ME:4145 Industrial Internet of Things

Lab #2
Using Google Sheets as a Cloud Database

Introduction

A key component of IIoT is data storage and access. Devices should be able to send, receive and share data generated by itself and other connected devices. While many commercial and open-source solutions exist, in this lab we will be using Google Sheets to serve as our cloud database.

What you will need

To complete this lab, you will need the following:

- Computer (RPi, laptop, desktop, etc.) with Python 3 installed.
- Gmail account

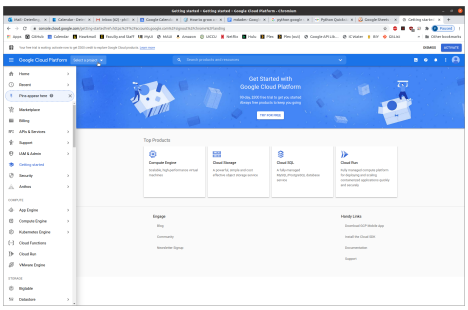
Setup

To setup Google Sheets Application Program Interface (API), we will be completing the following steps:

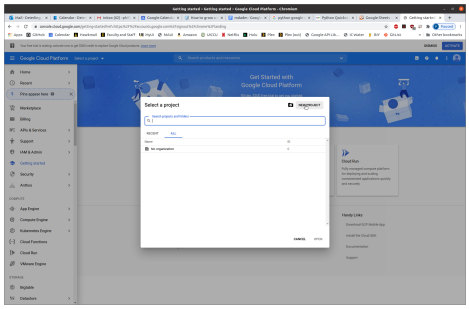
- Create a project on google cloud console
- Enable the Google Drive and Google Sheets APIs
- Create credentials
- Create and share a sheet
- Install modules with pip
- Connect to the google sheet via python code

Create a Project on Google Cloud Console

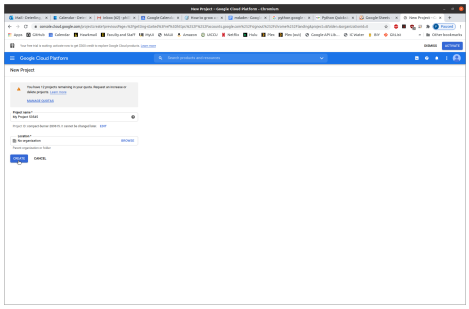
First thing we need to do is create a project on the Google Cloud Console. To do this go to the Google Cloud Console. In the upper left-hand corner click the Select a project dropdown and click NEW PROJECT. Enter a name for your new project and click CREATE. See Fig. 1 for these steps.



(a) Select a project.



(b) Name the project.



(c) Create new project.

Figure 1: Creating the project.

Enabling APIs

Next we need to enable the Google Drive and Google Sheets APIs. Click APIs & Services in the left dashboard as shown in Fig. 2. On this page select a project to use. Your recently created project should be visible. If not click SELECT PROJECT in the upper right corner as shown in Fig. 3. Go ahead and select that project by clicking on it and in the search bar (Fig. 4a) start typing "Google Drive API" and the API will pop up. Select this and click ENABLE as demonstrated in Fig. 4b.

Repeat this same process for the "Google Sheets API".

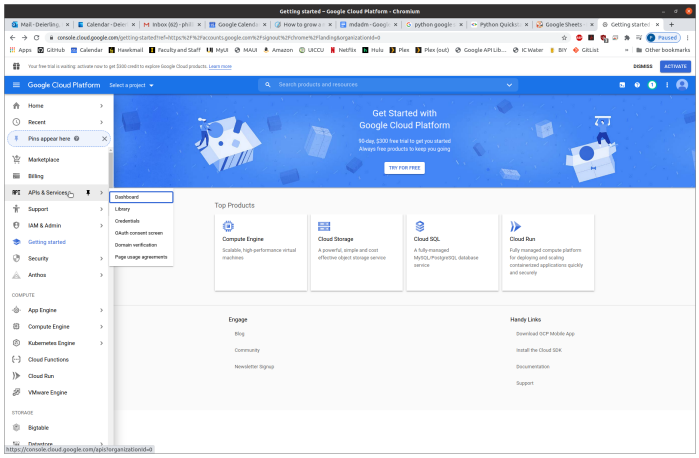


Figure 2: API dashboard.

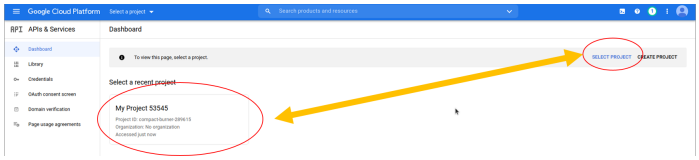
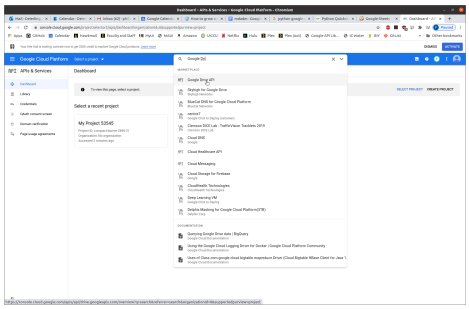
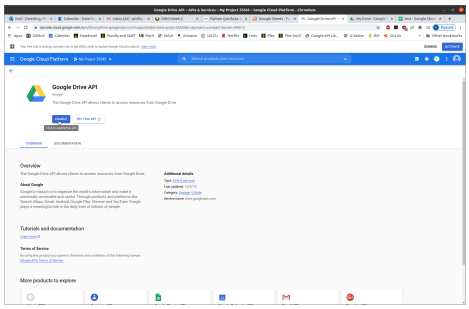


Figure 3: Select a project.



(a) API search.

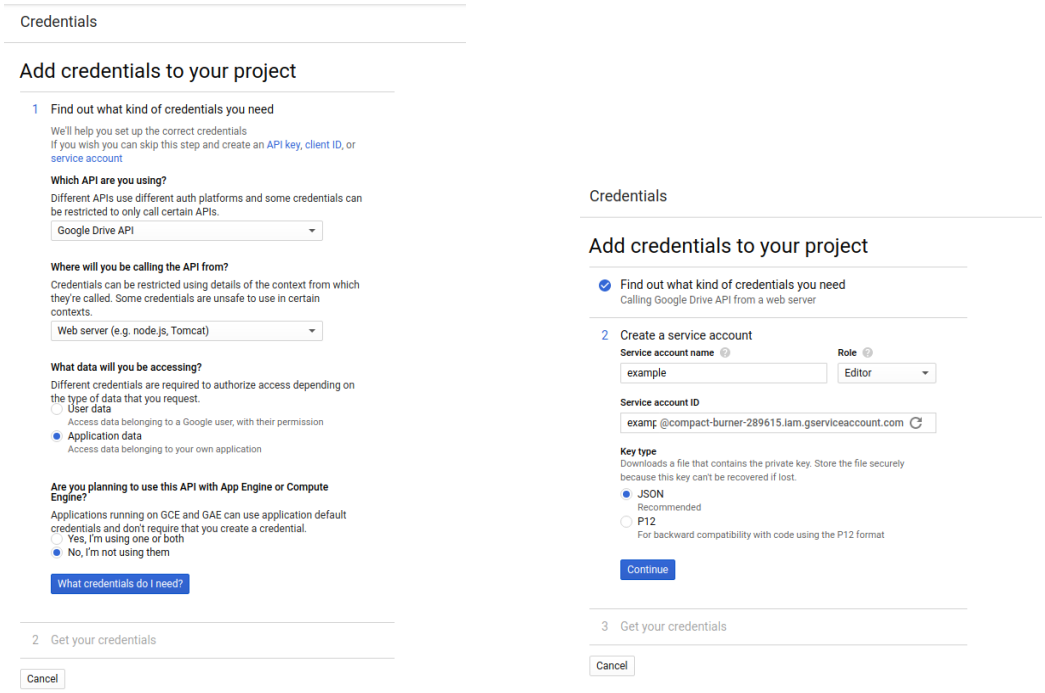


(b) Enable API.

Figure 4: Enabling Google Drive API.

Creating Credentials

Once you have added the APIs click CREATE CREDENTIALS in the right hand corner or on the left dashboard. Using the selections provided in Fig. 5a click What credentials do I need?. Here you need to give a service account name and assign a role. For role you want to select Project → Editor. For reference, see the example in Fig. 5b. Use the default JSON format and select Continue. This will download a .json file. Save this file somewhere that is confidential and easy for you to access. Feel free to rename this file something short if you wish, e.g., "creds.json".



(a) Settings. (b) Service account and role.

Figure 5: Creating credentials.

Creating and Sharing a Google Sheet

To create a database let's create a google sheet. In a new tab, go to your Google Drive and create and name a new sheet. Add a few rows and columns worth of information to the sheet. This can be random numbers, your class schedule, etc. Now open up the .json file that was downloaded previously and find the **client email**. Copy the email and share your google sheet with that email address making them an editor.

Install Modules with pip

In order to interact Python with Google Sheets we need to install a couple of Python modules. While there are more than one choice, we will be using the gspread and oauth2client modules in this lab. Install these modules using the following command

pip3 install gspread oauth2client

If you are using a Linux based OS, add "sudo" before pip3. If you are using something other than your RPi, make sure to install these modules on your RPi as well. We will be using them on the RPi later in future labs.

If your system says pip is not installed or recognized, you will need to install it. For Linux systems, install it with

sudo apt install python3-pip

If you are on Windows and Mac you may just need to enable pip. For further details please refer to the Python installation instructions used for your particular system or contact me directly.

Connect to Google Sheets via Python

Now that we have everything setup and installed, we can now access our sheet via a Python script. The example script "lab3_example.py" provided in the lab documents gives a basic introduction to reading and writing data to your sheet. Download, rename and modify this file as needed. Add print statements to see what is the code is doing. Write new data to the sheet and see if it updates the sheet in your browser. More information and documentation for gspread can be found here.

Task

Write a Python program that can write the date, time, and a random float between 0 and 100 to your database at a frequency of 1 data entry per 30 seconds with the newest data being on top. Your sheet should include a header for the date, time and random value. Furthermore, data must be formatted as follows:

1. date: year-mm-dd
2. time: hr:min:sec
3. round the random number to three decimal places

You sheet should contain data for at least **ONE** hour of time.

Submission Materials

Upload the following to ICON no later than the due date:

1. Python script/s
2. Copy of your google sheet containing your data