

Lecture 05

Google Earth Engine (GEE)

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

1.1. GEE overview

- **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](#)

1.1. GEE overview

- **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](#)

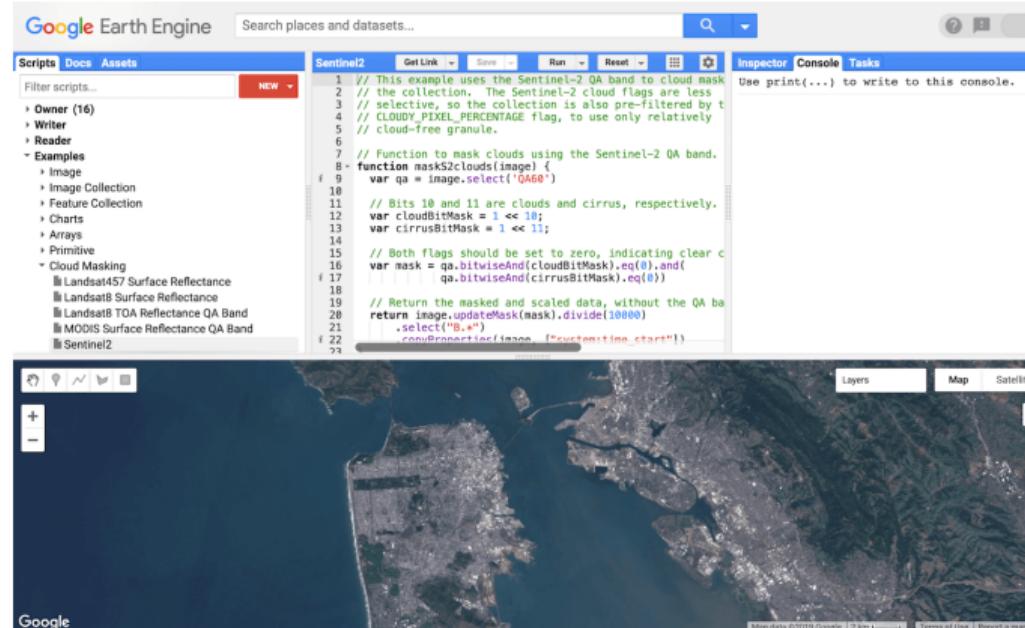
1.1. GEE overview

- **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](#)

1.2. JavaScript API: Earth Engine Code Editor

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. At the top, there's a navigation bar with 'Google Earth Engine' and a search bar labeled 'Search places and datasets...'. Below the search bar are buttons for 'Get Link', 'Save', 'Run', 'Reset', and a dropdown menu. To the right of these are 'Inspector', 'Console', and 'Tasks' tabs, with the 'Console' tab selected. A message in the console says 'Use print(...) to write to this console.' The main area contains a code editor with the following JavaScript code:

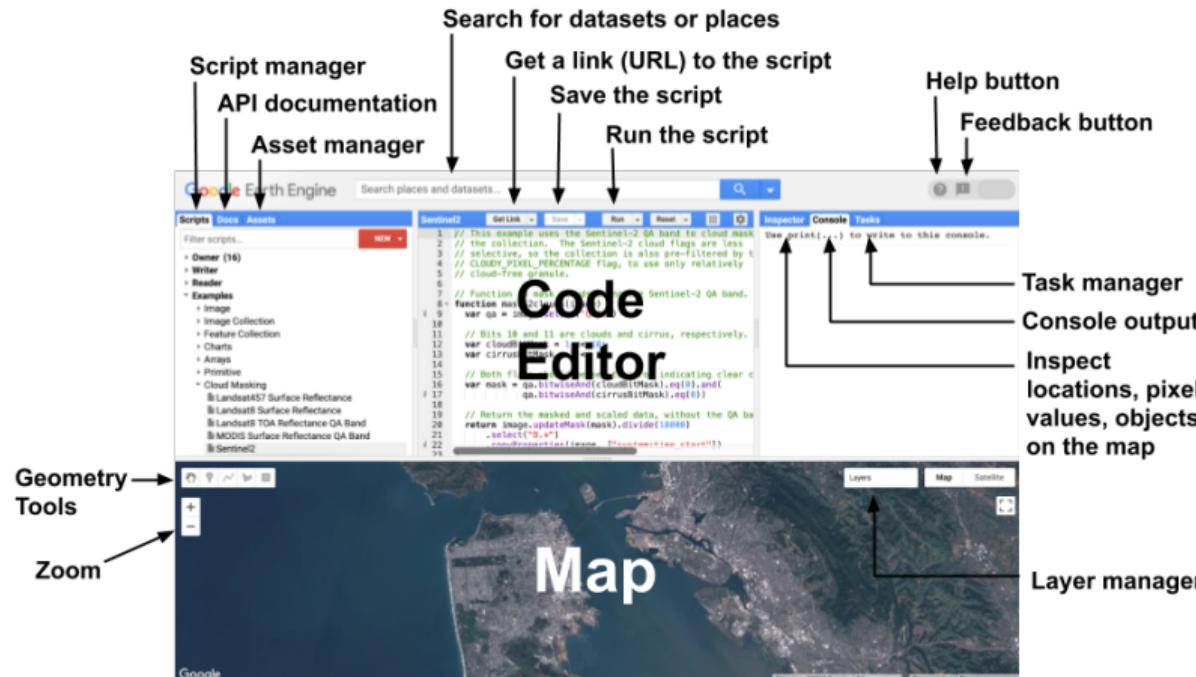
```
1 // This example uses the Sentinel-2 QA band to cloud mask
2 // the collection. The Sentinel-2 cloud flags are less
3 // selective, so the collection is also pre-filtered by t
4 // CLOUDY_PIXEL_PERCENTAGE flag, to use only relatively
5 // cloud-free granule.
6
7 // Function to mask clouds using the Sentinel-2 QA band.
8 function maskS2clouds(image) {
9   var qa = image.select('QA6B');
10
11   // Bits 10 and 11 are clouds and cirrus, respectively.
12   var cloudBitMask = 1 << 10;
13   var cirrusBitMask = 1 << 11;
14
15   // Both flags should be set to zero, indicating clear c
16   var mask = qa.bitwiseAnd(cloudBitMask).eq(0).and(
17     qa.bitwiseAnd(cirrusBitMask).eq(0));
18
19   // Return the masked and scaled data, without the QA ba
20   return image.updateMask(mask).divide(10000)
21     .select(['B.*'])
22     .copyProperties(image, ['customtime_start']);
23 }
```

Below the code editor is a map viewer showing a satellite image of a coastal area. The map includes zoom controls (+, -, ×), a location pin, and a compass rose. At the bottom of the map are buttons for 'Layers', 'Map', and 'Satellite'. The bottom of the interface features a footer with 'Map data ©2019 Google | 2 km | Terms of Use | Report a map error'.

1.2. JavaScript API: Earth Engine Code Editor

1. Earth Engine Code Editor (JavaScript API)

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



1.3. Python API: Google Colaboratory

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 a web browser window for 'Welcome To Colaboratory' at <https://colab.research.google.com>. The browser has a standard top bar with tabs, address bar, and zoom controls. On the left, there's a sidebar with a 'Table of contents' section containing links to 'Getting started', 'Data science', 'Machine learning', 'More Resources', and 'Featured examples'. Below this is a 'Section' button. The main content area has a heading 'What is Colab?' followed by text explaining it allows writing and executing Python in a browser. It lists benefits like zero configuration required, access to GPUs, and easy sharing. A note for students, data scientists, and AI researchers is present. Under the 'Getting started' section, it says the document is an interactive environment for writing and executing code. An example code cell is shown:

```
[ ] seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day
86400
```

Instructions for executing the code are provided, along with a note that variables defined in one cell can be used in others.

1.3. Python API: Google Colaboratory

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 'Welcome To Colaboratory' page at https://colab.research.google.com. On the left, there's a sidebar with a 'Table of contents' section containing links to 'Getting started', 'Data science', 'Machine learning', 'More Resources', and 'Featured examples'. Below this is a '+ Section' button. The main content area has a heading 'What is Colab?'. It explains that Colab, or 'Colaboratory', allows you to write and execute Python in your browser, with zero configuration required, access to GPUs free of charge, and easy sharing. It also mentions that Colab can make work easier for students, data scientists, and AI researchers. A 'Getting started' section is expanded, showing a code cell with the following Python script:

```
[ ] seconds_in_a_day = 24 * 60 * 60
seconds_in_a_day
86400
```

Below the code cell, instructions say to select the cell and press Command/Control+Enter to execute it. A note at the bottom states that variables defined in one cell can be used in other cells.

1.3. Python API: Google Colaboratory

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 interface in a web browser. The title bar says "Welcome To Colaboratory". The address bar shows the URL <https://colab.research.google.com>. The main content area displays the "What is Colab?" page. On the left, there is a sidebar with a "Table of contents" section containing links to "Getting started", "Data science", "Machine learning", "More Resources", and "Featured examples". Below this is a "Section" button. The main content area has a heading "What is Colab?", followed by a paragraph about Colab allowing you to write and execute Python in your browser. It lists three bullet points: "Zero configuration required", "Access to GPUs free of charge", and "Easy sharing". A note below states that whether you're a student, a data scientist or an AI researcher, Colab can make your work easier. It encourages watching the [Introduction to Colab](#) video. A "Getting started" section is expanded, showing a code cell with Python code to calculate seconds in a day. The code is:

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

 and the output is:

```
86400
```

. A note below says: "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." At the bottom, it says: "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

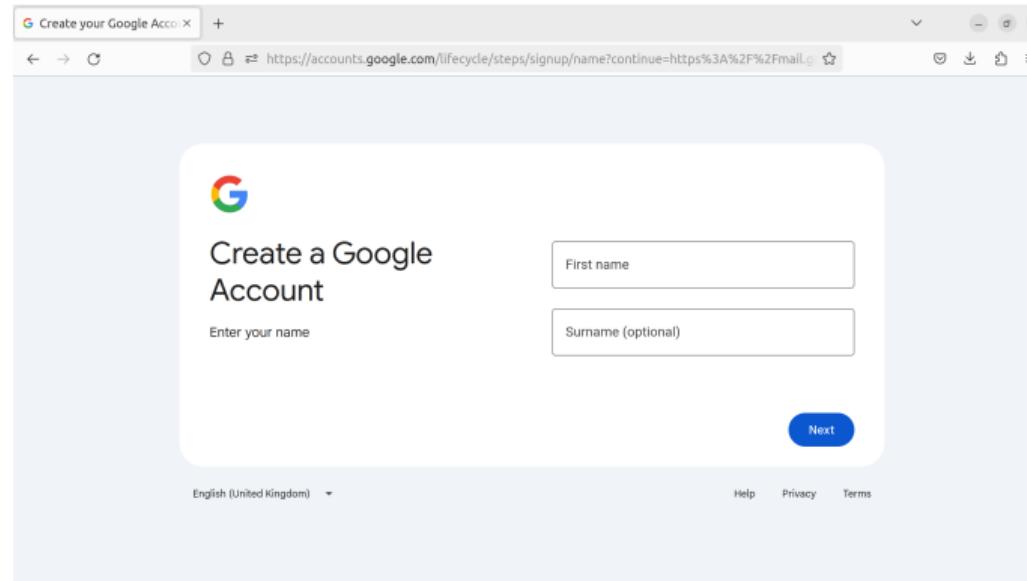
The steps required by Google to access and use GEE APIs are regularly evolving.

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

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

2.1. Create a Google account

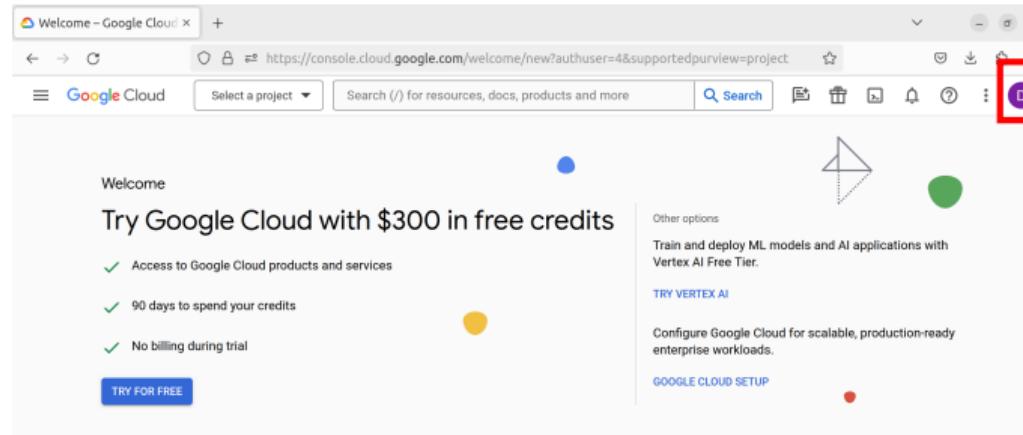
1. Create a Google account (if you have one, skip this step)



2.2. Create a Google Cloud project & enable GEE API

2. Create a Google Cloud project & enable GEE API

2.1 Access your account's Google Cloud Console



1. select your account here

Welcome

Try Google Cloud with \$300 in free credits

- ✓ Access to Google Cloud products and services
- ✓ 90 days to spend your credits
- ✓ No billing during trial

[TRY FOR FREE](#)

Other options

Train and deploy ML models and AI applications with Vertex AI Free Tier.

[TRY VERTEX AI](#)

Configure Google Cloud for scalable, production-ready enterprise workloads.

[GOOGLE CLOUD SETUP](#)

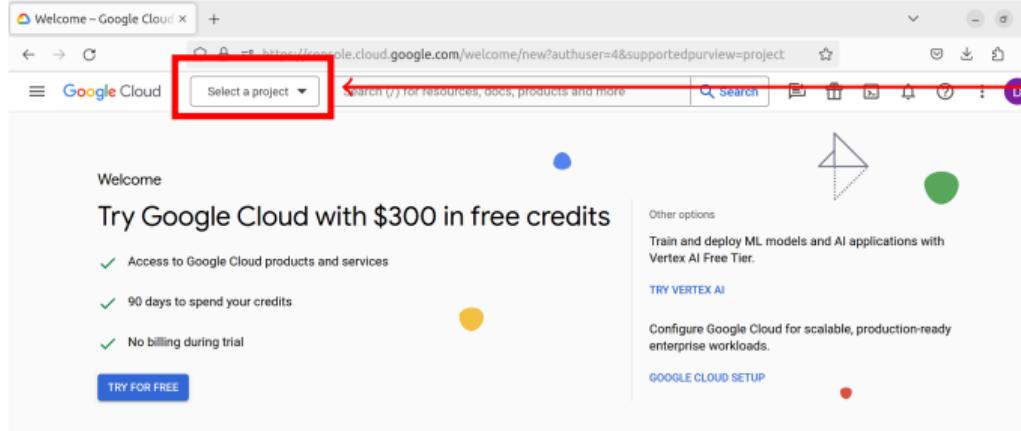
Popular getting started resources

Filter by: Web, mobile, game, storage | Containers, VMs, hybrid/multi, move workload | Data, AI/ML, SAP | Maps, APIs | General

Pre-built solution templates ⓘ

2.2. Create a Google Cloud project & enable GEE API

2. Create a Google Cloud project & enable GEE API

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

2. select a project

Welcome

Try Google Cloud with \$300 in free credits

- ✓ Access to Google Cloud products and services
- ✓ 90 days to spend your credits
- ✓ No billing during trial

TRY FOR FREE

Other options

Train and deploy ML models and AI applications with Vertex AI Free Tier.

TRY VERTEX AI

Configure Google Cloud for scalable, production-ready enterprise workloads.

GOOGLE CLOUD SETUP

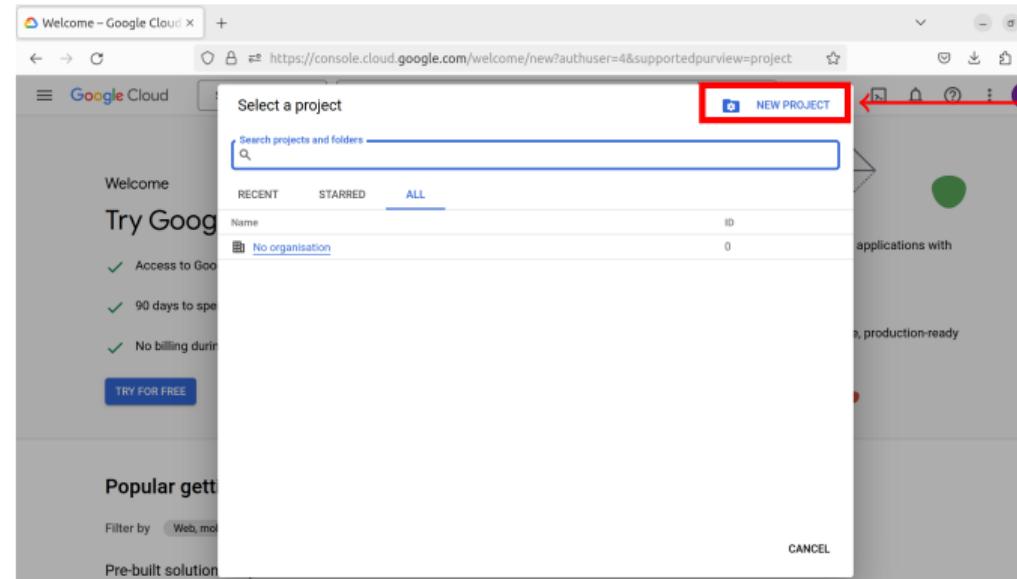
Popular getting started resources

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

Pre-built solution templates

2.2. Create a Google Cloud project & enable GEE API

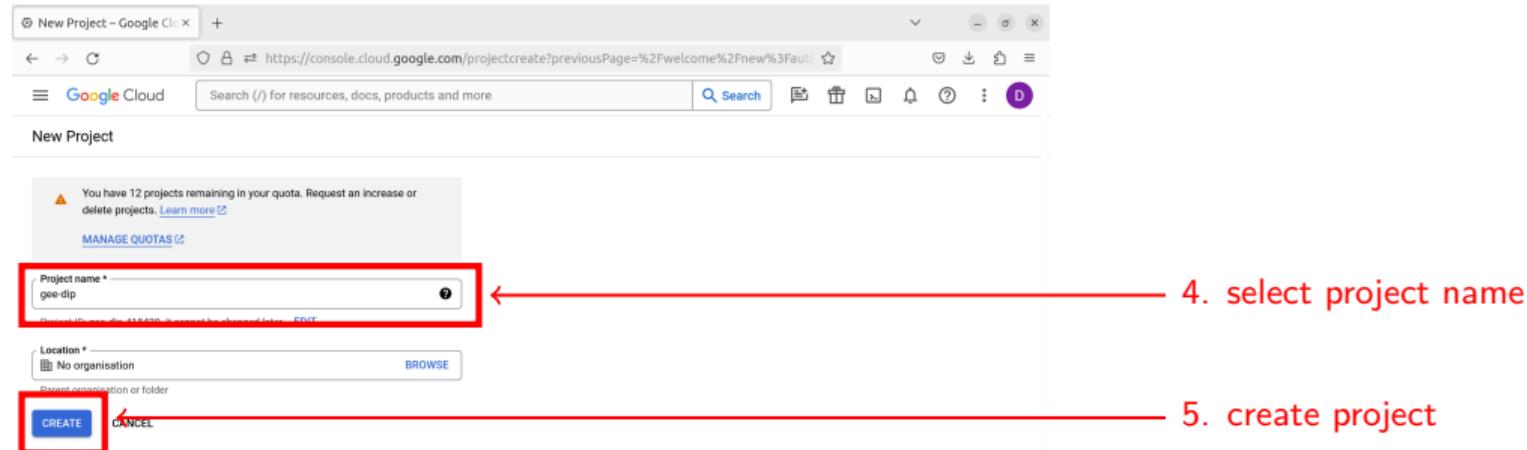
2. Create a Google Cloud project & enable GEE API

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

3. create new project

2.2. Create a Google Cloud project & enable GEE API

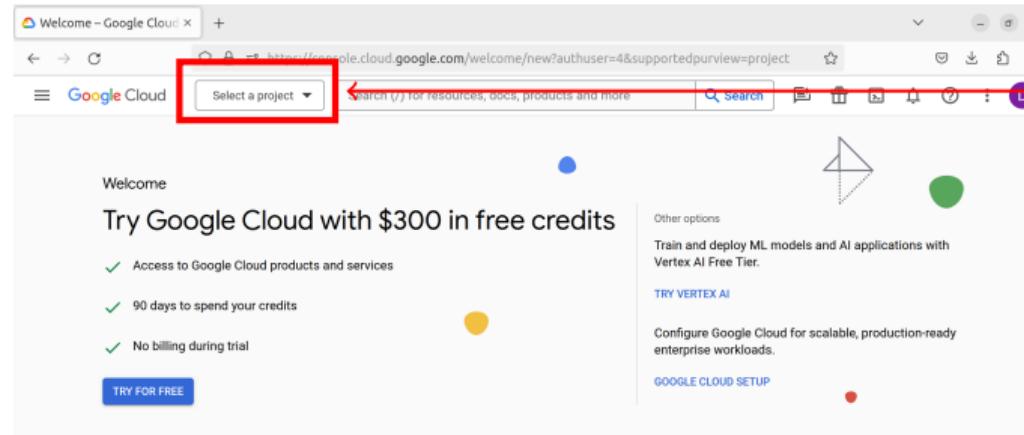
2. Create a Google Cloud project & enable GEE API

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

2.2. Create a Google Cloud project & enable GEE API

2. Create a Google Cloud project & enable GEE API

2.3 Enable GEE API in the newly created project



6. select the project

Welcome

Try Google Cloud with \$300 in free credits

- ✓ Access to Google Cloud products and services
- ✓ 90 days to spend your credits
- ✓ No billing during trial

TRY FOR FREE

Other options

Train and deploy ML models and AI applications with Vertex AI Free Tier.

TRY VERTEX AI

Configure Google Cloud for scalable, production-ready enterprise workloads.

GOOGLE CLOUD SETUP

Popular getting started resources

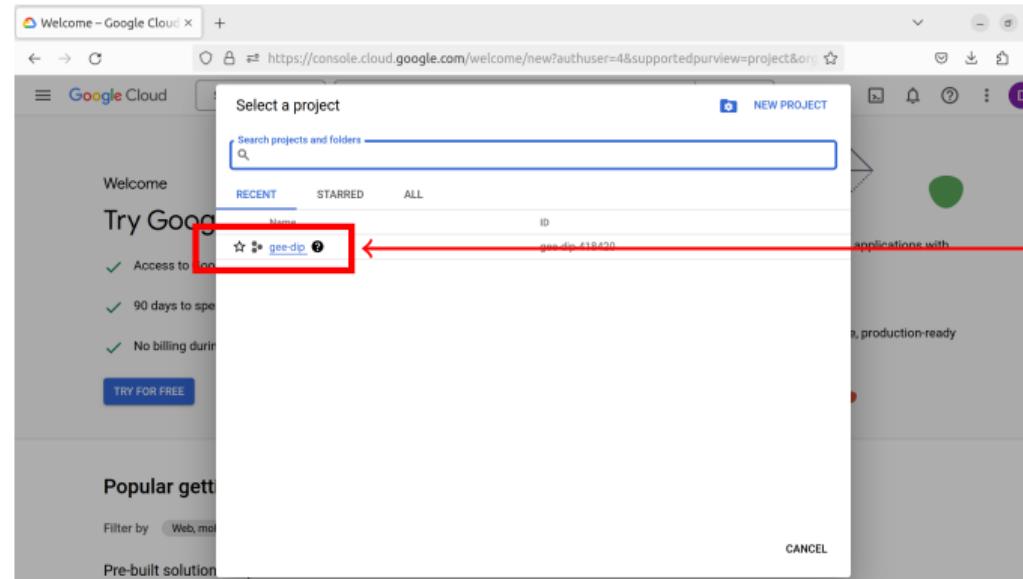
Filter by Web, mobile, game, storage Containers, VMs, hybrid/multi, move workload Data, AI/ML, SAP Maps, APIs General

Pre-built solution templates

2.2. Create a Google Cloud project & enable GEE API

2. Create a Google Cloud project & enable GEE API

2.3 Enable GEE API in the newly created project

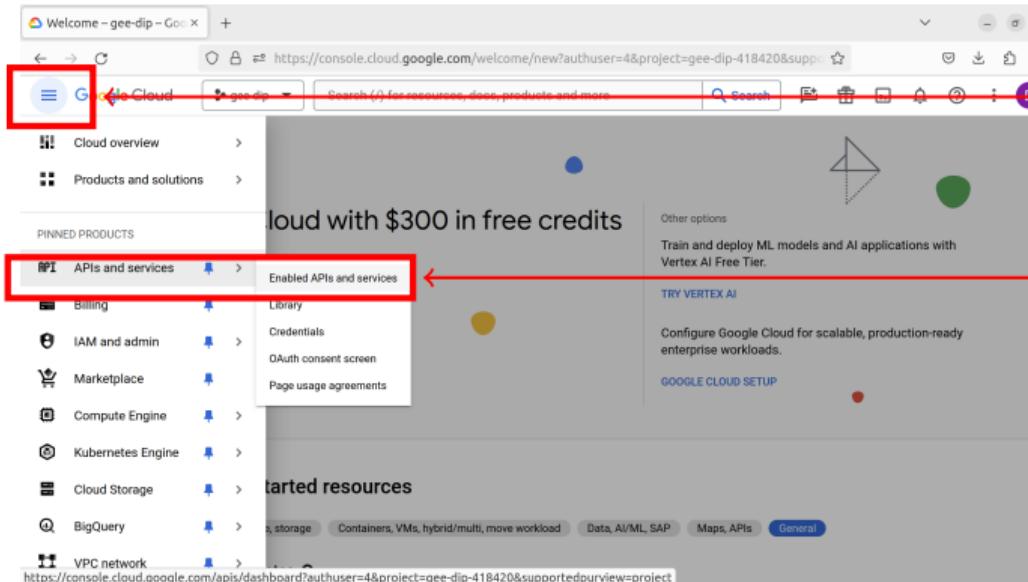


7. select the project

2.2. Create a Google Cloud project & enable GEE API

2. Create a Google Cloud project & enable GEE API

2.3 Enable GEE API in the newly created project



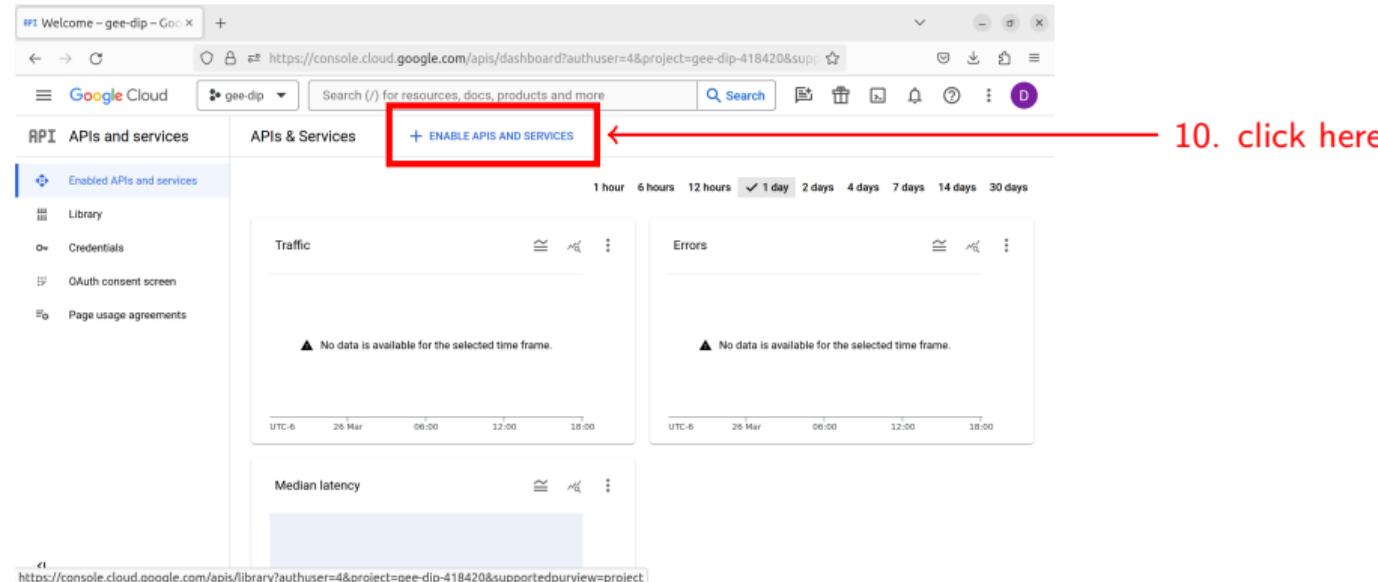
8. select navigation menu

9. "Enable API and services"

2.2. Create a Google Cloud project & enable GEE API

2. Create a Google Cloud project & enable GEE API

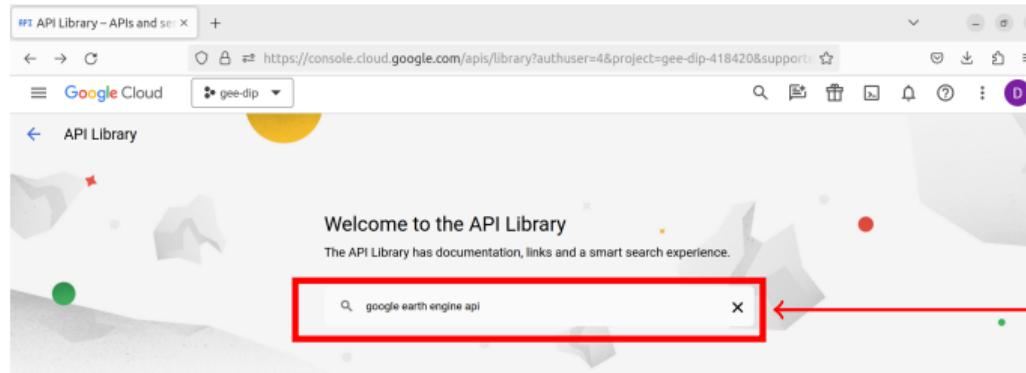
2.3 Enable GEE API in the newly created project



2.2. Create a Google Cloud project & enable GEE API

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”

The screenshot shows the Google Cloud API Library interface. A red box highlights the search bar at the top, which contains the text "google earth engine api". A red arrow points from the right side of the image towards the search bar. Below the search bar, there is a "VIEW ALL (23)" link. On the left, there are filters for "Visibility" (Public: 452, Private: 2) and "Category". The main area displays three items under the "Maps" category: "Maps SDK for Android" (452 results), "Maps SDK for iOS" (2 results), and "Maps JavaScript API".

2.2. Create a Google Cloud project & enable GEE API

2. Create a Google Cloud project & enable GEE API

2.3 Configure the newly created project

The screenshot shows a browser window titled "API APIs and services - gee-dip" with the URL <https://console.cloud.google.com/apis/library/browse?authuser=4&project=gee-dip-418420&supportedpurview=project>. The page displays the "API Library" for the "google earth engine api". A red box highlights the "Google Earth Engine API" entry, which includes a "Google" logo, a note about registration, and a description of the platform. A red arrow points to this highlighted area with the instruction "12. click here".

2.2. Create a Google Cloud project & enable GEE API

2. Create a Google Cloud project & enable GEE API

2.3 Configure the newly created project

The screenshot shows a browser window with the URL <https://console.cloud.google.com/apis/library/earthengine.googleapis.com?authuser=4&project=gee-dip>. The page displays the 'Google Earth Engine API' product details. At the top left is the Google logo and a gear icon. Below it is a brief description: 'Geospatial insights for a more sustainable world.' A prominent blue button labeled 'ENABLE' is highlighted with a red box and a red arrow pointing to it from the right, with the text '13. click "ENABLE"' written in red. Below the button, there are tabs for 'OVERVIEW' (which is underlined), 'PRICING', 'SUPPORT', and 'RELATED PRODUCTS'. The 'OVERVIEW' section contains an 'Important note' about registering the project, followed by a detailed description of Google Earth Engine's capabilities. The 'Additional details' section provides information such as Type: SaaS & APIs, Last product update: 01/08/2022, Category: Big data, Analytics, Science & research, Climate, and Service name: earthengine.googleapis.com.

2.2. Create a Google Cloud project & enable GEE API

2. Create a Google Cloud project & enable GEE API

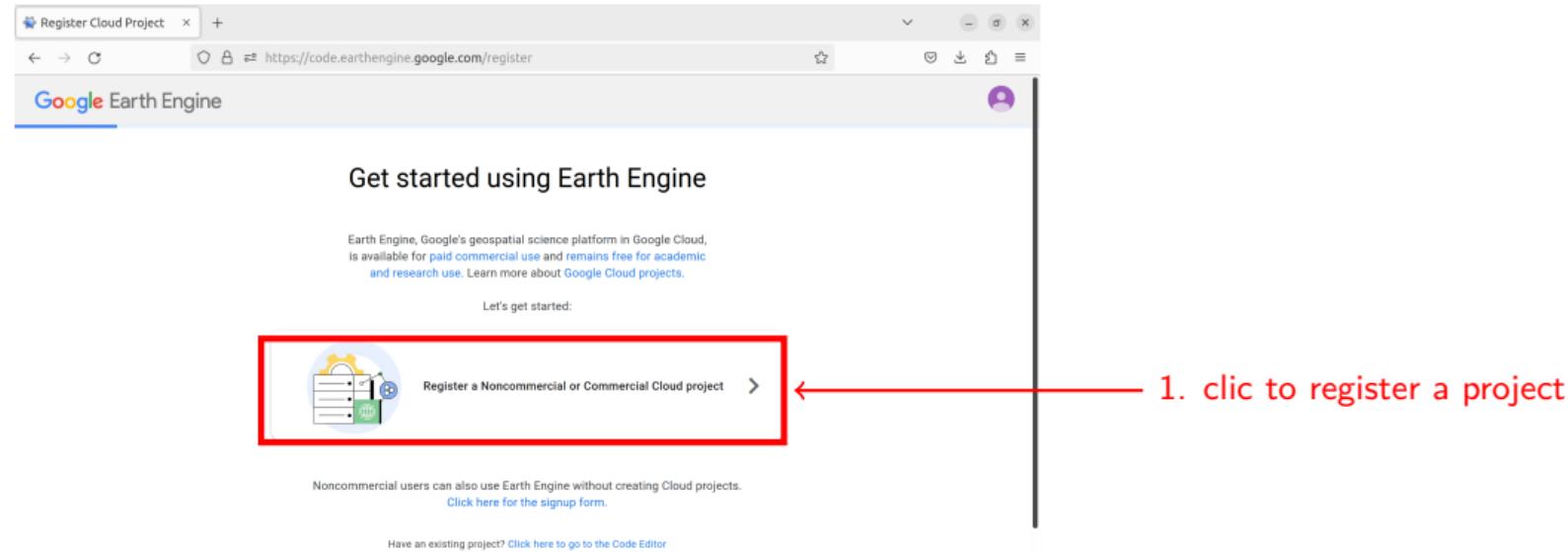
2.3 Enable GEE API in the newly created project

The screenshot shows the 'API/Service details' page for the Google Earth Engine API. The URL in the browser is <https://console.cloud.google.com/apis/api/earthengine.googleapis.com/metrics?project=gee-dip>. The left sidebar shows 'Enabled APIs and services'. The main area displays the 'Google Earth Engine API' with its service name as 'earthengine.googleapis.com' and type as 'Public API'. A red box highlights the 'Status Enabled' button. A red arrow points from the text 'GEE API is now enabled' to this button. Below the status, there's a 'Documentation' link and a 'VIEW DOCUMENTATION' button. At the bottom, there are tabs for 'METRICS', 'QUOTAS AND SYSTEM LIMITS', 'CREDENTIALS', and 'COST'. The 'METRICS' tab is selected, showing a dropdown for 'Select graphs' set to '4 Graphs', and a time range dropdown set to '30 days'. There are also 'Filters', 'Versions', 'Credentials', and 'Methods' dropdowns.

GEE API is now enabled

2.3. Register Google Cloud project for use with GEE

3. Register Google Cloud project project for use with GEE

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

2.3. Register Google Cloud project for use with GEE

3. Register Google Cloud project project for use with GEE

3.1 Register project

The screenshot shows a web browser window titled "Register Cloud Project" at the URL <https://code.earthengine.google.com/register>. The page is titled "How do you want to use Earth Engine?". It offers two options: "Paid usage" and "Unpaid usage". The "Unpaid usage" option is selected, indicated by a blue radio button. Below it, a dropdown menu is set to "Academia & Research". A red box highlights the "Unpaid usage" section, and a red arrow points from the text "2. select:" to this highlighted area.

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

2.3. Register Google Cloud project for use with GEE

3. Register Google Cloud 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

gee-dip gee-dip-418420

gee-dip gee-dip-418420

©2024 Google

3. select project

2.3. Register Google Cloud project for use with GEE

3. Register Google Cloud project for use with GEE

3.1 Register project

The screenshot shows a web browser window titled "Register Cloud Project" at the URL <https://code.earthengine.google.com/register?project=gee-dip-418420>. The page is titled "Confirm your Cloud Project information". It displays a "Project usage" section set to "Academia & Research" and a "Project info" section containing "gee-dip-418420" and "gee-dip". A red box highlights the "Project info" section. At the bottom, there is a "CONFIRM" button in a blue box, with a red arrow pointing to it from the right. A note at the bottom states "Project information cannot be changed later".

Project usage
Academia & Research

Project info
gee-dip-418420
gee-dip

CONFIRM

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

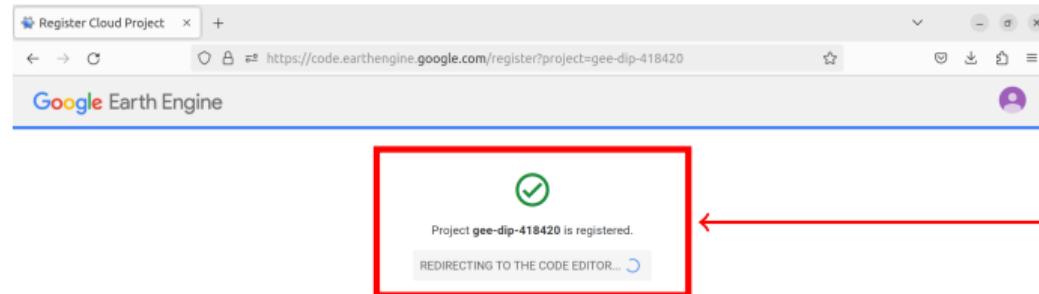
4. Confirm

Project information cannot be changed later

2.3. Register Google Cloud project for use with GEE

3. Register Google Cloud project project for use with GEE

3.2 Register project

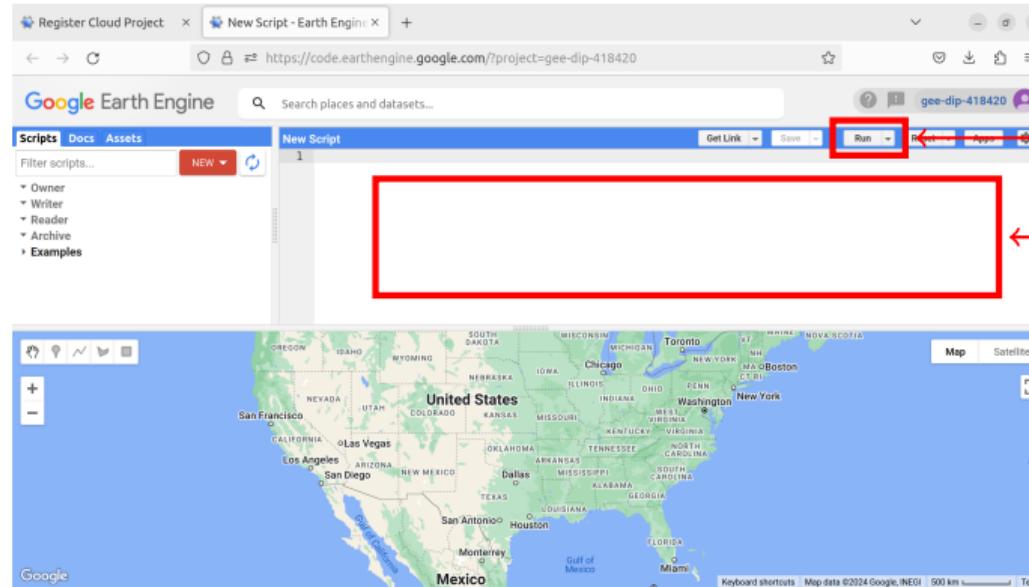


5. register successful
⇒ redirecting to Code Explorer
⇒ JavaScript IDE

2.3. Register Google Cloud project for use with GEE

3. Register Google Cloud project project for use with GEE

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

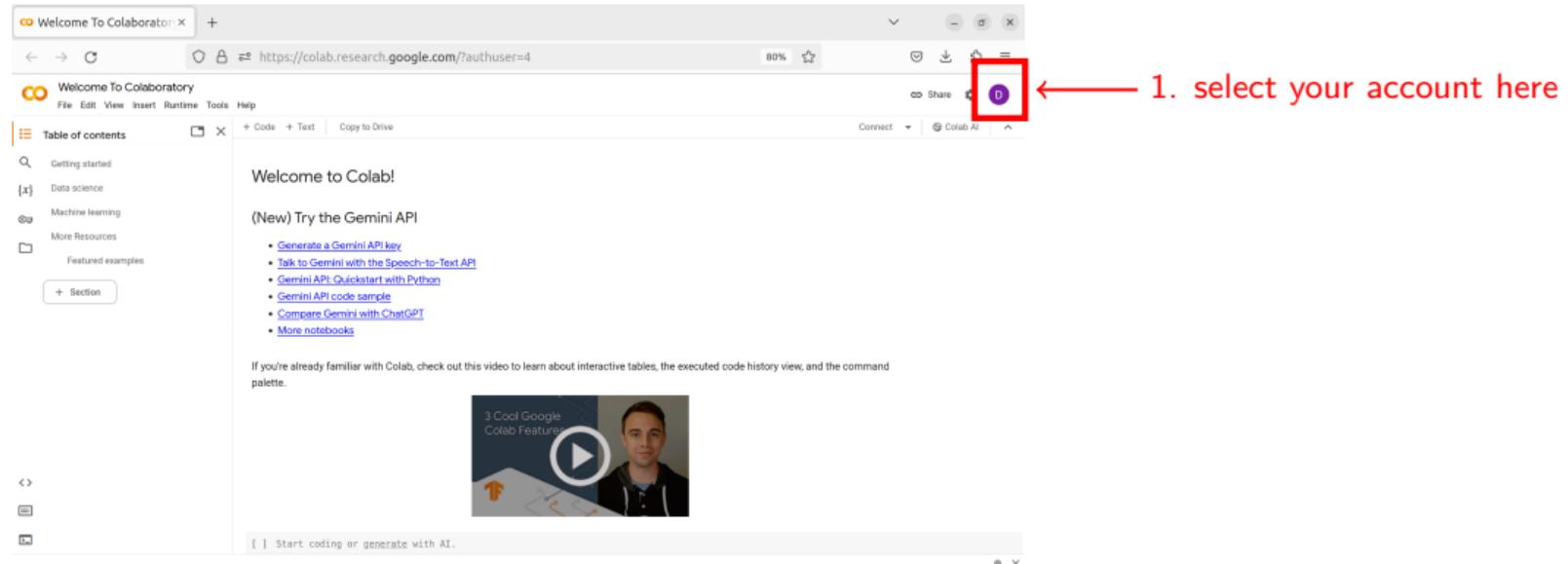


7. clic "Run" to execute code

6. type code in JavaScript
⇒ ex: official tutorials

2.4. Access GEE in Colab

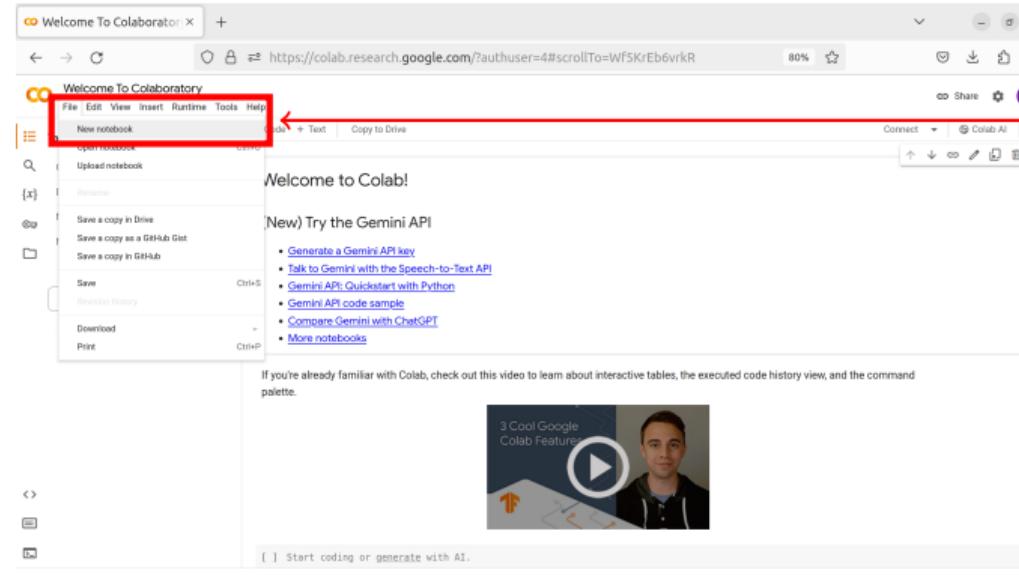
4. Access GEE in Colab

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

2.4. Access GEE in Colab

4. Access GEE in Colab

4.2 Create new notebook



2. File - New notebook

2.4. Access GEE in Colab

4. Access GEE in Colab

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

3. rename notebook (optional)

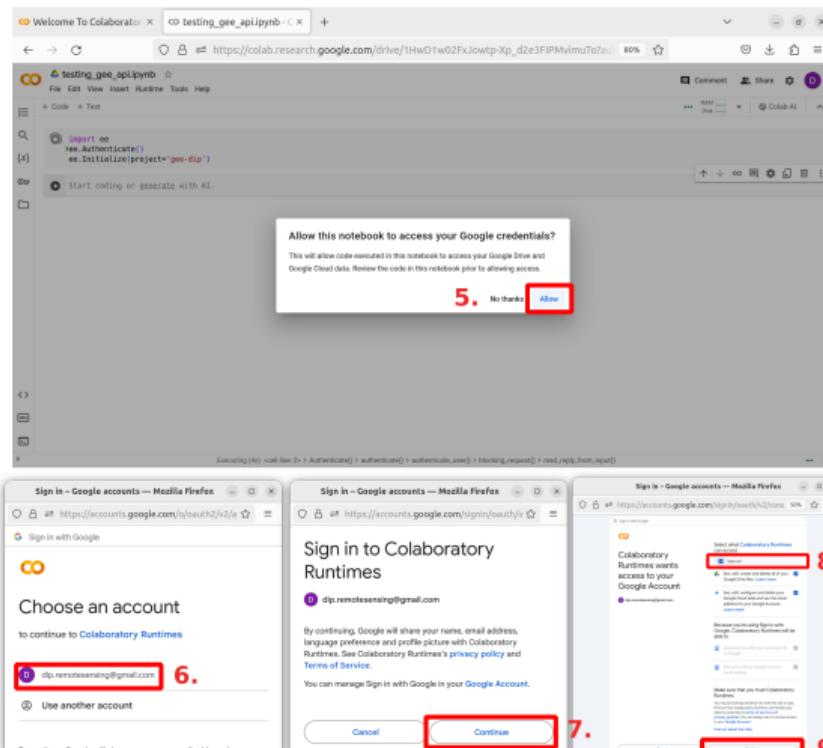
4. import Earth Engine lib (ee),
initialize using **project-id**,
(NOT project-name!),
and execute cell

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

2.4. Access GEE in Colab

4. Access GEE in Colab

4.3 Execute cell & give authorizations in pop-up windows



2.4. Access GEE in Colab

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
```

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

3.1. GEE data catalog

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

1. **Landsat** collections

- ⇒ [NASA/USGS Program](#), since 1972
- ⇒ 9 generations 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)

3.1. GEE data catalog

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) |

3.1. GEE data catalog

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) & Acqua (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.

3.1. GEE data catalog

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) & Acqua (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.

3.1. GEE data catalog

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

3.1. GEE data catalog

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

3.1. GEE data catalog

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 Popocatépetl
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
```

