import time

import pytest

from selenium.webdriver.common.by import By

from selenium.webdriver.support.ui import WebDriverWait

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

from selenium.webdriver.common.action\_chains import ActionChains

class LoginPage:

def \_\_init\_\_(self, driver):

self.driver = driver

self.username\_textbox = (By.NAME, "username")

self.password\_textbox = (By.NAME, "password")

self.login\_button = (By.XPATH, "//button[@type='submit']")

def enter\_username(self, username):

self.driver.find\_element(\*self.username\_textbox).send\_keys(username)

def enter\_password(self, password):

self.driver.find\_element(\*self.password\_textbox).send\_keys(password)

def click\_login(self):

self.driver.find\_element(\*self.login\_button).click()

class DashboardPage:

def \_\_init\_\_(self, driver):

self.driver = driver

self.pim\_menu = (By.XPATH, "//span[text()='PIM']")

self.add\_employee\_submenu = (By.XPATH, "//a[contains(text(), 'Add Employee')]")

self.employee\_list\_submenu = (By.XPATH, "//a[contains(text(), 'Employee List')]")

self.user\_dropdown = (By.XPATH, "//p[@class='oxd-userdropdown-name']")

self.logout\_button = (By.XPATH, "//a[text()='Logout']")

def hover\_over\_pim(self):

pim\_element = self.driver.find\_element(\*self.pim\_menu)

actions = ActionChains(self.driver)

actions.move\_to\_element(pim\_element).perform()

def click\_add\_employee(self):

self.driver.find\_element(\*self.add\_employee\_submenu).click()

def click\_employee\_list(self):

self.driver.find\_element(\*self.employee\_list\_submenu).click()

def click\_user\_dropdown(self):

WebDriverWait(self.driver, 10).until(EC.element\_to\_be\_clickable(self.user\_dropdown)).click()

def click\_logout(self):

WebDriverWait(self.driver, 10).until(EC.element\_to\_be\_clickable(self.logout\_button)).click()

class AddEmployeePage:

def \_\_init\_\_(self, driver):

self.driver = driver

self.first\_name\_textbox = (By.NAME, "firstName")

self.middle\_name\_textbox = (By.NAME, "middleName")

self.last\_name\_textbox = (By.NAME, "lastName")

self.save\_button = (By.XPATH, "//button[@type='submit']")

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

self.driver.find\_element(\*self.first\_name\_textbox).send\_keys(first\_name)

def enter\_middle\_name(self, middle\_name):

self.driver.find\_element(\*self.middle\_name\_textbox).send\_keys(middle\_name)

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

self.driver.find\_element(\*self.last\_name\_textbox).send\_keys(last\_name)

def click\_save(self):

self.driver.find\_element(\*self.save\_button).click()

WebDriverWait(self.driver, 10).until(

EC.visibility\_of\_element\_located((By.XPATH, "//div[@class='oxd-toast oxd-toast--success oxd-toast-container--top-right']"))

)

WebDriverWait(self.driver, 10).until(

EC.invisibility\_of\_element\_located((By.XPATH, "//div[@class='oxd-toast oxd-toast--success oxd-toast-container--top-right']"))

)

class EmployeeListPage:

def \_\_init\_\_(self, driver):

self.driver = driver

self.employee\_table\_rows = (By.XPATH, "//div[@class='oxd-table-body']/div[@class='oxd-table-row']")

# Adjust these column indices based on the actual table structure

self.first\_name\_column = 2

self.middle\_name\_column = 3

self.last\_name\_column = 4

def get\_employee\_names(self):

WebDriverWait(self.driver, 10).until(EC.presence\_of\_all\_elements\_located(self.employee\_table\_rows))

rows = self.driver.find\_elements(\*self.employee\_table\_rows)

employee\_names = []

for row in rows:

cells = row.find\_elements(By.XPATH, "./div[@class='oxd-table-cell']")

if len(cells) >= self.last\_name\_column:

first\_name = cells[self.first\_name\_column - 1].text

middle\_name = cells[self.middle\_name\_column - 1].text

last\_name = cells[self.last\_name\_column - 1].text

employee\_names.append(f"{first\_name} {middle\_name} {last\_name}".strip())

return employee\_names

def scroll\_down(self):

self.driver.execute\_script("window.scrollBy(0, document.body.scrollHeight);")

@pytest.fixture(scope="function")

def driver():

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()))

driver.maximize\_window()

yield driver

driver.quit()

def test\_orangehrm\_workflow(driver):

base\_url = "https://opensource-demo.orangehrmlive.com/web/index.php/auth/login"

login\_credentials = {"username": "Admin", "password": "admin123"}

employees\_to\_add = [

{"first\_name": "John", "middle\_name": "M", "last\_name": "Doe"},

{"first\_name": "Jane", "middle\_name": "", "last\_name": "Smith"},

{"first\_name": "Peter", "middle\_name": "K", "last\_name": "Jones"},

{"first\_name": "Alice", "middle\_name": "L", "last\_name": "Williams"},

]

added\_employee\_names = set()

# Initialize Page Objects

login\_page = LoginPage(driver)

dashboard\_page = DashboardPage(driver)

add\_employee\_page = AddEmployeePage(driver)

employee\_list\_page = EmployeeListPage(driver)

# 1. Login Flow

driver.get(base\_url)

login\_page.enter\_username(login\_credentials["username"])

login\_page.enter\_password(login\_credentials["password"])

login\_page.click\_login()

# 2. Navigate to PIM module -> Add Employee

dashboard\_page.hover\_over\_pim()

dashboard\_page.click\_add\_employee()

# 3. Add Employees

for employee in employees\_to\_add[:4]:

add\_employee\_page.enter\_first\_name(employee["first\_name"])

add\_employee\_page.enter\_middle\_name(employee["middle\_name"])

add\_employee\_page.enter\_last\_name(employee["last\_name"])

add\_employee\_page.click\_save()

added\_employee\_names.add(f"{employee['first\_name']} {employee['middle\_name']} {employee['last\_name']}".strip())

dashboard\_page.hover\_over\_pim()

dashboard\_page.click\_add\_employee()

# 4. Navigate to Employee List

dashboard\_page.hover\_over\_pim()

dashboard\_page.click\_employee\_list()

# 5. Verify Employees in the Employee List

time.sleep(2) # Give time for the list to load

found\_employees = set()

max\_scrolls = 5 # Limit scrolling to avoid infinite loops

scroll\_count = 0

while scroll\_count < max\_scrolls and found\_employees != added\_employee\_names:

current\_employee\_names = employee\_list\_page.get\_employee\_names()

for added\_name in list(added\_employee\_names - found\_employees):

if added\_name in current\_employee\_names:

print(f"Name Verified: {added\_name}")

found\_employees.add(added\_name)

if found\_employees != added\_employee\_names:

employee\_list\_page.scroll\_down()

time.sleep(1) # Small delay for content to load

scroll\_count += 1

if found\_employees == added\_employee\_names:

print("All added employees verified in the Employee List.")

else:

print("Not all added employees were found in the Employee List.")

print("Expected:", added\_employee\_names)

print("Found:", found\_employees)

# 6. Log Out from the Dashboard

dashboard\_page.click\_user\_dropdown()

dashboard\_page.click\_logout()