Documentation for the Web Automation Script

This documentation outlines the setup and execution steps for automating the interaction with the **FitPeo** webpage using Selenium WebDriver and Python. The automation script handles tasks like navigating to specific pages, interacting with web elements, and validating results as per the given test cases.

1. Objective

The objective of this script is to automate the following tasks on the **FitPeo Homepage** and **Revenue Calculator Page**:

- **Navigate to the FitPeo Homepage.**
- **Navigate to the Revenue Calculator Page.**
- **Scroll down to the slider section of the page.**
- **Adjust the slider to set its value to 820.**
- **Update the associated text field to 560.**
- **Ensure the slider updates accordingly when the text field is modified.**
- **Select specific CPT codes checkboxes.**
- **Validate the Total Recurring Reimbursement value.**

2. Prerequisites

Before running the script, ensure that the following tools and dependencies are installed:

a. Python

- Install Python 3.x (if not already installed): Python Official Site.

b. Selenium WebDriver

- Install the selenium Python package using pip:

```
pip install selenium
```

c. WebDriver (ChromeDriver)

- Download the appropriate ChromeDriver version based on your browser version:
    - ChromeDriver
- Ensure the chromedriver.exe is available in your system's PATH or specify the path in the script.

---

3. Setting up the Environment

a. Clone or Copy the Code

1. Clone or copy the automation script provided in this documentation.
2. Ensure that the script is located in a folder where the necessary dependencies (like selenium) are accessible.

b. Ensure WebDriver is Installed

Make sure that the WebDriver (chromedriver.exe) is installed and accessible. You can either download it and place it in the project directory or add it to the system's PATH.

4. Script Overview

The script follows the following flow:

1. **Navigate to FitPeo Homepage**: The script opens the FitPeo homepage URL.
2. **Navigate to Revenue Calculator Page**: Clicks the "Revenue Calculator" link on the homepage.
3. **Scroll Down to Slider Section**: Scrolls down the page to make the slider visible.
4. **Adjust the Slider**: Moves the slider to the value 820 and ensures that the associated text field updates.
5. **Update the Text Field**: Updates the text field to 560 and ensures that the slider reflects the new value.
6. **Select CPT Codes**: Selects the checkboxes for CPT-99091, CPT-99453, CPT-99454, and CPT-99474.
7. **Validate Total Recurring Reimbursement**: Validates the header showing the total recurring reimbursement is set to $27000.

---

5. Script Breakdown

Below is a breakdown of the individual functions and methods in the script:

a. open_website(url)

- **Purpose**: Opens the given URL and maximizes the browser window.

b. click_revenue_calculator()

- **Purpose**: Clicks on the "Revenue Calculator" link to navigate to the Revenue Calculator page.

c. find_elements()

- **Purpose**: Locates the slider handle, slider track, and value input field.

d. clear_and_set_value(value)

- **Purpose**: Clears the existing value in the text field and sets a new value.

e. check_value_in_textbox(expected_value)

- **Purpose**: Verifies that the value in the text box matches the expected value.

f. move_slider_to_target()

- **Purpose**: Moves the slider to the target value and ensures the text field and slider stay in sync.

g. run(url, target_value="560")

- **Purpose**: Runs the entire automation process, including opening the website, interacting with the slider, and verifying results.

h. generate_cpt_code_xpaths(self)

- **Purpose**: Generates a list of XPath expressions for the CPT codes defined in the self.list_of_CPT_codes list.

i. checkbox_code(self, locator)

- **Purpose**: Clicks on the checkbox for a specified CPT code (identified by the locator), scrolls to the element, and extracts the reimbursement value.
- **Parameters**:
  - locator: The XPath of the checkbox element for a specific CPT code.

j. count_of_patients(self)

- **Purpose**: Retrieves the total recurring reimbursement value and the number of patients from the page. It then calculates the total value of reimbursement for all patients.

k. validation_of_total_recurring_reimbursement_per_month(self)

- **Purpose**: Compares the calculated total recurring reimbursement value with the value retrieved from the webpage and raises an assertion error if they don't match.

5. How to Run the Script

1. **Install Dependencies**:
   - Ensure Python 3.x is installed.
   - Install Selenium using the following command
     "pip install selenium"
2. **Download Chrome WebDriver**:
   - Download and extract the appropriate version of **ChromeDriver** for your Chrome browser version.
3. **Set Up the Script**:
   - Save the script as a .py file (e.g., fitpeo_page.py).
   - Ensure chromedriver.exe is in the same directory as the script or is added to the system's PATH.
4. **Run the Script**:

```
python fitpeo_page.py
```

Troubleshooting Tips:

Ensure Compatibility:

- Verify that the version of ChromeDriver matches your version of Google Chrome.
- Make sure that Python and pip are correctly installed on your system.

ChromeDriver Path Issues:

- If Selenium can't find chromedriver.exe, you might need to specify the full path in your script like this:

```
from selenium import webdriver

driver = webdriver.Chrome(executable_path="C:/path/to/chromedriver.exe")
```

Replace **"C:/path/to/chromedriver.exe" with the actual location of the chromedriver.exe file.**

7. Handling Exceptions and Robustness

- The script uses WebDriverWait and expected_conditions to wait for elements to be visible or clickable. This ensures robustness in case of slow-loading web elements.
- If any step fails (such as an element not being found or a value mismatch), the script will raise an exception and terminate, providing clear feedback on what went wrong.

8.CONCLUSION:

This script automates the specified tasks on the **FitPeo Page**. By using Selenium WebDriver with Python, it ensures that the entire process, from navigating the website to validating values, is executed automatically and accurately. The script is modular, making it easy to maintain and extend.