

# *Assignment 2*

# Problem

## Bulk Download - 3 Items

The [Bulk Download Web Application](#) is an easy-to-use tool for downloading large quantities of satellite imagery and geospatial data. This application allows users to download groups of submitted scenes that can be automatically executed without the user physically downloading each scene. This web interface replaced the legacy Java version of the application.

☒ Landsat 7 ETM+ C2 L2 (3 pending selection)

Bulk Download

Order Name - Optional

Submitting...

Unable to get order number at this time. The order confirmation email will be sent to the e-mail address associated with this ERS account once the order is submitted. Please check your email later. Thank you for your patience.

**Faced a problem while downloading data in bulk.**

**Explored other open source websites for the time being to get started with the task**

```
var collection = ee.ImageCollection('COPERNICUS/S5P/OFFL/L3_CH4')
    .select('CH4_column_volume_mixing_ratio_dry_air')
    .filterDate('2019-06-01', '2019-07-16');

var band_viz = {
  min: 1750,
  max: 1900,
  palette: ['black', 'blue', 'purple', 'cyan', 'green', 'yellow', 'red']
};

Map.addLayer(collection.mean(), band_viz, 'S5P CH4');
Map.setCenter(0.0, 0.0, 2);
```

**Found this on Earth engine data catalog**

**Idea:**

- **Can do web scraping to get list of datasets.**
- **Then use web scraping again for extracting the API to access a specific dataset from this site.**

## function for downloading data

### API to access data

```
# Earth Engine collection and parameters
collection = ee.ImageCollection('COPERNICUS/S5P/NRTI/L3_CLOUD') \
    .select('cloud_fraction') \
    .filterDate('2019-7-13', '2023-07-13')
```

```
band_viz = {
    'min': 0,
    'max': 0.95,
    'palette': ['red', 'green', 'blue']
}
```

### Visualization parameter

```
# Download the image as a GeoTIFF
geotiff_path = os.path.join(output_dir, 'cloud_fraction.tif')
task = ee.batch.Export.image.toDrive(
    image=mean_image,
    description='CloudFraction',
    folder=output_dir,
    fileNamePrefix='cloud_fraction',
    scale=1000,
    region=mean_image.geometry().bounds().getInfo()['coordinates'],
    crs=mean_image.projection().crs().getInfo() # Specify the desired CRS
)
task.start()
```

### To download mean\_image

```
# Convert the downloaded file to a GeoTIFF
tif_path = os.path.join(output_dir, 'cloud_fraction.tif')
gdal.Translate(tif_path, geotiff_path)
```

### Convert to GeoTIFF