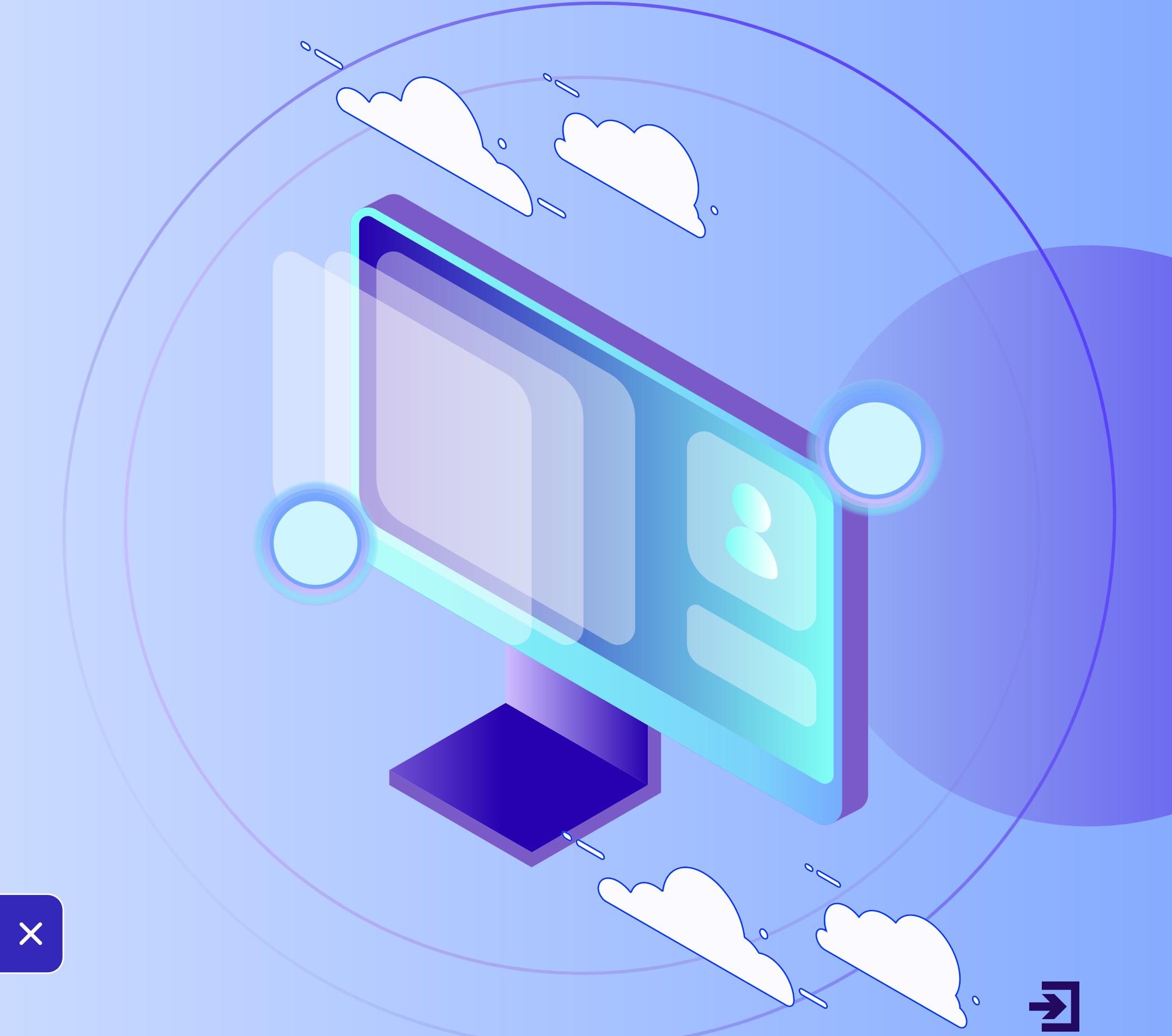




# WEB SCRAPING

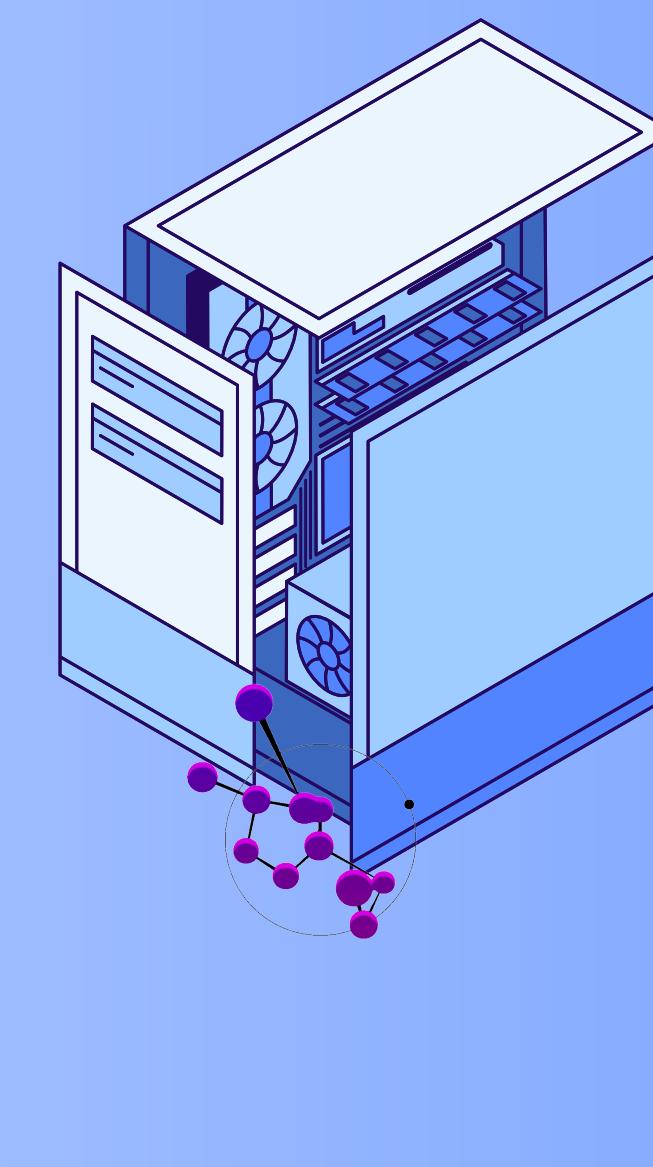
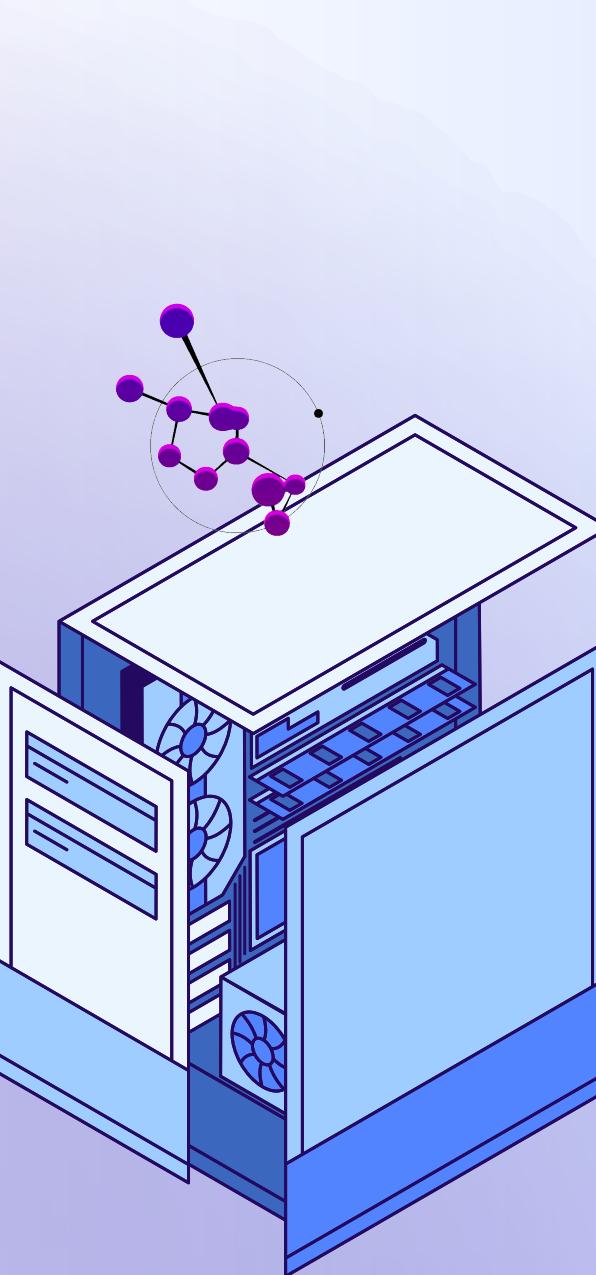


Natalia Machlus, Julia Taborek



# CZYM JEST WEB SCRAPING?

---



Web scraping, nazywany też zdrapywaniem lub skrobaniem stron, to zautomatyzowany proces pozyskiwania danych ze stron internetowych, najczęściej przy wykorzystaniu specjalnego programu czy skryptu (zwanego skraperem) naśladującego zachowanie człowieka przeglądającego stronę internetową.



# RODZAJE WEB SCRAPINGU

## SPECYFICZNY

- znana jest zarówno struktura, jak i zawartość stron internetowych,
- algorytm naśladuje zachowanie człowieka, który odwiedza stronę internetową i pobiera interesujące go informacje,
- wymaga dostosowania kodu do struktury konkretnej strony (różny układ interfejsu i sposób prezentacji danych, stosowanie różnych technologii do prezentacji treści)

## OGÓLNY

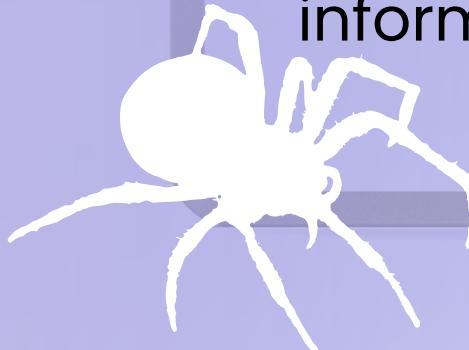
- użytkownik nie ma wcześniejszej wiedzy na temat treści i budowy strony internetowej,
- pobierana jest zawartość całej witryny, która następnie jest przetwarzana w celu wyodrębnienia potrzebnych informacji,
- stosowany do pozyskiwania informacji, dostępnych na różnych stronach w podobnych formatach.



# KONCEPCJA

## ROBOT INDEKSUJĄCY

- nazywany „crawlerem” lub „pająkiem internetowym”
- przemieszcza się po różnych stronach internetowych i zbiera linki do stron z przydatnymi informacjami dla skrobaka
- uznawany za przewodnik skrobaka, który pokazuje mu, gdzie ma szukać informacji.



**Web scrapping opiera się na współpracy dwóch robotów: robota indeksującego i skrobaka internetowego (webscrapera).**

## SKROBAK INTERNETOWY

- program, który wykorzystuje informacje od robota indeksującego w celu dokładnego i szybkiego wyodrębniania danych z konkretnej strony internetowej
- budowa i złożoność determinowana jest stawianymi im wymaganiami,
- używa lokalizatorów (selektorów) danych do wyszukiwania informacji, które użytkownik chce wyodrębnić z pliku HTML (strony internetowej).



# WYZWANIA WEB SCRAPINGU



## Wyzwania techniczne:

- zagrożenie związane z możliwością ściągnięcia zawirusowanego oprogramowania,
- braki danych spowodowane, np. brakiem dostępu do Internetu,
- scrapowanie może obciążać stronę utrudniając lub uniemożliwiając jej prawidłowe działanie

## Problemy prawne:

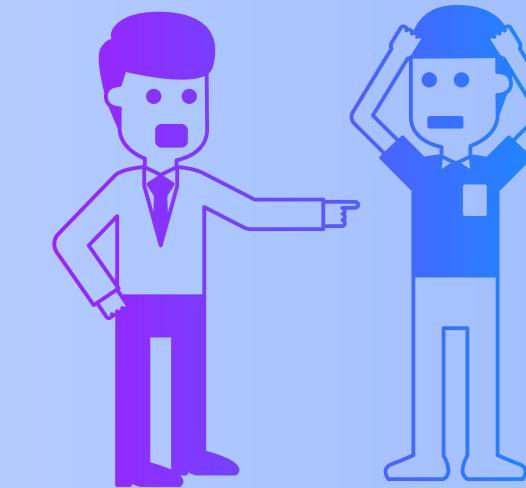
- brak jednolitych międzynarodowych wytycznych w zakresie scrapingu,
- możliwość wykorzystywania pobranych danych przez konkurencję właścicieli strony,
- właściciele stron internetowych powinni być informowani o wykorzystaniu ich danych, o ile jest to wykonalne



# DOSTĘP DO SCRAPOWANIA

## DZIAŁANIA ADMINISTRATORÓW STRONY INTERNETOWEJ

- Zmienne struktury strony internetowej, utrudniające dostosowanie do niej kodu,
- Blokowanie adresów IP na podstawie wykrytych podejrzanych zachowań,
- Kontrolowanie liczby zapytań poprzez zdefiniowanie limitu zapytań,
- Wymaganie potwierdzenia, że użytkownik jest człowiekiem,
- Możliwość zażądania zaniechania pobierania danych na drodze sądowej,
- Używanie pliku robots.txt. do komunikacji z robotem internetowym i wskazywanie tam części witryny, które nie powinny być przetwarzane ani skanowane.



## OBEJŚCIA WEB SCRAPERÓW

- Symulacja zachowania użytkownika:
  1. ustawianie zmiennych odstępów pomiędzy wysłanymi zapytaniami,
  2. losowe klikanie w części strony, przewijanie w górę i w dół, otwieranie dodatkowych kart,
  3. używanie różnych adresów IP.
- Wykorzystywanie API do pobierania danych
- Dzielenie procesu pobierania danych na więcej urządzeń

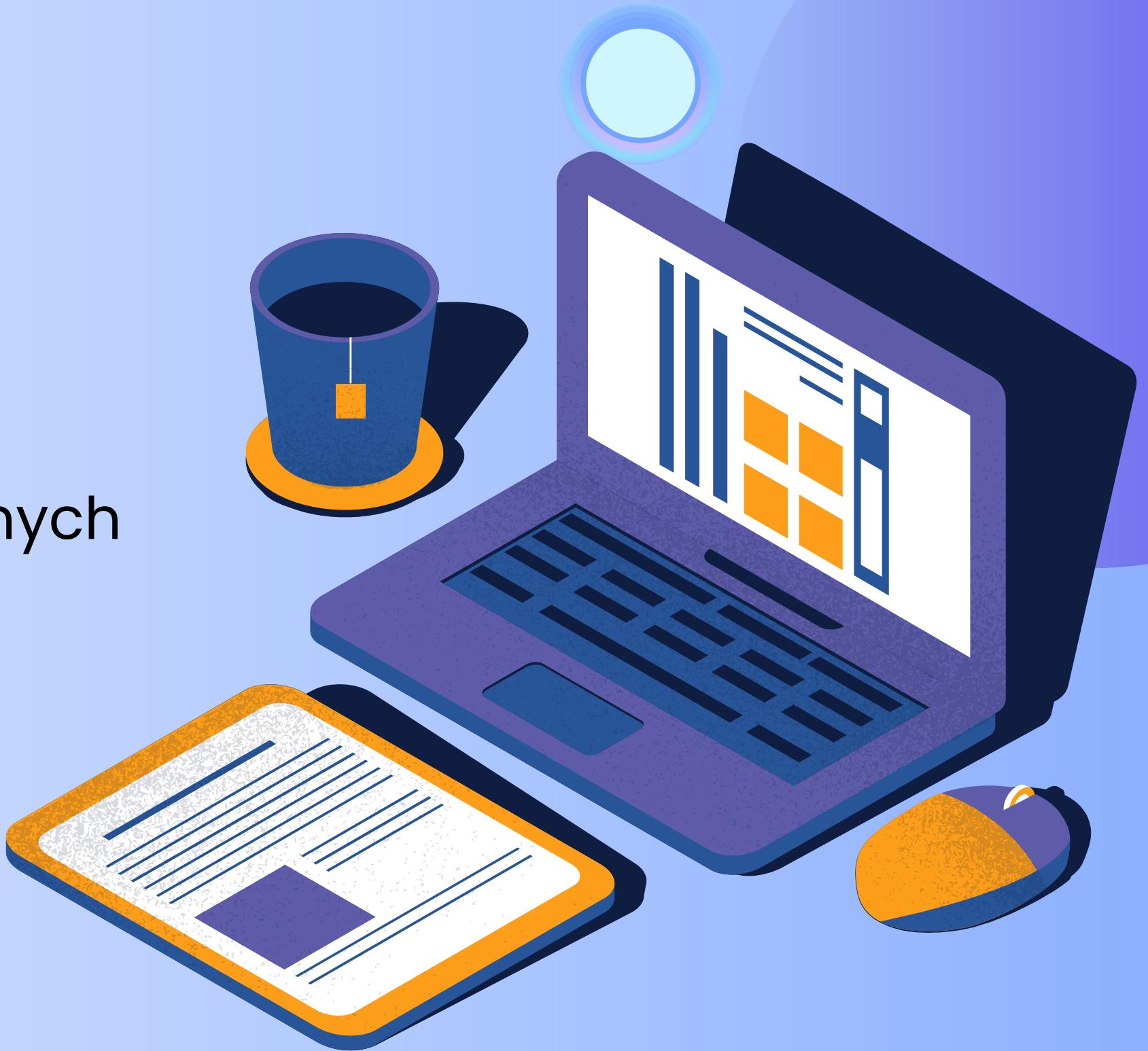


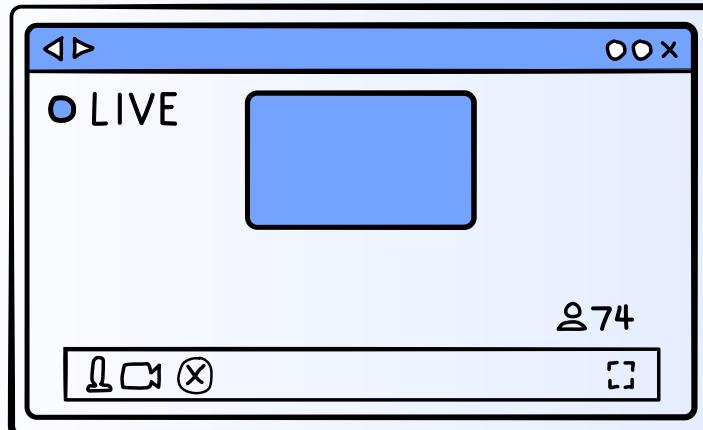


# PYTHON

Python oferuje kilka bibliotek pomocnych przy scrapowaniu danych:

- requests
- moduł urllib.request
- BeautifulSoup
- Selenium
- Lxml
- Playwright
- Scrapy





# REQUESTS

# Biblioteka pozwalająca na wysyłanie żądań HTTP przy użyciu języka Python



# STOSOWANIE

Działa jako protokół żądanie-odpowiedź między klientem a serwerem.

Żądanie HTTP zwraca obiekt odpowiedzi (Response Object) ze wszystkimi danymi odpowiedzi (treść, kodowanie, status itp.).

Metoda GET służy do pobierania informacji z danego serwera za pomocą danego identyfikatora URI. Metoda GET wysyła zakodowane informacje o użytkowniku dołączone do żądania strony.



# BEAUTIFULSOUP

**pakiet, który analizuje kod HTML (lub XML) oraz pomaga organizować i formatować dane internetowe w bardziej przyjazne struktury.**



## STOSOWANIE

1. Pozwala na wyodrębnienie potrzebnych informacji bezpośrednio z tagów i klas CSS.
2. Udostępnia kilka prostych metod i zwrotów do kierowania, wyszukiwania i zmieniania drzewa parsowania.
3. Automatycznie konwertuje przychodzące rekordy na Unicode, a wychodzące formularze na UTF-8. Nie trzeba myśleć o kodowaniu, chyba że dokument nie definiuje kodowania, a BeautifulSoup nie może go złapać. Wtedy wystarczy wybrać oryginalne kodowanie.
4. Biblioteka BeautifulSoup została zbudowana na bazie bibliotek parsujących HTML, takich jak html5lib, lxml, html.parser itp. Dzięki temu obiekt BeautifulSoup oraz biblioteka parsera mogą być tworzone w tym samym czasie.

```
 soup = BeautifulSoup(r.content, 'html.parser')
```



# BEAUTIFULSOUP



## PRZYKŁADY ZASTOSOWANIA:

### 1. Analizowanie kodu HTML:

```
print('Pobieranie tagu title z zawartością: ')
print(soup.title)

print('Pobiera nazwę tagu: ')
print(soup.title.name)

print('Pobiera zawartość tagu: ')
print(soup.title.string)

print('Pobiera nazwę tagu nadziednego: ')
print(soup.title.parent.name)
```

```
Pobieranie tagu title z zawartością:
<title>
    LEKsykon - normy i skale medyczne
</title>
Pobiera nazwę tagu:
title
Pobiera zawartość tagu:

    LEKsykon - normy i skale medyczne

Pobiera nazwę tagu nadziednego:
head
```

### 2. Znajdowanie elementów:

```
# wyszukanie w kodzie HTML wszystkich elementów <a> (linków)
# i pobieranie wartości atrybutu href, czyli adresu URL, na który prowadzi ten link.
for link in soup.find_all('a'):
    print(link.get('href'))
```

```
mailto:biuro@lekseek.com
https://twitter.com/lekseek
https://www.facebook.com/Lekseek
https://plus.google.com/108474817735515889515/about
https://www.youtube.com/user/lekseekpolska
```

### 3. Wyodrębnianie tekstu z klas

```
s = soup.find('div', class_='body_overlay')

lines = s.find_all('p')

for line in lines:
    print(line.text)
```

```
Twój portal informacji medycznej
Niezalogowany/a
Aplikacje Mobilne
Aktualizacja bazy 2024-11-02 01:45:01
LekSeek ® Polska © 2003-2018

Partnerzy
```



# PLAYWRIGHT

**zestaw narzędzi do automatyzacji przeglądarek, które wykorzystuje się również do web scrapingu**



## STOSOWANIE

1. Szczególnie przydatny do pobierania danych z dynamicznych aplikacji internetowych,
2. Umożliwia skryptowi interakcję z przeglądarką bez konieczności analizy wewnętrznego kodu strony i jej mechanizmów,
3. Umożliwiają scrapowanie ze stron opartych na skryptach JavaScript,
4. Omija blokady, ponieważ scraper uruchamia pełną przeglądarkę, co bardziej przypomina zachowanie człowieka niż samodzielne żądania HTTP,
5. Oferuje podejście asynchroniczne – rozpoczęcie innych zadań bez czekania na zakończenie poprzednich



# PORÓWNANIE

## BEAUTIFULSOUP

- ograniczona do Pythona
- stosowana w przypadku pobierania danych ze stron statycznych (wszystkie treści są załadowane po odświeżeniu),
- nie umożliwia interakcji ze stroną, użytkownik musi wcześniej pobrać kod HTML, aby móc go przetwarzać,
- każda operacja musi zostać zakończona, zanim zacznie się kolejna (podejście synchroniczne),
- łatwiejsza do nauki, zalecana dla początkujących

## PLAYWRIGHT

- wspiera różne języki programowania,
- lepszy do złożonych zadań przy pobieraniu dynamicznych treści,
- pełna interakcja z przeglądarką, umożliwiająca klikanie, przewijanie i wypełnianie formularzy,
- oferuje podejście synchroniczne i asynchroniczne,
- bardziej skomplikowany, często wymaga napisania rozbudowanego skryptu



# BIBLIOGRAFIA

1. Główny Urząd Statystyczny, (2022). *Nowoczesne technologie i nowe źródła danych w pomiarze inflacji*
2. Geeksforgeeks, (4.10.2024). *Python Web Scraping Tutorial*,  
<https://www.geeksforgeeks.org/python-web-scraping-tutorial/>
3. Geeksforgeeks, (3.06.2024). *How to Scrape Websites with BeautifulSoup and Python ?* <https://www.geeksforgeeks.org/how-to-scrape-websites-with-beautifulsoup-and-python/>
4. Real Python, (28.03.2024). *A Practical Introduction to Web Scraping in Python*,  
<https://realpython.com/python-web-scraping-practical-introduction/#scrape-and-parse-text-from-websites>
5. ScrapFly, (22.08.2024). *Web Scraping with Playwright and Python*,  
<https://scrapfly.io/blog/web-scraping-with-playwright-and-python/>
6. (14.11.2020) Mamczur M., *Web Scraping – co to i jakie są „dobre” praktyki?*,  
<https://mirosławmamczur.pl/web-scraping-co-to-i-jakie-sa-dobre-praktyki/>
7. (15.03.2023), Szczudło A., *Web scraping danych z internetu – jak to zrobić zgodnie z prawem*, <https://creativa.legal/legalny-scraping-danych/>



**DZIĘKUJEMY ZA  
UWAGĘ**

Natalia Machlus, Julia Taborek