

Exercise 2 –MAX 5p

PUT / PATCH, DELETE ,

1. PUT / PATCH + JSON. (3p)

Implement the following API interfaces using Flask. Use Postman for testing. You can continue the previous exercise or create a new project for this exercise.

First you need to create to global variables

app – flask application

allstudents – list of all students. Usually this is saved to database but we will come to databases later on.

```
from flask import Flask, request, jsonify, make_response

app = Flask(__name__)

allstudents = [
    {
        "student_number": 123123,
        "name": "Alice Brown",
        "credits": 130,
        "degree": "it"
    },
    {
        "student_number": 111222,
        "name": "Bob Jones",
        "credits": 157,
        "degree": "it"
    },
    {
        "student_number": 333444,
        "name": "Richard Brown",
        "credits": 57,
        "degree": "machine"
    }
]
```

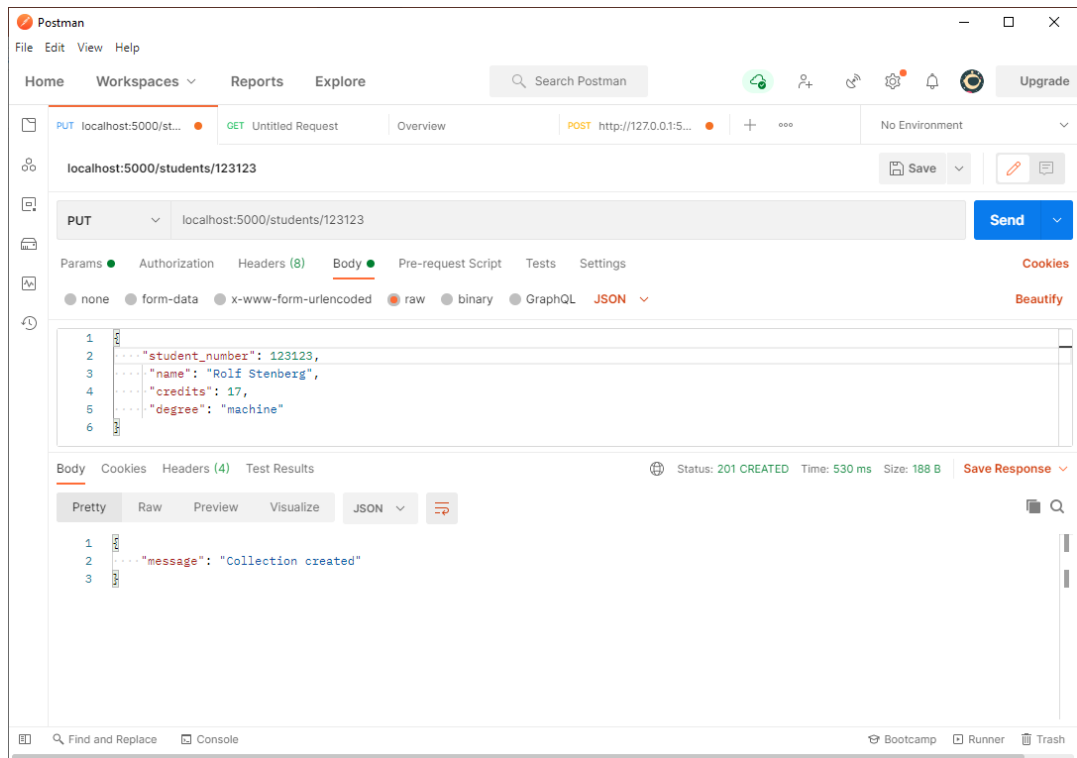
a. Create the route /students/<snumber>

Method: PUT

Data: data to be changed / saved in json format

Response:

- If student exists the response should include the json
{`"message": "Student replaced"`}, 200)
- If the student did not exist in the list, the student will be added and the following result is returned
(`{"message": "Collection created"}, 201`)

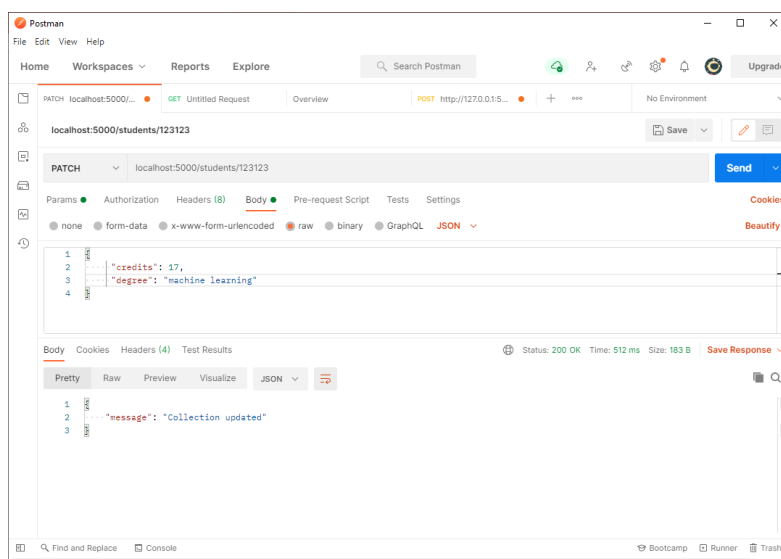


- b. Similarly implement API interface `/students/<snumber>` for PATCH method. The difference between PUT and PATCH is that in PATCH you only replace those fields which has been given in the json data

Metodi: PATCH

Response:

- If student exists the response should include the json `{"message": "Student updated"}, 200)`
- If the student did not exist in the list, the student will be added and the following result is returned `({"message": "New student created"}, 201)`



- c. Implement the route `/students/<snumber>`
Method: DELETE

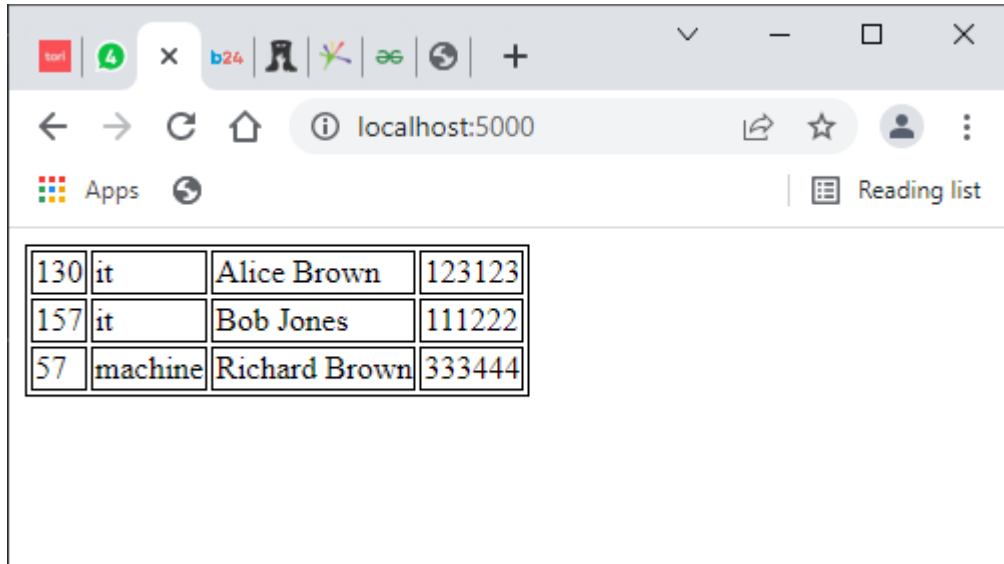
Deletes the student from the list

2. AXIOS, HTML / JAVASCRIPT, (Max 2p)

- a. Continue the previous flask implementation. Create the default route which returns index.html file.

```
@app.route('/')
def getHomePage():
    return render_template("index.html")
```

You should show all the students in the students in html page. You can do axios call from index.html file to backend and then show the results using vanilla JavaScript.



The screenshot shows a web browser window with the address bar set to `localhost:5000`. The page displays a table with student information. The table has four columns: an ID, a role, a name, and a phone number. The data is as follows:

130	it	Alice Brown	123123
157	it	Bob Jones	111222
57	machine	Richard Brown	333444

- b. Similarly implement the route which renders `addStudentsForm.html` page

```
3. @app.route('/add')
def showForm():
    return render_template("addStudentForm.html")
```

Send the form to backend route `/students` **using axios**. Method should be post. You can copy and paste the post method to backend implementation from the previous exercise.

When you receive the response you should empty the form and show the alert window with the text "Student added".

Student Number
112233

Student Name
Michael

Credits
32

Degree Programme
it

Add