

Coding Interview for Graduates

Name : Suraj Kumar

Email : surajkumar150399@gmail.com

Phone : 7764946860

Github : <https://github.com/suraj-raj-3000>

Coding Challenge:

- 1) Create an API that lists the title, description based on the category passed as an input parameter.
- 2) Create an API that would save a new entry with all the relevant properties which retrieves values from the endpoint GET /entries.
- 3) Question: what are the key things you would consider when creating/consuming an API to ensure that it is secure and reliable?

Solution:

Prerequisites:

- Python
- Flask
- Thunder Client (To pass the arguments I am using thunder client.)

Software used:

- visual Studio code editor

Key things to ensure that it is secure and reliable:

Organizations have a lot to lose with unprotected APIs, so make security a priority and build it into your APIs as they're being developed. API security shouldn't be an afterthought or considered "someone else's responsibility."

CSV file is:

| | A | B | C | D |
|---|---|---|----|---|
| 1 | A | B | C | |
| 2 | 5 | 3 | 5 | |
| 3 | 7 | 8 | 6 | |
| 4 | 9 | 9 | 10 | |
| 5 | 4 | 5 | 9 | |
| 6 | 5 | 8 | 13 | |
| 7 | 2 | 6 | 8 | |
| 8 | | | | |

Program:

```
from flask import Flask
from flask_restful import Resource, Api, reqparse
import pandas as pd
import ast
app = Flask(__name__)
api = Api(app)
class Users(Resource):
    def get(self):
        data = pd.read_csv('users.csv') # read CSV
        data = data.to_dict()
        return {'data': data}, 200

    def post(self):
        parser = reqparse.RequestParser() # initialize
        parser.add_argument('A', required=True)
        parser.add_argument('B', required=True)
        parser.add_argument('C', required=True)
        args = parser.parse_args() # parse arguments to dictionary

        # create new dataframe containing new values
        new_data = pd.DataFrame({
            'A': args['A'],
            'B': args['B'],
            'C': args['C'],
        })
        # read our CSV
        data = pd.read_csv('users.csv')

        if args['userId'] in list(data['userId']):
            return {
                'message': f"'{args['userId']}' already exists."
            }, 401
        else:
            # create new dataframe containing new values
            new_data = pd.DataFrame({
                'userId': args['userId'],
                'name': args['name'],
                'city': args['city'],
                'locations': [[]]
            })
            # add the newly provided values
            data = data.append(new_data, ignore_index=True)
            data.to_csv('users.csv', index=False) # save back to CSV
            return {'data': data.to_dict()}, 200 # return data with 200 OK

api.add_resource(Users, '/users')

if __name__ == '__main__':
    app.run()
```

Output:

Status: 200 OK Size: 105 Bytes Time: 11 ms

Response Headers⁵ Cookies Results Docs

```
1  {
2    "data": {
3      "A": {
4        "0": 5,
5        "1": 7,
6        "2": 9
7      },
8      "B": {
9        "0": 3,
10       "1": 8,
11       "2": 9
12     },
13     "C": {
14       "0": 5,
15       "1": 6,
16       "2": 10
17     }
18   }
19 }
```

After Add parameters

GET http://127.0.0.1:5000/users?A=2&B=6&C=8 Send

Status: 200 OK Size: 178 Bytes Time: 11 ms

Query Headers² Auth Body Tests Pre Run New

Query Parameters

☒ A 2 ▾

☒ B 6 ▾

☒ C 8 ▾

☐ parameter value

```
1  {
2    "data": {
3      "A": {
4        "0": 5,
5        "1": 7,
6        "2": 9,
7        "3": 4,
8        "4": 5,
9        "5": 2
10     },
11     "B": {
12       "0": 3,
13       "1": 8,
14       "2": 9,
15       "3": 5,
16       "4": 8,
17       "5": 6
18     },
19     "C": {
20       "0": 5,
21       "1": 6,
22       "2": 10,
23       "3": 9,
24       "4": 13,
25       "5": 8
26     }
27   }
28 }
```

1. How will you tackle the challenge above?

- Using Python and Flask, I first construct an API endpoint, following which I use Pandas to read a CSV file, initialise a reparse, add a new data Frame, and append more CSV files. He adds the data frame into the csv file whenever I make a Post request with a parameter.

2. What type of errors you would you check for?

- I first verified that if a data frame departs, an error occurs, and then I verified that if the API is not operating, he returns a 401 Forbidden error.

3. How might a user break your code?

- Someone might easily damage my code by providing a bad parameter because my code's validation is incomplete.