WIPRO NGA Program

Capstone Project Presentation – 05 August 2024

Project Title – Automating candymapper.com with Python and Selenium

Presented by – A Akhil

Introduction

- **Background:**
  - CandyMapper is a web site which is a funny demos site for testing.
  - This site is based on Halloween party theme where we can host and attend party.
  - This site consist of 5 modules namely Home, Join Us, Halloween Party, Launch Candy Mapper and more.
  - We have opted for web automation testing as it help us to save time and extract data for report more efficiently as compared to manual testing.

- **Problem Statement:**
  - Need to automated testing to ensure the functionality of candymapper.com
  - Challenges in manual testing and data extraction.

- **Objective:**
  - Develop automated test script to verify the functionality of the key feature on candymapper.com
  - Implementing data extraction for analysis or reporting purposes.

Project overview

- **Project Scope:**
  - Login functionality has been automated.
  - Account creation functionality has been automated.
  - Logout functionality has been automated.
  - Join us page has been automated.

- **Tools and Technology:**
  - **Python:** Programming language used.
  - **Selenium:** Tool for web automation.
  - **WebDriver:** Chrome WebDriver.
  - **Jira:** For testing board.

Methodology

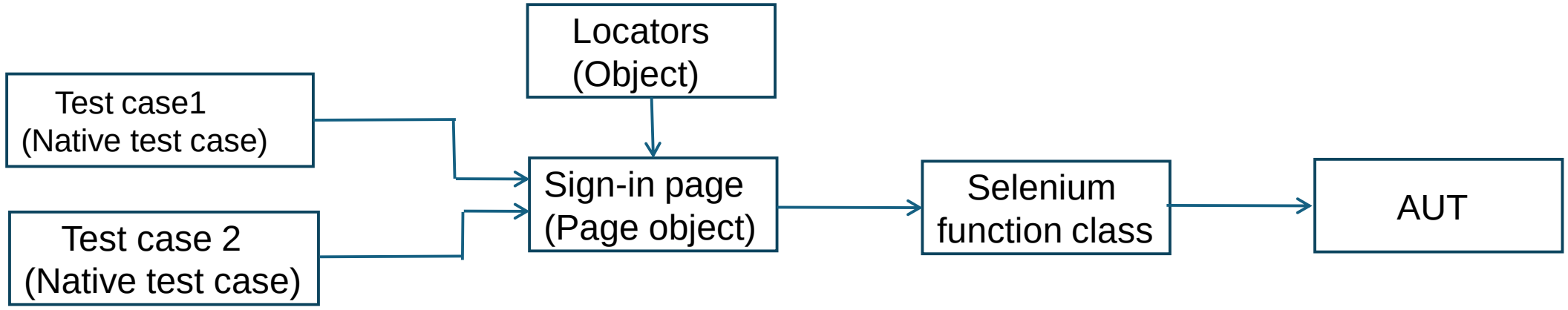
- **Approach:**
  - **SetUp:** We have installed the following things to setup our testing environment
    - ➢Created jira board and created test cases in zephyr scale.
    - ➢Installed PyCharm for Python user interface.
    - ➢Installed Selenium for web automation.
    - ➢Installed Html Reporter for generating test execution report
    - ➢We have installed chrome web driver as a WebDriver.

  - **Scripts:**
    - ➢We have followed page object model to create scripts.
    - ➢ There is 5 test scripts
      1. For closing pop-up: It closes the popup.
      2. For logging into account: It consist of all necessary action required for log-in into account
      3. For Log out: It consist of all necessary action required for logging out of account
      4. For Joins us: It contains script to create new account.
      5. For Creating account: It contains script for creating a new account.
    - ➢ Names of the test script starts with test_ (e.g.: test_login.py)

Methodology(Contd..)

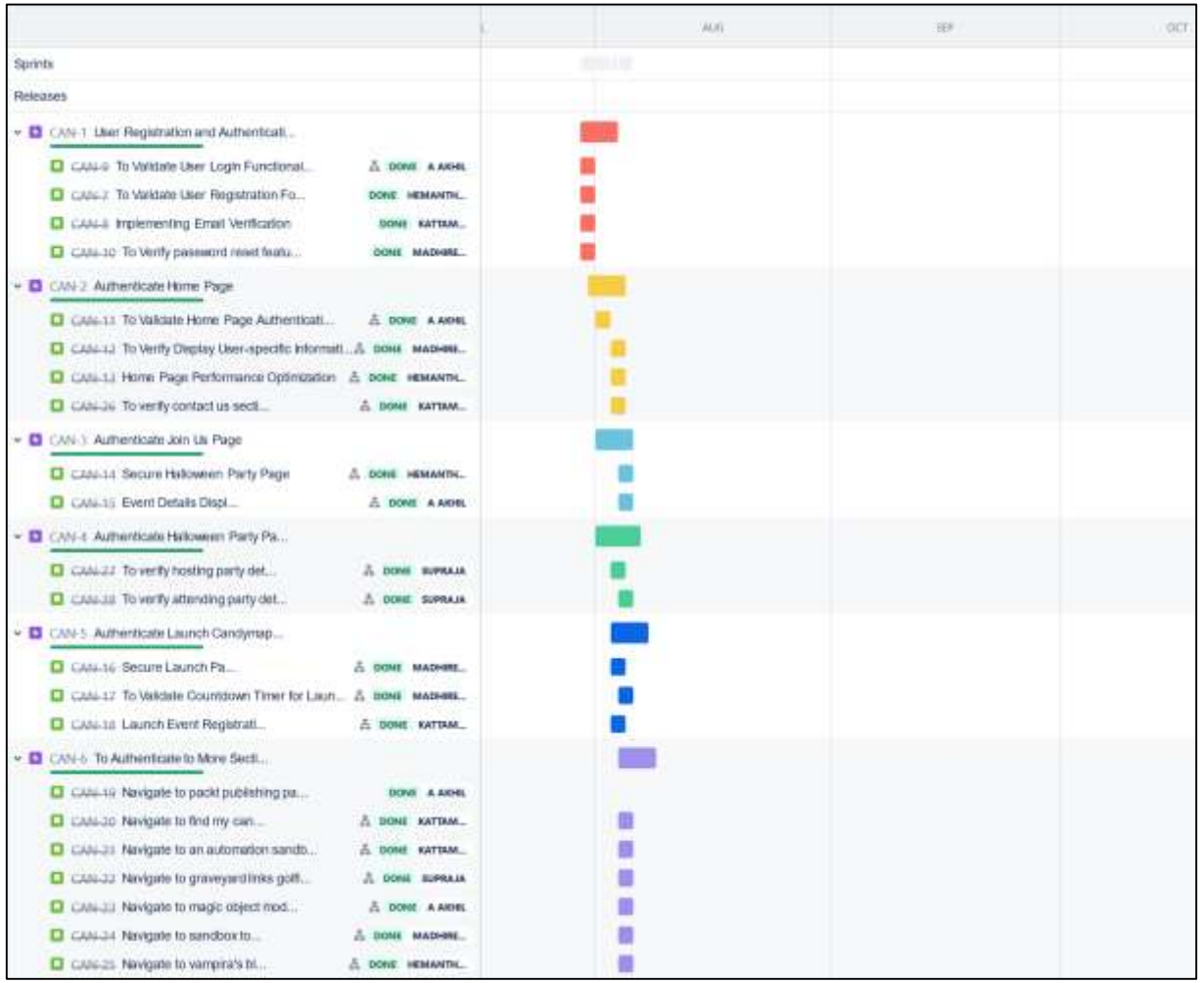
- **Testing Process**
  - **Test Cases:**
    1. To close pop-up which appear as soon we open the web site, if not close then we can not access other elements
    2. To authenticate sign with correct credentials so that login is authenticate.
    3. To sing-in with invalid credential to verify the login page functionality.
    4. To verify email is sent with credential while creating account to user.
    5. To verify if user is new to page then able to create account by click join us.

  - **Automation Flow:**



Scrum Board

- Jira Timeline



- Jira testing board.

Implementation Details

```python
class Testlogout(Web_driver_setup):

    def test_logout(self):
        try:
            driver = self.driver
            self.driver.get(Config.url)
            Sign_in = Login_Page(driver)
            # This will close pop-up
            Close_PopUp = PopUp(driver)
            Close_PopUp.click_popUp_close()
            # Now opening signIn page form naviation bar
            Sign_in.get_signIn_page()
            # entering credential and signing in
            Sign_in.get_user_name(Config.username)
            Sign_in.get_password(Config.password)
            Sign_in.click_signin_button()
            # Now logging off
            sign_out = Logout(driver)
            sign_out.click_logout()
        except:
            assert 1 == 0, 'Something went wrong'
```

- Close pop-up, Login, Logout Automation

```python
class TestCreateAccount(Web_driver_setup):
    def test_create_account(self):
        try:
            driver = self.driver
            self.driver.get(Config.url)
            # This will close pop-up
            Close_PopUp = PopUp(driver)
            Close_PopUp.click_popUp_close()
            # This will re-direct to signUp page
            account = CreateAccount(driver)
            # This fill registration details
            account.get_create_account_page()
            # Checking for sigUp page opening
            msg = account.get_message()
            assert msg == 'Create Account', 'Failed to load page'
            # This will fill all the required data
            account.get_create_account()
            account.click_create_account()
            # Getting success message of account creation
            success_msg = account.get_success_msg()
            assert success_msg == 'Check your email', 'Account creation failed'
        except:
            assert 1 == 0, 'Something went wrong'
```

- Create Account Automation.

learn..evolve

Implementation Details(Contd..)

```python
def test_join_us_login(self):
    try:
        driver = self.driver
        driver.get(Config.url)
        # This will close pop-up
        Close_PopUp = PopUp(driver)
        Close_PopUp.click_popUp_close()
        # Now opening signIn page form navigation bar
        Sign_in = Login_Page(driver)
        Sign_in.get_signIn_page()
        # entering credential and signing in
        msg = Sign_in.get_msg()
        assert msg == 'Account sign in', 'Failed to load signin page'
        Sign_in.get_user_name(Config.username)
        Sign_in.get_password(Config.password)
        Sign_in.click_signin_button()
        # Now clicking on join us from navigation bar
        join_us = JoinUs(driver)
        join_us.click_join_us()
        # Checking for valid messgae
        success_msg = join_us.login_success_join()
        assert success_msg == 'Account Login'
    except:
        assert 1 == 0, 'Something went wrong'
```

- **Automating Join Us after log-in**

- **Automating Join Us without login**

```python
def test_join_us_loggedOut(self):
    try:
        driver = self.driver
        driver.get(Config.url)
        # This will close pop-up
        Close_PopUp = PopUp(driver)
        Close_PopUp.click_popUp_close()
        # This will click on joinUs from navigation bar
        join_us = JoinUs(driver)
        join_us.click_join_us()
        # Checking for valid messgae
        success_msg = join_us.login_fail_join()
        assert success_msg == 'Account sign in'
    except:
        assert 1 == 0, 'Something went wrong'
```

Results

•Close Popup

Suite	Test Case	Status	Time (s)	Error Message
test_close_popup.py	test_closePopUp	PASS	5.0	

•Account Login

Suite	Test Case	Status	Time (s)	Error Message
test_login.py	test_wrong_login	PASS	25.32	
test_login.py	test_login	PASS	11.7	

•Account Logout

Suite	Test Case	Status	Time (s)	Error Message
test_logout.py	test_logout	PASS	37.04	

Results(Contd..)

- **Create Account Page**

Suite	Test Case	Status	Time (s)	Error Message
test_create_account.py	test_create_account	PASS	42.35	

- **Join Us Page**

Suite	Test Case	Status	Time (s)	Error Message
test_join_us.py	test_join_us_login	PASS	38.83	
test_join_us.py	test_join_us_loggedOut	PASS	27.74	

Discussion

- **Findings:**
  1. **We have to give proper mail id for login and account creation.**
  2. **We have to handle popup to continue with automation testing.**

- **Challenges:**
  1. **Finding locators, Specially dynamic locators**
  2. **Setting up of page object model.**

- **Benefits:**
  1. **Automation testing helps us to save time.**
  2. **Automation testing is more accurate than manual testing.**
  3. **Report and screenshot can be generated automatically**

Conclusion

- **Summary:**
  1. We have automated Candymapper.com for ensuring functionality.
  2. We have seen the flow of automation testing followed.
  3. We have seen execution reports and jira reports.

- **Contributions:**
  1. Made initial setup for testing environment.
  2. Automated Sign page, Create account page, Join us page.
  3. Generated report for the above automation.

- **Future work:**
  1. Validate email before creating account.