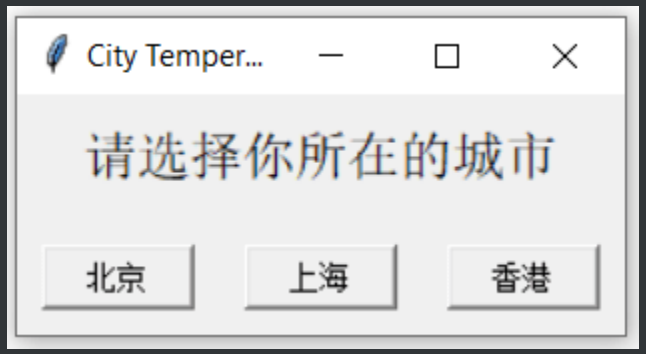
API

Demonstration and Vote!

This will be included as part of your in-class participation.

I have developed an APP. Let's try it.



In your opinion, how does it get the temperature data?

https://www.youtube.com/embed/s7wmiS2mSXY?enablejsapi=1

Let me show you the basics.

```python
import requests
url = "http://apis.juhe.cn/simpleWeather/query"
params = {
        "city": "北京",
        "key": "your API key",
    }
response = requests.get(url, params = params)
data = response.json()
temperature = data['result']['realtime']['temperature']
print(temperature)
```

You need to apply for your API key (password)

I am not sharing my key with you because a free key can only be used 50 times a day...

How to get the key?

1. Sign up an account here and activate it.
2. Sign up for an API key.

Exercise

Apply for your own API key and develop your own weather APP!

```python
import tkinter as tk
import requests
def get_temperature(city):
    url = "http://apis.juhe.cn/simpleWeather/query"
    params = { "city": city, "key": "your key"}
    response = requests.get(url, params=params)
    data = response.json()
    temperature = data['result']['realtime']['temperature']
    spaces_to_add = 3 - len(temperature)
    output_string = " " * spaces_to_add + temperature
    return output_string
def update_display(city):
    temperature = get_temperature(city)
    display_label.config(text="{}当前温度为: {}摄氏度".format(city, temperature))
root = tk.Tk()
root.title("City Temperature App")
display_label = tk.Label(root, text="    请选择你所在的城市    ", font=("Arial", 16))
display_label.grid(row=0, column=0, columnspan=3, pady=10)
cities = ["北京", "上海", "香港"]
for index, city in enumerate(cities):
    city_button = tk.Button(root, text=city, command=lambda c=city: update_display(c))
    city_button.grid(row=1, column=index, padx=10, pady=10, sticky="ew")
for i in range(3):
    root.grid_columnconfigure(i, weight=1)

root.grid_rowconfigure(1, weight=1)
root.mainloop()
```

Let’s try the following API for Chinese/English translation

Translation

```python
import requests
url = "https://api.oick.cn/api/fanyi"
params = {
    "text": "欢迎来到香港大学",
    }
response = requests.get(url, params = params)
data = response.json()
result = data['data']['result']
print(result)
```

Exercise: Design an APP so that users can input Chinese/English and the APP translates it into the other language

My APP created by GPT

```python
import tkinter as tk
import requests
def translate(text):
    url = "https://api.oick.cn/api/fanyi"
    params = {"text": text }
    response = requests.get(url, params = params)
    data = response.json()
    result = data['data']['result']
    return result

def translate_text():
    chinese_text = input_text.get("1.0", "end-1c")  # Get text from input text box
    english_text = translate(chinese_text)
    input_text.delete("1.0", "end")  # Clear the input text
    input_text.insert("1.0", english_text)  # Display translated text in the same text box

root = tk.Tk()
root.title("Chinese to English Translator")
input_text = tk.Text(root, height=10, width=50)
input_text.pack(pady=10)
translate_button = tk.Button(root, text="Translate", command=translate_text)
translate_button.pack(pady=10)
root.mainloop()
```

AviationStack is a platform which provides an API to track the status of flights. Its website is here:

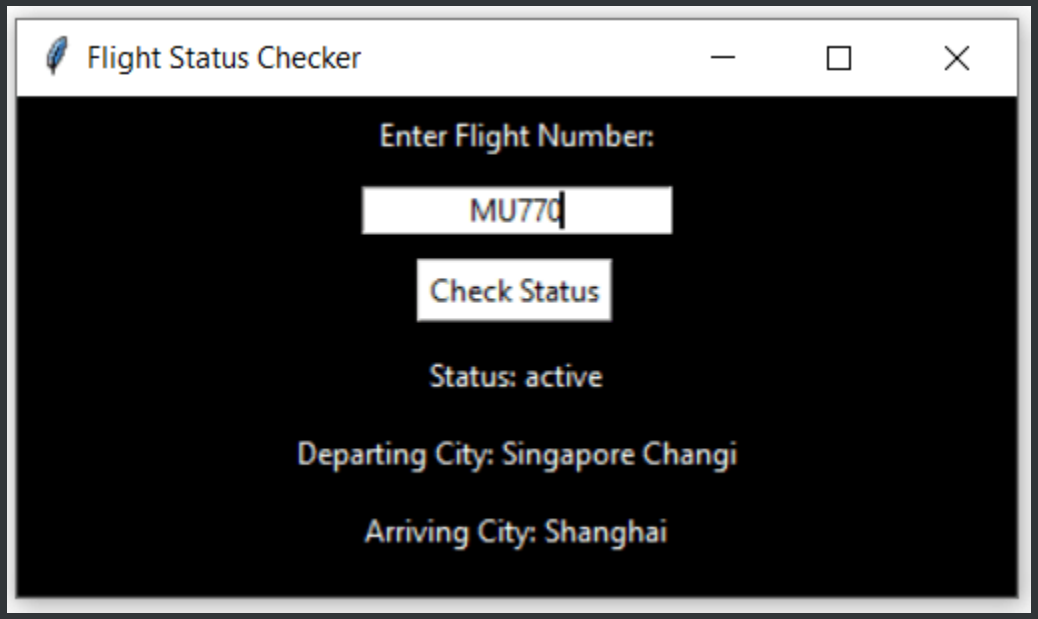
https://aviationstack.com/

You can find its API documentation here.

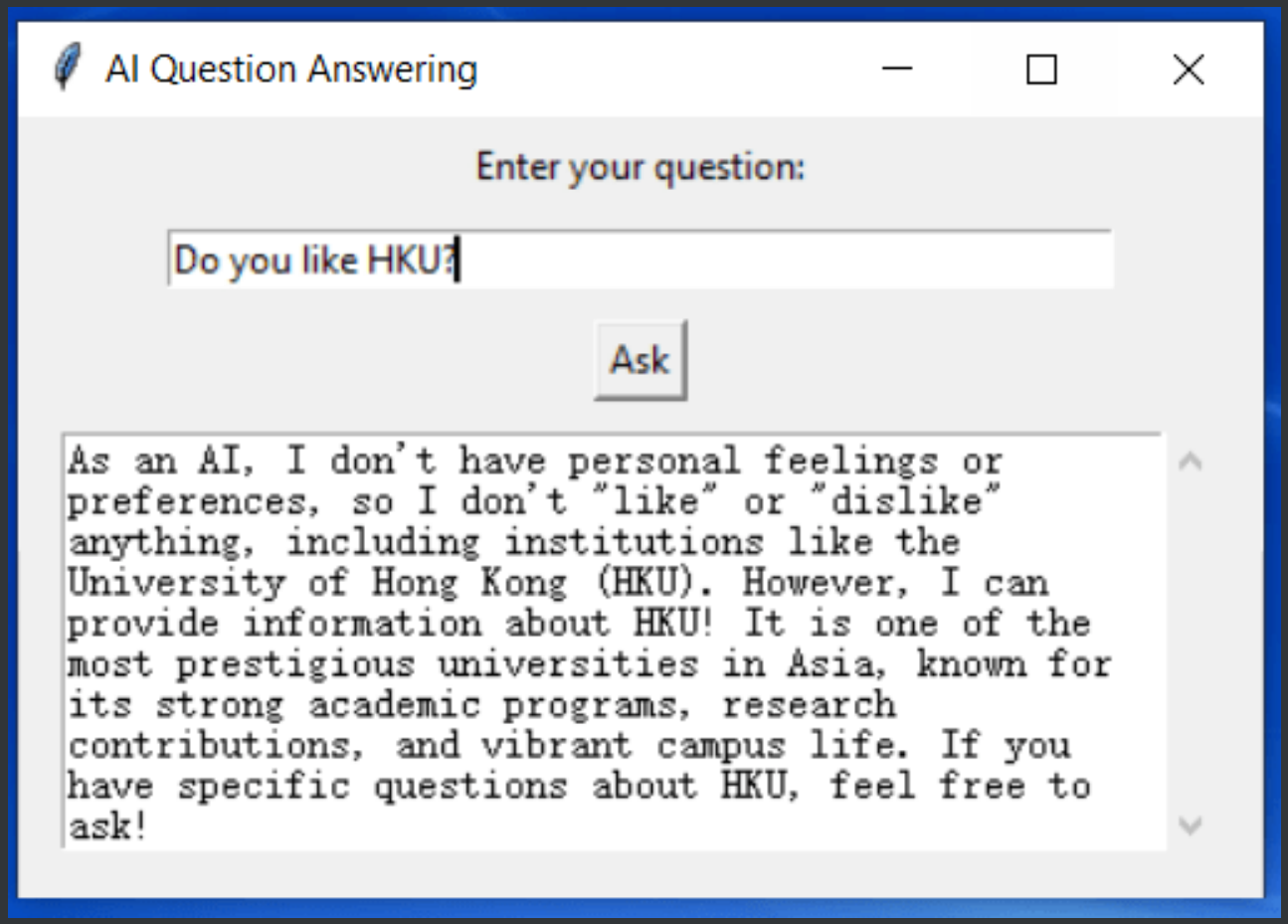
Exercise

- Acquire a free API Key from the platform
- Develop an APP, the APP takes flight number as input from the user, and displays its fight status, departing airport and arriving airport on the screen.
- Try to do everything yourself (with help from GPT)!

This is my APP.

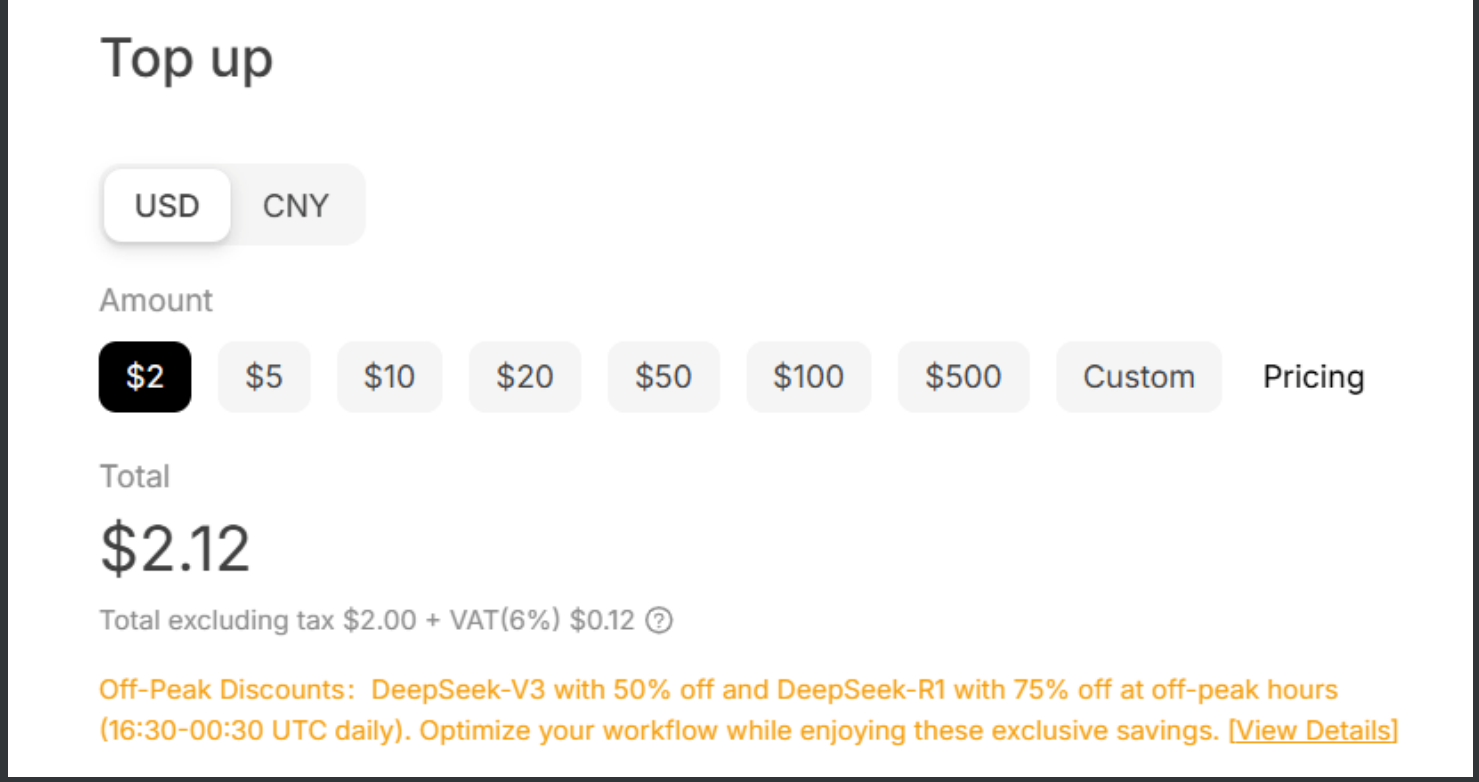


This is “my” AI (API)



Exercise

- Acquire an API Key from DeepSeek (https://platform.deepseek.com/)
- This is not free, but very cheap. You need to top up $2 or RMB 10 to use the API.
- Then, develop an APP which takes a question from user, sends the question to the API, and displays the answer from API to the user.



API keys

Your API keys are listed below. The API key is only visible and can be copied once at creation. Save it securely. Do not share your API key with others, or expose it in the browser or other client-side code. In order to protect the security of your account, DeepSeek may also automatically disable any API key that we've found has leaked publicly. Usage of API keys created before April 25, 2024, was not tracked.

Name	Key	Created	Last used
ximarketing	sk-16926**********2e3e	2025-02-26	2025-02-26

If you need help on how to use the API, check the document here: https://api-docs.deepseek.com/

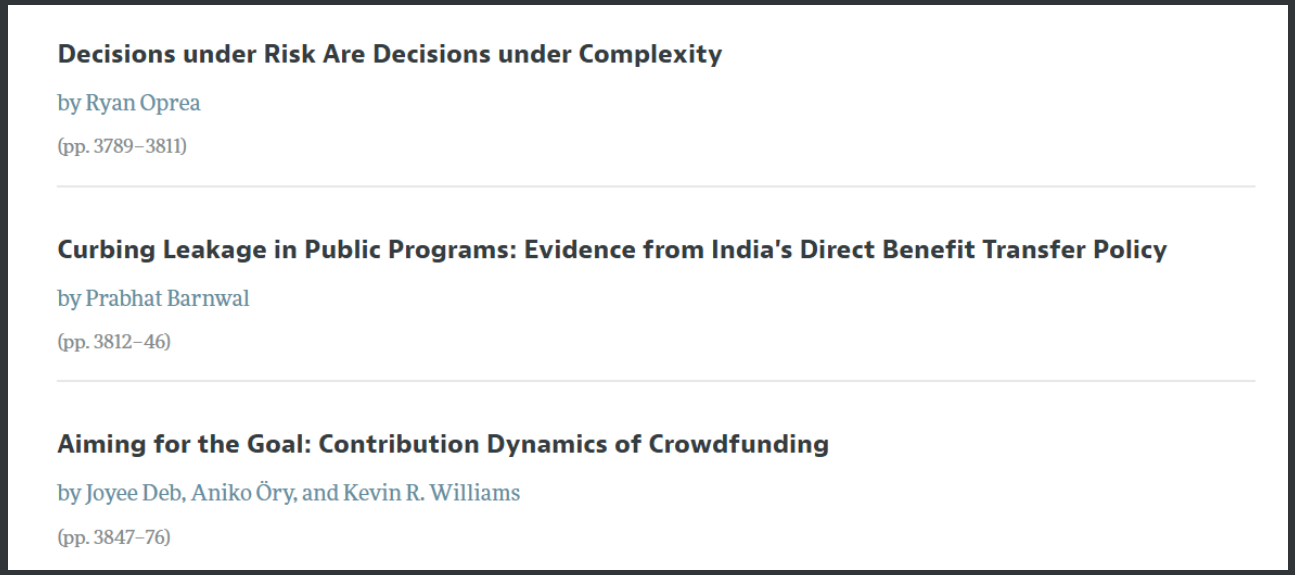
What are the main topics in modern economics?

Scraping made Simple

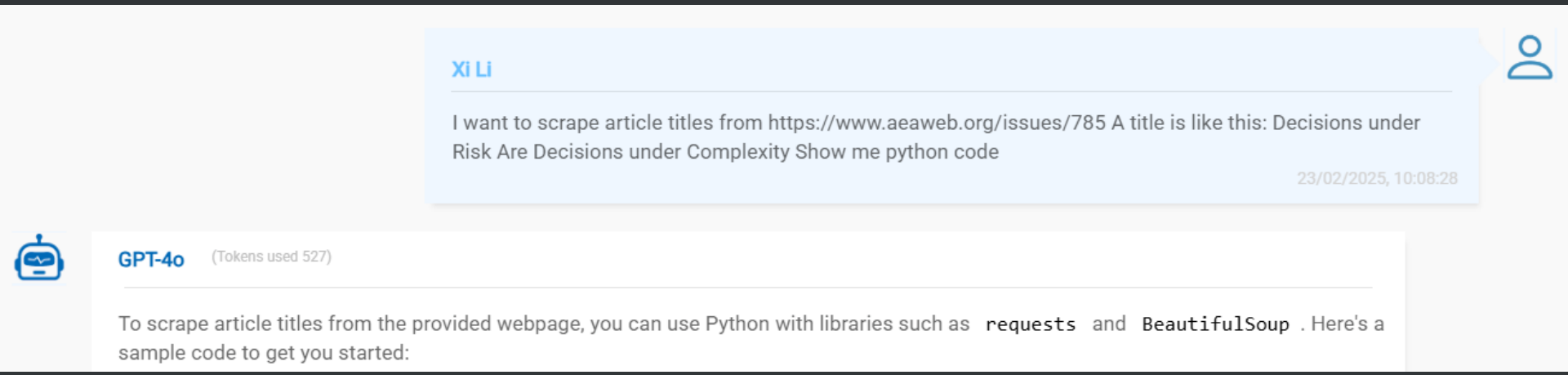
Scraping

Consider a simple URL:
https://www.aeaweb.org/issues/785

American Economic Review is an academic journal in economics (we don't care anyway). Let's try to write a scraper to download the article titles from the page:



Scraping


Xi Li
I want to scrape article titles from https://www.aeaweb.org/issues/785 A title is like this: Decisions under Risk Are Decisions under Complexity Show me python code
23/02/2025, 10:08:28
GPT-4o
(Tokens used 527)
To scrape article titles from the provided webpage, you can use Python with libraries such as requests and BeautifulSoup . Here's a sample code to get you started:

The code is a bit problematic...

```
import requests
from bs4 import BeautifulSoup
# URL of the page to scrape
url = "https://www.aeaweb.org/issues/785"
# Send a GET request to fetch the page content
response = requests.get(url)
# Check if the request was successful (status code 200)
if response.status_code == 200:
    # Parse the page content with BeautifulSoup
    soup = BeautifulSoup(response.content, 'html.parser')

    # Find all article titles
    # Note: The exact HTML structure of the page needs to be inspected to find the correct tags and classes
    titles = soup.find_all('h2', class_='title')  # Example: Change these tags and classes based on actual HTML structure

    # Print the titles
    for title in titles:
        print(title.get_text().strip())
else:
    print(f"Failed to retrieve the page. Status code: {response.status_code}")
```

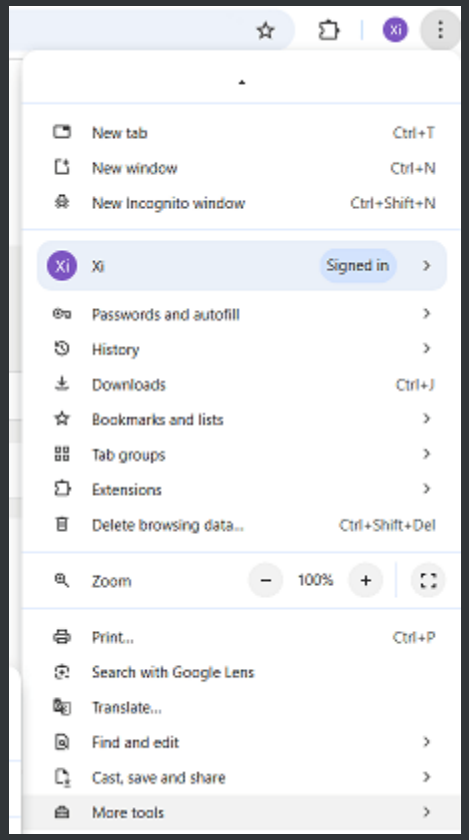
What's wrong with it?

```
import requests
from bs4 import BeautifulSoup
# URL of the page to scrape
url = "https://www.aeaweb.org/issues/785"
# Send a GET request to fetch the page content
response = requests.get(url)
# Check if the request was successful (status code 200)
if response.status_code == 200:
    # Parse the page content with BeautifulSoup
    soup = BeautifulSoup(response.content, 'html.parser')

    # Find all article titles
    # Note: The exact HTML structure of the page needs to be inspected to find the correct tags and classes
    titles = soup.find_all('h2', class_='title')  # Example: Change these tags and classes based on actual HTML structure

    # Print the titles
    for title in titles:
        print(title.get_text().strip())
else:
    print(f"Failed to retrieve the page. Status code: {response.status_code}")
```

Check HTML with Chrome

Open the webpage in your Chrome browser.

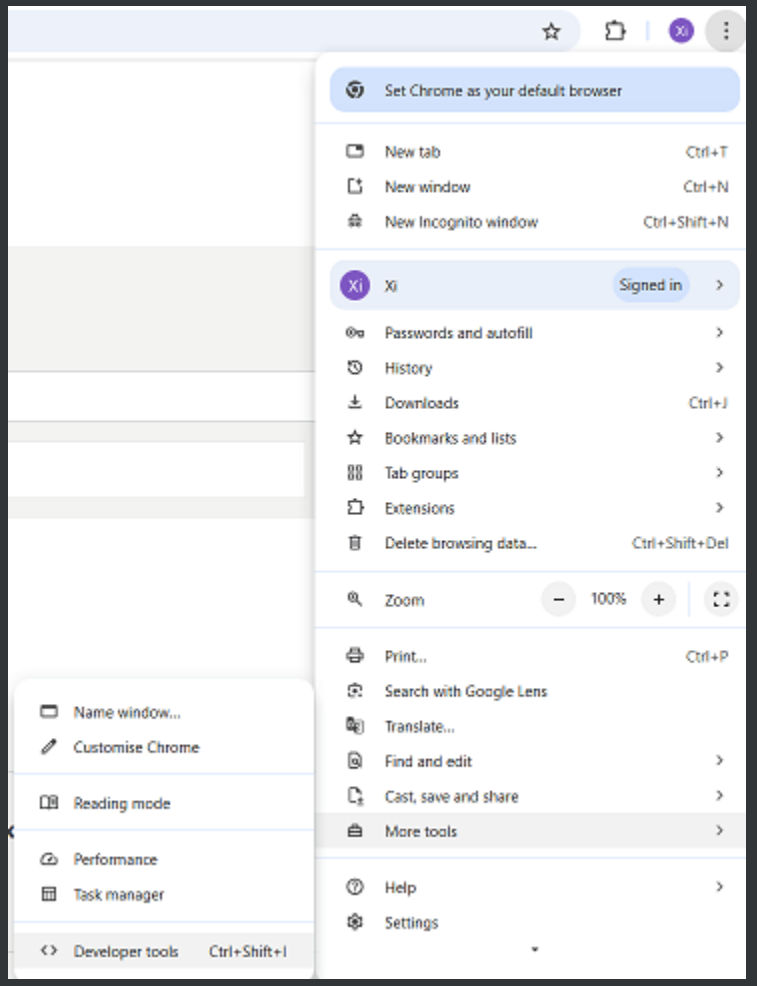
Click the upper right Chome setting button of your browser and you will be directed here.



Check HTML with Chrome

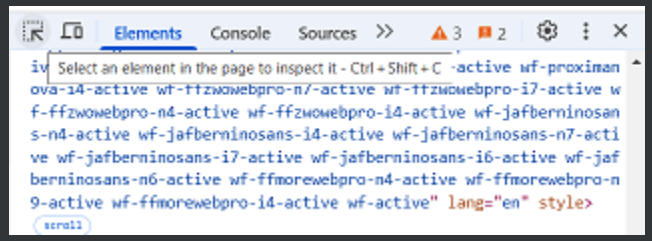
Choose “More tools”…

Choose “Developer tools”…



Check HTML with Chrome

Click the button and you will get to “select an element in the page to inspect it”.



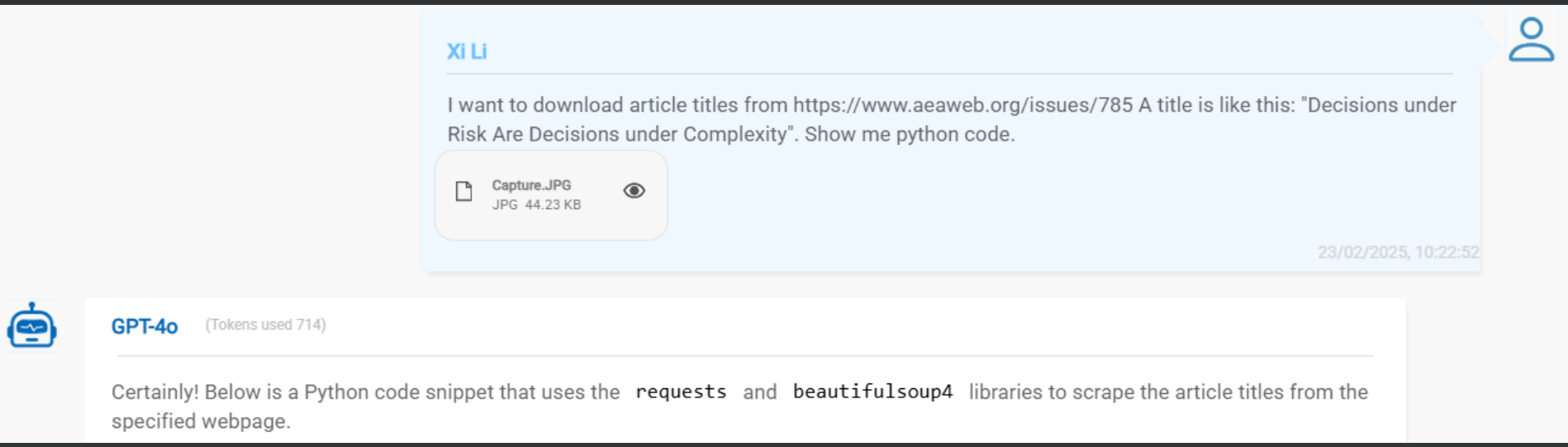
Alternatively, use “Ctrl + Shift + C.”

If needed, also click the button to switch between desktop and mobile version.

Check HTML with Chrome

The title information is here.

Upload your screenshot to ChatGPT!



The code I got

```
import requests
from bs4 import BeautifulSoup

# URL of the page to scrape
url = 'https://www.aeaweb.org/issues/785'

# Send a GET request to the page
response = requests.get(url)

# Parse the page content with BeautifulSoup
soup = BeautifulSoup(response.content,
'html.parser')

# Find all article titles
titles = soup.find_all('h3', class_='title')

# Print each title text
for title in titles:
    print(title.get_text(strip=True))
```

Scraping

Consider a simple URL:
https://www.aeaweb.org/issues/785

Try to download the links of the articles now, with the help of GPT. Here is an example of the link:

https://www.aeaweb.org/articles?id=10.1257/aer.20221227

Scraping

Xi Li

I want to scrape article links from this page: https://www.aeaweb.org/issues/785. A link is like this: https://www.aeaweb.org/articles?id=10.1257/aer.20221227 Show me python code

Capture2.JPG
JPG 35.32 KB

23/02/2025, 17:21:07

GPT-4o (Tokens used 764)

```
<article class="journal-article symposia-title" style="border:none;"> Articles
</article>
▼<article class="journal-article  art_18081" id="10.1257/aer.20221227">
  ▼<h3 class="title">
      <a href="/articles?id=10.1257/aer.20221227">Decisions under Risk Are
      Decisions under Complexity</a> == $0
    </h3>
  ▶<div class="article-item-authors">…</div>
    <div class="page-range">(pp. 3789-3811)</div>
```

My Code

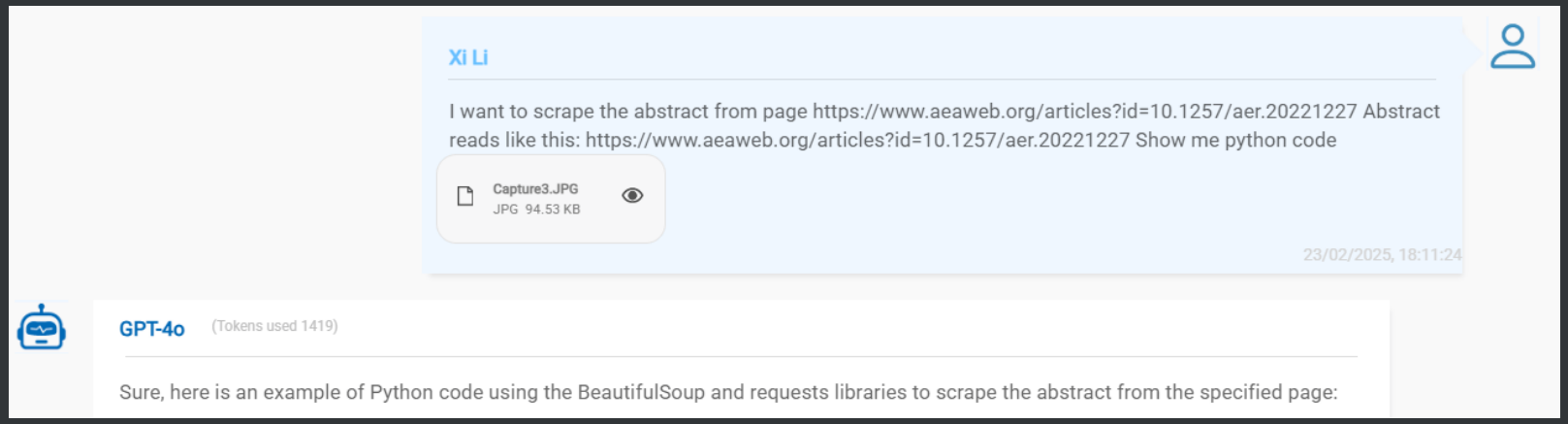
```
import requests
from bs4 import BeautifulSoup
# URL of the page to scrape
url = 'https://www.aeaweb.org/issues/785'
# Send a GET request to the page
response = requests.get(url)
response.raise_for_status()  # Check that the request was
successful
# Parse the page content with BeautifulSoup
soup = BeautifulSoup(response.content, 'html.parser')
# Find all article links
article_links = []
for a_tag in soup.find_all('a', href=True):
    href = a_tag['href']
    if href.startswith('/articles?id='):
        full_url = 'https://www.aeaweb.org' + href
        article_links.append(full_url)
# Print the article links
for link in article_links:
    print(link)
```

Exercise

- Go to https://www.aeaweb.org/issues/785 to download the links to each article.
- Go to each of the above links to download the abstract of the article.
- Print the all abstracts on the screen.

Exercise

First, I asked GPT to scrape the abstract for a given URL.



Exercise

I got the following code.

```
import requests
from bs4 import BeautifulSoup

# URL of the page to scrape
url = 'https://www.aeaweb.org/articles?id=10.1257/aer.20221227'

# Send a GET request to the page
response = requests.get(url)
response.raise_for_status()  # Check that the request was successful

# Parse the page content with BeautifulSoup
soup = BeautifulSoup(response.content, 'html.parser')

# Find the abstract section
abstract_section = soup.find('section', class_='article-information abstract')
if abstract_section:
    # Extract and print the abstract text
    abstract_text = abstract_section.get_text(strip=True, separator=' ')
    print('Abstract:', abstract_text)
else:
    print('Abstract section not found')
```

Exercise

I change it to a function “getabstract”:

```
import requests
from bs4 import BeautifulSoup

def getabstract(url):
    response = requests.get(url)
    response.raise_for_status()
    soup = BeautifulSoup(response.content, 'html.parser')
    abstract_section = soup.find('section', class_='article-information
abstract')
    if abstract_section:
        # Extract and print the abstract text
        abstract_text = abstract_section.get_text(strip=True, separator=' ')
        print('Abstract:', abstract_text)
    else:
        print('Abstract section not found')
```

Exercise

Combine the previous codes, you got it!

```python
import requests
from bs4 import BeautifulSoup

def getabstract(url):
    response = requests.get(url)
    response.raise_for_status()
    soup = BeautifulSoup(response.content, 'html.parser')
    abstract_section = soup.find('section', class_='article-information abstract')
    if abstract_section:
        # Extract and print the abstract text
        abstract_text = abstract_section.get_text(strip=True, separator=' ')
        print('Abstract:', abstract_text)
    else:
        print('Abstract section not found')


url = 'https://www.aeaweb.org/issues/785'
response = requests.get(url)
response.raise_for_status()  # Check that the request was successful
soup = BeautifulSoup(response.content, 'html.parser')
article_links = []
for a_tag in soup.find_all('a', href=True):
    href = a_tag['href']
    if href.startswith('/articles?id='):
        full_url = 'https://www.aeaweb.org' + href
        article_links.append(full_url)

for link in article_links:
    getabstract(link)
```

We can do more!

- On page https://www.aeaweb.org/journals/aer/issues, we can scrape the URL of each issue.
- On the issue page, we can scrape the URL of each article.
- On the article page, we can scrape the abstract of each article (it took me two hours).
- Then, we can save all abstracts in one single file.
- After that, we can perform topic modeling!

We can do more!

I have already uploaded the txt file to the course website, which can be found here:
http://ximarketing.github.io/data/AER_abstracts.txt

Please try the top words and LDA algorithm yourself and demonstrate the main topics in modern economics.

Exercise: Download the Photos of All HKU Business School Professors from https://www.hkubs.hku.hk/people/faculty?pg=1&staff_type=faculty&subject_area=all&track=professoriate&region=hku

Downloading from a single page:

```python
import requests
from bs4 import BeautifulSoup
import os
url = "https://www.hkubs.hku.hk/people/faculty?pg=1&staff_type=faculty&subject_area=all&track=professoriate&region=hku"
response = requests.get(url)
if response.status_code == 200:
    soup = BeautifulSoup(response.text, 'html.parser')

    photo_divs = soup.find_all('div', class_='people-item')
    photo_urls = []

    for div in photo_divs:
        img_tag = div.find('img')
        if img_tag and 'src' in img_tag.attrs:
            photo_url = img_tag['src']
            photo_urls.append(photo_url)

    os.makedirs('profile_photos', exist_ok=True)

    for idx, photo_url in enumerate(photo_urls):
        img_response = requests.get(photo_url)

        if img_response.status_code == 200:
            # Save the photo to the directory
            with open(f'profile_photos/photo_{idx + 1}.jpg', 'wb') as f:
                f.write(img_response.content)
            print(f'Downloaded: photo_{idx + 1}.jpg')
        else:
            print(f'Failed to download photo from {photo_url}')
else:
    print(f'Failed to retrieve the page. Status code: {response.status_code}')
```

I further asked the following question...

I have 10 URLs each containing 16 photos, and I want to download all these photos. The URLs are https://www.hkubs.hku.hk/people/faculty?pg=1&staff_type=faculty&subject_area=all&track=professoriate&region=hku to https://www.hkubs.hku.hk/people/faculty?pg=10&staff_type=faculty&subject_area=all&track=professoriate&region=hku

I am done!


photo_1
photo_2
photo_3
photo_4
photo_5
photo_6
photo_7
photo_8
photo_9
photo_10
photo_11
photo_12
photo_13
photo_14
photo_15
photo_16
photo_page1_1
photo_page1_2
photo_page1_3
photo_page1_4
photo_page1_5
photo_page1_6
photo_page1_7
photo_page1_8
photo_page1_9
photo_page1_10
photo_page1_11
photo_page1_12
photo_page1_13
photo_page1_14
photo_page1_15
photo_page1_16
photo_page2_1
photo_page2_2
photo_page2_3
photo_page2_4
photo_page2_5
photo_page2_6
photo_page2_7
photo_page2_8

The code I used

```python
import requests
from bs4 import BeautifulSoup
import os
base_url = "https://www.hkubs.hku.hk/people/faculty?pg={}&staff_type=faculty&subject_area=all&track=professoriate&region=hku"
os.makedirs('profile_photos', exist_ok=True)
for page in range(1, 11):
    url = base_url.format(page)
    print(f"Fetching page: {url}")
    response = requests.get(url)
    if response.status_code == 200:
        soup = BeautifulSoup(response.text, 'html.parser')
        photo_divs = soup.find_all('div', class_='people-item')
        for idx, div in enumerate(photo_divs):
            img_tag = div.find('img')
            if img_tag and 'src' in img_tag.attrs:
                photo_url = img_tag['src']
                img_response = requests.get(photo_url)
                if img_response.status_code == 200:
                    photo_filename = f'profile_photos/photo_page{page}_{idx + 1}.jpg'
                    with open(photo_filename, 'wb') as f:
                        f.write(img_response.content)
                    print(f'Downloaded: {photo_filename}')
                else:
                    print(f'Failed to download photo from {photo_url}')
    else:
        print(f'Failed to retrieve the page. Status code: {response.status_code}')
```

Try to scrape article titles from the following page:
https://pubsonline.informs.org/toc/mnsc/71/1

What else can we do with scraping?

What else can we do with scraping?

- Scraping a website with selenium.
- Web scraping with concurrency (parallel scraping).
- Changing IP addresses with VPN (using proxies/VPNs).

https://www.youtube.com/embed/cc26zFE8X1k?enablejsapi=1

Scraping with Selenium

You need to install selenium and download Chrome driver

Moreover, add Chrome driver to your system path

```
from selenium import webdriver
from selenium.webdriver.common.by import By
url = "https://pubsonline.informs.org/toc/mnsc/71/1"
driver = webdriver.Chrome()
driver.get(url)

titles = driver.find_elements(By.CLASS_NAME, 'issue-item__title')
# Extract and print the text of each title
for title in titles:
    article_title = title.find_element(By.TAG_NAME, 'a').text
    print(article_title)
# Close the WebDriver
driver.implicitly_wait(5)
driver.quit()
```

Class Review

Basic Operations

Linear Regression

Logistic Regression

Multinomial Logit Model

Artificial Neural Network

Data Visualization

Statistical Tests

Advanced Algorithms

Clustering (and Expectation-Maximization)

Monte Carlo Algorithm

Dynamic Programming

Divide and Conquer

Unstructured Data

Text Data: Word Frequency, Topic Modeling, Summarization

Image: Color, Brightness, Convolution, etc.

You can also find APIs for text and image analysis.

API and Scraper

How to apply for and use an API.

How to rely on AI to generate your own data scraper.