

Name – Rik Halder

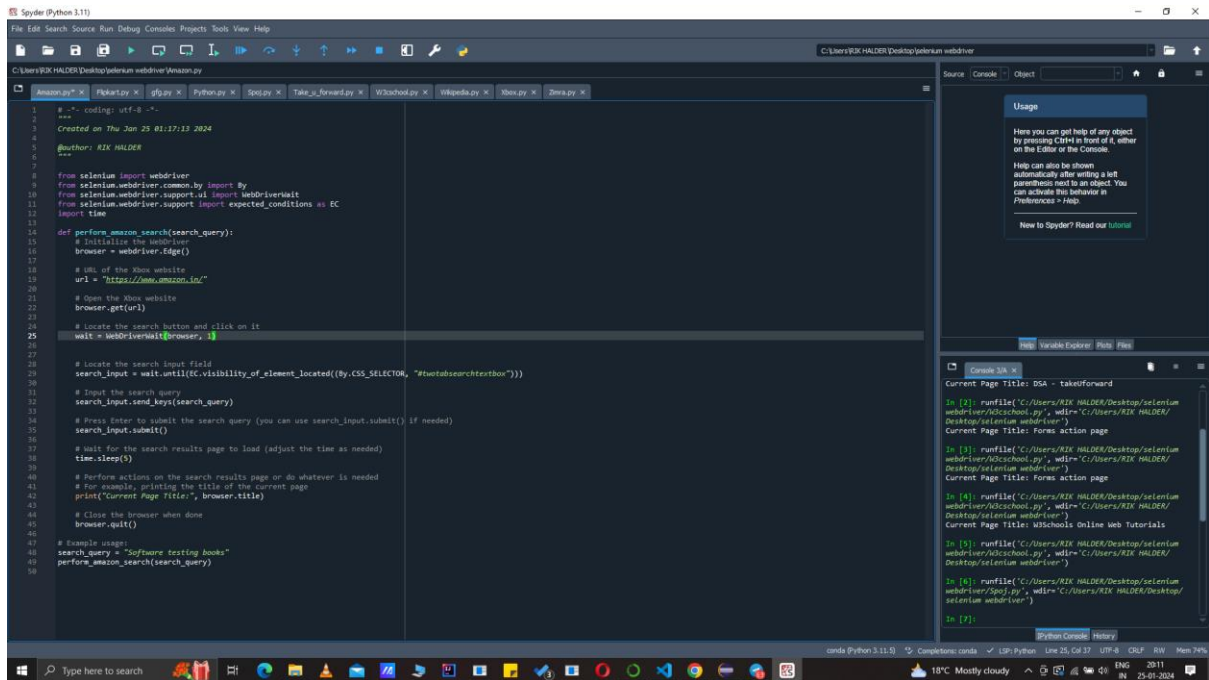
Roll No – 223CS3148

Software Testing Lab

Selenium Assignment 3

Selenium Webdriver Assignment

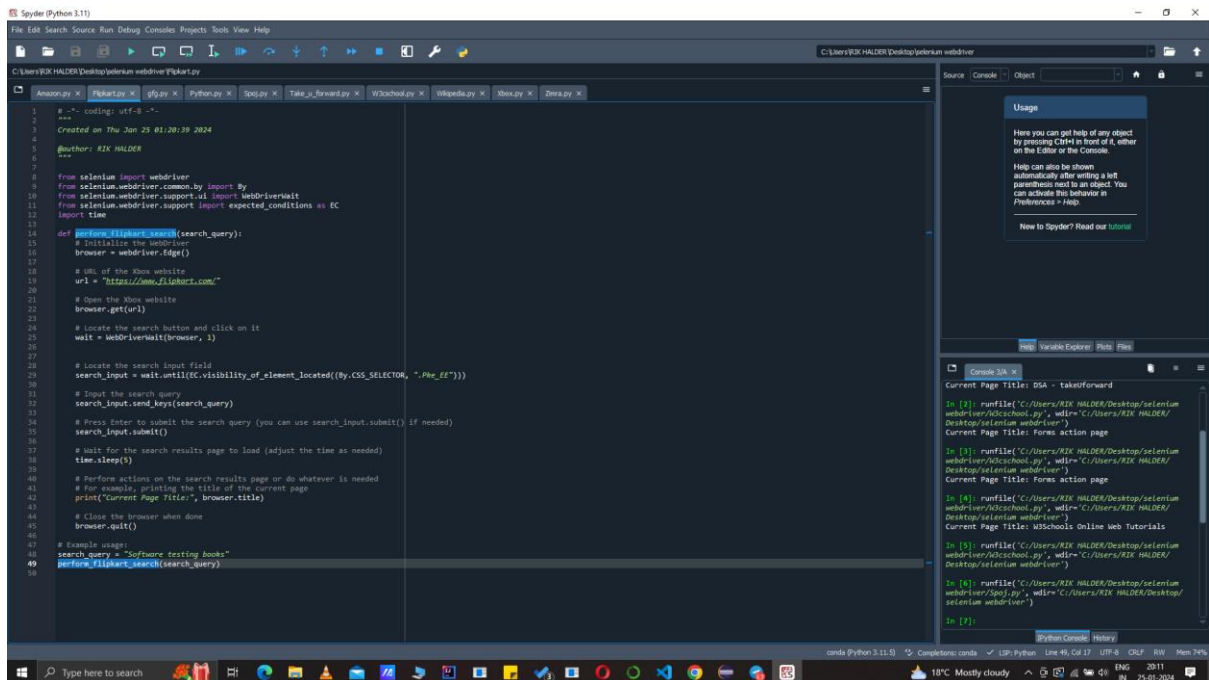
Website 1 – Amazon



The screenshot shows the Spyder Python IDE with a Selenium script for performing a search on Amazon. The script is written in Python and uses Selenium WebDriver to interact with the Amazon website. The script includes comments explaining each step, such as initializing the browser, navigating to the Amazon URL, locating the search button, and performing the search. The script is saved as 'Amazon.py' in the 'C:\Users\RIK HALDER\Desktop\selenium webdriver' directory. The console on the right shows the output of the script, indicating that the search was successful and the page title is 'DSA - takeforward'.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Jan 25 01:17:13 2024
4 @author: RIK HALDER
5 """
6
7 from selenium import webdriver
8 from selenium.webdriver.common.by import By
9 from selenium.webdriver.support.ui import WebDriverWait
10 from selenium.webdriver.support import expected_conditions as EC
11 import time
12
13 def perform_amazon_search(search_query):
14     # Initialize the webdriver
15     browser = webdriver.Chrome()
16
17     # URL of the Xbox website
18     url = "https://www.amazon.in/"
19
20     # Open the Xbox website
21     browser.get(url)
22
23     # Locate the search button and click on it
24     wait = WebDriverWait(browser, 10)
25     search_input = wait.until(EC.visibility_of_element_located((By.CSS_SELECTOR, "#twotabsearchtextbox")))
26
27     # Input the search query
28     search_input.send_keys(search_query)
29
30     # Press Enter to submit the search query (you can use search_input.submit() if needed)
31     search_input.submit()
32
33     # Wait for the search results page to load (adjust the time as needed)
34     time.sleep(5)
35
36     # Perform actions on the search results page or do whatever is needed
37     # For example, printing the title of the current page
38     print("Current Page Title:", browser.title)
39
40     # Close the browser when done
41     browser.quit()
42
43 # Example usage:
44 search_query = "Software testing books"
45 perform_amazon_search(search_query)
```

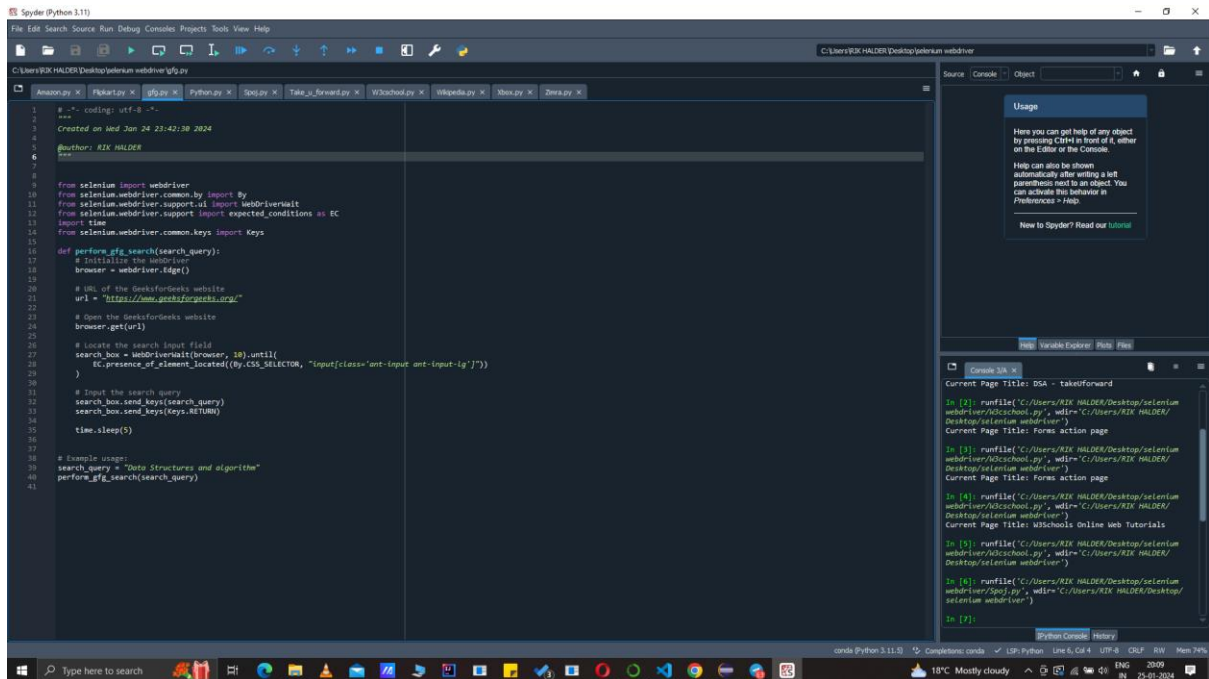
Website 2 – Flipkart



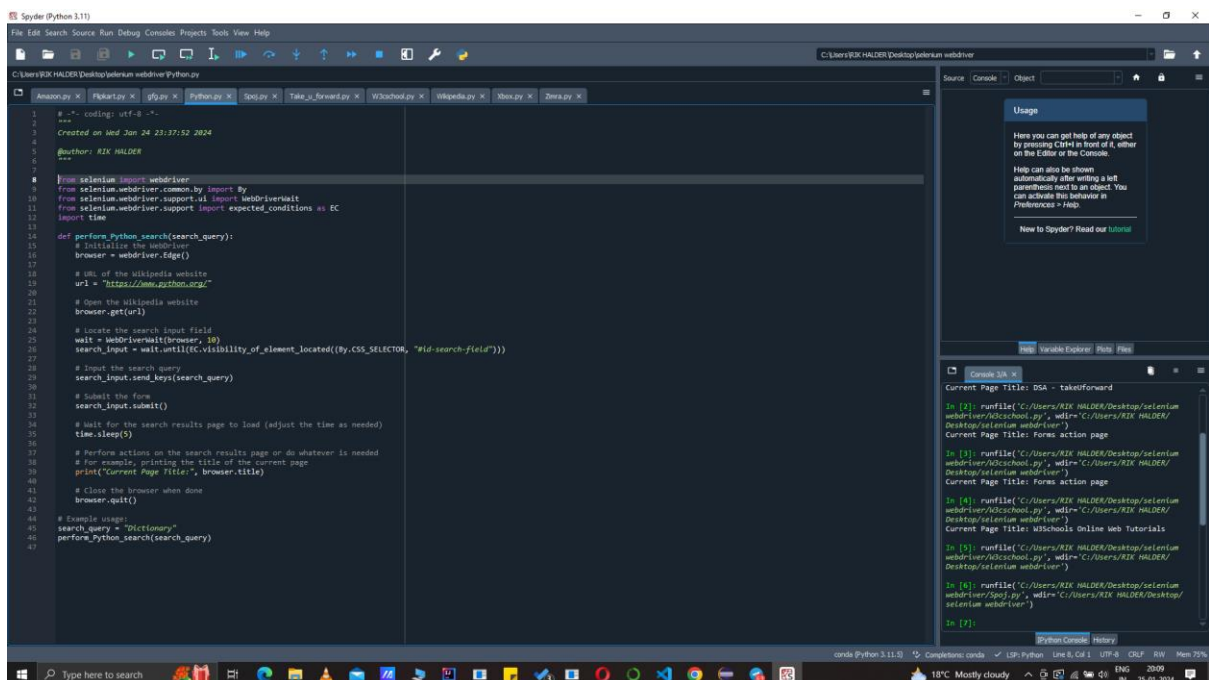
The screenshot shows the Spyder Python IDE with a Selenium script for performing a search on Flipkart. The script is written in Python and uses Selenium WebDriver to interact with the Flipkart website. The script includes comments explaining each step, such as initializing the browser, navigating to the Flipkart URL, locating the search button, and performing the search. The script is saved as 'Amazon.py' in the 'C:\Users\RIK HALDER\Desktop\selenium webdriver' directory. The console on the right shows the output of the script, indicating that the search was successful and the page title is 'DSA - takeforward'.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Jan 25 01:20:39 2024
4 @author: RIK HALDER
5 """
6
7 from selenium import webdriver
8 from selenium.webdriver.common.by import By
9 from selenium.webdriver.support.ui import WebDriverWait
10 from selenium.webdriver.support import expected_conditions as EC
11 import time
12
13 def perform_flipkart_search(search_query):
14     # Initialize the webdriver
15     browser = webdriver.Chrome()
16
17     # URL of the Xbox website
18     url = "https://www.flipkart.com/"
19
20     # Open the Xbox website
21     browser.get(url)
22
23     # Locate the search button and click on it
24     wait = WebDriverWait(browser, 10)
25     search_input = wait.until(EC.visibility_of_element_located((By.CSS_SELECTOR, ".Phe_EE")))
26
27     # Input the search query
28     search_input.send_keys(search_query)
29
30     # Press Enter to submit the search query (you can use search_input.submit() if needed)
31     search_input.submit()
32
33     # Wait for the search results page to load (adjust the time as needed)
34     time.sleep(5)
35
36     # Perform actions on the search results page or do whatever is needed
37     # For example, printing the title of the current page
38     print("Current Page Title:", browser.title)
39
40     # Close the browser when done
41     browser.quit()
42
43 # Example usage:
44 search_query = "Software testing books"
45 perform_flipkart_search(search_query)
```

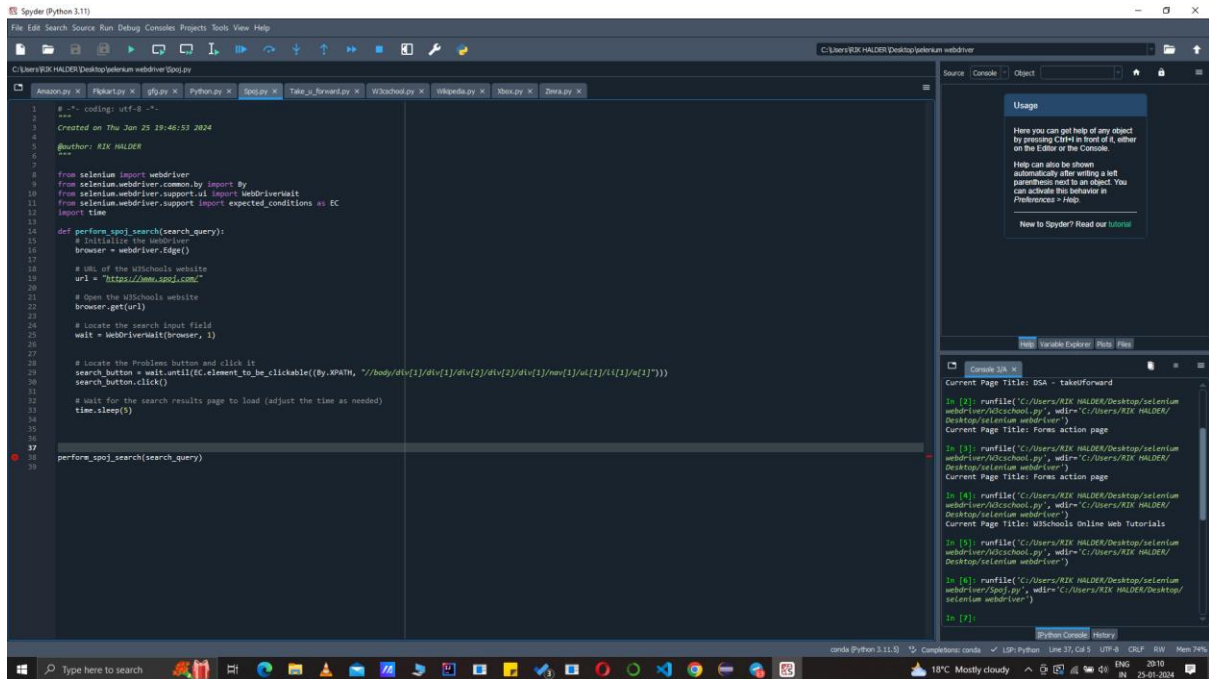
Website 3 – GFG



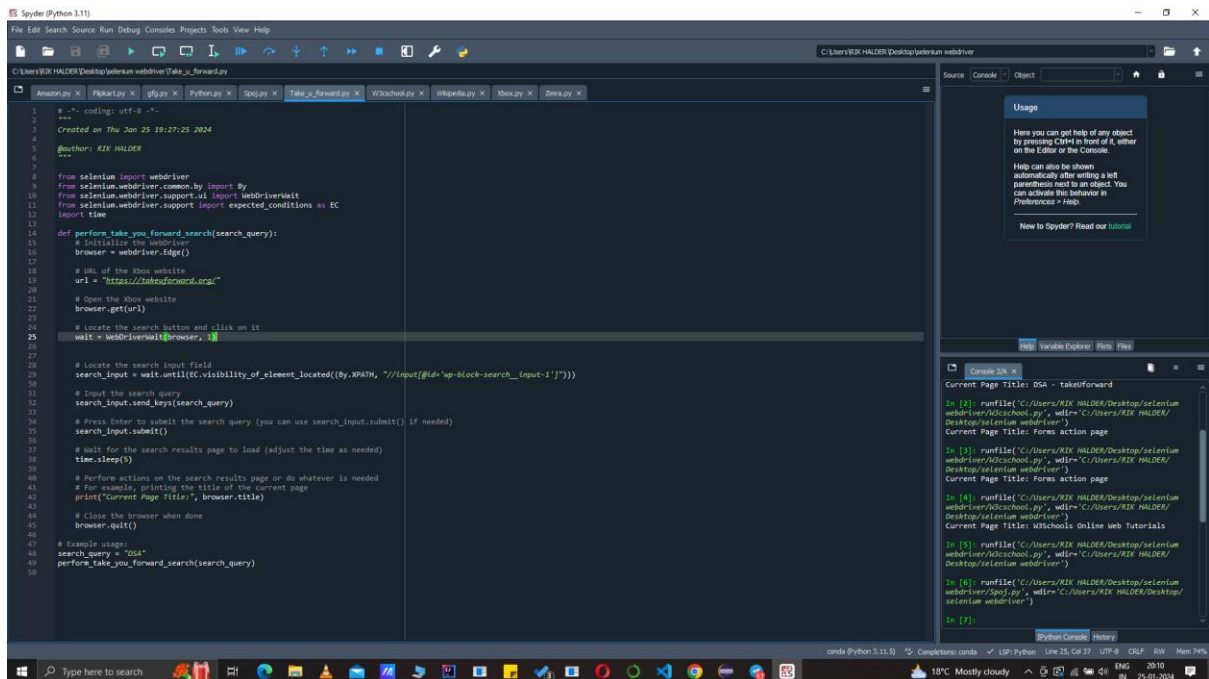
Website 4 – Python



Website 5 – SPOJ



Website 6 – Take_u_Forward



Website 7 – W3CSchool

The screenshot shows the Spyder Python IDE with a Selenium script for searching on W3CSchool. The script is as follows:

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Thu Jan 25 19:19:19 2024
4
5 @author: RIK HALDER
6 """
7
8 from selenium import webdriver
9 from selenium.webdriver.common.by import By
10 from selenium.webdriver.support.ui import WebDriverWait
11 from selenium.webdriver.support import expected_conditions as EC
12 import time
13
14 def perform_w3cschool_search(search_query):
15     # Initialize the webdriver
16     browser = webdriver.Edge()
17
18     # URL of the W3Schools website
19     url = "https://www.w3schools.com/"
20
21     # Open the W3Schools website
22     browser.get(url)
23
24     # Locate the search input field
25     wait = WebDriverWait(browser, 10)
26     search_input = wait.until(EC.visibility_of_element_located((By.XPATH, "//input[@id='search']")))
27
28     # Input the search query
29     search_input.send_keys(search_query)
30
31     # Locate the search button and click it
32     search_button = wait.until(EC.element_to_be_clickable((By.XPATH, "//button[@id='learntocode_searchbtn']")))
33     search_button.click()
34
35     # Wait for the search results page to load (adjust the time as needed)
36     time.sleep(5)
37
38
39
40
41 # Example usage:
42 search_query = "Web Development"
43 perform_w3cschool_search(search_query)
```

The right sidebar shows the 'Console' tab with the following output:

```
Current Page Title: DSA - takeForward
In [2]: runfile('C:/Users/RIK HALDER/Desktop/selenium/webdriver/w3cschool.py', wdir='C:/Users/RIK HALDER/Desktop/selenium/webdriver')
Current Page Title: Forms action page
In [3]: runfile('C:/Users/RIK HALDER/Desktop/selenium/webdriver/w3cschool.py', wdir='C:/Users/RIK HALDER/Desktop/selenium/webdriver')
Current Page Title: Forms action page
In [4]: runfile('C:/Users/RIK HALDER/Desktop/selenium/webdriver/w3cschool.py', wdir='C:/Users/RIK HALDER/Desktop/selenium/webdriver')
Current Page Title: W3Schools Online Web Tutorials
In [5]: runfile('C:/Users/RIK HALDER/Desktop/selenium/webdriver/w3cschool.py', wdir='C:/Users/RIK HALDER/Desktop/selenium/webdriver')
In [6]: runfile('C:/Users/RIK HALDER/Desktop/selenium/webdriver/w3cschool.py', wdir='C:/Users/RIK HALDER/Desktop/selenium/webdriver')
In [7]:
```

Website 8 – Wikipedia

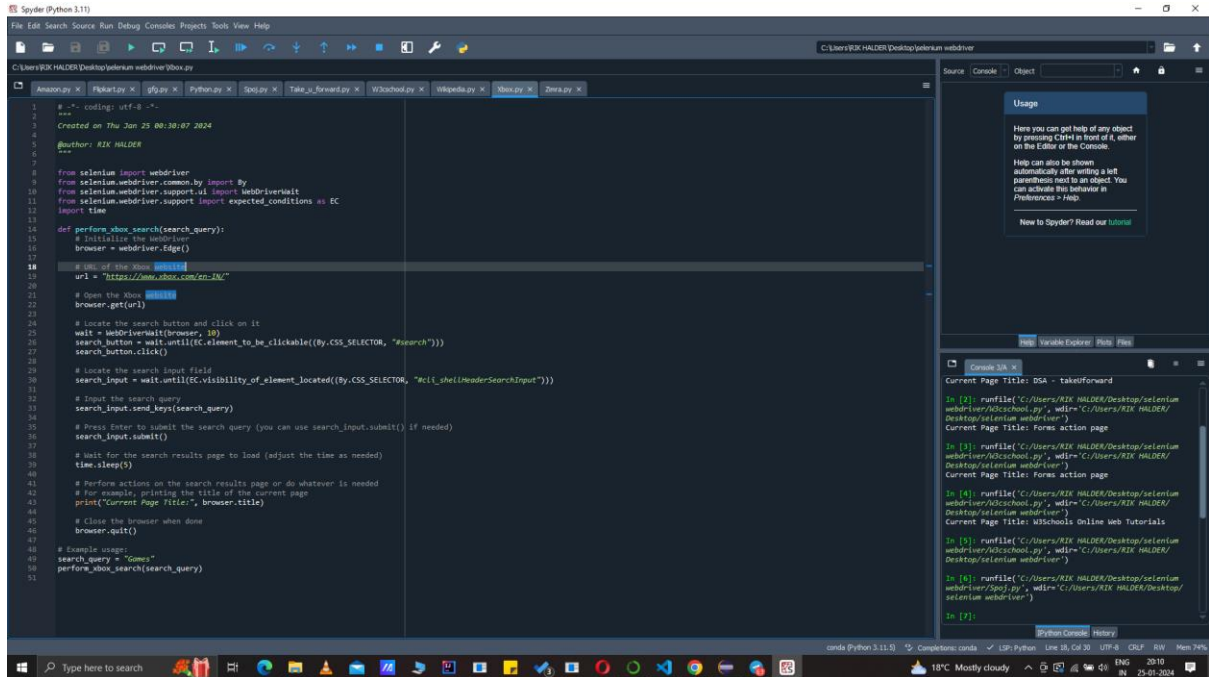
The screenshot shows the Spyder Python IDE with a Selenium script for searching on Wikipedia. The script is as follows:

```
1 from selenium import webdriver
2 from selenium.webdriver.common.by import By
3 from selenium.webdriver.support.ui import WebDriverWait
4 from selenium.webdriver.support import expected_conditions as EC
5 import time
6
7 def perform_wikipedia_search(search_query):
8     # Initialize the webdriver
9     browser = webdriver.Edge()
10
11     # URL of the Wikipedia website
12     url = "https://www.wikipedia.org/"
13
14     # Open the Wikipedia website
15     browser.get(url)
16
17     # Locate the search input field
18     wait = WebDriverWait(browser, 10)
19     search_input = wait.until(EC.visibility_of_element_located((By.CSS_SELECTOR, "input[name='search']")))
20
21     # Input the search query
22     search_input.send_keys(search_query)
23
24     # Submit the form
25     search_input.submit()
26
27     # Wait for the search results page to load (adjust the time as needed)
28     time.sleep(5)
29
30     # Perform actions on the search results page or do whatever is needed
31     # For example, printing the title of the current page
32     print("Current Page Title:", browser.title)
33
34     # Close the browser when done
35     browser.quit()
36
37 # Example usage:
38 search_query = "Software testing"
39 perform_wikipedia_search(search_query)
```

The right sidebar shows the 'Console' tab with the following output:

```
Current Page Title: DSA - takeForward
In [2]: runfile('C:/Users/RIK HALDER/Desktop/selenium/webdriver/w3cschool.py', wdir='C:/Users/RIK HALDER/Desktop/selenium/webdriver')
Current Page Title: Forms action page
In [3]: runfile('C:/Users/RIK HALDER/Desktop/selenium/webdriver/w3cschool.py', wdir='C:/Users/RIK HALDER/Desktop/selenium/webdriver')
Current Page Title: Forms action page
In [4]: runfile('C:/Users/RIK HALDER/Desktop/selenium/webdriver/w3cschool.py', wdir='C:/Users/RIK HALDER/Desktop/selenium/webdriver')
Current Page Title: W3Schools Online Web Tutorials
In [5]: runfile('C:/Users/RIK HALDER/Desktop/selenium/webdriver/w3cschool.py', wdir='C:/Users/RIK HALDER/Desktop/selenium/webdriver')
In [6]: runfile('C:/Users/RIK HALDER/Desktop/selenium/webdriver/w3cschool.py', wdir='C:/Users/RIK HALDER/Desktop/selenium/webdriver')
In [7]:
```

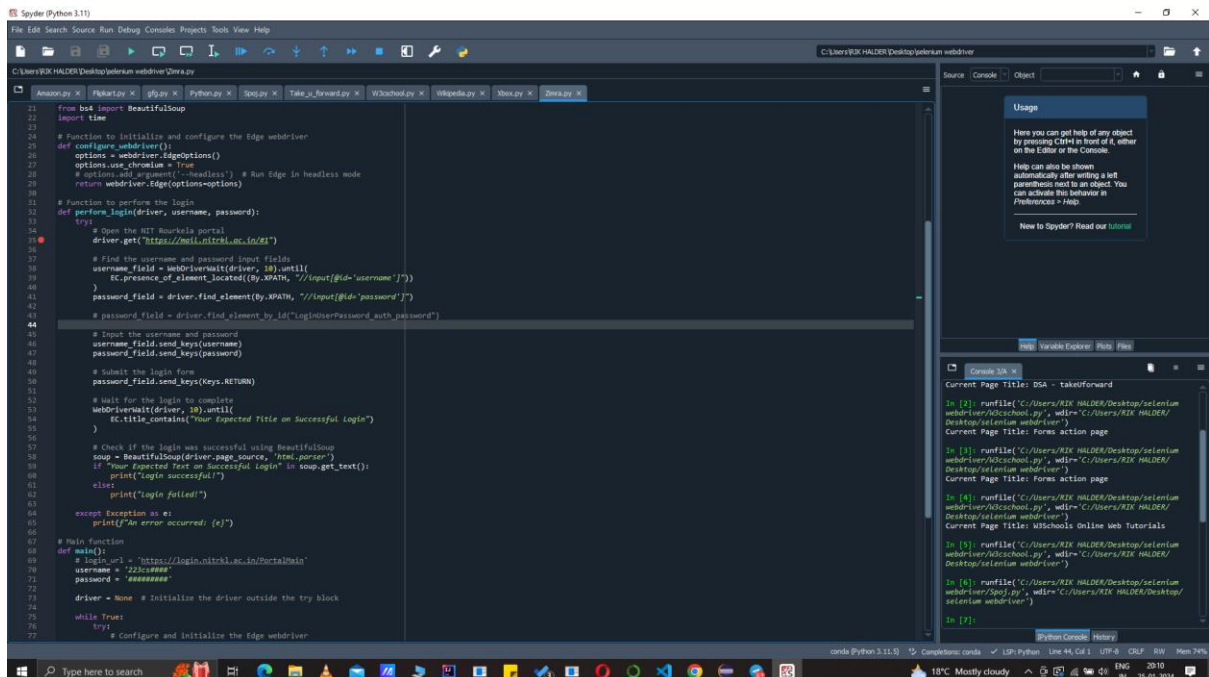

Website 9 – XBOX



The screenshot shows the Spyder Python IDE with a Selenium script for performing a search on the XBOX website. The script is located at C:\Users\RIK HALDER\Desktop\selenium\webdriver\ Xbox.py. The code includes imports for Selenium WebDriver, WebDriverWait, ExpectedConditions, and time. It defines a function perform_xbox_search that takes a search query as input. The function initializes a WebDriver, opens the XBOX website, waits for the search button to be clickable, clicks it, waits for the search input field to be visible, enters the search query, and submits it. It then waits for the search results page to load and prints the current page title. The script also includes a main function that prompts the user for a search query and calls the perform_xbox_search function.

```
1 # -*- coding: utf-8 -*-
2 """
3 Created on Jan 23 00:30:07 2024
4
5 @author: RIK HALDER
6 """
7
8 from selenium import webdriver
9 from selenium.webdriver.common.by import By
10 from selenium.webdriver.support.ui import WebDriverWait
11 from selenium.webdriver.support import expected_conditions as EC
12 import time
13
14 def perform_xbox_search(search_query):
15     # Initialize the webdriver
16     browser = webdriver.Edge()
17
18     # URL of the Xbox website
19     url = "https://www.xbox.com/en-IN/"
20
21     # Open the Xbox website
22     browser.get(url)
23
24     # Locate the search button and click on it
25     wait = WebDriverWait(browser, 10)
26     search_button = wait.until(EC.element_to_be_clickable((By.CSS_SELECTOR, "#search")))
27     search_button.click()
28
29     # Locate the search input field
30     search_input = wait.until(EC.visibility_of_element_located((By.CSS_SELECTOR, "#csl_headerSearchInput")))
31
32     # Input the search query
33     search_input.send_keys(search_query)
34
35     # Press Enter to submit the search query (you can use search_input.submit() if needed)
36     search_input.submit()
37
38     # Wait for the search results page to load (adjust the time as needed)
39     time.sleep(5)
40
41     # Perform actions on the search results page or do whatever is needed
42     # For example, printing the title of the current page
43     print("Current Page Title:", browser.title)
44
45     # Close the browser when done
46     browser.quit()
47
48 # Example usage:
49 search_query = "Xbox"
50 perform_xbox_search(search_query)
```

Website 10 – zimra



The screenshot shows the Spyder Python IDE with a BeautifulSoup script for performing a login on the Zimra website. The script is located at C:\Users\RIK HALDER\Desktop\selenium\webdriver\ Zimra.py. The code includes imports for BeautifulSoup, time, and urllib. It defines a function perform_login that takes a driver, username, and password as input. The function opens the Zimra website, finds the username and password input fields, enters the username and password, and submits the login form. It then waits for the login to complete and checks if the login was successful using BeautifulSoup. The script also includes a main function that prompts the user for a username and password and calls the perform_login function.

```
21 from bs4 import BeautifulSoup
22 import time
23
24 # Function to initialize and configure the Edge webdriver
25 def configure_webdriver():
26     options = webdriver.EdgeOptions()
27     options.add_argument("--headless") # Run Edge in headless mode
28     return webdriver.Edge(options=options)
29
30 # Function to perform the login
31 def perform_login(driver, username, password):
32     try:
33         # Open the Zimra website
34         driver.get("https://mola.nitri.ac.in/81")
35
36         # Find the username and password input fields
37         username_field = WebDriverWait(driver, 10).until(
38             EC.presence_of_element_located((By.XPATH, "//input[@id='username']")))
39         password_field = driver.find_element(By.XPATH, "//input[@id='password']")
40
41         # Input the username and password
42         username_field.send_keys(username)
43         password_field.send_keys(password)
44
45         # Submit the login form
46         password_field.send_keys(Keys.RETURN)
47
48         # Wait for the login to complete
49         WebDriverWait(driver, 10).until(
50             EC.title_contains("Your Expected Title on Successful Login"))
51
52         # Check if the login was successful using BeautifulSoup
53         soup = BeautifulSoup(driver.page_source, "html.parser")
54         if "Your Expected Text on Successful Login" in soup.get_text():
55             print("Login successful!")
56         else:
57             print("Login failed!")
58     except Exception as e:
59         print(f"An error occurred: {e}")
60
61 # Main function
62 def main():
63     # Login URL
64     login_url = "https://login.nitri.ac.in/PortalMain"
65     username = "22k22222"
66     password = "password"
67
68     driver = None # Initialize the driver outside the try block
69     while True:
70         try:
71             # Configure and initialize the Edge webdriver
72             driver = configure_webdriver()
73             # Perform login
74             perform_login(driver, username, password)
75         except Exception as e:
76             print(f"An error occurred: {e}")
77             continue
```

