Testing Challenge

Ekibimizde yer alacak olan Yazılım Test Mühendisi adaylarından bir test otomasyon çalışması bekliyoruz. Çalışma
kapsamında aşağıda belirtilen senaryoların, test otomasyon kodunun belirtilen süre içerisinde yazılıp tarafımıza
gönderilmesi beklenmektedir.

Not: N11.e otomatik logini engelleyici mevcutsa hepsiburada veya trendyol gibi başka bir site de kullanabilirsiniz.

Scenario-1
1. Basic Login-Logout Test:
a) Successfully Login to N11.com and Logout (you can create a test account)
b) Unsuccessful Login attempt and logging of error message in loginerror.txt

2. Search keyword test:
a) Search a keyword at n11.com searchbox that can find results and display the search results in Browser
b) Search a keyword at n11.com searchbox that cannot find any results and display in Browser
c) Log results from a) and b) to results.txt file (should be search term and results)
d) Bonus: Include screenshot from successful (a) and unsuccessful (b) attempt

3. Parameterize test data:
Read key values from a config file (e.g. config.txt):
Username: xxxxx
Password: xxxxx
SearchTerm: xxxx

Notes:
– Test files should include Test asserts (Assert.xxx).
– Test Code should be written with Selenium Webdriver Framework for Java or C# (C# preferred but Java is OK)
- Test Code syntax should conform to Cucumber/Gherkin format (include feature files)
– All tests should run both in Firefox and Chrome browsers. Test cases for both browsers must be demonstrated.

Scenario-2
Create a load test script for YouTube.com search module. Search some value on search page and inspect behavior of service under load.
Create script with Jmeter/Locust (1Vu is enough)

Scenario-3
Meyve ve sebze satan bir işletmenin api servis testlerini yazdığınız düşünün,
-Elimizde bize stok fiyat bilgisi dönen bir endpoint GET /allGrocery

```
{  "data": [
{  "id": 1,  "name": "apple", "price": 3, "stock": 100
}, {
"id": 2,  "name": "grapes", "price": 5, "stock": 50
} ]
}
```

- İsme göre cevap dönen bir endpoint GET /allGrocery/{name}

```
{  "data": [
{  "id": 1,  "name": "apple", "price": 3, "stock": 100
} ]
}
```

- Yeni ürün ekleyebildiğimiz bir endpoint POST /add

```
{  "id": 4,  "name": "string", "price": 12.3, "stock": 3
}
```

Bu bilgilere göre REST ile en az 3 api testi yazınız. Farklı 200, 400, 404 gibi farklı http status kodları karşılayabilmesi iyi olur.
Not: Grocery için bir mock servis oluşturabilirsiniz