1. **General Instructions:**
    1. Submit the project by uploading the project to Github and share it with yuval.menashe@forcepoint.com. Please keep the solution confidential.
    2. The project should be written in Python 3.8 or higher.
    3. The project should contain a read me file which documents the project structure, requirements, instructions to run, dependencies etc.
    4. The project should log normal and exceptional events.
    5. The code should be simple and readable.
    6. Implement error handling where needed.
    7. Make sure your code does not crash.
    8. All questions will be located under the same project
    9. The FE automation question should use PyTest as a testing framework.
    10. Using PyTest capabilities such as data driven, PyTest API, hooks, conftest, etc is highly encouraged.
    11. The project should implement a basic reporter which will display a summary of the test results (it does not have to include screenshots / recordings).
    12. The project may use open-source libraries.

Goodluck!

Python Question

Overview

In this task, you will create a simple mechanism for allocating rides using an app which offers shared rides shuttles from a train station in NYC for companies who has offices in Manhattan area to allow employees to reach their work easily.
The app is using an external transit agency to operate these shuttles.

Those requests need to be aggregated and requested from a transit agency by calling an external API, which we do not control, that returns a list of approved rides. We should then distribute the approved rides to the companies that requested them.
The approved rides may not fully cover the request (e.g a company asked for 300 rides, but there are only 100 rides available). In this case, you will have to find a proper way to distribute the partial result between the requesting companies.

Task Description

To simplify things, the requested rides are received from a CSV file of the following format:

company_name, destination, number_of_rides_requested

For example, a file can be:

Microsoft, 11 times sq, 300
Uber, 175 Greenwich S, 700
Facebook, 770 Broadway, 200

Notice that each combination of companies and destination may only appear once.
Also, number_of_rides_requested is always a multiple of 100 (As shuttles are using double-decker buses which has 100 seats).

The request from the external transit agency is represented by the provided function:

request_rides(requested_rides : Dict[str, int]) -> Dict[str, int]:

Where both requested rides and returned result (“approved rides”) is a map from destination to int (the transit agency is unaware of the requesting companies).
If you find it more comfortable working with objects, you can modify the function accordingly instead of using a dictionary.

So, the expected flow of your function should be:

1. **Read all requests from the CSV file.**
2. **Aggregate all the requests for each destination.**
3. **Call request_rides with the aggregated result.**
4. **Distribute the result to the clients.**

For simplicity's sake, the "distribution" can be done by writing it to a file in the same format as the received request **(company_name, destination, number_of_rides_approved).**

Distributing results to companies:

The **request_rides** function may return only a subset of the requested rides. For example, when requesting:

{“11 times sq”: 500, “175 Greenwich S”: 400, “770 Broadway”: 700}
It may return:
{“11 times sq”: 500, “175 Greenwich S”: 345}

Your logic will then have to distribute this result to the companies in a way you conceive as fair. Use the following guidelines:

1. **If only part of the request was approved, the distribution of the results should be proportional to the requested sum.**
2. **Since companies ask for rides in chunks of 100 rides, any chunk not divisible by 100 is less useful. So, for example, giving 200 rides to one client and 110 rides to another is better than giving 180 rides to the first and 130 rides to the second (even if the second distribution is “fairer”) since it provides 3 full “rounds” of 100.**
3. **All approved rides should be distributed.**
4. **No company should receive more than requested.**

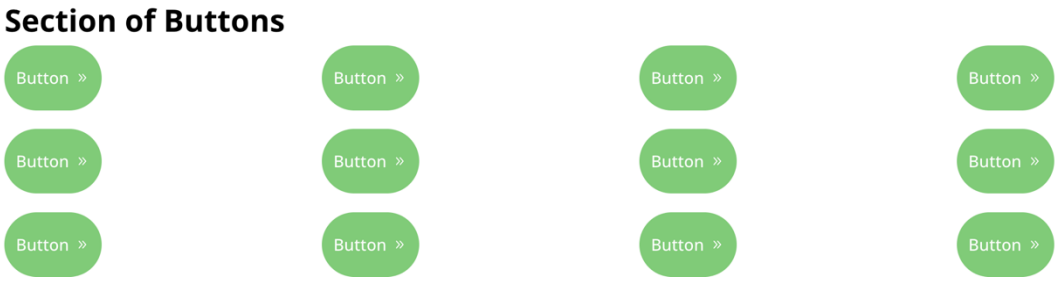
Important: you may add additional considerations as you see fit. Try to consider as many edge cases as possible. If you see specific scenarios which require more complex logic to resolve, feel free to write down what they are and general ideas to resolve them without doing the full implementation.

FE Automation Question.

Task Description:

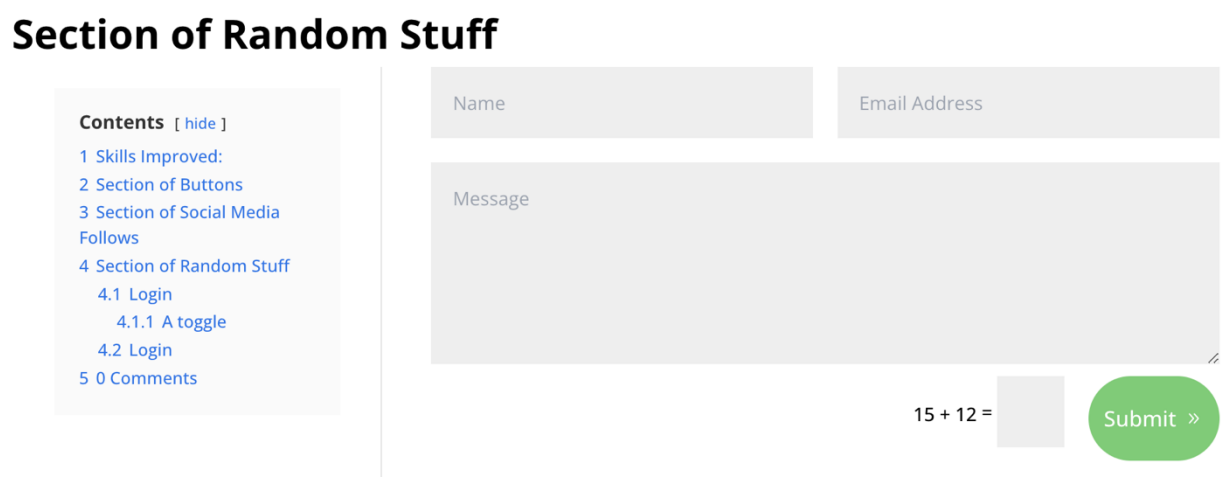
Verify the following behavior:

1. Navigate to: https://ultimateqa.com/complicated-page.
2. Count all buttons in 'Section of Buttons':



3. Verify all facebook button's href equal to 'https://www.facebook.com/Ultimateqa1/' in 'Section of Social Media Follows':

4. Fill in all fields (name / email / message / math exercise) in the form and click on submit in 'Section of Random Stuff':



5. Verify the message: “Thanks for contacting us” is displayed after submitting the form.