**Eurorepo Webscraper Analysis**

The webscraping script for Eurorepo uses Selenium to open the website and press the download button for the database. It then converts the resulting excel file into a csv file.

Script:

try:  
 from selenium import webdriver  
 from selenium.webdriver.common.by import By  
 from selenium.webdriver.support.ui import WebDriverWait  
 from selenium.webdriver.support import expected\_conditions  
 from selenium.webdriver.chrome.service import Service  
 from selenium.webdriver.chrome.options import Options  
 import time  
 import os  
  
 import pandas as pd  
  
 # REPLACE  
 download\_dir = '/Users/scrambledmacbook/PycharmProjects/WebScraping/Result'  
 url = '<https://database.eurepoc-dashboard.eu/>'  
 # REPLACE  
 service = Service('/Users/scrambledmacbook/PycharmProjects/WebScraping/chromedriver-mac-x64/chromedriver')  
 resultSize = len(os.listdir(download\_dir))  
 chrome\_options = Options()  
  
 prefs = {  
 "download.default\_directory": download\_dir,  
 "download.prompt\_for\_download": False,  
 "download.directory\_upgrade": True,  
 "safebrowsing.enabled": True  
 }  
  
 chrome\_options.add\_experimental\_option('prefs', prefs)  
  
 driver = webdriver.Chrome(service=service, options=chrome\_options)  
 driver.get(url)  
 print("got url")  
 time.sleep(5)  
  
 # Wait for the download button to be clickable and scroll to it  
 downloadButton = WebDriverWait(driver, 20).until(  
 expected\_conditions.element\_to\_be\_clickable((By.XPATH, '//\*[@id="download-button"]'))  
 )  
 print("found download button")  
 driver.execute\_script("arguments[0].scrollIntoView({behavior: 'smooth', block: 'center'});", downloadButton)  
 print("scrolled to download button")  
  
 # Wait for the download button to be visible and click it.  
 time.sleep(3)  
 downloadButton.click()  
 print("clicked download button")  
  
 # Wait for the file to download  
 time.sleep(20)  
 print('File is downloaded')  
  
 # find Excel file in Result  
 for file in os.listdir(download\_dir):  
 if file.endswith('.xlsx'):  
 excel\_file = file  
 break  
  
 # Read Excel file  
 df = pd.read\_excel(os.path.join(download\_dir, excel\_file))  
  
 # Convert to CSV  
 csv\_file = excel\_file.replace('.xlsx', '.csv')  
 df.to\_csv(os.path.join(download\_dir, csv\_file), index=False)  
  
 print(f'Converted {excel\_file} to {csv\_file}')  
except Exception as e:  
 print("Error: ", e)  
  
finally:  
 driver.quit()

Problems:

* Certain files used in the code are referenced based on their location in the machine used to write the code. These locations will have to change depending on where the code is stored.
* Selenium requires Chromedriver to be installed and its location referenced in the code.
* The script takes around a minute to execute.
* The csv file seems to be different from the one on Teams, likely because they were extracted from different parts of the Eurorepo website. There does not seem to be information missing, however. Further analysis is required.

Possible Solutions:

* Dynamically reference files.
* Dynamically set wait times depending on if a condition is met.
* Analyze the differences in the csv files and whether they are significant.