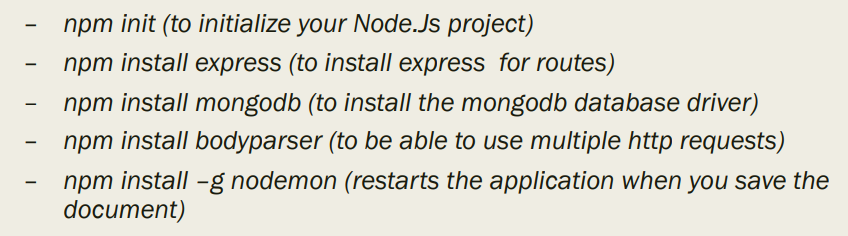
Gautier Lemaitre 22870 <https://github.com/Gautier78/back_end_dorset>

**1)Set up the environment**

**In our folder project:**

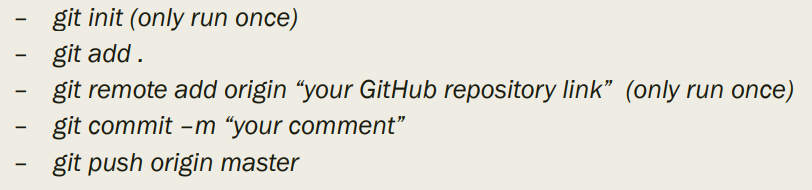
First, we must set up our project and the environment. We have multiple commands to write in the terminal:



**On Github:**

We create a new repository on Github, the mine is named “back\_end\_dorset”

Again, we have multiple commands to write in the terminal:



**On GitPod:**

In my case, I didn’t use GitPod. We can use it if we need help. Just write “gitpod.io/# “ before your repository link. So, anyone that have access can modify your code.

**On MongoDB:**

Once logged, you can create a new project on the top left corner. In this project you must create a new cluster. In this cluster, select AWS Cloud, the Ireland region and choose the free options. You can rename the cluster.

You will have the option to change the ip address. Use 0.0.0.0/0 or your Own ip to allow the connection with the database.

Then choose the second option “Connect your application” and copy/paste the long link in your main JavaScript file of your project.

The link look like this:

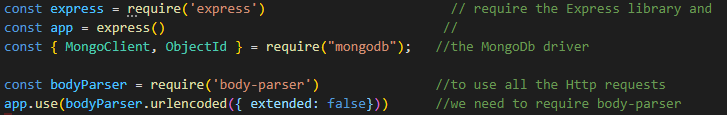
mongodb+srv://gautier:<password>@clusterbackend.wcefp.mongodb.net/<dbname>?retryWrites=true&w=majority

In yellow, this is the name of the admin, in green you have the password and the database name to replace

**2) Let’s start coding**

**Code/template breakout:**

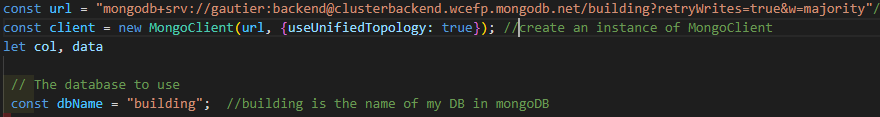
We have been using this code at each backend course:



It allows us to use the Express library, the MongoDb driver and Http requests.

**Connection to the Database:**

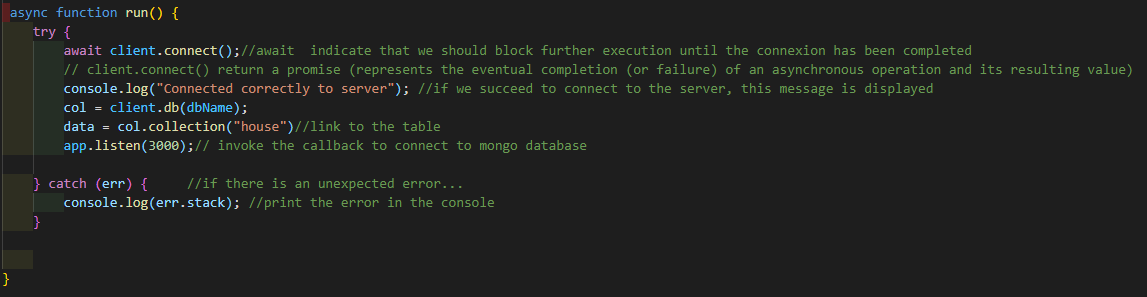
We can see again the long link. Just under I created an instance of Mongo Client and a function dbName equal to “building” because it’s the name that I’ve given to my database.



**Async function Run():**

wrap the calls to functions that interact with the database in a try/catch statement so it can handle any unexpected errors.

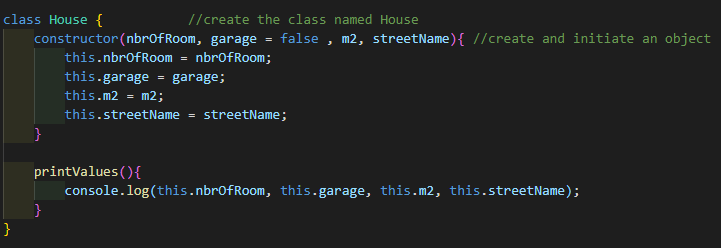
If the connection succeed, the function run the code in the Try but if the connection fail, the function run the code in the catch.



**The class**

Create a class and give it a name (in UpperCamelCase). In this class we build a special method named constructor, to create and initiate an object.

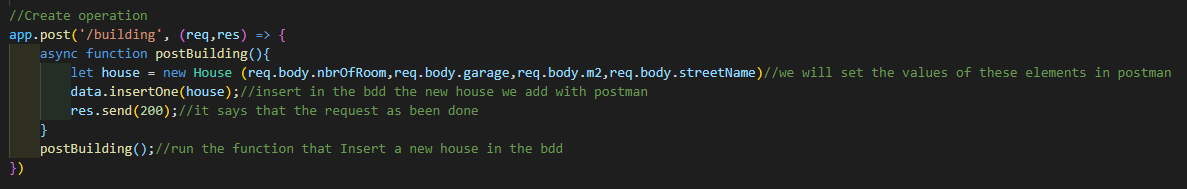
The parameters of the constructors are the column names that fills the table “house”.



**Crud Operations**

Create:

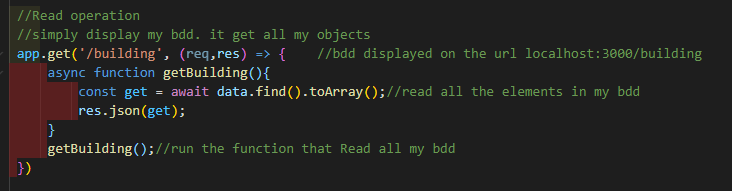
I create a new house and I define on Postman the nbrOfRoom, if there is a garage, the number of square meter and the name of the street. Once I have set the values on Postman, I send the request and it create a new house.



Read:

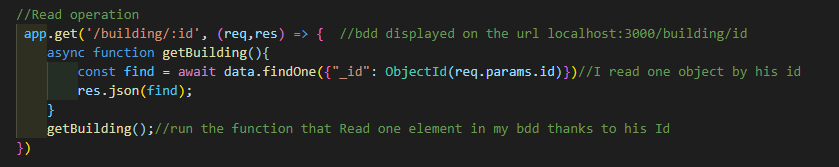
* All:

On the Url “building” all my database is read.



* One:

On the Url “building/id” replace the “id” in the url by a real id number and it will read the house link to this id.



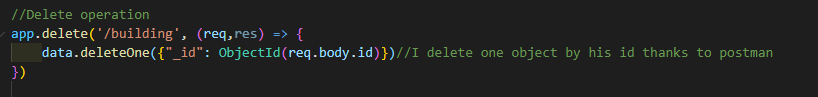
Update:

In postman, you have to select the id of the house you want to update. Then, you have to modify the option that you want to change. For exemple, you can select numberOfRoom that was equal to 6 before the update. You set it to 4 and then send the PUT request with postman. You will see in your database that the value has been updated.



Delete:

You can delete with postman a house thanks to his Id number. Once you have choose, send the request DELETE and it’s done.



**Sources:**

* The link underneath helped me to comment my code especially for the code that is about the database connexion or operations.

<https://www.mongodb.com/blog/post/quick-start-nodejs-mongodb--how-to-get-connected-to-your-database>

* To understand what app.listen(3000) do.

<https://stackoverflow.com/questions/55984104/connection-to-mongodb-server-using-node-is-unresponsive-on-localhost>

* Crud operations

<https://developer.mongodb.com/quickstart/node-crud-tutorial#std-label-node-tutorial-update>