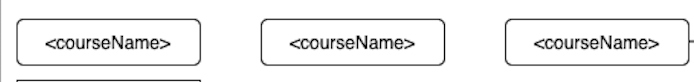
The goal of this assignment is to introduce you to using REST APIs and HTTP requests. For every action we are able to do on the front-end side, we want to call an API endpoint so the action will be handled by our back-end as well:

* Retrieve all the existing courses so we can display them in the top bar menu



and also render the students table based on the response

API endpoint: GET <http://localhost:3000/courses>

* Creating a new student in the UI should also save that new student in the database  
    
  API endpoint: POST <http://localhost:3000/students>
* Retrieve all the existing students so we can populate the student’s dropdown:

Graphical user interface, application, website

Description automatically generated

API endpoint: GET <http://localhost:3000/students>

* Whenever a student is being added to a course in the UI, the course’s students list should be also updated in the database

API endpoint: PUT <http://localhost:3000/courses/:courseId> (you should send the whole course object - in a real API back-end implementation, this case would have been covered in a different manner but due to the limitation which our mock REST API has we will stick to that)

* Whenever a student is being deleted from a course in the UI, the course’s students list should be also updated in the database:  
    
  API endpoint: PUT <http://localhost:3000/courses/:courseId> (same scenario as above)

Notes:

* We will be using the package json-server as our mock REST API – in the attachments you will find the db.json file which will serve as a mock database; please check the documentation of json-server in order to be able to install and start the mock server.
* In the db.json file, there are already some examples for both courses and students entities – feel free to adapt/change the object structure as needed based on your homework implementation
* 3000 port is the default port which the server will run on – if you run the server locally on a different port, please also adapt the API calls as needed.
* You can implement the API calls either by using fetch API or axios library (as we’ve discussed in the last meeting session) – please check the documentation for both of them and decide which one you’ll be working with.

References:

* <https://github.com/typicode/json-server>
* <https://developer.mozilla.org/en-US/docs/Web/API/Fetch_API/Using_Fetch>
* <https://www.npmjs.com/package/axios>
* <https://www.youtube.com/watch?v=6LyagkoRWYA&ab_channel=TraversyMedia>
* <https://www.youtube.com/watch?v=Oive66jrwBs&ab_channel=TraversyMedia>
* <https://www.youtube.com/watch?v=_8gHHBlbziw&ab_channel=DevEd>
* <https://www.youtube.com/watch?v=PoRJizFvM7s&t=1220s&ab_channel=TraversyMedia>
* <https://www.youtube.com/watch?v=iYM2zFP3Zn0&t=464s&ab_channel=TraversyMedia>
* <https://www.youtube.com/watch?v=Q-BpqyOT3a8&ab_channel=TraversyMedia>