Document Converter Take Home

Task

Write an API to convert documents between three different formats:

Format #1: String

String data is composed of ‘segments’ (i.e. lines), each of which is composed of multiple ‘elements’ (i.e. data values).

Segments/lines are delineated by a line separator character, and elements within a segment are delineated by element separator characters. In the example below, the separator characters are **~** and *****.

Example

```
ProductID*4*8*15*16*23~
ProductID*a*b*c*d*e~
AddressID*42*108*3*14~
ContactID*59*26~
```

The example above is composed of 4 segments. Each segment is composed of a segment name followed by a number of elements. The first two segments have five elements, the third has four, and the fourth has two.

Format #2: JSON

Constraints:

- Segments (lines) are nested in arrays and objects where the keys are the segment names followed by an incrementing integer from 1 … # of elements.

Example

```
{
   "ProductID": [
      {
         "ProductID1": "4",
         "ProductID2": "8",
         "ProductID3": "15",
         "ProductID4": "16",
         "ProductID5": "23"
      },
      {
         "ProductID1": "a",
         "ProductID2": "b",
         "ProductID3": "c",
         "ProductID4": "d",
         "ProductID5": "e"
      }
   ],
   "AddressID": [
      {
         "AddressID1": "42",
         "AddressID2": "108",
         "AddressID3": "3",
            "AddressID4": "14"
        }
    ],
    "ContactID": [
        {
            "ContactID1": "59",
            "ContactID2": "26"
        }
    ]
}
```

Format #3: XML

Example

```xml
<?xml version="1.0" encoding="UTF-8" ?>
<root>
  <ProductID>
    <ProductID1>4</ProductID1>
    <ProductID2>8</ProductID2>
    <ProductID3>15</ProductID3>
    <ProductID4>16</ProductID4>
    <ProductID5>23</ProductID5>
  </ProductID>
  <ProductID>
    <ProductID1>a</ProductID1>
    <ProductID2>b</ProductID2>
    <ProductID3>c</ProductID3>
    <ProductID4>d</ProductID4>
    <ProductID5>e</ProductID5>
  </ProductID>
  <AddressID>
    <AddressID1>42</AddressID1>
    <AddressID2>108</AddressID2>
    <AddressID3>3</AddressID3>
    <AddressID4>14</AddressID4>
  </AddressID>
  <ContactID>
    <ContactID1>59</ContactID1>
    <ContactID2>26</ContactID2>
  </ContactID>
</root>
```

Requirements

- Your API should accept a document in any of the above formats and allow the user to specify the format they want to convert it to.
- If the user sends a document in format #1 (string) they will need to specify their separator characters.
- Your solution should:
  - Validate inputs
  - Include a number of tests
  - Work if we run it locally (provide instructions)
  - Demonstrate how you like to structure a production application (file/folder structure)

- You can use libraries to convert between formats #2 and #3 (JSON & XML), but the logic for converting to and from format #1 (string) must be completely written by you.
- You will need to demo that your solution works for the example input in format #1 (string) that is attached.

Notes

- Consider how extensible your solution would be if we introduced 3 additional formats.
- The examples for each of the formats above are equivalent to each other.
- Orderful’s back end is built in Typescript using Nest.js & Node/Express. Ideally, your solution will be built with something similar, but overall we’d prefer a well-built & well-understood solution using technologies you’re comfortable with over one where you’re less sure of yourself.

What to expect in the interview

- We’ll get you to demo your solution and walk us through the app (so make sure it runs!), and then we’ll talk about the code.