In [ ]:
```
#Group 4 - Manpreet Misson; Timothy Gregorian; Sidi Mammar Omar
```

Web Scraping Exercise

Web Scraping allows you to gather large volumes of data from diverse and real-time online sources. This data can be crucial for enriching your datasets, filling in gaps, and providing current information that enhances the quality and relevance of your analysis. Web scraping enables you to collect data that might not be readily available through traditional APIs or databases, offering a competitive edge by incorporating unique and comprehensive insights. Moreover, it automates the data collection process, saving time and resources while ensuring a scalable approach to continuously updating and maintaining your datasets.

Ethical web scraping involves respecting website terms of service, avoiding overloading servers, and ensuring that the collected data is used responsibly and in compliance with privacy laws and regulations.

Use Python, `requests` , `BeautifulSoup` and/or `pandas` to scrape web data:

Import Libraries

In [43]:
```
# TODO
import requests
from bs4 import BeautifulSoup
import pandas as pd
import json
```

Define the Target URL

In [44]:
```
# TODO
url = "https://github.com/trending?since=weekly"
print(url)
```

```
https://github.com/trending?since=weekly
```

Send a Request to the Website

Do not forget to check the response status code

In [45]:
```
# TODO
response = requests.get(url)
if response.status_code != 200:
    print("Fehler beim Abrufen der Seite.")
else:
    print("Seite erfolgreich abgerufen. Status Code: 200")
```

```
Seite erfolgreich abgerufen. Status Code: 200
```

Parse the HTML Content

Use a library to access the HTMl content

In [46]:
```
# TODO
soup = BeautifulSoup(response.text, 'html.parser')
print(soup.prettify()[:500])
```

```
<!DOCTYPE html>
<html data-a11y-animated-images="system" data-a11y-link-underlines="true" data-co
lor-mode="auto" data-dark-theme="dark" data-light-theme="light" lang="en">
 <head>
  <meta charset="utf-8"/>
  <link href="https://github.githubassets.com" rel="dns-prefetch"/>
  <link href="https://avatars.githubusercontent.com" rel="dns-prefetch"/>
  <link href="https://github-cloud.s3.amazonaws.com" rel="dns-prefetch"/>
  <link href="https://user-images.githubusercontent.com/" rel="dns-prefetch"/>
```

Identify the Data to be Scraped

Write a couple of sentence on the data you want to scrape

TODO: I want to scrape the list of trending GitHub repositories for the past month. For each repository, I want to extract the following data:

- Repository name
- Description
- Programming language
- Number of stars
- Number of forks
- Additional metadata like contributors and stars gained during the period (if available)

Extract Data

Find specific elements and extract text or attributes from elements (handle pagination if necessary)

In [47]:
```
all_repo_data = []

repositories = soup.find_all('article', class_='Box-row')
for repo in repositories:
    full_name = repo.h2.a.get('href').strip('/')

    description_tag = repo.p
    description = description_tag.text.strip() if description_tag else "No descr
    lang_tag = repo.find('span', itemprop='programmingLanguage')
    language = lang_tag.text.strip() if lang_tag else "N/A"

    stars_tag = repo.find('a', href=lambda x: x and x.endswith('/stargazers'))
    stars = stars_tag.text.strip().replace(',', '') if stars_tag else "0"

    forks_tag = repo.find('a', href=lambda x: x and '/network/members' in x)
    forks = forks_tag.text.strip().replace(',', '') if forks_tag else "0"

    repo_url = f"https://github.com/{full_name}"

    repo_data = {
        "name": full_name,
        "description": description,
        "language": language,
        "stars": stars,
        "forks": forks,
        "url": repo_url
    }

    all_repo_data.append(repo_data)

    print("-" * 50)
    for key, value in repo_data.items():
        print(f"{key}: {value}")

with open("ScrapedData.json", "w", encoding="utf-8") as f:
    json.dump(all_repo_data, f, ensure_ascii=False, indent=2)
```

```
--------------------------------------------------
name: harry0703/MoneyPrinterTurbo
description: 利用AI大模型，一键生成高清短视频 Generate short videos with one click
using AI LLM.
language: Python
stars: 32179
forks: 0
url: https://github.com/harry0703/MoneyPrinterTurbo
--------------------------------------------------
name: GoogleCloudPlatform/kubectl-ai
description: AI powered Kubernetes Assistant
language: Go
stars: 4931
forks: 0
url: https://github.com/GoogleCloudPlatform/kubectl-ai
--------------------------------------------------
name: voideditor/void
description: No description
language: TypeScript
stars: 18445
forks: 0
url: https://github.com/voideditor/void
--------------------------------------------------
name: LadybirdBrowser/ladybird
description: Truly independent web browser
language: C++
stars: 42092
forks: 0
url: https://github.com/LadybirdBrowser/ladybird
--------------------------------------------------
name: Lightricks/LTX-Video
description: Official repository for LTX-Video
language: Python
stars: 5260
forks: 0
url: https://github.com/Lightricks/LTX-Video
--------------------------------------------------
name: zed-industries/zed
description: Code at the speed of thought – Zed is a high-performance, multiplaye
r code editor from the creators of Atom and Tree-sitter.
language: Rust
stars: 59535
forks: 0
url: https://github.com/zed-industries/zed
--------------------------------------------------
name: MODSetter/SurfSense
description: Open Source Alternative to NotebookLM / Perplexity / Glean, connecte
d to external sources such as search engines (Tavily, Linkup), Slack, Linear, Not
ion, YouTube, GitHub and more.
language: TypeScript
stars: 3810
forks: 0
url: https://github.com/MODSetter/SurfSense
--------------------------------------------------
name: ruanyf/weekly
description: 科技爱好者周刊，每周五发布
language: N/A
stars: 62055
forks: 0
url: https://github.com/ruanyf/weekly
```

```
--------------------------------------------------
name: hacksider/Deep-Live-Cam
description: real time face swap and one-click video deepfake with only a single
image
language: Python
stars: 66690
forks: 0
url: https://github.com/hacksider/Deep-Live-Cam
--------------------------------------------------
name: QwenLM/Qwen-Agent
description: Agent framework and applications built upon Qwen>=3.0, featuring Fun
ction Calling, MCP, Code Interpreter, RAG, Chrome extension, etc.
language: Python
stars: 8491
forks: 0
url: https://github.com/QwenLM/Qwen-Agent
--------------------------------------------------
name: LazyVim/LazyVim
description: Neovim config for the lazy
language: Lua
stars: 20740
forks: 0
url: https://github.com/LazyVim/LazyVim
--------------------------------------------------
name: mlabonne/llm-course
description: Course to get into Large Language Models (LLMs) with roadmaps and Co
lab notebooks.
language: Jupyter Notebook
stars: 50594
forks: 0
url: https://github.com/mlabonne/llm-course
--------------------------------------------------
name: NVIDIA/NeMo
description: A scalable generative AI framework built for researchers and develop
ers working on Large Language Models, Multimodal, and Speech AI (Automatic Speech
Recognition and Text-to-Speech)
language: Python
stars: 14326
forks: 0
url: https://github.com/NVIDIA/NeMo
--------------------------------------------------
name: gosom/google-maps-scraper
description: scrape data data from Google Maps. Extracts data such as the name, a
ddress, phone number, website URL, rating, reviews number, latitude and longitud
e, reviews,email and more for each place
language: Go
stars: 1948
forks: 0
url: https://github.com/gosom/google-maps-scraper
--------------------------------------------------
name: ranaroussi/yfinance
description: Download market data from Yahoo! Finance's API
language: Python
stars: 17412
forks: 0
url: https://github.com/ranaroussi/yfinance
--------------------------------------------------
name: Atmosphere-NX/Atmosphere
description: Atmosphère is a work-in-progress customized firmware for the Nintend
o Switch.
```

```
language: C++
stars: 16220
forks: 0
url: https://github.com/Atmosphere-NX/Atmosphere
--------------------------------------------------
name: xming521/WeClone
description: 🚀从聊天记录创造数字分身的一站式解决方案💡 使用微信聊天记录微调大语言模
型，让大模型有“那味儿”，并绑定到聊天机器人，实现自己的数字分身。 数字克隆/数字分身/数字
永生/声音克隆/LLM/大语言模型/微信聊天机器人/LoRA
language: Python
stars: 5415
forks: 0
url: https://github.com/xming521/WeClone
--------------------------------------------------
name: Blaizzy/mlx-audio
description: A text-to-speech (TTS), speech-to-text (STT) and speech-to-speech (S
TS) library built on Apple's MLX framework, providing efficient speech analysis o
n Apple Silicon.
language: Python
stars: 2054
forks: 0
url: https://github.com/Blaizzy/mlx-audio
--------------------------------------------------
name: 521xueweihan/HelloGitHub
description: 分享 GitHub 上有趣、入门级的开源项目。Share interesting, entry-level o
pen source projects on GitHub.
language: Python
stars: 109192
forks: 0
url: https://github.com/521xueweihan/HelloGitHub
--------------------------------------------------
name: Lightricks/ComfyUI-LTXVideo
description: LTX-Video Support for ComfyUI
language: Python
stars: 1713
forks: 0
url: https://github.com/Lightricks/ComfyUI-LTXVideo
```

Store Data in a Structured Format

Give a brief overview of the data collected (e.g. count, fields, ...)

In [48]:
```python
# TODO
# Anzahl gesammelter Repositories
print(f"Collected {len(repositories)} repositories.")

print("Each repository contains the following fields:")
print("- name: Repository full name (e.g., 'owner/repo')")
print("- description: Short description of the repository")
print("- language: Main programming language used")
print("- stars: Total number of stars")
print("- forks: Total number of forks")
print("- url: Direct link to the GitHub repo")
```

```
Collected 20 repositories.
Each repository contains the following fields:
- name: Repository full name (e.g., 'owner/repo')
- description: Short description of the repository
- language: Main programming language used
- stars: Total number of stars
- forks: Total number of forks
- url: Direct link to the GitHub repo
```

Save the Data

In [49]:

```
# TODO

try:
    with open("ScrapedData.json", "w", encoding="utf-8") as f:
        json.dump(all_repo_data, f, ensure_ascii=False, indent=2)
    print("Data stored successfully under ScrapedData.json")
except Exception as e:
    print(f"Failed to store data: {e}")
```

```
Data stored successfully under ScrapedData.json
```