# Exercises: HTTP and Databases Basics

Problems for Exercise for the [“Software Technologies” course @ SoftUni](https://softuni.bg/courses/software-technologies). Submit your solutions in the SoftUni judge system at <https://judge.softuni.bg/Contests/969>. **Install** “[Postman](https://www.getpostman.com/)” REST Client to **ease** your task.

# II. HTTP

## GitHub Repos for User "SoftUni-Tech-Module"

Use POSTMAN to send a **"GET"** request to receive all the repositories, after that all you have to do is **copy the URL of the request** and paste it as a solution in judge.

Don’t forget that you should use the API provided from GitHub: **https://api.github.com**

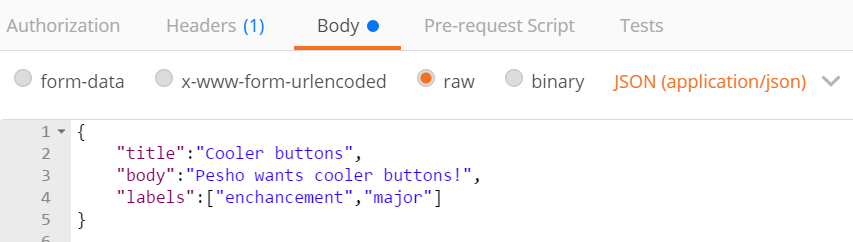
## GitHub Issue by id

Use POSTMAN to send a **"GET"** request to receive an Issue by id. Try getting issue with id 1 in repo "TestRepository" in "SoftUni-Tech-Module". Don't forget that you should use the API provided from GitHub: [**https://api.github.com**](https://api.github.com)**.**

**Post the HTTP code as a solution in judge.**

## GitHub Issue

Use POSTMAN to send a "POST" request to the server with the following JSON as body (send it as **application/JSON)**:



You need to use your GitHub **account credentials** to submit issues. Under the Authorization tab, select Basic and enter your username and password. Send the request to the URI <https://api.github.com/repos/SoftUni-Tech-Module/TestRepository/issues>

**Post the RESPONSE BODY as a solution in judge.**

## Patch Issue

After creating an Issue, mark down the id of your created Issue and send "PATCH" request, changing the title to "Ugly buttons".

**Paste the RESPONSE BODY as a solution in judge.**

# II. Databases

## Create Database

We are going to use HeidiSQL to create a database with MySQL server. HeidiSQL is a useful and reliable tool designed for web developers. With HeidiSQL we can connect to the MySQL server and execute SQL queries to create, read, edit and delete databases, tables and records.

Open HeidiSQL and CREATE DATABASE "softuni".

In judge copy the SQL query that was executed to create the database.

## Create Table

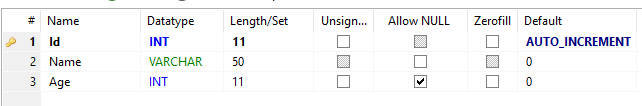
Using HeidiSQL in database "softuni" create table "students". You should specify three columns:

Name: id, Datatype: INT, Autoincrement, Primary Key

Name: Name, Datatype: VARCHAR

Name: Age, Datatype: INT, Allow NULL: true

Also set the id column as a primary key for the table.



Post the executed SQL query as your solution.

## Drop Table

Drop table 'students' using HeidiSQL. Use the executed query as a solution in judge.

## Drop Database

Drop database 'softuni'. Use the executed query as a solution in judge.

# III. Optional

## Firebase: All Books

Firebase is a cloud-based DB, **storage** and **app** platform (BaaS).

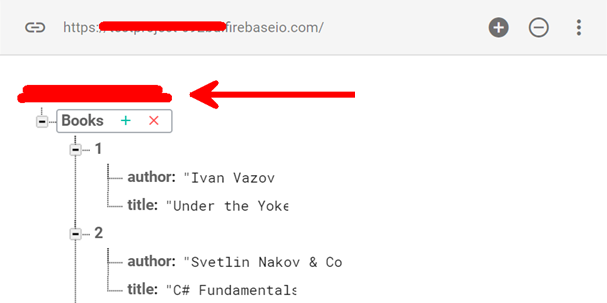
Register at: <https://console.firebase.google.com>.

Create a “**TestApp**” and in the create the **following** structure:

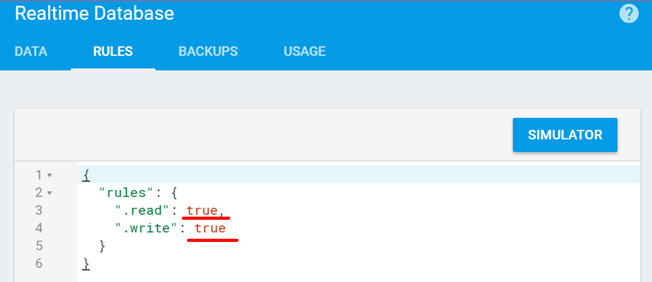


First task is to “**GET**” all books. To consume the request with **POSTMAN** your **url** should be the **following**: https://{databaseId}.firebaseio.com/.json.

**DatabaseId** is unique for every application. You can **find** yours from here:



We **should** also do one more configuration. Go to Database/Rules and set **.read** & **.write** actions to “**true**”. This will allow us to **send** request with **POSTMAN**. Beware that now everyone can **manipulate** our database and even **delete** it. (this is for **testing** purposes only).



## \*Firebase: Get Book #1

“**GET**” the Book with **id**: 1. Don’t forget the **.json** extension at the end (otherwise you will receive the whole **html**).

## \*Firebase: Create Book

To **create** a book, we will have to send a “**POST**” request and the JSON body should be in the **following** format:

{

"title":"New Title",

"author":"New Author"

}

## \*Firebase: Patch Book #7

The HTTP command “**PATCH**” **modifies** an existing HTTP **resource** (it can also create the resource if it does **not** exist). The JSON body should be in the **following** format:

{

"year": 1981,

"author": "Author Changed"

}

## \*Firebase: Change Book #7 Author

This time we have to execute a “**PUT**” command (the difference is that with “**PUT”** we can update a resource **partially**). In our case we have to **change** the author’s name to "**New author was assigned**".

**REQUEST**: https://{databaseId}.firebaseio.com/Books/7/author.json

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

"**New author was assigned**".