Selenium Lab Assignments

This set of assignments will help you practice Selenium WebDriver with C#. All exercises use the website https://the-internet.herokuapp.com which provides various web elements and scenarios for testing.

Prerequisites

- Complete the form submission guide and understand the basic concepts
- Visual Studio 2022 with a C# console application project set up
- Required NuGet packages: `Selenium.WebDriver`, `Selenium.Support`, and `WebDriverManager`
- **Important**: Read the Submission Guidelines section before starting to code your solution

Assignment 1: Basic Navigation and Verification

Objective: Navigate to the main page and verify the title and header.

Tasks:

1. Create a new C# console application
2. Navigate to https://the-internet.herokuapp.com
3. Print the page title
4. Find the heading element (`<h1>`) and print its text
5. Verify that there are multiple example links on the page (Hint: Use `FindElements`)
6. Print the total count of available examples

Assignment 2: Working with Checkboxes

Objective: Interact with checkboxes and verify their states.

Tasks:

1. Navigate to https://the-internet.herokuapp.com/checkboxes
2. Check the state of both checkboxes (selected or not)
3. Toggle the state of each checkbox (if selected, unselect; if unselected, select)
4. Verify and print the new states
5. Toggle them back to their original states

Assignment 3: Handling Dropdowns

Objective: Work with dropdown selectors.

Tasks:

1. Navigate to https://the-internet.herokuapp.com/dropdown
2. Print the currently selected option
3. Select "Option 1" using the select by visible text method
4. Verify your selection was successful
5. Select "Option 2" using the select by value or index method

6. Verify your selection was successful

Assignment 4: Basic Authentication

Objective: Handle basic authentication dialog.

Tasks:

1. Navigate to https://the-internet.herokuapp.com/basic_auth with embedded credentials: `https://admin:admin@the-internet.herokuapp.com/basic_auth`
2. Verify you successfully authenticated by checking for the success message
3. Print the success message to the console

Assignment 5: Dynamic Loading

Objective: Work with dynamically loaded elements and implement waits.

Tasks:

1. Navigate to https://the-internet.herokuapp.com/dynamic_loading/1
2. Click the "Start" button
3. Wait for the "Hello World!" text to appear (implement explicit wait)
4. Print the text once it appears
5. Repeat with https://the-internet.herokuapp.com/dynamic_loading/2
6. Compare both approaches in a comment

Assignment 6: Form Validation

Objective: Work with a form and validate error messages.

Tasks:

1. Navigate to https://the-internet.herokuapp.com/login
2. Submit the form without entering any credentials
3. Check if validation messages appear
4. Enter invalid credentials (username: "invalid", password: "invalid")
5. Submit the form
6. Capture and print the error message
7. Verify the error message contains text about invalid username

Assignment 7: Handling Alerts

Objective: Work with JavaScript alerts.

Tasks:

1. Navigate to https://the-internet.herokuapp.com/javascript_alerts
2. Click the "Click for JS Alert" button
3. Switch to the alert and print its text
4. Accept the alert
5. Verify the result text

6. Click the "Click for JS Confirm" button
7. Dismiss the confirm dialog (click Cancel)
8. Verify the result text

Assignment 8: Working with Frames

Objective: Interact with iframes.

Tasks:

1. Navigate to https://the-internet.herokuapp.com/iframe
2. Switch to the iframe (using either id, name, or index)
3. Clear the existing text in the editor
4. Type a new message: "This text was entered by Selenium automation"
5. Switch back to the main frame
6. Verify you can interact with elements outside the iframe

Assignment 9: Drag and Drop

Objective: Implement drag and drop functionality.

Tasks:

1. Navigate to https://the-internet.herokuapp.com/drag_and_drop
2. Get the initial text of both boxes A and B
3. Perform a drag and drop operation to move element A to element B
4. Verify if the operation was successful by checking if their positions swapped (Note: This may require JavaScript execution for reliable results)

Assignment 10: File Upload

Objective: Automate file upload process.

Tasks:

1. Create a small text file on your computer
2. Navigate to https://the-internet.herokuapp.com/upload
3. Locate the file input element
4. Send the path of your text file to the input element
5. Click the "Upload" button
6. Verify the file was uploaded successfully by checking the success message

Submission Guidelines

1. Create a single C# console application named "SeleniumAssignments"
2. Within this project, create separate classes for each assignment (e.g., `Assignment1.cs`, `Assignment2.cs`, etc.)
3. Each class should implement its functionality within a public method (e.g., `public void Run()`)
4. Create a menu-driven user interface in the Program.cs file that allows the user to select which assignment to run

5. Example menu implementation:

```
using System;

namespace SeleniumAssignments
{
    class Program
    {
        static void Main(string[] args)
        {
            bool exit = false;

            while (!exit)
            {
                Console.Clear();
                Console.WriteLine("Selenium Assignments Menu");
                Console.WriteLine("========================");
                Console.WriteLine("1. Basic Navigation and Verification");
                Console.WriteLine("2. Working with Checkboxes");
                Console.WriteLine("3. Handling Dropdowns");
                Console.WriteLine("4. Basic Authentication");
                Console.WriteLine("5. Dynamic Loading");
                Console.WriteLine("6. Form Validation");
                Console.WriteLine("7. Handling Alerts");
                Console.WriteLine("8. Working with Frames");
                Console.WriteLine("9. Drag and Drop");
                Console.WriteLine("10. File Upload");
                Console.WriteLine("0. Exit");
                Console.Write("\nEnter your choice (0-10): ");

                string choice = Console.ReadLine();

                switch (choice)
                {
                    case "1":
                        new Assignment1().Run();
                        break;
                    case "2":
                        new Assignment2().Run();
                        break;
                    case "3":
                        new Assignment3().Run();
                        break;
                    case "4":
                        new Assignment4().Run();
                        break;
                    case "5":
                        new Assignment5().Run();
                        break;
                    case "6":
                        new Assignment6().Run();
                        break;
                    case "7":
```

```
                        new Assignment7().Run();
                        break;
                    case "8":
                        new Assignment8().Run();
                        break;
                    case "9":
                        new Assignment9().Run();
                        break;
                    case "10":
                        new Assignment10().Run();
                        break;
                    case "0":
                        exit = true;
                        break;
                    default:
                        Console.WriteLine("Invalid choice. Press any key to
try again.");
                        Console.ReadKey();
                        break;
                }

                if (!exit)
                {
                    Console.WriteLine("\nPress any key to return to
menu...");
                    Console.ReadKey();
                }
            }
        }
    }
}
```

6. Include proper exception handling in each assignment class
7. Follow C# coding standards
8. Use explicit waits instead of Thread.Sleep() where appropriate
9. Add comments explaining your code
10. Ensure each assignment works correctly before submission
11. Consider creating a BaseAssignment class that handles common setup and teardown operations that other assignment classes can inherit from