ScrapeGraphAI

You Only Scrape Once

Our Team

Marco Vinciguerra

MSc Computer Engineering

mvincig11@gmail.com

Marco Perini

MSc Mechatronics Engineering

perinim.98@gmail.com

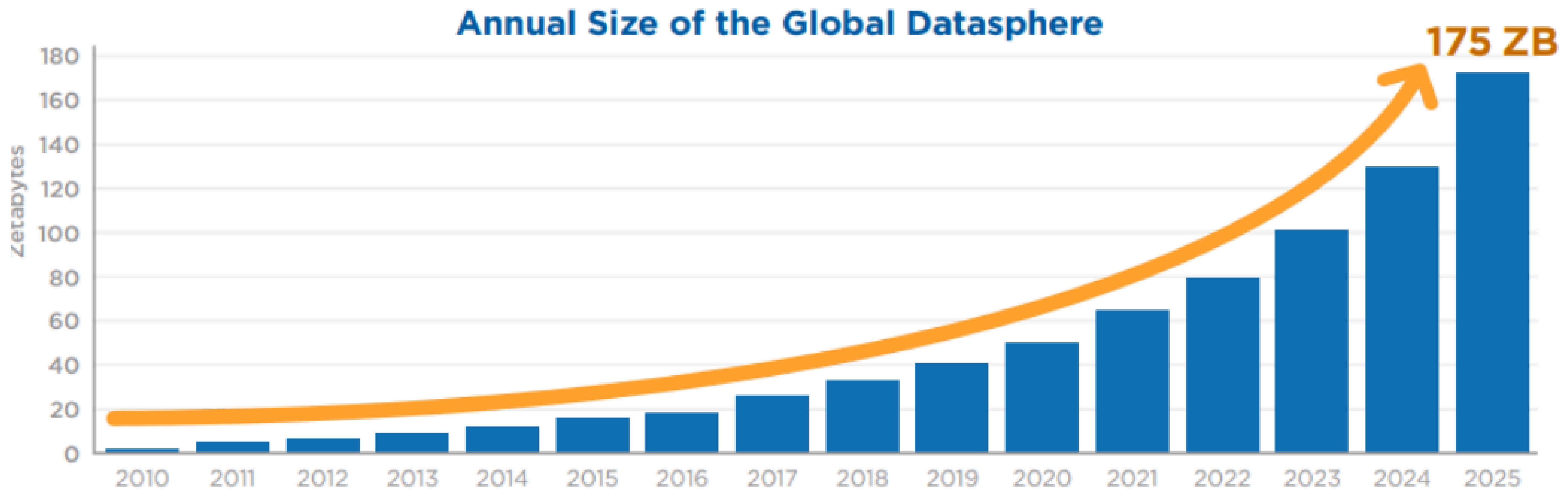
Lorenzo Padoan

lorenzo.padoan977@gmail.com

The era of Big Data

The era of big data

We live in a world that produces Zettabytes of data



Source: International Data Corporation (IDC)

The era of Big Data

We live in a **data-hungry** world

Analytics

Gemini

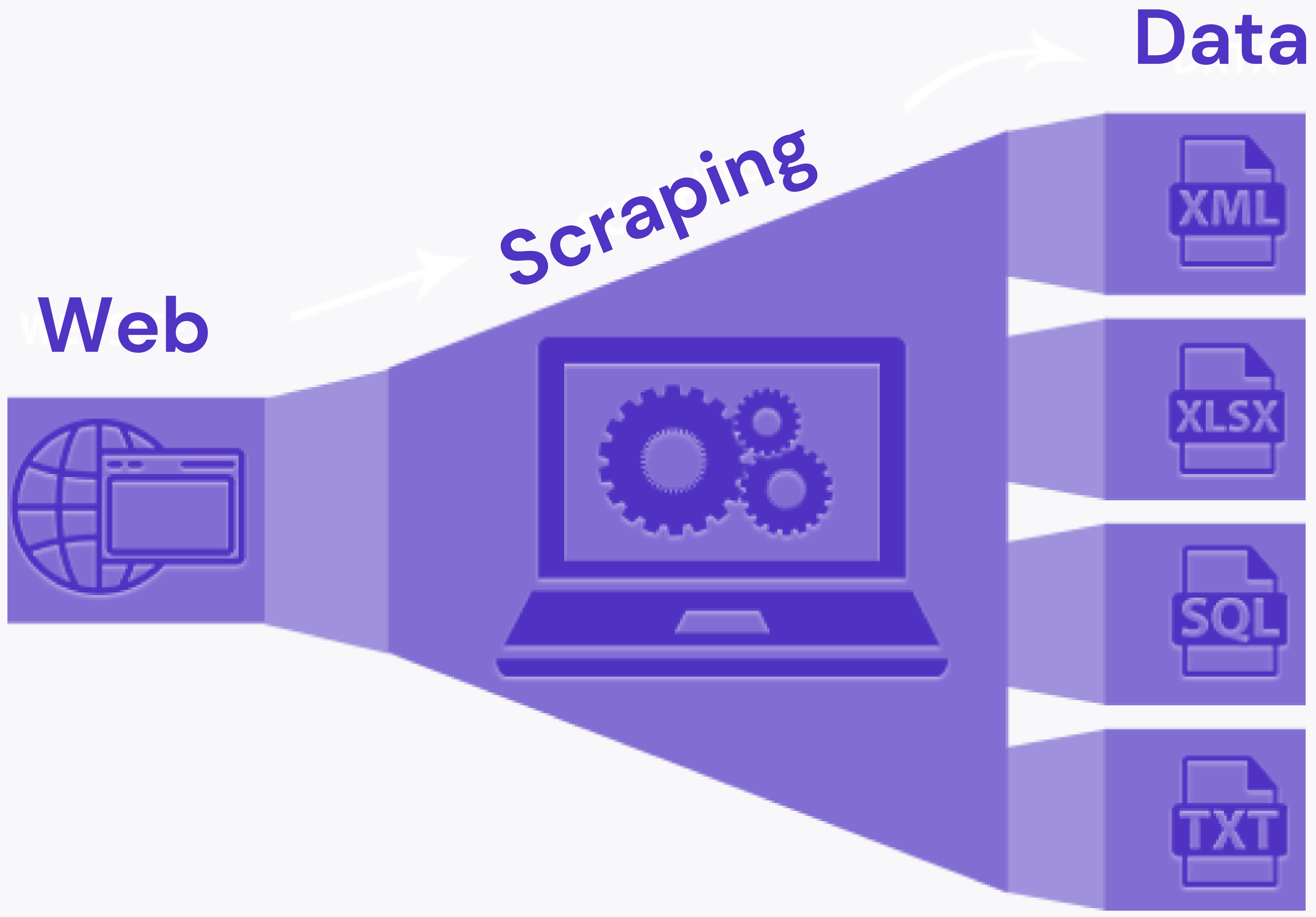
ChatGPT

Training of LLMs

Internet

Principal source of data

What is a scraper



Scraping is the act of extracting information from a data source and create structured data from unstructered ones

Common scraping tools

Dev tools

BeautifulSoup

Scrapy

Web services

Web Scraper

Octoparse

Our Question

Is it possible to **scrape** websites **without** any knowledge of **HTML**, just by **writing what** I want and **how** we want it?

Our Solution

Yes 🎉🎉🎉

with

Scrapegraph-ai

What is this spider? 🤔

ScrapeGraphAI

Python Library

Open Source

Scraping powered by LLM

Highly modular

Avaiable models

Ollama

Gemini

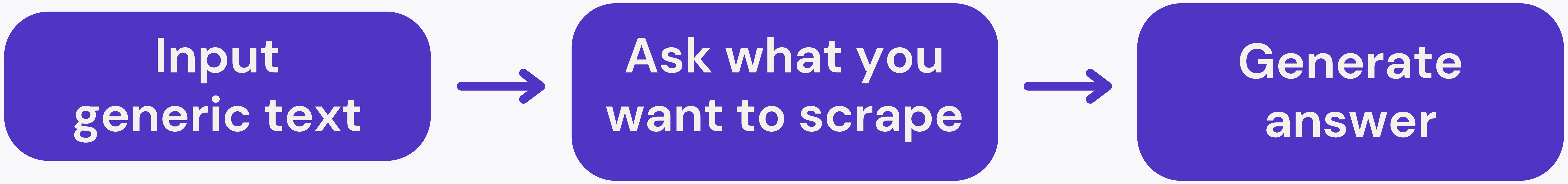
Nodes

A node is the basic component of the Scrapegraph library that allows to do a specific action

Graphs

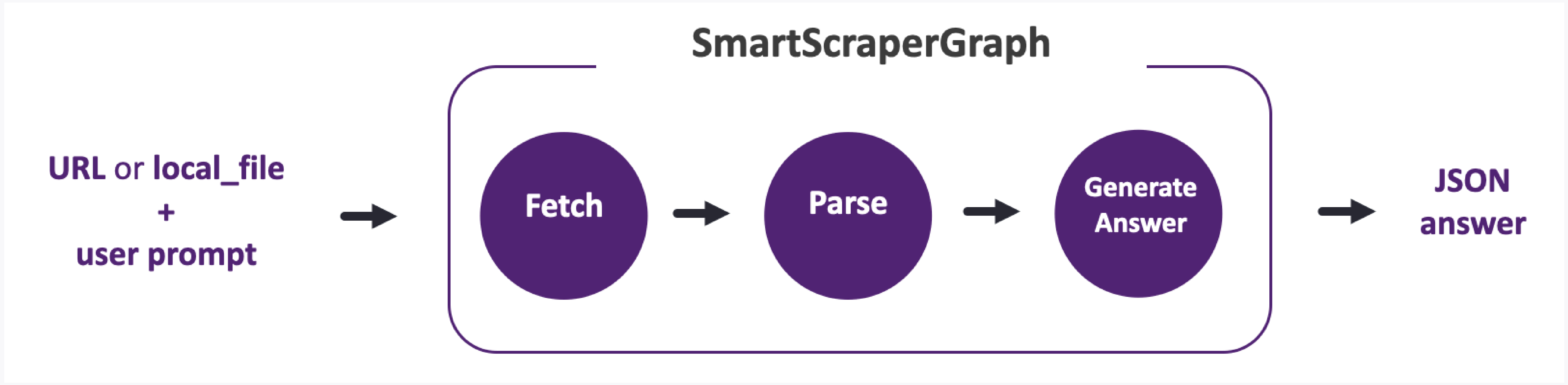
A graph is a concatenation of single nodes

Scraping from text



It is possible to insert various type of text,
Classic string, downloaded HTML code, XML etc...

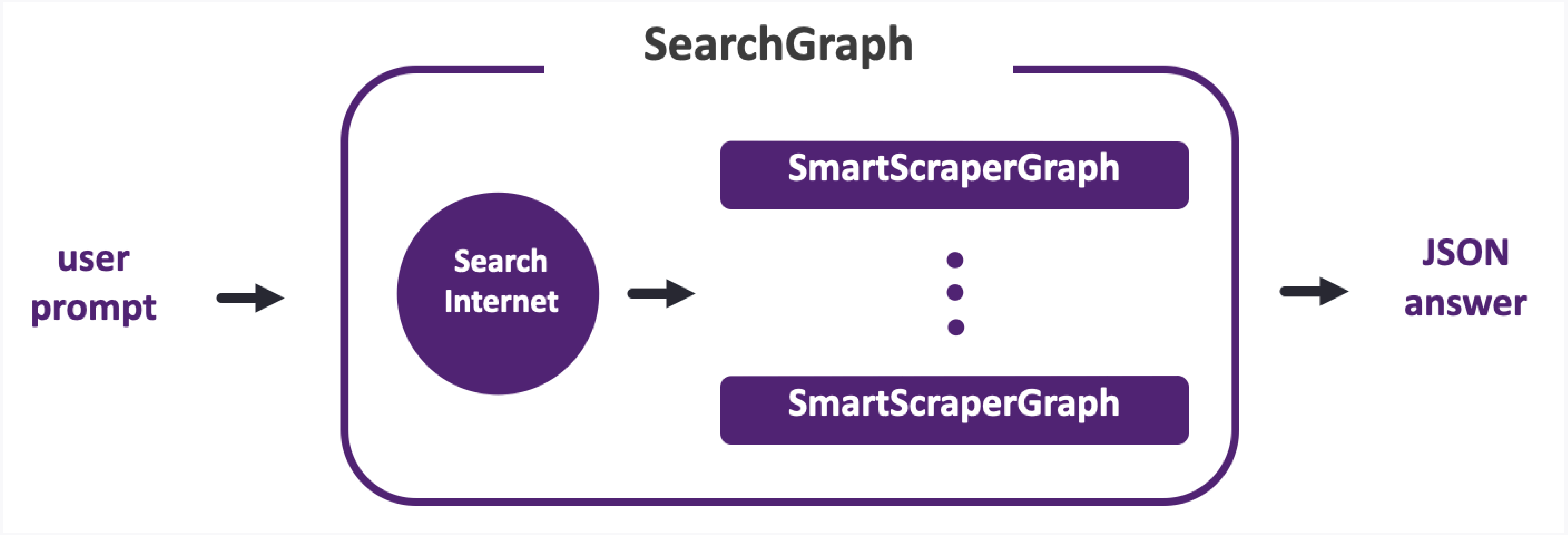
SmartScraper



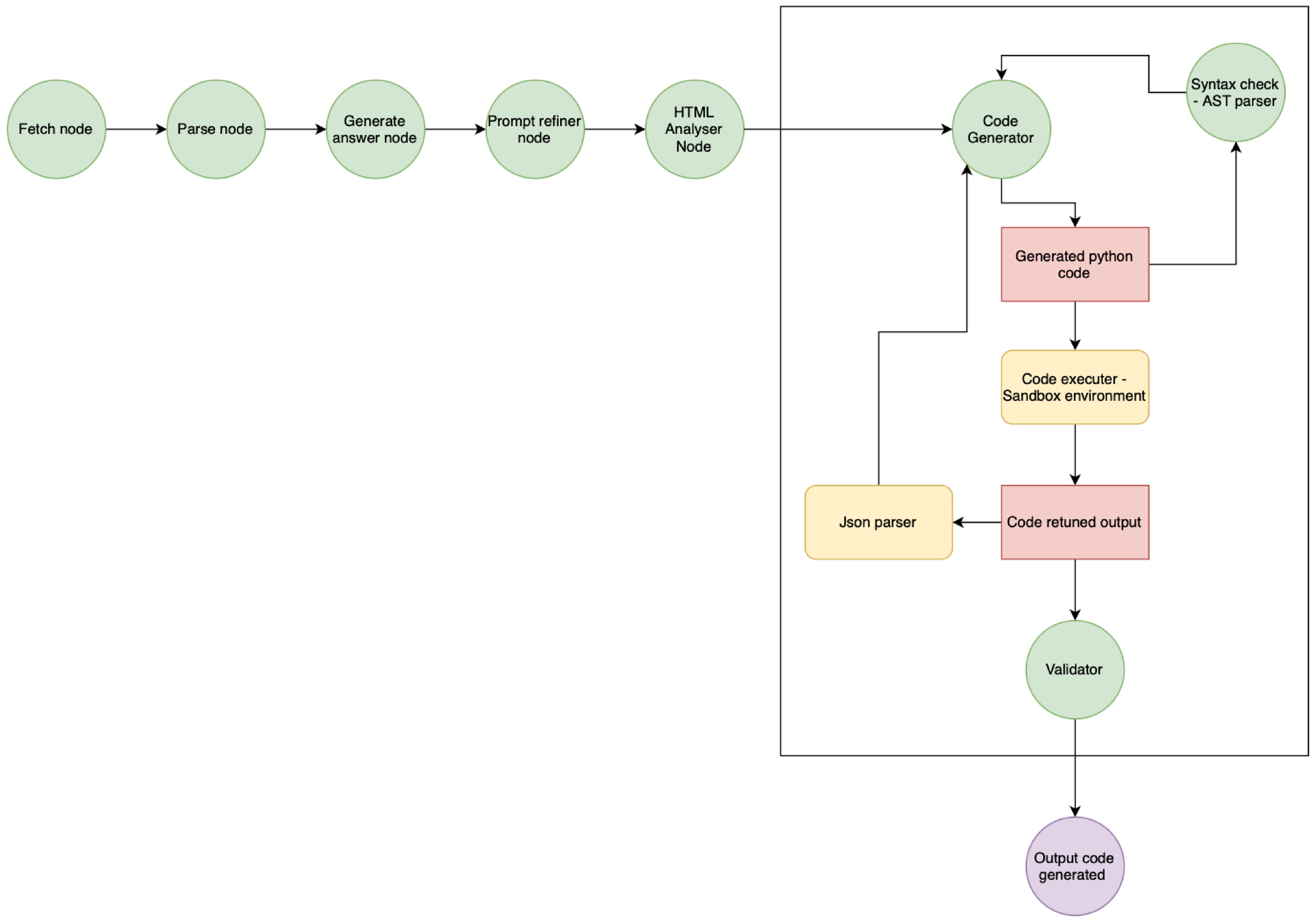
Components:

- **Fetch node:** fetchs the code through HTTP request
- **Parse node:** parse the HTML code in documents
- **Generate Answer node:** calls the LLM for extracting the answer

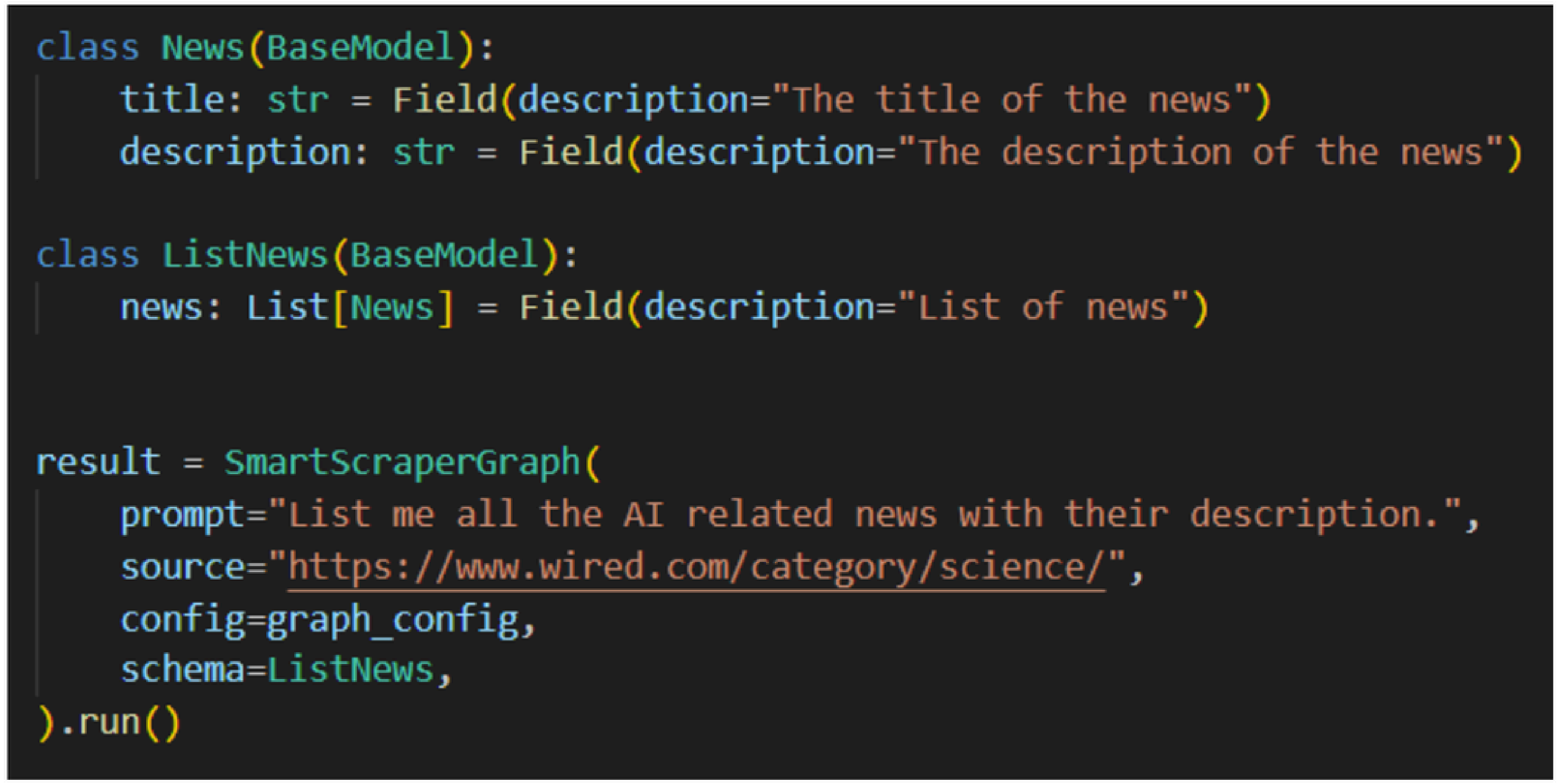
SearchGraph



CodeGeneratorGraph



Pydantic Schema



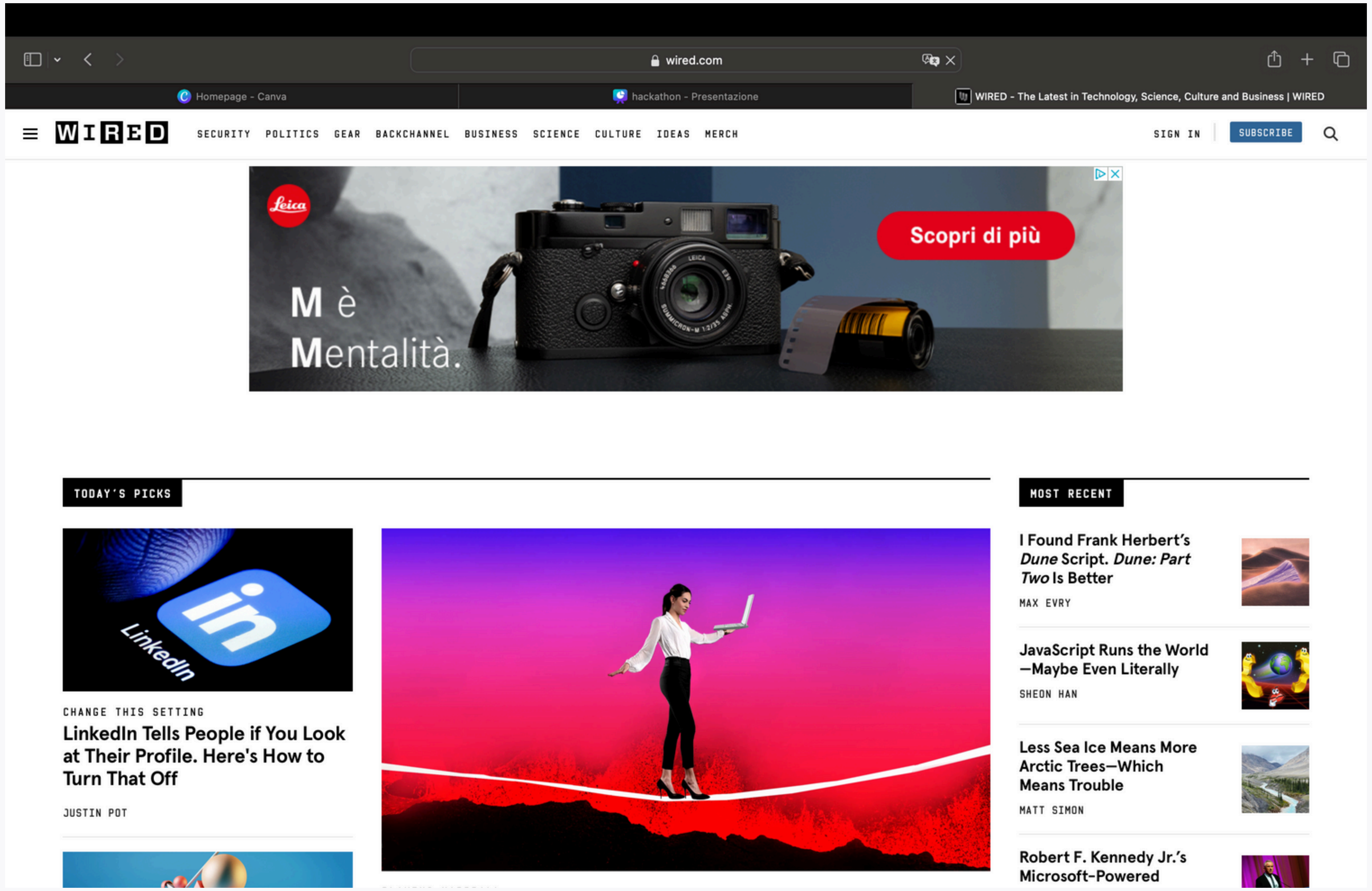
```python
class News(BaseModel):
    title: str = Field(description="The title of the news")
    description: str = Field(description="The description of the news")

class ListNews(BaseModel):
    news: List[News] = Field(description="List of news")


result = SmartScraperGraph(
    prompt="List me all the AI related news with their description.",
    source="https://www.wired.com/category/science/",
    config=graph_config,
    schema=ListNews,
).run()
```

```
{
  "news": [
    {
      "title": "Light-Based Chips ...",
      "description": "Optical neural networks ..."
    },
    ...
    {
      "title": "AI Is Building ...",
      "description": "Robots, computers, and ..."
    }
  ]
}
```

Comparative results

Let's suppose we want to extract the news titles from

https://www.wired.com



wired.com

Homepage - Canva

hackathon - Presentazione

WIRED - The Latest in Technology, Science, Culture and Business | WIRED

WIRED

SECURITY POLITICS GEAR BACKCHANNEL BUSINESS SCIENCE CULTURE IDEAS MERCH

SIGN IN SUBSCRIBE

Leica

M è Mentalità.

Scopri di più

TODAY'S PICKS

CHANGE THIS SETTING

LinkedIn Tells People if You Look at Their Profile. Here's How to Turn That Off

JUSTIN POT

MOST RECENT

I Found Frank Herbert's *Dune* Script. *Dune: Part Two* Is Better

MAX EVRY

JavaScript Runs the World—Maybe Even Literally

SHEON HAN

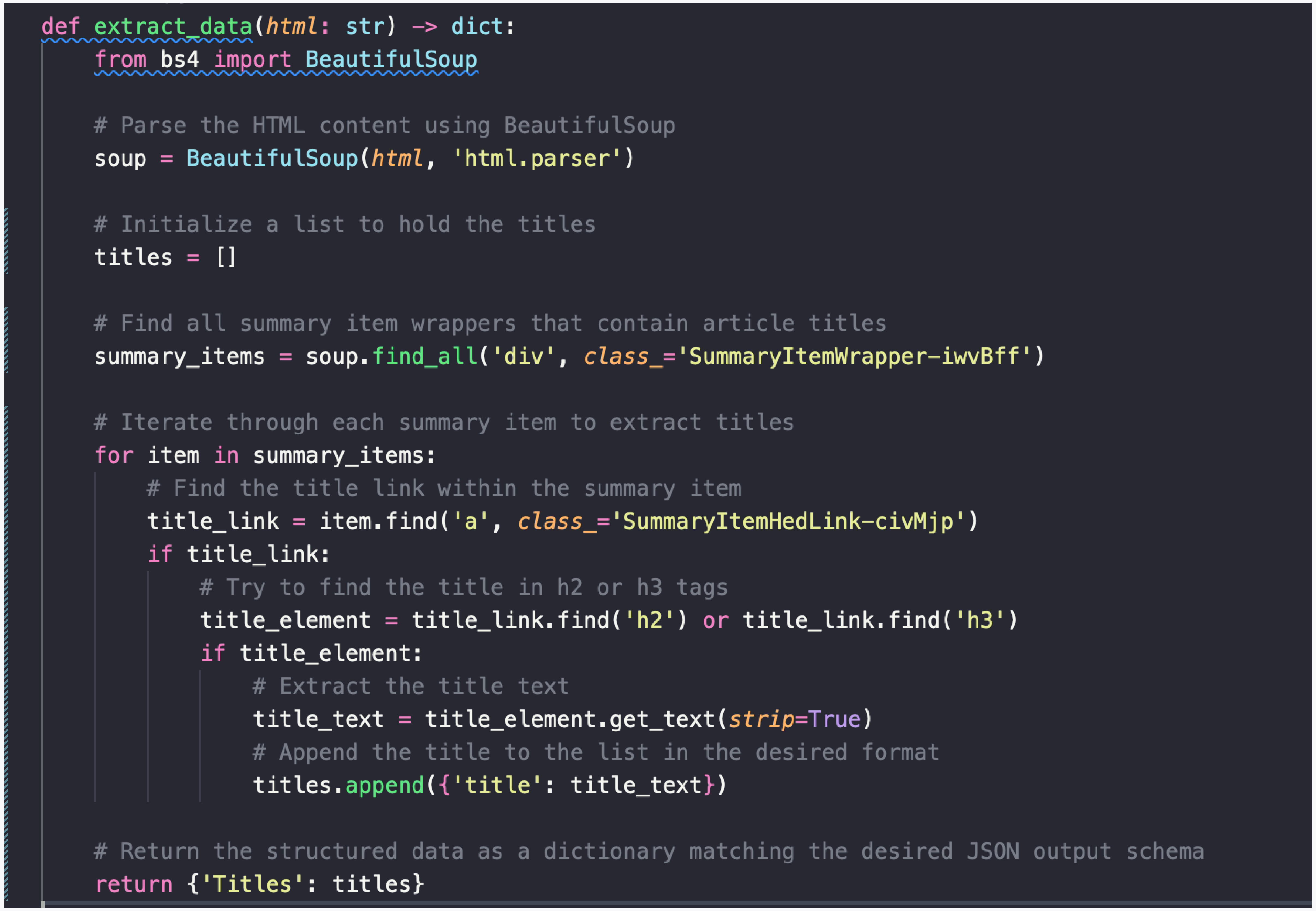
Less Sea Ice Means More Arctic Trees—Which Means Trouble

MATT SIMON

Robert F. Kennedy Jr.'s Microsoft-Powered

Comparative results

BeautifulSoup 🤮

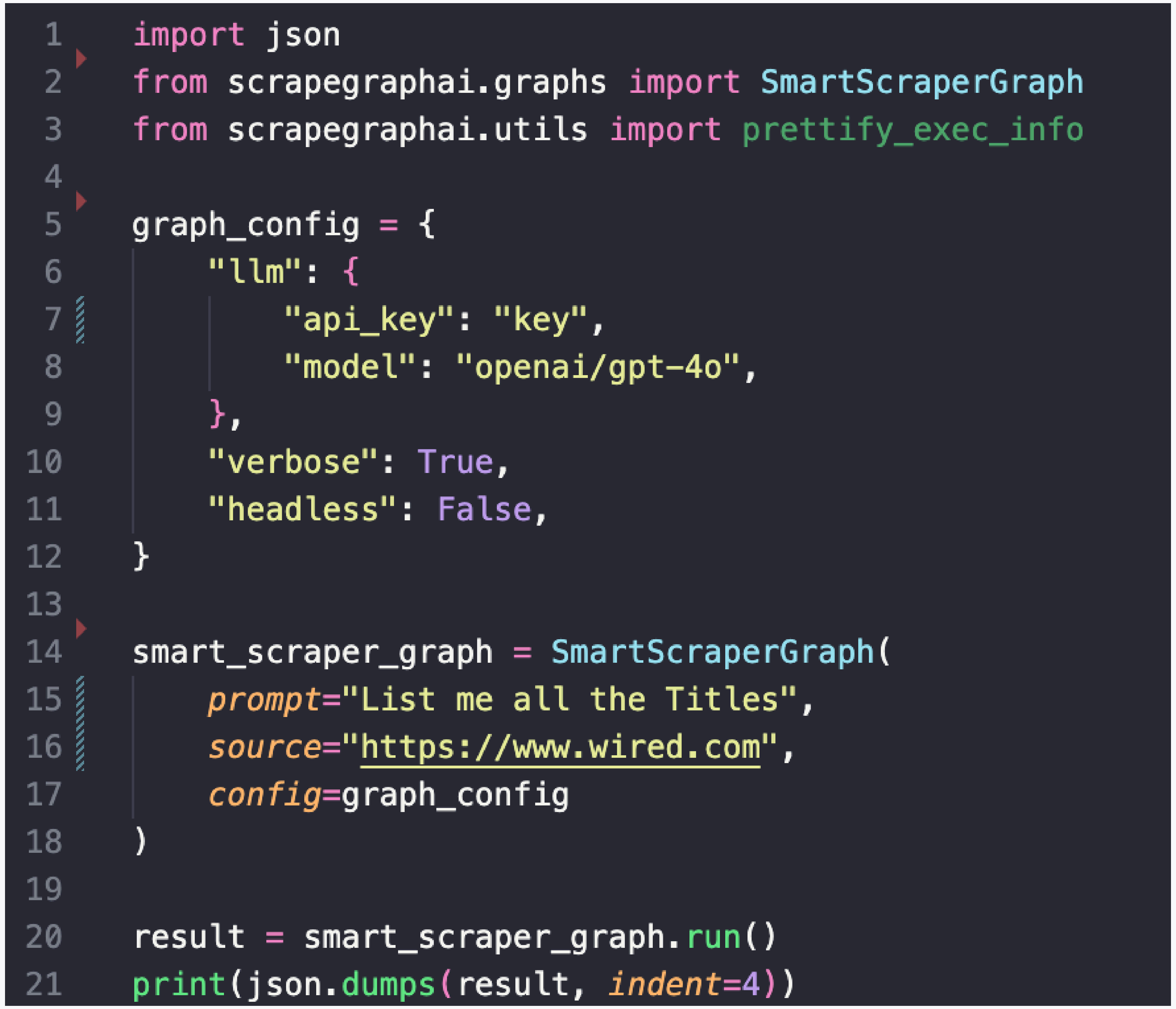


```python
def extract_data(html: str) -> dict:
    from bs4 import BeautifulSoup

    # Parse the HTML content using BeautifulSoup
    soup = BeautifulSoup(html, 'html.parser')

    # Initialize a list to hold the titles
    titles = []

    # Find all summary item wrappers that contain article titles
    summary_items = soup.find_all('div', class_='SummaryItemWrapper-iwvBff')

    # Iterate through each summary item to extract titles
    for item in summary_items:
        # Find the title link within the summary item
        title_link = item.find('a', class_='SummaryItemHedLink-civMjp')
        if title_link:
            # Try to find the title in h2 or h3 tags
            title_element = title_link.find('h2') or title_link.find('h3')
            if title_element:
                # Extract the title text
                title_text = title_element.get_text(strip=True)
                # Append the title to the list in the desired format
                titles.append({'title': title_text})

    # Return the structured data as a dictionary matching the desired JSON output schema
    return {'Titles': titles}
```

Comparative results

ScrapeGraphAI 🤩

More concise, less code, reusability.
For scraping another website
you just have to change
2 lines!!!!



```python
import json
from scrapegraphai.graphs import SmartScraperGraph
from scrapegraphai.utils import prettify_exec_info

graph_config = {
    "llm": {
        "api_key": "key",
        "model": "openai/gpt-4o",
    },
    "verbose": True,
    "headless": False,
}

smart_scraper_graph = SmartScraperGraph(
    prompt="List me all the Titles",
    source="https://www.wired.com",
    config=graph_config
)

result = smart_scraper_graph.run()
print(json.dumps(result, indent=4))
```

DEMO TIME!!

Pros of Scrapegraph-ai

- Low code and fast implementation
- Fast learning curve
- Fault tollerance due to dinamic HTML code
- Possibility to run local LLM
- Portability and no knowledge of HTML needed
- No possibles data leaks if you run local LLM

😍

Some numbers

After 8 months of development has:

- 15000+ stars on Github
- 1200+ forks
- 290k + downloads on pypi (pip)
- 2M + usage
- 700+ users on Discord

Repositories

ScrapeGraphAI

https://github.com/ScrapeGraphAI/Scrapegraph-ai

If you like the project feel free to leave a star ⭐

Website

ScrapeGraphAI's website

https://scrapegraphai.com

Take a look also to the web service!

Thank you for the attention

ScrapeGraphAI

You Only Scrape Once