HOW TO USE THE BACKEND APPLICATION

# Install all the needed libraries

* First, install Node.js on the website: <https://nodejs.org> (either 18.17.0 or 20.5.0 it doesn’t matter),
* Then, install NestJS writing this command in the console: **npm install -g @nestjs/cli,**
* After, use the console to navigate to the backend folder and type: **npm install**, it should install all the libraries used in the code,
* Finally, start the application with the command: **npm run start.**

# All the available functions

|  |  |  |  |
| --- | --- | --- | --- |
| **Name of the Function** | **Route** | **Purpose** | **Format of the request expected** |
| **Functions in activity.js file** | | | |
| getallActivities | …/api/activity (GET) | Return all the activities of the database | None |
| createActivity | …/api/activity (POST) | Create a new Activity in the database | {Name: …,  Description: …} |
| findActivityByName | …/api/activity/Name (GET) | Return an activity using its name | None |
| getActivityWithExperimentName | …/api/activity/experiment/Name (GET) | Find the activity(ies) connected to an experiment | None |
| getActivityWithMeasurementID | …/api/activity/measurement/ID (GET) | Find the activity(ies) connected to a measurement | None |
| getActivityWithSubjectID | …/api/activity/subject/ID (GET) | Find the activity(ies) connected to a subject | None |
| **Functions in device.js file** | | | |
| getallDevices | …/api/device (GET) | Return all the devices of the database |  |
| createDevice | …/api/device/sensor (POST) | Create a new Device with the sensors connected to it in the database |  |
| modifyDevicebyID | …/api/device/edit/ID (PUT) | Modify a device’s parameters and the parameters of the sensors connected to it |  |
| linkDeviceWithExperiment | …/api/device/experiment (POST) | Create a link between a device and an experiment |  |
| findDeviceByID | …/api/device/ID (GET) | Return a device using its ID in the URL |  |
| findDeviceByDescription | …/api/device/Description (GET) | Return a device using its description in the URL |  |
| getDeviceWithDeviceTypeName | …/api/device/devicetype/Name (GET) |  |  |
| getDeviceWithSensorSignalID | …/api/device/sensorsignal/ID (GET) |  |  |
| getDeviceWithExperimentName | …/api/device/experiment/Name (GET) |  |  |
| getDeviceWithSensorID | …/api/device/sensor/ID (GET) |  |  |
| getDeviceWithMeasurementID | …/api/device/measurement/ID (GET) |  |  |
| unlinkDeviceWithExperiment | …/api/device/experiment (DELETE) |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# How to add a new function

To add a function in an existing file, like for example “activity.js”, it must be in the Activity class and written like in the example code below:

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

* It is possible to write a function without entry parameters.
* It is also possible to directly use the parameter of the function as a variable if it is in the URL of the request.
* It is possible to use every name for the parameter of the function, but the code is written as this: the parameter is request if the information is from the body of the request, and it is params if the information is from the URL of the request.
* A function can use parameters from the body of the request and the URL at the same time, it is coded with request and params as parameters of the function.

Then, after creating the function, the route needs to be created. To continue with the example in the “activity.js” file, the route will be defined in the “activitycontroller.js” file, in the ActivityController class.

Une image contenant texte, capture d’écran, Police

Description générée automatiquement

This is an example of a route for a function using the body of the request and parameters from the URL. @HTTPMETHOD is the method used by the request (POST, PUT, GET…), then addedrouteifnecessary is if the function needs to use a new URL route, the basic one being ‘…/api/activity’ (in the activity files) and it will be like this ‘…/api/activity/ addedrouteifnecessary/parameterwanted’ in the case of the code example.

Then, @Bind is here to say that the function will use the Req() and Param() functions in this case to get all of the request with Req() and the parameter in the URL with Param() and stock them respectively in the request and params variable. Then it calls the function defined in “activity.js” with the request and what was written in the URL as parameters.

Une image contenant texte, capture d’écran, Police

Description générée automatiquement It is also possible to send a response and it is written like shown below.