Installation Guide DogWalker_API

By Gabriel Alexander Maldonado Rodriguez

Table of contents

1 Introduction

1.1 Purpose

This document is provided an installation guide for the DogWalker_API restful Api develop with the Express JS framework. To provide a comprehensive guide for the testing and deployment in production of this API.

The guide is mean to users with basic understanding of node.js cross-platform JavaScript runtime environment , server deployment, basic use of relational databases and de SQL computer language.

1.2 Scope

The document will provide and step by step guide to install , test and deploy the DogWalker_API restful Api. It will run down the user to the necessary libraries , packages , application, and tools require for the API.

Links will be provided with the sites to download the require tools and additional tutorials will be supplemented for the installation and configuration of this tools. For the rest of the guide pictures of the expected result, console outputs and example configuration files will be provided.

2 Installation Manual

2.1 Pre-requisites

2.1.1 Compatible operating system:

- Windows 11

Any operating system compatible with node.js should been able to run the API. But not all tools might be supported by its Operating System (OS). The complete list of compatible OS for node.js and installation guides are provided here:

https://nodejs.org/en/download/package-manager

The guide will cover only Windows 11

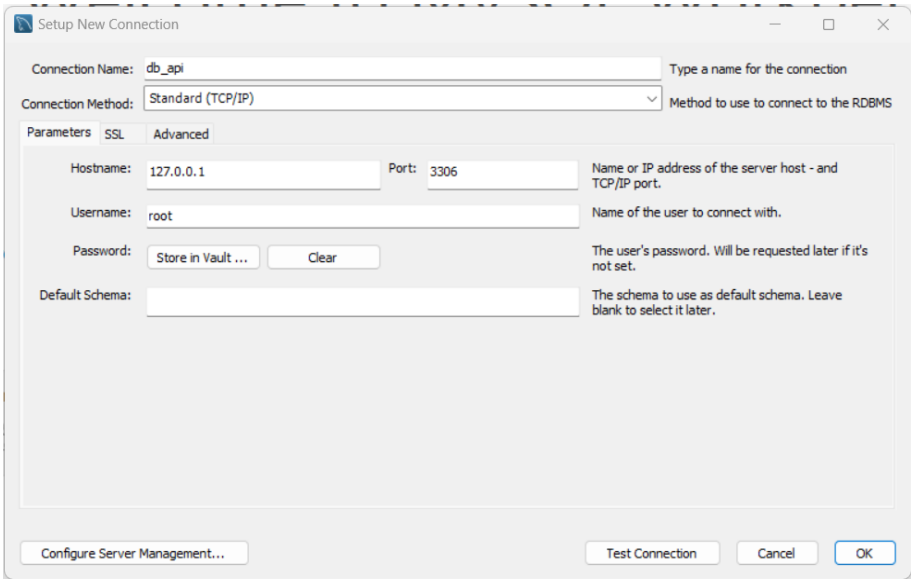
2.1.2 Software components:

Database: MySQL 8.0

- Windows: Download and install MySQL server following the installer: https://dev.mysql.com/downloads/installer/
- Ubuntu: Use the Linux shell install MySQL server using this tutorial: https://www.digitalocean.com/community/tutorials/how-to-install-mysql-on-ubuntu-20-04

MySQL Workbench is also require running SQL scripts and ease of the manager of the data base. For booth OS, the app can be download at https://dev.mysql.com/downloads/workbench/.

Ones the installation proses is competed , a new MySQL connection must be created via the Workbench with the password , user , hostname, and port of your preference. In case that your MySQL database is deploy elsewhere , it is still required to have a local database to test the API.



Picture 1 Set up connection window example

Run time environment: Node.js 18.16.1

- Windows: Download node.js throw https://nodejs.org/en/download and complete the installation procedure. The result should install node.js and npm. Check the installation proses running in the Windows console:

npm -v

node -v

Daemon Process Manager: PM2 9.7.1

To download PM2 Node.js and npm must be installed in the computer. For booth OS consoles run:

npm install pm2 9.7.1

And check the version with:

npm -v

2.1.3 Frameworks:

All the require frameworks can be installed using npm in the command console regardless of OS. They are necessary to run some command for testing and seeding

ORM Model: Sequilize 6.31.1

npm install -g sequelize-cli 6.31.1

npm install -g sequelize 6.31.1

Testing Framework: Jest 29.5.0

npm install -g jest 29.5.0

Testing report Generator: jest-unit

This library will be require generating test reports for the API and it will require the installation of additional tools:

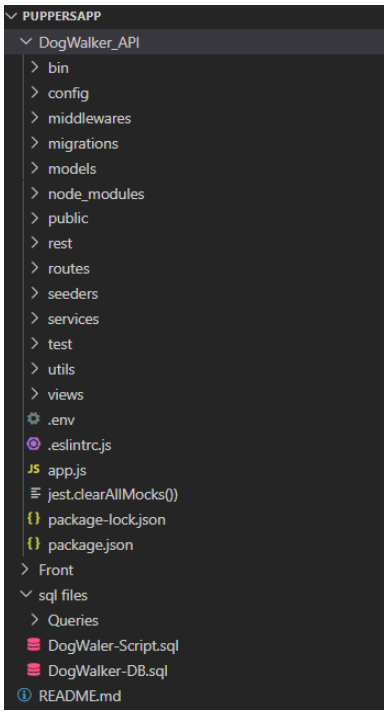
npm install jest-junit

npm install --global yarn

yarn add --dev jest-junit

2.2 Pre-installation Tasks

Check all the versions of the previously downloaded apps , tools, and frameworks. The wrong version can cause unexpected unacceptability issues between the Api and the computer.



Picture 2 Project File directory

Use a code Editing tool of your choice to open the “PuppersApp/DogWalker_API” in

the back end project folder containing and in the computer console open de route for the DogWalker_API folder.

For this program the recommended Editing tool is Visual Studio Code: https://code.visualstudio.com/download .

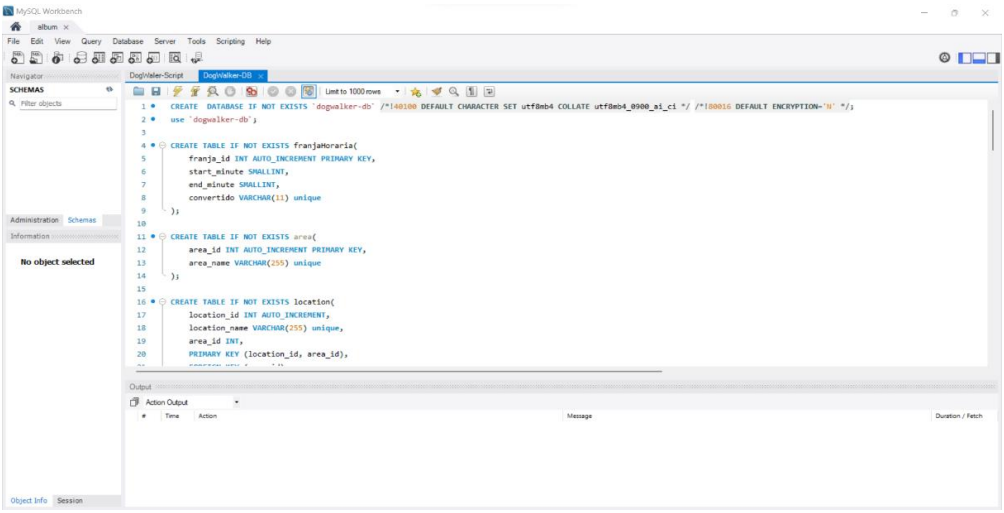
In addition, it is recommended to download this tools from the “Extension Marketplace”:

- expressjs
- npm
- npm Intellisense
- Sequelize Snippets
- JavaScript (ES6) code snippets

2.3 Installation Procedure

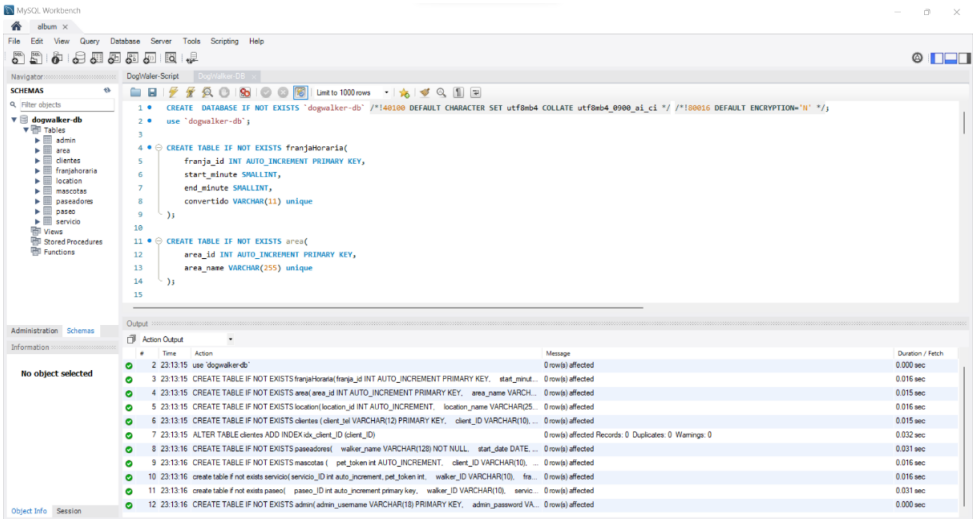
2.3.1 Database

1. In the local database open MySQL Workbench. In the previously created connection open the SQL script PuppersApp/sql files/DogWalker-DB.sql on the project file directory



Picture 3 DogWalker-DB.sql examples

2. Run the script. The result should be a new data base call dogwalker-db



Picture 4 DogWalker-DB.sql execution

This database is only for testing purposes. The one for deployment must be created with the same script in another connection or if possible in a separate server running

SQL. The provided script must be used to create the database structure. Changes in names will cause error for the API.

For the deployment database, it is necessary that the database must be populated with data. An SQL script is provided at PuppersApp/sql files/DogWalker-Script.sql of the file directory.

In case you have your own data. Consider that all password fields in paseadores , clientes and admin must be added to the database as a sha256 of realPassword + salt .Were salt is a field of the tree tables and is a 20 characters alphanumeric sequence.

2.3.2 Configuration

Package installation:

A series of libraries must be installed to run the API. These libraries are register in the PuppersApp/DogWalker_API/package.json which is supposed to content the next libraries at its dependencies and devDependencies fields:

Picture 5 package.json expected contend

All the script specified in Picture 5 is necessary to seeding , testing, and run the program. In case some package is absent in yout version of package.json, correct the mistake by adding it. The process to install all packages is:

1. Run the installation command in the console at PuppersApp/DogWalker_API

npm install

2. Check that all packages have been install using “npm list” at ../DogWalker_API . The expected output should be:

```
C:\Users\LENOVO\Downloads\ProyectoIsof\PuppersApp>cd DogWalker_API

C:\Users\LENOVO\Downloads\ProyectoIsof\PuppersApp\DogWalker_API>npm list
dogwalker-api@0.0.0 C:\Users\LENOVO\Downloads\ProyectoIsof\PuppersApp\DogWalker_API
├── compression@1.7.4
├── cookie-parser@1.4.6
├── cors@2.8.5
├── debug@2.6.9
├── dotenv@16.3.1
├── ejs@3.1.9
├── eslint-config-standard@17.1.0
├── eslint-plugin-import@2.28.0
├── eslint-plugin-n@16.0.1
├── eslint-plugin-promise@6.1.1
├── eslint@8.46.0
├── express-rate-limit@6.8.1
├── express@4.18.2
├── helmet@7.0.0
├── http-errors@1.6.3
├── jade@0.29.0
├── jest-junit@16.0.0
├── jest@29.6.2
├── joi@17.9.2
├── jsonwebtoken@9.0.1
├── morgan@1.9.1
├── mysql2@3.6.0
├── node-mocks-http@1.12.2
├── nodemon@3.0.1
├── sequelize-auto@0.8.8
├── sequelize-cli@6.6.1
├── sequelize@6.32.1
└── supertest@6.3.3
```

Picture 6 list of install packages for DogWalker_API

3. In case one of the packages has not been installed run:

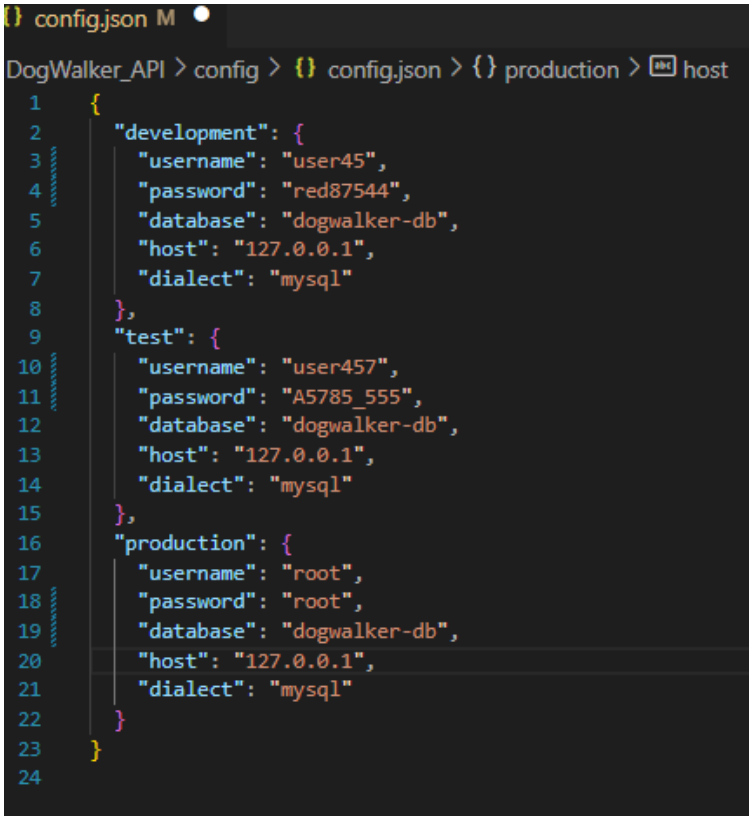
npm install <package name> <package version>

4. In case that a vulnerability is found during any of the installation proses run:

npm audit fix

Database backend connection:

Sequelise uses the PuppersApp/DogWalker_API/config/config.json to configure the connection between the database , the framework, and the main project. An example of this file is:



Picture 7 config.json example

Fild up the coreponding username, password , database and host in the config.json at the “test” and “development” fields acordin to test database created during the databse intalation at 2.3.1. For the “production” fiels put the your coresponding production database connection.

2.4 Tests

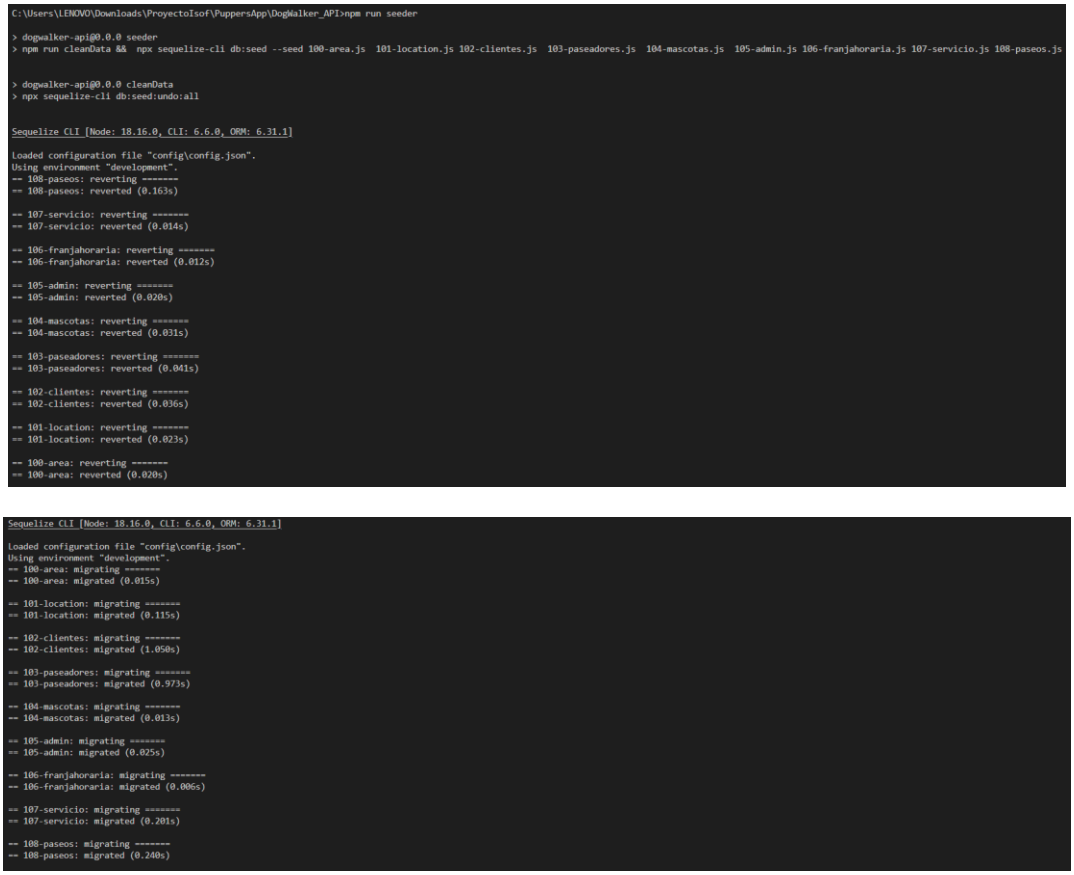
To secure that the installation process is correct and the connections between database base and API are complete. Four sets of tests must be run:

1. Change the application to test mode with the console at ../PuppersApp/DogWalker_API using :

   set NODE_ENV=test

2. Run the seeder script to populate the test database with test data:

   npm run seeder



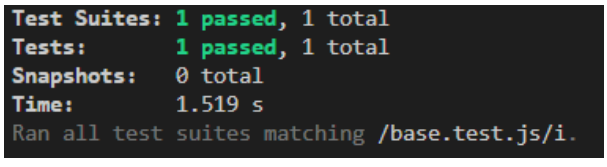
Picture 8 Expected execution of seeder

In case of a failure, the provided test database credentials could be wrong, or the database not existed. This seeder must be run before every test to ensure consistency of the database for each test run.

3. Run base test to check the server connection:

npm run test:base

The expected result should be Picture 8. Otherwise, the server is not working, and test cannot be continued.

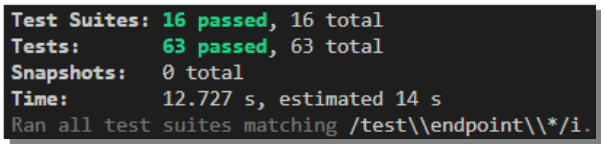


Picture 9 base.test.js example of successful test run

4. Run the services test to check se app functions at DogWalker_API/services:

npm test test:services

The expected result should be Picture 9. Otherwise, the functions are not working, and the API endpoints cannot be tested.

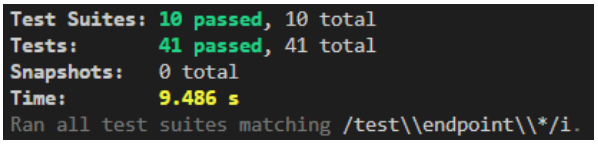


Picture 10 services example of successful test run

5. Run the endpoints test to check se API endpoints :

   npm run test:endpoint

   The expected result should be Picture 10. Otherwise, the data present in the database is not the test data or not all the endpoints are operating.



Picture 11 endpoint example of successful test run

6. Run the middleware test to check if the Api middlewares:

   To run the middleware, test the environment must set to development:

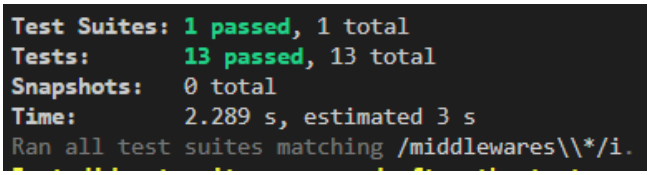
   a. Windows: *set NODE_ENV=production*

   b. Linux: *export NODE_ENV=production*

   Then run the test with

   npm run test:middlewares

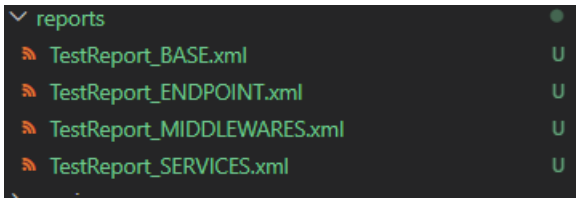
   The expected result should be Picture 11. Otherwise, the verification properties of the middleware should be failing. With out this the security of the API is compromise and should not be run in a public production environment



Picture 12 middleware example of successful test run

In case more tests need to be executed just for safety. It is necessary to rerun to the seeder at step 2 because the test at step 5 update and delete part of the data present in the database. If all the test has been past, the API can be deployed.

For every one of the 4 suites of test run a xml file with a report of each will be generated regardless of failure or success. The files are generated at PuppersApp/DogWalker_API/test/reports :



Picture 13 PuppersApp/DogWalker_API/test/reports expected files

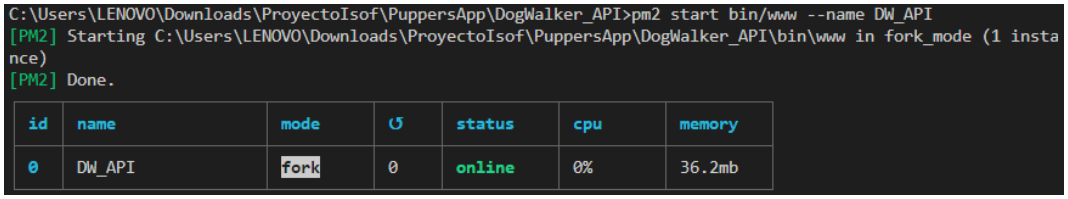
2.5 Application Server:

PM2 9.7.1 will take care of running the server in the background of the computer. For more information about this tool direct to https://pm2.keymetrics.io/docs/usage/quick-start/

1. Change the application to production mode with the console at ../PuppersApp/DogWalker_API using:
    a. Windows: *set NODE_ENV=production*
    b. Linux: *export NODE_ENV=production*
2. Change the server port to one of your preferences by changing the values of PORT at ../PuppersApp/DogWalker_API/.env . If the port is not available, the server will default to 3000.
3. Change the secret TOKEN_JSTVER at .env to a new token. This Token will be used to guaranty the security of all requests for the API, therefor chose a long and complex series of alphanumeric values.
4. Star the server using the pm2 command:

pm2 start bin/www --name <app_name>

If the server is running successfully the proses will be added to pm2 list of proses with the assign app_name:



Picture 14 pm2 prosses list

- Check that the server is running using this route: http://localhost:4001

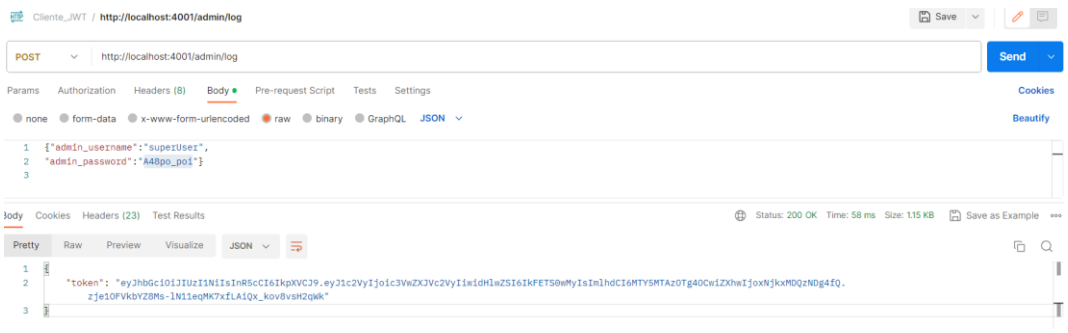
Picture 15 http://localhost:4001 example of expected response

In case of a failure, some problem could have been trigger by PM2.

To check if the connection with the production database has been stablish a top like “Postman” (https://www.postman.com/) must be use

1. Make a request with post to http://localhost:4001/admin/log and send in the body as a Json an admin_username and its corresponding password of the admin table. For example:

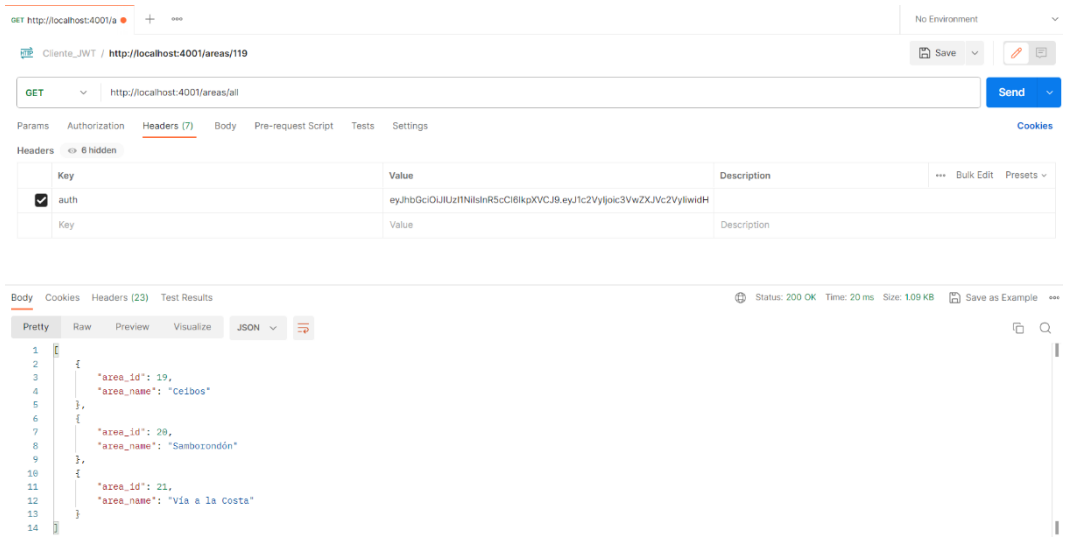
   Req.body = {"admin_username":" superUser ", "admin_password":" A48po_poi "}



Picture 16 http://localhost:4001/admin/log example of request and response

In case of an error code 400 the database connection has fail. And error codes may indicate a problem with the credentials.

2. With the token from the last request , send a get request to http://localhost:4001/areas/all with the header auth: token



Picture 17 http://localhost:4001/areas/all example of request and response

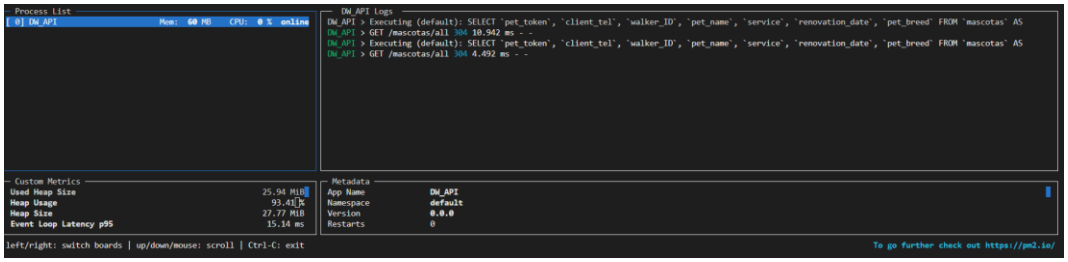
2.6 Post-installation

During execution the server could , in the worst-case scenario, fall. Therefore, pm2 offer some command to control de execution of the server and check its state.

- Restart server: pm2 restart <app_name>
- Reload server: pm2 reload <app_name>
- Stop server: pm2 stop <app_name>
- Delete/shutsown server: delete pm2 delete <app_name>

The current state of the server can be check using:

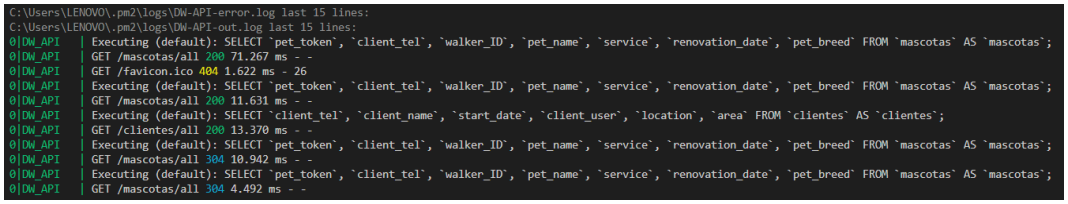
pm2 monit



Picture 18 PM2 Terminal Based Dashboard

And all the logs can be check with:

pm2 logs



Picture 19 PM2 logs example