The Assignment

For the assignment, you have to implement a microservice with a single REST endpoint. This endpoint should receive a single keyword as an input and should return a score for that same exact keyword. The score should be in the range [0 → 100] and represent the estimated search-volume (how often Amazon customers search for that **exact** keyword). A score of 0 means that the keyword is practically never searched for, 100 means that this is one of the hottest keywords in all of **amazon.com** right now.

Example API Request (note: 87 is not the real score)

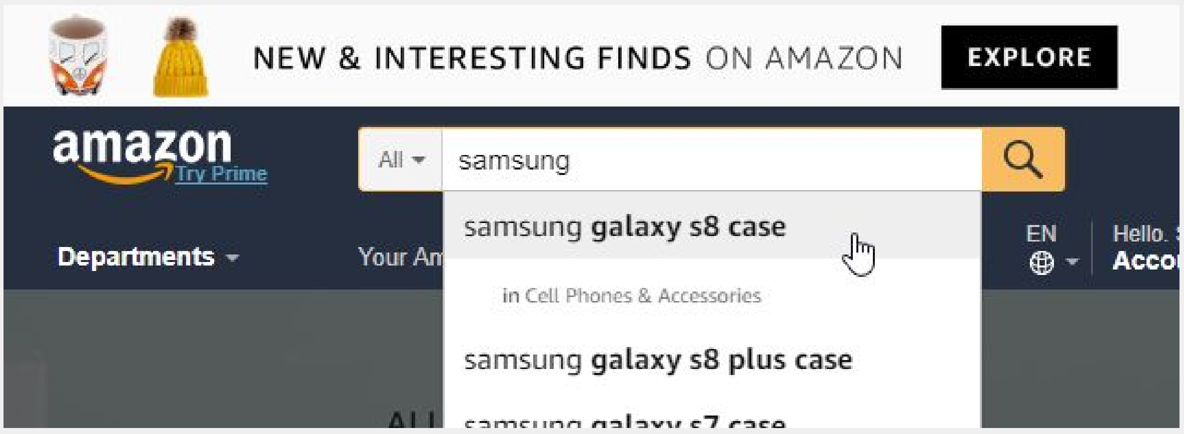
```
REQUEST GET http://localhost:8080/estimate?keyword=iphone+charger
RESPONSE
{
	“Keyword”:”iphone charger”,
	“score”:87
}
```

How are you supposed to do that?

Amazon do not provide any information on the search-volume of keywords. It is a well-kept company secret. Also, cross-references to search-volumes from other search-engines like google are not accurate, since the search behavior for product discovery is fundamentally different from general search. The only hint we get from Amazon about which keywords are important comes from their AJAX autocomplete API. You can reverse-engineer how it works by typing text into the search-box on amazon.com.

Amazon Autocomplete API

https://completion.amazon.com/search/complete


NEW & INTERESTING FINDS ON AMAZON
EXPLORE
amazon
Try Prime
All
samsung
samsung galaxy s8 case
in Cell Phones & Accessories
samsung galaxy s8 plus case
Departments
EN
Hello.

Your job is to extract a possibly precise estimation from the data you get from the autocomplete API by understanding the logic that Amazon applies when choosing the suggested keywords, reverse engineering it and finding a clever algorithm that utilizes the information to compute the search-volume score.

```
There is no hack or leak in the API. You will have to figure out a way to
work with the data that you get.
```

The microservice can not use any database or cache, all results have to be computed live. **Your microservice has an SLA of 10 seconds for a request round-trip**. You can perform as many calls to the autocomplete API as you want during those 10 seconds.

What you can assume about the Amazon API

Read this section carefully. The key to a good solution is in here. You might see some deviation from this description in the calls that you make, but for the scope of this assignment we will assume the following:

- For any search input, Amazon will only return up to 10 keywords, that have an exact prefix-match with the input.
- Any keyword with a relevant search-volume can be returned by the API.
- Whenever the API is called, it operates in 2 steps:
    1. Seek: Get all known keywords that match the prefix and create a Candidate-Set
    2. Sort/Return: Sort the Candidate-Set by search-volume and return the top 10 results.
- *hint: the order of the 10 returned keywords is comparatively insignificant!

What You have to deliver

- A document explaining the logic behind your approach
    - What assumptions did you make?
    - How does your algorithm work?
    - Do you think the (*hint) that we gave you earlier is correct and if so - why?
    - How precise do you think your outcome is and why?
- Buildable Java 8 code
    - Source code as zip, or checked in as git repo.
    - Use any frameworks / libraries you want
    - Javadoc is helpful
    - Tests are not necessary

How we will score it

We will rate your result based on the following: (first = most important)

1. Smart Idea / Algorithm
    a. Did you understand the assignment?
    b. Did you understand how Amazon’s API works?
    c. Did you draw the right conclusions?
    d. Did you think about all possible scenarios?
2. Clean Code
    a. Is it readable?
    b. Is it well structured?
    c. Does is actually work?
3. Project Setup
    a. Good structure and patterns?

Feel free to reach out in case of questions.

Good Luck