Full-Stack Code challenge

Step one: Perform a flight search from RUH - JED for 2 adults for any upcoming dates.

* Method: GET
* URI: <https://ae.almosafer.com/api/v3/flights/flight/search?query=RUH-JED/2023-11-20/2023-11-30/Economy/2Adult>
* Header params:
  + parameter name: Content-Type
  + parameter value: application/json
* Parse the JSON response and perform necessary assertions.

Step two: Retrieve the flight search results using json response from step one above.

* Method: POST
* URI: <https://ae.almosafer.com/api/v3/flights/flight/async-search-result>
* Header params:
  + parameter name: Content-Type
  + parameter value: application/json
* Request body: Full json response from step one above

[**Conditions**](https://github.com/tajawal/code-challenge/blob/master/QA_Full-Stack.md#conditions)

* Test scripts should be built using Rest Assured framework with Java, TestNG/Junit preferably. If you wish to use any other language/framework, please communicate this in advance.
* Perform needed assertions on the API response to ensure maximum test coverage.
* Generate test reports for executed scripts.

I skip this (I don’t have time to apply this)

## [Second Scenario:](https://github.com/tajawal/code-challenge/blob/master/QA_Full-Stack.md#second-scenario)

1. Navigate to​ a configurable base URL (almosafer.com)
2. Check for current set language. If language is already set to English then proceed with the next steps. If not, then first change language to English and then proceed.
3. Navigate to flights-home page, and enter the below criteria in flights search form to make flight search:
   * Origin - (from a random array of origins - length 5) (Example: DXB, AUH, SHJ, JED, RUH)
   * Destination - (from a random array of destinations - length 5) (Example: AMM, CAI, DEL, KHI, PAR)
   * Departure and Return Dates (randomly generated dates in the future)
   * Passengers (1Adult)
   * Cabin Class (Economy)
4. Wait for loading to be completed on the flight listing page
5. Use the sort feature to ensure that flights are sorted by 'cheapest'
6. After loading is complete, fetch and save the price of the first flight
7. Assert the minimum price displayed for the price-range-filter equals the price of first-flight in the list

### [Conditions:](https://github.com/tajawal/code-challenge/blob/master/QA_Full-Stack.md#conditions-1)

* Tests should be executed based on base URL passed in execution command.
* Code challenge should be done using Cypress + Javascript preferably. If you wish to use any other language/framework, please communicate this in advance.
* Perform the needed assertions on the UI components and pages to ensure maxiumum test coverage.
* Generate test reports for executed scripts.

HTML Cypress Report

