TALLER UNIDAD 2 BACKEND

PRESENTADO POR

JHON JAIRO CORAL ERASO

PRESENTADO A

ING. VICENTE AUX

UNIVERSIDAD DE NARIÑO

INGENIERÍA DE SISTEMAS

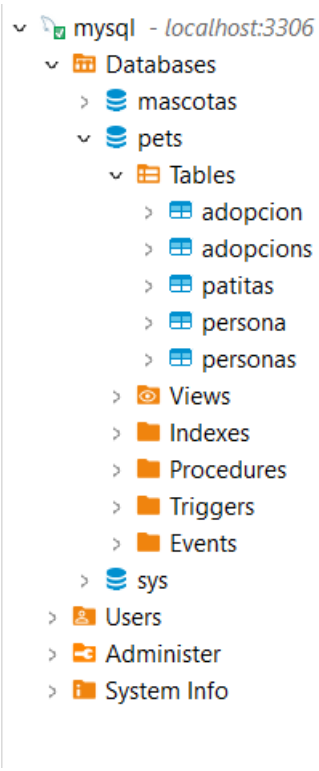
DIPLOMADO DE ACTUALIZACIÓN EN NUEVAS TECNOLOGÍAS PARA EL DESARROLLO DE SOFTWARE

2023

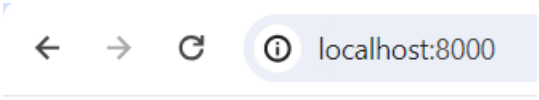
INFORME

1. Cree una base de datos MYSQL

- La base de datos se llama pets
- Tiene tres tablas
- La primera tabla se llama patitas tiene 6 columnas y tiene como llave primaria id_mascota
- La segunda tabla se llama personas tiene 6 columnas y tiene como llave primaria id_persona
- La tercera tabla se llama adopcions tiene 6 columnas y tiene como llave primaria a id_ registro y como llave foránea a id_mascota y a id_persona.
- Se crea un usuario denominado petss con una clave “1234” y así se les da los permisos a las tablas de la base de datos.

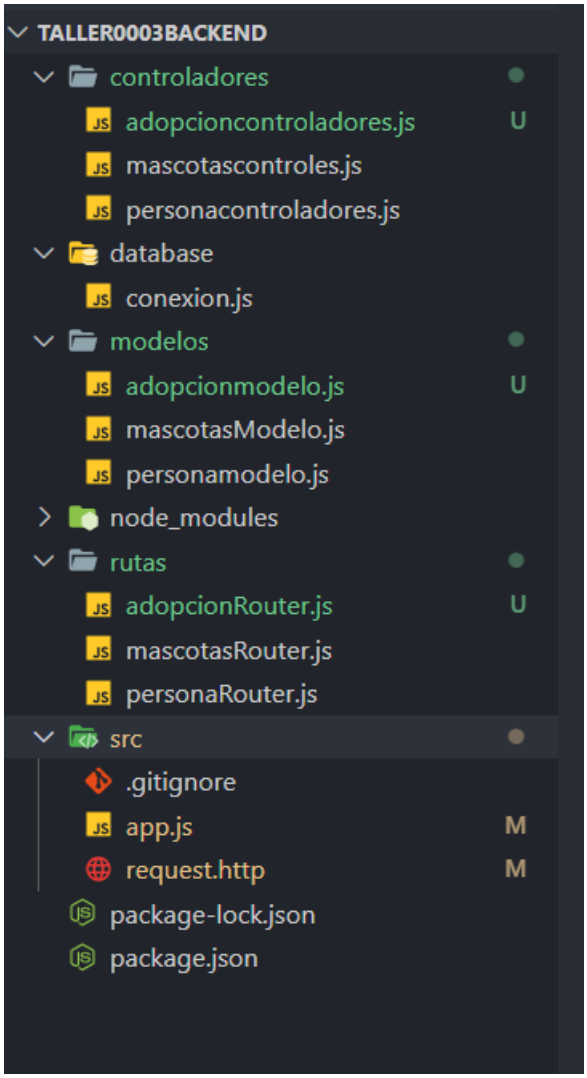


2. Desarrolle una aplicación Backend con el nombre del proyecto “TALLER0003BACKEND” implementada en NodeJS y ExpressJS que hizo uso de la base de datos denominada pets, se le hizo la respectiva conexión y permite el desarrollo de todas las tareas asociadas al registro y administración de las mascotas dadas en adopción por la empresa (Empresa Huellitas en el corazón), que se puede observar en el explorador en “loaclhost:8000”.
   Cabe destacara que para la elaboración se creó un nuevo controlador, modelo y ruta para cada tabla para una mejor optimización, también en la app.js se hace la importación para cada tabla y se le asigna la respectiva ruta, luego se abre la terminal y mediante el comando npm run start se observa que la base de datos está conectada exitosamente y el servidor esta inicializado en puerto 8000

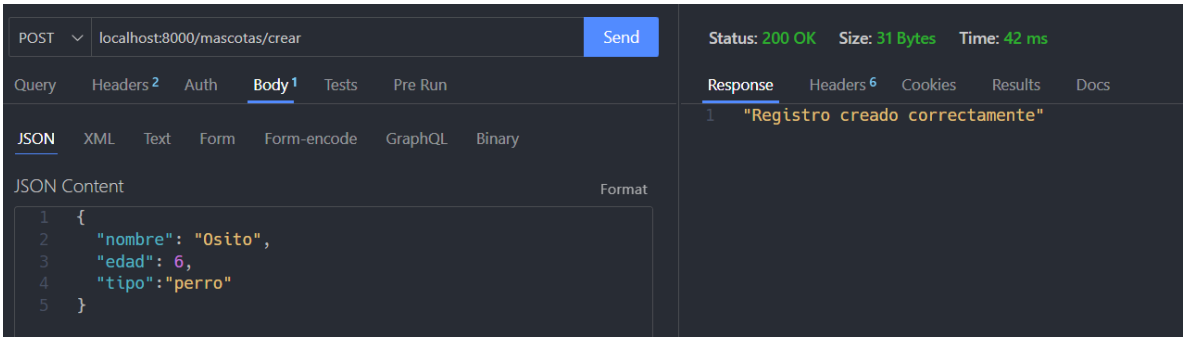


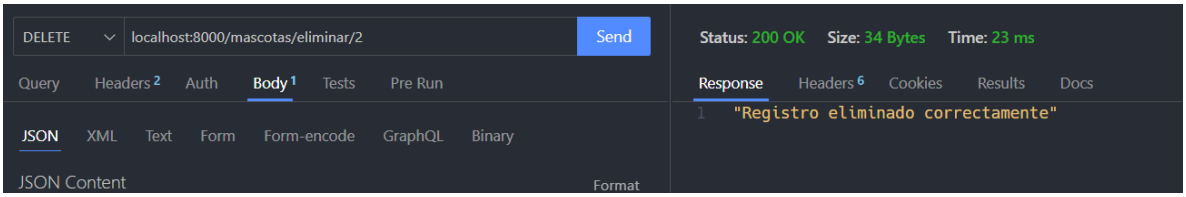
Empresa Huellitas en el corazon

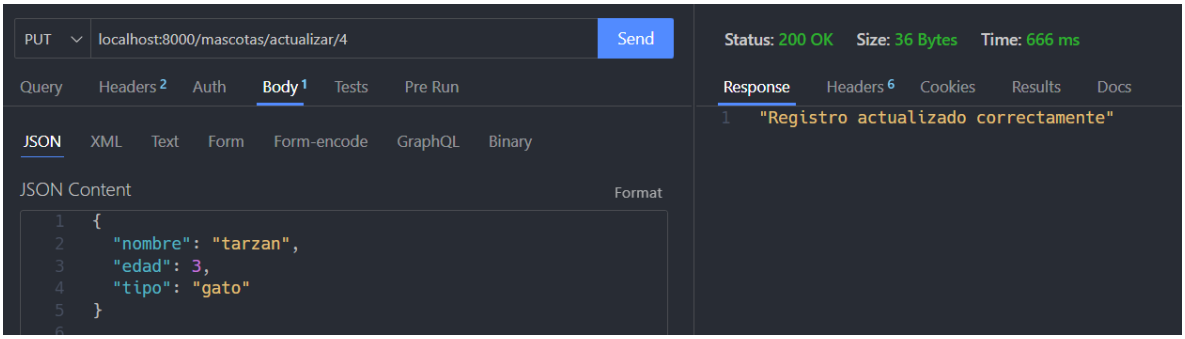
```
PS D:\DIPLOMADO\UNIDAD2\Taller0003Backend> npm run start

> taller0002backend@1.0.0 start
> nodemon ./src/app.js

[nodemon] 3.0.2
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,cjs,json
[nodemon] starting `node ./src/app.js`
Executing (default): SELECT 1+1 AS result
Executing (default): SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_TYPE = 'BASE TABLE' AND TABLE_NAME = 'patitas' AND TABLE_SCHEMA
= 'pets'
Base de datos conectada de manera exitosa
Executing (default): SHOW INDEX FROM `patitas`
Executing (default): SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_TYPE = 'BASE TABLE' AND TABLE_NAME = 'personas' AND TABLE_SCHEMA
 = 'pets'
Executing (default): SHOW INDEX FROM `personas`
Executing (default): SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_TYPE = 'BASE TABLE' AND TABLE_NAME = 'adopcions' AND TABLE_SCHEM
A = 'pets'
Executing (default): SHOW INDEX FROM `adopcions`
Servidor inicializado en puerto 8000
```

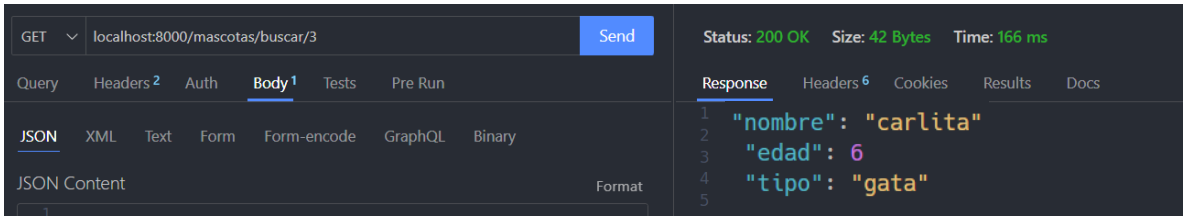


3.


POST
localhost:8000/mascotas/crear
Send
Query
Headers 2
Auth
Body 1
Tests
Pre Run
JSON
XML
Text
Form
Form-encode
GraphQL
Binary
JSON Content
Format
{
"nombre": "Osito",
"edad": 6,
"tipo":"perro"
}
Status: 200 OK
Size: 31 Bytes
Time: 42 ms
Response
Headers 6
Cookies
Results
Docs
"Registro creado correctamente"


DELETE
localhost:8000/mascotas/eliminar/2
Send
Query
Headers 2
Auth
Body 1
Tests
Pre Run
JSON
XML
Text
Form
Form-encode
GraphQL
Binary
JSON Content
Format
Status: 200 OK
Size: 34 Bytes
Time: 23 ms
Response
Headers 6
Cookies
Results
Docs
"Registro eliminado correctamente"


PUT
localhost:8000/mascotas/actualizar/4
Send
Query
Headers 2
Auth
Body 1
Tests
Pre Run
JSON
XML
Text
Form
Form-encode
GraphQL
Binary
JSON Content
Format
{
"nombre": "tarzan",
"edad": 3,
"tipo": "gato"
}
Status: 200 OK
Size: 36 Bytes
Time: 666 ms
Response
Headers 6
Cookies
Results
Docs
"Registro actualizado correctamente"


GET
localhost:8000/mascotas/buscar/3
Send
Query
Headers 2
Auth
Body 1
Tests
Pre Run
JSON
XML
Text
Form
Form-encode
GraphQL
Binary
JSON Content
Format
Status: 200 OK
Size: 42 Bytes
Time: 166 ms
Response
Headers 6
Cookies
Results
Docs
"nombre": "carlita"
"edad": 6
"tipo": "gata"