

Universidad de Nariño.

Ingeniería de Sistemas.

Diplomado de actualización en nuevas tecnologías para el desarrollo de Software.

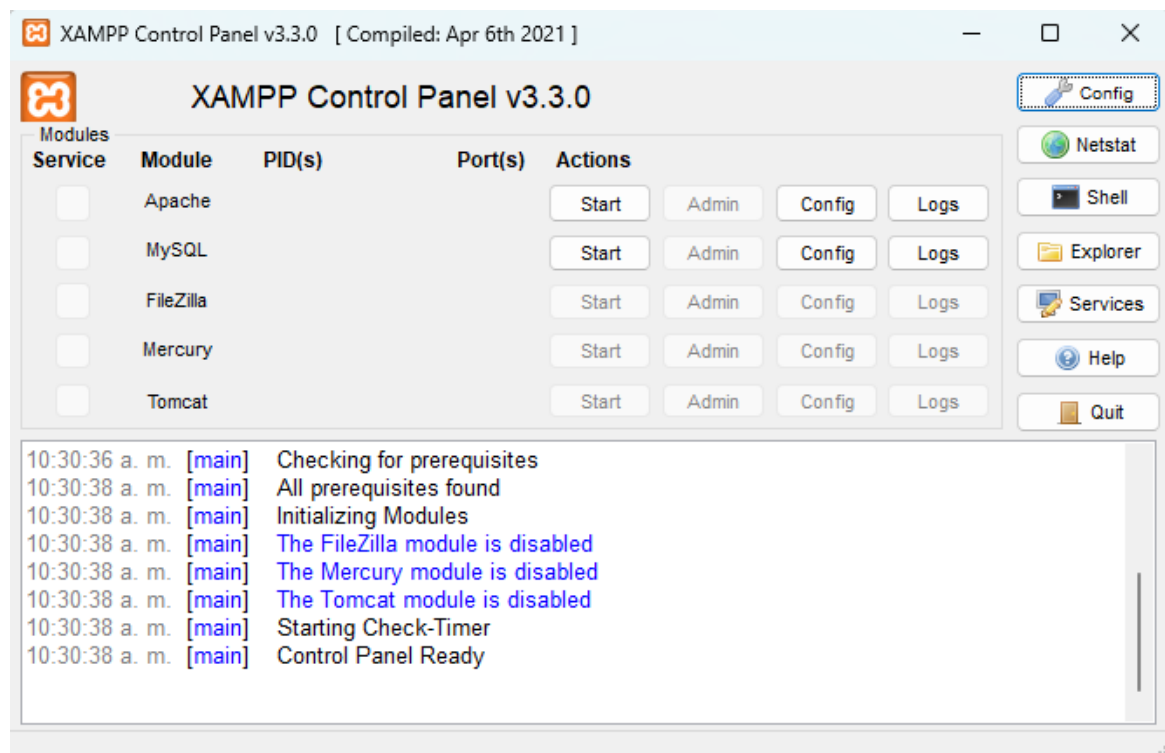
Estudiante: David Alejandro Rodríguez Acosta

Taller Unidad 2 Backend.

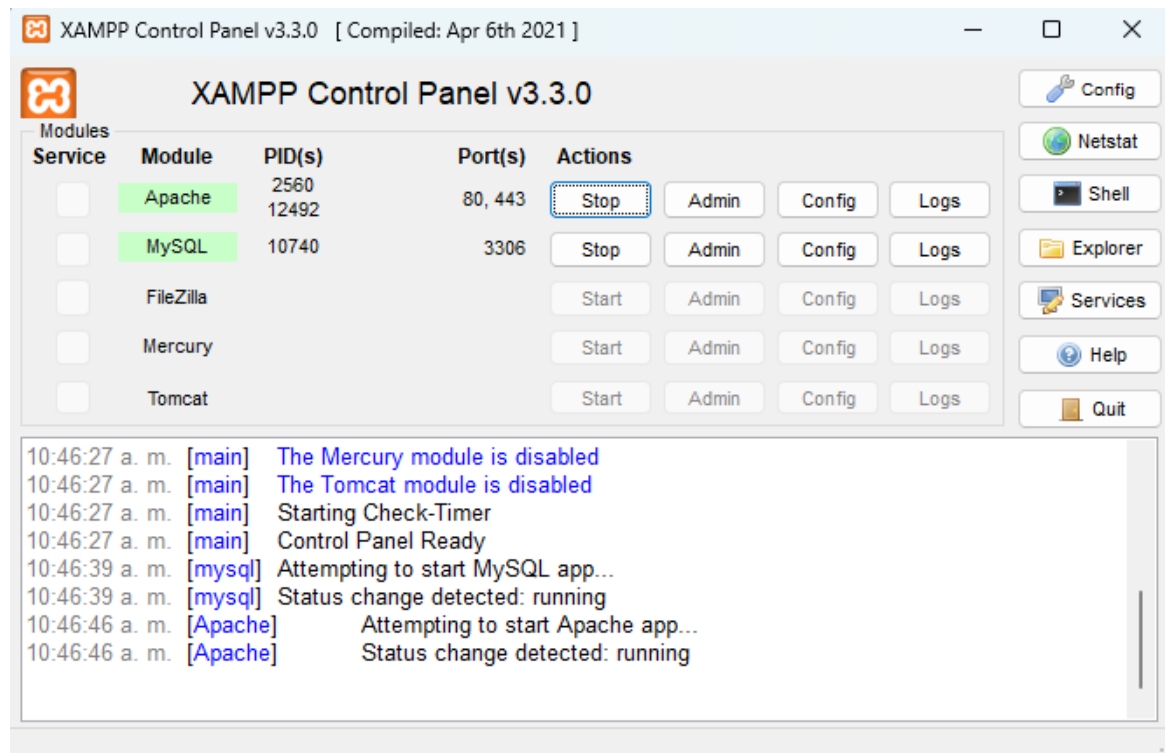
1. Crear una base de datos MYSQL que permita llevar el registro de mascotas (perros y gatos), así como también el proceso de solicitud de adopción de estas.

Para crear una base de datos MySQL utilizare el programa XAMPP el cual me permite crear y administrar la base de datos de manera gráfica.

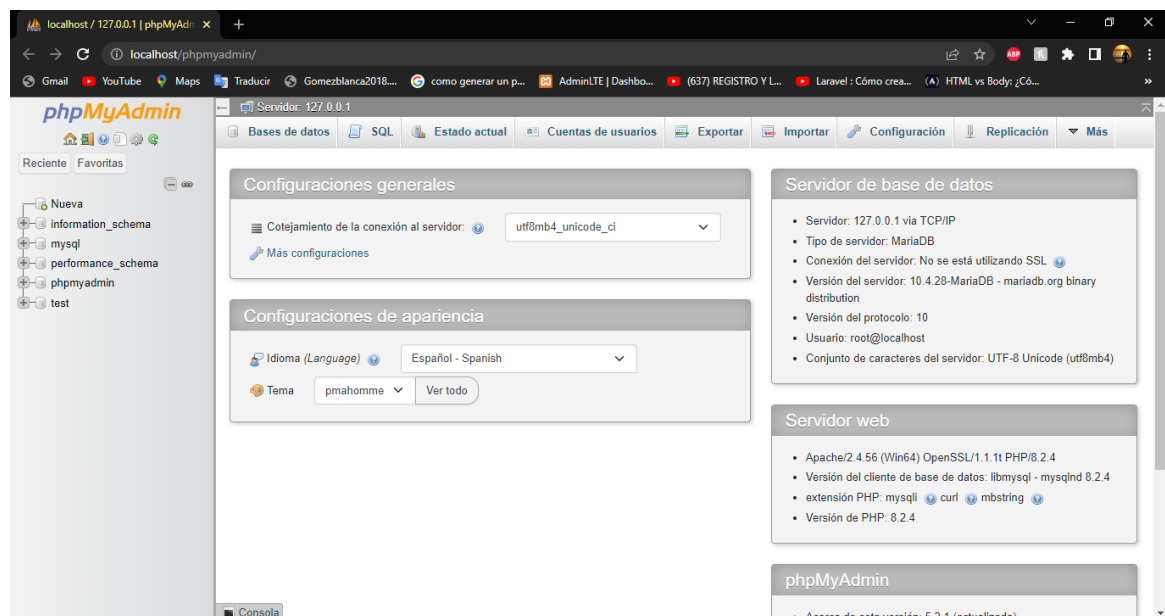
Primero se ejecuta el programa.



Luego se inician los servicios.

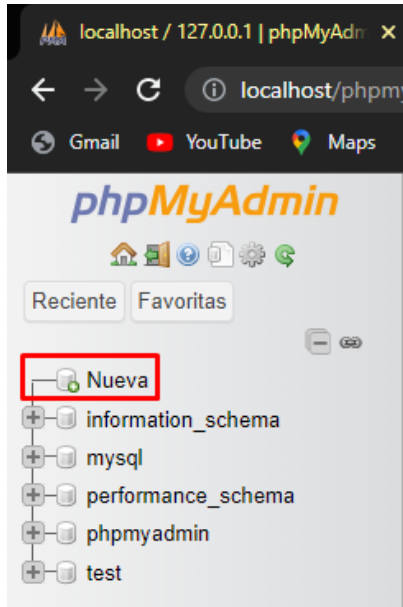


Al darle clic en “**Admin**” en la parte de MySQL automáticamente se abrirá el navegador en donde nos mostrará el panel de administración de MySQL.



Ya estando dentro del panel de administración de MySQL podemos crear la base de datos para llevar el registro de mascotas y el proceso de solicitud de adopción.

Damos click en “Nueva”



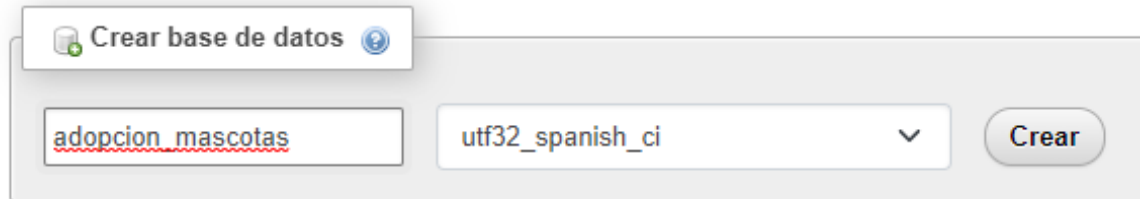
Se abrirá un nuevo panel en donde podremos crear nuestra base de datos,

Bases de datos

A screenshot of the 'Crear base de datos' (Create database) form in phpMyAdmin. The form has a title bar that says 'Crear base de datos' with a help icon. Below the title bar, there is a label 'Nombre de la base de datos' (Database name) next to a text input field containing the text 'utf8mb4_general_ci'. To the right of the input field is a 'Crear' (Create) button.

Crearemos la base de datos “**adopcion_mascotas**” y al lado del nombre de la base de datos seleccionamos “**utf32_spanish_ci**” y por defecto el motor ya es “**InnoDB**”, debería quedar así.

Bases de datos

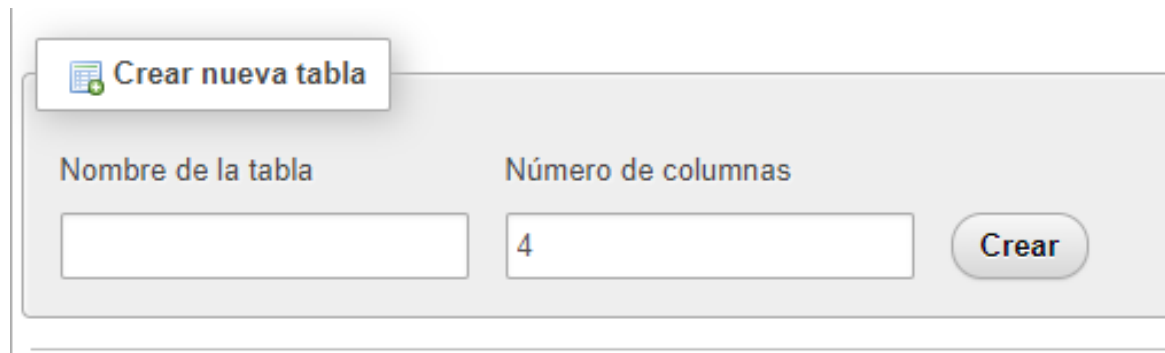


Crear base de datos

adopcion_mascotas utf32_spanish_ci

Crear

Después de dar clic en “**Crear**” nos aparecerá el siguiente panel.



Crear nueva tabla

Nombre de la tabla Número de columnas

4

Crear

Aquí vamos a crear las tablas para nuestra base de datos por lo tanto se usará dos tablas: La primera tabla se llamará “**mascotas**” la cual se utilizará para el registro de mascotas y tendrá 5 atributos, posteriormente se creará la tabla “**solicitudes_adopcion**” la cual se usará para gestionar las solicitudes de adopción, esta tendrá 4 atributos.

Tabla mascotas:

Nombre de la tabla: **mascotas** Agregar 1 columna(s) Continuar

Nombre	Tipo	Longitud/Valores	Predeterminado	Cotejamiento	Atributos	Nulo	Índice	A.I
id <small>Seleccionar desde las columnas centrales</small>	INT		Ninguno			<input type="checkbox"/>	PRIMARY <small>PRIMARY</small>	<input checked="" type="checkbox"/>
nombre <small>Seleccionar desde las columnas centrales</small>	VARCHAR	100	Ninguno			<input type="checkbox"/>	---	<input type="checkbox"/>
especie <small>Seleccionar desde las columnas centrales</small>	VARCHAR	50	Ninguno			<input type="checkbox"/>	---	<input type="checkbox"/>
edad <small>Seleccionar desde las columnas centrales</small>	INT		Ninguno			<input type="checkbox"/>	---	<input type="checkbox"/>
disponible <small>Seleccionar desde las columnas centrales</small>	BOOLEAN		Personalizado: TRUE			<input type="checkbox"/>	---	<input type="checkbox"/>

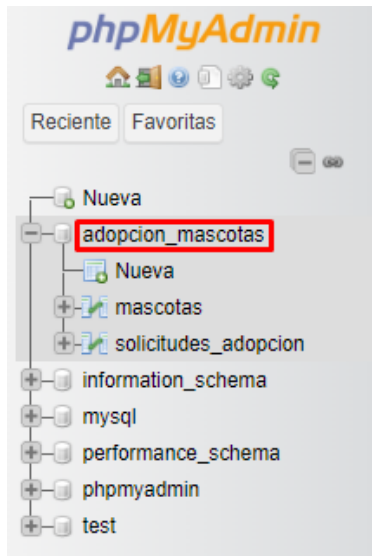
Tabla solicitudes_adopcion:

Nombre de la tabla: **solicitudes_adopcion** Agregar 1 columna(s) Continuar

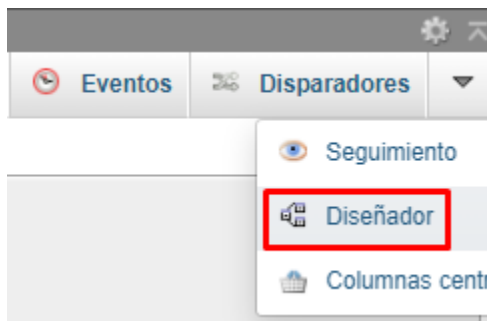
Nombre	Tipo	Longitud/Valores	Predeterminado	Cotejamiento	Atributos	Nulo	Índice	A.I
id <small>Seleccionar desde las columnas centrales</small>	INT		Ninguno			<input type="checkbox"/>	PRIMARY <small>PRIMARY</small>	<input checked="" type="checkbox"/>
nombre_solicitante <small>Seleccionar desde las columnas centrales</small>	VARCHAR	100	Ninguno			<input type="checkbox"/>	---	<input type="checkbox"/>
id_mascota <small>Seleccionar desde las columnas centrales</small>	INT		Ninguno			<input checked="" type="checkbox"/>	---	<input type="checkbox"/>
estado <small>Seleccionar desde las columnas centrales</small>	VARCHAR	50	Personalizado: Pendiente			<input type="checkbox"/>	---	<input type="checkbox"/>

Luego de haber creado las tablas debemos relacionarlas puesto que id_mascotas es una llave foránea, para eso hacemos lo siguiente:

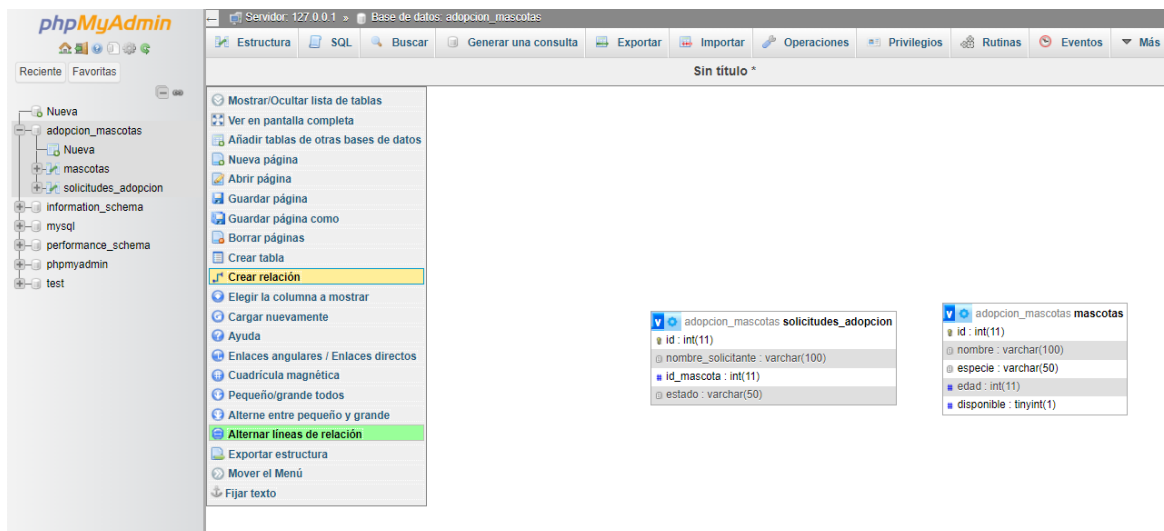
Damos clic en la base de datos “**adopción_mascotas**”



Nos dirigimos a la esquina superior derecha y seleccionamos la opción “**Diseñador**”



Una vez dentro, en el menú lateral izquierdo seleccionamos la opción “**Crear relación**”



Si por alguna razón la relación no deja ser creada desde el modo diseñador entonces podemos ejecutar una consulta SQL para crear la llave foránea así:

Seleccionamos la base de datos “**adopción_mascotas**”, en el menú de arriba seleccionamos “**SQL**” y ejecutamos la siguiente consulta (para ejecutar la consulta damos clic en el botón “**Continuar**” dentro de la opción SQL):

```
ALTER TABLE solicitudes_adopcion
ADD CONSTRAINT fk_solicitudes_adopcion_mascotas
FOREIGN KEY (id_mascota)
REFERENCES mascotas(id)
ON DELETE RESTRICT
ON UPDATE RESTRICT;
```

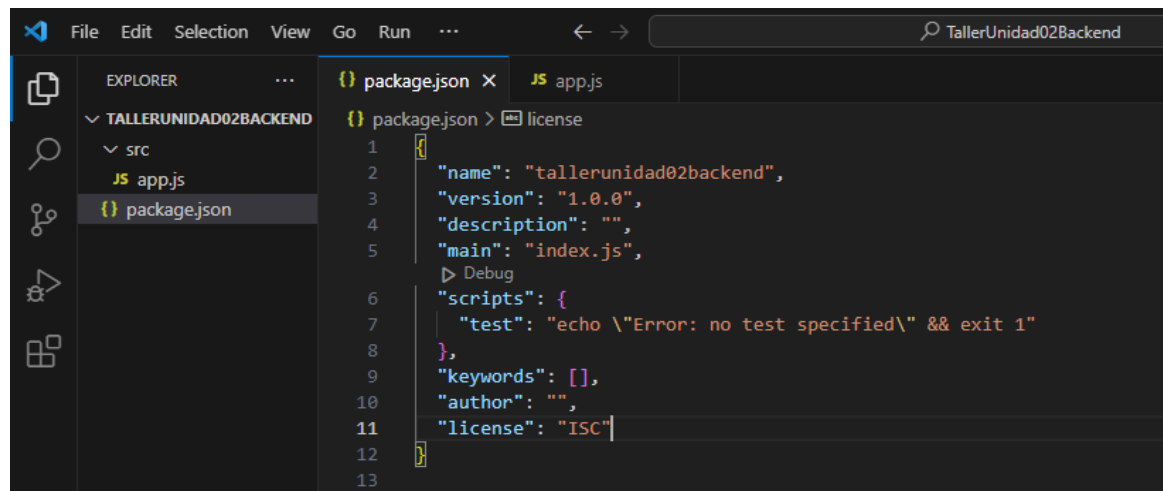
Y con esto tendríamos creada nuestra base de datos que permita llevar el registro de mascotas, así como también el proceso de solicitud de adopción de estas.

2. Desarrollar una aplicación Backend implementada en NodeJS y ExpressJS que haga uso de la base de datos del primer punto y que permita el desarrollo de todas las tareas asociadas al registro y administración de las mascotas dadas en adopción por la empresa (La empresa debe contar con un nombre).

Se debe hacer uso correcto de los verbos HTTP dependiendo de la tarea a realizar.

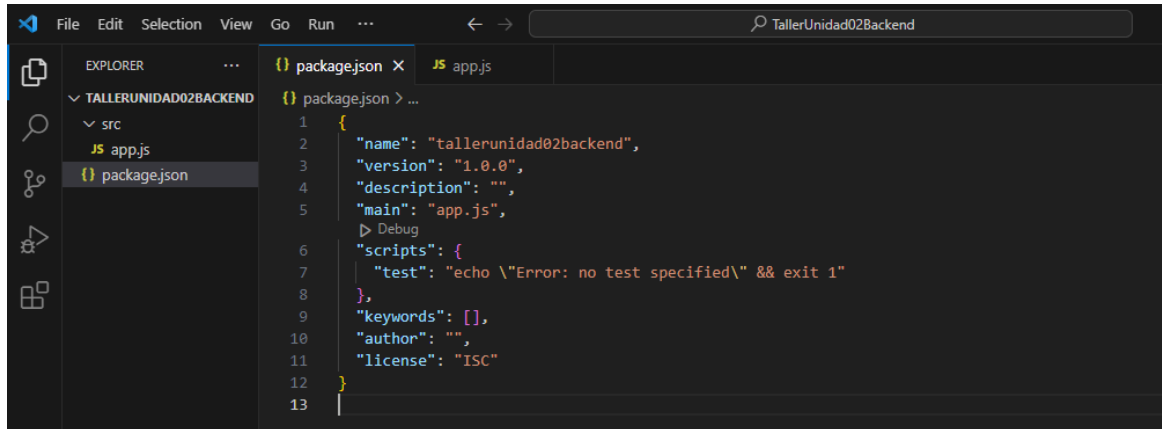
Para inicializar el proyecto con NodeJS escribimos en la consola **npm init -Y** para que posteriormente dentro de mi carpeta se cree un archivo llamado **package.json**

```
PS C:\Users\DAVID\Downloads\Diplomado\Talleres Backend\TallerUnidad02Backend> npm init -Y
```



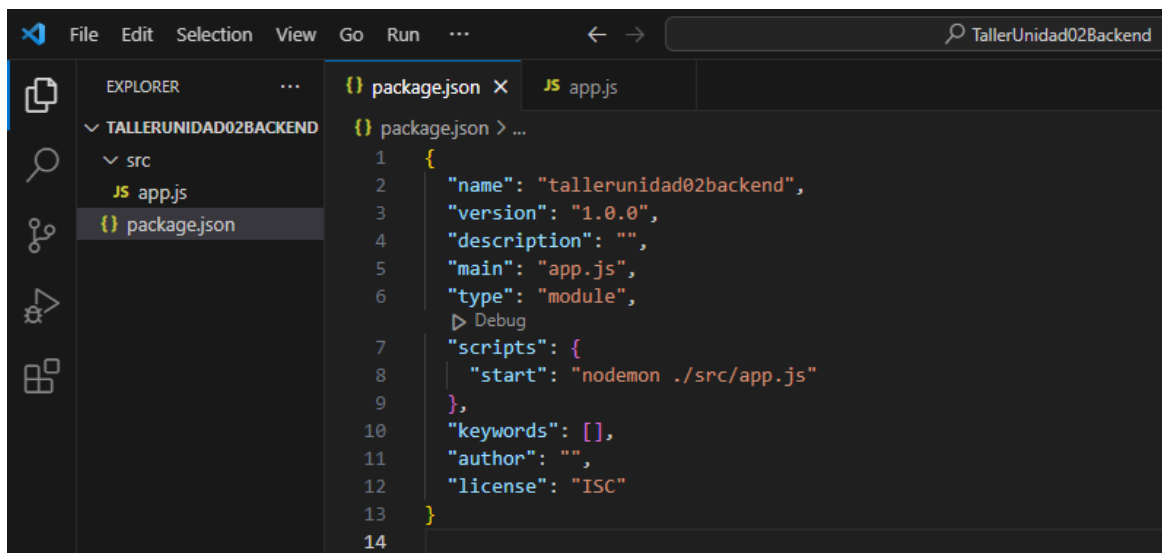
```
{
  "name": "tallerunidad02backend",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC"
}
```

Y dentro del archivo que se nos ha creado editamos la línea **"main": "index.js"**, que por defecto dice que **index.js** va a ser el archivo principal del proyecto, pues nosotros lo cambiaremos ahora a **app.js** y adicionalmente a esto creamos la carpeta **src** y dentro de ella el archivo **app.js**.



```
1 {
2   "name": "tallerunidad02backend",
3   "version": "1.0.0",
4   "description": "",
5   "main": "app.js",
6   "scripts": {
7     "test": "echo \"Error: no test specified\" && exit 1"
8   },
9   "keywords": [],
10  "author": "",
11  "license": "ISC"
12 }
```

Ahora la línea **"test": "echo \"Error: no test specified\" && exit 1"** la vamos a reemplazar por **"start": "nodemon ./src/app.js"** después agregamos la siguiente línea debajo de main **"type": "module"**,

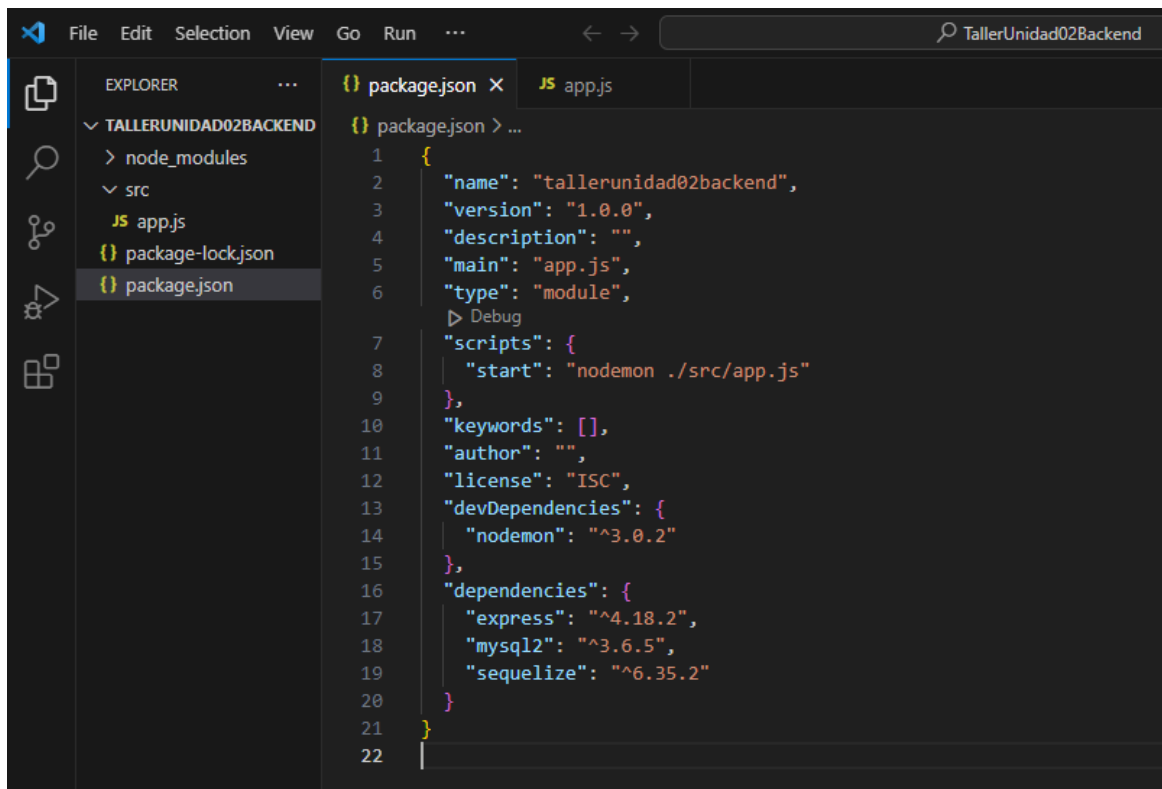


```
1 {
2   "name": "tallerunidad02backend",
3   "version": "1.0.0",
4   "description": "",
5   "main": "app.js",
6   "type": "module",
7   "scripts": {
8     "start": "nodemon ./src/app.js"
9   },
10  "keywords": [],
11  "author": "",
12  "license": "ISC"
13 }
```

Posteriormente instalamos las dependencias y dependencias de desarrollo:

```
● PS C:\Users\DAVID\Downloads\Diplomado\Talleres Backend\TallerUnidad02Backend> npm install nodemon -D
● PS C:\Users\DAVID\Downloads\Diplomado\Talleres Backend\TallerUnidad02Backend> npm install express
● PS C:\Users\DAVID\Downloads\Diplomado\Talleres Backend\TallerUnidad02Backend> npm install mysql2
● PS C:\Users\DAVID\Downloads\Diplomado\Talleres Backend\TallerUnidad02Backend> npm install sequelize
```

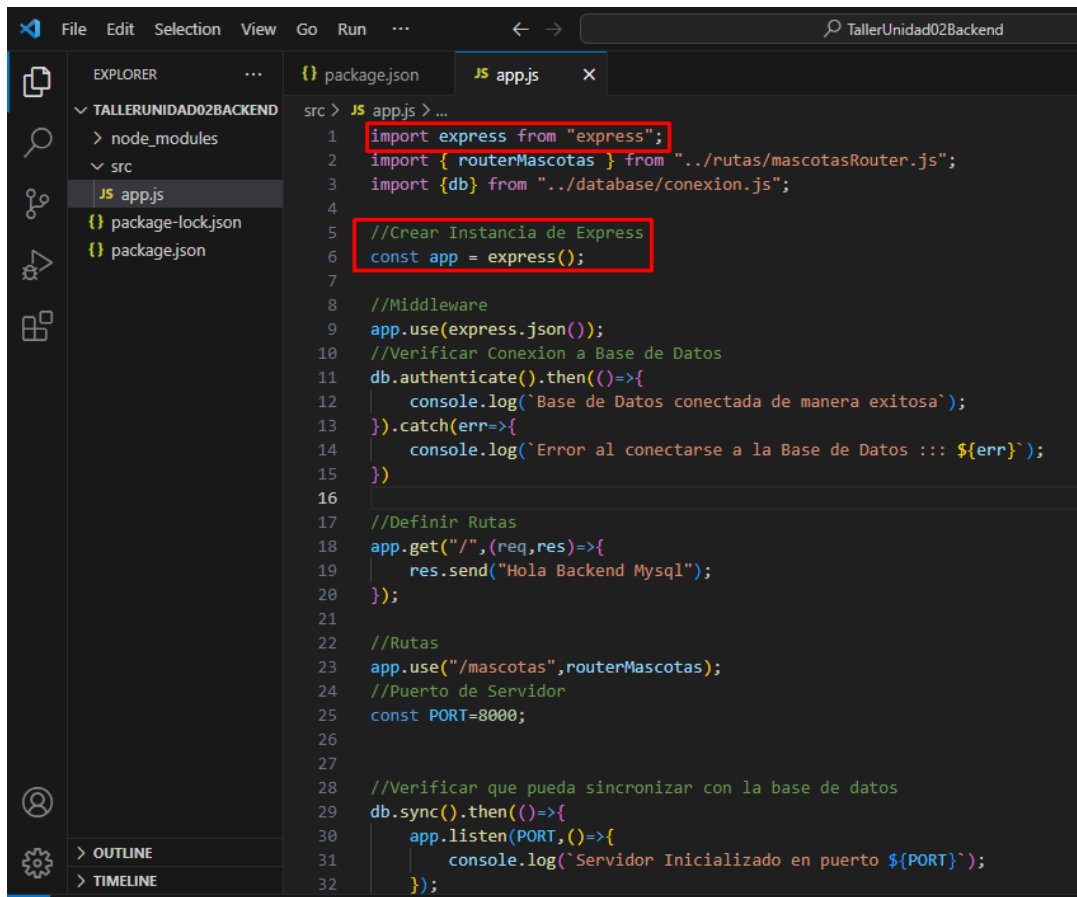

Una vez instalado todo esto el archivo **package.json** debería quedar de la siguiente manera



The screenshot shows the Visual Studio Code interface with the Explorer sidebar on the left and the Editor window on the right. The Explorer sidebar shows the project structure for 'TALLERUNIDAD02BACKEND', including 'node_modules', 'src', 'app.js', 'package-lock.json', and 'package.json'. The Editor window displays the content of 'package.json'.

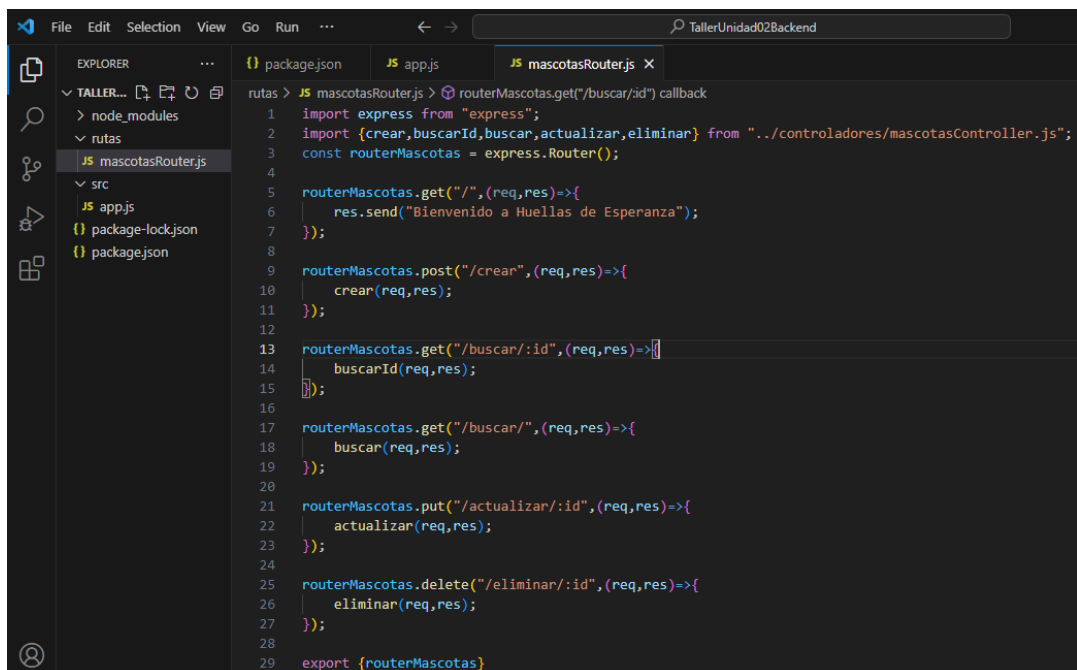
```
1  {
2    "name": "tallerunidad02backend",
3    "version": "1.0.0",
4    "description": "",
5    "main": "app.js",
6    "type": "module",
7    "scripts": {
8      "start": "nodemon ./src/app.js"
9    },
10   "keywords": [],
11   "author": "",
12   "license": "ISC",
13   "devDependencies": {
14     "nodemon": "^3.0.2"
15   },
16   "dependencies": {
17     "express": "^4.18.2",
18     "mysql2": "^3.6.5",
19     "sequelize": "^6.35.2"
20   }
21 }
```

Ahora en el archivo **app.js** empezamos a utilizar ExpressJS importando express e instanciándolo para poder trabajar



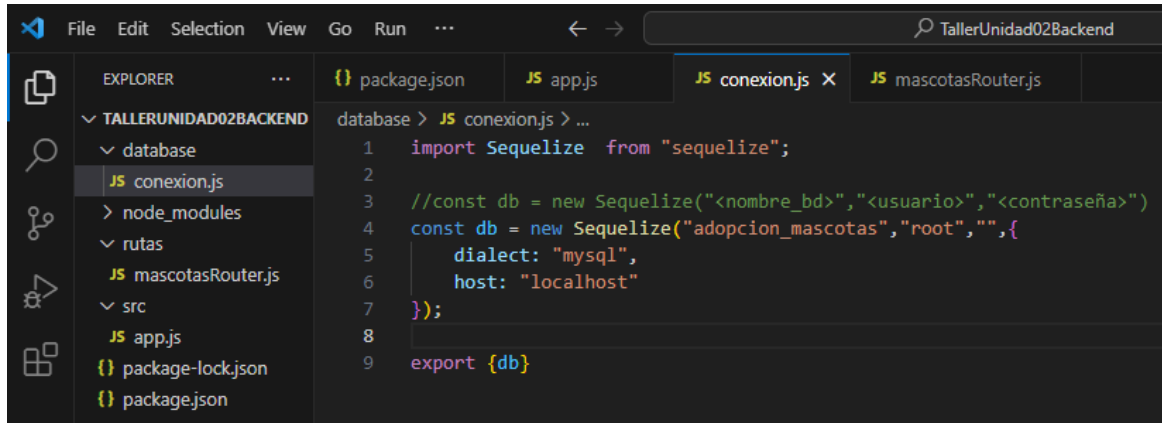
```
src > JS app.js > ...
1  import express from "express";
2  import { routerMascotas } from "../rutas/mascotasRouter.js";
3  import { db } from "../database/conexion.js";
4
5  //Crear Instancia de Express
6  const app = express();
7
8  //Middleware
9  app.use(express.json());
10 //Verificar Conexion a Base de Datos
11 db.authenticate().then(()=>{
12   console.log(`Base de Datos conectada de manera exitosa`);
13 }).catch(err=>{
14   console.log(`Error al conectarse a la Base de Datos ::: ${err}`);
15 })
16
17 //Definir Rutas
18 app.get("/",(req,res)=>{
19   res.send("Hola Backend Mysql");
20 });
21
22 //Rutas
23 app.use("/mascotas",routerMascotas);
24 //Puerto de Servidor
25 const PORT=8000;
26
27
28 //Verificar que pueda sincronizar con la base de datos
29 db.sync().then(()=>{
30   app.listen(PORT,()=>{
31     console.log(`Servidor Inicializado en puerto ${PORT}`);
32   });
33 });
```

Ahora creamos una carpeta en donde estarán las rutas de nuestro proyecto y dentro de esa carpeta crearemos el archivo **mascotasRouter.js**



```
rutas > JS mascotasRouter.js > routerMascotas.get("/buscar/:id") callback
1  import express from "express";
2  import { crear, buscarId, buscar, actualizar, eliminar } from "../controladores/mascotasController.js";
3  const routerMascotas = express.Router();
4
5  routerMascotas.get("/",(req,res)=>{
6   res.send("Bienvenido a Huellas de Esperanza");
7 });
8
9  routerMascotas.post("/crear",(req,res)=>{
10   crear(req,res);
11 });
12
13 routerMascotas.get("/buscar/:id",(req,res)=>{
14   buscarId(req,res);
15 });
16
17 routerMascotas.get("/buscar/",(req,res)=>{
18   buscar(req,res);
19 });
20
21 routerMascotas.put("/actualizar/:id",(req,res)=>{
22   actualizar(req,res);
23 });
24
25 routerMascotas.delete("/eliminar/:id",(req,res)=>{
26   eliminar(req,res);
27 });
28
29 export {routerMascotas}
```

Ahora vamos a vincular la base de datos creada en el punto anterior creando una nueva carpeta llamada **database** y dentro de ella el archivo **conexion.js**, dentro de este archivo vamos a utilizar **Sequelize** ya que este nos permite gestionar la base de datos

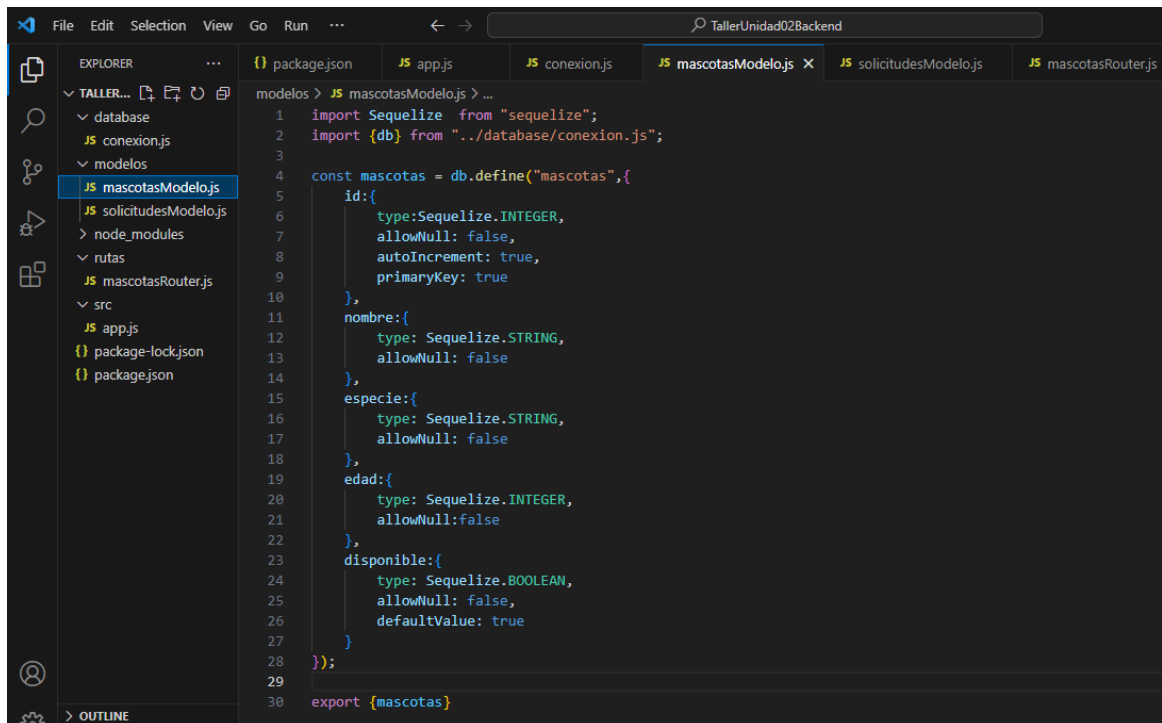


The screenshot shows the VS Code interface with the Explorer panel on the left displaying the project structure. The 'database' folder is expanded, showing 'conexion.js'. The main editor displays the code for 'conexion.js'.

```
1 import Sequelize from "sequelize";
2
3 //const db = new Sequelize("<nombre_bd>", "<usuario>", "<contraseña>")
4 const db = new Sequelize("adopcion_mascotas", "root", "", {
5   dialect: "mysql",
6   host: "localhost"
7 });
8
9 export {db}
```

Se puede crear un nuevo usuario para poder gestionar la base de datos, pero para hacerlo de manera más rápida utilizamos el usuario **root** y la contraseña que en este caso no habría

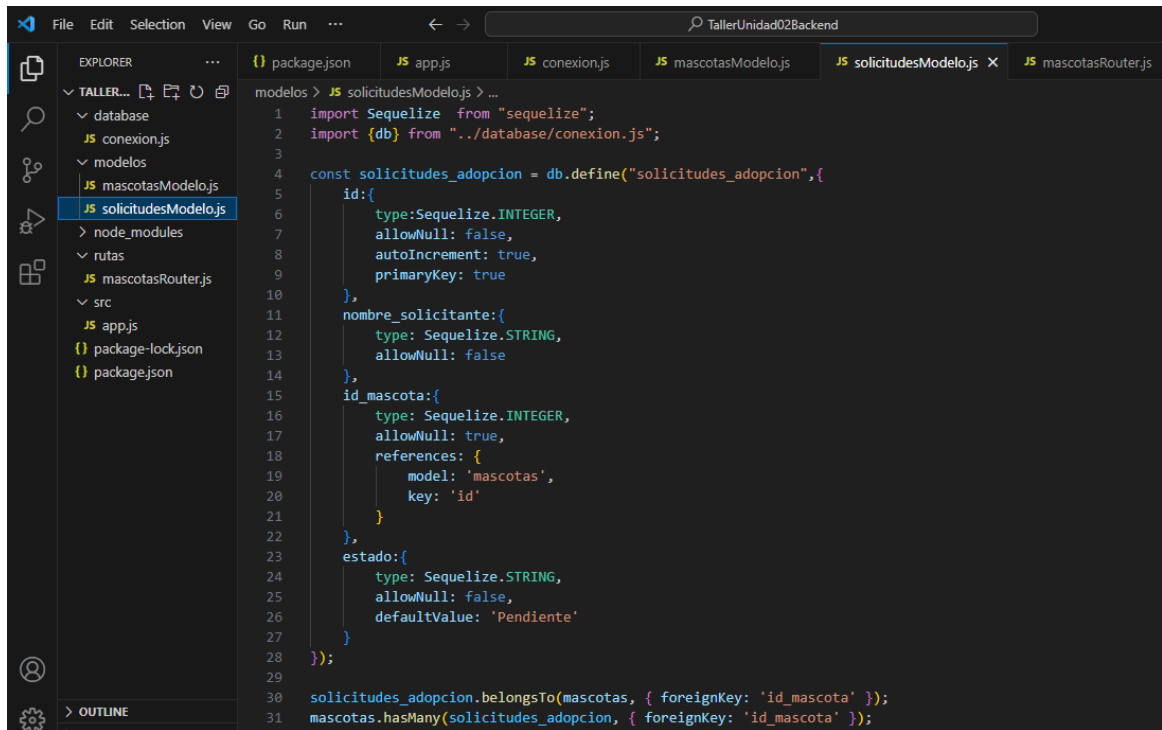
Posteriormente definimos los modelos de datos, para ello creamos una carpeta a la cual llamaremos **modelos** y dentro de ella creamos un archivo llamado **mascotasModelo.js** para la tabla mascotas



The screenshot shows the VS Code interface with the Explorer panel on the left displaying the project structure. The 'modelos' folder is expanded, showing 'mascotasModelo.js'. The main editor displays the code for 'mascotasModelo.js'.

```
1 import Sequelize from "sequelize";
2 import {db} from "../database/conexion.js";
3
4 const mascotas = db.define("mascotas", {
5   id: {
6     type: Sequelize.INTEGER,
7     allowNull: false,
8     autoIncrement: true,
9     primaryKey: true
10  },
11   nombre: {
12     type: Sequelize.STRING,
13     allowNull: false
14   },
15   especie: {
16     type: Sequelize.STRING,
17     allowNull: false
18   },
19   edad: {
20     type: Sequelize.INTEGER,
21     allowNull: false
22   },
23   disponible: {
24     type: Sequelize.BOOLEAN,
25     allowNull: false,
26     defaultValue: true
27   }
28 });
29
30 export {mascotas}
```

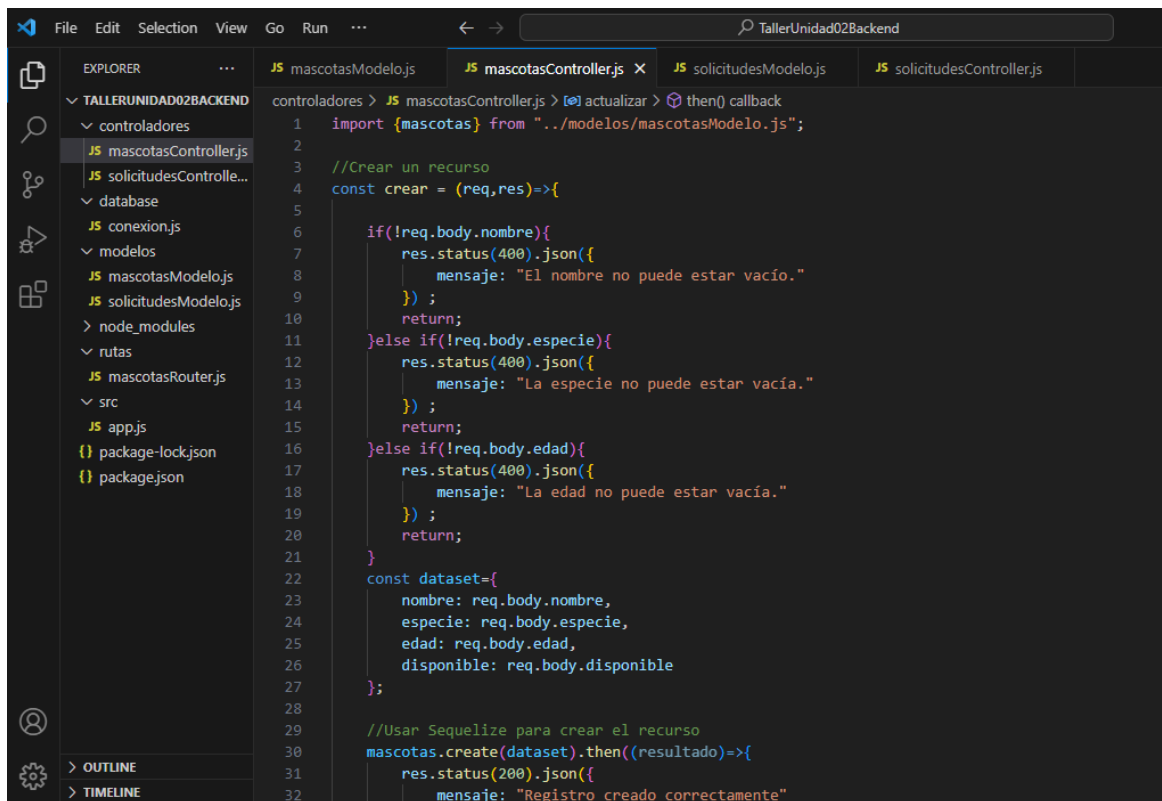
y otro archivo llamado **solicitudesModelo.js** para la tabla solicitudes_adopcion



The screenshot shows the Visual Studio Code interface. On the left, the Explorer sidebar displays a project structure with folders like 'database', 'modelos', 'rutas', and 'src'. The file 'JS solicitudesModelo.js' is selected. The main editor area shows the code for this file, which defines a Sequelize model for the 'solicitudes_adopcion' table. The code includes imports for 'Sequelize' and 'db', and defines the model with attributes like 'id', 'nombre_solicitante', 'id_mascota', and 'estado'. It also establishes relationships with the 'mascotas' model using 'belongsTo' and 'hasMany'.

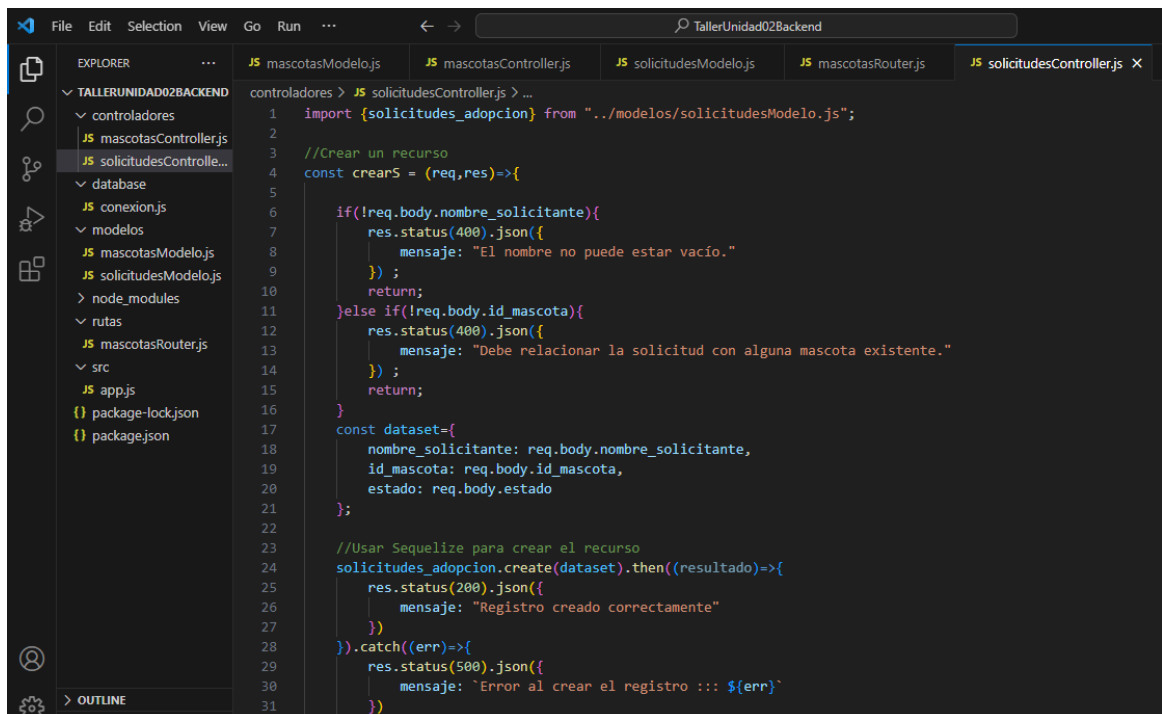
```
1 import Sequelize from "sequelize";
2 import {db} from "../database/conexion.js";
3
4 const solicitudes_adopcion = db.define("solicitudes_adopcion",{
5   id:{
6     type:Sequelize.INTEGER,
7     allowNull: false,
8     autoIncrement: true,
9     primaryKey: true
10  },
11  nombre_solicitante:{
12    type: Sequelize.STRING,
13    allowNull: false
14  },
15  id_mascota:{
16    type: Sequelize.INTEGER,
17    allowNull: true,
18    references: {
19      model: 'mascotas',
20      key: 'id'
21    }
22  },
23  estado:{
24    type: Sequelize.STRING,
25    allowNull: false,
26    defaultValue: 'Pendiente'
27  }
28 });
29
30 solicitudes_adopcion.belongsTo(mascotas, { foreignKey: 'id_mascota' });
31 mascotas.hasMany(solicitudes_adopcion, { foreignKey: 'id_mascota' });
```

Ahora crearemos todos los métodos, para ello creamos una carpeta la cual la nombraremos como **controladores** y esta contendrá un archivo llamado **mascotasController.js** para la tabla mascotas



```
1 import {mascotas} from "../modelos/mascotasModelo.js";
2
3 //Crear un recurso
4 const crear = (req,res)=>{
5
6     if(!req.body.nombre){
7         res.status(400).json({
8             mensaje: "El nombre no puede estar vacío."
9         });
10        return;
11    }else if(!req.body.especie){
12        res.status(400).json({
13            mensaje: "La especie no puede estar vacía."
14        });
15        return;
16    }else if(!req.body.edad){
17        res.status(400).json({
18            mensaje: "La edad no puede estar vacía."
19        });
20        return;
21    }
22    const dataset={
23        nombre: req.body.nombre,
24        especie: req.body.especie,
25        edad: req.body.edad,
26        disponible: req.body.disponible
27    };
28
29    //Usar Sequelize para crear el recurso
30    mascotas.create(dataset).then((resultado)=>{
31        res.status(200).json({
32            mensaje: "Registro creado correctamente"
```

Posteriormente creamos otro archivo en la misma carpeta llamado **solicitudesController.js**



```
1 import {solicitudes_adopcion} from "../modelos/solicitudesModelo.js";
2
3 //Crear un recurso
4 const crearS = (req,res)=>{
5
6     if(!req.body.nombre_solicitante){
7         res.status(400).json({
8             mensaje: "El nombre no puede estar vacío."
9         });
10        return;
11    }else if(!req.body.id_mascota){
12        res.status(400).json({
13            mensaje: "Debe relacionar la solicitud con alguna mascota existente."
14        });
15        return;
16    }
17    const dataset={
18        nombre_solicitante: req.body.nombre_solicitante,
19        id_mascota: req.body.id_mascota,
20        estado: req.body.estado
21    };
22
23    //Usar Sequelize para crear el recurso
24    solicitudes_adopcion.create(dataset).then((resultado)=>{
25        res.status(200).json({
26            mensaje: "Registro creado correctamente"
27        });
28    }).catch((err)=>{
29        res.status(500).json({
30            mensaje: "Error al crear el registro ::: ${err}"
31        });
32    });
33 }
```

3. Realizar verificación de las diferentes operaciones a través de un cliente grafico (Postman, Imnsomia, etc.), tomar capturas de pantalla que evidencien el resultado de las solicitudes realizadas.

PETICIONES PARA LA TABLA MASCOTAS CON REST CLIENT

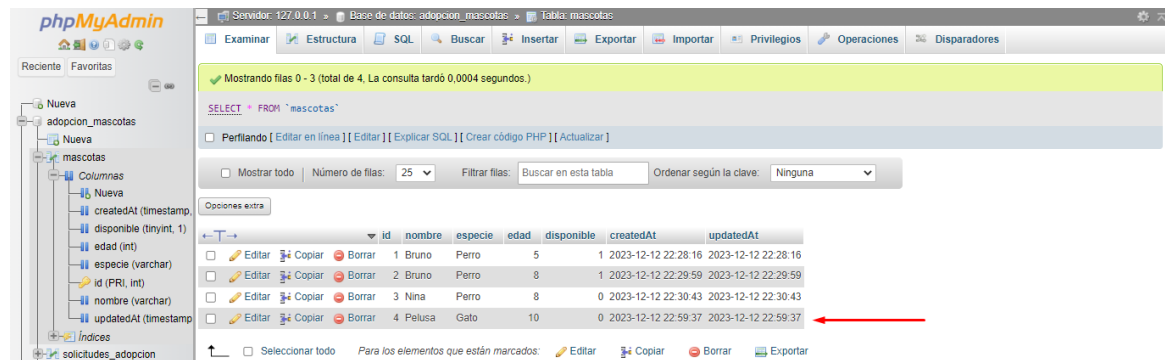
Petición realizada

```
requests.http > POST /mascotas/crear
1  ###
  Send Request
2  POST http://localhost:8000/mascotas/crear HTTP/1.1
3  Content-Type: application/json
4
5  {
6    "nombre": "Pelusa",
7    "especie": "Gato",
8    "edad": 10,
9    "disponible": ""
10 }
```

Respuesta recibida

```
Response(399ms) X
1  HTTP/1.1 200 OK
2  X-Powered-By: Express
3  Content-Type: application/json; charset=utf-8
4  Content-Length: 43
5  ETag: W/"2b-q67xZLRgZ8v2LE0Rt4QP7gs1ZYI"
6  Date: Wed, 13 Dec 2023 03:59:37 GMT
7  Connection: close
8
9  {
10   "mensaje": "Registro creado correctamente"
11 }
```

Verificación en la base de datos



The screenshot shows the phpMyAdmin interface for a database named 'adopcion_mascotas'. The 'mascotas' table is selected, and its structure is displayed on the left. The table has the following columns: id (int), nombre (varchar), especie (varchar), edad (int), disponible (tinyint), createdAt (timestamp), and updatedAt (timestamp). The table contains 4 rows of data. A red arrow points to the last row, which has id 4, nombre 'Pelusa', especie 'Gato', edad 10, and disponible 0.

id	nombre	especie	edad	disponible	createdAt	updatedAt
1	Bruno	Perro	5	1	2023-12-12 22:28:16	2023-12-12 22:28:16
2	Bruno	Perro	8	1	2023-12-12 22:29:59	2023-12-12 22:29:59
3	Nina	Perro	8	0	2023-12-12 22:30:43	2023-12-12 22:30:43
4	Pelusa	Gato	10	0	2023-12-12 22:59:37	2023-12-12 22:59:37

Petición realizada

```
###
Send Request
POST http://localhost:8000/mascotas/crear HTTP/1.1
Content-Type: application/json

{
  "nombre": "Ramona",
  "especie": "Gato",
  "edad": 13,
  "disponible": "true"
}
###
```

Respuesta recibida

```
Response(1092ms) X
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 43
5 ETag: W/"2b-q67xZLRgZ8v2LE0Rt4QP7gs1ZYI"
6 Date: Wed, 13 Dec 2023 04:08:45 GMT
7 Connection: close
8
9 {
10   "mensaje": "Registro creado correctamente"
11 }
```

Verificación en la base de datos

phpMyAdmin

Reciente

Favoritas

Nueva

adopcion_mascotas

Nueva

mascotas

solicitudes_adopcion

information_schema

mysql

performance_schema

phpmyadmin

test

Examinar

Estructura

SQL

Buscar

Insertar

Exportar

Importar

Privilegios

Operaciones

Disparadores

Mostrando filas 0 - 4 (total de 5, La consulta tardó 0.0003 segundos.)

SELECT * FROM `mascotas`

Perfilando

Editar en línea

Editar

Explicar SQL

Crear código PHP

Actualizar

Mostrar todo

Número de filas: 25

Filtrar filas: Buscar en esta tabla

Ordenar según la clave: Ninguna

Opciones extra

id

nombre

especie

edad

disponible

createdAt

updatedAt

☐

Editar

Copiar

Borrar

1

Bruno

Perro

5

1

2023-12-12 22:28:16

2023-12-12 22:28:16

☐

Editar

Copiar

Borrar

2

Bruno

Perro

8

1

2023-12-12 22:29:59

2023-12-12 22:29:59

☐

Editar

Copiar

Borrar

3

Nina

Perro

8

0

2023-12-12 22:30:43

2023-12-12 22:30:43

☐

Editar

Copiar

Borrar

4

Pelusa

Gato

10

0

2023-12-12 22:59:37

2023-12-12 22:59:37

☐

Editar

Copiar

Borrar

5

Ramona

Gato

13

1

2023-12-12 23:08:44

2023-12-12 23:08:44

Petición realizada

```
###
Send Request
✓ GET http://localhost:8000/mascotas/buscar/3 HTTP/1.1
###
```

Respuesta recibida

```
Response(11ms) X
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 148
5 ETag: W/"94-ps3Rn42Po3Fo+K5fbQQrFirSWLg"
6 Date: Wed, 13 Dec 2023 04:12:33 GMT
7 Connection: close
8
9 {
10   "id": 3,
11   "nombre": "Nina",
12   "especie": "Perro",
13   "edad": 8,
14   "disponible": false,
15   "createdAt": "2023-12-13T03:30:43.000Z",
16   "updatedAt": "2023-12-13T03:30:43.000Z"
17 }
```


Comprobación en la base de datos

phpMyAdmin

Reciente

Favoritas

Nueva

adopcion_mascotas

Nueva

mascotas

solicitudes_adopcions

information_schema

mysql

performance_schema

phpmyadmin

test

Servidor: 127.0.0.1

Base de datos: adopcion_mascotas

Tabla: mascotas

Examinar

Estructura

SQL

Buscar

Insertar

Exportar

Importar

Privilegios

Operaciones

Disparadores

Mostrando filas 0 - 4 (total de 5. La consulta tardó 0,0003 segundos.)

SELECT * FROM 'mascotas'

Perfilando

[Editar en línea]

[Editar]

[Explicar SQL]

[Crear código PHP]

[Actualizar]

☐ Mostrar todo

Número de filas: 25

Filtrar filas:

Ordenar según la clave: Ninguna

Opciones extra

</

Petición realizada

```
###
Send Request
PUT http://localhost:8000/mascotas/actualizar/2 HTTP/1.1
Content-Type: application/json

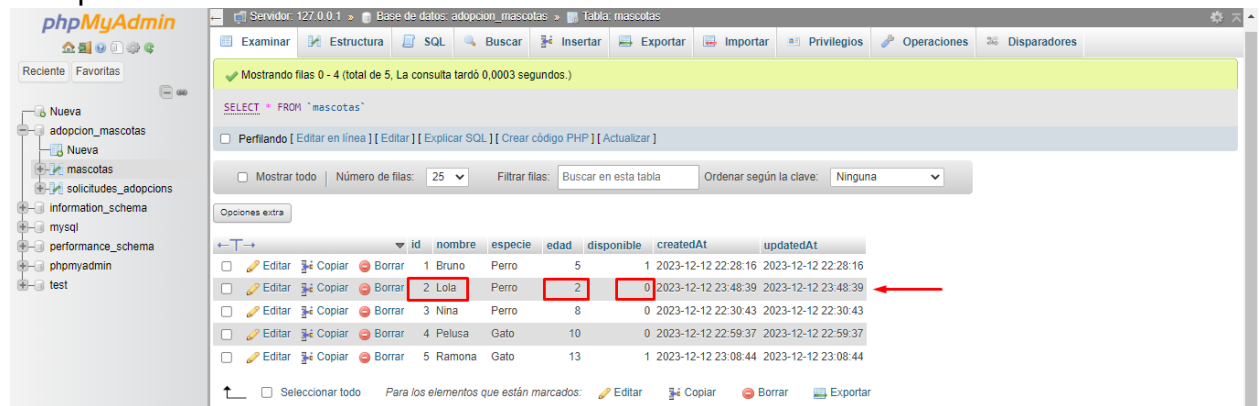
{
  "nombre": "Lola",
  "especie": "Perro",
  "edad": 2,
  "disponible": ""
}
###
```

Respuesta recibida

```
Response(44ms) X
```

```
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 34
5 ETag: W/"22-E2y8aRbSHGARTuWgI/yV7FkDoBc"
6 Date: Wed, 13 Dec 2023 04:48:39 GMT
7 Connection: close
8
9 {
10   "mensaje": "Registro Actualizado"
11 }
```

Comprobación en la base de datos



Mostrando filas 0 - 4 (total de 5, La consulta tardó 0,0003 segundos)

SELECT * FROM `mascotas`

Perfilando [Editar en línea] [Editar] [Explicar SQL] [Crear código PHP] [Actualizar]

☐ Mostrar todo Número de filas: 25 Filtrar filas: Buscar en esta tabla Ordenar según la clave: Ninguna

Opciones extra

				id	nombre	especie	edad	disponible	createdAt	updatedAt
<input type="checkbox"/>	Editar	Copiar	Borrar	1	Bruno	Perro	5	1	2023-12-12 22:28:16	2023-12-12 22:28:16
<input type="checkbox"/>	Editar	Copiar	Borrar	2	Lola	Perro	2	0	2023-12-12 23:48:39	2023-12-12 23:48:39
<input type="checkbox"/>	Editar	Copiar	Borrar	3	Nina	Perro	8	0	2023-12-12 22:30:43	2023-12-12 22:30:43
<input type="checkbox"/>	Editar	Copiar	Borrar	4	Pelusa	Gato	10	0	2023-12-12 22:59:37	2023-12-12 22:59:37
<input type="checkbox"/>	Editar	Copiar	Borrar	5	Ramona	Gato	13	1	2023-12-12 23:08:44	2023-12-12 23:08:44

☐ Seleccionar todo Para los elementos que están marcados: Editar Copiar Borrar Exportar

Petición realizada

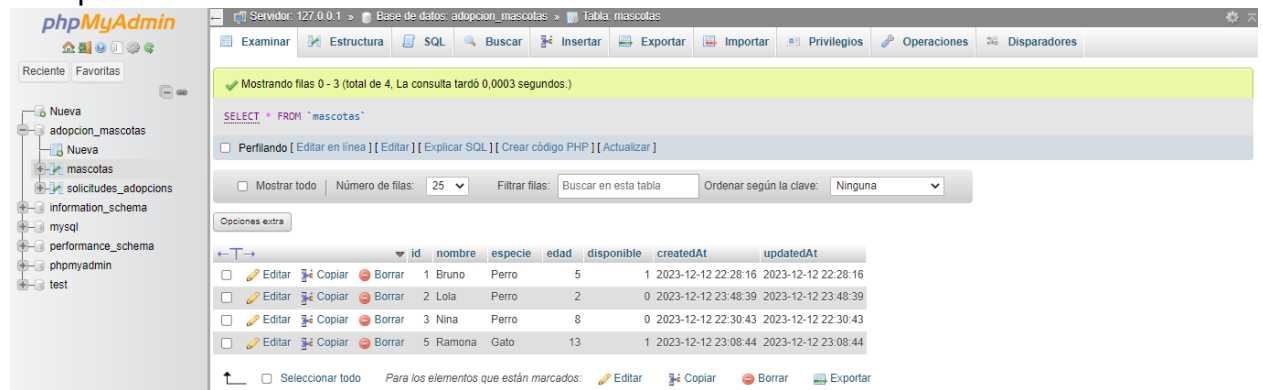
```
###
Send Request
DELETE http://localhost:8000/mascotas/eliminar/4 HTTP/1.1
```

Respuesta recibida

```
Response(32ms) X

1  HTTP/1.1 200 OK
2  X-Powered-By: Express
3  Content-Type: application/json; charset=utf-8
4  Content-Length: 32
5  ETag: W/"20-hkH0JCV/EfydTXVDeEcSo30bS8A"
6  Date: Wed, 13 Dec 2023 04:53:27 GMT
7  Connection: close
8
9  {
10   "mensaje": "Registro Eliminado"
11 }
```

Comprobación en la base de datos



Mostrando filas 0 - 3 (total de 4, La consulta tardó 0,0003 segundos.)

`SELECT * FROM `mascotas``

Perfilando [[Editar en línea](#)] [[Editar](#)] [[Explicar SQL](#)] [[Crear código PHP](#)] [[Actualizar](#)]

☐ Mostrar todo | Número de filas: 25 | Filtrar filas: | Ordenar según la clave: Ninguna

Opciones extra

				id	nombre	especie	edad	disponible	createdAt	updatedAt
<input type="checkbox"/>	Editar	Copiar	Borrar	1	Bruno	Perro	5	1	2023-12-12 22:28:16	2023-12-12 22:28:16
<input type="checkbox"/>	Editar	Copiar	Borrar	2	Lola	Perro	2	0	2023-12-12 23:48:39	2023-12-12 23:48:39
<input type="checkbox"/>	Editar	Copiar	Borrar	3	Nina	Perro	8	0	2023-12-12 22:30:43	2023-12-12 22:30:43
<input type="checkbox"/>	Editar	Copiar	Borrar	5	Ramona	Gato	13	1	2023-12-12 23:08:44	2023-12-12 23:08:44

☐ Seleccionar todo | Para los elementos que están marcados: [Editar](#) [Copiar](#) [Borrar](#) [Exportar](#)

Se pudo comprobar que las peticiones hacia la tabla mascotas realizadas en REST Client la cual es una extensión de Visual Studio Code si fueron exitosas todas.

PETICIONES PARA LA TABLA SOLICITUDES_ADOPCION CON REST CLIENT

Petición realizada

```
###
Send Request
POST http://localhost:8000/solicitudes/crear HTTP/1.1
Content-Type: application/json

{
  "nombre_solicitante": "Germán",
  "id_mascota": 5,
  "estado": "Pendiente"
}
###
```

Respuesta recibida

```
Response(60ms) X
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 43
5 ETag: W/"2b-q67xZLRgZ8v2LE0Rt4QP7gs1ZYI"
6 Date: Wed, 13 Dec 2023 05:05:38 GMT
7 Connection: close
8
9 {
10   "mensaje": "Registro creado correctamente"
11 }
```

Comprobación en la base de datos



Al tener el puntero del mouse sobre el numero de id_mascota podemos observar a quien pertenece ese numero en la tabla mascotas

Petición realizada

```
###
Send Request
POST http://localhost:8000/solicitudes/crear HTTP/1.1
Content-Type: application/json

{
  "nombre_solicitante": "Sandra",
  "id_mascota": 1,
  "estado": "Pendiente"
}
###
```

Respuesta recibida

```
Response(33ms) X
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 43
5 ETag: W/"2b-q67xZLRgZ8v2LE0Rt4QP7gs1ZYI"
6 Date: Wed, 13 Dec 2023 05:10:23 GMT
7 Connection: close
8
9 {
10   "mensaje": "Registro creado correctamente"
11 }
```

Comprobación en la base de datos



The screenshot shows the phpMyAdmin interface. On the left, the database structure is visible, including 'adopcion_mascotas' and 'solicitudes_adopcions'. The main panel displays the 'solicitudes_adopcions' table with the following data:

id	nombre_solicitante	id_mascota	estado	createdAt	updatedAt
1	Germán	5	Pendiente	2023-12-13 05:05:37	2023-12-13 05:05:37
2	Sandra	1	Pendiente	2023-12-13 05:10:23	2023-12-13 05:10:23

Petición realizada

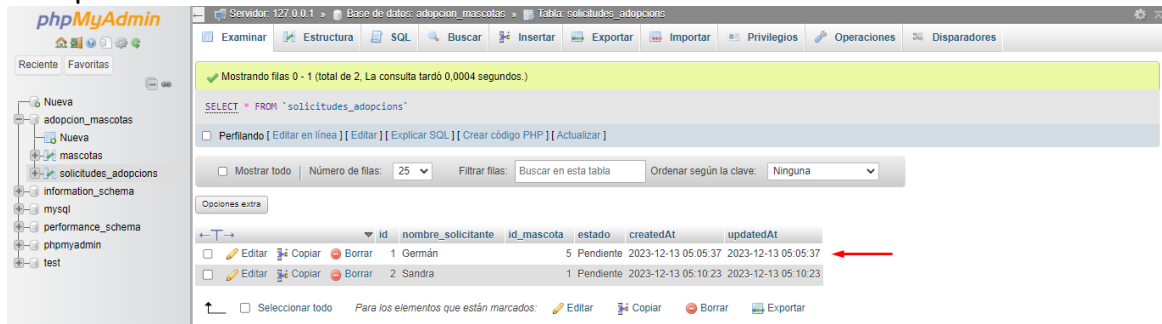
```
###
Send Request
GET http://localhost:8000/solicitudes/buscar/1 HTTP/1.1
###
```

Respuesta recibida

```
Response(19ms) X

1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 153
5 ETag: W/"99-fQ5R6q6UNPOo1VLtPKjHCyOYS5s"
6 Date: Wed, 13 Dec 2023 05:13:02 GMT
7 Connection: close
8
9 {
10   "id": 1,
11   "nombre_solicitante": "Germán",
12   "id_mascota": 5,
13   "estado": "Pendiente",
14   "createdAt": "2023-12-13T05:05:37.000Z",
15   "updatedAt": "2023-12-13T05:05:37.000Z"
16 }
```

Comprobación en la base de datos



Mostrando filas 0 - 1 (total de 2, La consulta tardó 0,0004 segundos.)

SELECT * FROM `solicitudes_adopciones`

Perfilando [Editar en línea] [Editar] [Explicar SQL] [Crear código PHP] [Actualizar]

☐ Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla | Ordenar según la clave: Ninguna

Opciones extra

	id	nombre_solicitante	id_mascota	estado	createdAt	updatedAt
<input type="checkbox"/>	1	Germán	5	Pendiente	2023-12-13 05:05:37	2023-12-13 05:05:37
<input type="checkbox"/>	2	Sandra	1	Pendiente	2023-12-13 05:10:23	2023-12-13 05:10:23

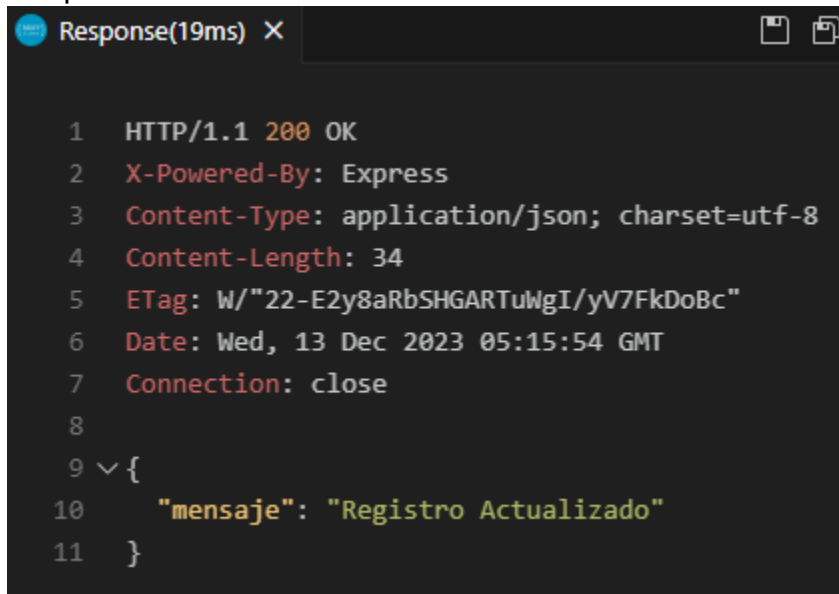
☐ Seleccionar todo | Para los elementos que están marcados: [Editar] [Copiar] [Borrar] [Exportar]

Petición realizada

```
###
Send Request
PUT http://localhost:8000/solicitudes/actualizar/2 HTTP/1.1
Content-Type: application/json

{
  "nombre_solicitante": "Sandra",
  "id_mascota": 1,
  "estado": "Aceptada"
}
###
```

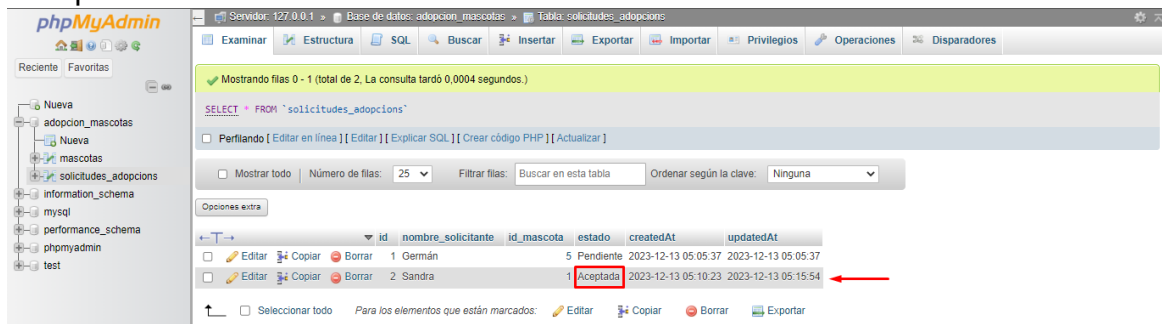
Respuesta recibida



Response(19ms) X

```
1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 34
5 ETag: W/"22-E2y8aRbSHGARTuWgI/yV7FkDoBc"
6 Date: Wed, 13 Dec 2023 05:15:54 GMT
7 Connection: close
8
9 {
10   "mensaje": "Registro Actualizado"
11 }
```

Comprobación en la base de datos



Mostrando filas 0 - 1 (total de 2. La consulta tardó 0,0004 segundos.)

SELECT * FROM `solicitudes_adopciones`

Perfilando [Editar en línea] [Editar] [Explicar SQL] [Crear código PHP] [Actualizar]

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla | Ordenar según la clave: Ninguna

Opciones extra

	id	nombre_solicitante	id_mascota	estado	createdAt	updatedAt
<input type="checkbox"/>	1	Germán	5	Pendiente	2023-12-13 05:05:37	2023-12-13 05:05:37
<input type="checkbox"/>	2	Sandra	1	Aceptada	2023-12-13 05:10:23	2023-12-13 05:15:54

Seleccionar todo | Para los elementos que están marcados: Editar | Copiar | Borrar | Exportar

Petición realizada

```
###
Send Request
DELETE http://localhost:8000/solicitudes/eliminar/2 HTTP/1.1
```

Respuesta recibida

```
Response(306ms) X

1 HTTP/1.1 200 OK
2 X-Powered-By: Express
3 Content-Type: application/json; charset=utf-8
4 Content-Length: 32
5 ETag: W/"20-hkH0JCv/EfydTXVDeEcSo30bS8A"
6 Date: Wed, 13 Dec 2023 05:18:31 GMT
7 Connection: close
8
9 {
10   "mensaje": "Registro Eliminado"
11 }
```

Comprobación en la base de datos



Mostrando filas 0 - 0 (total de 1. La consulta tardó 0,0003 segundos.)

SELECT * FROM `solicitudes_adopciones`

Perfilando [Editar en línea] [Editar] [Explicar SQL] [Crear código PHP] [Actualizar]

Mostrar todo | Número de filas: 25 | Filtrar filas: Buscar en esta tabla

Opciones extra

	id	nombre_solicitante	id_mascota	estado	createdAt	updatedAt
<input type="checkbox"/>	1	Germán	5	Pendiente	2023-12-13 05:05:37	2023-12-13 05:05:37

Seleccionar todo | Para los elementos que están marcados: Editar | Copiar | Borrar | Exportar