**RESEARCH ON SELENIUM**

**-**

**DONE BY**

**SHWETHA.T**

**INDHIRA SIVASAKTHI**

**WHAT IS SELENIUM?**

Selenium is an open-source tool suite designed for automating web browsers. It provides a range of tools and libraries for web browser automation, allowing users to interact with web applications just as a human would. Selenium supports multiple programming languages like Python, Java, C#, and JavaScript, and can be used with various browsers such as Chrome, Firefox, Edge, Safari, and others.

**WHY SELENIUM USED IN THE PROGRAM:**

In the provided program, Selenium is used for automating the following tasks:

Opening a Web Browser: Selenium launches an Edge browser instance.

Navigating to Web Pages: It directs the browser to open specific URLs, such as Google’s homepage.

Interacting with Web Elements: Selenium finds the search box on Google’s page, types a query, and submits it.

Simulating User Actions: It mimics user actions like clicking on the first search result.

Taking Screenshots: Selenium captures screenshots of the web pages at various stages.

These tasks are automated to perform repetitive or complex browser interactions without manual intervention, which can be particularly useful for testing web applications, scraping web data, or performing routine web tasks.

**SMALL DESCRIPTION OF THE PROJECT:**

This program automates the process of performing a Google search, interacting with the search results, and taking screenshots of the web pages. Here is a step-by-step breakdown of what the program does:

**Set Up and Initialize:**

Uses Selenium to control the Edge browser.

Opens Google’s homepage.

**Perform Google Search:**

Finds the search box on Google's homepage.

Enters the search query "Selenium WebDriver" into the search box.

Submits the search query by simulating pressing the Enter key.

**Wait for Results and Screenshot:**

Waits for a few seconds to ensure the search results page loads completely.

Takes a screenshot of the search results page.

Processes the screenshot by resizing it to 1920x1080 pixels and enhancing its sharpness, then saves the processed image.

**Interact with First Search Result:**

Finds the first search result link.

Clicks on the first search result to navigate to that page.

Waits for a few seconds to ensure the page loads completely.

**Screenshot of First Result Page:**

Takes a screenshot of the first result's page.

Processes this screenshot by resizing and enhancing it, then saves the processed image.

**Clean Up:**

Closes the browser to end the WebDriver session.

**LIBRARY:**

**1. Selenium**

Library: selenium

**Purpose:**

**webdriver**: Manages the web browser automation. Specifically, it allows controlling the Edge browser.

**Service**: Manages the service for the Edge WebDriver, which is necessary to start the browser instance.

**By**: Provides mechanisms to locate elements on a web page using different strategies (e.g., by name, CSS selector).

**Keys**: Simulates keyboard actions, such as pressing the Enter key.

**Usage in Program:**

Launching the Edge browser.

Navigating to web pages.

Finding and interacting with web elements (e.g., entering a search query, clicking links).

Taking screenshots of web pages.

**2. Pillow (PIL)**

Library: Pillow

**Purpose:**

**Image**: Provides capabilities to open, manipulate, and save images.

**ImageEnhance**: Offers tools to enhance images, such as adjusting sharpness, contrast, brightness, and color.

**Usage in Program:**

Opening the screenshots taken by Selenium.

Resizing images to a fixed resolution (1920x1080 pixels) to ensure consistency.

Enhancing the sharpness of images to improve clarity.

Saving the processed images to specified file paths.

**3. Time**

Library: time

**Purpose:**

Provides functions for time-related tasks.

**Usage in Program:**

Introducing delays (time.sleep()) to wait for web pages to load completely before taking screenshots or interacting with elements.

**SAMPLE CODE:**

from selenium import webdriver

from selenium.webdriver.edge.service import Service

from selenium.webdriver.common.by import By

from selenium.webdriver.common.keys import Keys

from PIL import Image, ImageEnhance

import time

# Specify the path to the EdgeDriver executable

driver\_path = r'C:\drivers\edgedriver\_win64\msedgedriver.exe' # Ensure this is the correct path

# Initialize the Edge driver with the specified path using Service

service = Service(driver\_path)

driver = webdriver.Edge(service=service)

# Function to preprocess the screenshot

def preprocess\_image(image\_path, output\_path):

with Image.open(image\_path) as img:

# Set a fixed resolution

fixed\_size = (1920, 1080)

img = img.resize(fixed\_size, Image.LANCZOS)

# Enhance the image clarity

enhancer = ImageEnhance.Sharpness(img)

img = enhancer.enhance(2.0)

# Save the processed image

img.save(output\_path)

try:

# Set a fixed window size for consistency

driver.set\_window\_size(1920, 1080)

# Open Google

driver.get("http://www.google.com")

print("Opened Google")

# Wait for the page to load

time.sleep(5)

print("Waited for the page to load")

# Take a screenshot of the Google homepage

screenshot\_path = r'C:\image\screenshot\_google.png'

processed\_screenshot\_path = r'C:\image\processed\_screenshot\_google.png'

driver.save\_screenshot(screenshot\_path)

preprocess\_image(screenshot\_path, processed\_screenshot\_path)

print(f"Processed screenshot of Google saved to {processed\_screenshot\_path}")

# Find the search box using its name attribute value

search\_box = driver.find\_element(By.NAME, "q")

# Type in the search query

search\_box.send\_keys("Selenium WebDriver")

search\_box.send\_keys(Keys.RETURN)

print("Performed search for 'Selenium WebDriver'")

# Wait for search results to load

time.sleep(5)

# Take a screenshot of the search results page

screenshot\_path = r'C:\image\screenshot\_search\_results.png'

processed\_screenshot\_path = r'C:\image\processed\_screenshot\_search\_results.png'

driver.save\_screenshot(screenshot\_path)

preprocess\_image(screenshot\_path, processed\_screenshot\_path)

print(f"Processed screenshot of search results saved to {processed\_screenshot\_path}")

# Click on the first search result

first\_result = driver.find\_element(By.CSS\_SELECTOR, "h3")

first\_result.click()

print("Clicked on the first search result")

# Wait for the page to load

time.sleep(5)

# Take a screenshot of the first result's page

screenshot\_path = r'C:\image\screenshot\_first\_result.png'

processed\_screenshot\_path = r'C:\image\processed\_screenshot\_first\_result.png'

driver.save\_screenshot(screenshot\_path)

preprocess\_image(screenshot\_path, processed\_screenshot\_path)

print(f"Processed screenshot of the first result's page saved to {processed\_screenshot\_path}")

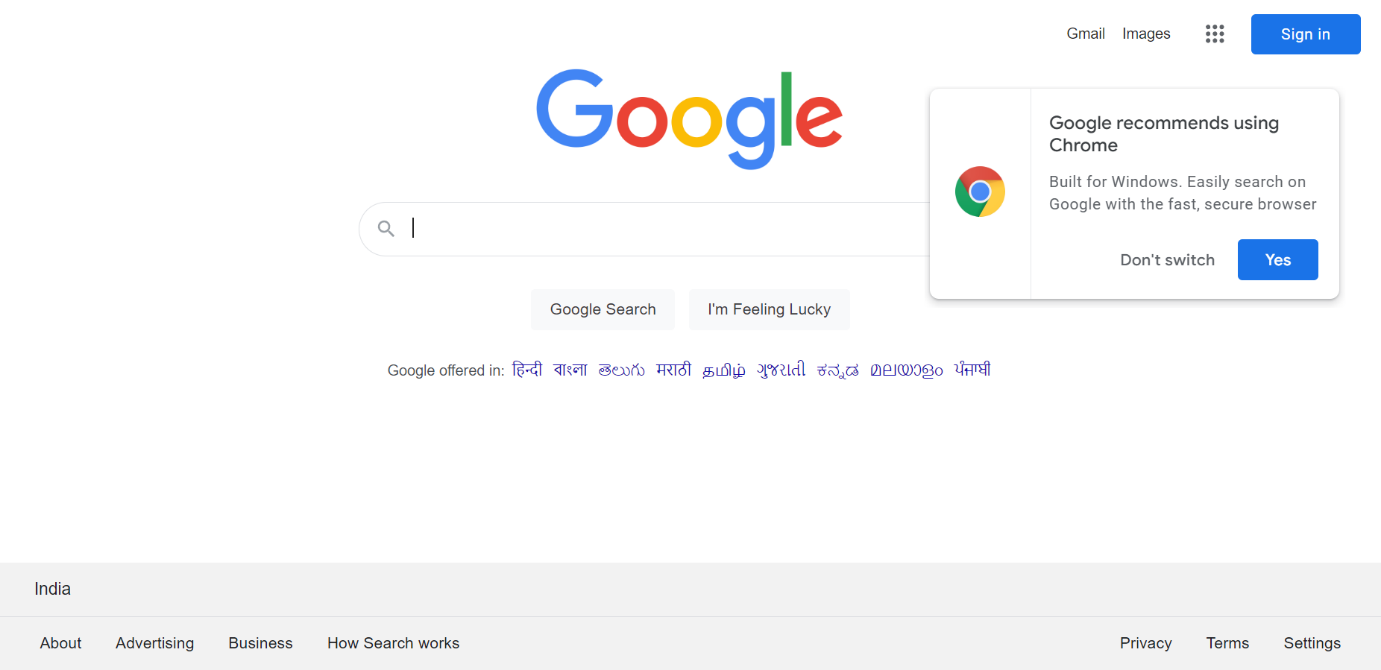
finally:

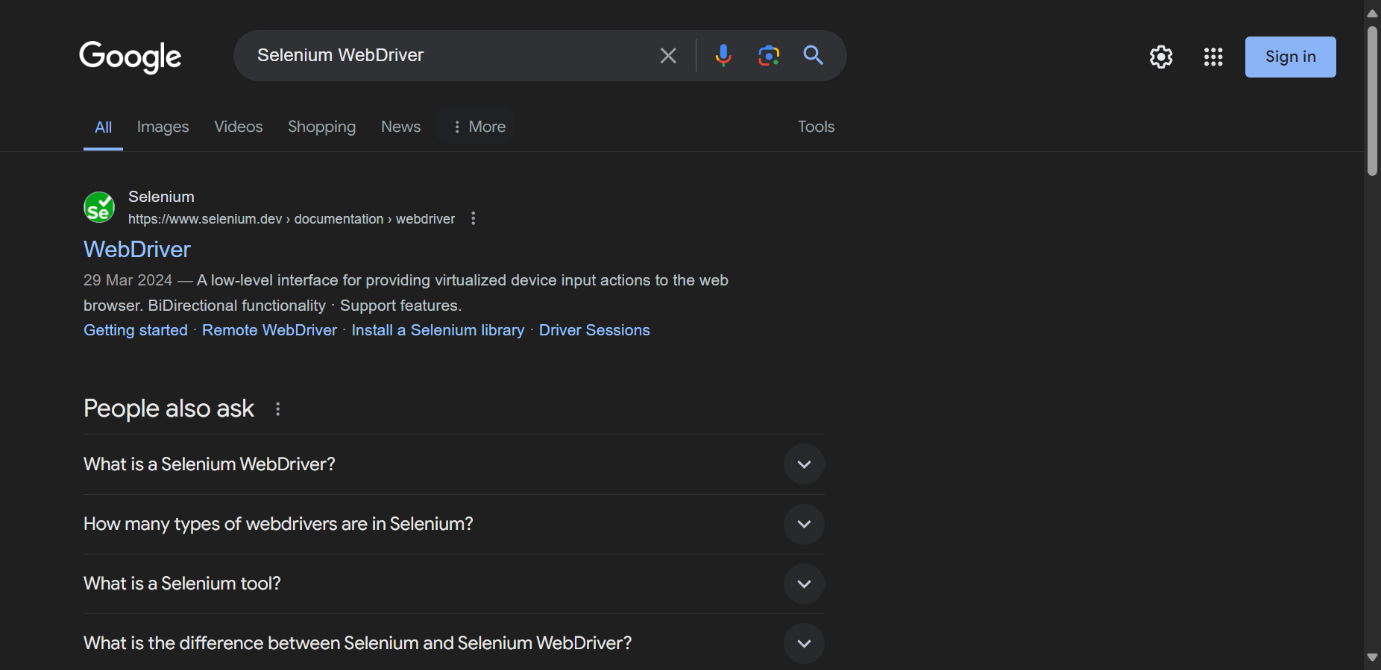
# Close the browser

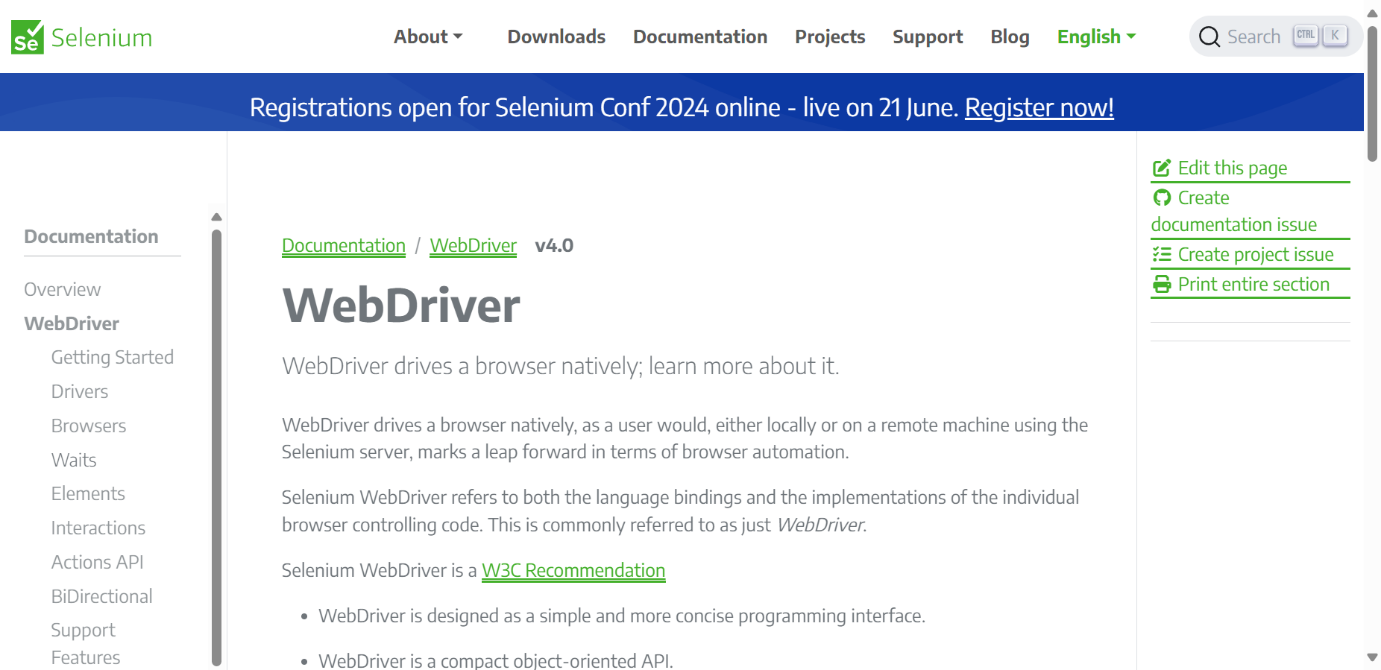
driver.quit()

print("Browser closed successfully.")

**OUTPUT:**







**AFTER RESOLUTION:**

