

Lecture 05

GEE Introduction:

setup, datasets, image visualization

2024-04-01

Sébastien Valade



UNIVERSIDAD NACIONAL
AUTÓNOMA DE
MÉXICO

1. Introduction

1. GEE overview
2. JavaScript API: Earth Engine Code Editor
3. Python API: Google Colaboratory

2. Setup GEE in GoogleColab

3. GEE quick start

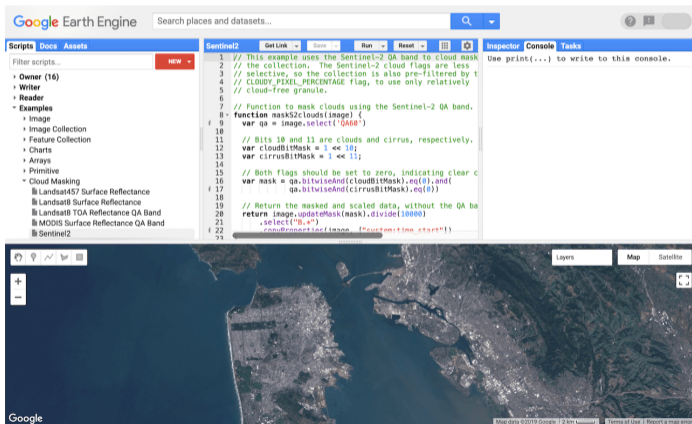
- **[Google Earth Engine](#)** (GEE) is a cloud-based computing platform for processing satellite imagery and other geospatial datasets.
- Provides access to:
 - large database of satellite imagery (including NASA, USGS, ESA, and other satellite missions)
 - large computational power needed to analyze those images
- Provides API (Application Programming Interfaces) for making requests to the servers in:
 - JavaScript ⇒ [Earth Engine Code Editor](#)
 - Python ⇒ [Google Colaboratory](#)

- [Google Earth Engine](#) (GEE) is a cloud-based computing platform for processing satellite imagery and other geospatial datasets.
- Provides access to:
 - large database of satellite imagery (including NASA, USGS, ESA, and other satellite missions)
 - large computational power needed to analyze those images
- Provides API (Application Programming Interfaces) for making requests to the servers in:
 - JavaScript ⇒ [Earth Engine Code Editor](#)
 - Python ⇒ [Google Colaboratory](#)

- [Google Earth Engine](#) (GEE) is a cloud-based computing platform for processing satellite imagery and other geospatial datasets.
- Provides access to:
 - large database of satellite imagery (including NASA, USGS, ESA, and other satellite missions)
 - large computational power needed to analyze those images
- Provides API (Application Programming Interfaces) for making requests to the servers in:
 - JavaScript ⇒ [Earth Engine Code Editor](#)
 - Python ⇒ [Google Colaboratory](#)

1. Earth Engine Code Editor (JavaScript API)

⇒ free web-based IDE (*Integrated Development Environment*) using the JavaScript API



1. Earth Engine Code Editor (JavaScript API)

⇒ free web-based IDE (*Integrated Development Environment*) using the JavaScript API

The screenshot shows the Google Earth Engine Code Editor interface. The main window is divided into several sections:

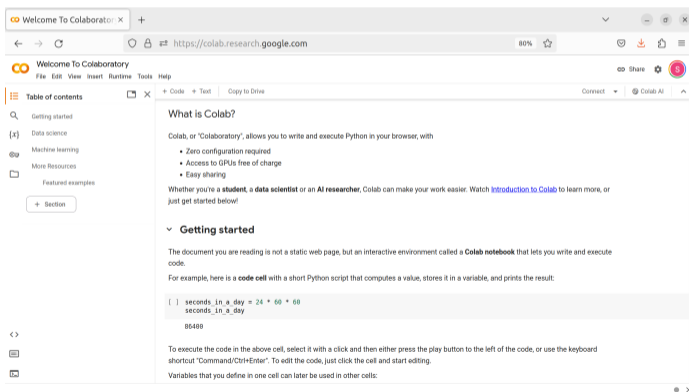
- Search for datasets or places:** A search bar at the top center.
- Script manager:** A sidebar on the left showing a list of scripts, including 'Owner (16)', 'Writer', 'Reader', 'Examples', 'Image', 'Feature Collection', 'Charts', 'Arrays', 'Primitive', and 'Cloud Masking'.
- API documentation:** A link in the top navigation bar.
- Asset manager:** A link in the top navigation bar.
- Code Editor:** The central area containing a JavaScript script for cloud masking. The code includes comments and function definitions for processing Sentinel-2 QA bands.
- Run the script:** A 'Run' button in the top navigation bar.
- Get a link (URL) to the script:** A 'Get Link' button in the top navigation bar.
- Help button:** A question mark icon in the top right corner.
- Feedback button:** A feedback icon in the top right corner.
- Task manager:** A panel on the right side of the code editor.
- Console output:** A panel on the right side of the code editor for displaying script output.
- Inspect locations, pixel values, objects on the map:** A panel on the right side of the code editor for inspecting map data.
- Geometry Tools:** A toolbar on the left side of the map.
- Zoom:** A zoom control on the left side of the map.
- Map:** The main map area showing a satellite view of a landscape.
- Layer manager:** A panel on the right side of the map for managing map layers.

2. Google Colaboratory (Python API)

⇒ free cloud-based Jupyter notebook environment for writing and executing Python code

⇒ avoids the need to set up a local development environment, i.e. software (libraries) & hardware (GPU)

⇒ provides access to GEE Python API, free GPU and TPU resources, enabling users to perform computationally intensive tasks



The screenshot shows the Google Colaboratory web interface in a browser. The address bar displays `https://colab.research.google.com`. The page title is "Welcome To Colaboratory". On the left, there is a "Table of contents" sidebar with links to "Getting started", "Data science", "Machine learning", "More Resources", and "Featured examples". The main content area is titled "What is Colab?" and contains the following text:

Colab, or "Colaboratory", allows you to write and execute Python in your browser, with

- Zero configuration required
- Access to GPUs free of charge
- Easy sharing

Whether you're a **student**, a **data scientist** or an **AI researcher**, Colab can make your work easier. Watch [Introduction to Colab](#) to learn more, or just get started below!

▼ **Getting started**

The document you are reading is not a static web page, but an interactive environment called a **Colab notebook** that lets you write and execute code.

For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result:

```
[ ] seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day

86400
```

To execute the code in the above cell, select it with a click and then either press the play button to the left of the code, or use the keyboard shortcut "Command/Ctrl+Enter". To edit the code, just click the cell and start editing.

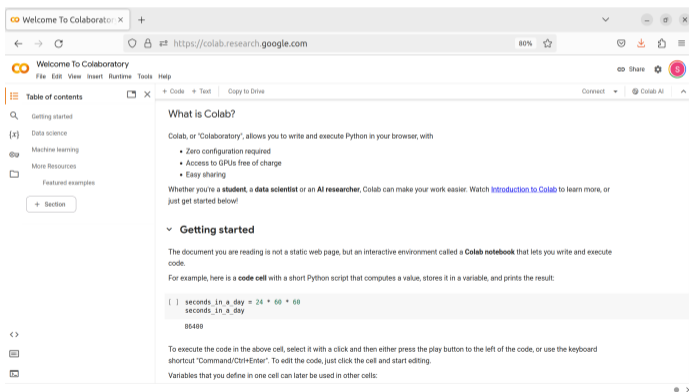
Variables that you define in one cell can later be used in other cells:

2. Google Colaboratory (Python API)

⇒ free cloud-based Jupyter notebook environment for writing and executing Python code

⇒ avoids the need to set up a local development environment, i.e. software (libraries) & hardware (GPU)

⇒ provides access to GEE Python API, free GPU and TPU resources, enabling users to perform computationally intensive tasks



The screenshot shows the Google Colaboratory web interface in a browser. The address bar displays `https://colab.research.google.com`. The page title is "Welcome To Colaboratory". The main content area is titled "What is Colab?" and contains the following text:

Colab, or "Colaboratory", allows you to write and execute Python in your browser, with

- Zero configuration required
- Access to GPUs free of charge
- Easy sharing

Whether you're a **student**, a **data scientist** or an **AI researcher**, Colab can make your work easier. Watch [Introduction to Colab](#) to learn more, or just get started below!

Getting started

The document you are reading is not a static web page, but an interactive environment called a **Colab notebook** that lets you write and execute code.

For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result:

```
[ ] seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day

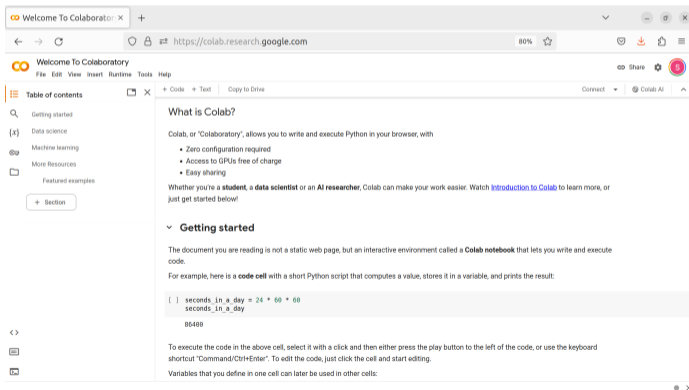
86400
```

To execute the code in the above cell, select it with a click and then either press the play button to the left of the code, or use the keyboard shortcut "Command/Ctrl+Enter". To edit the code, just click the cell and start editing.

Variables that you define in one cell can later be used in other cells:

2. Google Colaboratory (Python API)

- ⇒ free cloud-based Jupyter notebook environment for writing and executing Python code
- ⇒ avoids the need to set up a local development environment, i.e. software (libraries) & hardware (GPU)
- ⇒ provides access to GEE Python API, free GPU and TPU resources, enabling users to perform computationally intensive tasks



The screenshot shows the Google Colaboratory web interface in a browser. The address bar displays `https://colab.research.google.com`. The page title is "Welcome To Colaboratory". On the left, there is a "Table of contents" sidebar with links for "Getting started", "Data science", "Machine learning", "More Resources", and "Featured examples". The main content area is titled "What is Colab?" and contains the following text:

Colab, or "Colaboratory", allows you to write and execute Python in your browser, with

- Zero configuration required
- Access to GPUs free of charge
- Easy sharing

Whether you're a **student**, a **data scientist** or an **AI researcher**, Colab can make your work easier. Watch [Introduction to Colab](#) to learn more, or just get started below!

▼ **Getting started**

The document you are reading is not a static web page, but an interactive environment called a **Colab notebook** that lets you write and execute code.

For example, here is a **code cell** with a short Python script that computes a value, stores it in a variable, and prints the result:

```
[ ] seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day

86400
```

To execute the code in the above cell, select it with a click and then either press the play button to the left of the code, or use the keyboard shortcut "Command/Ctrl+Enter". To edit the code, just click the cell and start editing.

Variables that you define in one cell can later be used in other cells:

1. Introduction

2. Setup GEE in GoogleColab

1. Create a Google account
2. Create a Google Cloud project & enable GEE API
3. Register Google Cloud project for use with GEE
4. Access GEE in Colab

3. GEE quick start

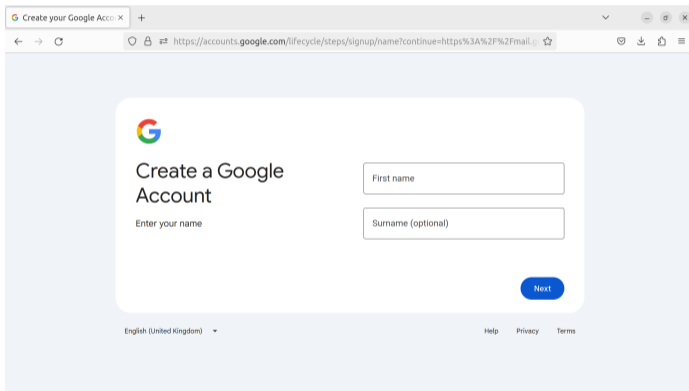
Nota Bene

The steps required by Google to access and use GEE APIs are regularly evolving.

⇒ the steps described are those required as of March-2024

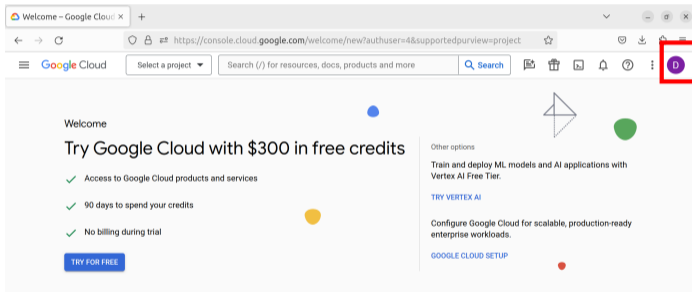
⇒ visit the [Earth Engine access guidelines](#) for the most up-to-date information

1. Create a Google account (if you have one, skip this step)



2. Create a Google Cloud project & enable GEE API

2.1 Access your account's Google Cloud Console



1. select your account here

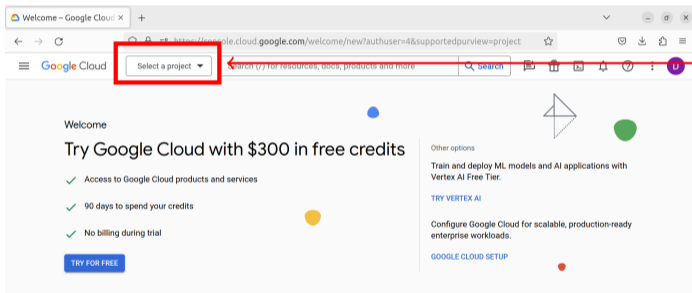
Popular getting started resources

Filter by [Web, mobile, game, storage](#) [Containers, VMs, hybrid/multi, move workload](#) [Data, AI/ML, SAP](#) [Maps, APIs](#) [General](#)

Pre-built solution templates

2. Create a Google Cloud project & enable GEE API

2.2 Create a new project in your [Google Cloud Console](#)



2. select a project

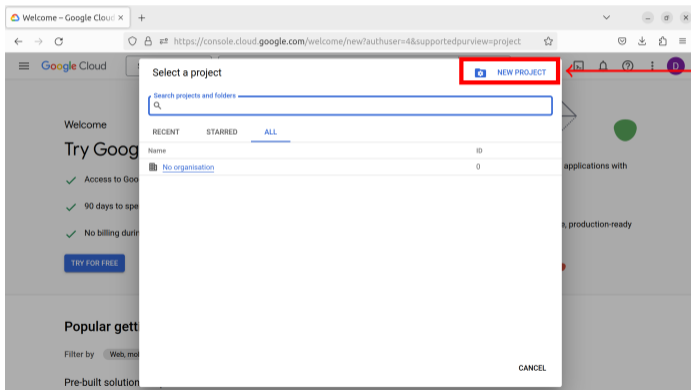
Popular getting started resources

Filter by [Web, mobile, game, storage](#) [Containers, VMs, hybrid/multi, move workload](#) [Data, AI/ML, SAP](#) [Maps, APIs](#) [General](#)

Pre-built solution templates

2. Create a Google Cloud project & enable GEE API

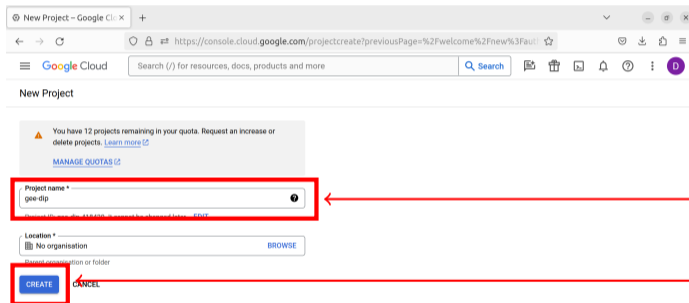
2.2 Create a new project in your [Google Cloud Console](#)



3. create new project

2. Create a Google Cloud project & enable GEE API

2.2 Create a new project in your [Google Cloud Console](#)

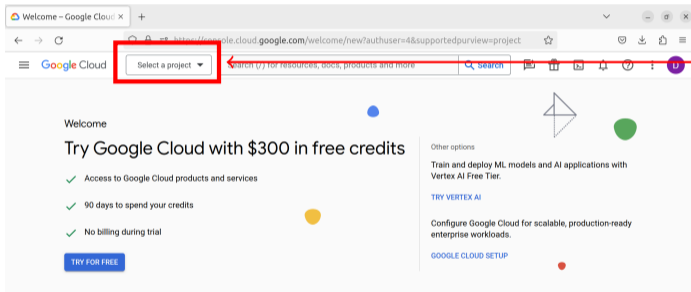


4. select project name

5. create project

2. Create a Google Cloud project & enable GEE API

2.3 Enable GEE API in the newly created project



6. select the project

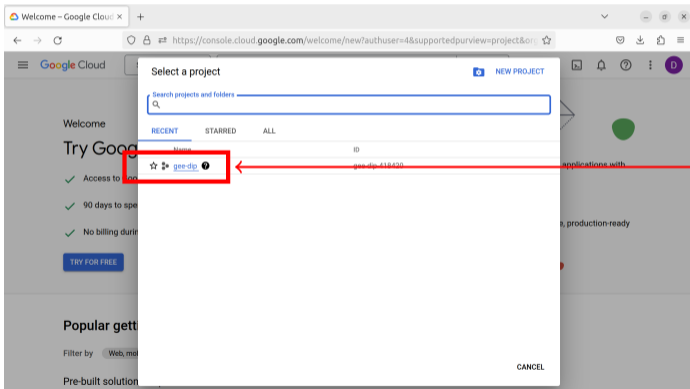
Popular getting started resources

Filter by [Web, mobile, game, storage](#) [Containers, VMs, hybrid/multi, move workload](#) [Data, AI/ML, SAP](#) [Maps, APIs](#) [General](#)

Pre-built solution templates 

2. Create a Google Cloud project & enable GEE API

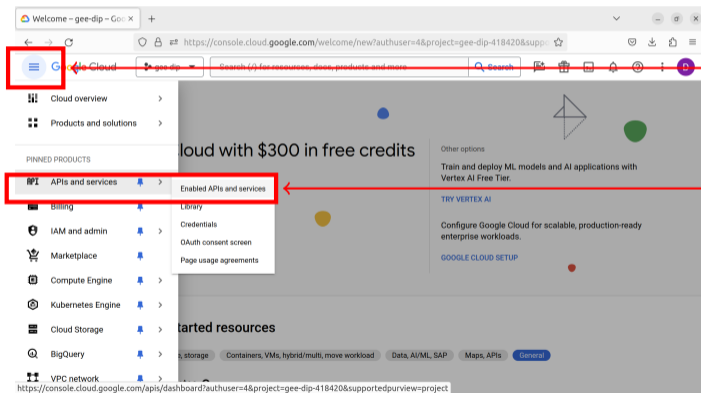
2.3 Enable GEE API in the newly created project



7. select the project

2. Create a Google Cloud project & enable GEE API

2.3 Enable GEE API in the newly created project

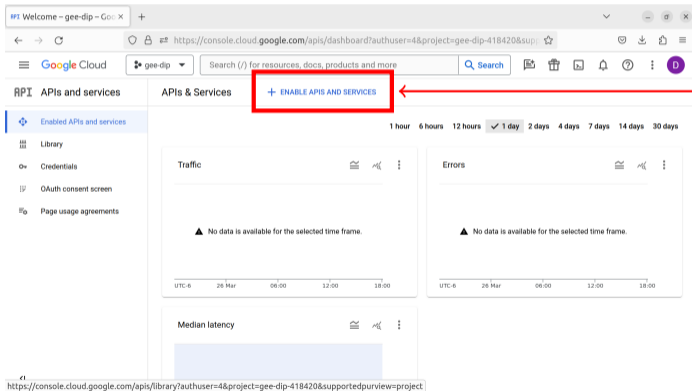


8. select navigation menu

9. "Enable API and services"

2. Create a Google Cloud project & enable GEE API

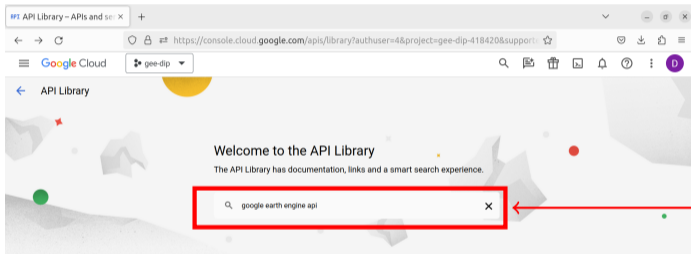
2.3 Enable GEE API in the newly created project



10. click here

2. Create a Google Cloud project & enable GEE API

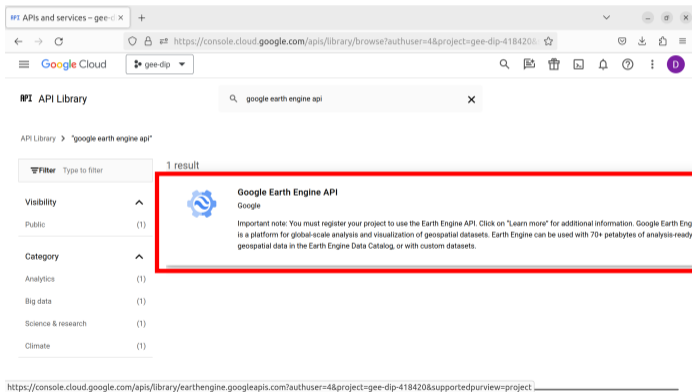
2.3 Enable GEE API in the newly created project



11. search/select "Google Earth Engine API"

2. Create a Google Cloud project & enable GEE API

2.3 Configure the newly created project



← 12. click here

2. Create a Google Cloud project & enable GEE API

2.3 Configure the newly created project

Product details

Google Earth Engine API

Geospatial insights for a more sustainable world.

ENABLE

[OVERVIEW](#) [PRICING](#) [SUPPORT](#) [RELATED PRODUCTS](#)

Overview

Important note: You must register your project to use the Earth Engine API. Click on "Learn more" for additional information.

Google Earth Engine is a platform for global-scale analysis and visualization of geospatial datasets. Earth Engine can be used with 70+ petabytes of analysis-ready geospatial data in the Earth Engine Data Catalog, or with custom datasets.

Additional details

Type: [SaaS & APIs](#)

Last product update: 01/08/2022

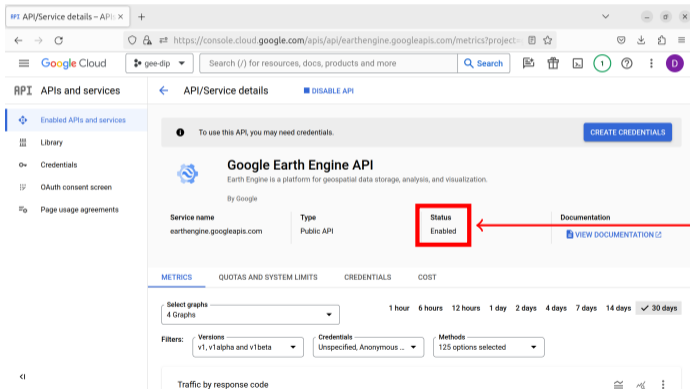
Category: [Big data](#), [Analytics](#), [Science & research](#), [Climate](#)

Service name: earthengine.googleapis.com

13. click "ENABLE"

2. Create a Google Cloud project & enable GEE API

2.3 Enable GEE API in the newly created project

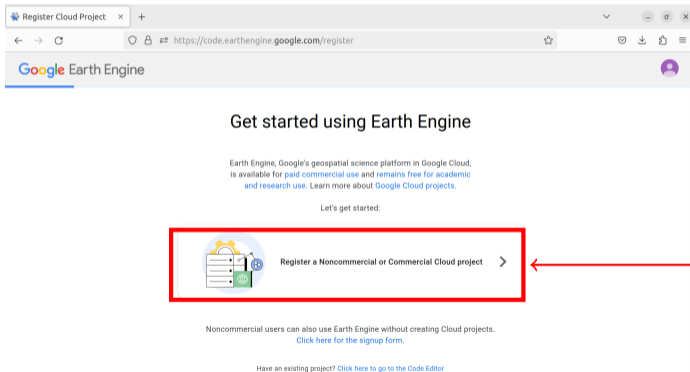


The screenshot shows the Google Cloud console interface for the Google Earth Engine API. The page title is "API/Service details - API: x". The URL is "https://console.cloud.google.com/apis/api/earthengine.googleapis.com/metrics?project=...". The left sidebar shows "APIs and services" with "Enabled APIs and services" selected. The main content area shows the "Google Earth Engine API" details. The "Status" is "Enabled", which is highlighted with a red box. A red arrow points from the text "GEE API is now enabled" to this box. The "Service name" is "earthengine.googleapis.com" and the "Type" is "Public API". There are tabs for "METRICS", "QUOTAS AND SYSTEM LIMITS", "CREDENTIALS", and "COST". The "METRICS" tab is active, showing a "Select graphs" dropdown with "4 Graphs" selected, and filters for "Versions" (v1, v1alpha and v1 beta), "Credentials" (Unspecified, Anonymous ...), and "Methods" (125 options selected).

GEE API is now enabled

3. Register Google Cloud project project for use with GEE

3.1 Access register page at <https://code.earthengine.google.com/register>



1. clic to register a project

3. Register Google Cloud project project for use with GEE

3.1 Register project



How do you want to use Earth Engine?

- Paid usage**
Commercial businesses, government operations. [See examples](#)
- Unpaid usage**
Non-profits, education, government research, training, media.
[See examples](#)

Project type*
Academia & Research

Please note: If you will be accessing Earth Engine as a customer of a Google Cloud Platform reseller, please contact your reseller for terms and pricing governing your use of Earth Engine.

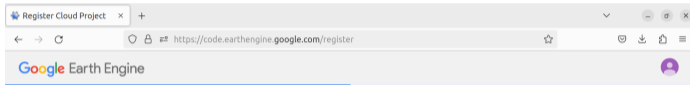
BACK

NEXT

2. select:
- Unpaid usage
 - Academia & Research

3. Register Google Cloud project project for use with GEE

3.1 Register project



Create or choose a Cloud Project to register

Create a new project in Google Cloud, or choose one you are authorized to access to enable the API:

Create a new Google Cloud Project

Choose an existing Google Cloud Project

Project

Refresh

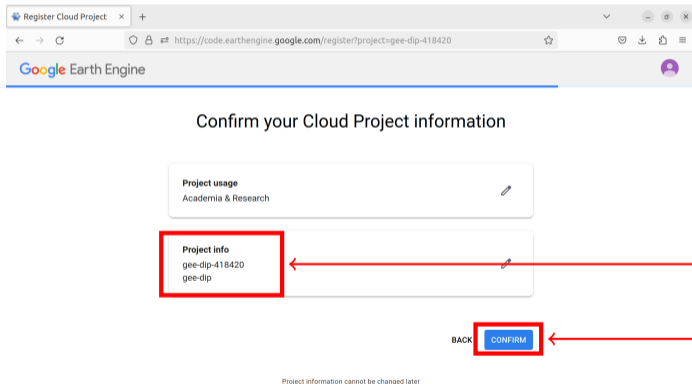
All Cloud Projects

gee-dip	gee-dip-418420
---------	----------------

← 3. select project

3. Register Google Cloud project project for use with GEE

3.1 Register project

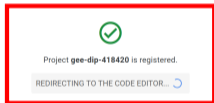


Project info:
- **project-id**
- project-name

4. Confirm

3. Register Google Cloud project project for use with GEE

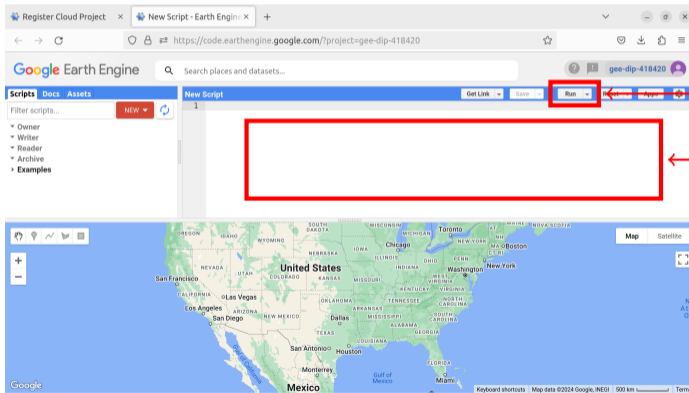
3.2 Register project



5. register successful
⇒ redirecting to Code Explorer
⇒ JavaScript IDE

3. Register Google Cloud project for use with GEE

3.4 Try accessing GEE in **Code Editor** (JavaScript IDE)

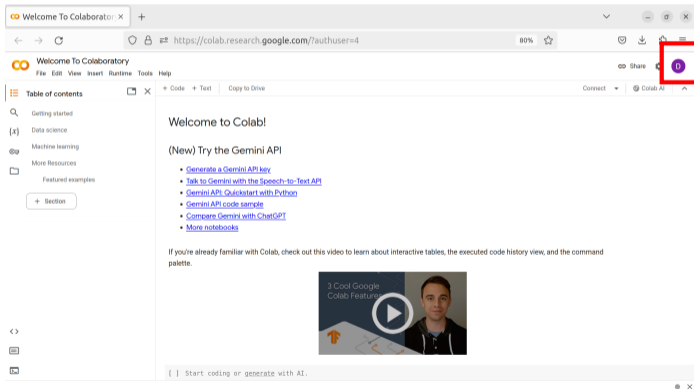


7. clic "Run" to execute code

6. type code in JavaScript
⇒ ex: official tutorials

4. Access GEE in Colab

4.1 Access Google Colaboratory at <https://colab.research.google.com/>

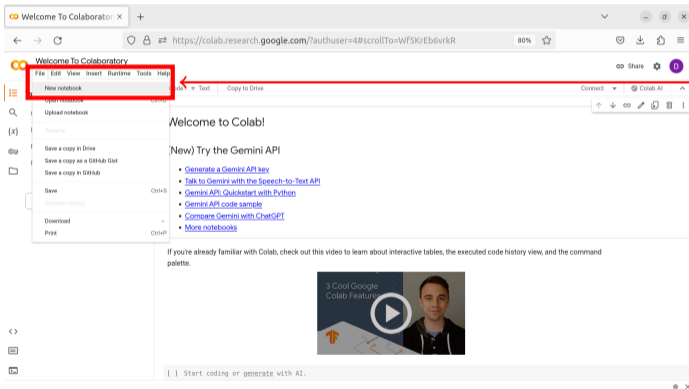


The screenshot shows a web browser window with the URL `https://colab.research.google.com/?authuser=4`. The page title is "Welcome To Colaboratory". In the top right corner, there is a "Share" button and a circular profile icon containing the letter "D". This profile icon is highlighted with a red square, and a red arrow points from the text "1. select your account here" to it. The main content area of the page displays "Welcome to Colab!" and "(New) Try the Gemini API" with several links. At the bottom, there is a video player with the title "3 Cool Google Colab Features".

1. select your account here

4. Access GEE in Colab

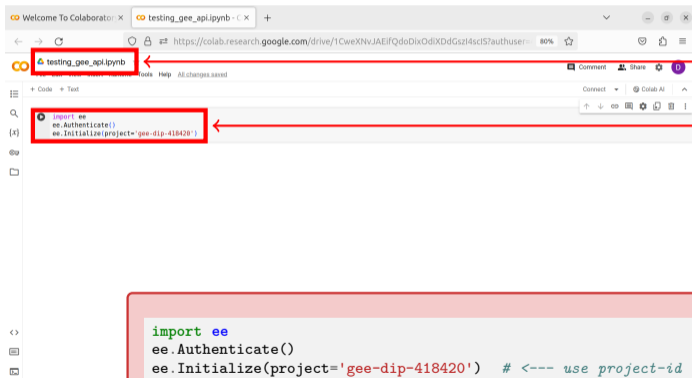
4.2 Create new notebook



2. File - New notebook

4. Access GEE in Colab

4.3 Import ee library & initialize with **project-id** (in which GEE API was enabled)



The screenshot shows a Google Colaboratory notebook interface. The notebook name is "testing_gee_api.ipynb", which is highlighted with a red box and an arrow pointing to the text "3. rename notebook (optional)". Below the notebook name, there is a code cell containing the following Python code:

```
import ee
ee.Authenticate()
ee.Initialize(project='gee-dip-418420')
```

 This code cell is also highlighted with a red box and an arrow pointing to the text "4. import Earth Engine lib (ee), initialize using **project-id** (NOT project-name!), and execute cell".

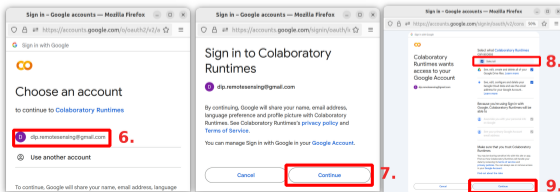
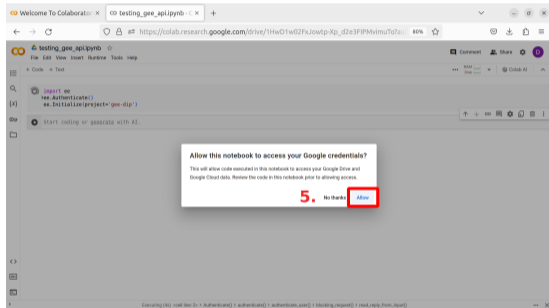
3. rename notebook (optional)

4. import Earth Engine lib (ee), initialize using **project-id** (NOT project-name!), and execute cell

```
import ee
ee.Authenticate()
ee.Initialize(project='gee-dip-418420') # <--- use project-id (NOT project-name!)
```

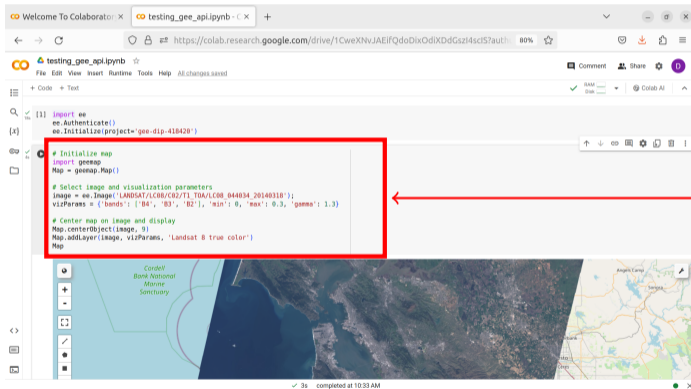
4. Access GEE in Colab

4.3 Execute cell & give authorizations in pop-up windows



4. Access GEE in Colab

4.4 Start coding with GEE in Colab !



```
[1]: import ee
     ee.Authenticate()
     ee.Initialize(project='gee-dip-418428')

# Initialize map
import geemap
Map = geemap.Map()

# Select image and visualization parameters
image = ee.Image('LANDSAT/LC08/C02/T1_TOA/LC08_044034_20140318');
vizParams = {'bands': ['B4', 'B3', 'B2'], 'min': 0, 'max': 0.3, 'gamma': 1.3}

# Center map on image and display
Map.centerObject(image, 9)
Map.addLayer(image, vizParams, 'Landsat 8 true color')
Map
```

3s completed at 10:33 AM

10. start coding!
(ex: Tutorial intro-to-python-api)

1. Introduction

2. Setup GEE in GoogleColab

3. GEE quick start

1. GEE data catalog

2. GEE data model

3. Jumpstart into image visualization

GEE's public [data archive](#) includes >40 years of **satellite imagery** expanded daily:

1. **Landsat** collections

⇒ [NASA/USGS Program](#), since 1972

⇒ 9 generation of satellites (polar-orbiting):

- **Landsat-1** (1972) - **Landsat-3** (1978): optical & infrared imaging (VIS/NIR)
- **Landsat-4** (1982) - **Landsat-9** (2021): optical & infrared imaging (VIS/NIR/SWIR/TIR)

⇒ GEE archive includes:

- Landsat 1-5	(1972–1999)	Sensor: MSS (Multispectral Scanner)
- Landsat 4	(1982–1993)	Sensor: TM (Thematic Mapper)
- Landsat 5	(1984–2012)	Sensor: TM (Thematic Mapper)
- Landsat 7	(1999–2021)	Sensor: ETM+ (Enhanced Thematic Mapper Plus)
- Landsat 8	(2013–Present)	Sensor: OLI/TIRS (Op. Land Imager / Therm. Infrared Sensor)
- Landsat 9	(2021–Present)	Sensor: OLI/TIRS (Op. Land Imager / Therm. Infrared Sensor)

GEE's public [data archive](#) includes >40 years of **satellite imagery** expanded daily:

2. Sentinel collections

⇒ [ESA/Copernicus Program](#), since 2014

⇒ constellation of satellites consisting comprising various sensors:

- **Sentinel-1**: radar imaging (C-band SAR)
- **Sentinel-2**: optical & infrared imaging (VIS/SWIR)
- **Sentinel-3**: optical & infrared imaging (VIS/SWIR/TIR)
- **Sentinel-5P**: ultra-violet, optical, infrared imaging (UV/VIS/NIR/SWIR)

⇒ GEE archive includes:

- | | | |
|---------------|----------------|--------------------------------------------------------------------------|
| - Sentinel 1 | (2014–Present) | Sensor: SAR (C-band), GRD scenes (Ground Range Detected) |
| - Sentinel 2 | (2015–Present) | Sensor: MSI (Multispectral Instrument) |
| - Sentinel 3 | (2016–Present) | Sensor: OLCI (Ocean and Land Color Instrument) |
| - Sentinel 5P | (2018–Present) | Sensor: TROPOMI (TROPOspheric Monitoring Instrument) |

GEE's public [data archive](#) includes >40 years of **satellite imagery** expanded daily:

3. **MODIS** collections

- ⇒ NASA's "Moderate Resolution Imaging Spectroradiometer"
- ⇒ sensor on board 2 satellites: Terra (since 1999) & Aqua (since 2002)
- ⇒ GEE archive includes: daily surface spectral reflectances from MODIS, as well as several derived products (e.g., vegetation indices, snow cover, etc)

4. High-Resolution Imagery

- ⇒ GEE archive currently includes: [Planet SkySat](#) Multispectral imagery, and aerial imagery acquired by the NAIP (*National Agriculture Imagery Program*) during the agricultural growing seasons in the continental U.S.

GEE's public [data archive](#) includes >40 years of **satellite imagery** expanded daily:

3. **MODIS** collections

- ⇒ NASA's "Moderate Resolution Imaging Spectroradiometer"
- ⇒ sensor on board 2 satellites: Terra (since 1999) & Aqua (since 2002)
- ⇒ GEE archive includes: daily surface spectral reflectances from MODIS, as well as several derived products (e.g., vegetation indices, snow cover, etc)

4. **High-Resolution Imagery**

- ⇒ GEE archive currently includes: [Planet SkySat](#) Multispectral imagery, and aerial imagery acquired by the NAIP (*National Agriculture Imagery Program*) during the agricultural growing seasons in the continental U.S.

In addition to satellite imagery, GEE also includes **other scientific datasets**:

1. Digital Elevation Models (DEMs) collections

⇒ DEMs describe Earth's topography

⇒ GEE archive includes:

- global DEMs: [SRTM DEM](#) (NASA's Shuttle Radar Topography Mission) data at 30-meter resolution, [Copernicus DEM](#) (ESA) data at 30-meter resolution, ALOS
- regional DEMs at higher resolutions

2. Thematic datasets:

- [Surface Temperature](#): includes land and sea surface temperature products derived from several spacecraft sensors, including MODIS, ASTER, and AVHRR, in addition to raw Landsat thermal data
- [Climate](#): includes climate models generate both long-term climate predictions and historical interpolations of surface variables
- [Atmospheric](#): includes ozone data from NASA's TOMS and OMI instruments and the MODIS Monthly Gridded Atmospheric Product

In addition to satellite imagery, GEE also includes **other scientific datasets**:

1. Digital Elevation Models (DEMs) collections

⇒ DEMs describe Earth's topography

⇒ GEE archive includes:

- global DEMs: [SRTM DEM](#) (NASA's Shuttle Radar Topography Mission) data at 30-meter resolution, [Copernicus DEM](#) (ESA) data at 30-meter resolution, ALOS
- regional DEMs at higher resolutions

2. Thematic datasets:

- [Surface Temperature](#): includes land and sea surface temperature products derived from several spacecraft sensors, including MODIS, ASTER, and AVHRR, in addition to raw Landsat thermal data
- [Climate](#): includes climate models generate both long-term climate predictions and historical interpolations of surface variables
- [Atmospheric](#): includes ozone data from NASA's TOMS and OMI instruments and the MODIS Monthly Gridded Atmospheric Product

In addition to satellite imagery, GEE also includes **other scientific datasets**:

2. Thematic datasets (continued):

- **Weather**: includes forecasted and measured conditions over short periods of time, including precipitation, temperature, humidity, and wind, and other variables. Includes in particular NOAA's Global Forecast System (GFS) and the NCEP Climate Forecast System (CFSv2)
- **Land Cover**: includes the physical landscape in terms of land cover classes such as forest, grassland, and water
- **Cropland**: includes a number of cropland data products
- **Other Geophysical Data**: includes data from other satellite image sensors

The GEE data model revolves around the following components:

- **Image objects**

- ⇒ `ee.Image`

- ⇒ Image objects represent raster data (i.e., satellite imagery, climate data, or any gridded data)

- ⇒ Image objects consist of one or more bands, where each band represents a different type of information (e.g., red, green, blue bands for RGB imagery)

- **Geometry objects**

- ⇒ `ee.Geometry`

- ⇒ Geometry objects represent vector data (i.e., points, lines, or polygons)

- ⇒ Geometry objects support different geometries: `Point` (a list of coordinates in some projection), `LineString` (a list of points), `LinearRing` (a closed `LineString`), `Polygon` (a list of `LinearRings` where the first is a shell and subsequent rings are holes), as well as `MultiPoint`, `MultiLineString`, and `MultiPolygon`

- **Feature objects**

- ⇒ `ee.Feature`

- ⇒ Feature objects are `Geometry` objects with attributes

- ⇒ Feature objects store a `Geometry` object (or null) and a *properties* property storing a dictionary of other properties

- **Collection objects**

- ⇒ Collections are groups of `Image` or `Feature` objects

- ⇒ `ee.ImageCollection`: group of `Image` objects, which can be organized and filtered based on various criteria such as date, metadata, or spatial location

- ⇒ `ee.FeatureCollection`: group of `Feature` objects

3.3. Jumpstart into image visualization

```

# Initialize
import geemap
import ee
ee.Authenticate()
ee.Initialize(project='gee-dip-418420') # Initialize using project-id with enabled GEE API
Map = geemap.Map() # Initialize map

# Select image and visualization parameters
image = ee.Image('LANDSAT/LC08/C02/T1_TOA/LC08_026047_20200116'); # Landsat 8 Top of Atmosphere (TOA) image over Popocatepetl
vis_param = {'bands': ['B4', 'B3', 'B2'], 'min': 0, 'max': 0.3, 'gamma': 1.3} # Select bands for true color RGB

# Center map on image and display
Map.centerObject(image, 9)
Map.addLayer(image, vizParams, 'Landsat 8 true color')
Map

```

