Diplomado de actualización en nuevas tecnologías para el desarrollo de software

Taller Final

Hecho por:

Anthony Danilo Parra

Universidad de Nariño

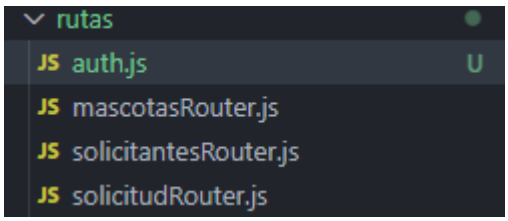
San Juan de Pasto

2023

Para el taller final se hizo la Funcionalidad de Login y Búsqueda y se hicieron las siguientes modificaciones

Para el login:

En la parte del servidor se colocó una ruta llamada auth.js



Dentro se encuentra el codigo para el login, en el cual hay un *Módulo que define las rutas relacionadas con la autenticación de usuarios.*

Utiliza Express y Sequelize para interactuar con la base de datos MySQL.

Endpoint de inicio de sesión.

Permite a los usuarios iniciar sesión comparando las credenciales proporcionadas con las almacenadas en la base de datos.

```
});
```

Endpoint de registro.

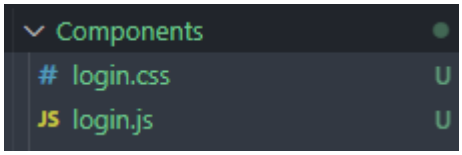
Permite a los usuarios registrarse almacenando sus credenciales en la base de datos.

```
 *
 * @function
 * @name POST /register
 * @param {Object} req - Objeto de solicitud de Express.
 * @param {Object} res - Objeto de respuesta de Express.
 * @returns {Object} - Objeto JSON con un mensaje indicando el resultado del registro.
 */
router.post('/register', async (req, res) => {
  const { username, password } = req.body;

  try {
    // Insertar el nuevo usuario en la base de datos
    const result = await db.query('INSERT INTO usuarios (nombre_usuario, contrasena) VALUES (?, ?)', { replacements: [username, password] });

    res.send({ message: 'Usuario registrado exitosamente' });
  } catch (error) {
    console.error('Error en el registro:', error);
    res.status(500).send({ error: 'Error en el servidor' });
  }
});

// Exportar el router para su uso en otros módulos
export { router };
```

Por otro lado, en Mysql hay que crear una tabla usuario, para lo cual se utilizó el siguiente codigo:

```sql
CREATE TABLE usuarios (
  id INT AUTO_INCREMENT PRIMARY KEY,
  nombre_usuario VARCHAR(255) NOT NULL,
  contrasena VARCHAR(255) NOT NULL
);
```

Ahora en React se hizo tro componente llamadao login.js



Dentro de login se encuentra, *Componente de React que representa la página de inicio de sesión y registro.*

Permite a los usuarios autenticarse o registrarse, mostrando mensajes de éxito o error.

Maneja la autenticación (inicio de sesión o registro) al enviar una solicitud al servidor.

Redirige a la página de mascotas después de un inicio de sesión exitoso.

```
};
```

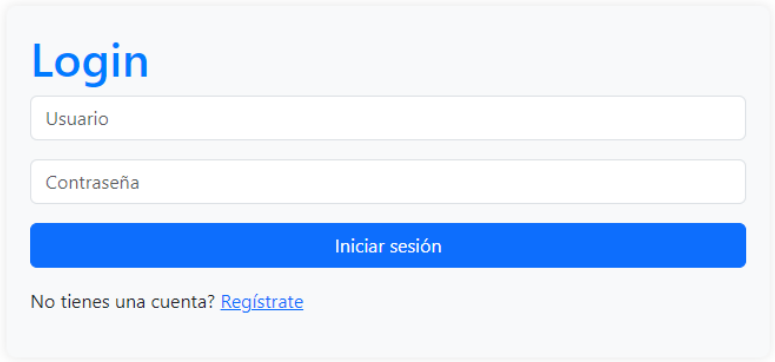
```
// Renderiza el formulario de inicio de sesión o registro
return (
  <div className="container mt-5">
    <div className="col-md-6 offset-md-3 auth-form">
      <h1>{isLogin ? 'Login' : 'Registro'}</h1>
      <form onSubmit={handleAuth}>
        <div className="mb-3">
          <input
            type="text"
            className="form-control"
            placeholder="Usuario"
            value={username}
            onChange={(e) => setUsername(e.target.value)}
          />
        </div>
        <div className="mb-3">
          <input
            type="password"
            className="form-control"
            placeholder="Contraseña"
            value={password}
            onChange={(e) => setPassword(e.target.value)}
          />
        </div>
        <button type="submit" className="btn btn-primary">
          {isLogin ? 'Iniciar sesión' : 'Registrarse'}
        </button>
      </form>
```

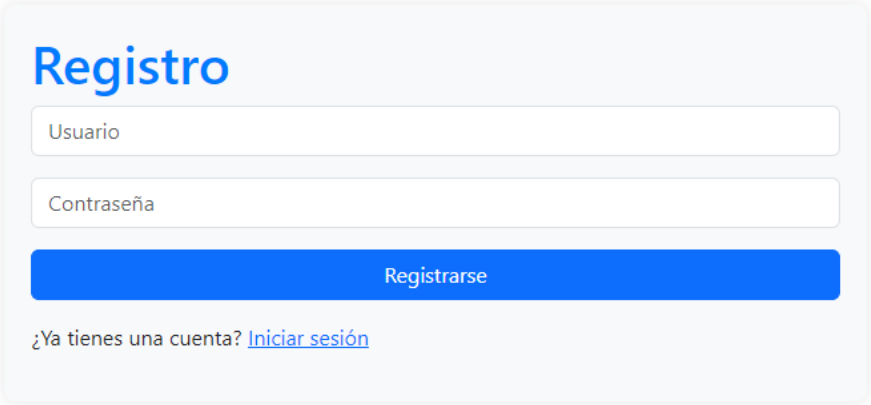
```jsx
        {/* Muestra el mensaje de éxito o error */}
        {message && <p className="mt-3">{message}</p>}

        {/* Muestra enlaces para cambiar entre inicio de sesión y registro */}
        {isLogin ? (
          <p className="mt-3">
            No tienes una cuenta?{' '}
            <Link to="#" onClick={() => setIsLogin(false)}>
              Regístrate
            </Link>
          </p>
        ) : (
          <p className="mt-3">
            ¿Ya tienes una cuenta?{' '}
            <Link to="#" onClick={() => setIsLogin(true)}>
              Iniciar sesión
            </Link>
          </p>
        )}
      </div>
    </div>
  );
};

export default Login;
```

Por otro lado en login.css se metieron algunos estilos

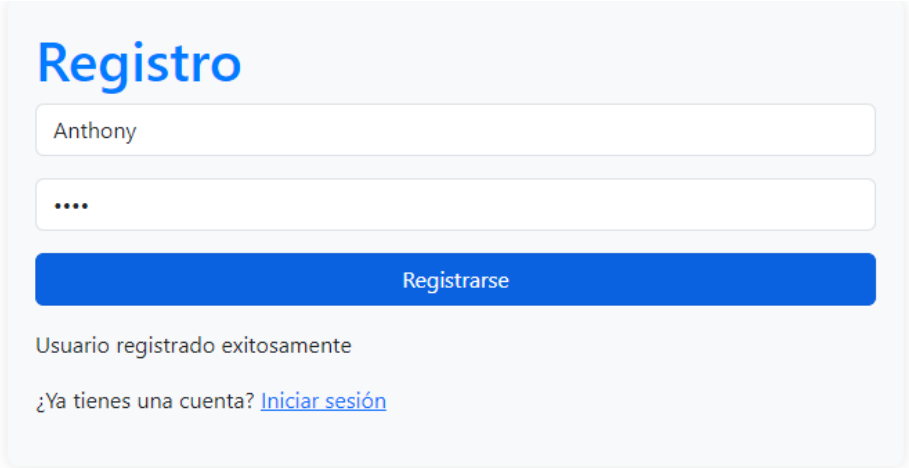
```
Click here to ask Blackbox to help you code faster
.auth-form {
    background-color: #f8f9fa;
    padding: 20px;
    border-radius: 8px;
    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
  }

  .auth-form h1 {
    color: #007bff;
  }

  .auth-form button {
    width: 100%;
  }
```

Y visualmente queda así:





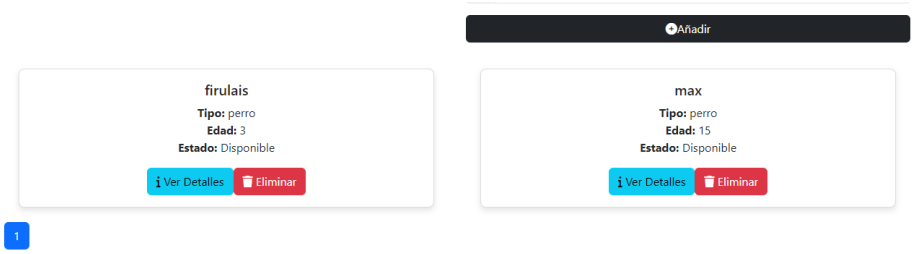
El registro de un usuario



Se ve reflejado en la base de datos

7	Anthony	1234
NULL	NULL	NULL

Y cuando inicia sesión se redirige directamente a la pagina de mascotas



Ahora para la busqueda se hizo el siguiente codigo:

Se declara una nueva variable

```
const [busqueda, setBusqueda] = useState("");
```

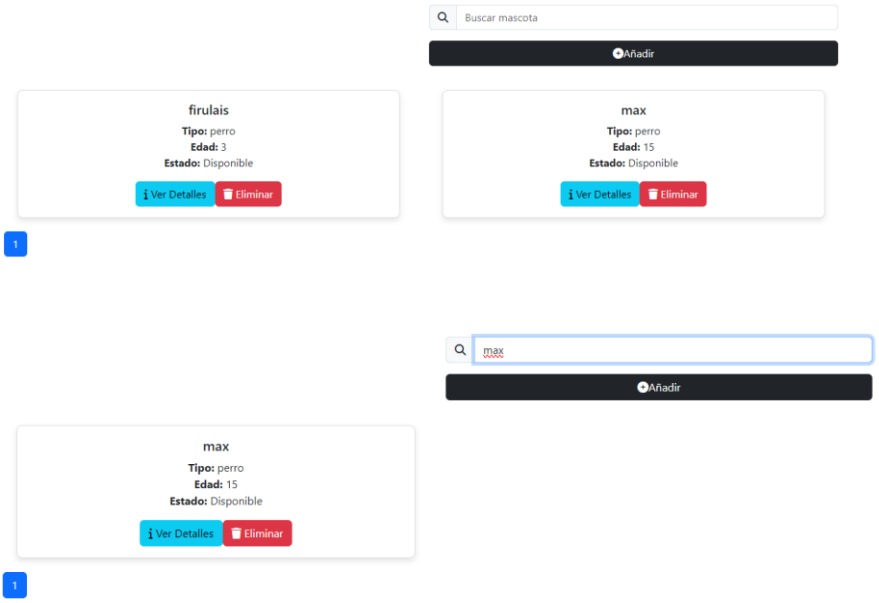
Dentro de renderizarMascotasPorPagina se filtran las mascotas

```
// Filtra las mascotas actuales según el término de búsqueda
const mascotasFiltradas = mascotasActuales.filter((mascota) =>
  mascota.nombre.toLowerCase().includes(busqueda.toLowerCase())
);

// Calcula el número total de filas necesarias
const totalFilas = Math.ceil(mascotasFiltradas.length / 3);
```

Y en el return, se coloca un nuevo div encima del boton de añadir

```
<div className="input-group mb-3">
  <span className="input-group-text">
    <i className="fa-solid fa-search"></i>
  </span>
  <input
    type="text"
    className="form-control"
    placeholder="Buscar mascota"
    value={busqueda}
    onChange={(e) => setBusqueda(e.target.value)}
  />
</div>
```

Y así, se vería visualmente



Cabe recalcar, que para solicitantes y solicitudes funciona exactamente igual