Digital Twin

The aim of this collab is to retrieve data related to “Digital Twin” topics from GitHub, as well as from the Software Heritage platform.

```
import requests
import csv
import time
import pandas as pd
import matplotlib.pyplot as plt
```

```
GITHUB_TOKEN = "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"
```

You need to add you're GitHub token

```
HEADERS = {"Authorization": f"token {GITHUB_TOKEN}"}
```

```
QUERY = "digital twin"
MAX_RESULTS = 2000
PER_PAGE = 100
```

Limitation of 2000 result (can be changed but anyway github api limitation is 1000, no limitation on Software Heritage)

```
def search_github_repositories(query, max_results=500, per_page=100, sort_by="stars"):
    repositories = []
    page = 1

    while len(repositories) < max_results:
        url = f"https://api.github.com/search/repositories?q={query}&per_page={per_page}&page={page}&sort={sort_by}"
        response = requests.get(url, headers=HEADERS)
        try:
            data = response.json()
        except requests.exceptions.JSONDecodeError:
            print("Error: Empty or badly formatted response received from GitHub API")
            break

        if response.status_code != 200:
            print(f"Error : {response.status_code} - {data.get('message')}")
            break

        repositories.extend(data['items'])

        if len(data['items']) < per_page:
            break

        page += 1
        time.sleep(1)  # Pause to avoid API limitations

    return repositories[:max_results]
```

```
def get_repository_details(repo):
    repo_url = repo['url']
    contributors_url = repo_url + "/contributors"
    commits_url = repo_url + "/commits"

    try:
        contributors_response = requests.get(contributors_url, headers=HEADERS)
        commits_response = requests.get(commits_url, headers=HEADERS)
        contributors_count = len(contributors_response.json()) if contributors_response.status_code == 200 else 0
        commits_count = len(commits_response.json()) if commits_response.status_code == 200 else 0
    except requests.exceptions.JSONDecodeError:
        contributors_count = 0
        commits_count = 0

    open_issues_count = repo['open_issues_count']

    return contributors_count, commits_count, open_issues_count
```

```
def save_to_csv(repositories, filename="digital_twin_repos_gitHub.csv"):
    with open(filename, mode='w', newline='', encoding='utf-8') as file:
        writer = csv.writer(file)
        writer.writerow(["Name", "Stars", "Forks", "Language", "Description", "URL", "Contributors", "Commits", "Open Issues"])
```

```
        for repo in repositories:
            contributors, commits, open_issues = get_repository_details(repo)
            writer.writerow([
                repo['name'],
                repo['stargazers_count'],
                repo['forks_count'],
                repo['language'],
                repo['description'],
                repo['html_url'],
                contributors,
                commits,
                open_issues
            ])
```

```
print(f"Search GitHub projects for : {QUERY}")
repos = search_github_repositories(QUERY, MAX_RESULTS, PER_PAGE, sort_by="stars")
print(f"{len(repos)} projects found.")
save_to_csv(repos)
print("The results were saved in 'digital_twin_repos_gitHub.csv'")
```

```
Search GitHub projects for : digital twin
Error : 422 - Only the first 1000 search results are available
1000 projects found.
The results were saved in 'digital_twin_repos_gitHub.csv'
```

We have obtained 1000 GitHub repo related to the subject of “digital twin”.

We will now move on to the Software Heritage search.

```
def search_projects_by_metadata():
    metarequest = 0
    projects = []
    base_url = "https://archive.softwareheritage.org/api/1/origin/metadata-search/"
    params = {"fulltext": {QUERY}, "per_page": 50}
    next_page = base_url

    while next_page and metarequest < MAX_RESULTS:
        metarequest += 50
        response = requests.get(next_page, params=params)
        if response.status_code == 200:
            data = response.json()
            projects.extend(data)  # Ajouter les résultats
            # L'API de Software Heritage ne fournit pas toujours un lien "next"
            if len(data) < 50:
                break  # Plus de pages à récupérer

            time.sleep(1)

        else:
            print("Erreur lors de la requête:", response.status_code, response.text)
            break

    return projects
```

```
print(f"Search Software Heritage projects for : {QUERY}")
projects_metadata = search_projects_by_metadata()
print(f"{len(projects_metadata)} projects found.")
df = pd.DataFrame(projects_metadata)
file_path = "digital_twin_repos_softwareHeritage.csv"
df.to_csv(file_path, index=False)
print("The results were saved in 'digital_twin_repos_softwareHeritage.csv'")
```

```
Search Software Heritage projects for : digital twin
2800 projects found.
The results were saved in 'digital_twin_repos_softwareHeritage.csv'
```

We have obtained 1400 Software repo related to the subject of “digital twin”.

We are now going to compare the result

```
file_repos_gitHub = "digital_twin_repos_gitHub.csv"
file_repos_softwareHeritage = "digital_twin_repos_softwareHeritage.csv"
```

```
df_repos_gitHub = pd.read_csv(file_repos_gitHub)
df_repos_softwareHeritage = pd.read_csv(file_repos_softwareHeritage)
```

```
df_repos_gitHub["URL"] = df_repos_gitHub["URL"].str.lower().str.strip()
df_repos_softwareHeritage["url"] = df_repos_softwareHeritage["url"].str.lower().str.strip()
```

```
duplicates = df_repos_gitHub[df_repos_gitHub["URL"].isin(df_repos_softwareHeritage["url"])]
```

```
output_file = "duplicate_repositories.csv"
duplicates.to_csv(output_file, index=False)
print(f"Number of duplicates found : {len(duplicates)}")
print(f"The results were recorded in : {output_file}")
```

```
Number of duplicates found : 25
The results were recorded in : duplicate_repositories.csv
```

Only 22 git repo are common to both databases.

We will retrieve detailed information from the gitHub repo found on software heritage

```
def extract_github_repo(url):
    if "github.com" in url:
        parts = url.split("github.com/")[-1].split("/")
        if len(parts) >= 2:
            return f"{parts[0]}/{parts[1]}"
    return None
```

```
new_repos = df_repos_softwareHeritage[~df_repos_softwareHeritage["url"].isin(df_repos_gitHub["URL"])]
```

```
new_repos["github_repo"] = new_repos["url"].apply(extract_github_repo)
new_repos = new_repos.dropna(subset=["github_repo"])
```

```
<ipython-input-18-4d77948eed0a>:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-v
  new_repos["github_repo"] = new_repos["url"].apply(extract_github_repo)
```

We have all the url of the new repo github not already recovered

```
def collectDataFromRepo(repos,headers=HEADERS):
  repo_data = []

  for repo in repos["github_repo"]:
    url = f"https://api.github.com/repos/{repo}"

    try:
        response = requests.get(url, headers=headers)
        if response.status_code == 200:
            data = response.json()
            repo_data.append({
                "Name": data.get("name"),
                "Stars": data.get("stargazers_count"),
                "Forks": data.get("forks_count"),
                "Language": data.get("language"),
                "Description": data.get("description"),
                "URL": data.get("html_url"),
                "Contributors": len(requests.get(data["contributors_url"], headers=headers).json()),
                "Commits": len(requests.get(data["commits_url"].replace("{/sha}", ""), headers=headers).json()),
                "Open Issues": data.get("open_issues_count"),
            })
        else:
            print(f"Error for {repo} : {response.status_code}")
    except Exception as e:
        print(f"Exception for {repo} : {e}")

    time.sleep(1)

  return repo_data
```

```
repo_data = collectDataFromRepo(new_repos)
df_new_repos = pd.DataFrame(repo_data)
output_file = "new_digital_twin_repos.csv"
df_new_repos.to_csv(output_file, index=False)

print(f"Data for {len(df_new_repos)} new repo.")
print(f"File saved : {output_file}")
```

```
Error for azure/digital-twin-model-identifier : 404
Exception for sobercfd/cfd-digital-twin : Expecting value: line 1 column 1 (char 0)
Error for hiteshyadav007/c-iot-1 : 404
Error for wldt/wldt-ikea-tradfri : 404
Error for wldt/wldt-coap-example : 404
Exception for madmuc/dt : Expecting value: line 1 column 1 (char 0)
Exception for skumarjain11/iottwinmaker : Expecting value: line 1 column 1 (char 0)
Exception for dainguyenhuu/daidigital_coder : Expecting value: line 1 column 1 (char 0)
Exception for sameuniversum/digi-you : Expecting value: line 1 column 1 (char 0)
Exception for aryagargi07/digital-twin : Expecting value: line 1 column 1 (char 0)
Exception for sainitripti/server : Expecting value: line 1 column 1 (char 0)
Error for azure/digital-twin-model-identifier : 404
Exception for sobercfd/cfd-digital-twin : Expecting value: line 1 column 1 (char 0)
Error for hiteshyadav007/c-iot-1 : 404
Error for wldt/wldt-ikea-tradfri : 404
Error for wldt/wldt-coap-example : 404
Exception for madmuc/dt : Expecting value: line 1 column 1 (char 0)
Exception for skumarjain11/iottwinmaker : Expecting value: line 1 column 1 (char 0)
Exception for dainguyenhuu/daidigital_coder : Expecting value: line 1 column 1 (char 0)
Exception for sameuniversum/digi-you : Expecting value: line 1 column 1 (char 0)
Exception for aryagargi07/digital-twin : Expecting value: line 1 column 1 (char 0)
Exception for sainitripti/server : Expecting value: line 1 column 1 (char 0)
Error for azure/digital-twin-model-identifier : 404
Exception for sobercfd/cfd-digital-twin : Expecting value: line 1 column 1 (char 0)
Error for hiteshyadav007/c-iot-1 : 404
Error for wldt/wldt-ikea-tradfri : 404
Error for wldt/wldt-coap-example : 404
Exception for madmuc/dt : Expecting value: line 1 column 1 (char 0)
Exception for skumarjain11/iottwinmaker : Expecting value: line 1 column 1 (char 0)
Exception for dainguyenhuu/daidigital_coder : Expecting value: line 1 column 1 (char 0)
Exception for sameuniversum/digi-you : Expecting value: line 1 column 1 (char 0)
Exception for aryagargi07/digital-twin : Expecting value: line 1 column 1 (char 0)
Exception for sainitripti/server : Expecting value: line 1 column 1 (char 0)
Error for azure/digital-twin-model-identifier : 404
Exception for sobercfd/cfd-digital-twin : Expecting value: line 1 column 1 (char 0)
Error for hiteshyadav007/c-iot-1 : 404
Error for wldt/wldt-ikea-tradfri : 404
Error for wldt/wldt-coap-example : 404
Exception for madmuc/dt : Expecting value: line 1 column 1 (char 0)
Exception for skumarjain11/iottwinmaker : Expecting value: line 1 column 1 (char 0)
Exception for dainguyenhuu/daidigital_coder : Expecting value: line 1 column 1 (char 0)
Exception for sameuniversum/digi-you : Expecting value: line 1 column 1 (char 0)
Exception for aryagargi07/digital-twin : Expecting value: line 1 column 1 (char 0)
Exception for sainitripti/server : Expecting value: line 1 column 1 (char 0)
Error for azure/digital-twin-model-identifier : 404
Exception for sobercfd/cfd-digital-twin : Expecting value: line 1 column 1 (char 0)
Error for hiteshyadav007/c-iot-1 : 404
Error for wldt/wldt-ikea-tradfri : 404
Error for wldt/wldt-coap-example : 404
Exception for madmuc/dt : Expecting value: line 1 column 1 (char 0)
Exception for skumarjain11/iottwinmaker : Expecting value: line 1 column 1 (char 0)
Exception for dainguyenhuu/daidigital_coder : Expecting value: line 1 column 1 (char 0)
Exception for sameuniversum/digi-you : Expecting value: line 1 column 1 (char 0)
Exception for aryagargi07/digital-twin : Expecting value: line 1 column 1 (char 0)
Error for abh4git/digitaltwin : 404
Exception for sainitripti/server : Expecting value: line 1 column 1 (char 0)
Data for 1442 new repo.
File saved : new_digital_twin_repos.csv
```

We finally have 740 new github repositories from Software heritage. Some repositories are no longer available, others are not hosted by github.

We'll now take a look at some key data on recovered repositories

```
import matplotlib.pyplot as plt
import seaborn as sns
```

```
# Fusionner les DataFrames sur la colonne 'URL'
merged_df = df_repos_gitHub.set_index('URL').combine_first(df_new_repos.set_index('URL')).reset_index()
```

```
print(df_repos_gitHub.info())
print(df_new_repos.info())
print(merged_df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
```

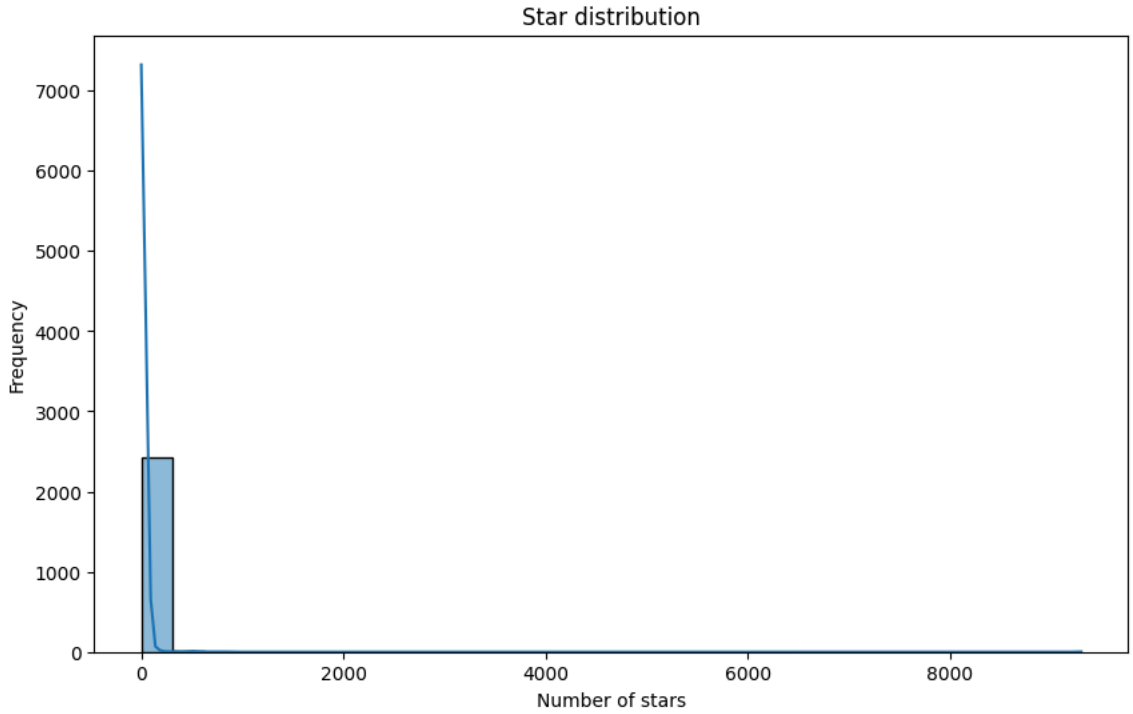
```
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Name          1000 non-null   object
 1   Stars         1000 non-null   int64
 2   Forks         1000 non-null   int64
 3   Language      895 non-null    object
 4   Description   766 non-null    object
 5   URL           1000 non-null   object
 6   Contributors  1000 non-null   int64
 7   Commits       1000 non-null   int64
 8   Open Issues   1000 non-null   int64
dtypes: int64(5), object(4)
memory usage: 70.4+ KB
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1442 entries, 0 to 1441
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Name          1442 non-null   object
 1   Stars         1442 non-null   int64
 2   Forks         1442 non-null   int64
 3   Language      753 non-null    object
 4   Description   1442 non-null   object
 5   URL           1442 non-null   object
 6   Contributors  1442 non-null   int64
 7   Commits       1442 non-null   int64
 8   Open Issues   1442 non-null   int64
dtypes: int64(5), object(4)
memory usage: 101.5+ KB
None
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2442 entries, 0 to 2441
Data columns (total 9 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   URL           2442 non-null   object
 1   Name          2442 non-null   object
 2   Stars         2442 non-null   int64
 3   Forks         2442 non-null   int64
 4   Language      1648 non-null   object
 5   Description   2208 non-null   object
 6   Contributors  2442 non-null   int64
 7   Commits       2442 non-null   int64
 8   Open Issues   2442 non-null   int64
dtypes: int64(5), object(4)
memory usage: 171.8+ KB
None
```

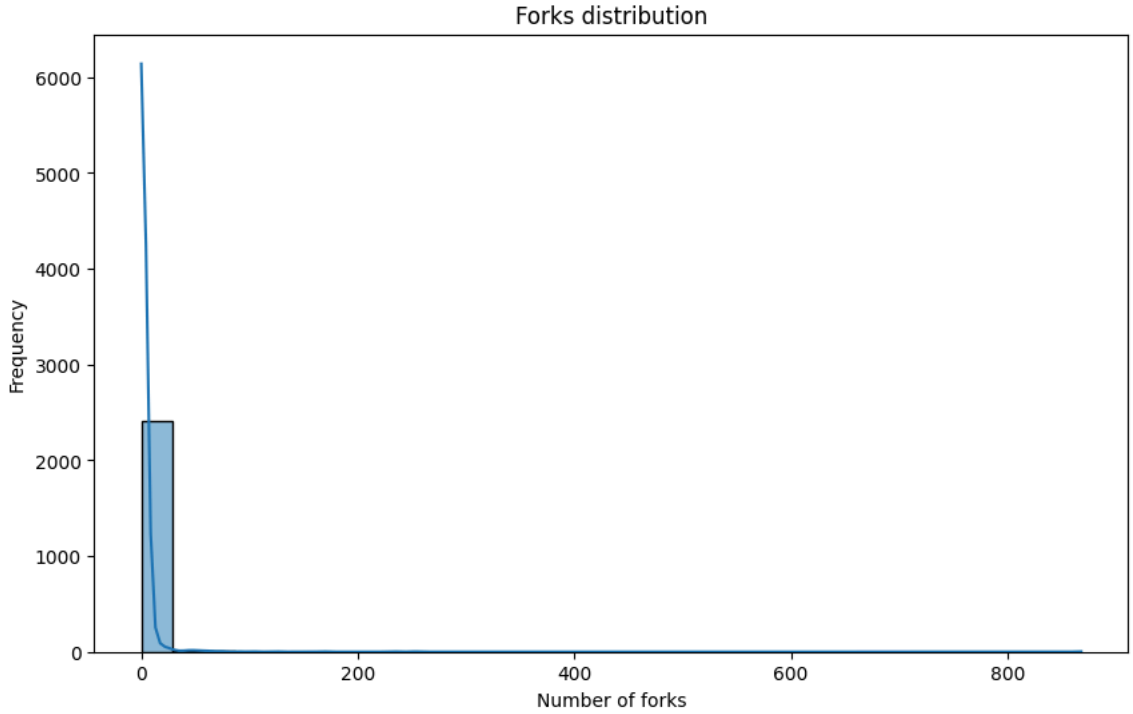
```
print(df_repos_gitHub.describe())
print(df_new_repos.describe())
print(merged_df.describe())
```

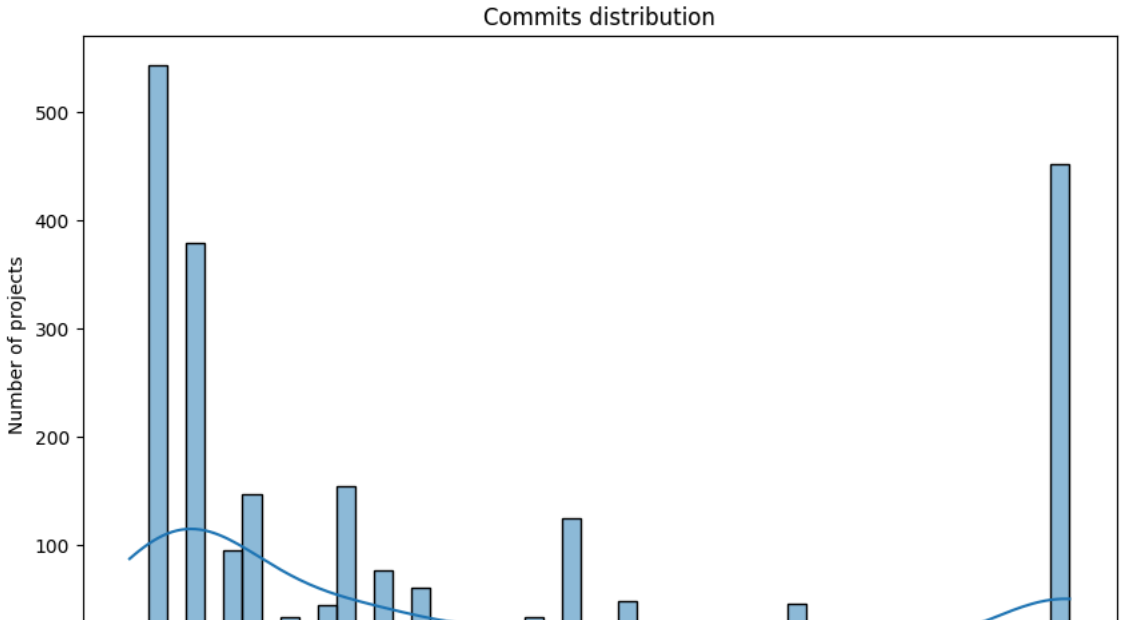
```
              Stars        Forks  Contributors      Commits  Open Issues
count  1000.000000  1000.000000   1000.000000  1000.000000  1000.000000
mean     22.415000     4.795000      2.484000    17.918000     2.321000
std     298.787136    31.321207      3.453066    11.664447     9.569511
min       1.000000     0.000000      0.000000     0.000000     0.000000
25%       1.000000     0.000000      1.000000     6.000000     0.000000
50%       3.000000     1.000000      1.000000    19.000000     0.000000
75%       7.000000     2.000000      2.000000    30.000000     1.000000
max    9300.000000   868.000000     30.000000    30.000000   170.000000
              Stars        Forks  Contributors      Commits  Open Issues
count  1442.000000  1442.000000   1442.000000  1442.000000  1442.000000
mean      0.027739     0.277393      1.221221     5.740638     0.776699
std       0.164282     0.650118      0.724524     7.091207     4.109036
min       0.000000     0.000000      0.000000     1.000000     0.000000
25%       0.000000     0.000000      1.000000     1.000000     0.000000
50%       0.000000     0.000000      1.000000     2.000000     0.000000
75%       0.000000     0.000000      1.000000     7.000000     0.000000
max       1.000000     3.000000      4.000000    30.000000    25.000000
              Stars        Forks  Contributors      Commits  Open Issues
count  2442.000000  2442.000000   2442.000000  2442.000000  2442.000000
mean      9.195332     2.127355      1.738329    10.727273     1.409091
std     191.460963    20.166222      2.361249    11.010932     6.929809
min       0.000000     0.000000      0.000000     0.000000     0.000000
25%       0.000000     0.000000      1.000000     2.000000     0.000000
50%       0.000000     0.000000      1.000000     6.000000     0.000000
75%       2.000000     1.000000      2.000000    18.000000     0.000000
max    9300.000000   868.000000     30.000000    30.000000   170.000000
```

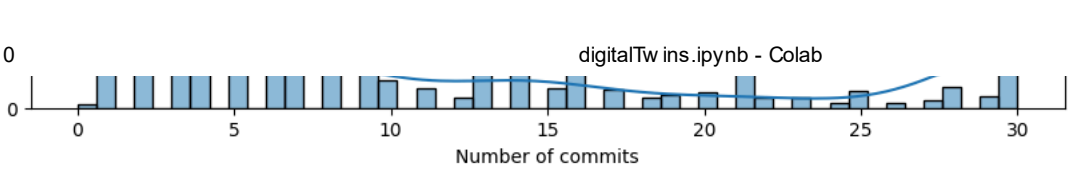
```
plt.figure(figsize=(10, 6))
sns.histplot(merged_df['Stars'], bins=30, kde=True)
plt.title('Star distribution')
plt.xlabel('Number of stars')
plt.ylabel('Frequency')
plt.show()
```

```
plt.figure(figsize=(10, 6))
sns.histplot(merged_df['Forks'], bins=30, kde=True)
plt.title('Forks distribution')
plt.xlabel('Number of forks')
plt.ylabel('Frequency')
plt.show()

plt.figure(figsize=(10, 6))
sns.histplot(merged_df['Commits'], bins=50, kde=True)
plt.title('Commits distribution')
plt.xlabel('Number of commits')
plt.ylabel('Number of projects')
plt.show()
```


Star distribution
Frequency
Number of stars
0
1000
2000
3000
4000
5000
6000
7000
0
2000
4000
6000
8000

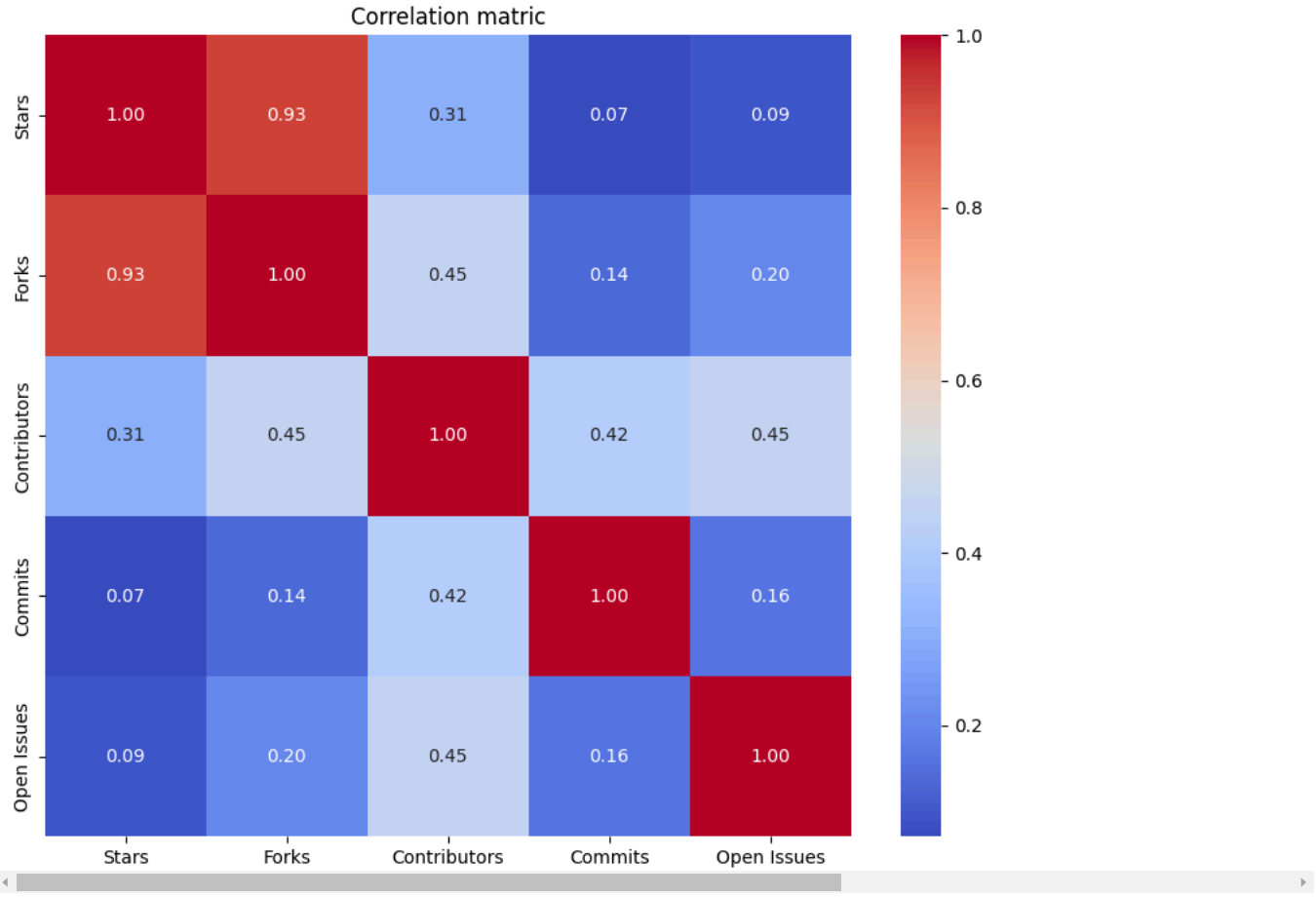

Forks distribution
Frequency
Number of forks
0
1000
2000
3000
4000
5000
6000
0
200
400
600
800


Commits distribution
Number of projects
100
200
300
400
500

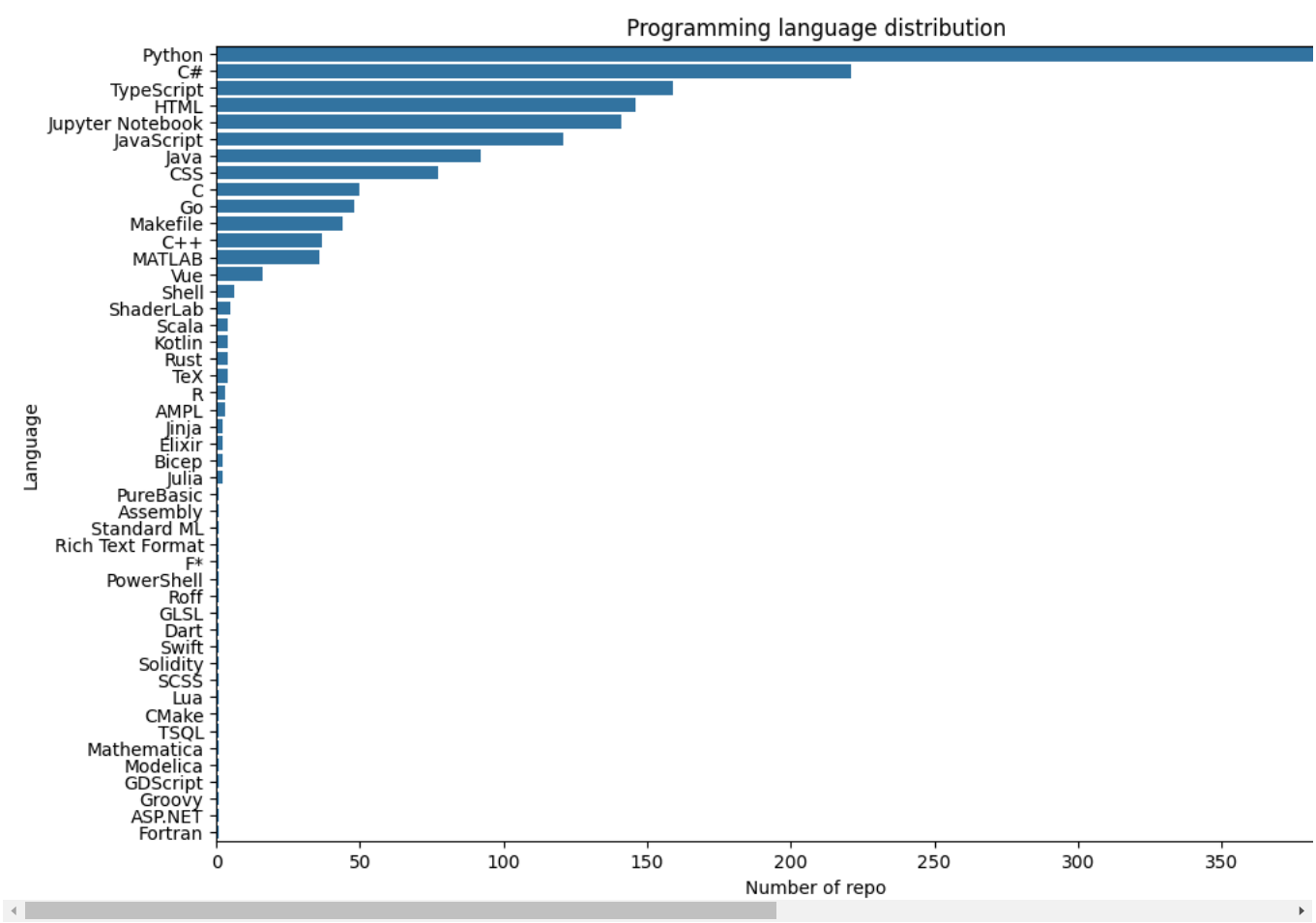



```
numeric_columns = merged_df.select_dtypes(include=['float64', 'int64']).columns

correlation_matrix = merged_df[numeric_columns].corr()
print(correlation_matrix)

plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Correlation matric')
plt.show()
```

```
                Stars     Forks  Contributors   Commits  Open Issues
Stars        1.000000  0.934079      0.305614  0.071325     0.092956
Forks        0.934079  1.000000      0.453613  0.135088     0.199202
Contributors 0.305614  0.453613      1.000000  0.415674     0.449461
Commits      0.071325  0.135088      0.415674  1.000000     0.164146
Open Issues  0.092956  0.199202      0.449461  0.164146     1.000000
```




```
plt.figure(figsize=(12, 8))
sns.countplot(y='Language', data=merged_df, order=merged_df['Language'].value_counts().index)
plt.title('Programming language distribution')
plt.xlabel('Number of repo')
plt.ylabel('Language')
plt.show()
```

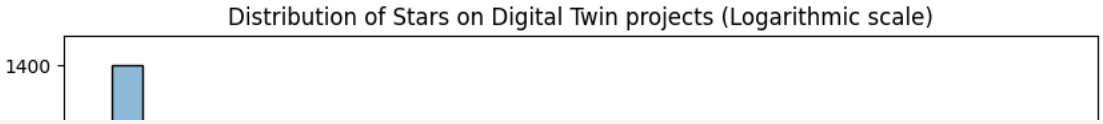


Try to add log for visibility

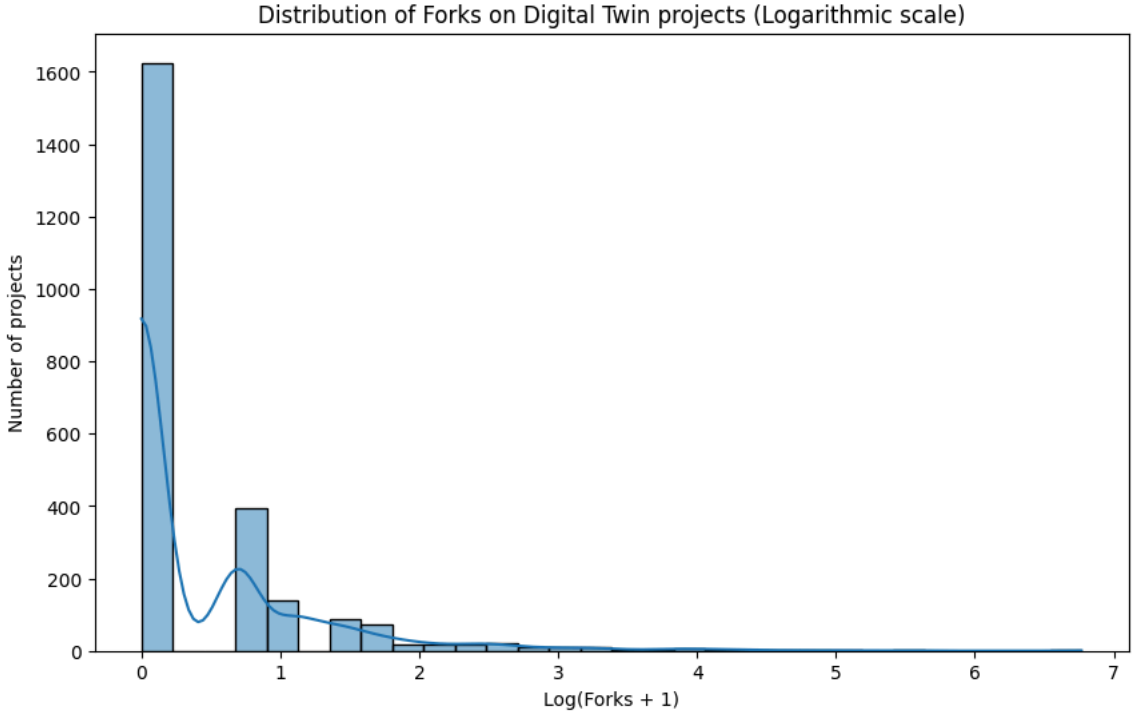
```
import numpy as np
```

```
merged_df['Log_Stars'] = np.log1p(merged_df['Stars'])  # log(1 + x) pour éviter log(0)
merged_df['Log_Forks'] = np.log1p(merged_df['Forks'])
merged_df['Log_Commits'] = np.log1p(merged_df['Commits'])
merged_df['Log_Open_Issues'] = np.log1p(merged_df['Open Issues'])
```

```
plt.figure(figsize=(10, 6))
sns.histplot(merged_df['Log_Stars'], bins=30, kde=True)
plt.title("Distribution of Stars on Digital Twin projects (Logarithmic scale)")
plt.xlabel("Log(Stars + 1)")
plt.ylabel("Number of projects")
plt.show()
```




```
plt.figure(figsize=(10, 6))
sns.histplot(merged_df['Log_Forks'], bins=30, kde=True)
plt.title("Distribution of Forks on Digital Twin projects (Logarithmic scale)")
plt.xlabel("Log(Forks + 1)")
plt.ylabel("Number of projects")
plt.show()
```




```
plt.figure(figsize=(10, 6))
sns.histplot(merged_df['Log_Commits'], bins=30, kde=True)
plt.title("Distribution of commits on Digital Twin projects (Logarithmic scale)")
plt.xlabel("Log(Commits + 1)")
```