaggle discussion shared earlier, they plan to add more GPU machines.
2. Sometimes, the runtime just dies intermittently. There may be many underlying causes for this.
3. The amount of ram available is ~13GB which is too good given it is free.But with large networks like our resnet in lesson 1, there are memory warnings most of the times. While trying the final full network with unfreeze and differential learning rates, I almost always ran into issues which I am suspecting is due to the memory.