# Short Introduction to Google Earth Engine

**Rhorom Priyatikanto**
WorldPop - University of Southampton

## Pre-requisites

1. Google account
2. Basic skill in *Python* programming language
3. Basic understanding in GIS concepts, including raster and vector data processing and analysis

## Table of contents



## What is Google Earth Engine

Google Earth Engine, or GEE in short, is a powerful cloud-based geospatial analysis platform developed by Google. It provides researchers, scientists, and developers with access to a vast repository of satellite imagery, geospatial datasets, and computational resources, enabling them to analyze and visualise changes on the Earth's surface at a planetary scale.

Key features and capabilities of GEE includes:

- **Massive Data Catalog**: GEE hosts petabytes of satellite imagery from various sources, including Landsat, Sentinel, MODIS, and more. It also incorporates other geospatial data like climate data, terrain models, and demographic information.
- **Planetary-Scale Analysis**: The platform leverages Google's cloud infrastructure to provide immense computational power, allowing users to process and analyse massive datasets quickly and efficiently.
- **Time Series Analysis**: GEE facilitates the study of changes over time by providing historical satellite imagery and tools for analysing temporal trends and patterns.
- **API and Development Tools**: GEE offers a well-documented API (Application Programming Interface) in JavaScript and Python, enabling users to write custom scripts and algorithms to extract insights from the data.
- **Interactive Interface**: The GEE Code Editor provides a user-friendly web-based interface for writing and executing code, visualising data, and creating interactive maps.
- **Collaboration and Sharing**: Users can share their code, algorithms, and results with others, fostering collaboration and knowledge exchange within the GEE community.

GEE has a wide range of applications across various fields, including:

- **Environmental Monitoring**: Tracking deforestation, land cover change, water resources, and other environmental parameters.
- **Agriculture**: Assessing crop health, monitoring drought conditions, and optimising agricultural practices.
- **Climate Change Research**: Analysing long-term climate trends, assessing the impact of climate change, and developing adaptation strategies.
- **Disaster Management**: Monitoring natural disasters like floods, wildfires, and earthquakes, and aiding in disaster response efforts.
- **Urban Planning**: Analysing urban growth patterns, assessing infrastructure development, and planning for sustainable cities.
- **Public Health**: Tracking disease outbreaks, analysing environmental factors influencing health, and supporting public health interventions.

GEE also offers several benefits:

- **Accessibility**: GEE democratises access to geospatial data and analysis tools, making them available to a wider audience of researchers and decision-makers.
- **Scalability**: The platform's cloud-based architecture enables users to handle large-scale analyses that would be difficult or impossible with traditional desktop tools.
- **Speed**: GEE's powerful computational resources significantly accelerate data processing and analysis, saving valuable time.
- **Collaboration**: The platform fosters a collaborative environment where users can share knowledge, code, and data, accelerating research and innovation.

## Getting started

To start using GEE, follow these steps:

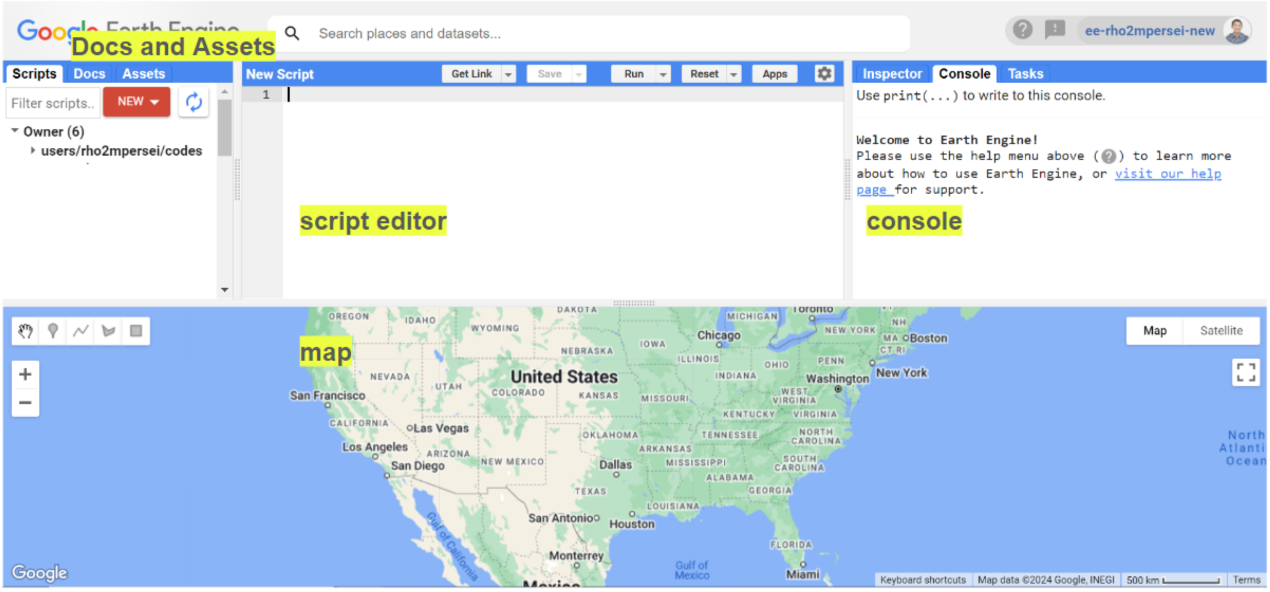
1. Go to the Google Earth Engine website and select "Get Started" menu to register. You'll need a Google account to sign up. If you don't have one, you can create one for free.

2. Fill out the application form, providing details about your intended use of GEE. Submit your application and wait for approval. This process may take a few days.

3. Once your application is approved, you'll receive an email notification. You can then access the GEE Code Editor at: https://code.earthengine.google.com/

4. Familiarize yourself with the Code Editor interface, which includes:

    - Script Editor: Where you'll write your JavaScript or Python code.
    - Console: To view output and error messages.
    - Map: To visualize your data and analysis results.
    - Docs: Access to the GEE documentation and API reference.
    - Assets: To manage your uploaded data and scripts.




5. Begin writing your own Javascript to analyze and visualize data.

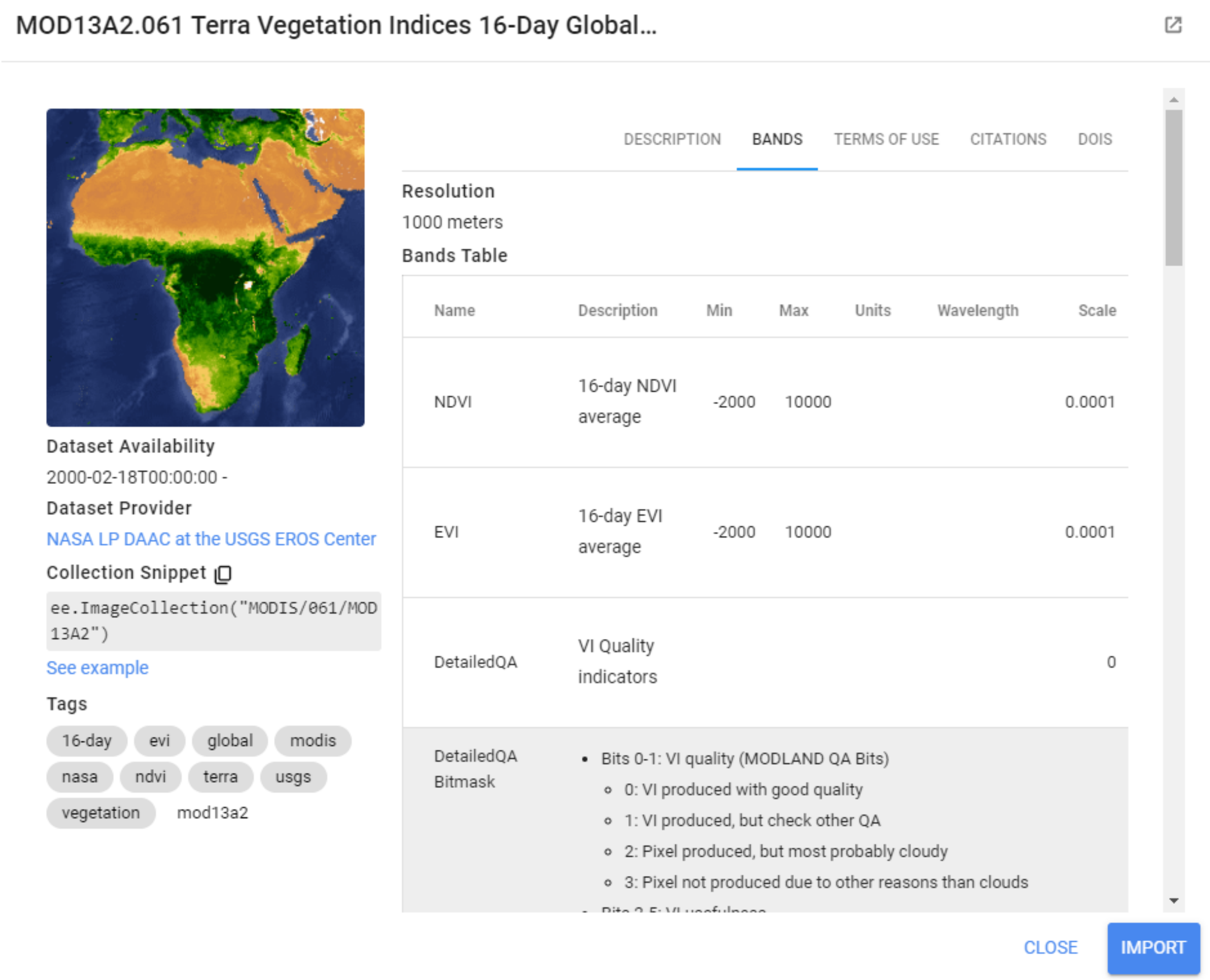
6. Utilize the [Earth Engine API reference]() to discover available functions and classes. The Docs tab on the left panel of the Code Editor can also be helpful.

7. Join the community. The GEE community is a valuable resource for learning and getting help.

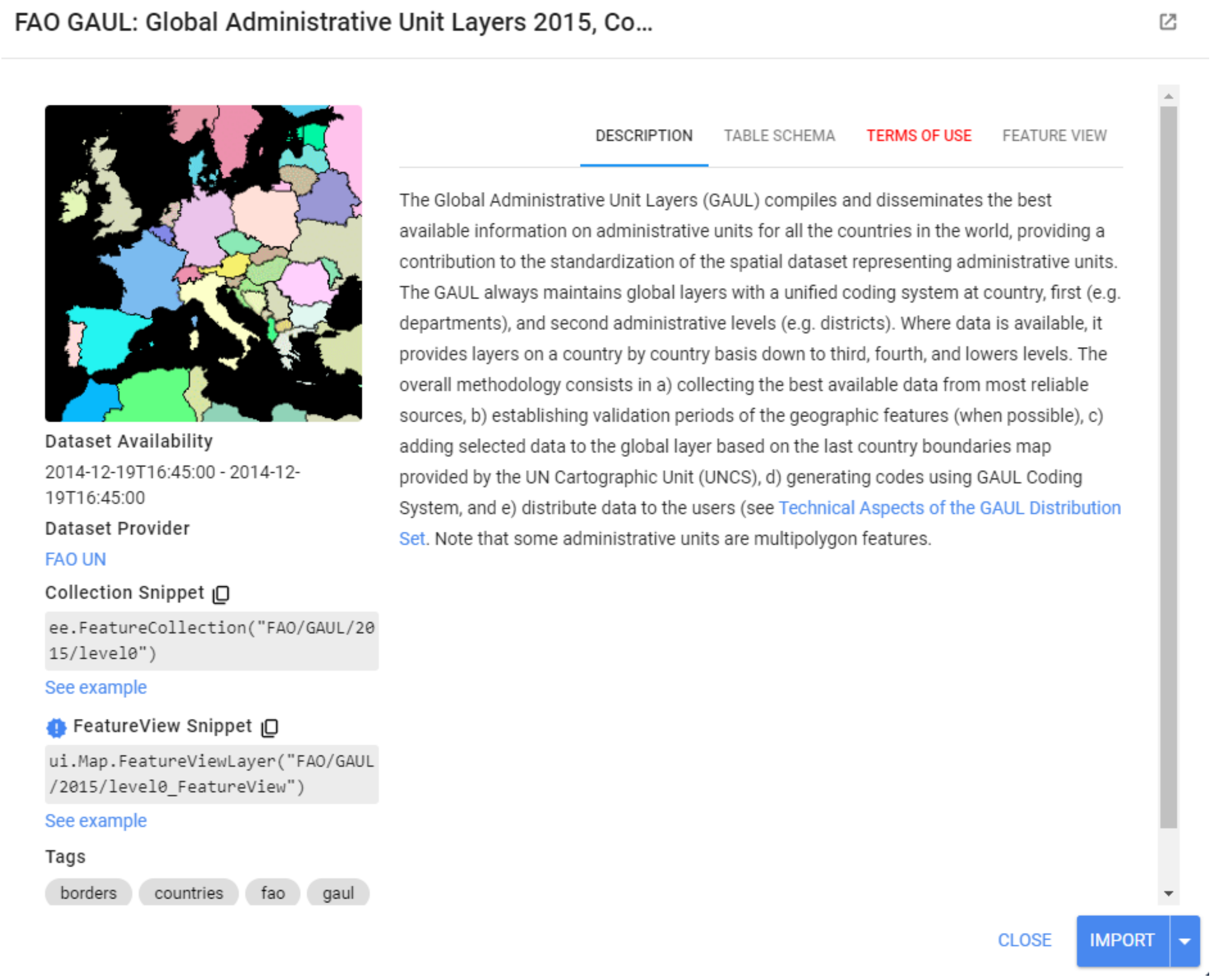
## Entities in GEE

Google GEE primarily stores and provides access to two main types of geospatial data:

- **Raster Data (Images)**:
    - Satellite Imagery: GEE hosts a vast collection of satellite imagery from various sensors and platforms, including Landsat, Sentinel, MODIS, and many others. This imagery covers a wide range of spectral bands and resolutions, enabling diverse applications from land cover classification to change detection. `Image` data type represents a single raster data that contains one or more bands. Each band has its own name, data type, scale, mask, and projection.
    - Image Collections: GEE organises related images, such as a time series of satellite imagery, into Image Collections. An `ImageCollection` may contain multiple `Image` with different spatial extent or sequence of `Image` with distinguished timestamps. We can filter images in the collection using some methods.
    - Other Raster Data: In addition to satellite imagery, GEE also stores other raster datasets like digital elevation models (DEMs), land surface temperature data, and various thematic maps.




- **Vector Data (Features)**:
  - Geometry: A set of coordinates define a `Geometry`. Type of geometries supported by GEE, include points, lines, and polygons. GEE also supports multi-point, multi-linestring, and multi-polygon geometries.
  - Feature: A collection of data attached with geospatial information can be describe as `Feature` data type in GEE. Basically, it is a `Geometry` with data added to it.
  - Feature Collections: GEE allows you to work with vector data in the form of `FeatureCollection`. These collections can represent geometries and each feature can have associated attributes.
  - Shapefiles and GeoJSON: You can import your own vector data in formats like shapefiles or GeoJSON into GEE for analysis and integration with other datasets.
  - Table Data: GEE can also handle tabular data, which can be linked to vector features or used independently for analysis.




Other fundamental data structures in GEE include `Dictionary`, `List`, `Array`, `Date`, `Number` and `String`. These data structures are the same as in Javascript language, but they are handled differently on the server-side.

## Raster analysis

GEE offers a comprehensive set of tools and functions for conducting a wide range of raster analyses. Here's an overview of the main types of raster analyses you can perform in GEE:

- **Image Manipulation and Preprocessing**:
  - Cropping: Extract a specific region of interest from a larger image. `Image.clip(Geometry)`
  - Masking: Hide or exclude specific pixels or areas based on certain criteria (e.g., clouds, water bodies). `Image.mask(Mask)`
  - Resampling: Change the spatial resolution of an image (e.g., from 30m to 10m pixels).
  - Mosaicking: Combine multiple overlapping images into a seamless mosaic. `ImageCollection.mosaic()`
  - Band Math: Perform arithmetic operations on image bands to create new indices or transformations. `(ImageA.add(ImageB)).divide(ImageC)`
  - Image Visualization: Adjust brightness, contrast, and color palettes to enhance image visualization.
- **Spectral Analysis**:
  - Band Ratios: Calculate ratios between different spectral bands to highlight specific features or properties.

  - Spectral Indices: Compute various spectral indices (e.g., NDVI, EVI, SAVI) for vegetation analysis, land cover classification, and other applications.
  - Spectral Unmixing: Decompose mixed pixels into their constituent spectral components.
- **Temporal Analysis**:
  - Change Detection: Identify and quantify changes over time (e.g., deforestation, urban expansion, crop growth) using multi-temporal image analysis techniques.
  - Time Series Analysis: Analyze trends and patterns in image data over time (e.g., vegetation phenology, land surface temperature variations).
  - Image Differencing: Subtract one image from another to highlight differences between two points in time.
  - Image Composites: Create composite images from multiple images over a time period to reduce noise and highlight persistent features.
- **Spatial Analysis**:
  - Neighborhood Operations: Perform calculations on pixels based on their neighboring pixels (e.g., focal statistics, convolution filters). `Image.convolve(Kernel)`
  - Zonal Statistics: Calculate statistics (e.g., mean, sum, standard deviation) of pixel values within specified zones or regions. `Image.reduceRegion(Geometry)`
  - Image Classification: Assign land cover classes or other thematic categories to pixels based on their spectral characteristics.
  - Supervised and Unsupervised Classification: Utilize machine learning algorithms for image classification tasks.
  - Image Segmentation: Partition an image into meaningful segments based on spectral and spatial properties.
- **Terrain Analysis**:
  - Slope and Aspect Calculation: Derive slope and aspect maps from digital elevation models (DEMs).
  - Viewshed Analysis: Determine areas visible from a specific point or location.
  - Watershed Delineation: Identify drainage basins and watersheds based on terrain characteristics.
- **Other Analyses**:
  - Image Regression: Develop predictive models to estimate variables based on image data.
  - Accuracy Assessment: Evaluate the accuracy of image classification or other analysis results.
  - Image Fusion: Combine data from different sensors to enhance spatial and spectral information.

## Vector analysis

GEE provides a robust set of tools and functionalities for vector analysis, enabling you to perform a wide range of spatial operations and analyses on vector data.

- **Basic Vector Operations**:
  - Creating Vector Data: You can create vector features (points, lines, polygons) directly in GEE or import them from various formats like Shapefiles, GeoJSON, KML, or CSV.
  - Filtering: Filter features based on their attributes (e.g., select all points within a certain elevation range). `FeatureCollection.filter()`
  - Mapping: Apply functions to each feature in a collection (e.g., calculate the area of each polygon). `FeatureCollection.map(function())`
  - Reducing: Aggregate features in a collection to a single value (e.g., calculate the total area of all polygons).

- **Spatial Analysis**:
  - Buffering: Create buffer zones around points, lines, or polygons. `Geometry.buffer(size)`
  - Overlay Analysis: Perform spatial overlays to identify intersections, unions, and differences between vector datasets.
  - Distance Calculations: Compute distances between features or from features to other geometries. `GeometryA.distance(GeometryB)`
  - Spatial Joins: Combine attributes from different vector datasets based on their spatial relationships (e.g., find the nearest road to each point).
- **Spatial Queries**:
  - Selecting Features: Select features that meet specific spatial criteria (e.g., find all points within a given polygon). `FeatureCollection.filter()`
  - Clipping: Clip a vector dataset to a specific area of interest.
  - Intersection: Find the overlapping areas between two vector datasets. `FeatureA.intersection(FeatureB)`
  - Union: Combine two vector datasets into a single dataset. `FeatureA.union(FeatureB)`
  - Difference: Find the areas in one vector dataset that are not in another. `FeatureA.difference(FeatureB)`
- **Vector-Raster Interaction**:
  - Zonal Statistics: Calculate statistics (e.g., mean, sum, standard deviation) of raster pixel values within vector zones. `Image.reduceRegions(Feature)`
  - Vector to Raster Conversion: Convert vector data into raster format (e.g., create a land cover map from polygon features). `Feature.reduceToImage()`
  - Raster to Vector Conversion: Convert raster data into vector format (e.g., extract polygons from classified images). `Image.reduceToVectors()`
  - Sampling: Extract raster pixel values at specific vector point locations. `Image.sample()`
- **Advanced Vector Operations**:
  - Geometry Transformations: Simplify geometries, calculate centroids, and perform other geometric operations.
  - Topological Operations: Identify shared boundaries, adjacent polygons, and other topological relationships.
  - Vector-Based Machine Learning: Apply machine learning algorithms to vector data for classification, regression, or clustering tasks.

## Data visualisation

GEE allows users to create informative and visually appealing maps, charts, and other graphics. Here's an overview of the types of data visualization you can do in GEE:

- **Image Visualization**:
  - RGB Composites: Combine multiple spectral bands (e.g., red, green, blue) to create natural color or false-color composite images.
  - Color Palettes: Apply different color palettes to highlight specific features or ranges of values in raster data.
  - Histograms: Visualize the distribution of pixel values in an image.
  - Scatter Plots: Explore relationships between different bands or indices.
  - Time Series Animation: Create animations to visualize changes in image data over time.

- **Vector Visualization**:
    - Point, Line, and Polygon Rendering: Display vector data as points, lines, or polygons with customizable colors, sizes, and styles.
    - Add labels to vector features for easy identification.
    - Display density or intensity of point data as a heatmap.

- **Interactive Mapping**:
    - Layer Control: Add and remove layers to the map interactively.
    - Zoom and Pan: Explore data at different scales and locations.
    - Pop-up Information: Display attribute information or charts when clicking on features.
    - Time Slider: Visualize changes in data over time using a time slider.

- **Charting**:
    - Line Charts: Plot time series data or changes in values over a spatial dimension.
    - Bar Charts: Compare values across categories or regions.
    - Scatter Plots: Visualize relationships between two variables.
    - Histograms: Show the distribution of values in a dataset.
    - Pie Charts: Display proportions of different categories.

- **Exporting Visualizations**:
    - Images: Export maps and charts as image files (e.g., PNG, JPEG).
    - Videos: Export time series animations as video files (e.g., MP4).

# Common workflow

1. Data Acquisition and Preprocessing:
    - Image/ImageCollection Selection: Identify the appropriate raster datasets (satellite imagery, elevation models, land cover data, etc.) from the Earth Engine Data Catalog. Uploading data into assets can also be another alternative to get the data ready for further processes.
    - Filtering: Filter images based on specific criteria like date range, cloud cover, or sensor type.
    - Region of Interest (ROI) Definition: Define the geographic area of interest using geometries (points, lines, polygons).
    - Image Preprocessing: Perform operations like masking clouds, atmospheric correction, or topographic normalization to prepare the data for analysis.
    - Band Math/Indices Calculation: Compute new bands or indices using mathematical expressions on existing bands (e.g., NDVI, EVI).
    - Image Mosaicking/Compositing: Combine multiple overlapping images to create a seamless image or a composite representing the best pixels from a collection.
2. Processing and Analysis: This step is very dependent on the objectives of the job.
3. Post-processing and Visualization:
    - Map Visualization: Display results on an interactive map using different color palettes, styles, or layer combinations.
    - Charting: Create charts or graphs to visualize time series data, histograms, or other statistical summaries.

  - Export: Export results as GeoTIFFs, tables, or other formats for further analysis or sharing.