In this session, you will create a RESTful API which can be used to create and delete user records. Responses to the questions should be recorded in your e-portfolio.

You are advised to use these techniques to create an API for your team's submission in Unit 6. Remember that you can arrange a session with the tutor during office hours for more support, if required.

Using the **Jupyter Notebook workspace**, create a file named api.py and copy the following code into it (a copy is provided for upload to Codio/GitHub): You can **install Jupyter Notebook on your local machine following these instructions** or via the **University of Essex Software Hub**.

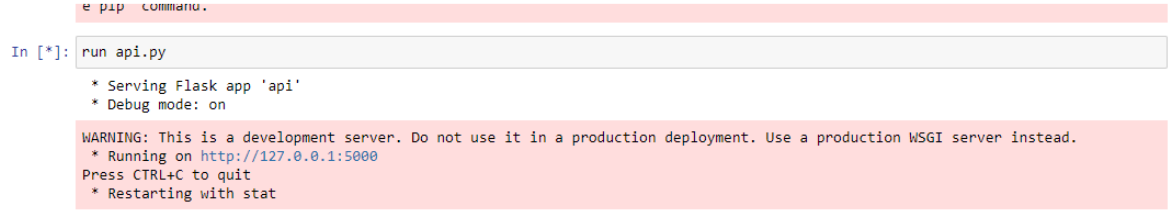
#source of code: **https://codeburst.io/this-is-how-easy-it-is-to-create-a-rest-api-8a25122ab1f3**

```
from flask import Flask
from flask_restful import Api, Resource, reqparse

app = Flask(__name__)
api = Api(app)

users = [
    {
        "name": "James",
        "age": 30,
        "occupation": "Network Engineer"
    },
    {
        "name": "Ann",
        "age": 32,
        "occupation": "Doctor"
    },
    {
        "name": "Jason",
        "age": 22,
        "occupation": "Web Developer"
    }
]

class User(Resource):
    def get(self, name):
        for user in users:
            if(name == user["name"]):
                return user, 200
        return "User not found", 404

    def post(self, name):
        parser = reqparse.RequestParser()
        parser.add_argument("age")
        parser.add_argument("occupation")
        args = parser.parse_args()

        for user in users:
            if(name == user["name"]):
                return "User with name {} already exists".format(name), 400

        user = {
```

```
            "name": name,
            "age": args["age"],
            "occupation": args["occupation"]
        }
        users.append(user)
        return user, 201

    def put(self, name):
        parser = reqparse.RequestParser()
        parser.add_argument("age")
        parser.add_argument("occupation")
        args = parser.parse_args()

        for user in users:
            if(name == user["name"]):
                user["age"] = args["age"]
                user["occupation"] = args["occupation"]
                return user, 200

        user = {
            "name": name,
            "age": args["age"],
            "occupation": args["occupation"]
        }
        users.append(user)
        return user, 201

    def delete(self, name):
        global users
        users = [user for user in users if user["name"] != name]
        return "{} is deleted.".format(name), 200

api.add_resource(User, "/user/")

app.run(debug=True)
```

Question 1

Run the API.py code. Take a screenshot of the terminal output. What command did you use to run the code?

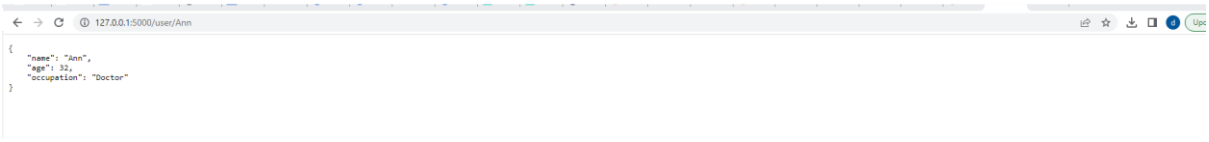


Question 2

Run the following command at the terminal prompt:
w3m **http://127.0.0.1:5000/user/Ann**

What happens when this command is run, and why?

Only Ann details are retrieved



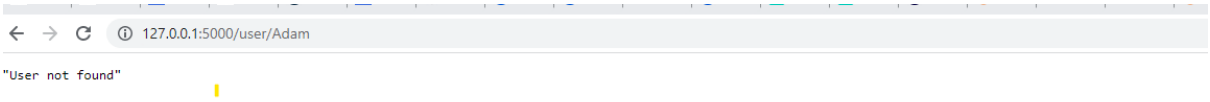
Question 3

Run the following command at the terminal prompt:

w3m **http://127.0.0.1:5000/user/Adam**

What happens when this command is run, and why?

The user Adam is not in the data set therefore an error has occurred.



Question 4

What capability is achieved by the flask library?

Flash provides a way to create a backend server with the flexibility to install extensions such database interface and encryption []wee, 2018]

L wee, 2018 This is how easy it is to create a REST API available from: https://codeburst.io/this-is-how-easy-it-is-to-create-a-rest-api-8a25122ab1f3 []accessed on 17 December 2020