เสี่ยอามสั่งลุย

PROJECT

PRESENTATION

6431325721 Nunthaphop Jamsupthavorn

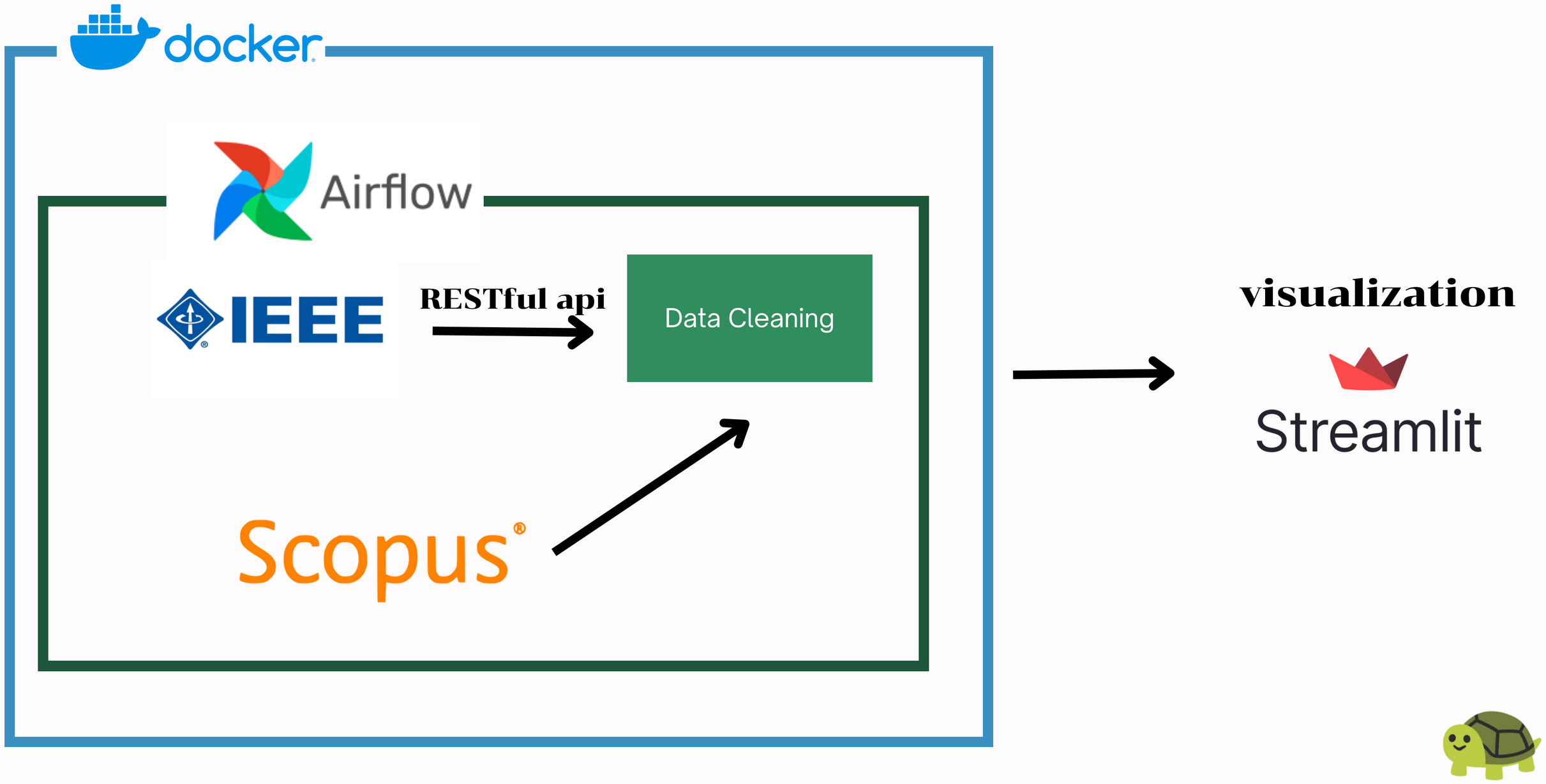
6431338921 Rutthee Raywatkhunanon

6432072021 Thammasorn Thammasarangkoon

6531325821 Pawarit Klinsiengdee

OVERVIEW

Unlocking Insights

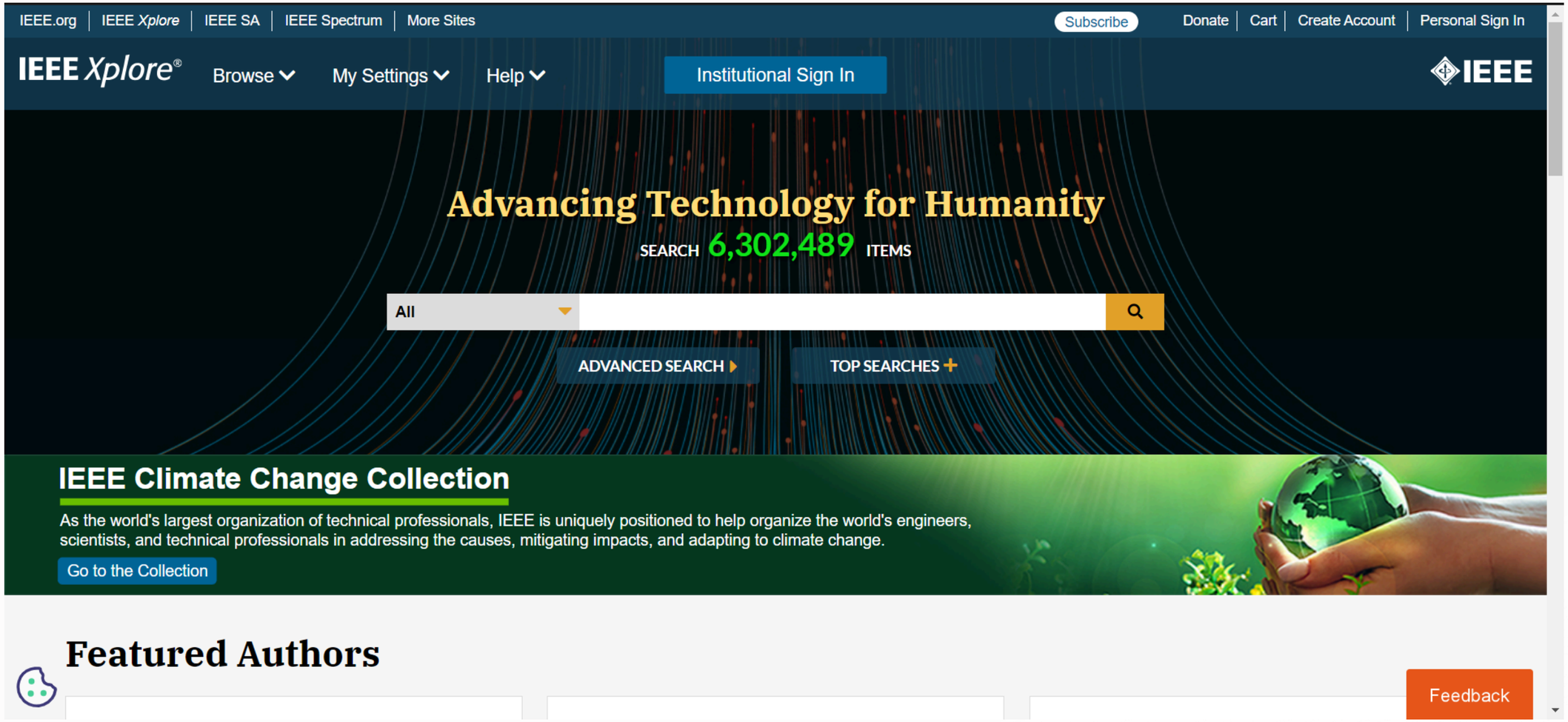

docker
Airflow
IEEE
RESTful api
Data Cleaning
Scopus®
visualization
Streamlit

DATA EXTRACTION

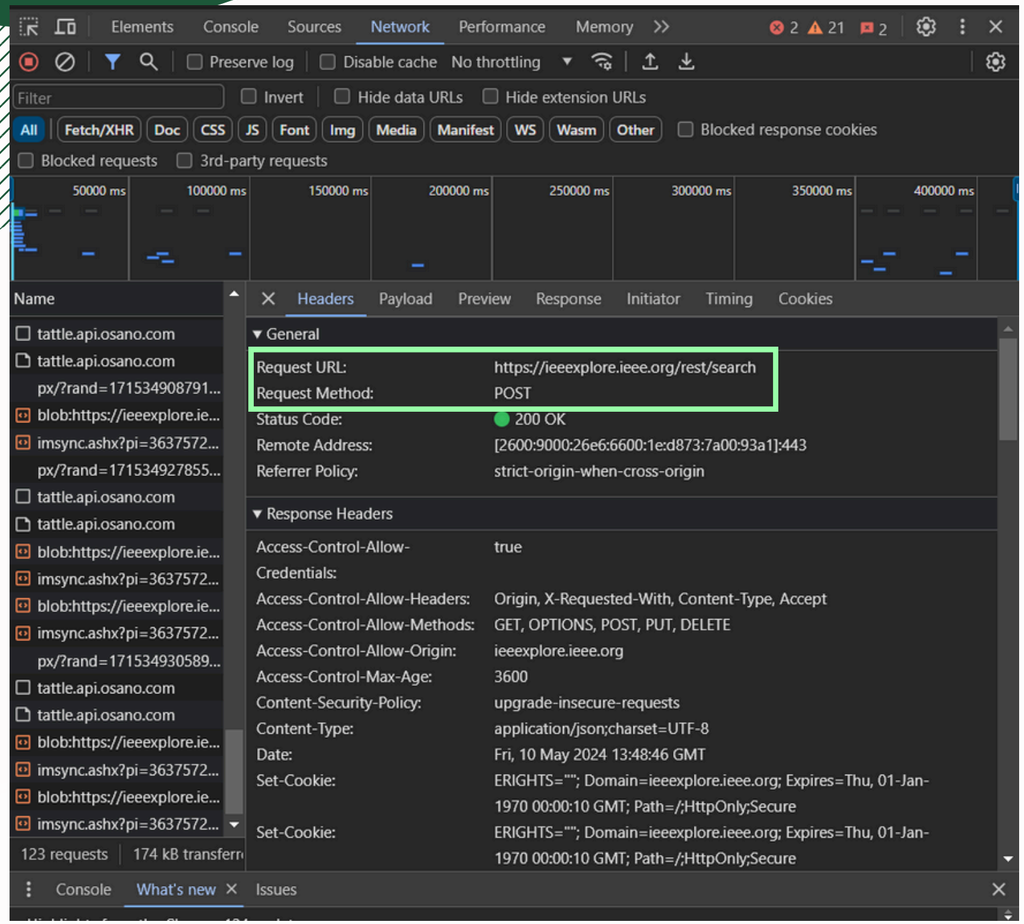
REST API Scraping

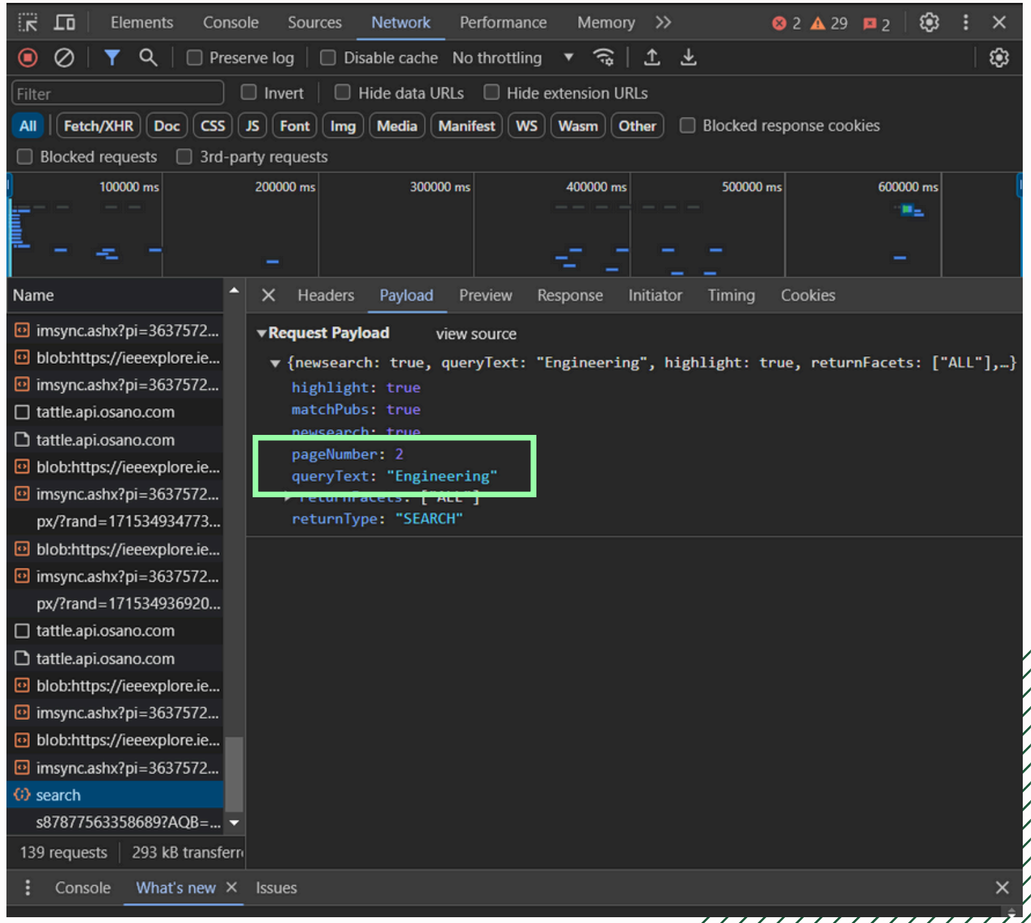
EXPLORING THE SITE

https://ieeexplore.ieee.org/Xplore/home.jsp



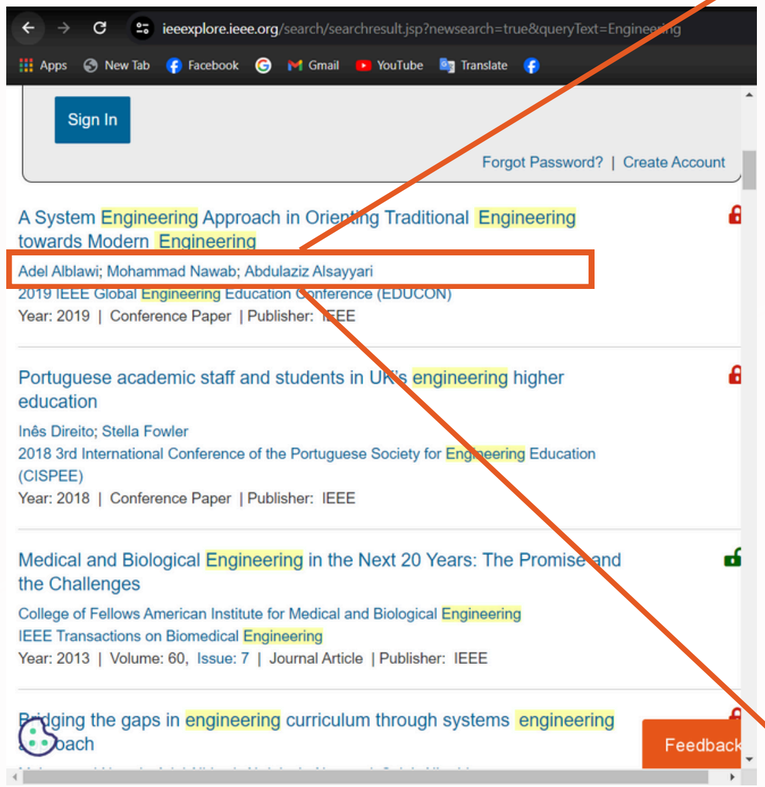
Lists Endpoint & QueryParams

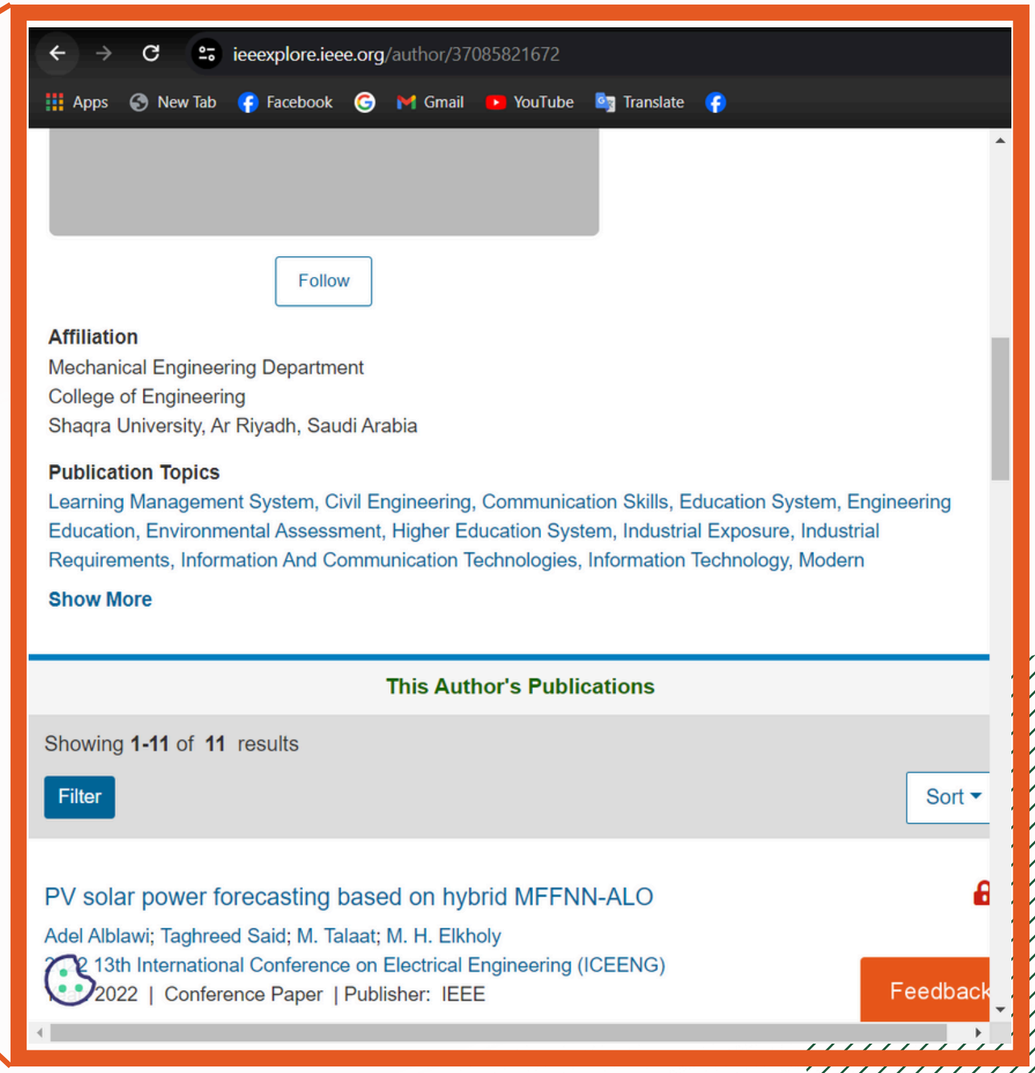




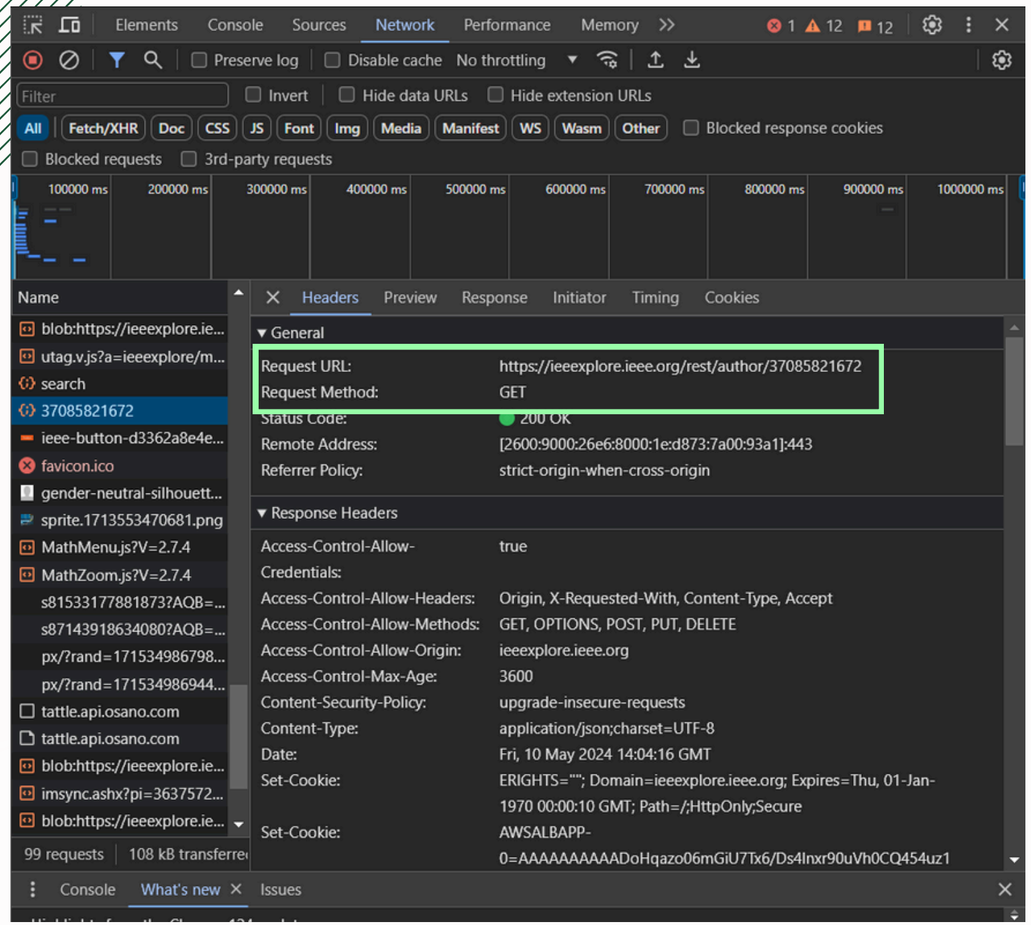
```python
def scrape_ieee(query, num_pages):
    url = "https://ieeexplore.ieee.org/rest/search"
    headers = {
        "accept": "application/json, text/plain, */*",
        "accept-language": "th-TH,th;q=0.9",
        "content-type": "application/json",
        "origin": "https://ieeexplore.ieee.org",
        "priority": "u=1, i",
        "referer": f"https://ieeexplore.ieee.org/search/searchresult.jsp?newsearch=true&queryText={query}",
        "sec-ch-ua": "\"Chromium\";v=\"124\", \"Google Chrome\";v=\"124\", \"Not-A.Brand\";v=\"99\"",
        "sec-ch-ua-mobile": "?0",
        "sec-ch-ua-platform": "\"Windows\"",
        "sec-fetch-dest": "empty",
        "sec-fetch-mode": "cors",
        "sec-fetch-site": "same-origin",
        "user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/124.0.0.
        "x-security-request": "required"
    }
    for page in range(1, num_pages+1):
        data = {
            "newsearch": True,
            "queryText": query,
            "highlight": True,
            "returnFacets": ["ALL"],
            "returnType": "SEARCH",
            "matchPubs": True,
            "pageNumber": page
        }

        response = requests.post(url, headers=headers, json=data)
        response.raise_for_status()  # Raise an exception for unsuccessful requests

        data = response.json()
        records = data.get('records', [])
        all_records.extend(records)

    # Create DataFrame from all records
    df = pd.json_normalize(all_records)
    return df
```

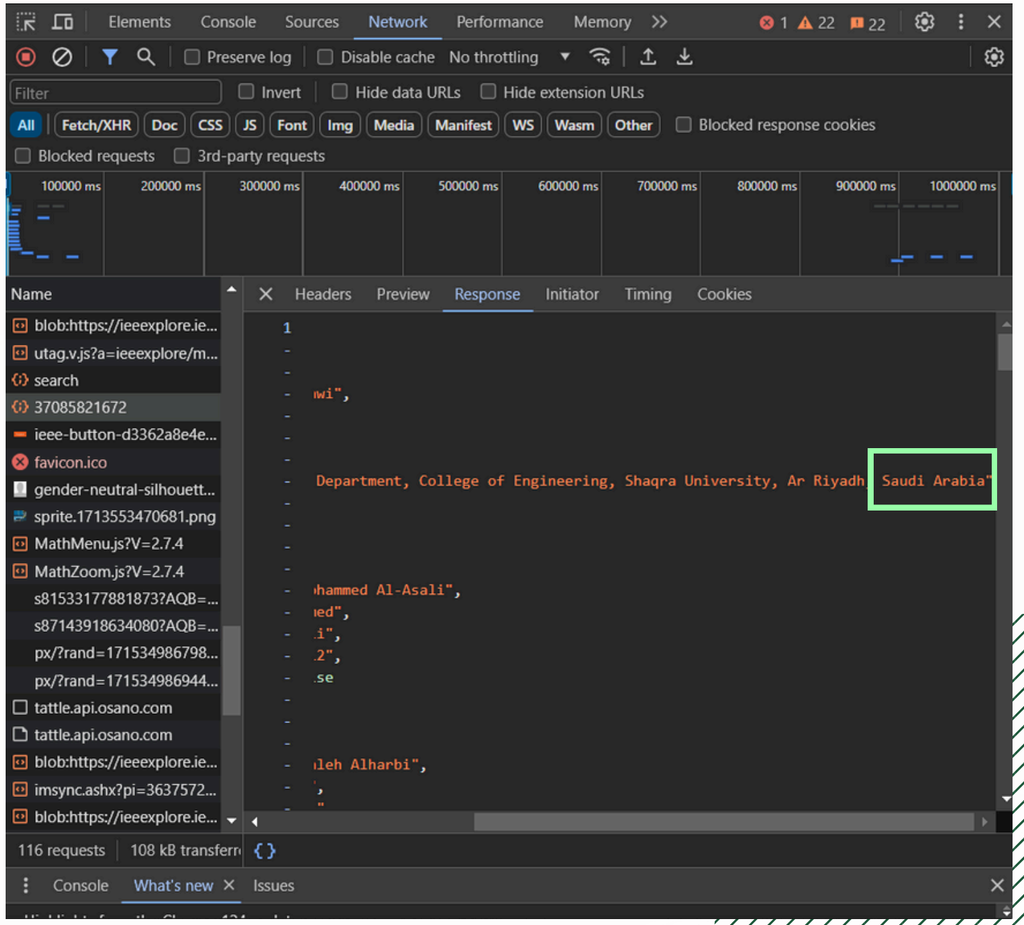
Useful Insights of Finding Author's Detail





Author Details Endpoint & QueryParams





```python
def scrape_each_author(author):
    url = f"https://ieeexplore.ieee.org/rest/author/{authorId}"
    headers = {
        "accept": "application/json, text/plain, */*",
        "accept-language": "th-TH,th;q=0.9",
        "content-type": "application/json",
        "origin": "https://ieeexplore.ieee.org",
        "priority": "u=1, i",
        "referer": "https://ieeexplore.ieee.org/search/searchresult.jsp?newsearch=true&queryText=Engineering",
        "sec-ch-ua": "\"Chromium\";v=\"124\", \"Google Chrome\";v=\"124\", \"Not-A.Brand\";v=\"99\"",
        "sec-ch-ua-mobile": "?0",
        "sec-ch-ua-platform": "\"Windows\"",
        "sec-fetch-dest": "empty",
        "sec-fetch-mode": "cors",
        "sec-fetch-site": "same-origin",
        "user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/"
    }

    response = requests.get(url, headers=headers)
    if response.status_code != 200:
        memoization[authorId] = None
        return None
    response.raise_for_status()  # Raise an exception for unsuccessful requests

    data = response.json()
    country = get_country(data, authorId)
    return country
```

SCRAPING RESULTS

	A	B	C	D	E	F	G	H	I	J	K	L
1		authors	publicationNur	publicationDate	articleNumbe	articleTitle	downloa	abstract	articleConten	authorsAffilationCountry	extracted_class	
2	0	[{'preferredName': 'Md. Shake Farid Ud	10440723	13-15 Dec. 2023	10441199	Analysis of SARGable	14	Being lived in the era of the 4th In	Conferences	['BGD', 'BGD', 'BGD']	MEDI	
3	1	[{'preferredName': 'Khin Myat Kyu', 'nor	9261768	4-5 Nov. 2020	9261805	Enhancement of Que	143	During couples of decades, the p	Conferences	['MMR', 'MMR']	MEDI	
4	2	[{'preferredName': 'Madhuri A. Potey', '	6495610	22-23 Feb. 2013	6514421	A survey of query log	503	[::Query::] log is the pouch of valu	Conferences	['IOT', 'IOT', 'IOT']	MEDI	
5	3	[{'preferredName': 'Xiaoyi Zhou', 'norma	8272882	9-10 Dec. 2017	8283258	Multi-query Optimiza	144	As the main [::query::] language	Conferences	['CHN', 'CHN', 'CHN']	MEDI	
6	4	[{'preferredName': 'Jessie Ooi', 'normal	7322100	19-21 Aug. 2015	7333094	A survey of query exp	777	The ineffectiveness of informatic	Conferences	['MYS', 'MYS', 'MMR', 'MYS']	MEDI	
7	5	[{'preferredName': 'Ziyang Liao', 'norma	8904043	14-17 Oct. 2019	8909632	Disjunctive Sets of Pl	55	This paper proposes a method of	Conferences	['JPN', 'JPN']	MEDI	
8	6	[{'preferredName': 'Chandan Sharma',	9285973	21-25 Sept. 2020	9286038	FLUX: From SQL to G	304	With the influx of Web 3.0 the foc	Conferences	['NZL']	MEDI	
9	7	[{'preferredName': 'Xingyu Peng', 'norm	9644301	11-13 June 2021	9644500	Distributed dynamic	142	With the massive increase of gra	Conferences	['CHN', 'CHN']	MEDI	
10	8	[{'preferredName': 'Bolong Zheng', 'nor	8725877	8-11 April 2019	8731602	Answering Why-Not (	227	With the proliferation of geo-text	Conferences	['CHN', 'CHN', 'DNK', 'AUS', '	MEDI	
11	9	[{'preferredName': 'Diya Thomas', 'norn	6305052	9-11 Aug. 2012	6305549	Location Dependent	358	With advancement in Location B	Conferences	[None]	MEDI	
12	10	[{'preferredName': 'James Callan', 'nor	9473905	30-30 May 2021	9474401	Optimising SQL Quer	184	Structured [::Query::] Language	Conferences	['UKR', 'GBR']	MEDI	
13	11	[{'preferredName': 'S.B. Misal', 'normal	8440575	8-9 Sept. 2017	8455009	DBQA: Multi-Environ	106	In today's computational w	Conferences	['IOT', 'IND', 'IOT']	MEDI	
14	12	[{'preferredName': 'Yuanyuan Xu', 'norn	8766218	19-21 Dec. 2017	8789129	Dynamic Optimizatio	152	When using Keyword relational d	Conferences	['CHN']	MEDI	
15	13	[{'preferredName': 'Qingfeng Zhang', 'n	6921524	4-7 Aug. 2014	6923120	QScheduler: A Tool fo	263	Parallel [::query::] processing in	Conferences	['CHN', 'CHN', 'CHN']	MEDI	
16	14	[{'preferredName': 'L. Saranya', 'norma	6915233	3-5 Jan. 2014	6921736	Optimal top-K querie	107	An effective [::query::] processin	Conferences	['IOT']	MEDI	
17	15	[{'preferredName': 'Zhe Fan', 'normaliz	4629386	Oct.-Dec. 2014	6583915	Towards Efficient Aut	539	Graphs are powerful tools suitab	Journals	['HKG', 'HKG', 'HKG', 'CHN',	MEDI	
18	16	[{'preferredName': 'Mateusz Dziedzic', '	9171991	19-24 July 2020	9177660	Bipolar Queries and F	39	Two similar approaches to the m	Conferences	['BEL', 'BEL', 'POL', 'POL']	MEDI	
19	17	[{'preferredName': 'Samini Subramania	6287639	2021	9351924	Improved Centralized	709	eXtensible Markup Language (XM	Journals	['MYS', 'MYS', 'MYS']	MEDI	
20	18	[{'preferredName': 'Shikha Mehta', 'nor	8509799	2-4 Aug. 2018	8530467	Empirical Evidence o	300	A SQL [::query::] may be express	Conferences	['IOT', 'IOT', 'IOT', None]	MEDI	
21	19	[{'preferredName': 'Manoj Muniswamai	9377717	10-13 Dec. 2020	9378310	Approximate Query P	529	Big Data analytics is used in deci	Conferences	[None, None, None]	MEDI	
22	20	[{'preferredName': 'Amjad Qtaish', 'nor	7337353	10-11 Aug. 2015	7352557	Query mapping techn	107	Extensible Markup Language (XM	Conferences	['SAU', 'MYS']	MEDI	
23	21	[{'preferredName': 'Eman El-Dawy', 'no	6044612	6-6 Sept. 2011	6053927	Multi-level continuou	188	Most of the current work on skyli	Conferences	['EGY', 'EGY', 'EGY']	MEDI	
24	22	[{'preferredName': 'Weining Zhang Zha	8605299	17-20 Dec. 2018	8605751	Using Containers to E	185	Emergent software container tec	Conferences	['USA', None]	MEDI	
25	23	[{'preferredName': 'Abhilasha Kate', 'nc	8466240	29-31 March 2018	8474639	Conversion of Natura	1855	This paper present an approach	Conferences	['IOT', None, 'IOT', 'IOT']	MEDI	
26	24	[{'preferredName': 'Chantal Montgome	9245604	25-26 Sept. 2020	9245462	Towards a Natural La	204	Tackling the information retrieva	Conferences	['CAN', 'CAN', 'CAN']	MEDI	
27	25	[{'preferredName': 'Yunjun Gao', 'norma	69	1-May-15	6940263	Efficient Reverse Top	1042	Reverse k nearest neighbor (RkN	Journals	['CHN', 'CHN', 'SGP', 'CHN']	MEDI	
28	26	[{'preferredName': 'Wang Hairong', 'no	8318023	13-16 Dec. 2017	8323006	The research and app	52	To solve the problem of empty or	Conferences	['CHN', 'CHN']	MEDI	
29	27	[{'preferredName': 'George Obaido', 'nc	8693291	6-8 March 2019	8703620	Generating Narration	201	In the software industry, Structur	Conferences	['USA', 'ZAF', 'ZAF']	MEDI	

What we want to scrape

MEDICAL
ectracted_class: MEDI

ENGINEERING
ectracted_class: ENGI

BIOCHEMICAL
ectracted_class: BIOC

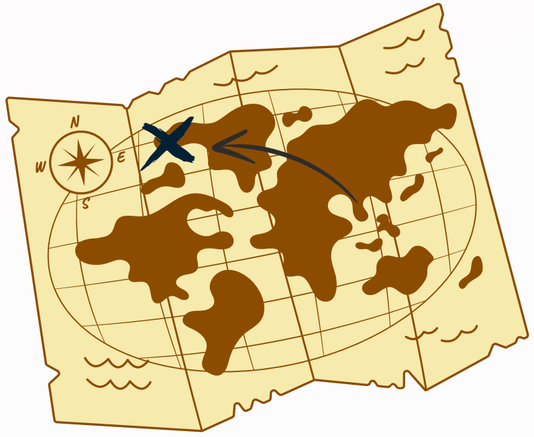
VISUALIZATION

Streamlit library

Visualization Objectives

1

Top Classification on Scopus Data



2

Paper’s Collaboration on Scopus Data (Geospatial Analysis)

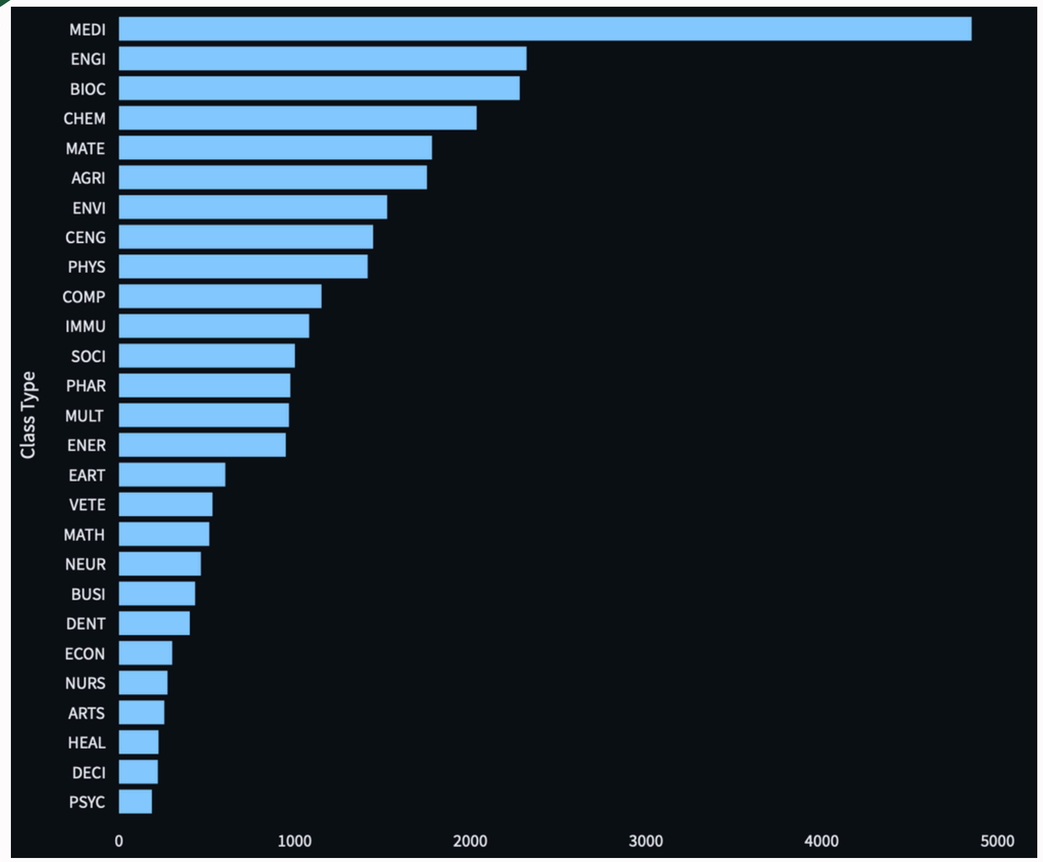
3

Foreign Paper on Top 3 Class (Geospatial Analysis)

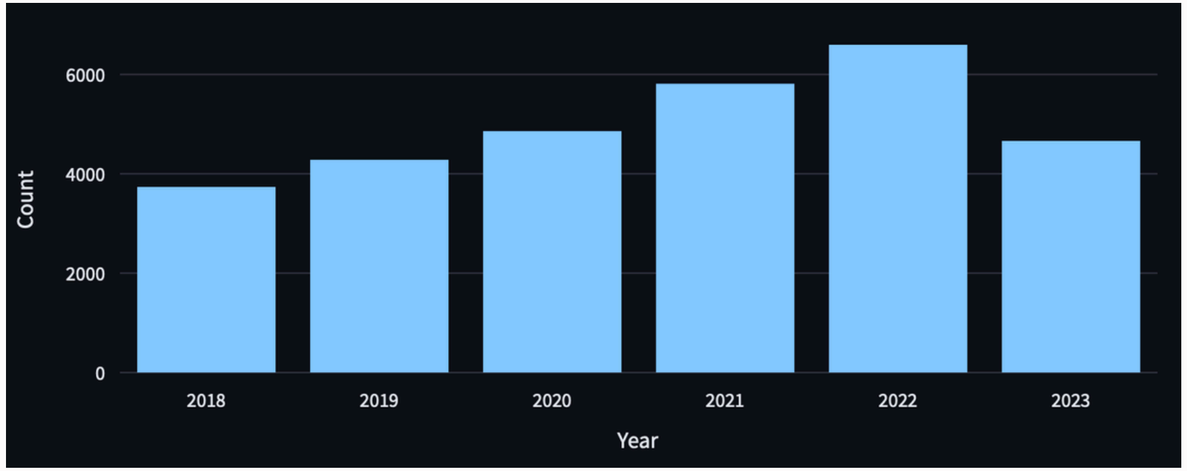
PART I:

CLASSIFICATION TYPE ON SCOPUS DATA

Chulalongkorn’s Academic Paper Research Statistic during 2018 - 2023

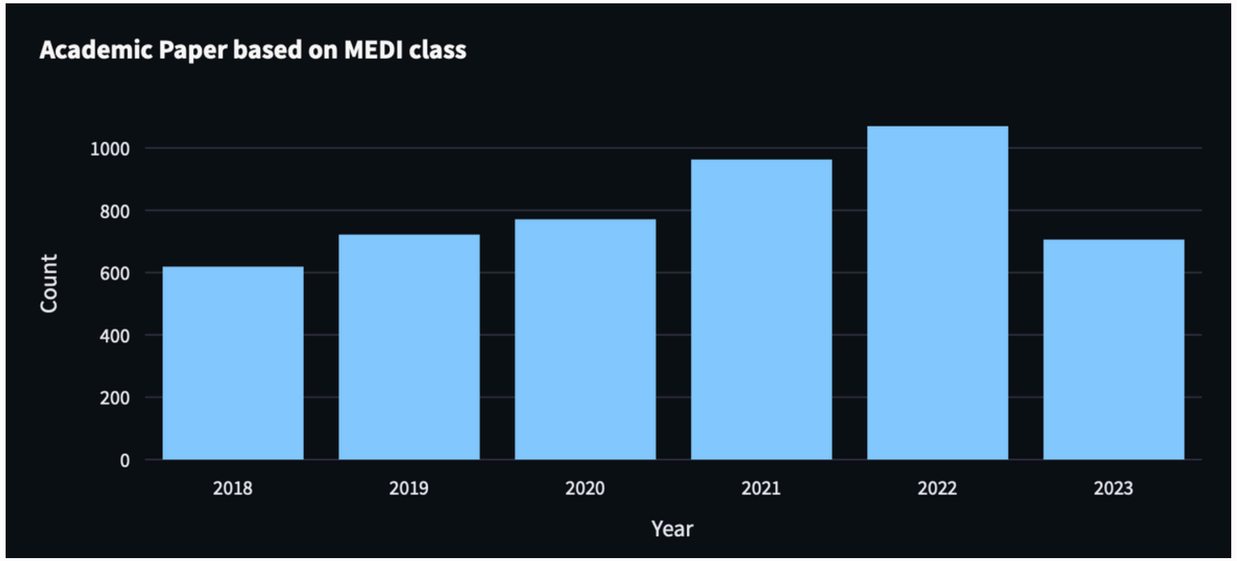


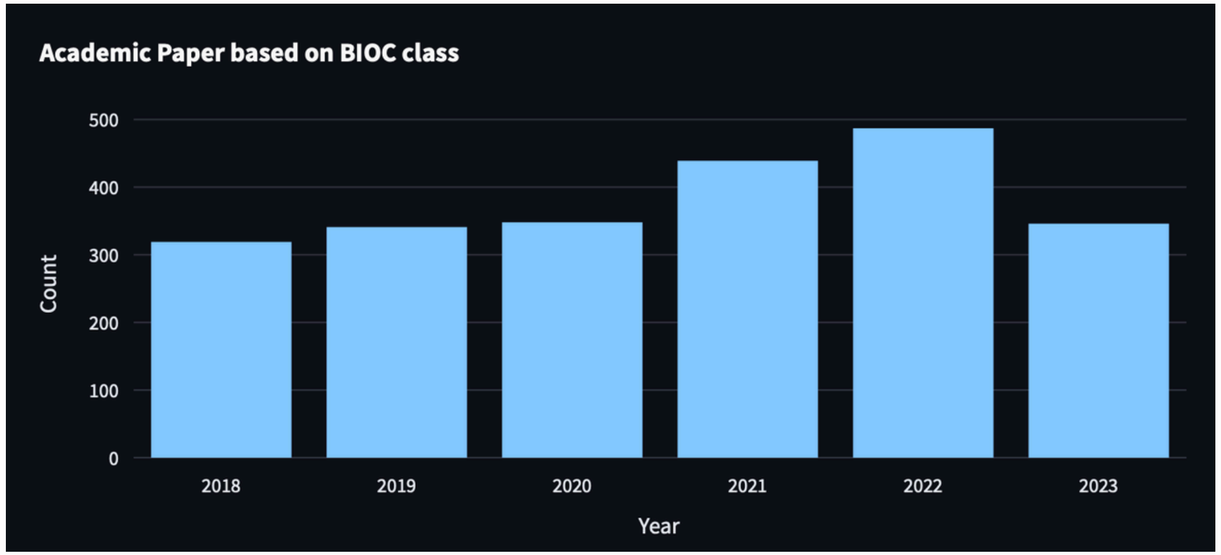
by Classification Type (SUBJABBR Field)

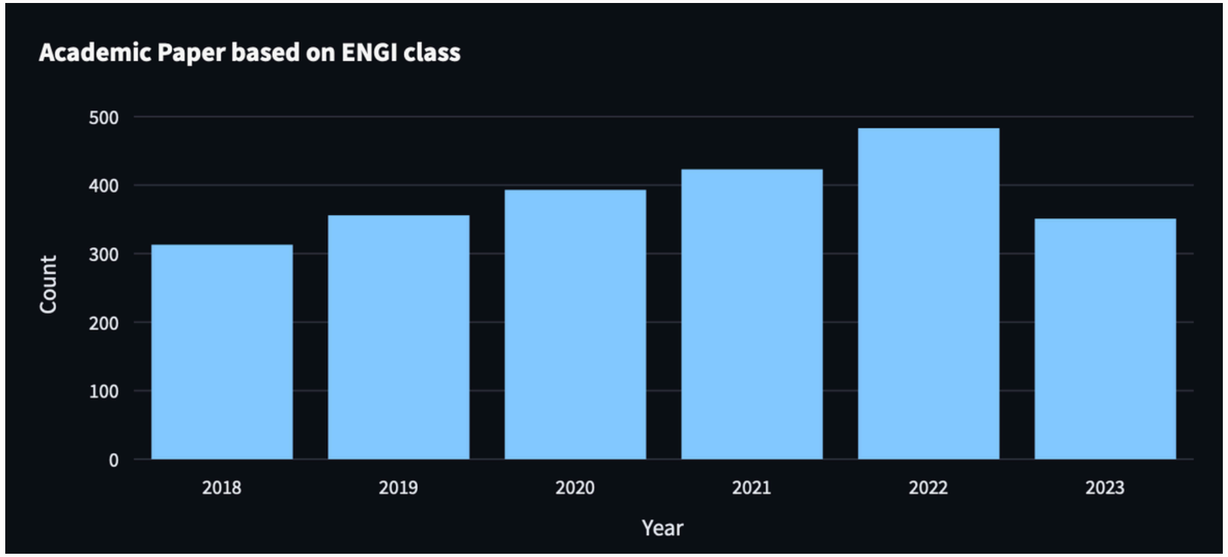


by year

Top 3 Classification Type





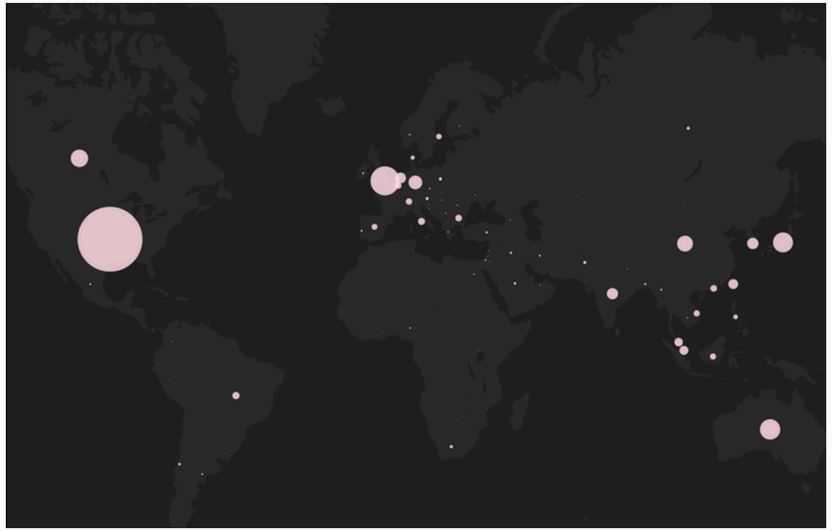


PART II:

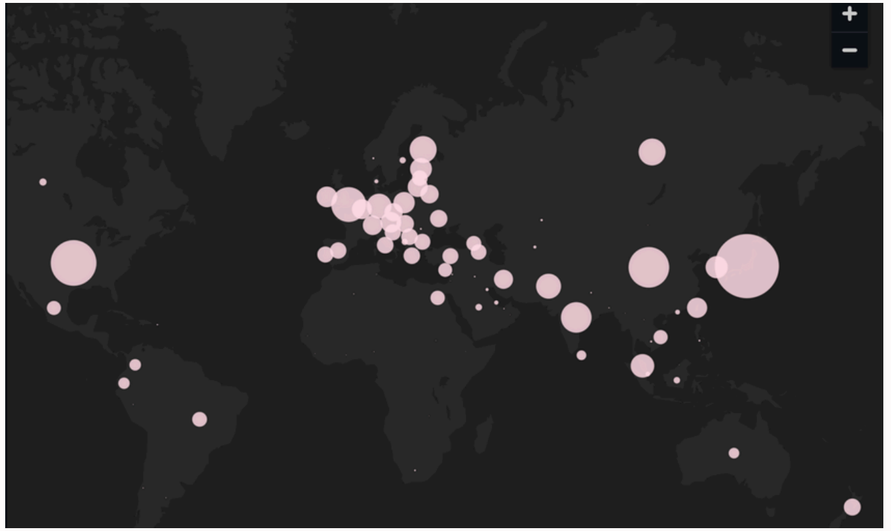
CHULALONGKORN'S RESEARCH COLLABORATION (SCOPUS DATA)

Chulalongkorn’s Academic Paper Research Collaboration during 2018 - 2023

Medical



Engineering



BioChemical

PART III:

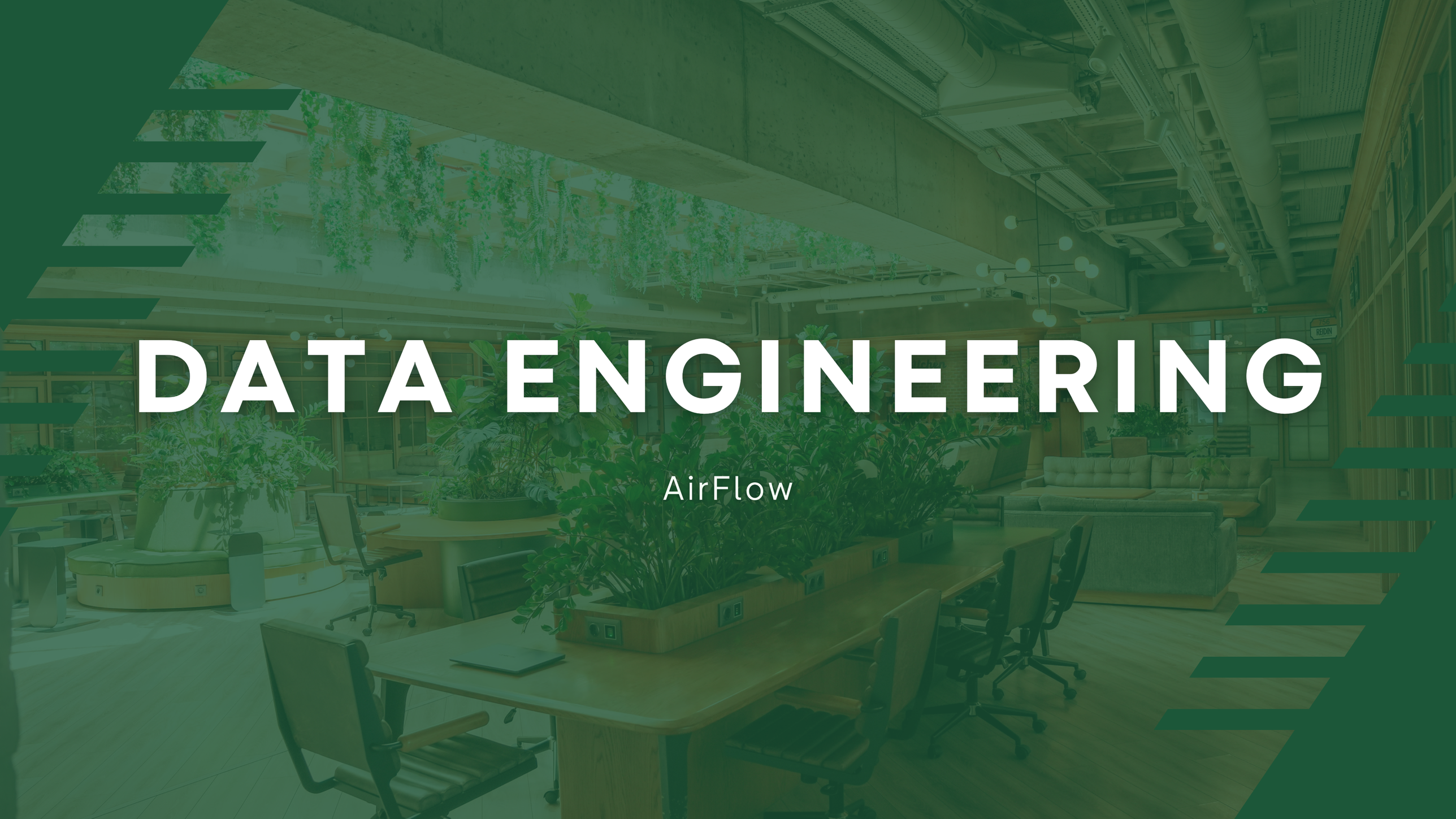
FOREIGN PAPER ON TOP 3 CLASSIFICATION TYPE (SCRAPING DATA)

Foreign Scraped Academic Paper

Medical

Engineering

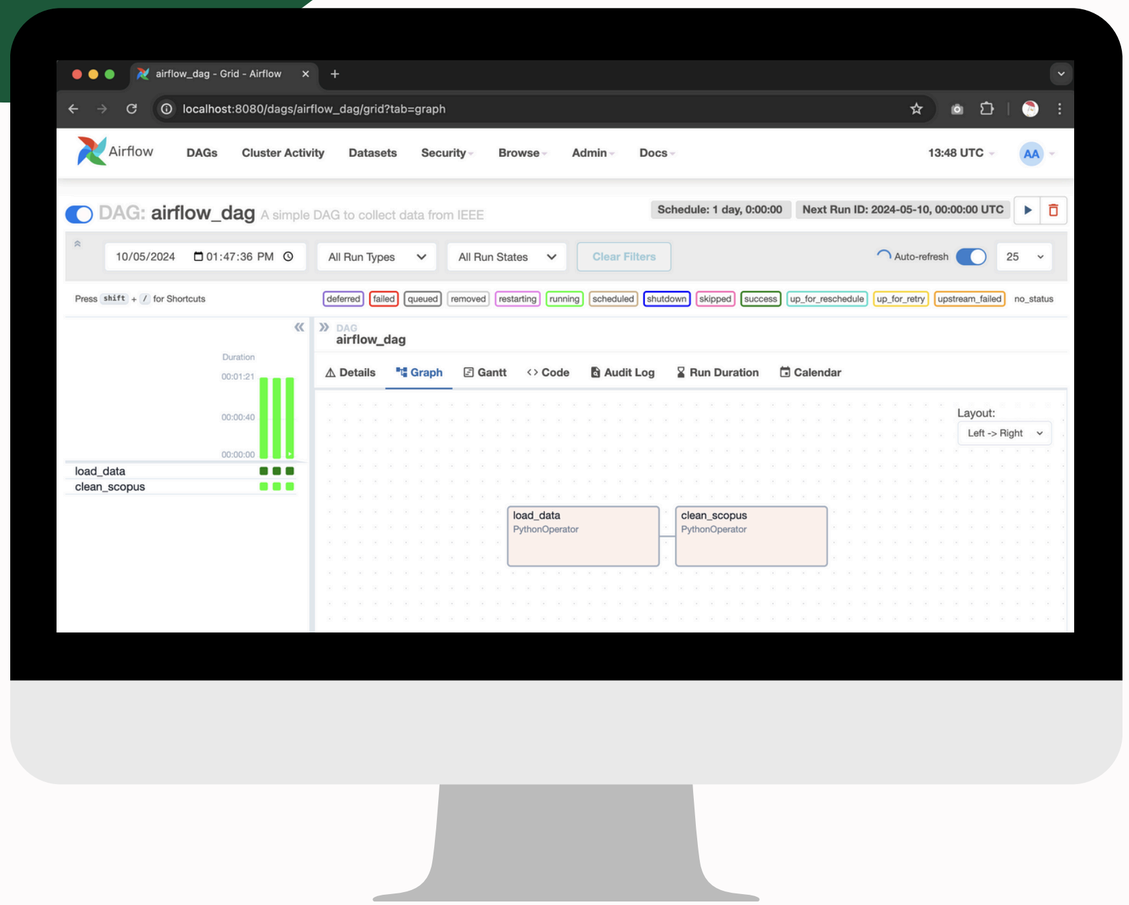
BioChemical



DATA ENGINEERING

AirFlow

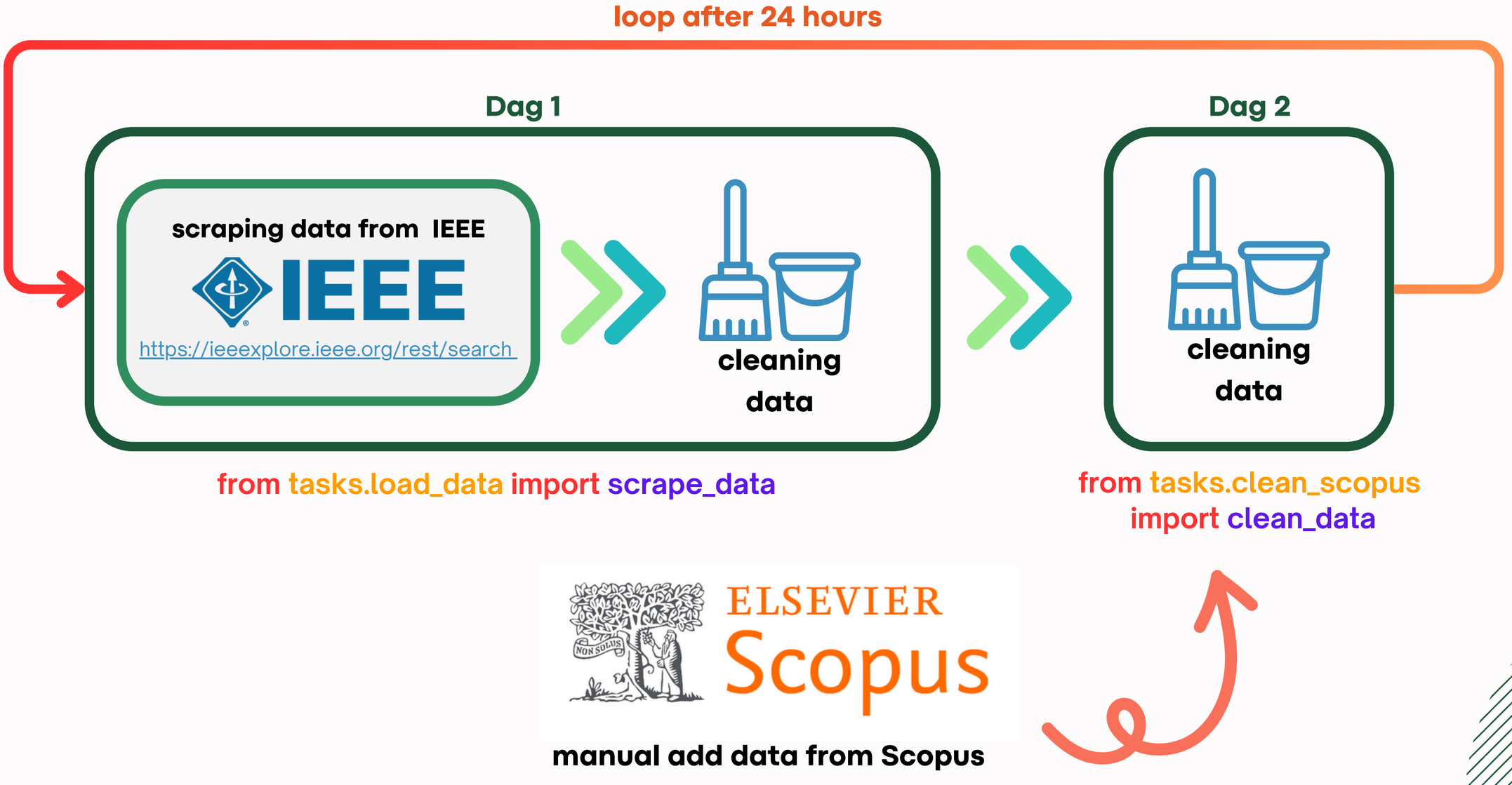
AirFlow



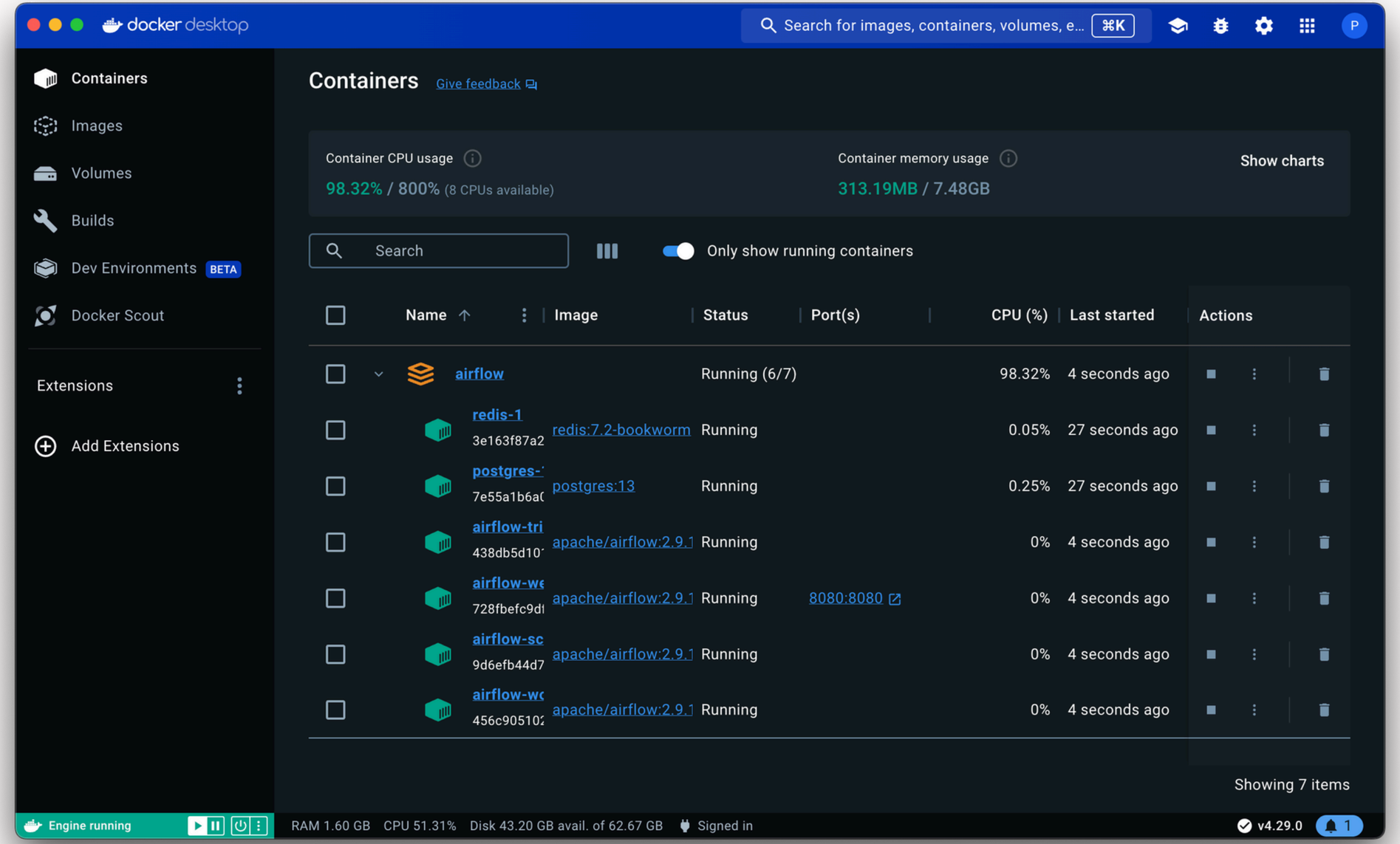
2 Dags

- Retrieve&Clean data from IEEE
- Cleaning Data

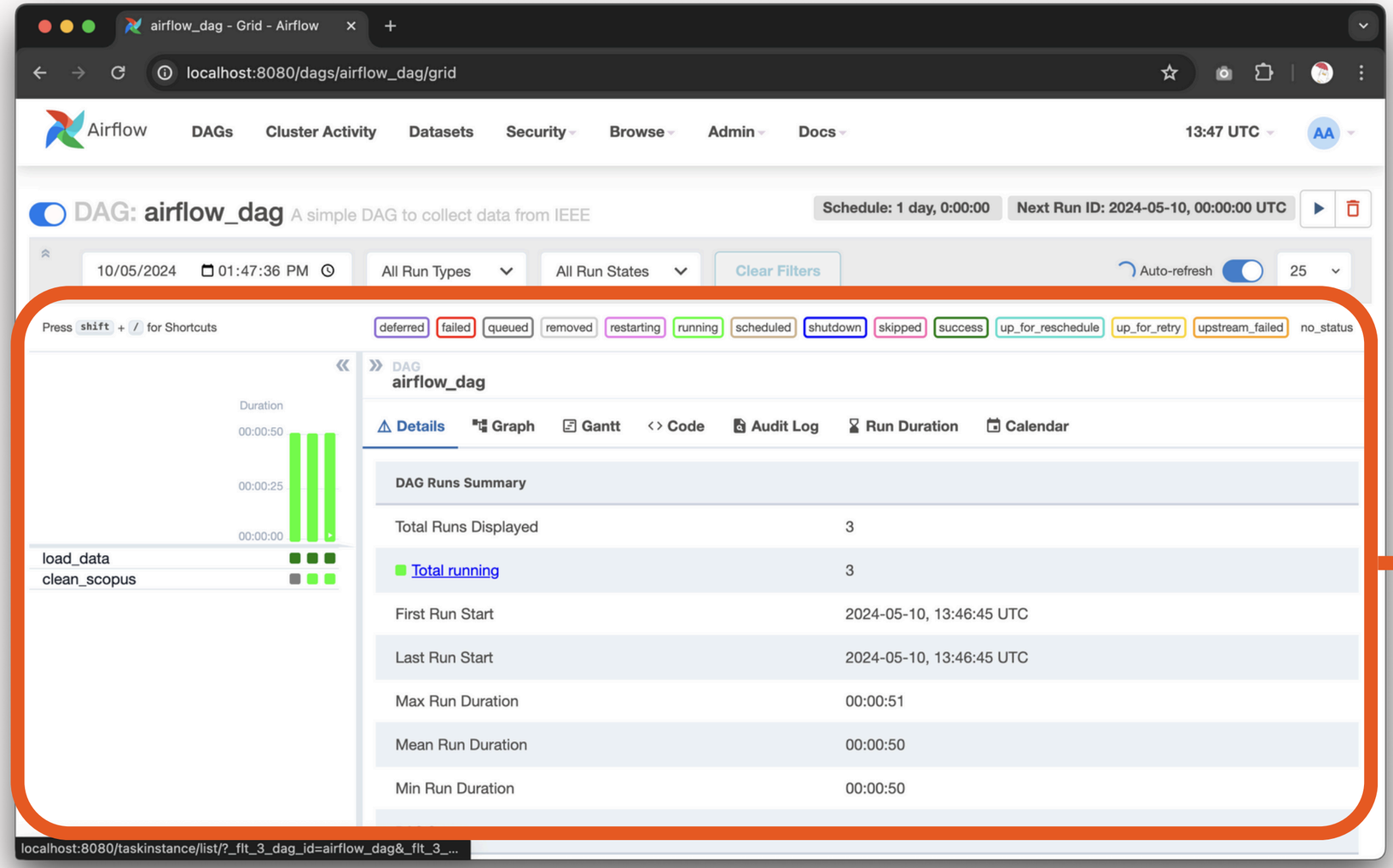
Airflow: Process



Airflow: Docker

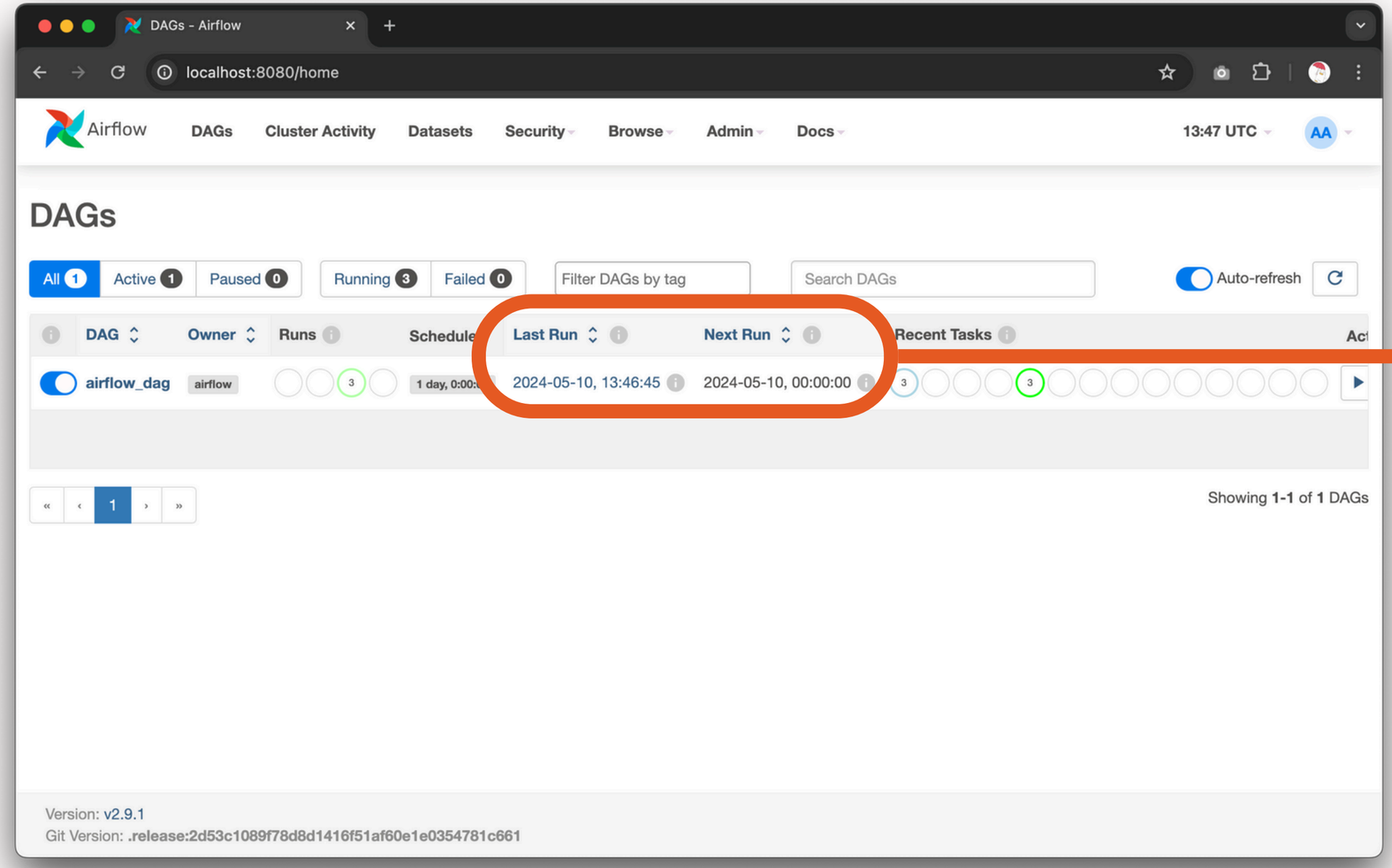


Airflow: Running Status



Running Status

Airflow: Automate



Daily run at 00:00

Clean_IEEE

Divide papers according to author

```python
def scrape_each_author(author):
    if "id" not in author:
        return None
    authorId = author["id"]
    global memoization
    if authorId in memoization.keys():
        return memoization[authorId]

    url = f"https://ieeexplore.ieee.org/rest/author/{authorId}"
    headers = {
        "accept": "application/json, text/plain, */*",
        "accept-language": "th-TH,th;q=0.9",
        "content-type": "application/json",
        "origin": "https://ieeexplore.ieee.org",
        "priority": "u=1, i",
        "referer": "https://ieeexplore.ieee.org/search/searchresult.jsp?newsearch=true&queryText=Engineering",
        "sec-ch-ua": '"Chromium";v="124", "Google Chrome";v="124", "Not-A.Brand";v="99"',
        "sec-ch-ua-mobile": "?0",
        "sec-ch-ua-platform": '"Windows"',
        "sec-fetch-dest": "empty",
        "sec-fetch-mode": "cors",
        "sec-fetch-site": "same-origin",
        "user-agent": "Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/",
    }

    response = requests.get(url, headers=headers)
    if response.status_code != 200:
        memoization[authorId] = None
        return None
    response.raise_for_status()  # Raise an exception for unsuccessful requests

    data = response.json()
    if not len(data):
        memoization[authorId] = None
        return None
    currentAffilations = data[0].get("currentAffiliations", [])
```

```python
def scrape_data():
    selected_df["authors"]

    selected_df["authorsAffilationCountry"] = selected_df["authors"].apply(
        lambda x: [scrape_each_author(author) for author in x]
    )
    finished_df = selected_df
    finished_df["extracted_class"] = ""

    finished_df.loc[:999, "extracted_class"] = "MEDI"
    finished_df.loc[1000:1999, "extracted_class"] = "ENGI"
    finished_df.loc[2000:2999, "extracted_class"] = "BIOC"
    cur_path = os.path.dirname(os.path.realpath(__file__))
    path = os.path.join(cur_path, "scraped_data.csv")
    finished_df.to_csv(path)
```

Clean_Scopus

```python
def clean_data():
    cur_path = os.path.dirname(os.path.realpath(__file__))
    fp = os.path.join(cur_path, "scopus_data")
    years = ["2018", "2019", "2020", "2021", "2022", "2023"]
    dfs = []

    for year in years:
        folder_path = os.path.join(fp, year)
        for filename in os.listdir(folder_path):
            file_path = os.path.join(folder_path, filename)
            if os.path.isfile(file_path):
                with open(file_path, "r") as file:
                    try:
                        data = json.load(file)
                        df = pd.json_normalize(data)
                        dfs.append(df)
                        print(f"Successfully reading {file_path}")
                    except json.JSONDecodeError:
                        print(f"Skipping {file_path}. Not a valid JSON file.")
            # count_test += 1
            # if(count_test == 5) :
            #     break
    df = pd.concat(dfs, ignore_index=True)
```

```python
def map_country_coordinates(country_code):
    return country_coordinates.get(country_code, {"Latitude": None, "Longitude": None})


def extract_affiliation_country(author_group):
    try:
        return [entry["affiliation"]["@country"] for entry in author_group]
    except KeyError:
        return None


def extract_class(cell_value):
    # print(isinstance(cell_value, str))
    if isinstance(cell_value, str):
        return cell_value
    else:
        return ", ".join([d["$"] for d in cell_value])


def to_str(cell_value):
    return str(cell_value)


def extract_values(row):
    tags = row["paper_type"][-1]
    return pd.Series({"tags": tags})
```

Clean_Scopus

```python
import pandas as pd
import os
import json

country_coordinates = {
    "tha": {"Latitude": 13.75, "Longitude": 100.5167},
    "usa": {"Latitude": 37.0902, "Longitude": -95.7129},
    "gbr": {"Latitude": 51.509865, "Longitude": -0.118092},
    "deu": {"Latitude": 51.1657, "Longitude": 10.4515},
    "jpn": {"Latitude": 36.2048, "Longitude": 138.2529},
    "nor": {"Latitude": 60.472, "Longitude": 8.4689},
    "bel": {"Latitude": 50.5039, "Longitude": 4.4699},
    "swe": {"Latitude": 60.1282, "Longitude": 18.6435},
    "aus": {"Latitude": -25.2744, "Longitude": 133.7751},
    "idn": {"Latitude": -0.7893, "Longitude": 113.9213},
    "chn": {"Latitude": 35.8617, "Longitude": 104.1954},
    "chl": {"Latitude": -35.6751, "Longitude": -71.543},
    "ind": {"Latitude": 20.5937, "Longitude": 78.9629},
    "pak": {"Latitude": 30.3753, "Longitude": 69.3451},
    "mex": {"Latitude": 23.6345, "Longitude": -102.5528},
    "nld": {"Latitude": 52.1326, "Longitude": 5.2913},
    "twn": {"Latitude": 23.6978, "Longitude": 120.9605},
    "btn": {"Latitude": 27.5142, "Longitude": 90.4336},
    "mmr": {"Latitude": 21.9162, "Longitude": 95.956},
    "bra": {"Latitude": -14.235, "Longitude": -51.9253},
    "bgr": {"Latitude": 42.7339, "Longitude": 25.4858},
    "ita": {"Latitude": 41.8719, "Longitude": 12.5674},
    "can": {"Latitude": 56.1304, "Longitude": -106.3468},
    "arm": {"Latitude": 40.0691, "Longitude": 45.0382},
    "aut": {"Latitude": 47.5162, "Longitude": 14.5501},
    "blr": {"Latitude": 53.7098, "Longitude": 27.9534},
    "col": {"Latitude": 4.5709, "Longitude": -74.2973},
    "hrv": {"Latitude": 45.1, "Longitude": 15.2},
    "cyp": {"Latitude": 35.1264, "Longitude": 33.4299},
    "cze": {"Latitude": 49.8175, "Longitude": 15.473},
    "ecu": {"Latitude": -1.8312, "Longitude": -78.1834},
    "egy": {"Latitude": 26.8206, "Longitude": 30.8025},
    "est": {"Latitude": 58.5953, "Longitude": 25.0136},
```

	country_code	latitude	longitude
0	tha	13.750000	100.516700
1	usa	37.090200	-95.712900
2	gbr	51.509865	-0.118092
3	deu	51.165700	10.451500
4	jpn	36.204800	138.252900
...	...	...	...
161	brb	13.193900	-59.543200
162	fsm	7.425600	150.550800
163	cpv	16.538800	-23.041800
164	lbr	6.428100	-9.429500
165	mlt	35.937500	14.375400

SUMMARY

DATA ENGINEER

What my friends think I do

What my mom thinks I do

What society thinks I do

What my spouse thinks I do

What I think I do

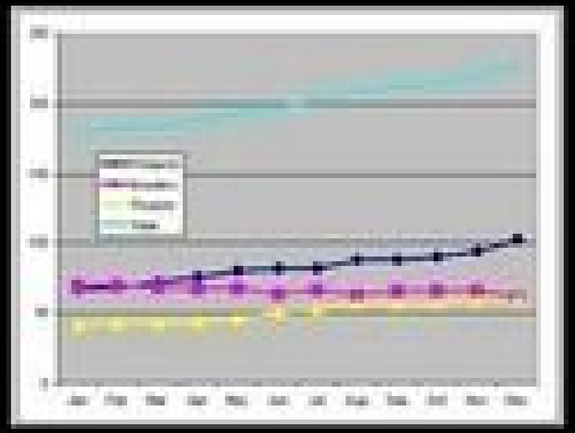
What I actually do

Data Scientist

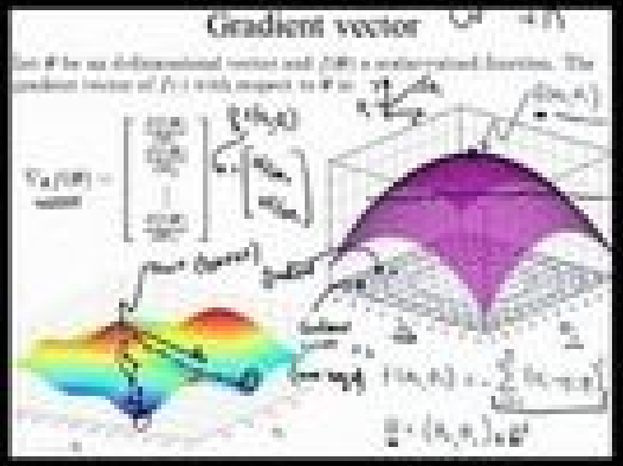
What my friends think I do

What my mom thinks I do

What society thinks I do



What my boss thinks I do



What I think I do

What I actually do

THANK YOU