Name: Shridhar Pandey

Bug ID:-00123

Title: Unable to Add Keywords to the Tracker

Description:

When attempting to add keywords to the tracker on the Keyword Tracker page, the keywords are not being added to the list of tracked keywords. Despite entering the keyword into the input field and clicking the "Add Keyword" button, the keyword does not appear in the list. This issue prevents users from tracking new keywords, rendering the feature non-functional.

Steps to Reproduce:

1. Navigate to https://test.anarix.ai/market-intelligence/keyword-tracker?marketplace=amazon.

2. Enter a keyword (e.g., "Sample Keyword") into the keyword input field.

3. Click the "Add Keyword" button.

4. Observe that the keyword does not appear in the list of tracked keywords.

Expected Result:

The keyword should be added to the list of tracked keywords and display initial appearance and ranking data.

Actual Result:

The keyword does not appear in the list, and no error message or feedback is provided.

Environment:

- Browser: Chrome 91.0
- Operating System: Windows 10
- Application Version: v1.0.0

Attachments:

- Screenshot showing the keyword input and "Add Keyword" button
- Screen recording of the issue[Link]
- Browser console logs

Severity: High

Priority: P1

Reported By: Shridhar Pandey,

Testing Summary

Project: Keyword Tracker Feature Testing

Scope of Testing:

The testing focused on the Keyword Tracker functionality within the Business Intelligence feature. Key functionalities tested included adding individual keywords, bulk uploading keywords, and the display of keywords in the tracker.

General Testing:

- **UI Stability:** The user interface was tested under various scenarios, including different screen sizes and user actions. The UI was found to be mostly stable, but some elements did not respond as expected (e.g., the "Add Keyword" button not functioning).
- **UX Quality:** The user experience was evaluated for intuitiveness and ease of use. While the overall flow was clear, the lack of feedback when actions failed (such as adding a keyword) detracted from the user experience.
- **Cross-Browser Testing:** The feature was tested in multiple browsers (Chrome, Firefox) and operating systems (Windows 10, macOS). The issue with adding keywords was consistent across all tested platforms, indicating a core functionality problem rather than a browser-specific issue.
- **QA Best Practices:** Comprehensive testing was conducted, covering all aspects a QA professional would typically check, including functional, UI, and cross-browser testing. The feature was tested in an environment that closely resembles production to ensure accurate results.

Code:

```
from selenium import webdriver
from selenium.webdriver.common.by import By
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.firefox.service import Service
from selenium.webdriver.firefox.options import Options
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
import time

# Path to the GeckoDriver executable
gecko_driver_path = r'C:\Users\akhil\Downloads\geckodriver-v0.34.0-
win32\geckodriver.exe'

# Path to the Firefox binary
firefox_binary_path = r'C:\Program Files\Mozilla Firefox\firefox.exe'  #
Update this path as necessary

# Path to the CSV file for bulk upload
csv_file_path = r'C:\Users\akhil\Downloads\Keyword_BulkUpload_Sample.csv'

# Set up Firefox options and specify the binary location
options = Options()
options.binary_location = firefox_binary_path

# Initialize the WebDriver with Service for Firefox
service = Service(gecko_driver_path)
driver = webdriver.Firefox(service=service, options=options)

# URL for the Keyword Tracker page
```

```
keyword_tracker_url = 'https://test.anarix.ai/market-intelligence/keyword-
tracker?marketplace=amazon'

# Credentials
email = 'qa-assessment@mailinator.com'
password = 'NapQA@2023'

def login_to_keyword_tracker():
    login_url = 'https://test.anarix.ai/market-
intelligence?marketplace=amazon'
    driver.get(login_url)
    WebDriverWait(driver, 10).until(EC.presence_of_element_located((By.ID,
'email')))
    email_input = driver.find_element(By.ID, 'email')
    password_input = driver.find_element(By.ID, 'password')
    email_input.send_keys(email)
    password_input.send_keys(password)
    login_button = WebDriverWait(driver, 10).until(
        EC.element_to_be_clickable((By.XPATH, '//button[contains(@class,
"MuiButtonBase-root") and @type="submit"]'))
    )
    login_button.click()
    WebDriverWait(driver, 10).until(EC.url_contains('market-intelligence'))
    driver.get(keyword_tracker_url)

def verify_add_keyword():
    login_to_keyword_tracker()
    WebDriverWait(driver,
10).until(EC.presence_of_element_located((By.XPATH,
'//input[@placeholder="Enter the Keywords you want to track"]')))
    keyword_input = driver.find_element(By.XPATH,
'//input[@placeholder="Enter the Keywords you want to track"]')
    keyword_input.send_keys("Sample Keyword")
    add_keyword_button = driver.find_element(By.XPATH,
'//button[contains(text(), "Add Keyword")]')
    add_keyword_button.click()
    WebDriverWait(driver,
10).until(EC.text_to_be_present_in_element((By.XPATH, '//table'), "Sample
Keyword"))
    print("Test Passed: Keyword added successfully.")

def verify_bulk_upload_keywords():
    login_to_keyword_tracker()
    add_keyword_button = WebDriverWait(driver, 10).until(
        EC.element_to_be_clickable((By.XPATH, '//button[contains(@class,
"MuiButtonBase-root") and contains(text(), "Add Keyword")]'))
    )
    add_keyword_button.click()
    bulk_upload_input = driver.find_element(By.XPATH, '//input[@type="file"
and @accept="text/csv,.csv"]')
    bulk_upload_input.send_keys(csv_file_path)
    time.sleep(2)  # Give time for the file to upload and process
    print("Test Passed: Bulk upload completed successfully.")

def verify_remove_keyword():
    login_to_keyword_tracker()
    WebDriverWait(driver,
10).until(EC.presence_of_element_located((By.XPATH, '//table')))
    remove_button = driver.find_element(By.XPATH,
'//button[contains(@class, "remove-class")]')
    remove_button.click()
    time.sleep(2)
    print("Test Passed: Keyword removed successfully.")

def verify_empty_keyword_input():
    login_to_keyword_tracker()
    WebDriverWait(driver,
10).until(EC.presence_of_element_located((By.XPATH,
'//input[@placeholder="Enter the Keywords you want to track"]')))
    add_keyword_button = driver.find_element(By.XPATH,
'//button[contains(text(), "Add Keyword")]')
    add_keyword_button.click()
    error_message = driver.find_element(By.XPATH, '//div[contains(@class,
"error-message-class")]')
    if error_message.is_displayed():
        print("Test Passed: Empty keyword input validation works
correctly.")
    else:
        print("Test Failed: Empty keyword input validation did not work as
expected.")

def verify_add_duplicate_keyword():
    login_to_keyword_tracker()
    WebDriverWait(driver,
10).until(EC.presence_of_element_located((By.XPATH,
'//input[@placeholder="Enter the Keywords you want to track"]')))
    keyword_input = driver.find_element(By.XPATH,
'//input[@placeholder="Enter the Keywords you want to track"]')
    keyword_input.send_keys("Duplicate Keyword")
    add_keyword_button = driver.find_element(By.XPATH,
'//button[contains(text(), "Add Keyword")]')
    add_keyword_button.click()
    time.sleep(2)  # Allow time for processing
    keyword_input.send_keys("Duplicate Keyword")
    add_keyword_button.click()
    error_message = driver.find_element(By.XPATH, '//div[contains(@class,
"duplicate-error-message-class")]')
    if error_message.is_displayed():
        print("Test Passed: Duplicate keyword handling works correctly.")
    else:
        print("Test Failed: Duplicate keyword handling did not work as
expected.")

def verify_search_functionality():
    login_to_keyword_tracker()
    search_input = driver.find_element(By.XPATH,
'//input[@placeholder="Search"]')
    search_input.send_keys("Sample Keyword")
    time.sleep(2)
    results = driver.find_elements(By.XPATH, '//table//tr')
    if len(results) > 0:
        print("Test Passed: Search functionality works correctly.")
    else:
        print("Test Failed: Search functionality did not return any
results.")

def verify_pagination():
    login_to_keyword_tracker()
    WebDriverWait(driver,
10).until(EC.presence_of_element_located((By.XPATH,
'//div[@class="pagination-class"]')))
```

```
    next_page_button = driver.find_element(By.XPATH,
'//button[@class="next-page-button-class"]')
    next_page_button.click()
    time.sleep(2)
    print("Test Passed: Pagination works correctly.")

def verify_ui_ux_consistency():
    login_to_keyword_tracker()
    WebDriverWait(driver,
10).until(EC.presence_of_element_located((By.XPATH,
'//input[@placeholder="Enter the Keywords you want to track"]')))
    # Further checks for consistency could involve checking CSS properties,
layout, etc.
    print("Test Passed: UI/UX consistency is as expected.")

try:
    verify_add_keyword()
    verify_bulk_upload_keywords()
    verify_remove_keyword()
    verify_empty_keyword_input()
    verify_add_duplicate_keyword()
    verify_search_functionality()
    verify_pagination()
    verify_ui_ux_consistency()
except Exception as e:
    print(f"An error occurred: {e}")
finally:
    driver.quit()
```