

## **Manual de informacion, instalacion y uso de My U Library (Jorge Alexander Parada Aldana)**

### **INFORMACION:**

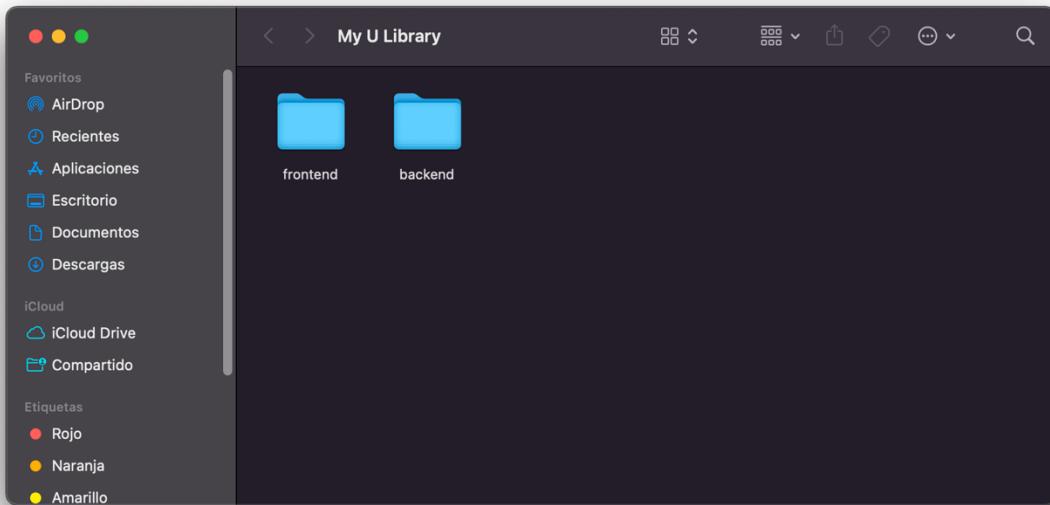
My U Library es un pequeño sistema de biblioteca creado con ReactJS, NodeJS y PostgreSQL (utilizando JWT, Redux, Sequelize, Express)

Tecnologias, IDE's y OS utilizados (Cada uno en su version de arquitectura ARM64 compatible con el chip Apple Silicon M1):

- Version de NodeJS: 17.5.0 (Febrero 2022)
- Postman v9.12.2
- Version de ReactJS: 17.0.2
- Version de PostgreSQL: 14.2 (pgAdmin4, SQL Shell psql)
- Version de npm: 8.4.1
- Visual Studio Code

### **INSTALACION:**

- 1- Descarga el proyecto desde el repositorio de Git-Hub y descomprimelo
- 2- Abre la carpeta, dentro de ella visualizaras 2 carpetas mas, Frontend y Backend junto con este manual.



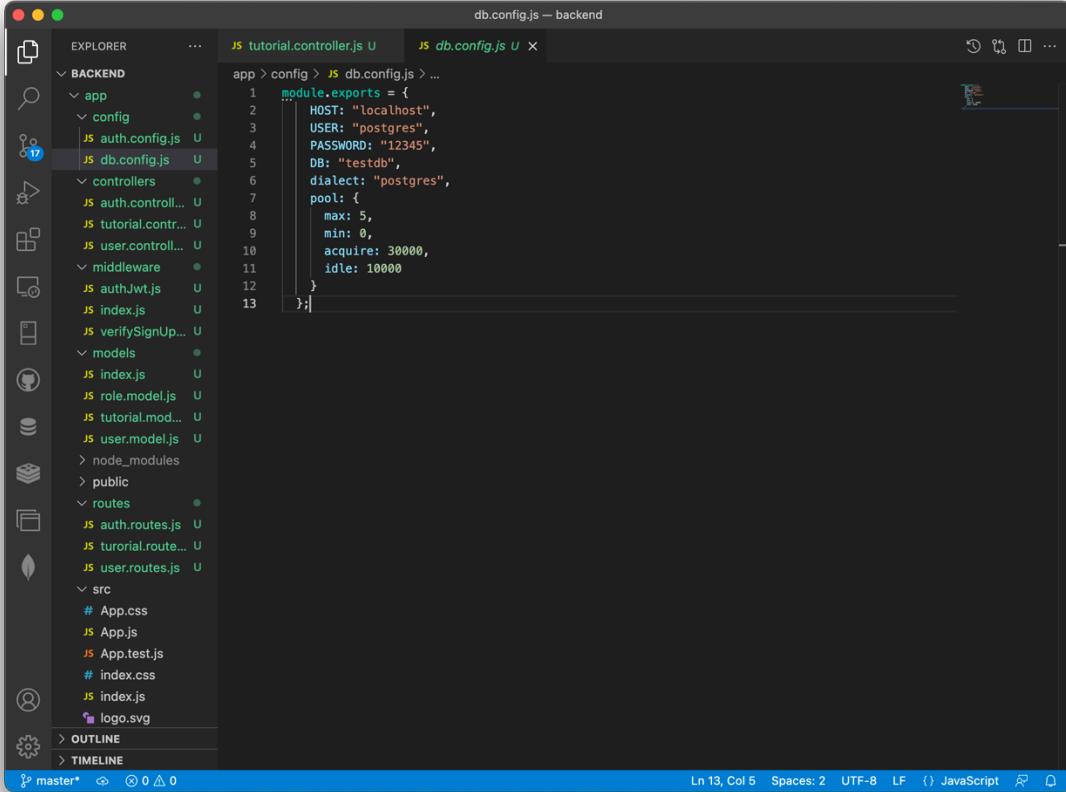
### 3- Abrelos con Visual Studio Code

The screenshot shows two instances of Visual Studio Code side-by-side. The left instance is for the 'FRONTEND' project, which includes files such as 'register.component.js', 'App.js', and 'login.component.js'. The right instance is for the 'BACKEND' project, which includes files like 'tutorial.controller.js'. Both code editors are displaying their respective files with syntax highlighting and line numbers.

### 4- Abre una nueva terminal en cada proyecto e instala las dependencias necesarias con el comando “npm install”

The screenshot shows a single instance of Visual Studio Code with a terminal window at the bottom. The terminal is executing the command 'npm install' in the 'frontend' project directory. The output of the command is visible in the terminal window.

- 5- En el **BACKEND** entra a la ruta “config/db.config.js”, en este archivo encontraras el nombre de la base de datos y usuario, la contraseña cambiala a la que tienes en pgAdmin4.



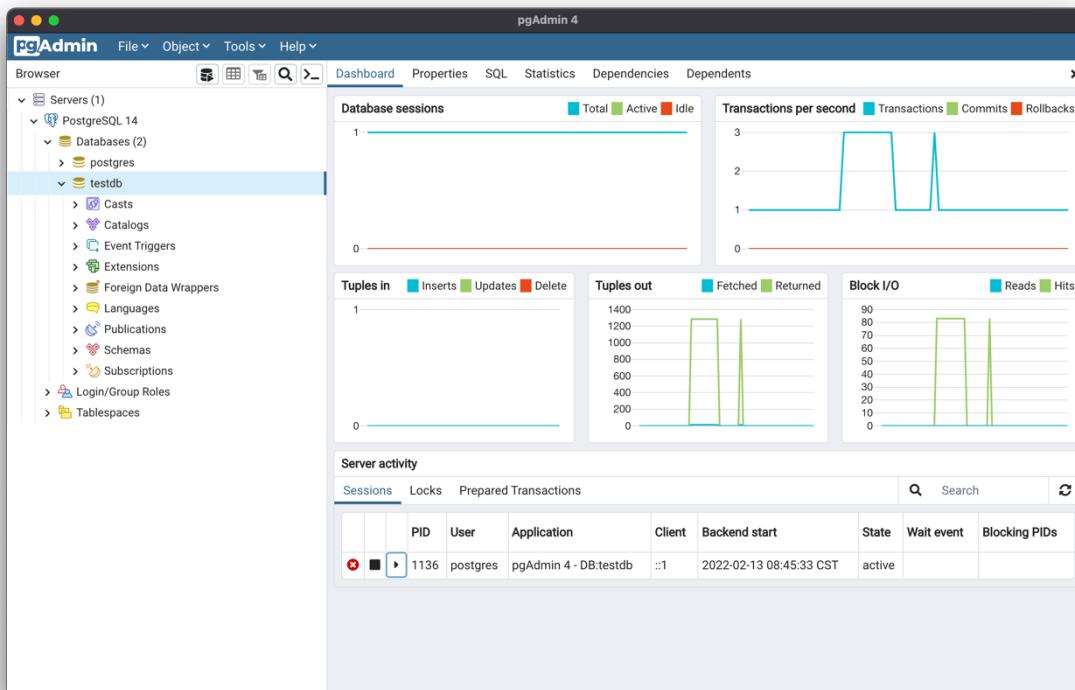
```

db.config.js — backend
JS tutorial.controller.js U JS db.config.js U X
EXPLORER ... JS db.config.js — backend
BACKEND app config auth.config.js U db.config.js U controllers auth.controller.js U tutorial.controller.js U user.controller.js U middleware authJwt.js U index.js verifySignUp.js models index.js role.model.js tutorial.model.js user.model.js node_modules public routes auth.routes.js U tutorial.routes.js U user.routes.js U src App.css App.js App.test.js index.css index.js logo.svg OUTLINE TIMELINE
app > config > JS db.config.js > ...
1 module.exports = {
2   HOST: "localhost",
3   USER: "postgres",
4   PASSWORD: "12345",
5   DB: "testdb",
6   dialect: "postgres",
7   pool: {
8     max: 5,
9     min: 0,
10    acquire: 30000,
11    idle: 1000
12  }
13 };

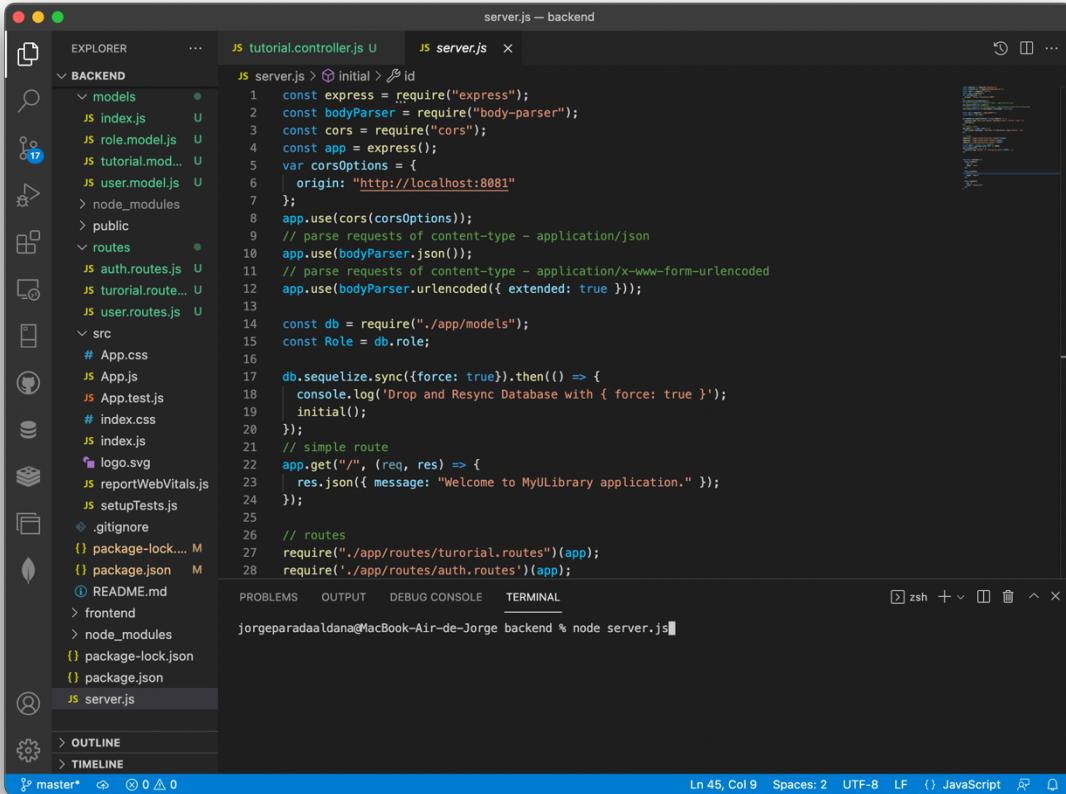
```

Ln 13, Col 5 Spaces: 2 UTF-8 LF {} JavaScript ⌂ ⌂

- 6- Abre pgAdmin4 y crea la base de datos con el nombre que tienes en el archivo db.config.js del backend

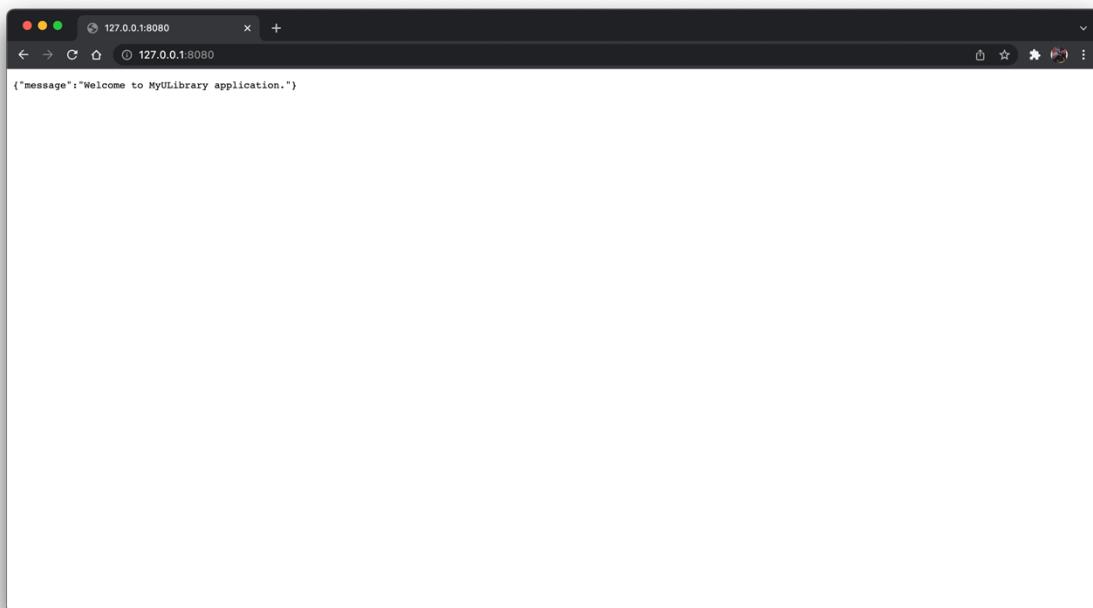


7. Ejecuta el comando “node server.js” para ejecutar el backend en el puerto 8080 de tu navegador.

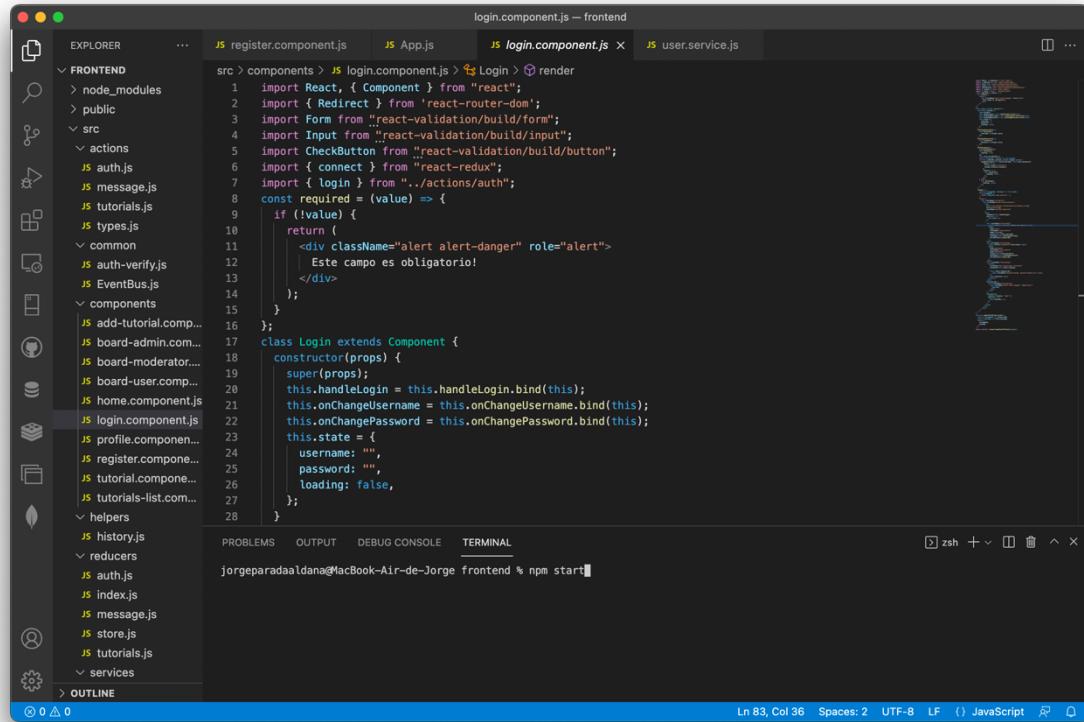


```
const express = require("express");
const bodyParser = require("body-parser");
const cors = require("cors");
const app = express();
var corsOptions = {
  origin: "http://localhost:8081"
};
app.use(cors(corsOptions));
// parse requests of content-type - application/json
app.use(bodyParser.json());
// parse requests of content-type - application/x-www-form-urlencoded
app.use(bodyParser.urlencoded({ extended: true }));
const db = require("./app/models");
const Role = db.role;
db.sequelize.sync({force: true}).then(() => {
  console.log("Drop and Resync Database with { force: true }");
  initial();
});
// simple route
app.get("/", (req, res) => {
  res.json({ message: "Welcome to MyULibrary application." });
});
// routes
require("./app/routes/tutorial.routes")(app);
require("./app/routes/auth.routes")(app);
jorgeparaaldana@MacBook-Air-de-Jorge backend % node server.js
```

8- Ingresa a <http://127.0.0.1:8080> en tu navegador, veras que aparecera esto, si es asi, felicidades, conectaste el backend correctamente.

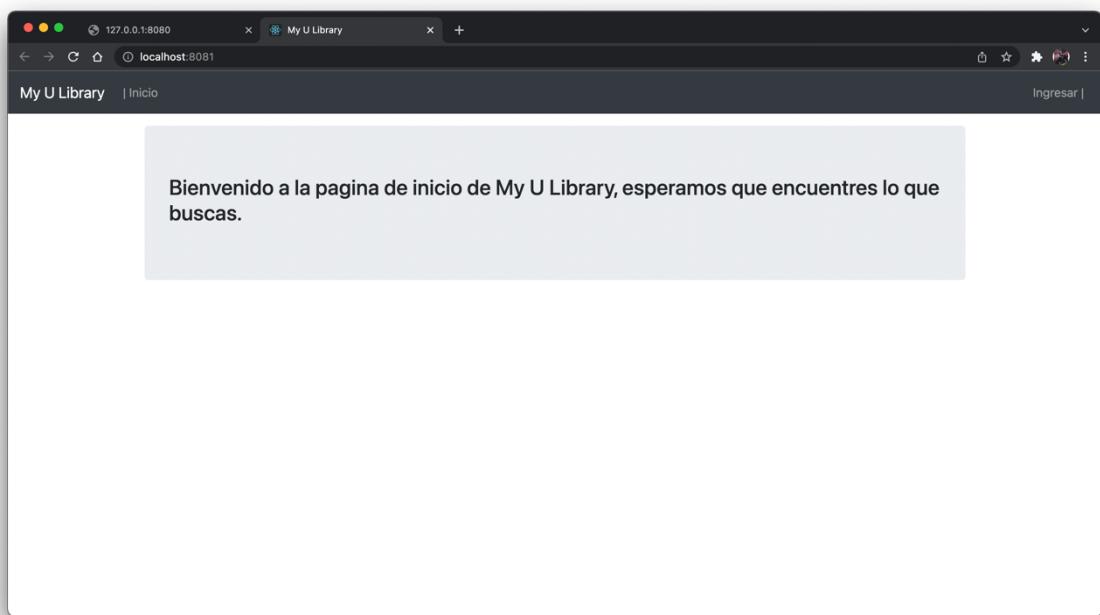


- 9- Ahora en nuestra carpeta **FRONTEND** ejecutamos el comando “npm start” para inicializar el proyecto, se nos abrirá una nueva ventana en nuestro navegador por defecto.



```
login.component.js — frontend
src > components > JS login.component.js > Login > render
1 import React, { Component } from "react";
2 import { Redirect } from 'react-router-dom';
3 import Form from "react-validation/build/form";
4 import Input from "react-validation/build/input";
5 import CheckButton from "react-validation/build/button";
6 import { connect } from "react-redux";
7 import { login } from "../actions/auth";
8 const required = (value) => {
9   if (!value) {
10     return (
11       <div className="alert alert-danger" role="alert">
12         Este campo es obligatorio!
13       </div>
14     );
15   }
16 };
17 class Login extends Component {
18   constructor(props) {
19     super(props);
20     this.handleLogin = this.handleLogin.bind(this);
21     this.onChangeUsername = this.onChangeUsername.bind(this);
22     this.onChangePassword = this.onChangePassword.bind(this);
23     this.state = {
24       username: "",
25       password: "",
26       loading: false,
27     };
28   }
}
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
jorgeparadaaldana@MacBook-Air-de-Jorge frontend % npm start
```

- 10- Listo, visualizas la página de inicio de My U Library únicamente con la opción de “ingresar”.



11- Abrimos POSTMAN para ingresar algunos usuarios con rol de administradores y usuarios normales para poder probar las API's del proyecto (Recuerda configurar el entorno asi como esta en la imagen)

The screenshot shows the Postman application interface. A POST request is being made to `http://localhost:8080/api/auth/signup`. The request body is set to JSON and contains the following data:

```
1 "username": "jorge",
2 "email": "user@jorge.com",
3 "password": "12345678",
4 "roles" : ["user"]
```

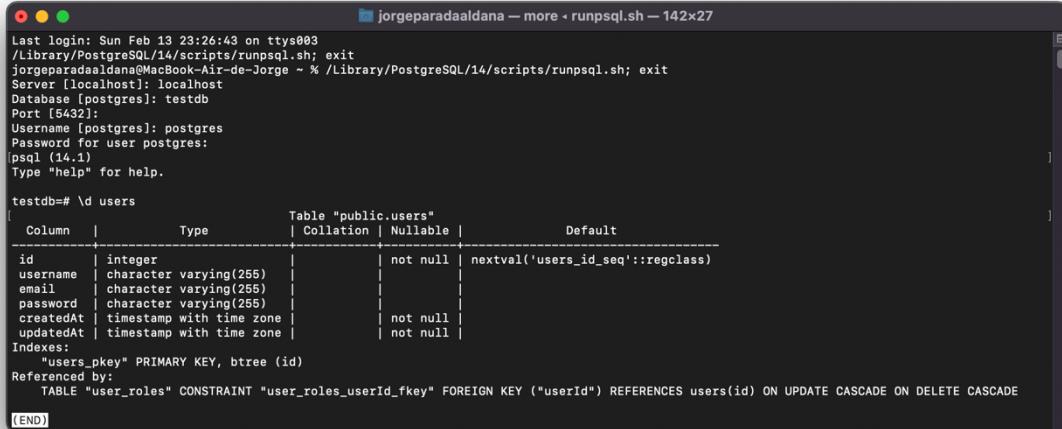
The response status is 200 OK, time: 90 ms, size: 424 B. The message in the response is "Usuario registrado correctamente!".

The screenshot shows the Postman application interface. A POST request is being made to `http://localhost:8080/api/auth/signup`. The request body is set to JSON and contains the following data:

```
1 "username": "jorgeadmin",
2 "email": "admin@jorge.com",
3 "password": "12345678",
4 "roles" : ["admin"]
```

The response status is 200 OK, time: 77 ms, size: 424 B. The message in the response is "Usuario registrado correctamente!".

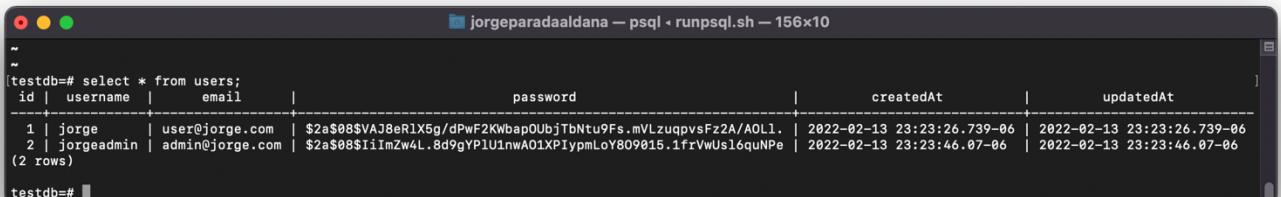
12- Checamos nuestra bd de usuarios y tablas creadas (se crean automaticamente porque fueron definidas en la carpeta *models* del Backend asi que recuerda iniciar sesion con las credenciales en el SQL Shell psql)



```
Last login: Sun Feb 13 23:26:43 on ttys003
/Library/PostgreSQL/14/scripts/runpsql.sh; exit
jorgeparadaalldana@MacBook-Air-de-Jorge ~ % /Library/PostgreSQL/14/scripts/runpsql.sh; exit
Server [localhost]: localhost
Database [postgres]: testdb
Port [5432]:
Username [postgres]: postgres
Password for user postgres:
pgsql (14.1)
Type "help" for help.

testdb=# \d users
Table "public.users"
 Column | Type | Collation | Nullable | Default
-----+-----+-----+-----+-----+
 id   | integer | not null | nextval('users_id_seq)::regclass
 username | character varying(255) |
 email | character varying(255) |
 password | character varying(255) |
 createdAt | timestamp with time zone | not null |
 updatedAt | timestamp with time zone | not null |
Indexes:
 "users_pkey" PRIMARY KEY, btree (id)
Referenced by:
 TABLE "user_roles" CONSTRAINT "user_roles_userId_fkey" FOREIGN KEY ("userId") REFERENCES users(id) ON UPDATE CASCADE ON DELETE CASCADE
(END)
```

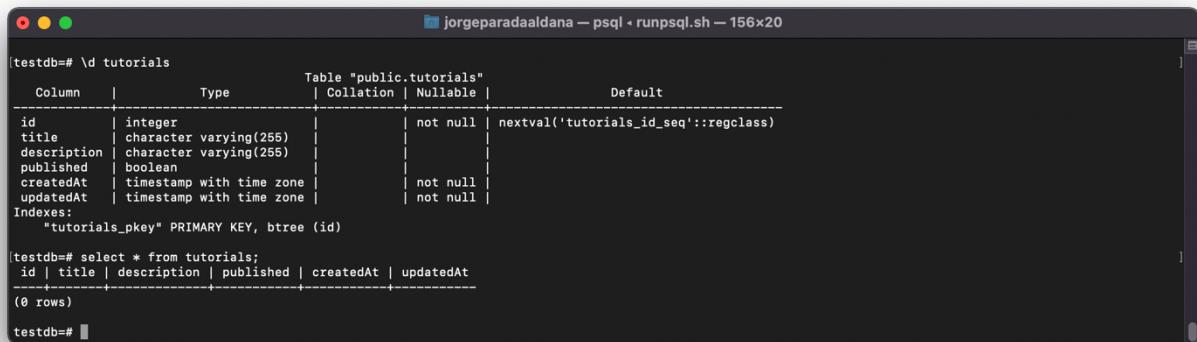
13- Las tablas fueron creadas correctamente y verificamos que los datos de usuarios fueron guardados correctamente



```
[testdb=# select * from users;
 id | username | email | password | createdAt | updatedAt
---+-----+-----+-----+-----+-----+
 1 | jorge | user@jorge.com | $2a$08$VAJ8eRlX5g/dPwf2Kwbp0UbjtBnTu9Fs.mVLzupvzf2A/o0L | 2022-02-13 23:23:26.739-06 | 2022-02-13 23:23:26.739-06
 2 | jorgedadmin | admin@jorge.com | $2a$08$iiimZw4L.8d9gYPlu1nwAO1XPiypmLoY809015.1frVwUs16quNPe | 2022-02-13 23:23:46.07-06 | 2022-02-13 23:23:46.07-06
(2 rows)

testdb=#
```

14- Verificamos que las tablas de los libros tambien fueron creadas correctamente



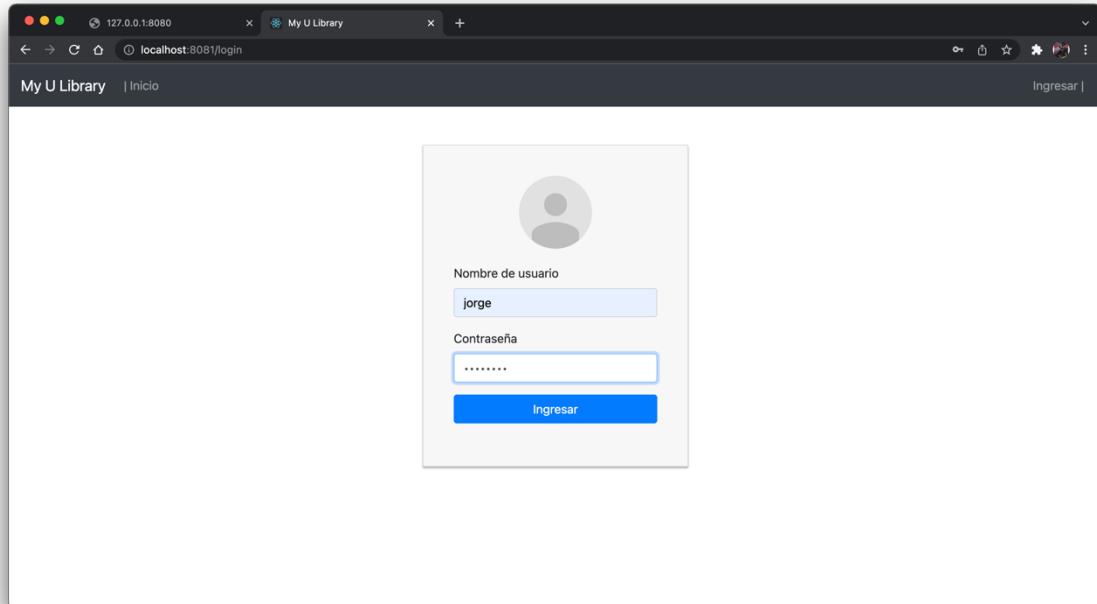
```
[testdb=# \d tutorials
Table "public.tutorials"
 Column | Type | Collation | Nullable | Default
-----+-----+-----+-----+-----+
 id   | integer | not null | nextval('tutorials_id_seq)::regclass
 title | character varying(255) |
 description | character varying(255) |
 published | boolean |
 createdAt | timestamp with time zone | not null |
 updatedAt | timestamp with time zone | not null |
Indexes:
 "tutorials_pkey" PRIMARY KEY, btree (id)

[testdb=# select * from tutorials;
 id | title | description | published | createdAt | updatedAt
---+-----+-----+-----+-----+-----+
(0 rows)

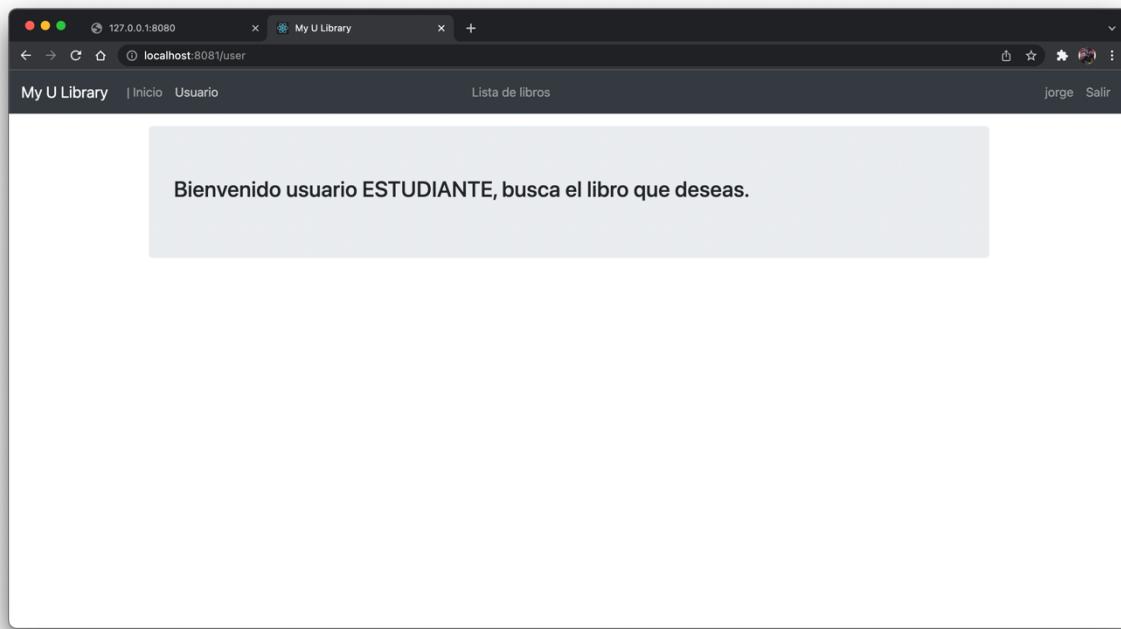
testdb=#
```

## USO

15- Ingresamos al sistema con las credenciales proporcionadas en la opcion de ingresar, primero con la de usuario normal o estudiante.



16- Veremos que tenemos un mensaje de bienvenida y la opcion de lista de libros y demas para buscar los libros que deseamos.



My U Library | Inicio Usuario Lista de libros jorge Salir

Search by title Buscar

**Lista de libros disponibles**

Para saber si el libro esta disponible, por favor da click sobre el...

Eliminar todos

My U Library | Inicio Usuario Lista de libros jorge Salir

**jorge Perfil**

**Token:** eyJhbGciOiJIUzI1Nils ... 0EgoCHEqB7KtazCUuYYc

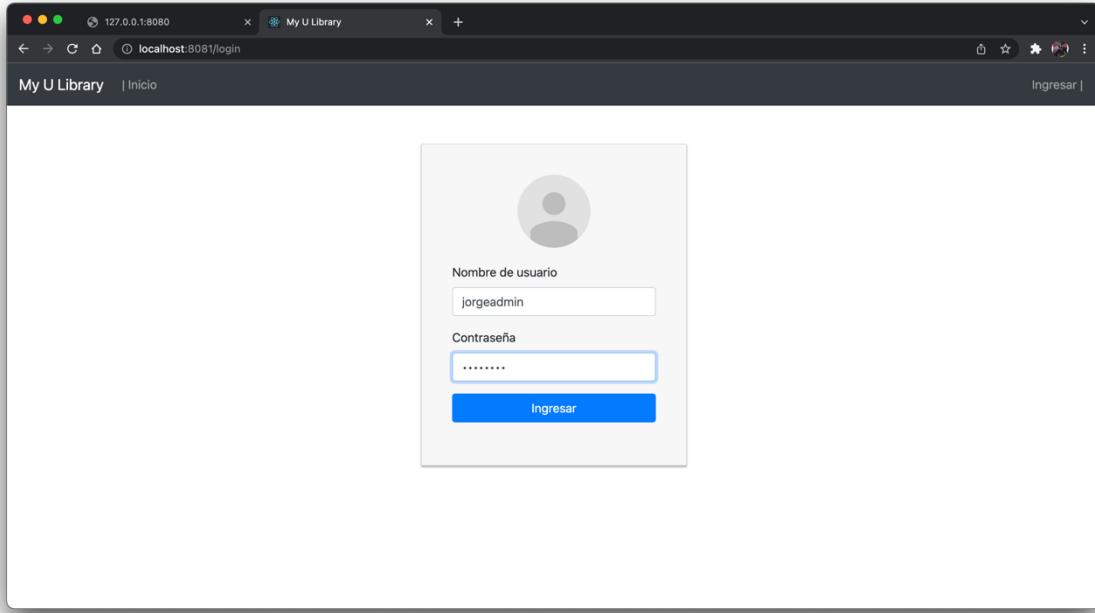
**Id:** 1

**Email:** user@jorge.com

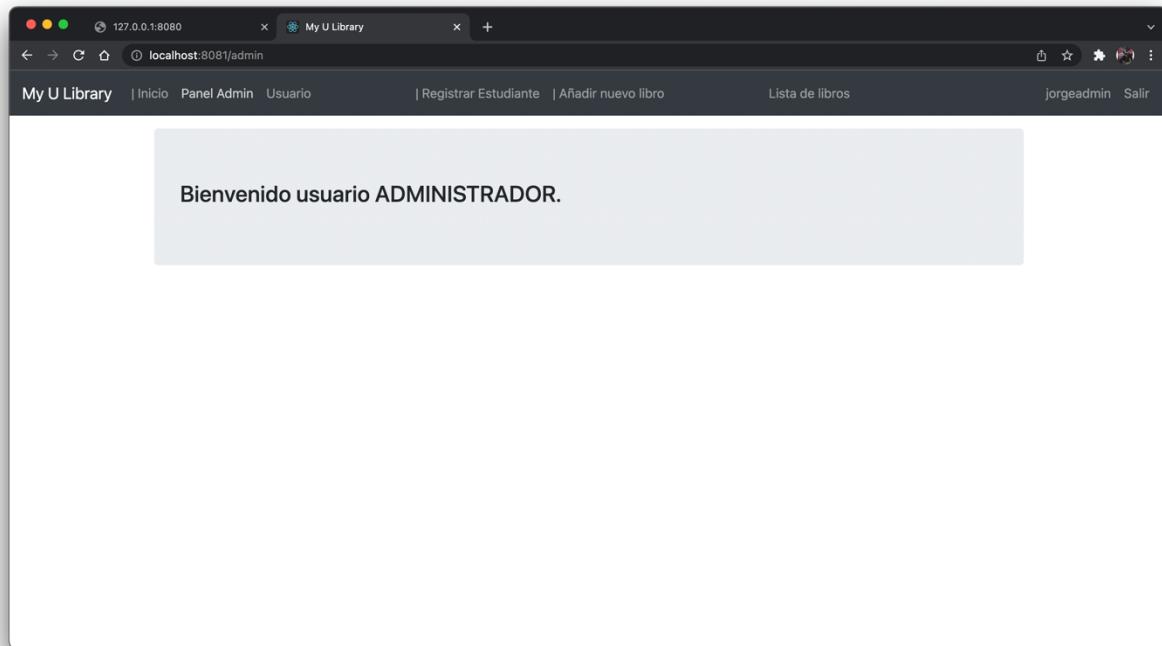
**Rol:**

- ROLE\_USER

17- Ahora ingresamos al rol admin con las credenciales creadas en postman.



18- Veremos que el administrador o bibliotecario tiene las opciones de “añadir libros, ver la lista de ellos, editar, actualizar, eliminar, publicar y crear cuenta de usuarios normales o estudiantes”



My U Library | Inicio | Panel Admin | Usuario | Registrar Estudiante | Añadir nuevo libro | Lista de libros | jorgeadmin | Salir

Nombre de estudiante

Email

Contraseña

**Crear usuario**

My U Library | Inicio | Panel Admin | Usuario | Registrar Estudiante | Añadir nuevo libro | Lista de libros | jorgeadmin | Salir

Título de Libro

Descripción

**Añadir**

My U Library | Inicio | Panel Admin | Usuario | Registrar Estudiante | Añadir nuevo libro | Lista de libros | jorgeadmin | Salir

### jorgeadmin Perfil

Token: eyJhbGciOiJIUzI1Ni...LjFYdv01CikUNXxusx5k

Id: 2

Email: admin@jorge.com

Rol:

- ROLE\_ADMIN

## 19- Agregamos datos.

My U Library | Inicio | Panel Admin | Usuario | Registrar Estudiante | Añadir nuevo libro | Lista de libros | jorgeadmin | Salir

Título de Libro  
Álgebra de Baldor

Descripción  
Álgebra es un libro del matemático y profesor cubano Aurelio Baldor.

Añadir

20. Verificamos si aparece en la lista de libros y damos clic sobre su nombre,

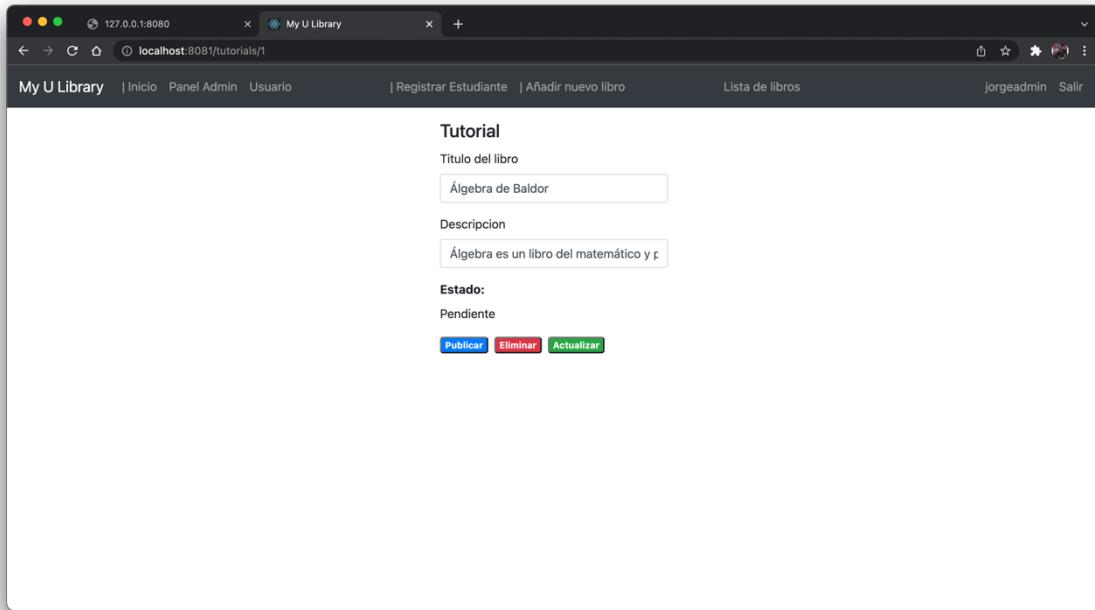
My U Library | Inicio | Panel Admin | Usuario | Registrar Estudiante | Añadir nuevo libro | Lista de libros | jorgeadmin | Salir

Search by title Buscar

Lista de libros disponibles

Libro
<b>Título del libro:</b> Álgebra de Baldor <b>Descripción:</b> Álgebra es un libro del matemático y profesor cubano Aurelio Baldor. La primera edición se produjo el 19 de junio de 1941. El Álgebra de Baldor contiene un total de 5790 ejercicios, que equivalen a 19 ejercicios en cada prueba en promedio. <b>Estado:</b> Pendiente <a href="#">Editar</a>

21. Editamos el libro y veremos estas opciones.



22. Verificamos en nuestra bd si todos los datos se almacenan. Y en efecto si lo hacen.

```
jorgeparadaaldana — psql < runpsql.sh — 172x15

[ testdb=# select * from users;
 id | username | email | password | createdAt | updatedAt
---+-----+-----+-----+-----+-----+-----+
 1 | jorge | user@jorge.com | $2a$08$VAJ8eRlX5g/dPwf2Kwbp0UbjTbNtu9Fs.mVlzuqpvsFz2A/A0L1. | 2022-02-13 23:23:26.739-06 | 2022-02-13 23:23:26.739-06
 2 | jorgeadmin | admin@jorge.com | $2a$08$iiImZw4L.8d9gYPlU1nwA01XP1ympmLoY809015.1frVwUs16quNPe | 2022-02-13 23:23:46.07-06 | 2022-02-13 23:23:46.07-06
(2 rows)

[ testdb=# select * from tutorials;
 id | title | description | published | createdAt | updatedAt
---+-----+-----+-----+-----+-----+
 1 | Álgebra de Baldor | Álgebra es un libro del matemático y profesor cubano Aurelio Baldor. | f | 2022-02-14 00:12:40.664-06 | 2022-02-14 00:24:03.953-06
(1 row)

testdb=# ]
```

NOTA:

El sistema carece hasta el momento de las opciones:

- . Prestar libro a estudiante
  - . Regresar libro
  - . Conteo de cuantos libros hay en stock de ese mismo
  - . Reajuste en el menu de admin.
- (Es mi segundo avance de la prueba.)