Web Scraping Code Document (Selenium)

INTRODUCTION

This document provides a detailed overview of the web scraping scripts developed using Selenium for two suppliers: Kossen and Verschuren. The scripts are designed to automate the data extraction process, gathering relevant tire price and availability information from the suppliers' websites.

SUPPLIER 1: KOSSEN

Script Overview The script for supplier Kossen automates the process of logging into the Kossen website, navigating to the tire section, and extracting tire prices and other relevant details.

Steps to Execute the Script

1. **Install Selenium:**

   `pip install selenium.`

2. **Download Chrome WebDriver:** Download the Chrome WebDriver from here and place it in your system PATH.
3. **Credentials:** `'username'` and `'password'` in the script with your actual Kossen website credentials.
4. **Run the Script:** Execute the script to start the data extraction process.

ocker-compose.yml .gitlab-ci.yml Kossen.py

```python
driver.get('https://www.trucktyres.com/home-nl/')
time.sleep(10)

username_selector = "div.topbar > div > div.login > form > input[type=text]:nth-child(1)"
username_field = WebDriverWait(driver, timeout: 10).until(
    EC.element_to_be_clickable((By.CSS_SELECTOR, username_selector))
)
username_field.send_keys('tyresupport')

password_selector = "div.topbar > div > div.login > form > input[type=password]:nth-child(2)"
password_field = WebDriverWait(driver, timeout: 10).until(
    EC.element_to_be_clickable((By.CSS_SELECTOR, password_selector))
)
password_field.send_keys('venlo')

login_button_selector = "div.topbar > div > div.login > form > button"
login_button = WebDriverWait(driver, timeout: 10).until(
    EC.element_to_be_clickable((By.CSS_SELECTOR, login_button_selector))
)
login_button.click()
time.sleep(30)

driver.get('https://www.trucktyres.com/catalogus/?/0/')
WebDriverWait(driver, timeout: 10).until(EC.presence_of_element_located((By.CSS_SELECTOR, "div.midwrapper")))
rows = driver.find_elements(By.CSS_SELECTOR, value: "div.division_75 > div > table > tbody > tr")
data = []

for row in rows:
    cols = row.find_elements(By.TAG_NAME, value: 'td')
    if cols:
        data.append([
            cols[0].text,
            cols[1].text,
            cols[2].text,
            cols[3].text,
            cols[4].text,
            cols[1].find_element(By.TAG_NAME, value: 'a').get_attribute('href')
        ])

df = pd.DataFrame(data, columns=['Merk', 'Omschrijving', 'Afmetingen', 'Prijs', 'Voorraad', 'Link'])
```

SUPPLIER 2 VERSCHUREN

Script Overview The script for supplier Verschuren automates the login process, navigates to the tire section, and scrapes tire prices and other details from the supplier's website.

Steps to Execute the Script

1. **Install Selenium:**

   ```
   pip install selenium
   ```

2. **Download Chrome WebDriver:** Download the Chrome WebDriver from here and place it in your system PATH.
3. **Credentials:** Replace `'username'` and `'password'` in the script with your actual Verschuren website credentials.
4. **Run the Script:** Execute the script to start the data extraction process.

ker-compose.yml | .gitlab-ci.yml | Kossen.py | Verschuren.py

```python
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.common.by import By
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.chrome.service import Service
from webdriver_manager.chrome import ChromeDriverManager
import pandas as pd
import time
```

docker-compose.yml | .gitlab-ci.yml | Kossen.py | Verschuren.py

```python
    # Prepare to search
    wait.until(EC.presence_of_element_located((By.CSS_SELECTOR, "input.product-search")))
    search_box = driver.find_element(By.CSS_SELECTOR, value: "input.product-search")

    # DataFrame to store all scraped data
    all_data = []

    # Function to scrape the current page
    AmmarAlmasri-Henhous
    def scrape_current_page():
        rows = driver.find_elements(By.CSS_SELECTOR, value: "table.product-search-results tbody tr.showRowInPage")
        page_data = []
        for row in rows:
            cols = row.find_elements(By.TAG_NAME, value: "td")
            if cols:
                order_button = cols[6].find_element(By.TAG_NAME, value: 'a')
                price = order_button.get_attribute('data-product-price')
                page_data.append({
                    'Maat': cols[0].text,
                    'Merk': cols[1].text,
                    'Type': cols[2].text,
                    'Voorraad': cols[3].text,
                    'Prijs': price
                })
        return page_data

    # Search and scrape data for each term
    for term in search_terms:
        search_box.clear()
        search_box.send_keys(term)
        search_box.send_keys(Keys.RETURN)
        time.sleep(10)  # Wait for the results to load

        while True:
            all_data.extend(scrape_current_page())
            next_page_button = driver.find_elements(By.CSS_SELECTOR, value: "li.navarrow.arrowRight.showArrow")
            if not next_page_button or not next_page_button[0].is_enabled():
                break
            next_page_button[0].click()
            time.sleep(2)  # Ensure the next page has loaded
    driver.quit()
    return all_data
```

Fetching Documentation...