QA Challenge

As a QA, your responsibility will be to define a process to avoid regressions in a form generation system.

The system is an application that receives a JSON with variables and a DOCX template where those variables should be displayed on the form.

For this challenge, you will analyze a report, "74B" for a court in Ontario.

This document generation system consists of:

- Text that can be replaced by a variable.
- Paragraphs that can be shown or hidden depending on certain variable values.
- Checkboxes that appear selected or unselected depending on certain variable values.
- Tables containing lists of items that will replicate lines depending on the size of some array variable.

rcp-e-74b-0921 (7).docx

Delivery

- Describe which section of the form you think is important to automate
- Describe which section of the form you think is more efficient to do Manual test and why
- Create 3 different scenarios using Gherkin language
  - Testing different sections or functionality
  - Introduce any step definition you judge necessary to describe the test
- Automation code is not required for this challenge
  - if you want to validate your scenarios you can use the code below to help speedup your validation.

- You will prepare an presentation with maximum 30 min about this challenge

Step definition

```
const {Then} = require("@cucumber/cucumber");
const WordExtractor = require("word-extractor");
const fs = require('fs');
Then('file {string} is expected:', async function (file, table)
    const extractor = new WordExtractor()
    const DOCXBuffer = fs.readFileSync(file);
    const extracted = await extractor.extract(DOCXBuffer)
    const content = extracted
        .getBody()
        .split('\n')
        .filter((line) => /[a-zA-Z0-9]/.test(line))
        .map((line) => line.replace(/^[ ]+/, ''))
        .join('')
    const _table = table.hashes()
    const errors = []
    _table.map((row) => {
        const expects = row['Expects']
        const value = row['Value']
        if (expects === 'exists' && !content.includes(value)) {
            errors.push(`"${value}" was expected to exist on the
        }else if (expects === 'non exists' && content.includes(v
            errors.push(`"${value}" was expected to not exist or
        }
    })
    if (errors.length) {
        throw new Error(errors.join('\n'))
    }
})
```

Example of usage:

```
Scenario: 74B - Sample Scenario
  When file "rcp-e-74b-0921.docx" is expected:
    | Expects | Value
    | exists  | I have attached or caused to be attached to ea
    | exists  | AFFIDAVIT OF SERVICE OF APPLICATION FOR A CERT
```