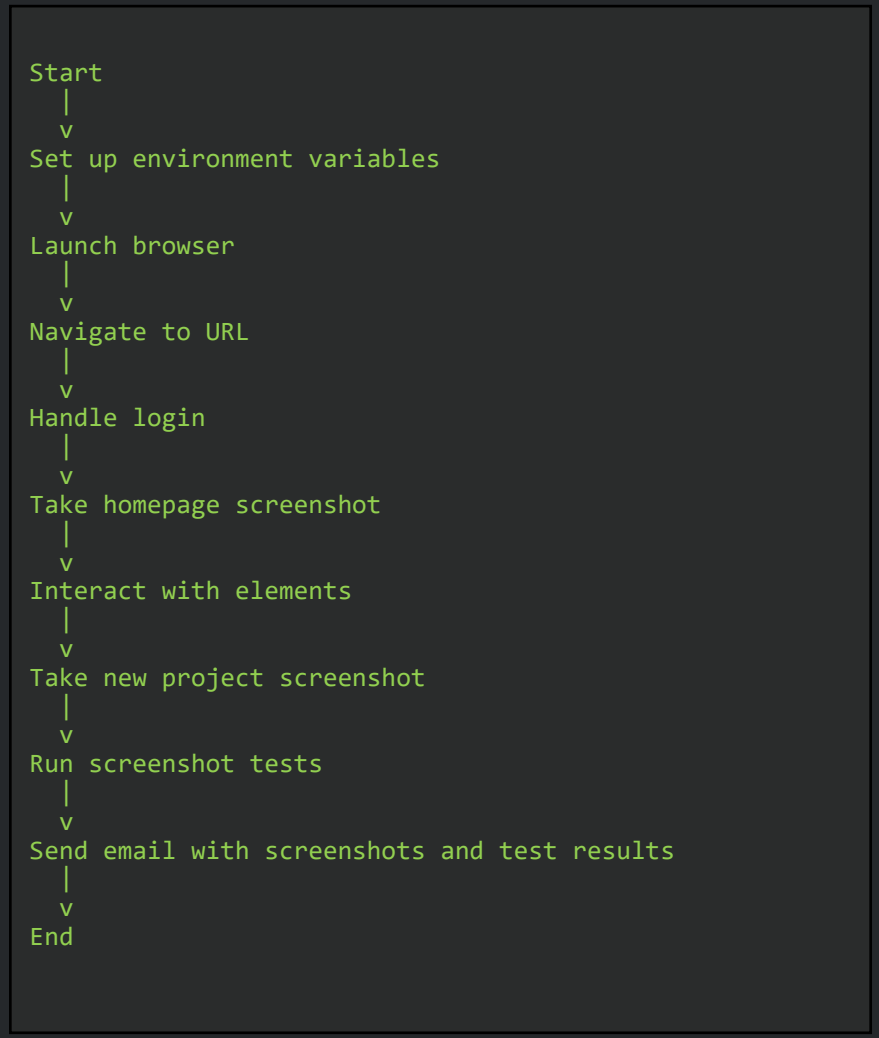
# Automation using Playwright

# FLOWCHART


Start
Set up environment variables
Launch browser
Navigate to URL
Handle login
Take homepage screenshot
Interact with elements
Take new project screenshot
Run screenshot tests
Send email with screenshots and test results
End

# Libraries used

logging:
This library is used to configure and manage logging in your script.
It helps you track the progress and identify any issues by logging messages to a file and the console.

os:
The os module provides a way to interact with the operating system.
It is used to handle file paths, environment variables, and other OS-related tasks.

asyncio:
This library is used for writing asynchronous code in Python. It allows you to run multiple tasks concurrently,
which is useful for handling I/O-bound operations like web scraping and network requests.

playwright.async_api:
Playwright is a library for automating web browsers.
The async_api module provides asynchronous functions to interact with web pages, such as launching a browser, navigating to a URL, and taking screenshots.

**smtplib:**
This library is used for sending emails using the Simple Mail Transfer Protocol (SMTP). It allows you to connect to an SMTP server and send emails with attachments.

**email.mime.multipart, email.mime.text, email.mime.image:**
These modules are part of the email package and are used to create and manage email messages with multiple parts, such as text and image attachments.

**pytest:**
Pytest is a testing framework for Python.
It is used to write and run tests, making it easier to ensure that your code works as expected.

**PIL (Python Imaging Library):**
The PIL library, specifically the Image module, is used for opening, manipulating, and saving image files.
It helps you verify that the screenshots taken by Playwright are valid images.

**sys:**
The sys module provides access to some variables and functions that interact with the Python runtime environment.
It is used to handle command-line arguments and manage the standard input/output streams.

io:
The io module provides tools for working with streams, such as reading and writing data to files or in-memory buffers.
The StringIO class is used to capture the output of the tests.

datetime:
The datetime module supplies classes for manipulating dates and times.
It is used to generate timestamps for logging and email subjects.

Step 1: Environment Variables for setting up HTTP and HTTPS proxies

```
$env:HTTP_PROXY = "http://10.74.207.16:8080"
$env:HTTPS_PROXY = "http://10.74.207.16:8080"
```

Step 2: Installing necessary packages

```
python -m pip install --upgrade pip --proxy=http://10.74.207.16:8080
pip install --proxy=http://10.74.207.16:8080 playwright
pip install --proxy=http://10.74.207.16:8080 pytest-playwright
```

Step 3: Setting up the virtual environment

```
# Deactivate virtual environment first
Deactivate

# Remove the existing virtual environment
Remove-Item -Recurse -Force venv

# Create a new virtual environment
python -m venv newvenv

# Activate the new environment
.\venv\Scripts\activate
```

Importing modules and configuring logging

```
import logging
import os
import asyncio
from playwright.async_api import async_playwright
import smtplib
from email.mime.multipart import MIMEMultipart
from email.mime.text import MIMEText
from email.mime.image import MIMEImage
import pytest
from PIL import Image
import sys
from io import StringIO
import datetime

# Configure logging
logging.basicConfig(
    level=logging.INFO,
    format='%(asctime)s - %(levelname)s - %(message)s',
    handlers=[
        logging.FileHandler('automation.log'),
        logging.StreamHandler()
    ]
)
logger = logging.getLogger(__name__)
```