SEO Software Engineer Task

General notes:

Through this test, we would like to get a taste of your coding skills and also to see how you would approach and solve a simple problem. We are not looking for a perfect solution. We don’t expect you to spend more than 2-3 hours on this task. Please do not overthink it - we know that certain things definitely take more than planning.

Task spec:

We would like you to implement a simple class (you can also write a function if you like, depending on your own design) to find matching strings from a list of given strings without considering the order of the characters.

You will write a Python class which takes a list of strings in the constructor. This class will then have another function called ***find*** which takes a string to be matched. In the end this ***find*** function will return all the strings from the list which contain the EXACT same characters and number of characters as our given string. We do not care about the order of the characters in the strings, but only to find all the matching strings.

For example, the given list of strings could be as follows:

[“helloworld”, “foo”, “bar”, “stylight_team”, “seo”’]

Calling the ***find*** function with a parameter of “eos” should yield string seo. If more than one string in the string list matches the input, please return all of them.

Important instructions:

- Please state your assumptions. If you feel something is unclear, please just make an assumption and document it down so that we can understand.
- We hope you can use Python 3.8 or higher to implement this task.
- The codes should be clean and well documented. Please find the balance between comments and self-explanatory codes.
- Please minimise the number of external library dependencies as much as possible if you can.
- We value performance and efficiency so please make sure your codes are optimised as much as possible.

- Tests would also be much appreciated on our side. Feel free to choose any testing framework and write tests to “show off” the quality of your solution, its performance, its edge cases and so on.
- To better understand the solution, it would be great if you could add several bullet points explaining the approach in words. Please also give pros/cons of your approach and if there are, some use cases where it might fail.
- When the time is up, send us a zip archive of all source files and a `README` with instructions on how to build and run your app. Any other materials that can help us understand your approach are also welcome to be included.

Please also answer these questions:

- What do time complexity and space complexity look like in your approach? Can they be optimised?
- If the size of the initial string list is very large, would that influence the efficiency of the approach? And what if the number of “***find***” requests gets extremely large? Do you need to restructure/rethink the approach?