1.JSON Response:

Trying to retrieve the JSON response from the Power BI dashboard using Chrome DevTools, but it appears to be encrypted and requires admin access.

2.Selenium:

Alternatively, I'm using selenium to extract the data successfully for one dashboard.

This is the script I've worked on so far, but it's returning data in an unstructured format.

I'm successfully extracting data using selenium from the powerBI dashboard URL, but the data is currently in an unstructured format.

Currently I'm figuring out on handling pagination within the dashboard and iterating the script for each date.

from selenium import webdriver

from selenium.webdriver.chrome.service import Service

from selenium.webdriver.common.by import By

from selenium.webdriver.common.keys import Keys

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as EC

from webdriver\_manager.chrome import ChromeDriverManager

import pandas as pd

import time

options = webdriver.ChromeOptions()

driver = webdriver.Chrome(service=Service(ChromeDriverManager().install()), options=options)

oecd\_url = "https://www.oecd.org/en/data/tools/beps-mli-matching-database.html"

driver.get(oecd\_url)

wait = WebDriverWait(driver, 30)

try:

iframe = wait.until(EC.presence\_of\_element\_located((By.TAG\_NAME, "iframe")))

driver.switch\_to.frame(iframe)

print(" Switched to Power BI iframe.")

except:

print(" No iframe found! Ensure the Power BI dashboard is embedded here.")

driver.quit()

exit()

possible\_xpaths = [

'//\*[@id="30db48be-c1c9-9dd7-ce18-7924e28c0b46"]',

"//input[@class='date-slicer-datepicker']",

"//input[@type='text' and contains(@aria-label, 'End date')]",

"//input[@type='text']",

]

date\_picker = None

for xpath in possible\_xpaths:

try:

date\_picker = wait.until(EC.presence\_of\_element\_located((By.XPATH, xpath)))

print(f" Date Picker Found Using XPath: {xpath}")

break

except:

continue

if not date\_picker:

print(" Date Picker not found, check XPath manually.")

driver.quit()

exit()

driver.execute\_script("arguments[0].click();", date\_picker)

dates = ["3/3/2025"]

all\_data = []

for date in dates:

try:

date\_picker.clear()

date\_picker.send\_keys(date)

date\_picker.send\_keys(Keys.RETURN)

time.sleep(5)

time.sleep(10)

data\_div\_xpath = '//\*[@id="pvExplorationHost"]/div/div/exploration/div/explore-canvas/div/div[2]/div/div[2]/div[2]/visual-container-repeat/visual-container-group[2]/transform/div/div[2]/visual-container-group/transform/div/div[2]/visual-container[4]/transform/div/div[3]/div/div/visual-modern/div/div/div[2]/div[1]'

try:

data\_container = wait.until(EC.presence\_of\_element\_located((By.XPATH, data\_div\_xpath)))

extracted\_text = data\_container.text.strip()

print(f" Extracted Data: {extracted\_text}")

all\_data.append([date, extracted\_text])

except Exception as e:

print(f" Failed to extract data for {date}: {e}")

except Exception as e:

print(f"Error processing date {date}: {e}")

df = pd.DataFrame(all\_data, columns=["Date", "Extracted Data"])

df.to\_csv("PowerBI\_Extracted\_Data.csv", index=False)

print(" Scraping Completed Successfully! Data saved to PowerBI\_Extracted\_Data1.csv")

driver.quit()

3.Playwright:

from selectolax.parser import HTMLParser

from playwright.sync\_api import sync\_playwright

def extractBody(url):

    with sync\_playwright() as p:

        browser = p.chromium.launch(headless=True)

        page = browser.new\_page()

        page.goto(url)

        TIMEOUT = 30000

        page.wait\_for\_load\_state("networkidle", timeout=TIMEOUT)

        locator = page.locator("h2.cmp-title\_\_text")

        locator.wait\_for(state="visible", timeout=TIMEOUT)

        return page.inner\_html("body")

def extractBudget(html):

    tree = HTMLParser(html)

    budgetDivs = tree.css("h2.cmp-title\_\_text")

    budgetData = []

    for div in budgetDivs:

        budgetData.append(div.text())

    return budgetData

if \_\_name\_\_ == "\_\_main\_\_":

    url = "https://www.oecd.org/en/data/tools/beps-mli-matching-database.html"

    html = extractBody(url)

    budgetData = extractBudget(html)

    for data in budgetData:

        print(data)