C# Pre-Interview Coding Test - Api

Welcome to the Reckon coding test!

Build the code as if this was just one of many tasks that you have to do in one day, using whatever libraries / frameworks you feel comfortable with unless explicitly instructed not to. The tasks are fairly straightforward and should take no longer than an hour, there are no major "gotchas" but there will be input and output that exist beyond the samples provided. When finished, please create a repository and push the unzipped, raw source to a public bitbucket / github repository and then submit a link to this repository. Please submit working c# / .net source code to solve the problem along with any supporting code that you might have used in testing.

Preface:

For the supporting infrastructure of the Api, use whatever patterns, frameworks, middleware, or libraries you prefer.

For the string matching part of the Api, Do **NOT** use any extended functionality or packages of c# to solve this problem. (For example, don't use the string finding methods of indexOf, Substring, Contains, regular expression classes, etc or any nuget modules related to string / text processing).

Importantly, the samples listed below should not be considered the only inputs and outputs.

Requirements

Overview

1. Given the two endpoints provided below, provide an Api written in C# running on http://localhost:9999/
2. Your new api will find all the occurrences of a particular set of characters in a string, using the two supplied endpoints as reference data
3. Your new api shoud post the result of this search to the supplied endpoint

Detail

We have two endpoints available that provide the text, and the subTexts to search for, and one endpoint to accept the results

The api endpoints provided can be quite unreliable, if it's the case that one of the endpoints fail, have the code try again until successful results are returned.

Endpoint1: https://join.reckon.com/test2/textToSearch provides the text to search in. An example output might be:

```
{
    "text": "Peter told me (actually he slurrred) that peter the pickle piper piped a pitted pickle before he pet
}
```

Endpoint2: https://join.reckon.com/test2/subTexts provides a list of subTexts that we need to search the output of Endpoint1 for. An example output might be

```
{
    "subTexts": [
        "Peter",
        "peter",
        "Pick",
        "Pi",
        "Z"
    ]
}
```

- The solution should match the subtexts against the text , outputting the positions of the beginning of each match for the subtext within the textToSearch.
- The set of characters can occur anywhere within the string.
- Multiple matches are possible
- Matching is case insensitive
- If no matches have been found, “<No Output>” is generated
- Your api endpoint should POST the results to the endpoint at https://join.reckon.com/test2/submitResults with the results in the format listed below.
- The api endpoints provided can be quite unreliable, if it's the case that one of the endpoints fail, try them again until succesful results are returned.

Expected Output

For the sample data listed above, this is the data that is expected to be processed by our endpoint.

```
{
    "candidate" : "<your name>",
    "text: "Peter told me (actually he slurrred) that peter the pickle piper piped a pitted pickle before he pet
    "results": [
        {
            "subtext": "Peter",
            "result": "1, 43, 98"
        },
        {
            "subtext": "peter",
            "result": "1, 43, 98"
        },
        {
            "subtext": "Pick",
            "result": "53, 81"
        },
        {
            "subtext": "Pi",
            "result": "53, 60, 66, 74, 81"
        },
        {
            "subtext": "Z",
            "result": "<No Output>"
        }
    ]
}
```