# Lab: HTTP and REST

Problems with exercise and homework for the ["JS Front-End" Course @ SoftUni.](https://softuni.bg/trainings/4240/js-front-end-october-2023)

**1. REST Countries**

**NOTE: Install** "[Postman](https://www.getpostman.com/)" REST Client to **ease** your tasks.

Your first task is to get detailed information about Bulgaria.

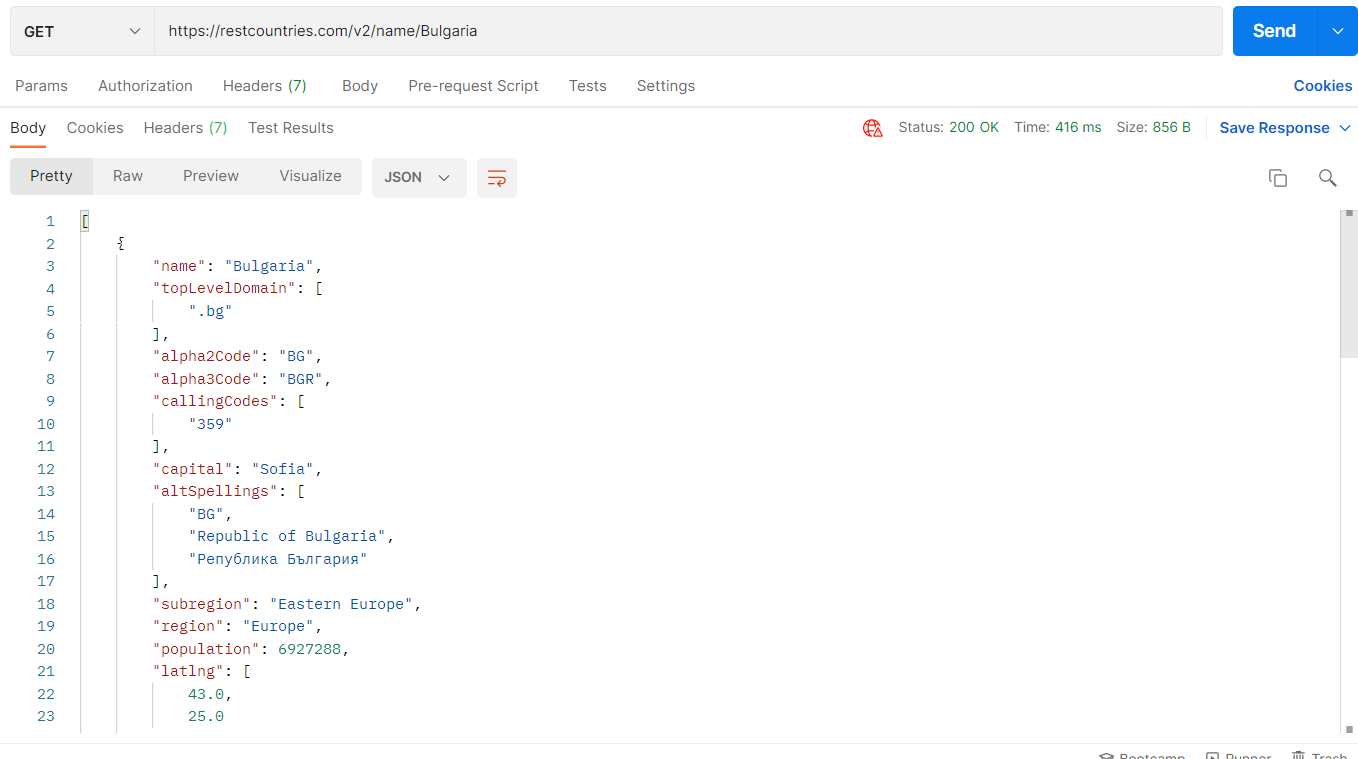
* Send a "**GET**" request to the link given below.
* **Copy** the response in JSON format.

**REQUEST**:

<https://restcountries.com/v2/name/Bulgaria>



**RESPONSE**:



## 2. GitHub Repos

Your task is to **write** a JS function that **loads** a github repository **asynchronously with AJAX**. You should **use the Fetch API.** Obtain the data by making a **GET** request to the following URL: **“https://api.github.com/users/testnakov/repos”**

Transform the **body** to **text** with **res.text()** and in the second **then()** block of the **Promise** replace the text content of a **div** element with **id "res"** with the value from the response. **Do not format** the response in any way.

### Examples

Shape

Description automatically generated

Text

Description automatically generated

## 3. Github Repos By Username

Your task is to **write** a JS function that **executes** an **AJAX** request with **Fetch API** and loads all user **github repositories** by a given username (taken from an input field with **id "username"**) into a **list** (each repository as a **list-item**) with **id** "**repos**". Use the properties full\_name and html\_url of the returned objects to create a link to each repo’s GitHub page. If an **error** occurs (like 404 "Not Found"), **append** to the list a list-item with **text** the current instead. Clear the contents of the list before any new content is appended. See the **highlighted lines** of the skeleton for formatting details of each list item.

### Examples

Graphical user interface, text, application

Description automatically generated

A picture containing text

Description automatically generated

## 4. Github Commits

Write a JS program that loads all commit messages and their authors from a github repository using a given HTML.

The loadCommits() function should get the username and repository from the HTML textboxes with IDs "username" and "repo" and make a GET request to the **Github API**:  
**https://api.github.com/repos/<username>/<repository>/commits**

Swap <username> and <repository> with the ones from the HTML:

* In case of **success**, for **each** entry add a list item (<li>) in the unordered list (<ul>) with id "commits" with text in the following format:

"<commit.author.name>: <commit.message>"

* In case of an **error**, add a single list item (<li>) with text in the following format:  
  "Error: <error.status> (Not Found)"

### Screenshots:

Graphical user interface, text, application, chat or text message

Description automatically generated

Graphical user interface, text, application

Description automatically generated

## 5. Book Library

First task is to "**GET**" all books. To consume the request with **POSTMAN** your **url** should be the **following**: **http://localhost:3030/jsonstore/collections/books**

Using the provided skeleton, write the missing functionalities.

Load all books by clicking the button "LOAD ALL BOOKS"



### Get Book

This functionality is not needed in this task, but you can try it with postman by sending request to "GET" the Book with id:" d953e5fb-a585-4d6b-92d3-ee90697398a0". Send a GET request to this URL:

**http://localhost:3030/jsonstore/collections/books/:id**

### Create Book

Write functionality to create a new book, when the submit button is clicked. Before sending the request be sure the fields are not empty (make validation of the input). To **create** a book, you have to send a "**POST**" request and the JSON body should be in the **following** format:

{

"author": "New Author",

"title": "New Title"

}

### Update Book

By clicking the edit button of a book, change the form like this:



The HTTP command "**PUT**" **modifies** an existing HTTP **resource**. The URL is:

**http://localhost:3030/jsonstore/collections/books/:id**

The JSON body should be in the **following** format:

{

"author": "Changed Author",

"title": "Changed Title"

}

### Delete Book

By clicking the delete button you have to delete the book, without any confirmation. To delete a book use "**DELETE**" command and send **REQUEST**: [**http://localhost:3030/jsonstore/collections/books/:id**](http://localhost:3030/jsonstore/collections/books/:id)

## Submitting Your Solution

Place in a **ZIP** file the content of the given resources including your solution. Exclude the node\_modules & tests folders. Upload the archive to Judge.



Картина, която съдържа текст

Описанието е генерирано автоматично

Картина, която съдържа текст

Описанието е генерирано автоматично