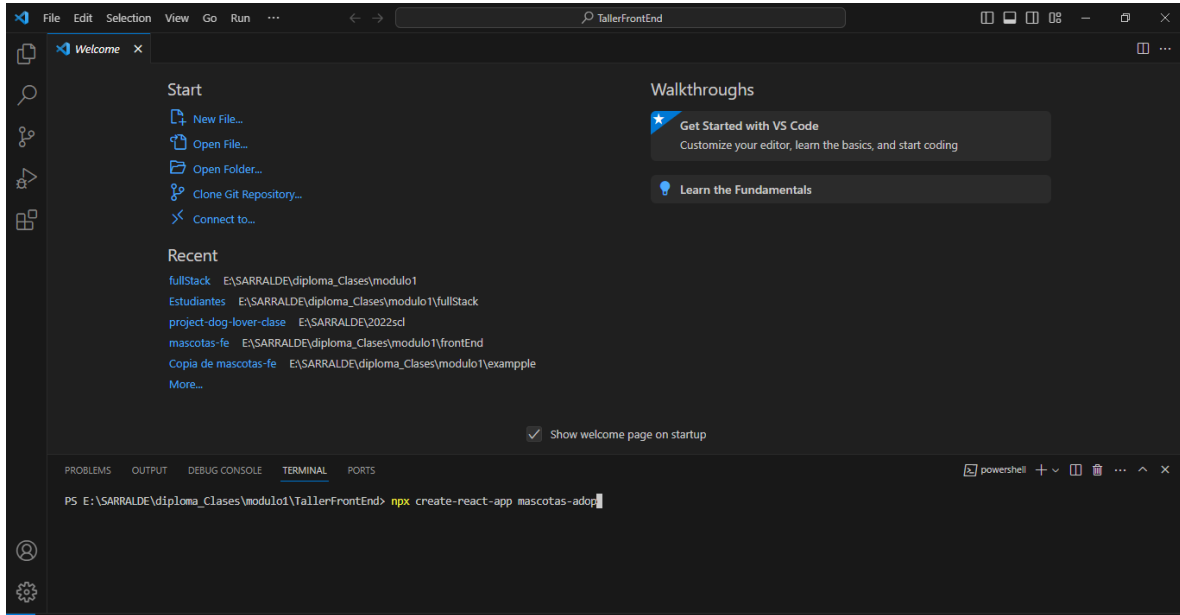


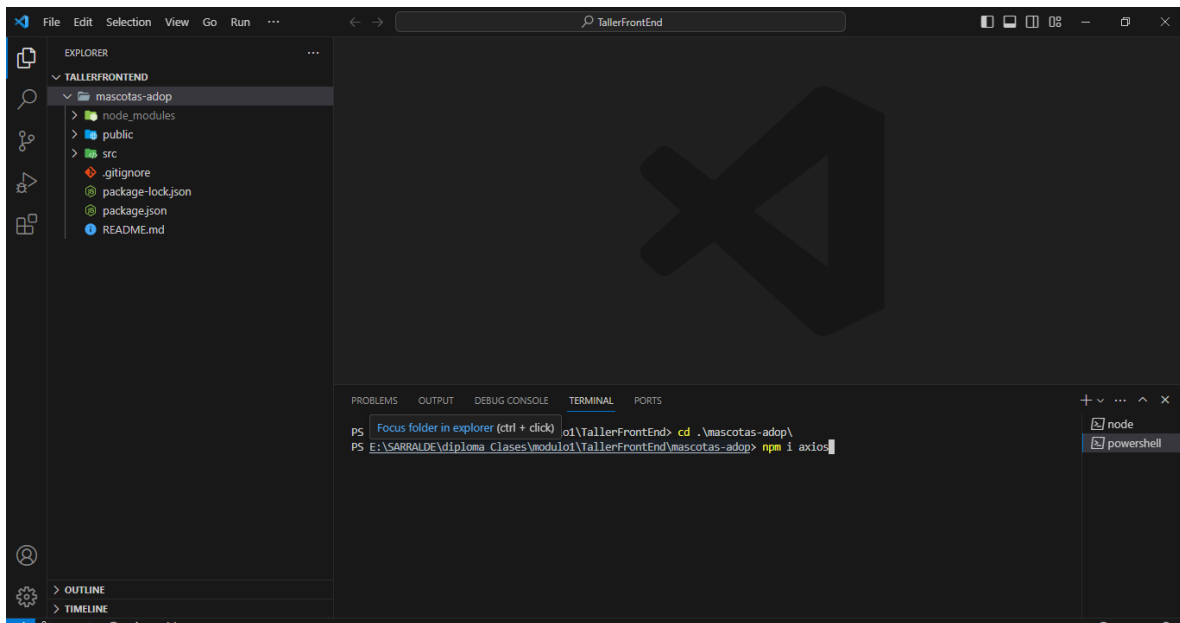
Taller de FrontEnd final

Con Npx ejecutaremos un scrip para el proyecto.

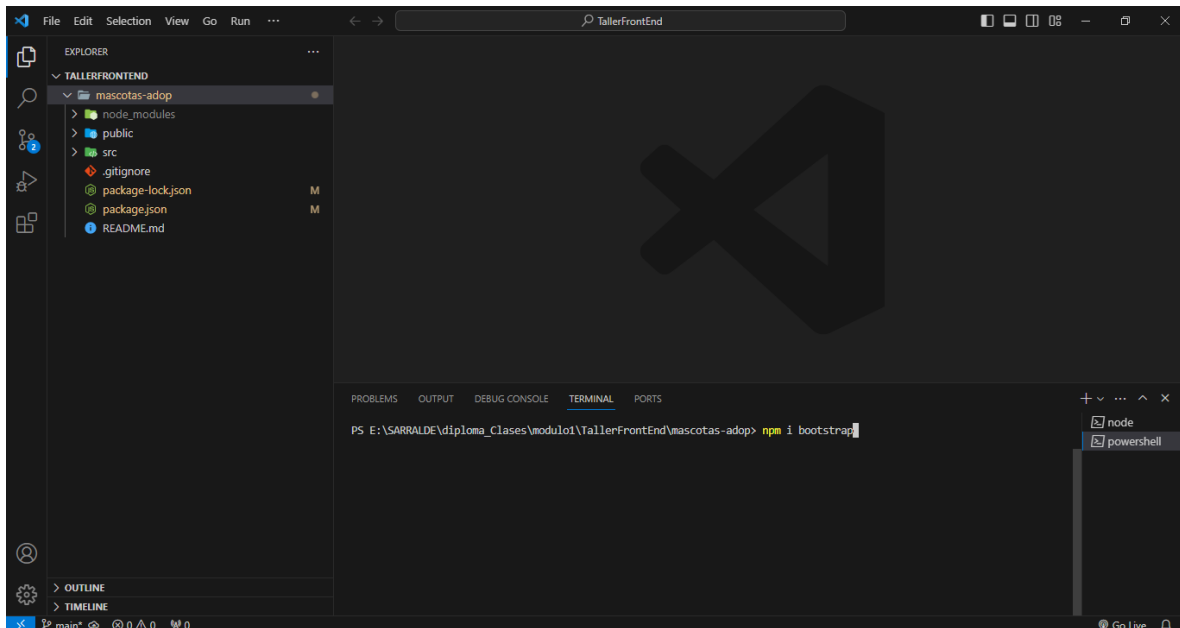
Creamos un proyecto inicial con React el cual lo llamaremos **mascotas-adop**



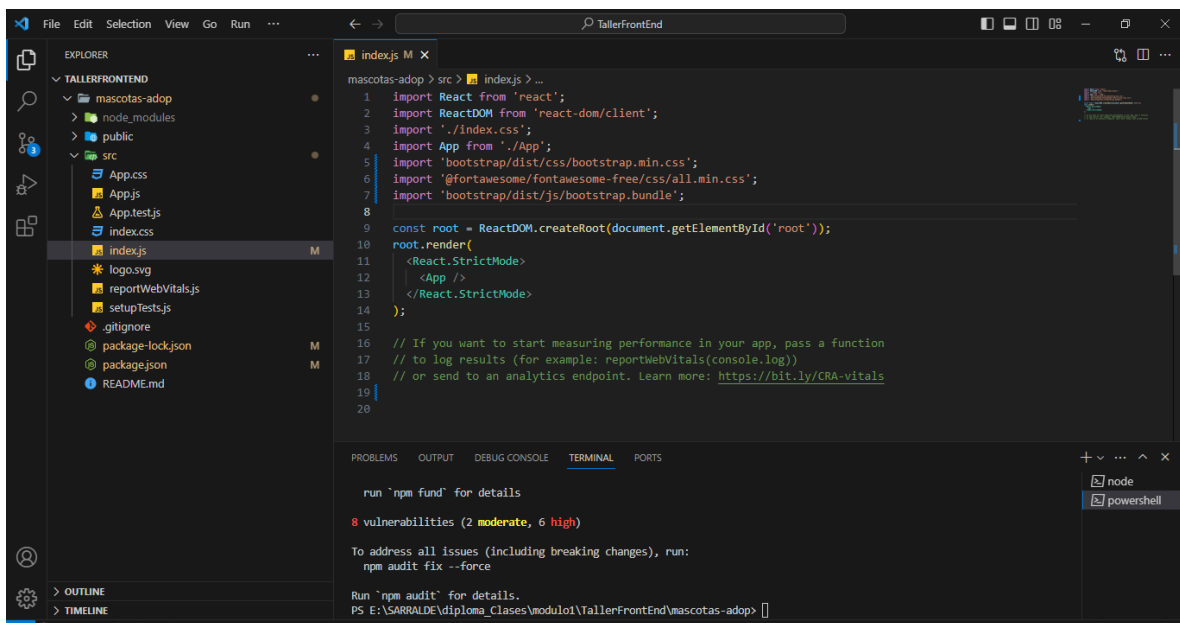
Ahora vamos a instalar **axios** la cual nos permite hacer consulta al api del backend



Instalaremos el bootstrap para manejar los diseños



Luego vamos a importar las cosas que necesitamos para nuestro proyecto



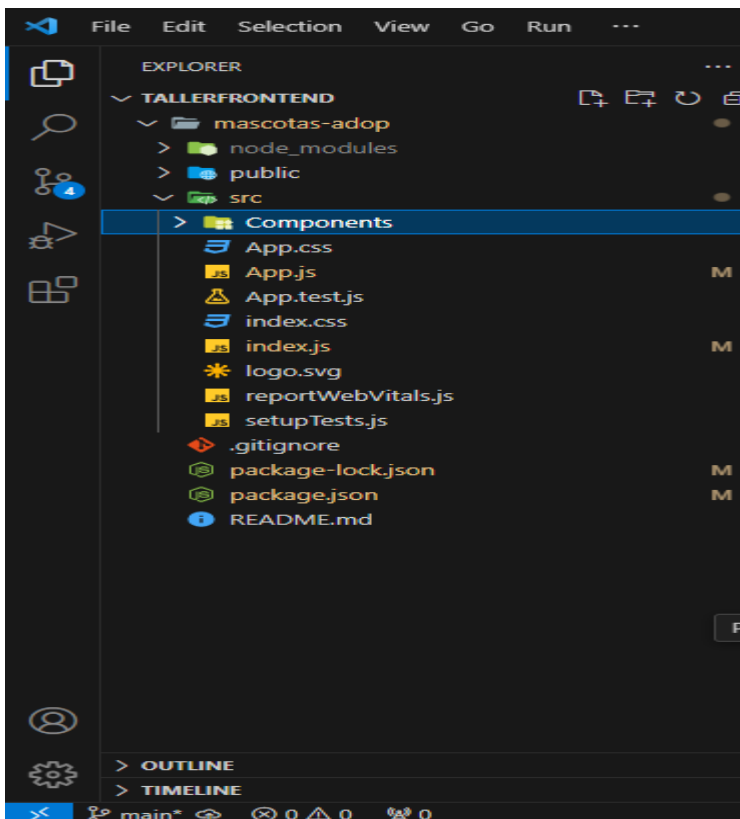
Aquí importamos unos módulos para manejar rutas

Y definimos unos componentes

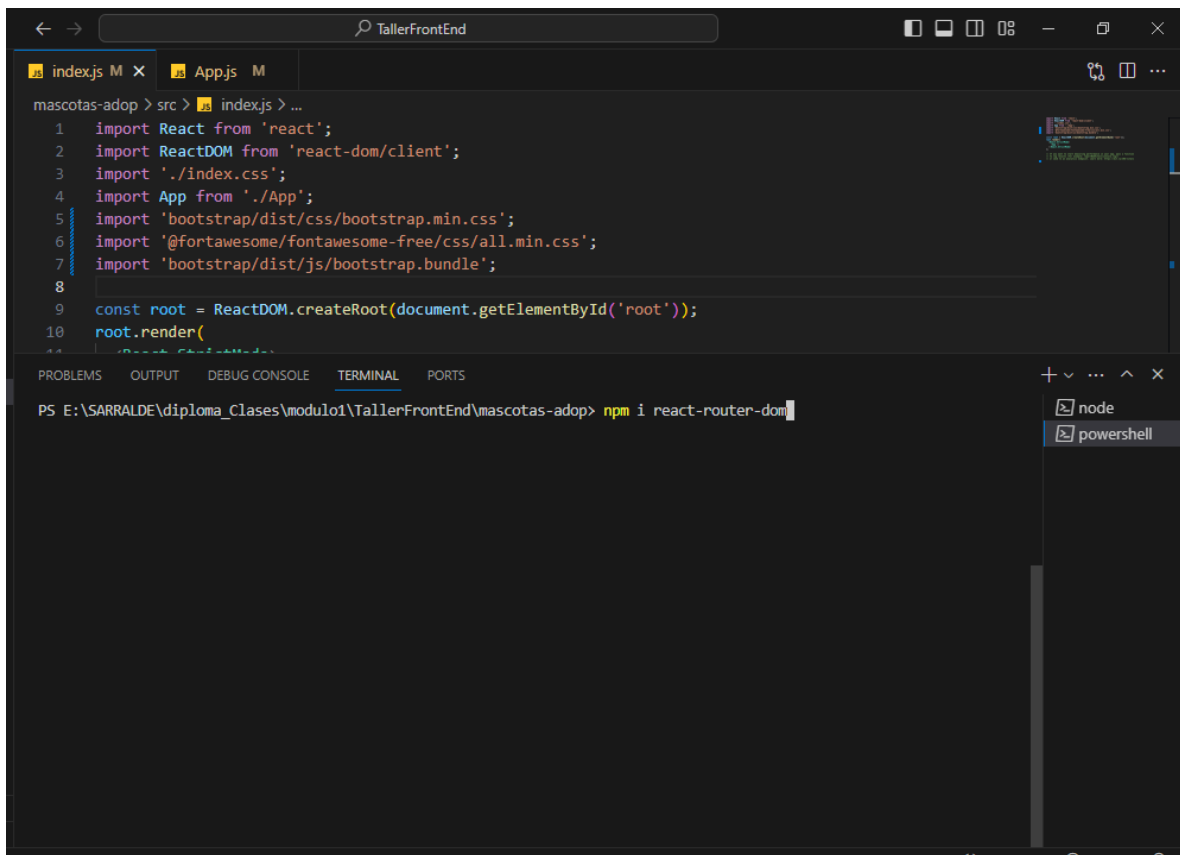
```
src > App.js > ...
1 | import {BrowserRouter, Routes, Route} from 'react-router-dom';
2 | import MascotasComponent from './Components/MascotasComponent.js';
3 | import './App.css';
4 |
5 | function App() {
6 |   return (
7 |     <BrowserRouter>
8 |       <Routes>
9 |         <Route path="/" element={<MascotasComponent></MascotasComponent>}></Route>
10 |       </Routes>
11 |     </BrowserRouter>
12 |   );
13 | }
14 |
15 | export default App;
17
```

Vamos crear una carpeta de componentes dentro src

Y dentro de componentes un archivo component



Instalamos el react-router-dom



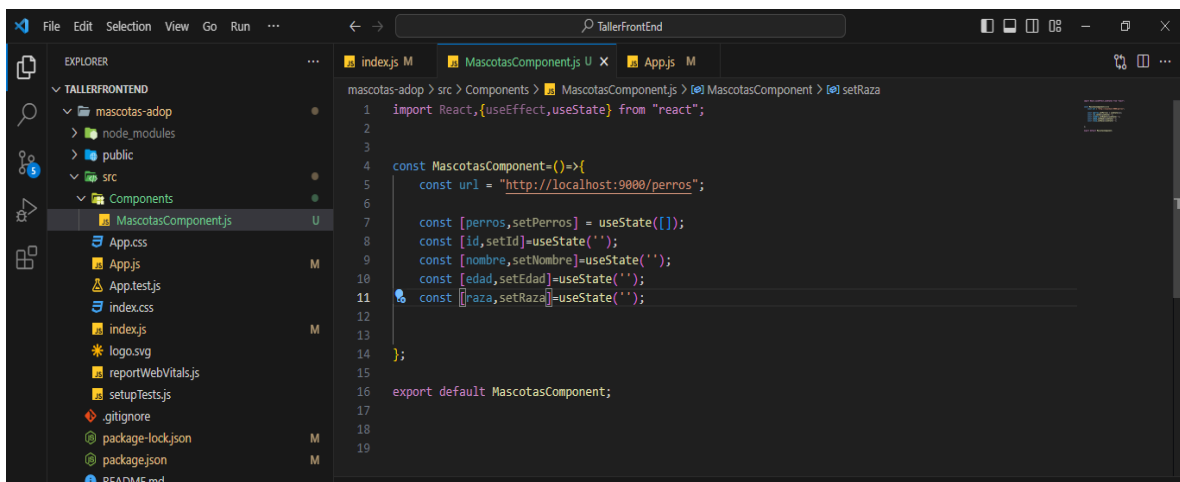
```
indexjs M x Appjs M
mascotas-adop > src > indexjs > ...
1 import React from 'react';
2 import ReactDOM from 'react-dom/client';
3 import './index.css';
4 import App from './App';
5 import 'bootstrap/dist/css/bootstrap.min.css';
6 import '@fortawesome/fontawesome-free/css/all.min.css';
7 import 'bootstrap/dist/js/bootstrap.bundle';
8
9 const root = ReactDOM.createRoot(document.getElementById('root'));
10 root.render(
11   <React.StrictMode>
12     <App />
13   </React.StrictMode>
14 );
15
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
PS E:\SARRALDE\diploma_clases\modulo1\TallerFrontEnd\mascotas-adop> npm i react-router-dom
```

Vamos a definir una contante **url** donde colocaremos la ruta del proyecto del backend

Y importamos useEfect lo cual nos ayuda a manejar los estados dentro de la aplicación

Definimos arreglos y como va recibir todas las mascotas le enviamos vacío el **const perros**

Luego agregamos las variables que necesitamos



```
File Edit Selection View Go Run ...
TallerFrontEnd
mascotas-adop
node_modules
public
src
Components
MascotasComponent.js U
App.css
App.js M
App.test.js
index.css
index.js M
logo.svg
reportWebVitals.js
setupTests.js
.gitignore
package-lock.json M
package.json M
README.md

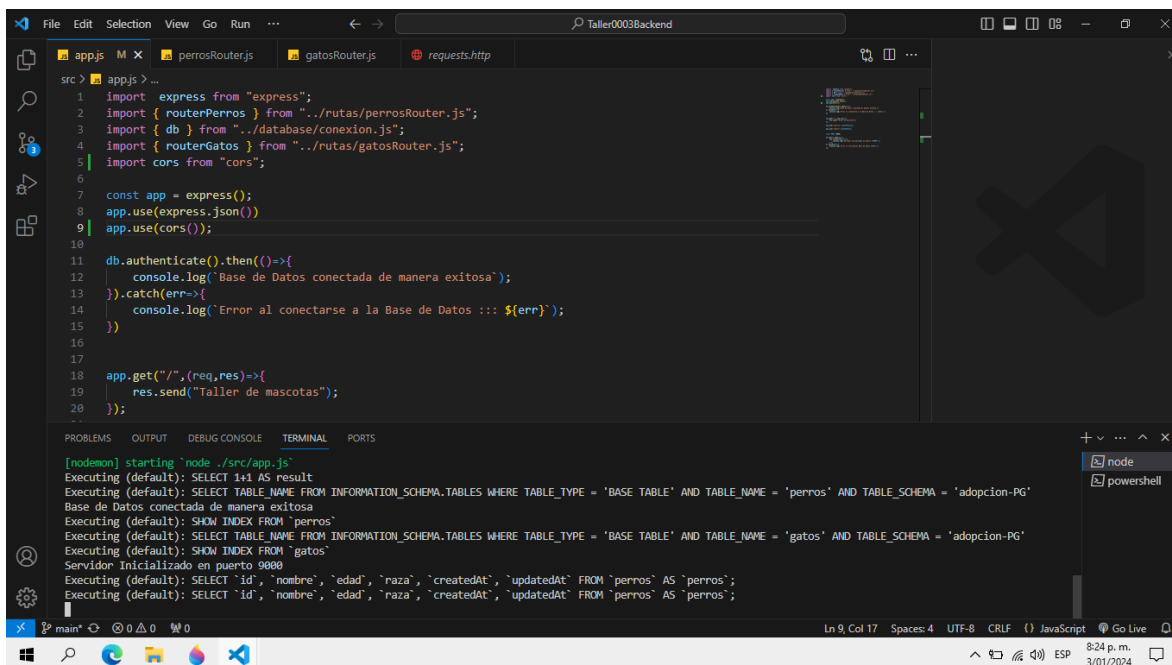
indexjs M MascotasComponent.js U Appjs M
mascotas-adop > src > Components > MascotasComponent.js > MascotasComponent > setRaza
1 import React,{useEffect,useState} from 'react';
2
3
4 const MascotasComponent=()=>{
5   const url = "http://localhost:9000/perros";
6
7   const [perros,setPerros] = useState([]);
8   const [id,setId]=useState('');
9   const [nombre,setNombre]=useState('');
10  const [edad,setEdad]=useState('');
11  const [raza,setRaza]=useState('');
12
13
14 };
15
16 export default MascotasComponent;
17
18
19
```

Aquí usamos el useEffect para sacar la lista de las mascotas y luego

Defino una const getPerros para hacer consulta al api

```
12     const [raza, setRaza] = useState('');
13
14     useEffect(() => {
15       getPerros();
16     }, []);
17
18     const getPerros = async () => {
19       const respuesta = await axios.get(`${url}/buscar`);
20       setPerros(respuesta.data);
21     };
22
23   };
24
25   export default MascotasComponent;
26
```

En la parte del backend tendremos que instalar cors e impórtalo ya que sin esta parte cuando hacemos el llamado desde el front nos va denegar la información



```
File Edit Selection View Go Run ... Taller0003Backend
src > app.js > ...
1 import express from "express";
2 import { routerPerros } from "../rutas/perrosRouter.js";
3 import { db } from "../database/conexion.js";
4 import { routerGatos } from "../rutas/gatosRouter.js";
5 import cors from "cors";
6
7 const app = express();
8 app.use(express.json());
9 app.use(cors());
10
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 app.get("/", (req, res) => {
18   res.send("Taller de mascotas");
19 });
20
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
[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 = 'perros' AND TABLE_SCHEMA = 'adopcion-PG'
Base de Datos conectada de manera exitosa
Executing (default): SHOW INDEX FROM 'perros'
Executing (default): SELECT TABLE_NAME FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_TYPE = 'BASE TABLE' AND TABLE_NAME = 'gatos' AND TABLE_SCHEMA = 'adopcion-PG'
Executing (default): SHOW INDEX FROM 'gatos'
Servidor Inicializado en puerto 9000
Executing (default): SELECT 'id', 'nombre', 'edad', 'raza', 'createdAt', 'updatedAt' FROM 'perros' AS 'perros';
Executing (default): SELECT 'id', 'nombre', 'edad', 'raza', 'createdAt', 'updatedAt' FROM 'perros' AS 'perros';
```

Ln 9, Col 17 Spaces: 4 UTF-8 CRLF JavaScript Go Live 8:24 p. m. 3/01/2024

Definimos un arreglo para visualizar los datos con un map

Y definimos una llave única que sería nuestro id

```
perros.map((perro, i) => (  
  <tr key={perro.id}>  
    <td>{perro.id}</td>  
    <td>{perro.nombre}</td>  
    <td>{perro.edad}</td>  
    <td>{perro.raza}</td>  
    <td>{perro.url}</td>  
    <td>{perro.descripcion}</td>  
    <td>
```

Vamos a crear el botón para añadir nuevas mascotas

Después vamos a crear un modal para cuando le demos al botón no salga una ventana emergente y colocar los datos de la mascota

En este modal que colocaremos 2 opciones una crear un registro y otro si es para actualizar el registro

```
29  
30   const openModal = (opcion, id, nombre, edad, raza, descripcion) => {  
31     setId('');  
32     setNombre('');  
33     setEdad('');  
34     setRaza('');  
35     setDescripcion('');  
36     setOperacion(opcion);  
37     if (opcion == 1) {  
38       setTitulo("Registrar Perros")  
39     }  
40     else if (opcion == 2) {  
41       setTitulo("Editar Nombre");  
42       setId(id);  
43       setNombre(nombre);  
44       setEdad(edad);  
45       setRaza(raza);  
46       setDescripcion('');  
47     }  
48   };
```

Y luego aplicamos el modal en el botón añadir para que pueda realizar la acción con el onClick

```
onClick={() => openModal(1)}
className="btn btn-dark"
class="my-butt"
data-bs-toggle="modal"
data-bs-target="#modalMascotas"
```

Vamos a crear una cont de validar

Hacemos validación de los campos a la hora de escribir par que no queden vacíos

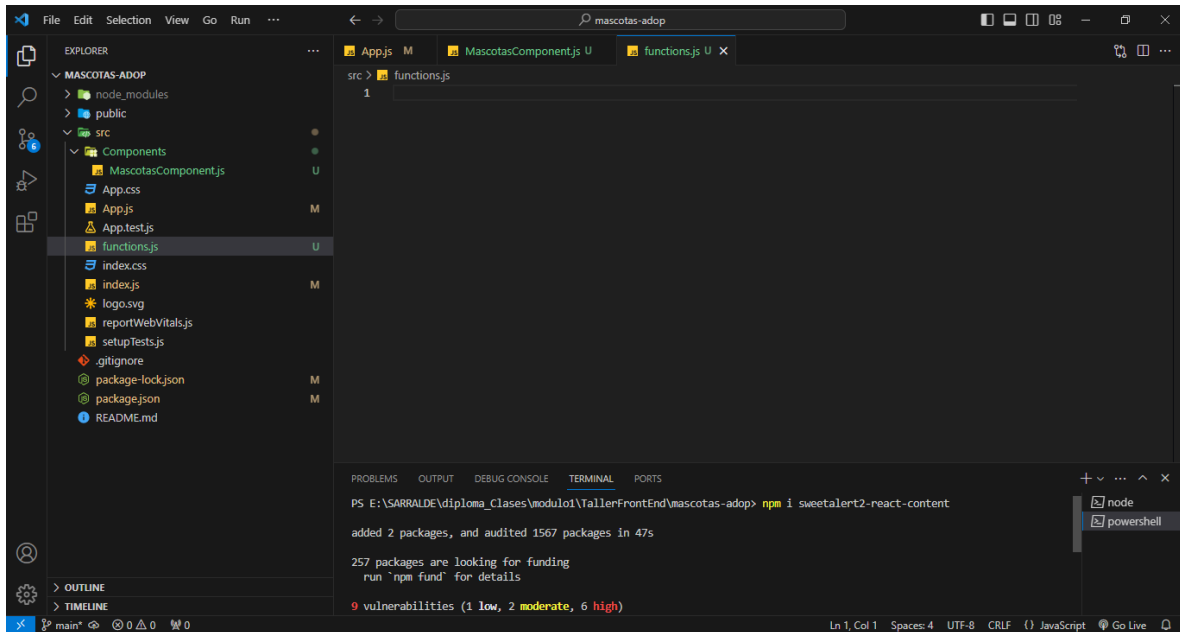
También vamos a programar una alerta la cual se mostrará en una venta emergente

También colocaremos una url para completar el verbo (**crear, actualizar**)

```
50 const validar = ()=>{
51   let parametros;
52   let metodo;
53   if(nombre.trim()==''){
54     console.log("Debe escribir un Nombre");
55     mostrarAlerta("Debe escribir un Nombre")
56   }
57   else if(edad.trim()==''){
58     console.log("Debe escribir una edad");
59     mostrarAlerta("Debe escribir una edad")
60   }
61   else if(raza.trim()==''){
62     console.log("Debe escribir la raza");
63     mostrarAlerta("Debe escribir la raza")
64   }
65   else if(descrip_cor.trim()==''){
66     console.log("Debe escribir una descriocion corta");
67     mostrarAlerta("Debe escribir una drecripcion corta")
68   }
69   else{
70     if(operacion==1){
71       parametros={
72         urlExt: `${url}/crear`,
73         nombre: nombre.trim(),
74         edad: edad.trim(),
75         raza: raza.trim(),
76         descrip_cor: descrip_cor.trim()
77       };
78       metodo="POST";
79     }
80     else{
81       parametros={
82         urlExt: `${url}/actualizar/${id}`,
83         nombre: nombre.trim(),
84         edad: edad.trim(),
85         raza: raza.trim(),
86         descrip_cor: descrip_cor.trim()
87       };
88       metodo="PUT";
89     }
90     enviarSolicitud(metodo, parametros);
91   }
92 };
93
```

Vamos a crear un `function.js` para crear componentes apartes

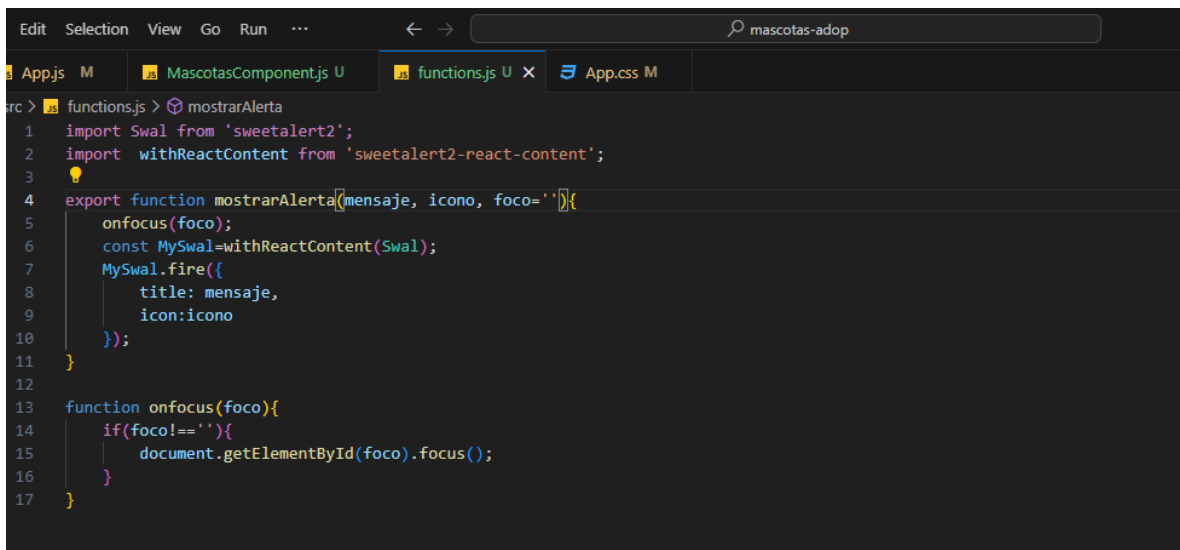
Debemos instalar **sweet-alert** para que funcione



Creamos una función **alert** para poder utilizarla

la cual tendrá un **mensaje, icono y foco**

y luego vamos a importar esta función en el `MascotasComponent`



A la hora de añadir un registro en el la ventana se debe enlazar

Y así mismo los hacemos con los demás campos

```
<input
  type="text"
  id="nombre"
  className="form-control"
  placeholder="Nombre"
  value={nombre}
  onChange={(e) => setNombre(e.target.value)}
></input>
</div>
```

Luego en el botón actualizar

Cuando la demos en el botón actualizar nos debe mostrar los datos que vamos a actualizar en la ventana por eso colocamos el modal

```
onClick={() => openModal(2, perro.id, perro.nombre, perro.edad, perro.raza, perro.descrip_cor)}
className="btn btn-warning"
data-bs-toggle="modal"
data-bs-target="#modalMascotas">
<i className="fa-solid fa-edit"></i>
```

Vamos a crear una función para el botón eliminar

En la cual solo se pedirá el id y el nombre

```
const eliminarMascota=(id,nombre)=>{
  const MySwal = withReactContent(Swal);
  MySwal.fire({
    title: `Estas seguro de eliminar la mascota ${nombre} ?`,
    icon: 'question',
    text: 'Se eliminará Definitivamente',
    showCancelButton: true,
    confirmButtonText: 'Si, eliminar',
    cancelButtonText: 'Cancelar'
  }).then((result)=>{
    if(result.isConfirmed){
      setId(id);
      enviarSolicitud("DELETE",{urlExt: `${url}/eliminar/${id}`,id:id})
    }
    else{
      mostrarAlerta("No se elimino la mascota","info");
    }
  })
}
```

Luego en el botón de eliminar debes colocar la función eliminar con el onClick

La cual cuando se elimine solo va a pedir el id y el nombre para poder eliminar

```
<button
  onClick={() => eliminarMascota(perro.id, perro.nombre)}
  className="btn btn-danger">
  <i className="fa-solid fa-edit"></i>
</button>
</td>
```

- Vamos a crear una paginación

Vamos a crear unas constantes

La cual nos permite hacer la paginación

```
const [currentPage, setCurrentPage] = useState(1)
const recordsPerPage = 3;
const lastIndex = currentPage * recordsPerPage;
const firstIndex = lastIndex - recordsPerPage;
const records = perros.slice(firstIndex, lastIndex);
const npage = Math.ceil(perros.length / recordsPerPage);
const numbers = [...Array(npage + 1).keys()].slice(1);
```

En este código lo que hacemos es crear los botones para la paginación

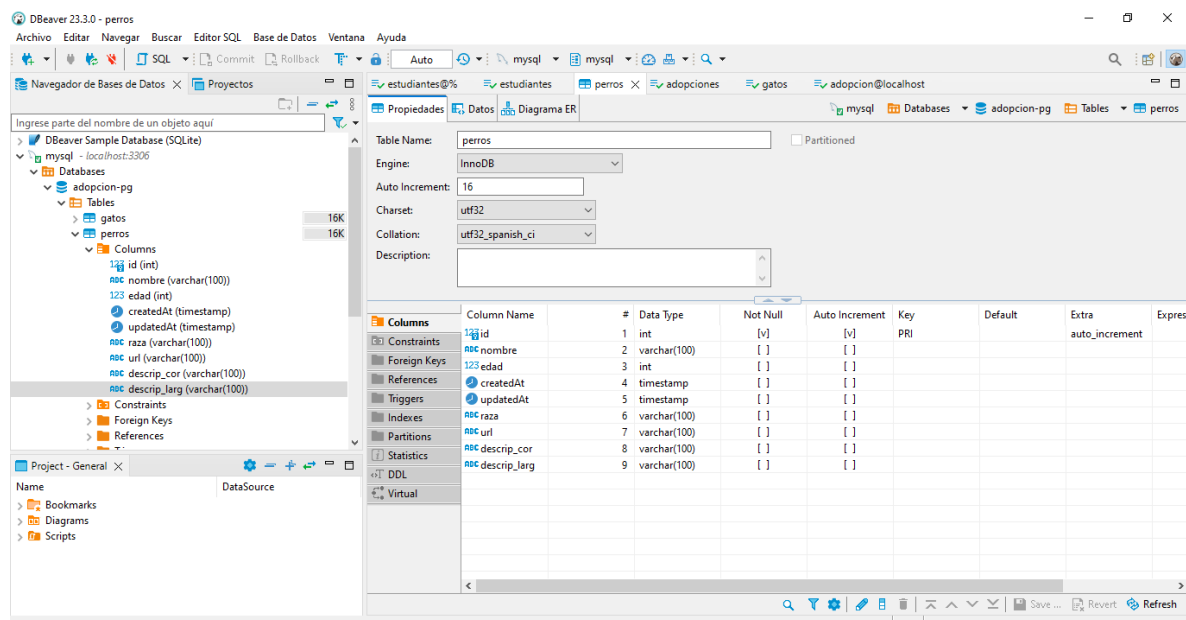
```
<nav>
  <ul className="pagination">
    <li className="page-item">
      <a href="#" className="page-link" onClick={prePage}>prev</a>
    </li>
    {
      numbers.map((n, i) => (
        <li className={`page-item ${currentPage === n ? 'active' : ''}`} key={i}>
          <a href="#" className="page-link"
            onClick={() => changeCPage(n)}>{n}</a>
        </li>
      ))
    }
    <li className="page-item">
      <a href="#" className="page-link" onClick={nextPage}>next</a>
    </li>
  </ul>
</nav>
```

Creamos las casillas faltantes para

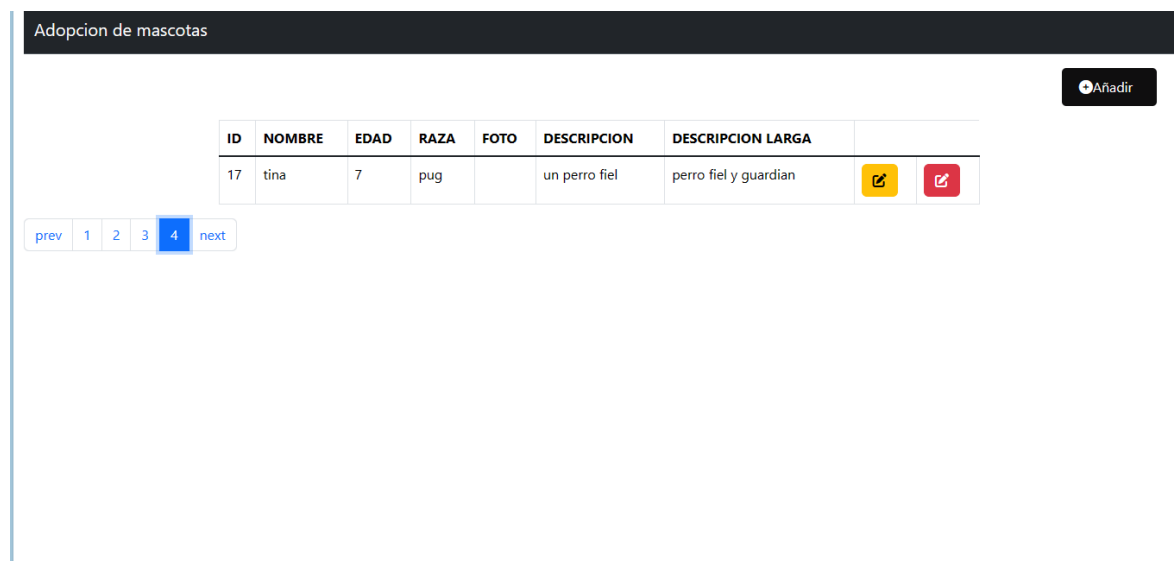
La foto

Descripción corta

Descripción larga



Mostramos la paginación



Mostramos el registro de una mascota


Adopcion de mascotas


| ID | NOMBRE | EDAD |
|----|--------|------|
| 5 | luca | 5 |
| 7 | jako | 1 |
| 8 | mau | 8 |


prev 1 2 3 4 next


Añadir


Registrar Perros

 holga

 9

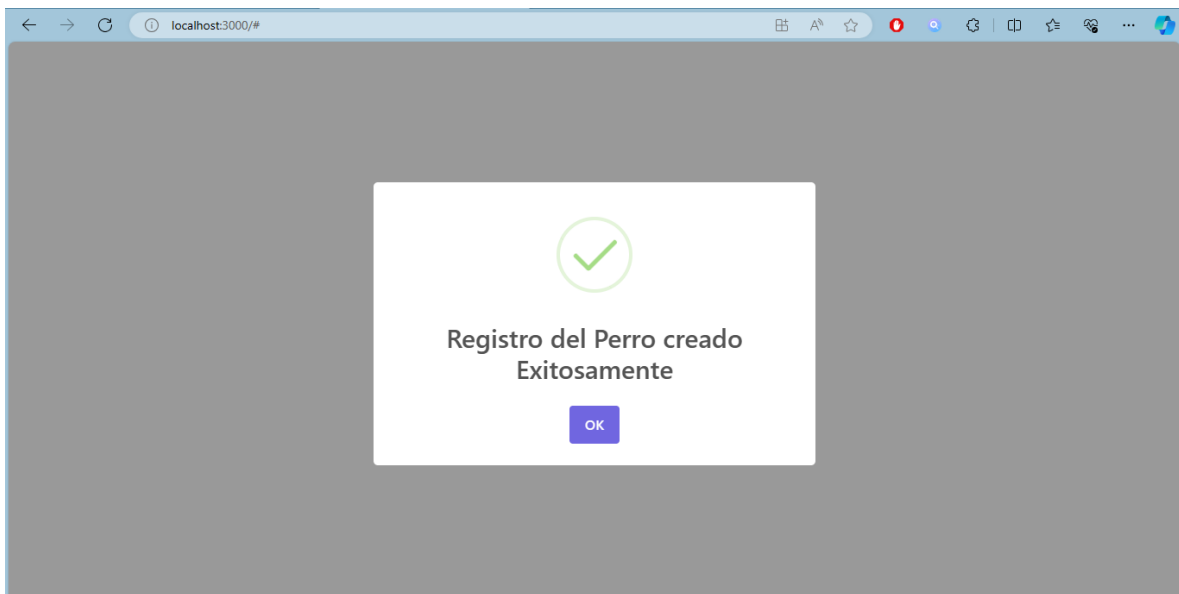
 labrado

 un perro de casa

 un perro con un sentido de protección

Guardar

Cerrar



Mostramos que la mascota que se creo se muestra en la tabla

Adopcion de mascotas

Añadir

| ID | NOMBRE | EDAD | RAZA | FOTO | DESCRIPCION | DESCRIPCION LARGA | | |
|----|--------|------|---------|------|------------------|---------------------------------------|--|--|
| 17 | tina | 7 | pug | | un perro fiel | perro fiel y guardian | | |
| 18 | holga | 9 | labrado | | un perro de casa | un perro con un sentido de protección | | |

prev

1

2

3

4

next

Adopcion de mascotas

Añadir

| ID | NOMBRE | EDAD | RAZA | FOTO | DESCRIPCION | DESCRIPCION LARGA | | |
|----|--------|------|----------|------|--|---|--|--|
| 13 | zoco | 8 | pug | | son perro normalmente de estatura baja | | | |
| 14 | mike | 1 | labrador | | son perro files | | | |
| 16 | milo | 2 | rotwile | | es un perro fiel | es un perro agresivo cuando cuida las cosas | | |

prev

1

2

3

4

next