Генерация отчетов Allure

В windows powershell вводим команды:

Set-ExecutionPolicy RemoteSigned -Scope CurrentUser

iex (New-Object net.webclient).downloadstring('https://get.scoop.sh')

scoop install allure

убедимся в том, что всё в порядке командой allure

В Pycharm в settings добавляем два пакета: allure-pytest и allure-python-commons

import allure  
from allure\_commons.types import AttachmentType

в base\_page добавим импорты:

from selenium.webdriver import Keys, ActionChains  
from selenium.webdriver.support import expected\_conditions as EC  
from selenium.webdriver.support.ui import WebDriverWait as wait

и новую функцию:

def element\_is\_visible(self, locator, timeout=10):  
 return wait(self.browser, timeout).until(EC.visibility\_of\_element\_located(locator))

В LoginPageLocators

Добавим локатор

PROFILE = (By.XPATH, '//\*[@id="app"]/header/div')

В файле login\_page напишем функцию, которая возвращает нам искомый элемент -Profile

def find\_profile(self):  
 profile = self.browser.find\_element(\*LoginPageLocators.PROFILE)  
 return profile

И теперь в самом тесте напишем ассерт:

profile = page.find\_profile()  
assert profile

теперь перед тестом напишем декоратор с описанием фичи, которую мы тестируем:

@allure.feature('user\_login')

И ещё один декоратор, который конкретизирует данный метод:

@allure.story('Вводим валидный email и пароль')

Добавляем декоратор, который проставляет критичность теста.

Они бывают: blocker critical normal minor trivial

@allure.severity('blocker')

Добавляем отслеживание шагов:

with allure.step('Делаем скриншот')  
 allure.attach(browser.get\_screenshot\_as\_png(), name='result8', attachment\_type=AttachmentType.PNG)

Запускаем наш тест в терминале командой с указанием папки, куда будут падать отчеты:

pytest .\tests\test\_login\_page.py --alluredir results

Теперь идем опять в powershell, переходим в наш проект

cd .\PycharmProjects\

cd имя проекта

allure serve results