**Assignment 7 - Create Account Test with Pytest Fixtures**

* **Program:**

import pytest

from faker import Faker

from types import SimpleNamespace

from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import WebDriverWait

from selenium.webdriver.support import expected\_conditions as expected

class TestCreateAccount:

@pytest.fixture(autouse=True)

def setup(self, driver):

self.driver = driver

self.wait = WebDriverWait(driver, 10)

self.fake = Faker()

@pytest.fixture

def account(self):

account = SimpleNamespace()

account.title = "Mrs."

account.first\_name = self.fake.first\_name\_female()

account.last\_name = self.fake.last\_name()

account.full\_name = account.first\_name + " " + account.last\_name

account.email = self.fake.email()

account.password = "Password1!"

print(account)

self.account = account

return account

def test\_create\_account(self, account):

self.navigate\_to\_create\_account()

self.select\_social\_title(account.title)

self.enter\_first\_name(account.first\_name)

self.enter\_last\_name(account.last\_name)

self.enter\_email(account.email)

self.enter\_password(account.password)

self.check\_required\_checkboxes()

self.click\_save()

assert self.logged\_in()

assert self.get\_account\_name() == account.full\_name

def test\_create\_account\_mr(self, account):

self.navigate\_to\_create\_account()

self.select\_social\_title("Mr.")

self.enter\_first\_name(account.first\_name)

self.enter\_last\_name(account.last\_name)

self.enter\_email(account.email)

self.enter\_password(account.password)

self.check\_required\_checkboxes()

self.click\_save()

assert self.logged\_in()

assert self.get\_account\_name() == account.full\_name

def open\_create\_account\_page(self):

self.driver.get("https://shop.one-shore.com/index.php?controller=authentication&create\_account=1")

def navigate\_to\_create\_account(self):

self.driver.get("https://shop.one-shore.com")

sign\_in\_button\_locator = By.PARTIAL\_LINK\_TEXT, "Sign in"

heading\_locator = By.CSS\_SELECTOR, "h1"

create\_account\_button\_locator = By.CSS\_SELECTOR, ".no-account > a"

self.driver.find\_element(\*sign\_in\_button\_locator).click()

heading = self.driver.find\_element(\*heading\_locator).text

assert heading == "Log in to your account"

self.driver.find\_element(\*create\_account\_button\_locator).click()

heading = self.driver.find\_element(\*heading\_locator).text

assert heading == "Create an account"

def select\_social\_title(self, title:str):

title\_locator = By.CSS\_SELECTOR, ".radio-inline"

for element in self.driver.find\_elements(\*title\_locator):

print(element.text)

if element.text == title:

element.click()

def enter\_first\_name(self, firstname):

first\_name\_field = self.driver.find\_element(By.NAME, "firstname")

first\_name\_field.send\_keys(firstname)

def enter\_last\_name(self, lastname):

last\_name\_field = self.driver.find\_element(By.NAME, "lastname")

last\_name\_field.send\_keys(lastname)

def enter\_email(self, email):

email\_field = self.driver.find\_element(By.NAME, "email")

email\_field.send\_keys(email)

def enter\_password(self, password):

password\_field = self.driver.find\_element(By.NAME, "password")

password\_field.send\_keys(password)

def check\_required\_checkboxes(self):

checkboxes\_locator = By.XPATH, "//input[@type='checkbox']"

checkboxes = self.driver.find\_elements(\*checkboxes\_locator)

print(f"checkboxes: {len(checkboxes)}")

required\_checkboxes = list(

filter(lambda checkbox: checkbox.get\_attribute("required"), checkboxes)

)

print(f"required\_checkboxes: {len(required\_checkboxes)}")

[box.click() for box in required\_checkboxes if not box.is\_selected()]

def click\_save(self):

save\_button\_locator = By.CSS\_SELECTOR, ".register-form button[type=submit]"

save\_button = self.driver.find\_element(\*save\_button\_locator)

save\_button.click()

def logged\_in(self):

self.wait.until(expected.title\_is("ONESHORE DEMO SHOP"))

user\_info = self.driver.find\_element(By.CSS\_SELECTOR, ".user-info")

if "Sign out" not in user\_info.text:

return False

return True

def get\_account\_name(self):

return self.driver.find\_element(By.CSS\_SELECTOR, ".user-info .account").text

* **Output:**

=============================================================== test session starts ================================================================

platform win32 -- Python 3.8.8, pytest-6.2.3, py-1.10.0, pluggy-0.13.1

rootdir: D:\Automation Assignments\Assignments\Selenium

plugins: anyio-2.2.0, Faker-9.7.1

collected 2 items

test\_create\_account\_Assignment\_7.py

DevTools listening on ws://127.0.0.1:49693/devtools/browser/4300e60b-4103-4c50-a4bd-c9920d6772d8

.[10004:3132:1031/210031.241:ERROR:chrome\_browser\_main\_extra\_parts\_metrics.cc(230)] crbug.com/1216328: Checking Bluetooth availability started. Please report if there is no report that this ends.

[10004:13300:1031/210031.242:ERROR:device\_event\_log\_impl.cc(214)] [21:00:31.241] USB: usb\_device\_handle\_win.cc:1048 Failed to read descriptor from node connection: A device attached to the system is not functioning. (0x1F)

[10004:13300:1031/210031.244:ERROR:device\_event\_log\_impl.cc(214)] [21:00:31.243] USB: usb\_device\_handle\_win.cc:1048 Failed to read descriptor from node connection: A device attached to the system is not functioning. (0x1F)

[10004:3132:1031/210031.244:ERROR:chrome\_browser\_main\_extra\_parts\_metrics.cc(233)] crbug.com/1216328: Checking Bluetooth availability ended.

[10004:3132:1031/210031.248:ERROR:chrome\_browser\_main\_extra\_parts\_metrics.cc(236)] crbug.com/1216328: Checking default browser status started. Please report if there is no report that this ends.

[10004:3132:1031/210031.277:ERROR:chrome\_browser\_main\_extra\_parts\_metrics.cc(240)] crbug.com/1216328: Checking default browser status ended.

DevTools listening on ws://127.0.0.1:53979/devtools/browser/73684fda-4b14-4d4f-8fba-e34577f83f2d

.[12448:10092:1031/210052.924:ERROR:chrome\_browser\_main\_extra\_parts\_metrics.cc(230)] crbug.com/1216328: Checking Bluetooth availability started. Please report if there is no report that this ends.

[12448:5792:1031/210052.926:ERROR:device\_event\_log\_impl.cc(214)] [21:00:52.926] USB: usb\_device\_handle\_win.cc:1048 Failed to read descriptor from node connection: A device attached to the system is not functioning. (0x1F)

[12448:5792:1031/210052.928:ERROR:device\_event\_log\_impl.cc(214)] [21:00:52.928] USB: usb\_device\_handle\_win.cc:1048 Failed to read descriptor from node connection: A device attached to the system is not functioning. (0x1F)

[12448:10092:1031/210052.943:ERROR:chrome\_browser\_main\_extra\_parts\_metrics.cc(233)] crbug.com/1216328: Checking Bluetooth availability ended.

[12448:10092:1031/210052.944:ERROR:chrome\_browser\_main\_extra\_parts\_metrics.cc(236)] crbug.com/1216328: Checking default browser status started. Please report if there is no report that this ends.

[12448:10092:1031/210052.980:ERROR:chrome\_browser\_main\_extra\_parts\_metrics.cc(240)] crbug.com/1216328: Checking default browser status ended.

[100%]

================================================================= warnings summary =================================================================

C:\ProgramData\Anaconda3\lib\site-packages\pyreadline\py3k\_compat.py:8

C:\ProgramData\Anaconda3\lib\site-packages\pyreadline\py3k\_compat.py:8: DeprecationWarning: Using or importing the ABCs from 'collections' instead

of from 'collections.abc' is deprecated since Python 3.3, and in 3.9 it will stop working

return isinstance(x, collections.Callable)

-- Docs: https://docs.pytest.org/en/stable/warnings.html

========================================================== 2 passed, 1 warning in 43.06s ===========================================================

