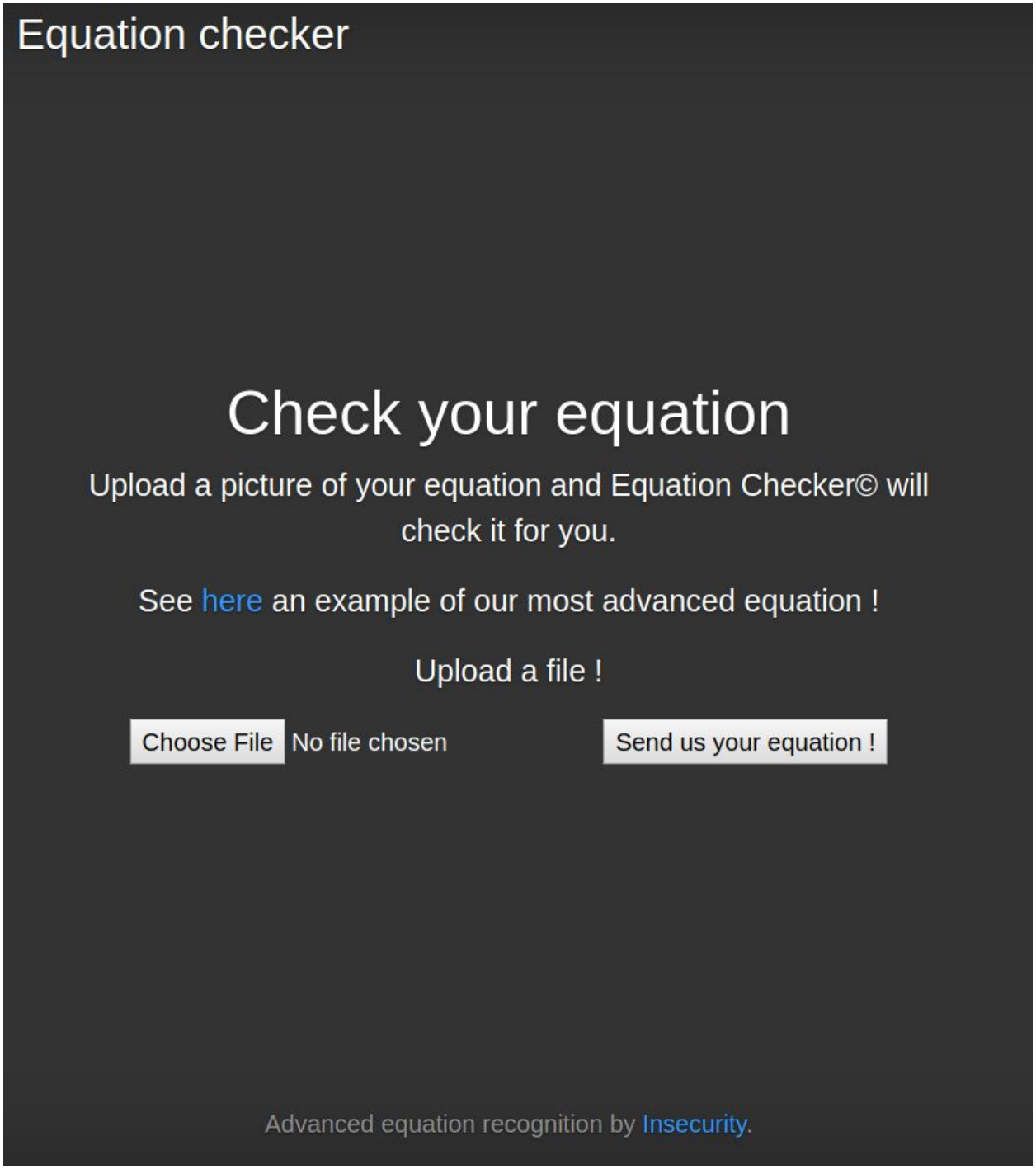
WEB: OCR

We are given the following web page:



If we click on “here”, we see an example of the equation. What the APP does is to get in input an image containing an equation and it calculates the result.

We can first inspect the web page and we can see immediately a hint:

```
  <script src="https://code.jquery.com/jquery-3.2.1.slim.min.js" integrity
</body>
<!-- TODO : Remove me : -->
<!-- /debug-->

</html>
```

Let's go on 127.0.0.1:5000/debug : the page contains the source code that executes the application!

```
#!/usr/bin/python3
from flask import Flask,request,send_from_directory,render_template,abort
import pytesseract
from PIL import Image
from re import sub
from io import BytesIO
app = Flask(__name__)
app.config.update(
    MAX_CONTENT_LENGTH = 500 * 1024
)
x = open("private/flag.txt").read()


@app.route('/',methods=['GET'])
def ind():
    return render_template("index.html")

@app.route('/debug',methods=['GET'])
def debug():
    return send_from_directory('./', "server.py")
```

It is a Python code. At the beginning, we can see that a variable called *x* is initialized, and this variable contains the flag. How can we access it?

```
if "==" in formated_text:
    parts = formated_text.split("==",maxsplit=2)
    pa_1 = int(eval(parts[0]))
    pa_2 = int(eval(parts[1]))
    if pa_1 == pa_2:
        return render_template('result.html', result = "Wow, it works !")
    else:
        return render_template('result.html', result = "Sorry but it seems that %d is not equal to %d"%(pa_1,pa_2))
```

All the magic is given by the function ***eval***, which is a well known function that allows injections-like codes. We can start analyzing the first block of the if-else statement. The equation is divided by the '=' simbol. So we have the left hand of the equation, and the right one. With the function *eval*, we execute the code contained in each side, and if the two sides are equal it returns the message "wow, it works!", otherwise it prints the content of the sides.

We need to further understand this concept. We can consider the first example:

1 + 1 = 3 - 1

The code splits in the two sides, (1 + 1, 3 - 1) and then it executes what is contained inside (2, 2).
These two numbers are the same, great.

I suggest having a look on this blog. We can exploit the function by injecting malicious info, for example, we can create the following images for inferring the first two chars of the flag:

ord(x[0]) = 0

ord(x[1]) = 0

In this way we are going to print the numeric value of the character 0 in the flag; since it won't be equal to 0, the program will print that content! We just need to create several 'ad hoc' images for the entire length of the flag, et voila, the flag is captured:

INSA{0cr_L0ng}