

Vizsgaremek- Segédanyag

Struktúraváltó képzés

Junior automata tesztelő szak

1.3.0 verzió

Módosítás leírása	Verziószám	Elvégezte	Dátum
Első változat	1.0.0	Józsa Tamás	2021.07.07.
Második változat, kiegészítések a pytest feladattal kapcsolatban	1.1.0	Józsa Tamás	2021.07.14.
Harmadik iteráció, kiegészítve a CI/CD, github actions feladat részletes leírásával	1.2.0	Józsa Tamás	2021.07.21.
Negyedik verzió, kiegészítve a prezentáció és védés információkkal	1.3.0	Józsa Tamás	2021.08.10.

Tartalomjegyzék

Bevezető	3
A vizsgaremek kivonata a Program Követelményből	3
Első hét teendői	4
Áttekintés.....	4
Teljes javasolt ütemterv áttekintése	4
Első hét részletes leírása	5
Második hét részletes leírása.....	5
Mit is jelent a pytest formátumra hozás?	6
Harmadik hét részletes leírása	7
Felkészülés a vizsgaremek védésre	10
Prezentáció	10
Védés menete	10

Bevezető

Ez a dokumentum abban hivatott segíteni, hogy a Képzési Programban kifejtett vizsgaremeket a hallgató megfelelő ütemezésben, magabiztosan el tudja készíteni. A hangsúly az ütemezésen van. Úgy gondoljuk, hogy ha a hallgató nagyjából a haladási vizsgája után kezd neki a vizsgaremek kidolgozásának akkor az alábbi ütemezés követésével nem lesz nehéz dolga a hallgatónak. A dokumentum célja, hogy jól tagolható részekre bontsa a vizsgaremek elkészítésének folyamatát.

A vizsgaremek kivonata a Program Követelményből

FIGYELEM: a mindenkor Program Követelmény az irányadó, ha közben megváltozott akkor kérlek az abban foglaltakkal helyettesítsd az itt található információkat!

A vizsgázónak a vizsgát megelőzően egy komplex web alkalmazás alapján felület tesztelési projektet kell elkészítenie, saját döntése alapján egy egyénileg választott web alkalmazás alapján.

A választott teszt alkalmazásnak legalább az alábbi funkcióit kell lefedni tesztekkel:

- Regisztráció
- Bejelentkezés
- Adatkezelési nyilatkozat használata
- Adatok listázása
- Több oldalas lista bejárása
- Új adat bevitel
- Ismételt és sorozatos adatbevitel adatforrásból
- Meglévő adat módosítás
- Adat vagy adatok törlése
- Adatok lementése felületről
- Kijelentkezés

A vizsgaremek benyújtásának módja:

A kész csomagot a vizsga előtt minimum 7 nappal kell a vizsgabizottsághoz benyújtani GitHub vagy más hasonló szolgáltatás segítségével megosztva. A megosztott anyagnak tartalmaznia kell az alábbiakat:

- A tesztek forráskódja
- A tesztelt alkalmazás elérési helye (GitHub vagy hasonló) és üzembehelyezési módja (Readme.MD vagy más dokumentáció az alkalmazás telepítése / elindítási módja)
- A tesztek dokumentációja a forráskódban és/vagy teszt dokumentációban (XLS vagy más táblázatos fájlok formájában)
- Tesztek futtatásának manuális és automatizált módja
- Valamilyen formában vezetői tesztjelentés

A vizsgaremek bemutatására és megvédésére maximum 15 perc áll a vizsgázó rendelkezésére.

Első hét teendői

Áttekintés

Az haladási vizsga hetén (első héten), bár a hallgató még nem feltétlenül érzi úgy, de már minden tudása meg kell legyen, hogy a vizsgaremeket sikeresen abszolválja.

Tekintsük át ezeket a tudásanyagokat:

- Manuális tesztesetek a választott alkalmazásra (példánkban a Conduit alkalmazás excel tesztesetek)
- A választott alkalmazás ismerete (példánkban a Conduit alkalmazás)
- A futtatás módjának ismerete és a futtatási környezet (docker, docker-compose)
- Python programozás környezet (python, pip, Pycharm, venv)
- Selenium webdriver környezet (python-selenium, webdriver-manager)
- Selenium és Python programozási alapok (selenium lokátorok, .send_keys(), .text())
- Python alapú teszt keretrendszer (pytest)
- Központi forráskód megosztó ismerete (git, Github)
- CI környezet ismerete (Github Actions)

Ezzel a tudással a birtokában a hallgatónak neki kell állnia a "projektnek" 😊

Teljes javasolt ütemterv áttekintése

- **5. képzési hét:** Első selenium python tesztek implementálása és futtatása lokálisan
 - a. Tesztek egyelőre csak sima python modulokba(fileokba)
 - b. Legalább a funkcionalitás fele legyen meg tesztekkel fedve
 - c. Küldjük be Github-ra
 - d. Ráfordítás kb 1 óra
- **6. képzési hét:** Hozzuk pytest formátumra a tesztek és futtassuk lokálisan pytest környezettel
 - a. Tesztek most már pytest formában kell, hogy megjelenjenek és csak pytest futtató környezettel futtatjuk őket a saját gépünkön
 - b. Legalább a funkcionalitás fele legyen meg tesztekkel fedve, de ha van időnk akkor adjunk még hozzá néhány tesztet
 - c. Küldjük be Github-ra
 - d. Ráfordítás kb 1 óra
- **7. képzési hét:** Github actions rendbehozatala, allure report generálás ellenőrzése
 - a. Az előzőleg lokálisan sikeresen futtatott pytest tesztek most akkor hozzuk futtatható állapotba a Github Actions YAML file rendbehozatalával.
 - b. Legalább a funkcionalitás fele legyen meg tesztekkel fedve, de ha van időnk akkor adjunk még hozzá néhány tesztet
 - c. Küldjük be Github-ra
 - d. Ráfordítás kb 1 óra
- **8. képzési hét:** Prezentáció összeállítása a védésre
 - a. Egyszerű PPT vagy bármi más, amit szeretnél, kb 3 képernyő, a lényeg összefoglalása
 - b. A hiányzó tesztesetek automatizálása
 - c. Küldjük be Github-ra
 - d. Ráfordítás kb 1 óra

Első hét részletes leírása

Na ez az a hét, ahol van már neked valamilyen működő alkalmazásod és sok tudásod, hogy tesztekét írd. Most mindegy, hogy a Github Actions YAML rendben van-e, mindegy, hogy milyen állapotban van a Github repo. Ami fontos, hogy kezdj neki az alkalmazás kiszemelt funkcionalitásának teszteléséhez.

Ehhez szükség lesz a nulladik, illetve első képzési héten elkészített teszteset leírásaidra. Ezért csináltad őket.

Ha ezek nincsenek meg akkor most számolj sokkal több időt mert ezeket meg kell írnod és be kell adnod a vizsgaremekkel együtt.

Ha megvannak akkor vedd elő és válassz ki 5 különböző funkcionalitást tesztelő tesztet. Olvasd át újra a Program Követelményben lévő felsorolást, hogy milyen funkcionalitást kell minimum lefedned. Ebben a listában szereplő funkcionalitást válassz.

A tanult módon hozzád létre egy új python file-t a projektben, vedd elő az első manuális tesztleírásodat és kodold le az első selenium-os tesztet. Oda hozod létre a file-t ahova akarod. A Github Actions miatt majd később aggódunk és bármikor át is lehet helyezni a file-t.

Folytasd a többi kiválasztott tesztel. Figyelj az időre. Bármennyi időt is szemeltél ki erre a hétre (javaslom az 1 órát) ott állj meg. Az idő elteltével commitold és pushold a tesztek a github repodba.

Ennek a hétnek a végére rendelkezned kell egy mappával a Github repoban (commitolva és pusholva) amiben egy vagy több python file-ok van. Ezek a file-ok megvalósítják a kiválasztott teszteseteket. Futtatni simán pythonnal kell őket, mondjuk picharm futtatás funkcióval.

Ha mondjuk én kihúzom a git repodat a saját gépemre és futtatón a python fileodate (vagy file-jaidat) akkor (feltéve, hogy a gépemen fut az alkalmazásod) le kell futnia a teszteknek.

Ha ez megvan akkor készen vagy az első héttel.

Második hét részletes leírása

Eddig sikerült (kézzel-lábbal) összehoznod pár tesztet. Lehet, hogy jelenleg egy Python forrás fájlba öntötted őket össze, lehet, hogy pont túl sok fájlod van már és nem látszód ki alóla. Egyik sem baj, mindent jól csináltál.

Érdemes ezen a héten a képzési programban meghirdetett minimum funkcionalitás felét lefedni. Ne feledd egy funkcionalitás már egy tesztel fedett. Itt a lista referenciaként:

- Regisztráció
- Bejelentkezés
- Adatkezelési nyilatkozat használata
- Adatok listázása
- Több oldalas lista bejárása

- Új adat bevitel
- Ismételt és sorozatos adatbevitel adatforrásból
- Meglévő adat módosítás
- Adat vagy adatok törlése
- Adatok lementése felületről
- Kijelentkezés

Ráérsz még az összeset megcsinálni, most kb a fele legyen meg. Ez abszolválható 5-6 teszttel. Ezen a héten folytatni fogod a tesztek írását, és arra foglak kérni, hogy szánj rá 5 percet, hogy pytest formátumra hozd a tesztjeidet.

Mit is jelent a pytest formátumra hozás?

Mondjuk egy tesztet szerkezetileg valahogy így néz most ki:

```
driver.get("https://react-card-2a6c5.web.app/")
time.sleep(2)

# buttons = driver.find_elements_by_xpath('//*[@class="shelf-item__buy-btn"]')
buttons = driver.find_elements_by_class_name("shelf-item__buy-btn")

for button in buttons:
    button.click()
    driver.find_element_by_class_name("float-cart__close-btn").click()
    time.sleep(0.35)

driver.find_element_by_class_name("bag").click()
time.sleep(0.5)
result_text = driver.find_element_by_class_name("sub-price__val").text
assert result_text == "$ 440.00"
```

Ennek a tesztnek a pytesztesített változata így néz ki:

```
def test_webshop():
    driver.get("https://react-card-2a6c5.web.app/")
    time.sleep(2)

    # buttons = driver.find_elements_by_xpath('//*[@class="shelf-item__buy-btn"]')
    buttons = driver.find_elements_by_class_name("shelf-item__buy-btn")

    for button in buttons:
        button.click()
        driver.find_element_by_class_name("float-cart__close-btn").click()
        time.sleep(0.35)

    driver.find_element_by_class_name("bag").click()
    time.sleep(0.5)
    result_text = driver.find_element_by_class_name("sub-price__val").text
    assert result_text == "$ 440.00"
```

Arra figyelj még, hogy lehet, hogy a teszted fájlnevét is változtatnod kell majd:

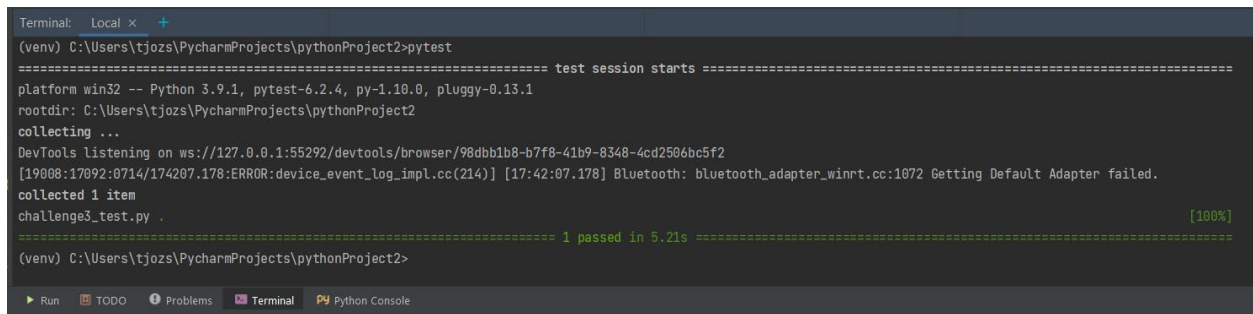
mywebshop.py → mywebshop_test.py

A siker érdekében viszont minden más korábbi próbálkozásodnál (egyéb python fileok a repóban) ne szerepeljen a test_ vagy a _test a fájlnevekben:

other_test.py → other.py

Ezek után úgy tudod leellenőrizni a pytestesítés sikerét, hogy a terminálban a projected mappájában add ki a pytest parancsot.

Hasonlót kell látnod:



```
Terminal: Local x +
(env) C:\Users\tjozs\PycharmProjects\pythonProject2>pytest
===== test session starts =====
platform win32 -- Python 3.9.1, pytest-6.2.4, py-1.10.0, pluggy-0.13.1
rootdir: C:\Users\tjozs\PycharmProjects\pythonProject2
collecting ...
DevTools listening on ws://127.0.0.1:55292/devtools/browser/98dbb1b8-b7f8-41b9-8348-4cd2566bc5f2
[19008:17092:0714/174207.178:ERROR:device_event_log_impl.cc(214)] [17:42:07.178] Bluetooth: bluetooth_adapter_winrt.cc:1072 Getting Default Adapter failed.
collected 1 item
challenge3_test.py . [100%]
===== 1 passed in 5.21s =====
(env) C:\Users\tjozs\PycharmProjects\pythonProject2>
```

Harmadik hét részletes leírása

Mostanára a saját gépeden rendesen futnak azok a tesztek amiket pytest formára hoztál. A pytest már ad valami fajta reportot. A következő lépés bizony az, hogy elővegyük újra a mumust, a github action-t.

Első lépés: ami fut a gépeden az fusson Github Action segítségével a github repodban automatikusan.

Ehhez idézd fel az első hét anyagát. Az egyik legfontosabb dolog, hogy a Github workflow-t össze tudd állítani.

Az alábbi lépéseket kell lefedned, hogy a conduit alkalmazás elinduljon, a selenium tesztek futni tudjanak pytest segítségével:

1. Legyen egy workflow yaml fileod: a conduit repoban a .github/workflows mappában hozz létre egy tests.yml file-t. Lehet más is a neve, csak legyen egy file-od.

Az alapvető keretet másold át a cicd-py-example workflow file-odból, mondjuk valami ilyesmi kell legyen:

```
name: Pytest_conduit

on:
  push:
    branches: [ master ]
  pull_request:
    branches: [ master ]
```

```
jobs:
  build:

    runs-on: ubuntu-latest

    steps:
    - uses: actions/checkout@v2
```

2. Ezzel a workflow-d addig már elérjük, hogy amikor módosítasz a repo-n akkor kiszedi a forrásokat. A következő lépés a teljes futtató környezet telepítése. Ez az alábbi elemekből kell álljon:
 - a. Python

```
- name: Set up Python 3.9.5
  uses: actions/setup-python@v2
  with:
    python-version: 3.9.5
```

- b. A python csomagok amiket használni fogunk (én a tests mappába tettem a requirements.txt-t és az alapján telepítem fel:

```
- name: Install dependencies
  run: |
    python -m pip install --upgrade pip
    if [ -f tests/requirements.txt ]; then pip install -r
    tests/requirements.txt; fi
```

- c. Nagy szükségünk van egy web böngészőre a gépen. Abban fogjuk futtatni a tesztjeinket

```
- name: Install Chrome
  run: |
    sudo apt install google-chrome-stable
```

- d. El kell indítanunk docker compose segítségével a conduit-ot

```
- run: docker-compose up -d
```

- e. viszont ez időt vesz igénybe, és azért, hogy ne egy fehér képernyőt próbáljunk tesztelgetni 45 másodpercig meg kell állítani a végrehajtást

```
- name: Sleep for 45 seconds
  run: sleep 45s
  shell: bash
```

3. Ha ez mind megvan akkor indíthatjuk a tesztjeinket pytest segítségével:

```
- name: Test with pytest
  run: |
    pytest
```

4. Pár tipp, hogy ne legyünk annyira elveszettek ha valami nem megy:
 - a. Bármilyen keletkező logfile-t, pl a docker konténerek logfilejait is össze tudja gyűjteni a github workflow:

```
- name: Collect docker logs
  uses: jwalton/gh-docker-logs@v1
  with:
    dest: './logs'
```

- b. A logs mappába file-ok kerülnek, amiben a napló bejegyzéseket lehet olvasgatni, mostmár csak fel kell töltenünk őket egy tárhelybe. Erre is van helyben megoldás


```
- name: Archive execution artifacts
  uses: actions/upload-artifact@v2
  with:
    name: docker-logs
    path: ./logs
```

- c. Ez nem mást fog csinálni, mint a web felületen a Github repo oldalon megtalálható lesz egy-egy futtatásnál a feltöltött zip file ami a logs mappa teljes tartalmát tartalmazza majd abból a futtatási körből:

The screenshot shows the GitHub Actions interface for a workflow named 'Update tests.yml Pytest_conduit #13'. The workflow was triggered by a push to the master branch and completed successfully. The summary shows one job, 'build (3.9.5)', which completed. The workflow file 'tests.yml' is shown with the trigger 'on: push' and a matrix build. The annotations section shows a warning about pip version 21.1.1 being used instead of 21.1.3. The artifacts section shows two artifacts: 'application-screenshot' (41.1 KB) and 'docker-logs' (14.6 KB).

Name	Size
application-screenshot	41.1 KB
docker-logs	14.6 KB

Az utolsó lépés amit meg kell oldanod, az az allure futtatás és a github pages deployment. Ezeket ha már sikerült a cicd-py-example vagy a selenium-py-peldatarban megoldanod akkor onnan csak át kell emelni a kódot, ha nem akkor most van itt a lehetőség, hogy ezt megpróbáld. Használhatod puszkaként az én egyik repomat is: <https://github.com/plresearch999/selenium-py-peldatar/blob/master/.github/workflows/main.yml>

Felkészülés a vizsgaremek védésre

A vizsgaremeked már minden bizonnyal a beadás határán vagy már az első átnézésen is túl van. Annyi dolgod maradt, hogy készítsd egy prezentációt és felkészülsz a védésre.

Prezentáció

Bármiben megcsinálhatod a prezentációt. Ott van például a Google Slides (<https://www.google.com/slides/about/>) vagy a Microsoft Power Point. Mindegy is mit választasz, csak legyen valami, amiben megmutatod mit értél el.

Egy jó prezentációnál figyelembe kell vened azt a tényt, hogy minél több információ van a dián, amit éppen mutatsz, annál hosszabb idő lesz elmondani, hogy mit is tartalmaz a dia.

Ökölszabály, hogy egy közepesen zsúfolt dia kb 3 percet igényel a prezentációból.

Neked 5 perced van a dia bemutatására.

Ennek megfelelően a következő diákat javaslom:

1. Neved és e-mailcímed, prezentáció címe (Junior automatizált tesztelő szakirány – Vizsgaremek védés)
 - a. max 1 perc
2. A választott alkalmazás rövid bemutatása. (egy képernyőkép a global feedről) Mellette felsorolva a tesztesetek nevei és hogy melyik milyen funkcionálisitást fed le.
 - a. max 1,5 perc
3. Automatizálás. Github action workflow, vagy workflow-k lépései és egy képernyőkép a lefutott workflowról
 - a. max 1,5 perc
4. Vezetői tesztjelentés képernyőkép és egy hivatkozás
 - a. max 1 perc
5. Köszönő dia, viszont látásra.
 - a. nem kell idő, csak kiteszi az ember a végére.

Védés menete

A védés MS Teams-en keresztül lesz. Az értekezlet helyszínén jelen lesz a bizottság és a levezető elnök, te pedig egyedül leszel bent a szobában Teams-en. Webkamera és képernyőmegosztás szükséges. Nincs mitől tartani, mert csak 5 pont-ot lehet kapni a védésre. Azért mindenképp meg kell jelenned és kérünk szépen, hogy tartsd be az időkorlátokat.