SET-221
Software Testing Technologies

LAB # 08

LAB Title

Installation of Selenium Basics.
Setting up Selenium for web application testing.

Assessment of CLO: 04, PLO: 05

Student Name:			
Roll No.			
Semester		Session	

S. No.	Perf. Level / Criteria	Excellent (2.5)	Good (2)	Satisfactory (1.5)	Needs Improvement (0 ~ 1)	Marks Obtained
1	Project Execution & Implementation	Fully functional, optimized, and well-structured.	Minor errors, mostly functional.	Some errors, requires guidance.	Major errors, non-functional, or not Performed.	
2	Results & Debugging Or Troubleshooting	Accurate results with effective debugging Or Troubleshooting.	Mostly correct, some debugging Or Troubleshooting needed.	Partial results, minimal debugging Or Troubleshooting.	Incorrect results, no debugging Or Troubleshooting, or not attempted.	
3	Problem-Solving & Adaptability (VIVA)	Creative approach, efficiently solves challenges.	Adapts well, minor struggles.	Some adaptability, needs guidance.	Lacks innovation or no innovation, unable to solve problems.	
4	Report Quality & Documentation	Clear, structured, with detailed visuals.	Mostly clear, minor gaps.	Some clarity issues, missing details.	Poorly structured, lacks clarity, or not submitted.	
					Total Marks Obtained Out of 10	

Experiment evaluated by

Instructor's Name	Engr.Bushra Aziz		
Date		Signature	

Objective: To guide you through the process of installing Python, setting up Selenium in Visual Studio, and writing your first basic Selenium test script using Python.

Prerequisites:

- A computer with a compatible operating system (Windows, macOS, Linux).
- Visual Studio 2022 installed.

Part 1: Installing Python (If Not Already Installed)

If you already have Python installed and configured in Visual Studio, you can skip to Part 2.

Step 1.1: Check if Python is installed

1. Open **Visual Studio 2022**.
2. Go to **View > Other Windows > Python Environments**.
3. In the "Python Environments" window, check if any Python environments are listed. If you see one or more environments, Python is likely installed and configured. Note the Python version(s) listed.

Step 1.2: Download Python

If no Python environments are listed, you need to install Python:

1. Open your web browser and go to the official Python website: https://www.python.org/downloads/
2. Download the latest stable version of Python for your operating system. Look for the button that says "Download Python X.Y.Z" (where X.Y.Z is the version number).
3. Run the downloaded installer.

Step 1.3: Install Python

1. During the installation process, **make sure to check the box that says "Add Python X.Y to PATH"**. This is crucial for making Python accessible from the command line and Visual Studio.
2. Click "**Install Now**" to proceed with the installation.
3. Once the installation is complete, click "**Close**".

Step 1.4: Verify Python Installation in Visual Studio

1. Close and reopen **Visual Studio 2022**.
2. Go to **View > Other Windows > Python Environments**.
3. You should now see the newly installed Python environment listed. Visual Studio automatically detects Python installations.

Part 2: Installing the Python Development Workload in Visual Studio

If you didn't select the Python development workload during the initial Visual Studio installation, follow these steps:

Step 2.1: Open Visual Studio Installer

1. Close **Visual Studio 2022**.
2. Search for "Visual Studio Installer" in the Start Menu (Windows) or find it in your Applications folder (macOS).
3. Run the **Visual Studio Installer**.

Step 2.2: Modify Visual Studio Installation

1. In the Visual Studio Installer, find your Visual Studio 2022 installation and click the "**Modify**" button.

Step 2.3: Select Python Development Workload

1. In the Visual Studio Installer window, go to the "**Workloads**" tab.
2. Scroll down and find the "**Python development**" workload.
3. **Check the box** next to "**Python development**".
4. On the right-hand side, you can see the details of the components that will be installed. Ensure that the core Python interpreter and other essential tools are selected.
5. Click the "**Install**" or "**Modify**" button in the bottom right corner to start the installation of the Python development workload.
6. Once the installation is complete, close the Visual Studio Installer and open **Visual Studio 2022**.

Part 3: Creating a Python Project in Visual Studio

If you already have a Python project, you can skip this part.

Step 3.1: Create a New Project

1. In **Visual Studio 2022**, go to **File > New > Project...**.
2. In the "Create a new project" window, search for "**Python**" in the search bar.
3. Select the "**Python Application**" template.
4. Click "**Next**".

Step 3.2: Configure Project Details

1. Enter a **Project name** (e.g., SeleniumAutomation).
2. Choose a **Location** to save your project.
3. Click "**Create**". Visual Studio will create a new Python project with a default .py file.

Part 4: Installing the Selenium Package

Now, let's install the Selenium library using pip within Visual Studio.

Step 4.1: Open Python Environments Window

1. In **Visual Studio 2022**, with your Python project open, go to **View > Other Windows > Python Environments**.

Step 4.2: Select Your Python Environment

1. In the "Python Environments" window, ensure that the desired Python environment for your project is selected. You'll see information about the installed Python version and packages.

Step 4.3: Open the Packages (PyPI) Tab

1. In the "Python Environments" window, look for the tab labeled "**Packages (PyPI)**". Click on it.

Step 4.4: Search for Selenium

1. In the search bar under the "Packages (PyPI)" tab, type "**selenium**" and press **Enter** or click the search icon.

Step 4.5: Install Selenium

1. You will see the selenium package listed in the search results. Click the "**Install**" button (it usually looks like a small down arrow next to the package name).
2. Visual Studio will start installing the Selenium package and its dependencies. You can monitor the progress in the **Output** window (go to **View > Output** if you don't see it).

Step 4.6: Verify Installation

1. Once the installation is complete, you should see the selenium package listed under your Python environment in the "Python Environments" window.

Part 5: Installing the Chrome WebDriver

Selenium requires a browser-specific driver to interact with web browsers. We'll focus on Google Chrome here.

Step 5.1: Determine Your Chrome Browser Version

1. Open **Google Chrome**.
2. Click on the **three vertical dots** (Menu) in the top-right corner.
3. Go to **Help > About Google Chrome**.
4. Note down the **major version number** (the first part of the version string).

Step 5.2: Download ChromeDriver

1. Open your web browser and go to the ChromeDriver downloads page: https://chromedriver.chromium.org/downloads
2. **Find the ChromeDriver version that closely matches your Chrome browser's major version.**
3. Click on the link corresponding to your Chrome version.
4. Download the ChromeDriver for your operating system (e.g.,

   - chromedriver_win32.zip for Windows 32-bit,
   - chromedriver_win64.zip for Windows 64-bit,
   - chromedriver_linux64.zip for Linux,
   - chromedriver_mac64.zip or chromedriver_mac_arm64.zip for macOS).

5. **Extract the downloaded .zip file.** You will find the chromedriver executable (chromedriver.exe on Windows, chromedriver on macOS and Linux).

Step 5.3: Configure ChromeDriver Path (Recommended)

To make your Selenium scripts run without explicitly specifying the ChromeDriver path in your code, it's best to add the directory containing the chromedriver executable to your system's PATH environment variable.

- **Windows:**
    1. Search for "Environment Variables" in the Start Menu and open "Edit the system environment variables".
    2. Click the "Environment Variables..." button.
    3. In the "System variables" section, find the [1] variable named "Path" and select it.
    4. Click "Edit...", then "New", and add the **full path to the directory where you extracted chromedriver.exe**.
    5. Click "OK" on all open windows. You might need to restart Visual Studio for the changes to take effect.
- **macOS and Linux:**
    1. Open your terminal.
    2. Edit your shell's configuration file (e.g., ~/.bashrc, ~/.zshrc).
    3. Add the following line, replacing /path/to/chromedriver_directory with the actual path:

       Bash

       ```
       export PATH="$PATH:/path/to/chromedriver_directory"
       ```
    4. Save the file and run source ~/.bashrc or source ~/.zshrc to apply the changes.

Alternatively (Less Recommended): You can place the chromedriver executable in the same directory as your Python script.

Part 6: Writing Your First Selenium Test Script

Now, let's write a simple Python script to open a web browser using Selenium in Visual Studio.

Step 6.1: Create a New Python File

1. In Solution Explorer, right-click on your project.
2. Go to **Add** > **New Item...**.
3. Select "**Python File**" and give it a name (e.g., first_test.py). Click "**Add**".

Step 6.2: Write the Selenium Script

1. Open the first_test.py file and enter the following code:

   Python

```python
from selenium import webdriver
from selenium.webdriver.chrome.service import Service as ChromeService
from webdriver_manager.chrome import ChromeDriverManager

driver = webdriver.Chrome(service=ChromeService(ChromeDriverManager().install()))
```

```python
def verify_title():
    driver.get("https://www.google.com")
    expected_title = "Google"
    actual_title = driver.title
    if actual_title == expected_title:
        print(f"Title verification successful! Actual title: '{actual_title}'")
    else:
        print(f"Title verification failed! Expected: '{expected_title}', Actual:
'{actual_title}'")
    driver.quit()

if __name__ == '__main__':
    verify_title()
```

Step 6.3: Run the Script

1. In Solution Explorer, right-click on the first_test.py file.
2. Select "**Run Python File**".

Step 6.4: Observe the Results

1. A Chrome browser window should open, navigate to https://www.example.com, and then close.
2. In the Visual Studio **Output** window (View > Output), you should see the title of the webpage printed:

   The title of the page is: Google

Task:

1. Write and implement a test case to verify your website Title.
2. Write and implement a test case to URL of your website.