

Restful Api - Node.js

Link github: https://github.com/BiancaGabriela06/restful_api_node

Proiectul conține 2 categorii de produse, telefoane și laptopuri, pentru care se utilizează metode HTTP precum GET, POST, DELETE, PUT și PATCH. Metodele se regăsesc în **routes/laptops.js**, respectiv **routes/telephones.js**.

Pentru baza de date am utilizat MySQL. Am instalat în proiect mysql2 (comanda **npm install --save mysql2**), iar în fișierul **database.js** m-am conectat la baza de date. Fișierul conține numele, parola, userul și host-ul.

Serverul (server.js) pornește aplicația pe portul 2255. Am utilizat Postman pentru teste.

Pentru metoda http PUT am folosit **bodyParser** pe care l-am integrat în ambele fișiere ale categoriilor de produse.

Pentru validări am folosit **joi** (comanda **npm install joi**) și am creat doua obiecte pentru fiecare categorie de produse, unul pentru validările metodei PUT, care să permită adăugarea fără parametrul de descriere (PhoneDesc / LaptopDesc) și unul pentru validările metodei UPDATE, unde este esențial ca body-ul să conțină și id-ul obiectului.

Pentru a porni aplicația în consolă: npm start

```
PS C:\Users\bianc\restful_api_node> npm start
> restful_api_node@1.0.0 start
> nodemon server.js

[nodemon] 2.0.21
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node server.js`
Connected to port 2255
DB Connected successfully
```

Metoda **POST** conține validări. Dacă lipsește descrierea, produsul se poate adăuga în baza de date. Dacă lipsește numele sau brand-ul, produsul nu se poate adăuga.

The screenshot shows the REST Client interface with a POST request to `localhost:2255/telephones`. The request body is a JSON object: `{ "PhoneName": "Xiaomi Redmi 9A", "PhoneBrand": "Xiaomi", "PhonePrice": 479.99, "PhoneDesc": "Baterie imensa de 5000mAh. Are o autonomie de 34 de" }`. The response status is 201, and the body contains the message "Telephone added".

The screenshot shows the Postman application interface. At the top, the request method is 'POST' and the URL is 'localhost:2255/telephones'. Below this, there are tabs for 'Params', 'Authorization', 'Headers (8)', 'Body', 'Pre-request Script', and 'Tests'. The 'Body' tab is currently selected and underlined. Under the 'Body' tab, there are radio buttons for different body types: 'none', 'form-data', 'x-www-form-urlencoded', 'raw' (which is selected), 'binary', and 'GraphQL'. The 'raw' body contains a JSON object:

```
{
  "PhoneName": "Samsung Galaxy A23",
  "PhoneBrand": "Samsung",
  "PhonePrice": 899.00
}
```

At the bottom of the interface, there are tabs for 'Body', 'Cookies', 'Headers (7)', and 'Test Results'. The 'Body' tab is selected. Below these tabs, there are buttons for 'Pretty', 'Raw', 'Preview', and 'Visualize'. To the right of these buttons is a dropdown menu set to 'HTML' and a red icon. At the very bottom, a status bar shows '1 Telephone added'.

The screenshot shows the Chrome DevTools Network tab. The selected request is a POST to `localhost:2255/telephones`. The body is a JSON object: `{ "PhoneName": "Samsung Galaxy A23", "PhonePrice": 899.00 }`. The status is 400 (Bad Request). The error message at the bottom of the page is "Invalid Request. PhoneName or PhoneBrand missing".

Metoda **Patch** actualizează un produs cu body-ul dat. Metoda **Put** actualizează și ea un produs, dar dacă acel produs nu se află în baza de date, îl adaugă.

PUT localhost:2255/laptops

Params Authorization Headers (8) **Body** Pre-request Script

none form-data x-www-form-urlencoded **raw** binary

```
1 {
2   ... "LaptopId": 1,
3   ... "LaptopName": "Asus X515",
4   ... "LaptopBrand": "Asus",
5   ... "LaptopPrice": 1499.00
6 }
7
```

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize HTML ↻

1 Laptop added

PATCH localhost:2255/laptops

Params Authorization Headers (8) **Body** Pre-request Script Test Results

none form-data x-www-form-urlencoded **raw** binary

```
1 {
2   ... "LaptopId": 1,
3   ... "LaptopName": "Asus X515 Edit",
4   ... "LaptopBrand": "Asus",
5   ... "LaptopPrice": 1499.00
6 }
7
```

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize HTML ↻

1 Laptop updated

Limit to 1000 rows

1 • SELECT * FROM apidb.laptops;

Result Grid Filter Rows: Edit: Export/Import:

	LaptopId	LaptopName	LaptopBrand	LaptopPrice	LaptopDesc
	1	Asus X515 Edit	Asus	1499	NULL
	22	Xiamo 12	Xiaomi Edit	4500	NULL
▶*	NULL	NULL	NULL	NULL	NULL

Metoda **GET** citește un produs după un id unic. Aici am făcut și căutarea după nume, brand, preț. Dacă req.query conținea nume și preț (de exemplu), atunci se afișau produsele cu numele și prețul respectiv. Dacă aveam și parametrul sign care putea fi >/ <, atunci se afișează produsele de la brandul respectiv, cu prețul mai mare/mai mic decât cel dat ca parametru.

GET localhost:2255/telephones?phoneid=24

Params Authorization Headers (8) Body Pre-request Script Tests Settings

Query Params

KEY	VALUE
phoneid	24

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "PhoneId": 24,
4     "PhoneName": "Samsung Galaxy A23",
5     "PhoneBrand": "Samsung",
6     "PhonePrice": 899,
7     "PhoneDesc": null
8   }
9 ]
```

GET localhost:2255/telephones?phonebrand=Iphone&&phoneprice=5000&&sign=>

Params Authorization Headers (8) Body Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```
1 [
2   {
3     "PhoneName": "Iphone X",
4     "PhoneBrand": "Iphone",
5     "PhonePrice": 5000.99
6   },
7   {
8     "PhoneId": 20,
9     "PhoneName": "Iphone 15",
10    "PhoneBrand": "Iphone",
11    "PhonePrice": 5599.99,
12    "PhoneDesc": "Good camera"
13  },
14  {
15    "PhoneId": 27,
16    "PhoneName": "Iphone X",
17    "PhoneBrand": "Iphone",
18    "PhonePrice": 5000.99,
19    "PhoneDesc": null
20  }
21 ]
```

Metoda **Delete** șterge un produs după id. Dacă produsul cu id-ul respectiv lipsește se va afișa un mesaj corespunzător.

DELETE

localhost:2255/telephones/2

Params

Authorization

Headers (8)

Body

Pre-request Script

☐ none

☐ form-data

☐ x-www-form-urlencoded

☒ raw

☐ binary

```
1 {
2   ... "PhoneName": "Iphone X",
3   ... "PhoneBrand": "Iphone",
4   ... "PhonePrice": 5000.99
5   ...
```

Body

Cookies

Headers (7)

Test Results

Pretty

Raw

Preview

Visualize

HTML

```
1 The telephone with id 2 is not in database
```

DELETE

localhost:2255/telephones/27

Params

Authorization

Headers (8)

Body

Pre-request Script

☐ none

☐ form-data

☐ x-www-form-urlencoded

☒ raw

☐ binary

```
1 {
2   ... "PhoneName": "Iphone X",
3   ... "PhoneBrand": "Iphone",
4   ... "PhonePrice": 5000.99
5   ...
```

Body

Cookies

Headers (7)

Test Results

Pretty

Raw

Preview

Visualize

HTML

```
1 The Telephone with id 27 was deleted
```