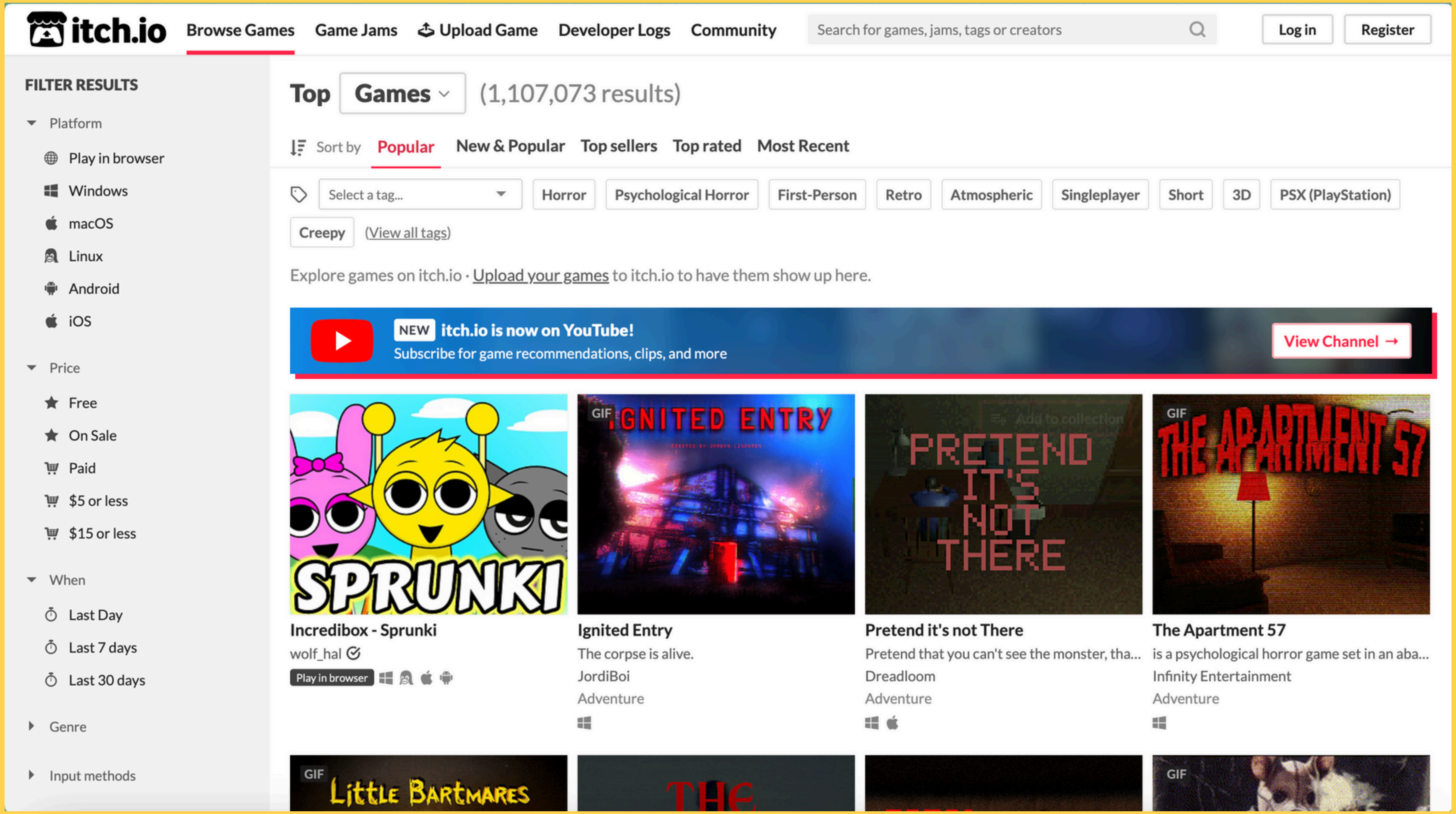
DATA102 Data Scraping Homework

ITCH.IO

GROUP 7

GAME MASTER

ITCH.IO



itch.io · Browse Games · Game Jams · Upload Game · Developer Logs · Community · Search for games, jams, tags or creators · Log in · Register

FILTER RESULTS

Platform
- Play in browser
- Windows
- macOS
- Linux
- Android
- iOS

Price
- Free
- On Sale
- Paid
- $5 or less
- $15 or less

When
- Last Day
- Last 7 days
- Last 30 days

Genre

Input methods

Top Games (1,107,073 results)

Sort by Popular · New & Popular · Top sellers · Top rated · Most Recent

Select a tag... · Horror · Psychological Horror · First-Person · Retro · Atmospheric · Singleplayer · Short · 3D · PSX (PlayStation) · Creepy · (View all tags)

Explore games on itch.io · Upload your games to itch.io to have them show up here.

NEW itch.io is now on YouTube!
Subscribe for game recommendations, clips, and more
View Channel →

Incredibox - Sprunki
wolf_hal
Play in browser

Ignited Entry
The corpse is alive.
JordiBoi
Adventure

Pretend it's not There
Pretend that you can't see the monster, tha...
Dreadloom
Adventure

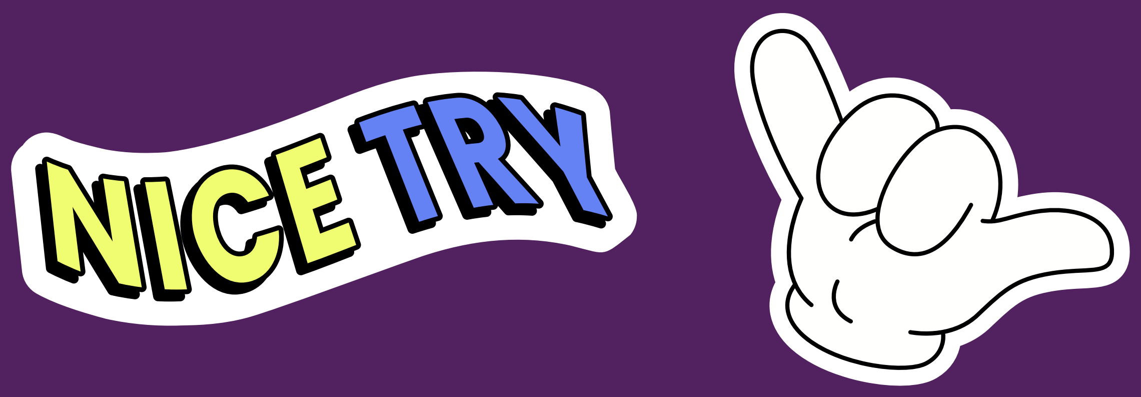
The Apartment 57
is a psychological horror game set in an aba...
Infinity Entertainment
Adventure

READY

BACKGROUND + RATIONALE

What the website is about
Why did we choose to scrape the website

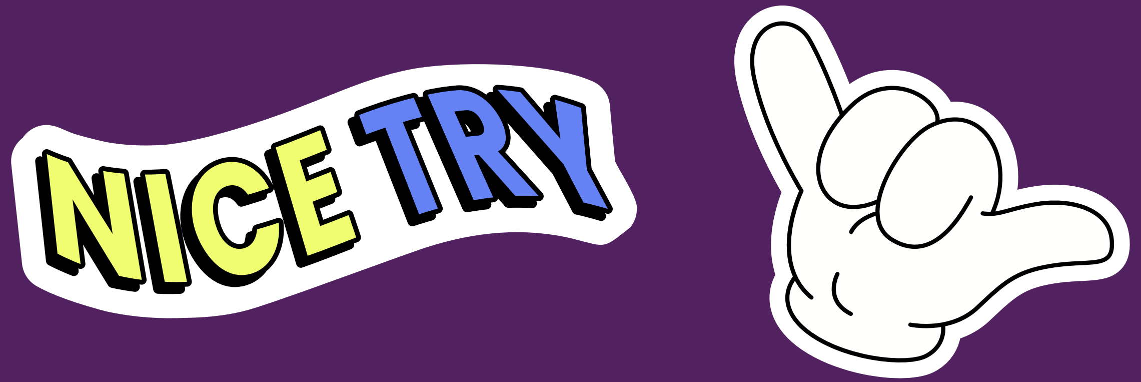
WHAT IS THE WEBSITE ABOUT



01 Itch.io is an online marketplace and community for independent game developers to publish, distribute, and sell their games

02 It supports free and paid games, game jams, and various creative projects, including tabletop RPGs and digital assets

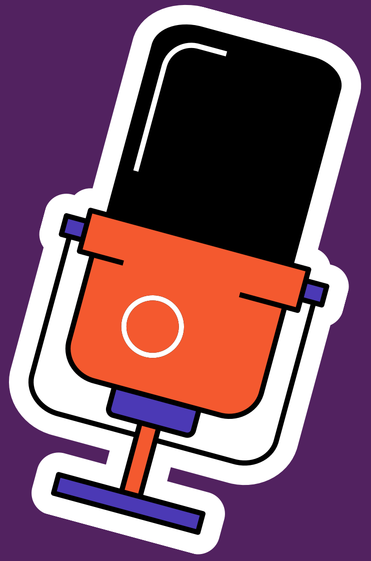
WHY CHOOSE TO SCRAPE THE WEBSITE



01 Itch.io hosts a variety of interesting indie games and assets

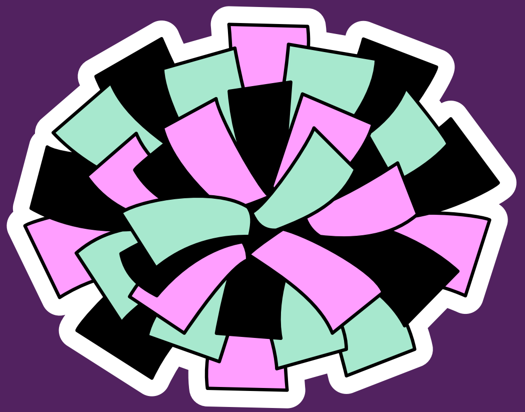
02 Scraping data from Itch’s catalog can help in creating recommender systems that focus on the indie scene of video games and market analysis

03 It could also be useful for gathering data on game trends, developer activity, pricing models, or user reviews



CHALLENGES ENCOUNTERED

CHALLENGES ENCOUNTERED



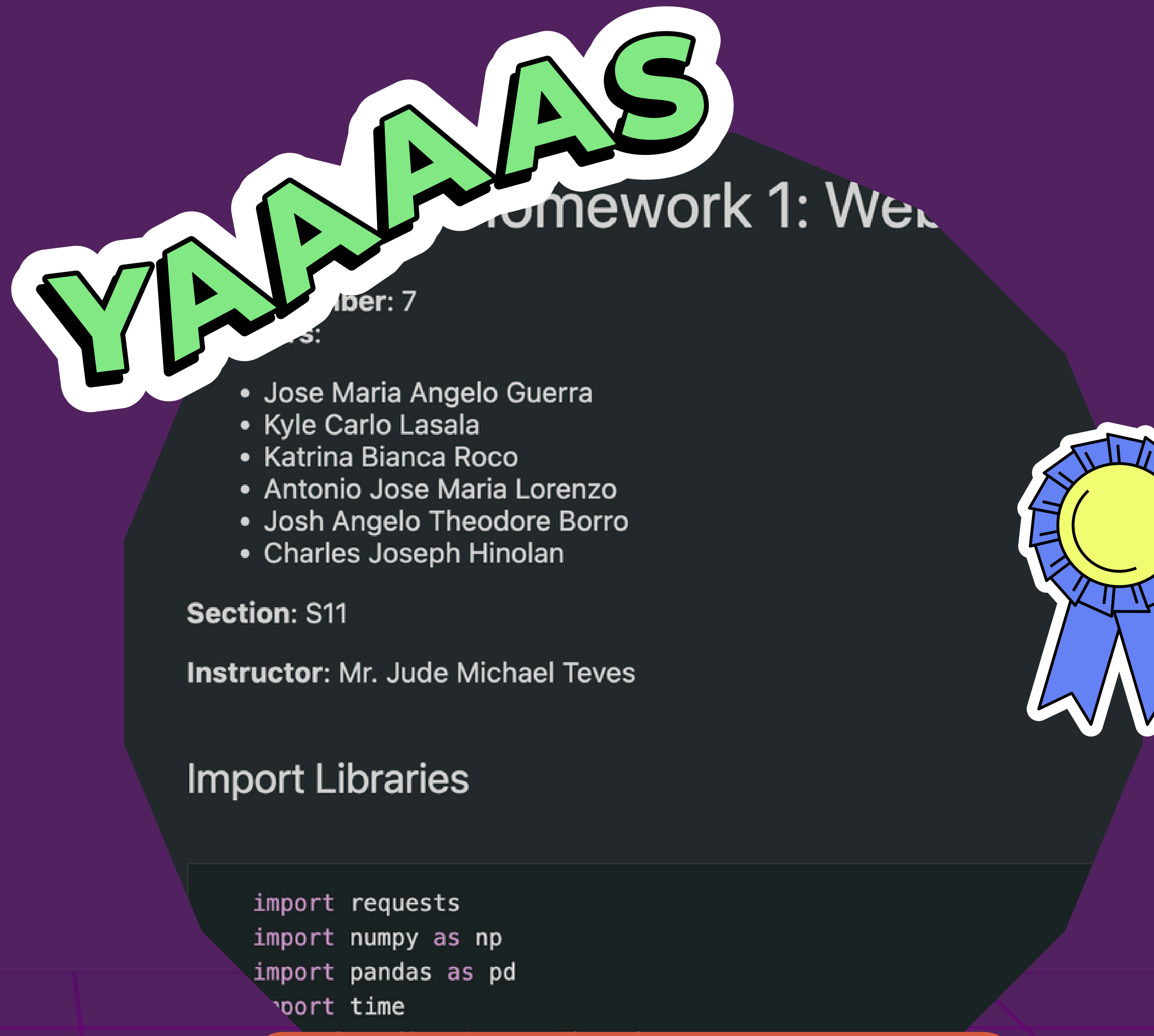
01 Security reasons for some websites

02 Items with inconsistent divs present

03 Inconsistent table formatting for each game
(extracting could have been easier with the lxml library)

04 Time-consuming

PROJECT NOTEBOOK DEMONSTRATION



omework 1: We

ber: 7

s:

- Jose Maria Angelo Guerra
- Kyle Carlo Lasala
- Katrina Bianca Roco
- Antonio Jose Maria Lorenzo
- Josh Angelo Theodore Borro
- Charles Joseph Hinolan

Section: S11

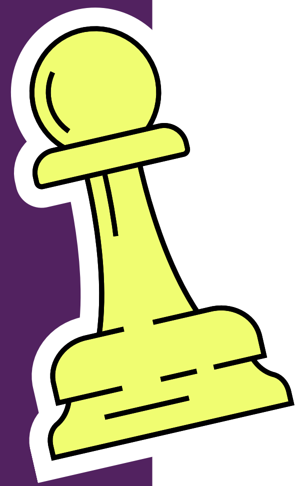
Instructor: Mr. Jude Michael Teves

Import Libraries

```
import requests
import numpy as np
import pandas as pd
 port time
```

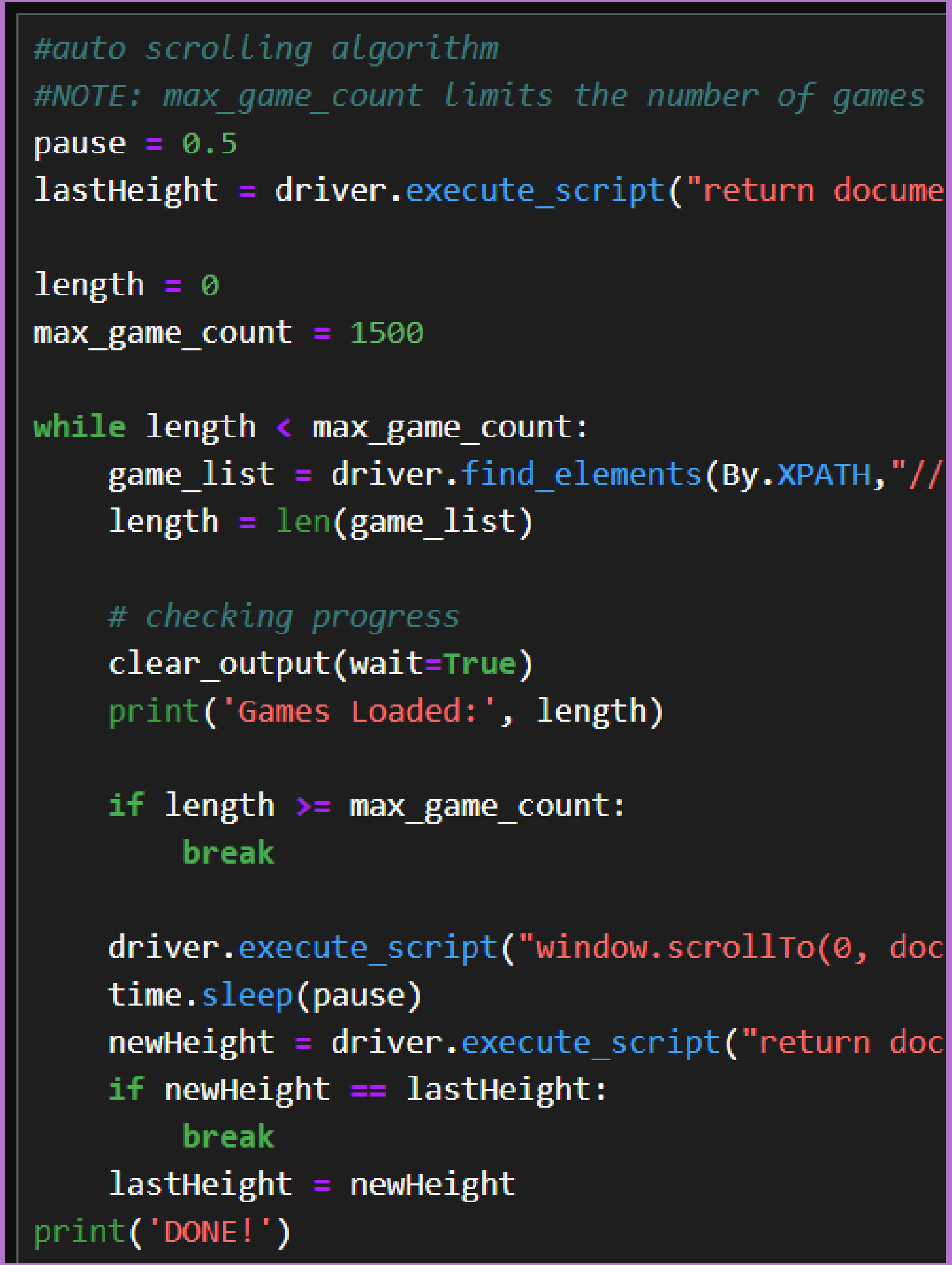
JUPYTER NOTEBOOK

SCRAPING PROCESS

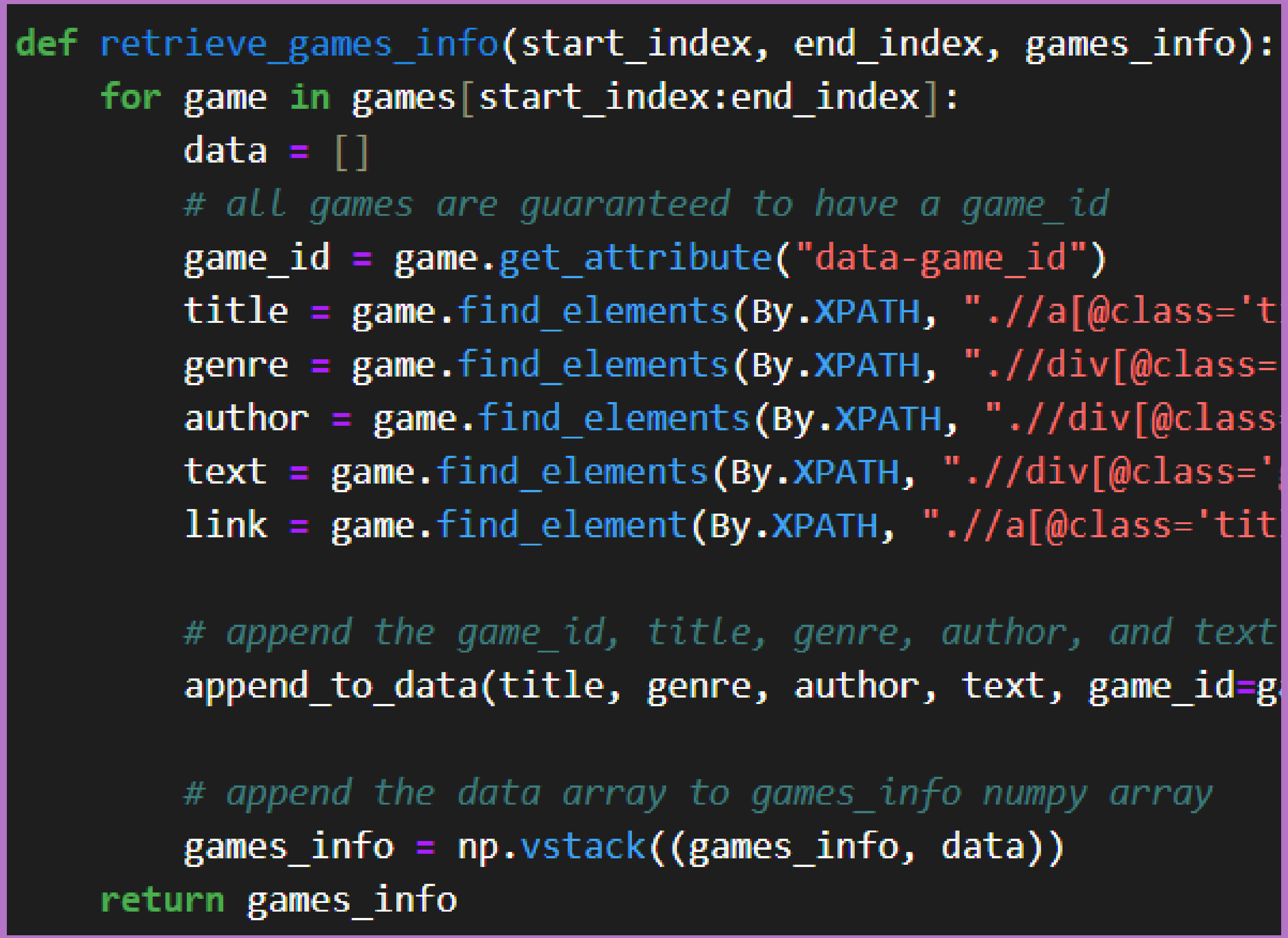


1. AUTO SCROLLING ALGORITHM

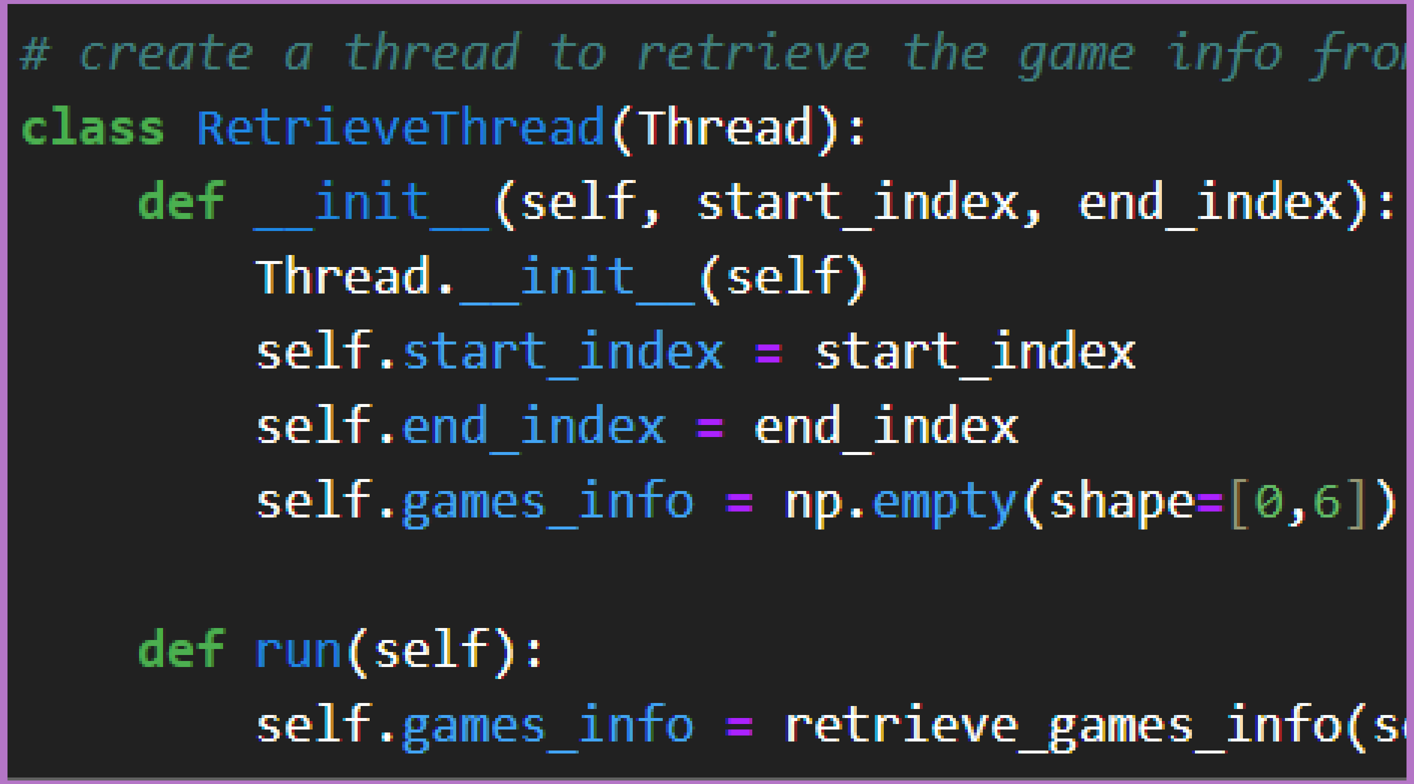
Since the website uses lazy loading through the infinite scroll, the first procedure after opening the Itch's catalog is to scroll the website until the list reaches 1500 games.



```
#auto scrolling algorithm
#NOTE: max_game_count limits the number of games
pause = 0.5
lastHeight = driver.execute_script("return docume

length = 0
max_game_count = 1500

while length < max_game_count:
    game_list = driver.find_elements(By.XPATH,"//
    length = len(game_list)

    # checking progress
    clear_output(wait=True)
    print('Games Loaded:', length)

    if length >= max_game_count:
        break

    driver.execute_script("window.scrollTo(0, doc
    time.sleep(pause)
    newHeight = driver.execute_script("return doc
    if newHeight == lastHeight:
        break
    lastHeight = newHeight
print('DONE!')
```



```
def retrieve_games_info(start_index, end_index, games_info):
    for game in games[start_index:end_index]:
        data = []
        # all games are guaranteed to have a game_id
        game_id = game.get_attribute("data-game_id")
        title = game.find_elements(By.XPATH, ".//a[@class='t
        genre = game.find_elements(By.XPATH, ".//div[@class=
        author = game.find_elements(By.XPATH, ".//div[@class
        text = game.find_elements(By.XPATH, ".//div[@class='
        link = game.find_element(By.XPATH, ".//a[@class='tit

        # append the game_id, title, genre, author, and text
        append_to_data(title, genre, author, text, game_id=g

        # append the data array to games_info numpy array
        games_info = np.vstack((games_info, data))
    return games_info
```



```
# create a thread to retrieve the game info fro
class RetrieveThread(Thread):
    def __init__(self, start_index, end_index):
        Thread.__init__(self)
        self.start_index = start_index
        self.end_index = end_index
        self.games_info = np.empty(shape=[0,6])

    def run(self):
        self.games_info = retrieve_games_info(s
```

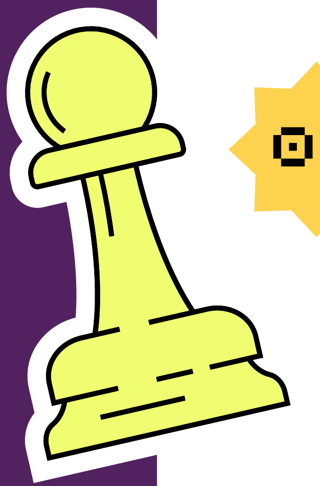
2. SCRAPING ITCH.IO

01 Game ID, Title, Genre, Author, Game Text, and Link are scraped features found in Itch’s catalog.

02 Multithreading was implemented to scrape these features faster.

03 Save the Catalog Info features into a pandas DataFrame.

3. SCRAPING EACH GAME SITE

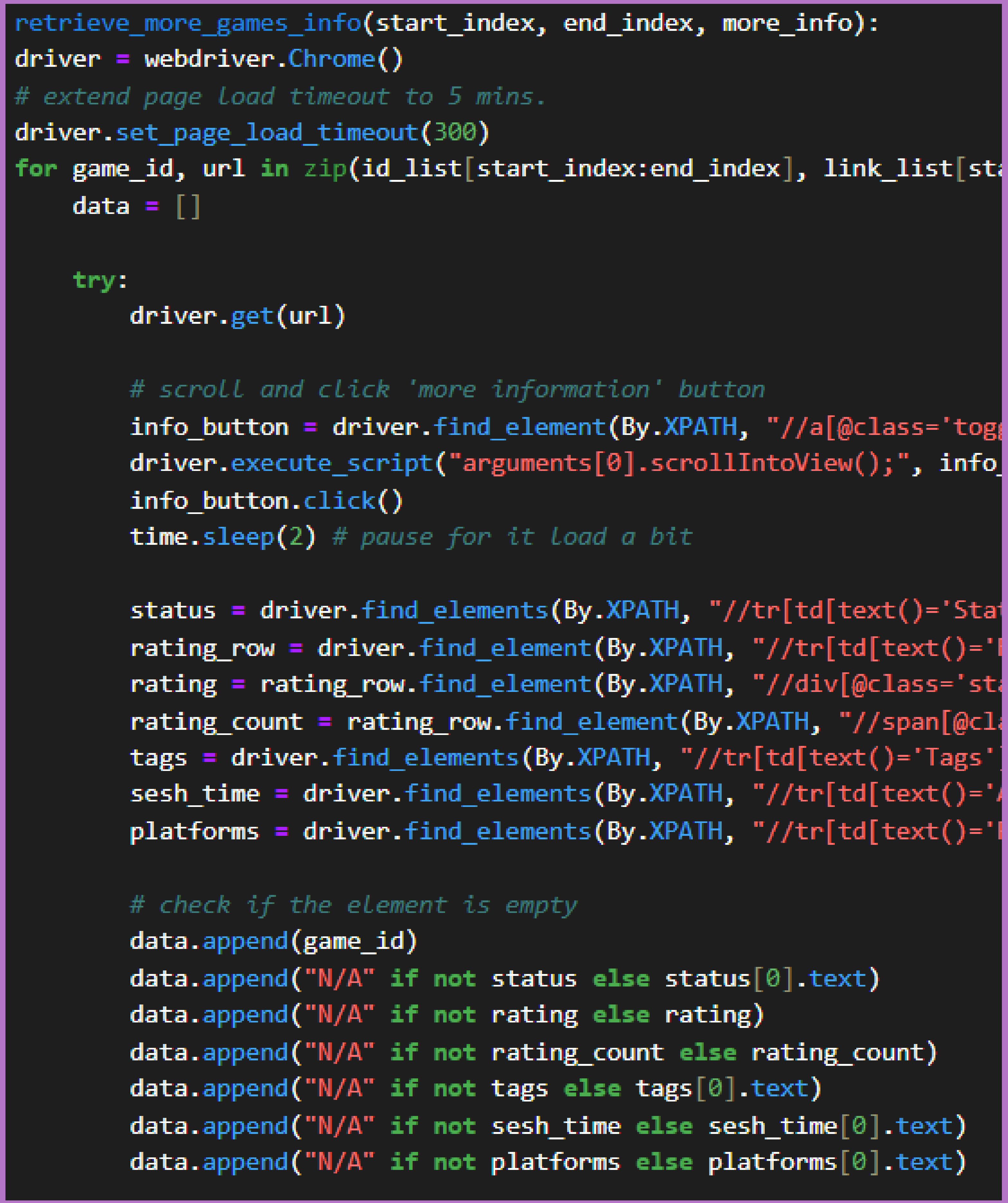


01 The link feature of each game is used to access the game site.

02 The game site contains the status, rating, rating count, tags, average session time, and platforms features.

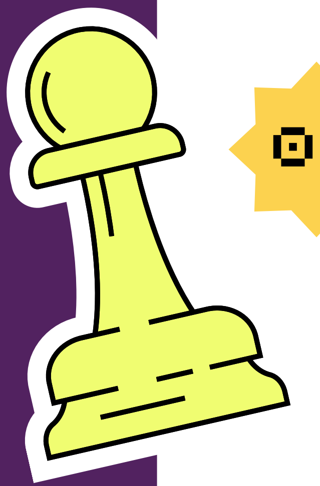
03 Multithreading was implemented to instantiate multiple web drivers for faster scraping.

04 Save the Game Site Info features into a pandas DataFrame.



```
retrieve_more_games_info(start_index, end_index, more_info):
driver = webdriver.Chrome()
# extend page load timeout to 5 mins.
driver.set_page_load_timeout(300)
for game_id, url in zip(id_list[start_index:end_index], link_list[sta
    data = []

    try:
        driver.get(url)

        # scroll and click 'more information' button
        info_button = driver.find_element(By.XPATH, "//a[@class='togg
        driver.execute_script("arguments[0].scrollIntoView();", info_
        info_button.click()
        time.sleep(2) # pause for it load a bit

        status = driver.find_elements(By.XPATH, "//tr[td[text()='Sta
        rating_row = driver.find_element(By.XPATH, "//tr[td[text()='
        rating = rating_row.find_element(By.XPATH, "//div[@class='sta
        rating_count = rating_row.find_element(By.XPATH, "//span[@cla
        tags = driver.find_elements(By.XPATH, "//tr[td[text()='Tags']
        sesh_time = driver.find_elements(By.XPATH, "//tr[td[text()='
        platforms = driver.find_elements(By.XPATH, "//tr[td[text()='

        # check if the element is empty
        data.append(game_id)
        data.append("N/A" if not status else status[0].text)
        data.append("N/A" if not rating else rating)
        data.append("N/A" if not rating_count else rating_count)
        data.append("N/A" if not tags else tags[0].text)
        data.append("N/A" if not sesh_time else sesh_time[0].text)
        data.append("N/A" if not platforms else platforms[0].text)
```

3. SCRAPING EACH GAME SITE

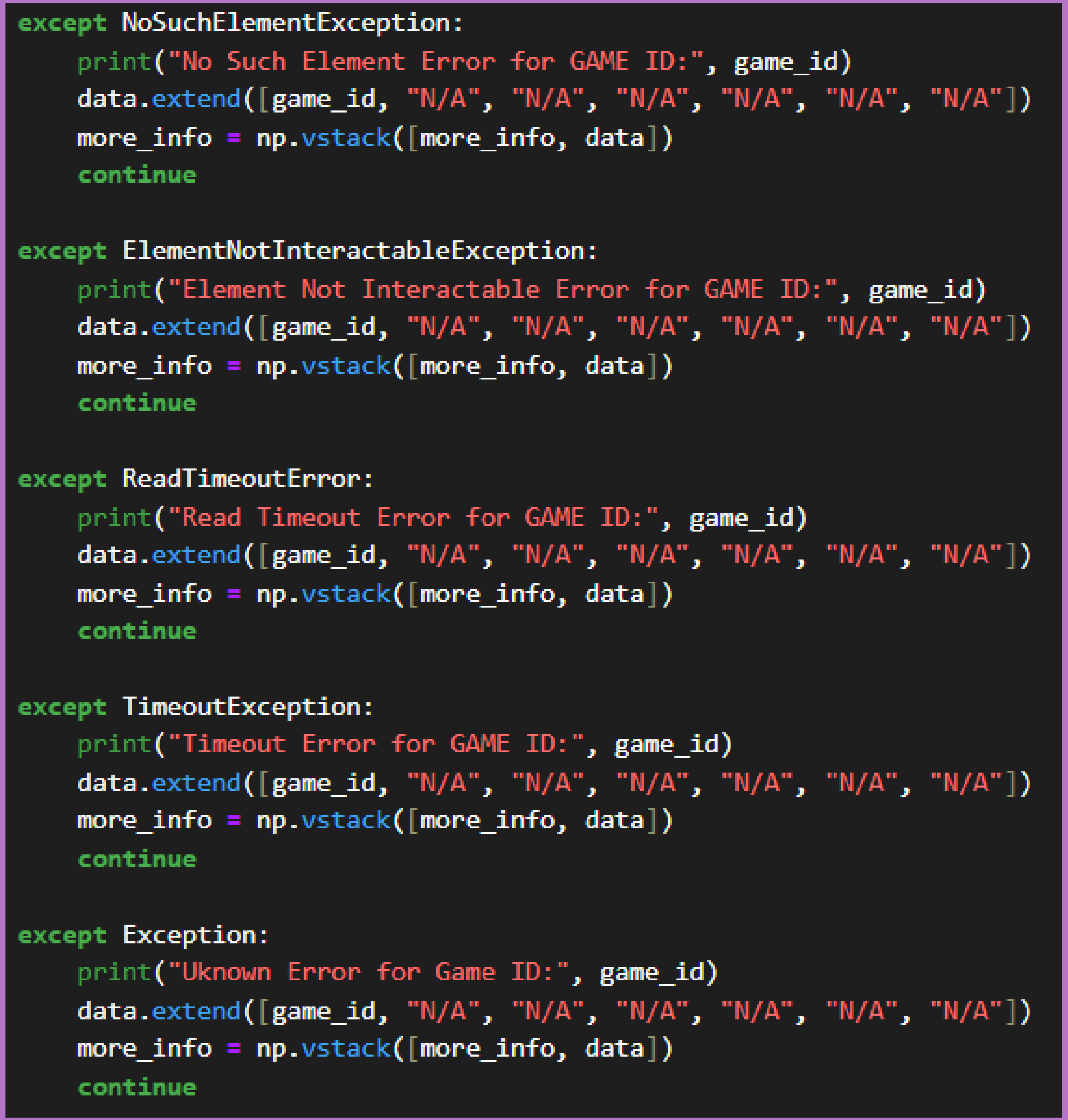


01 The link feature of each game is used to access the game site.

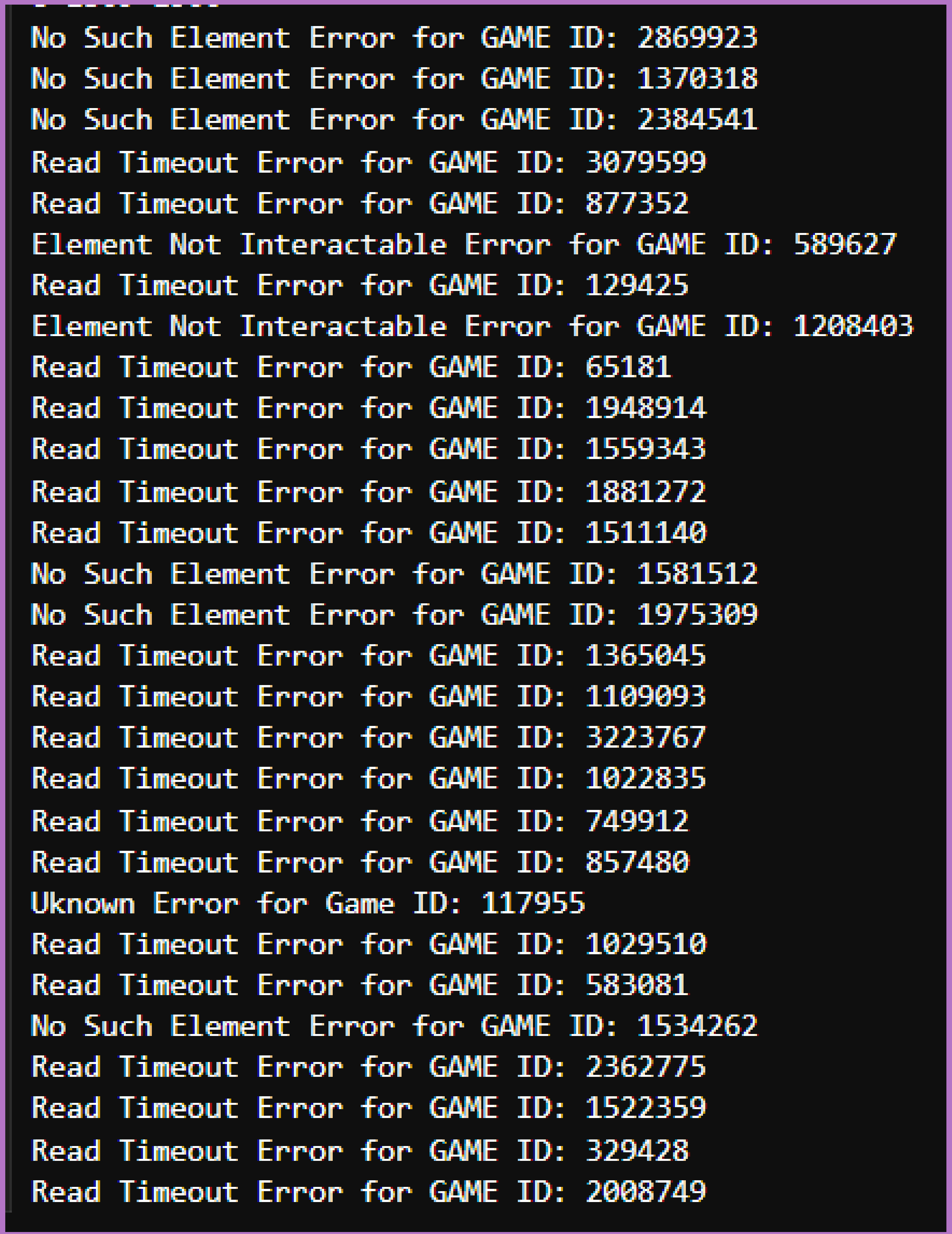
02 The game site contains the status, rating, rating count, tags, average session time, and platforms features.

03 Multithreading was implemented to instantiate multiple web drivers for faster scraping.

04 Save the Game Site Info features into a pandas DataFrame.



```
except NoSuchElementException:
    print("No Such Element Error for GAME ID:", game_id)
    data.extend([game_id, "N/A", "N/A", "N/A", "N/A", "N/A", "N/A"])
    more_info = np.vstack([more_info, data])
    continue

except ElementNotInteractableException:
    print("Element Not Interactable Error for GAME ID:", game_id)
    data.extend([game_id, "N/A", "N/A", "N/A", "N/A", "N/A", "N/A"])
    more_info = np.vstack([more_info, data])
    continue

except ReadTimeoutError:
    print("Read Timeout Error for GAME ID:", game_id)
    data.extend([game_id, "N/A", "N/A", "N/A", "N/A", "N/A", "N/A"])
    more_info = np.vstack([more_info, data])
    continue

except TimeoutException:
    print("Timeout Error for GAME ID:", game_id)
    data.extend([game_id, "N/A", "N/A", "N/A", "N/A", "N/A", "N/A"])
    more_info = np.vstack([more_info, data])
    continue

except Exception:
    print("Uknown Error for Game ID:", game_id)
    data.extend([game_id, "N/A", "N/A", "N/A", "N/A", "N/A", "N/A"])
    more_info = np.vstack([more_info, data])
    continue
```



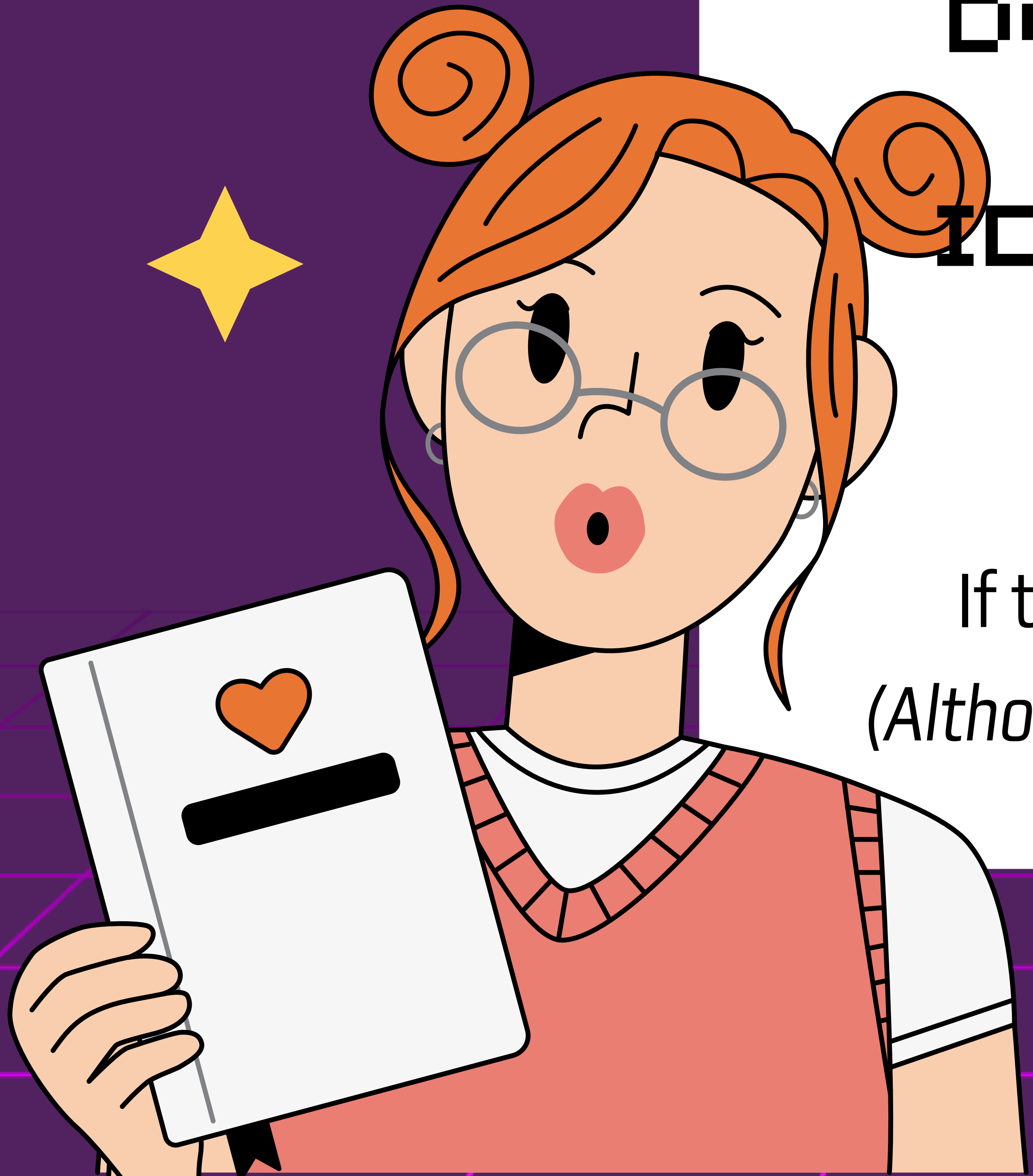
```
No Such Element Error for GAME ID: 2869923
No Such Element Error for GAME ID: 1370318
No Such Element Error for GAME ID: 2384541
Read Timeout Error for GAME ID: 3079599
Read Timeout Error for GAME ID: 877352
Element Not Interactable Error for GAME ID: 589627
Read Timeout Error for GAME ID: 129425
Element Not Interactable Error for GAME ID: 1208403
Read Timeout Error for GAME ID: 65181
Read Timeout Error for GAME ID: 1948914
Read Timeout Error for GAME ID: 1559343
Read Timeout Error for GAME ID: 1881272
Read Timeout Error for GAME ID: 1511140
No Such Element Error for GAME ID: 1581512
No Such Element Error for GAME ID: 1975309
Read Timeout Error for GAME ID: 1365045
Read Timeout Error for GAME ID: 1109093
Read Timeout Error for GAME ID: 3223767
Read Timeout Error for GAME ID: 1022835
Read Timeout Error for GAME ID: 749912
Read Timeout Error for GAME ID: 857480
Uknown Error for Game ID: 117955
Read Timeout Error for GAME ID: 1029510
Read Timeout Error for GAME ID: 583081
No Such Element Error for GAME ID: 1534262
Read Timeout Error for GAME ID: 2362775
Read Timeout Error for GAME ID: 1522359
Read Timeout Error for GAME ID: 329428
Read Timeout Error for GAME ID: 2008749
```

4. MANUAL CHECKING OF ERROR GAME SITES

01 Some game sites had errors during scraping. These errors are due to slow connection, page timeout, and unexpected website format.

02 These errors are inevitable so manual checking is done.

03 Concatenate any retrieved information to the previous data frame.



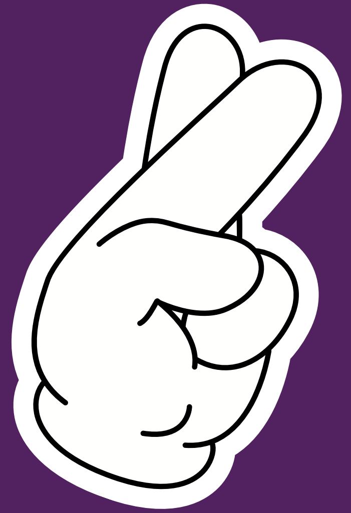
DOES THE DATA COLLECTED CONTAIN PERSONALLY IDENTIFIABLE INFORMATION (PII)?

Author: Low Risk

If the author name indicated is a real person’s full name

(Although a studio name or a pseudonym is not considered PII)

OTHER LEARNINGS



01

Data Quality Considerations

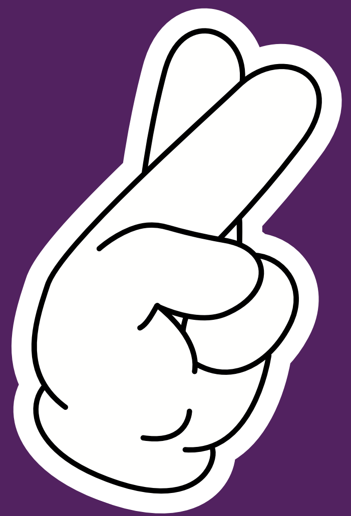
Some games might lack genre, game text, and other data. We decided to apply a placeholder imputation of “N/A”

02

Scraping Errors

While scraping the game sites, there are a lot of unexpected errors. Solving and catching these errors is difficult and may sometimes need manual checking.

OTHER LEARNINGS



03

Multithreaded Scraping

Scraping the game sites is a time-consuming process because either the connection is slow or the page takes too long to load. To make the scraping faster, multithreading is implemented. Although parallelism is not implemented, concurrency is enough to make the scraping a bit faster.

THANK YOU FOR PLAYING!