



---

# MEMORIA ONLINE CHAT

---

Proyecto Final



30 DE MAYO DE 2022

JORGE VALENCIA ABAD  
IES RIBERA DE CASTILLA

## ÍNDICE

1.	Introducción .....	2
1.1.	Nombre del proyecto:.....	2
1.2.	Qué es:.....	2
2.	Tecnologías Realizadas .....	2
2.1.	FRONT:.....	2
2.2.	BACK:.....	4
2.3.	BASE DE DATOS:.....	6
2.4.	CONTROL DE VERSIONES:.....	6
3.	Explicación de Aplicación: .....	7
3.1.	Funcionalidad sin iniciar sesión en la página:.....	7
3.2.	Funcionalidad al iniciar sesión en la página:.....	8
4.	Estructuración de directorios:.....	11
5.	Conclusiones .....	12
6.	Bibliografía .....	12

## 1. Introducción

### 1.1. Nombre del proyecto:

**Online Chat**

### 1.2. Qué es:

Este proyecto consiste en la realización de un sitio web de chatting, donde cualquier persona puede crear una sala de chat, que puede ser tanto privada como pública, o bien unirse a una sala, ya creada anteriormente por otro usuario, para charlar y conocer gente.

Para poder crear o unirse a estas salas, el sitio web proporcionará la opción de registrarse, ya que no hay otra manera de disfrutar de Online Chat.

## 2. Tecnologías Realizadas

### 2.1. FRONT:

Para la parte visual de la página he utilizado un framework denominado **React**.

**React** es un framework de Javascript de código abierto diseñada para crear interfaces de usuario con el objetivo de facilitar el desarrollo de aplicaciones en una sola página. Es mantenido por Facebook y la comunidad de software libre.

**React** intenta ayudar a los desarrolladores a construir aplicaciones que utilizan datos que cambian continuamente. Su objetivo es ser sencillo y fácil de combinar.

*Middleware:*

Aparte de **React** he utilizado otros paquetes, como son:

**Socket.io Client:** Paquete que nos permite conectarnos con el paquete de back socket.io, para las llamadas bidireccionales del chat.

```
1 const [socket, setSocket] = useState(io());
2 socket.emit("join_room", nameRoom);
```

```
1
2 await socket.emit("send_message", messageData);
3 setMessageList((list) => [...list, messageData]);
4 setCurrentMessage("");
```

**Use Form:** Paquete que puede administrar y validar un formulario de react.

```
1 return () => {
2   //render />
3   //render />
4   //Form onSubmit={handleSubmit(onSubmit)}
5   //Form onSubmit={handleSubmit(onSubmit)}
6   //Form onSubmit={handleSubmit(onSubmit)}
7   //Form onSubmit={handleSubmit(onSubmit)}
8   //Form onSubmit={handleSubmit(onSubmit)}
9   //Form onSubmit={handleSubmit(onSubmit)}
10  //Form onSubmit={handleSubmit(onSubmit)}
11  //Form onSubmit={handleSubmit(onSubmit)}
12  //Form onSubmit={handleSubmit(onSubmit)}
13  //Form onSubmit={handleSubmit(onSubmit)}
14  //Form onSubmit={handleSubmit(onSubmit)}
15  //Form onSubmit={handleSubmit(onSubmit)}
16  //Form onSubmit={handleSubmit(onSubmit)}
17  //Form onSubmit={handleSubmit(onSubmit)}
18  //Form onSubmit={handleSubmit(onSubmit)}
19  //Form onSubmit={handleSubmit(onSubmit)}
20  //Form onSubmit={handleSubmit(onSubmit)}
21  //Form onSubmit={handleSubmit(onSubmit)}
22  //Form onSubmit={handleSubmit(onSubmit)}
23  //Form onSubmit={handleSubmit(onSubmit)}
24  //Form onSubmit={handleSubmit(onSubmit)}
25  //Form onSubmit={handleSubmit(onSubmit)}
26  //Form onSubmit={handleSubmit(onSubmit)}
27  //Form onSubmit={handleSubmit(onSubmit)}
28  //Form onSubmit={handleSubmit(onSubmit)}
29  //Form onSubmit={handleSubmit(onSubmit)}
30  //Form onSubmit={handleSubmit(onSubmit)}
31  //Form onSubmit={handleSubmit(onSubmit)}
32  //Form onSubmit={handleSubmit(onSubmit)}
33  //Form onSubmit={handleSubmit(onSubmit)}
34  //Form onSubmit={handleSubmit(onSubmit)}
35  //Form onSubmit={handleSubmit(onSubmit)}
36  //Form onSubmit={handleSubmit(onSubmit)}
37  //Form onSubmit={handleSubmit(onSubmit)}
38  //Form onSubmit={handleSubmit(onSubmit)}
39  //Form onSubmit={handleSubmit(onSubmit)}
40  //Form onSubmit={handleSubmit(onSubmit)}
41  //Form onSubmit={handleSubmit(onSubmit)}
42  //Form onSubmit={handleSubmit(onSubmit)}
43  //Form onSubmit={handleSubmit(onSubmit)}
44  //Form onSubmit={handleSubmit(onSubmit)}
45  //Form onSubmit={handleSubmit(onSubmit)}
46  //Form onSubmit={handleSubmit(onSubmit)}
47  //Form onSubmit={handleSubmit(onSubmit)}
48  //Form onSubmit={handleSubmit(onSubmit)}
49  //Form onSubmit={handleSubmit(onSubmit)}
50  //Form onSubmit={handleSubmit(onSubmit)}
51  //Form onSubmit={handleSubmit(onSubmit)}
52  //Form onSubmit={handleSubmit(onSubmit)}
53  //Form onSubmit={handleSubmit(onSubmit)}
54  //Form onSubmit={handleSubmit(onSubmit)}
55  //Form onSubmit={handleSubmit(onSubmit)}
56  //Form onSubmit={handleSubmit(onSubmit)}
57  //Form onSubmit={handleSubmit(onSubmit)}
58  //Form onSubmit={handleSubmit(onSubmit)}
59  //Form onSubmit={handleSubmit(onSubmit)}
60  //Form onSubmit={handleSubmit(onSubmit)}
61  //Form onSubmit={handleSubmit(onSubmit)}
62  //Form onSubmit={handleSubmit(onSubmit)}
63  //Form onSubmit={handleSubmit(onSubmit)}
64  //Form onSubmit={handleSubmit(onSubmit)}
65  //Form onSubmit={handleSubmit(onSubmit)}
66  //Form onSubmit={handleSubmit(onSubmit)}
67  //Form onSubmit={handleSubmit(onSubmit)}
68  //Form onSubmit={handleSubmit(onSubmit)}
69  //Form onSubmit={handleSubmit(onSubmit)}
70  //Form onSubmit={handleSubmit(onSubmit)}
71  //Form onSubmit={handleSubmit(onSubmit)}
72  //Form onSubmit={handleSubmit(onSubmit)}
73  //Form onSubmit={handleSubmit(onSubmit)}
74  //Form onSubmit={handleSubmit(onSubmit)}
75  //Form onSubmit={handleSubmit(onSubmit)}
76  //Form onSubmit={handleSubmit(onSubmit)}
77  //Form onSubmit={handleSubmit(onSubmit)}
78  //Form onSubmit={handleSubmit(onSubmit)}
79  //Form onSubmit={handleSubmit(onSubmit)}
80  //Form onSubmit={handleSubmit(onSubmit)}
81  //Form onSubmit={handleSubmit(onSubmit)}
82  //Form onSubmit={handleSubmit(onSubmit)}
83  //Form onSubmit={handleSubmit(onSubmit)}
84  //Form onSubmit={handleSubmit(onSubmit)}
85  //Form onSubmit={handleSubmit(onSubmit)}
86  //Form onSubmit={handleSubmit(onSubmit)}
87  //Form onSubmit={handleSubmit(onSubmit)}
88  //Form onSubmit={handleSubmit(onSubmit)}
89  //Form onSubmit={handleSubmit(onSubmit)}
90  //Form onSubmit={handleSubmit(onSubmit)}
91  //Form onSubmit={handleSubmit(onSubmit)}
92  //Form onSubmit={handleSubmit(onSubmit)}
93  //Form onSubmit={handleSubmit(onSubmit)}
94  //Form onSubmit={handleSubmit(onSubmit)}
95  //Form onSubmit={handleSubmit(onSubmit)}
96  //Form onSubmit={handleSubmit(onSubmit)}
97  //Form onSubmit={handleSubmit(onSubmit)}
98  //Form onSubmit={handleSubmit(onSubmit)}
99  //Form onSubmit={handleSubmit(onSubmit)}
100 //Form onSubmit={handleSubmit(onSubmit)}
```

**React Router:** Paquete que se utiliza para marcar las rutas de las páginas del sitio web.



```
1 function App() {  
2   return (<>  
3  
4     <UserContext:  
5       <Router>  
6         <Routes>  
7           <Route path="/" element={<Home />} />  
8           <Route path="/register" element={<Register />} />  
9           <Route path="/login" element={<login />} />  
10          <Route path="/createroom" element={<CreateRoom />} />  
11          <Route path="/joinroom" element={<JoinRoom />} />  
12          <Route path="/profile" element={<Profile />} />  
13        </Routes>  
14      </Router>  
15    </UserContext>  
16  </>)  
17 }  
18 }
```

## 2.2. BACK:

Para la parte no visible de la página he utilizado un framework denominado **Node.js** que es un entorno en tiempo de ejecución multiplataforma. Es de código abierto y está basado en el lenguaje de programación JavaScript, asíncrono.

Fue creado con el enfoque de ser útil en la creación de programas de red altamente escalables

Al contrario que la mayoría del código JavaScript, en este caso, no se ejecuta en un navegador, sino en el servidor.

También he utilizado el entorno de trabajo **Express** para desarrollar aplicaciones la APIs de la página donde regajo los datos de la base de datos.

*Middleware:*

Aparte de **Node.js** he utilizado otros paquetes, como son:

**Socket.io:** Paquete que proporciona un canal bidireccional entre la parte servidor y parte cliente (socket.io client) donde se establece una conexión WebSocket. Se ha utilizado para la creación de salas y la a la hora de enviar y recibir los mensajes introducidos por los usuarios.

```
1 const io = new Server(server);
2 io.on("connection", socket => {
3   console.log('User Connected: ${socket.id}');
4   socket.on("message", (message) => {
5     console.log('Message: ' + message)
6   });
7   socket.on("join_room", (data) => {
8     socket.join(data);
9     console.log('User with ID: ${socket.id} joined room: ${data}');
10  });
11  socket.on("send_message", (data) => {
12    socket.to(data.room).emit("receive_message", data);
13  });
14  socket.on("disconnect", () => {
15    console.log("User Disconnected", socket.id);
16  });
17 });
```

**JSON Web Token:** es un estándar abierto basado en JSON propuesto por IETF para la creación de tokens. Esto se ha utilizado para guardar la información del usuario y poder acceder a ella cuando quiera.

```
1 const newToken = jwt.sign(user, process.env.SECRET);
```

**PG:** node-postgres es una colección de módulos node.js para interactuar con su base de datos PostgreSQL.

```
1 import pg from "pg";
2 const Pool = pg.Pool;
3
4 const pool = new Pool({
5   user: "postgres",
6   password: "4Ud3f3kD",
7   host: "localhost",
8   port: 5432,
9   database: "onlinechat"
10 });
11 export default pool;
```

### 2.3. BASE DE DATOS:

El sistema de gestión de bases de datos que se ha utilizado en el proyecto es **PostgreSQL**.

**PostgreSQL**, sistema de base de datos relacional de código abierto publicado bajo la licencia **PostgreSQL**.

He utilizado **PostgreSQL**, ya que me ha sido fácil crear una API con ella. Aunque en mi aplicación no se pueda guardar los mensajes, dado que es un chat instantáneo.



### 2.4. CONTROL DE VERSIONES:

El control de versiones que he utilizado ha sido GIT. GIT es un sistema de control de versiones distribuido, creado por Linus Tolvars. Este control de versiones me ha salvado la vida muchas veces, ya que en un momento en el desarrollo hubo un fallo en la aplicación por cuestiones de “playfills” y a través de GIT puedo volver al “commit” anterior y solucionar el problema.



### 3. Explicación de Aplicación:

En este apartado se explicará el funcionamiento de la página y todo lo que se puede hacer en ella.

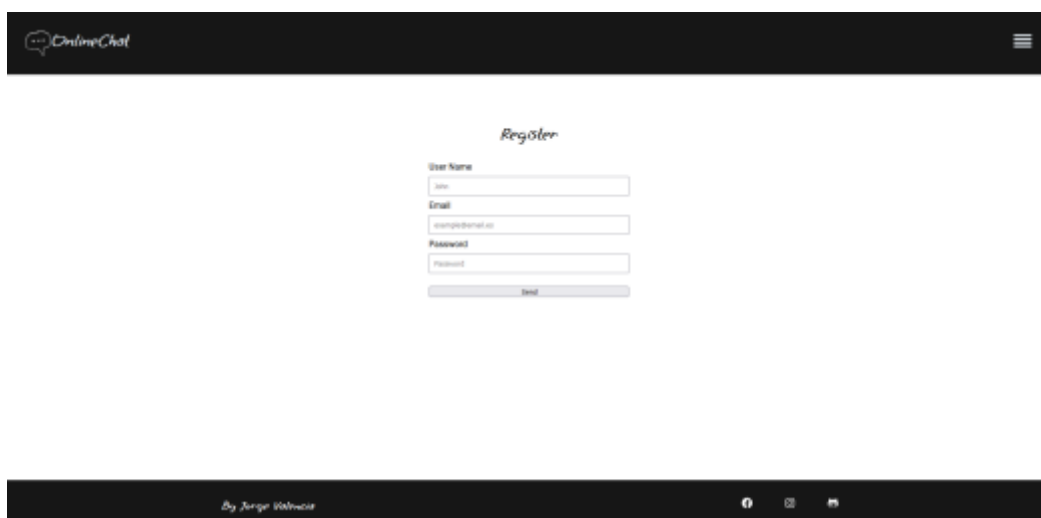
#### 3.1. Funcionalidad sin iniciar sesión en la página:

Lo que se puede hacer es acceder desde el menú hamburguesa a tres partes de la página:

- Página de Home:

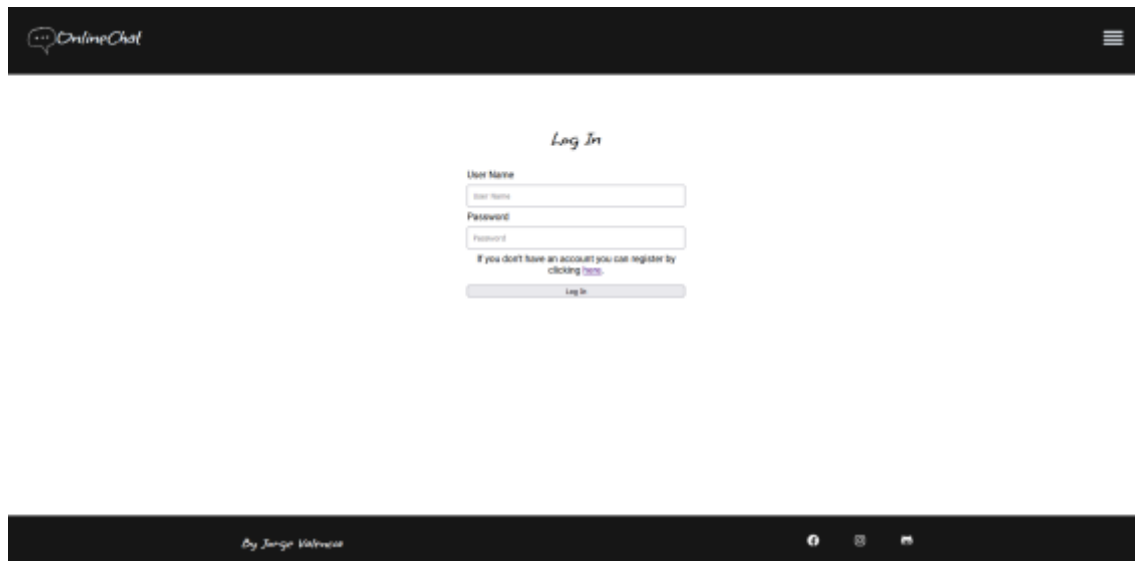


- Página de Log In:  
Apartado donde puedes iniciar sesión en la página.





- Página de Register:  
Apartado donde puedes registrarte como usuario en la página.



The screenshot shows a web browser window with a dark header and footer. The header contains the 'OnlineChat' logo and a hamburger menu icon. The main content area is white and features a 'Log In' section. This section includes a title 'Log In', two input fields labeled 'User Name' and 'Password', and a 'Log In' button. Below the password field, there is a link that says 'If you don't have an account you can register by clicking [here](#).' The footer contains the text 'By Jorge Valencia' and three social media icons.

### 3.2. Funcionalidad al iniciar sesión en la página:

Lo que se puede hacer es:

- Crear salas de chat:  
En este sitio web puedes crear dos tipos de salas:
  - Salas privadas: Que están compuestas por el nombre de la sala y la contraseña.
  - Salas Públicas: Que solo están compuestas por el nombre de la sala.

Para crear estas salas la aplicación proporciona el siguiente formulario.



The screenshot shows a form titled 'Create A Chat'. It has two main sections: 'Room Types' and 'Room Name'. The 'Room Types' section has a dropdown menu with 'Public' selected. The 'Room Name' section has a text input field with the placeholder text 'My Room...'. Below these sections is a 'Join A Room' button.

- Unirse a una sala de chat:  
Puedes entrar a las salas Introduciendo los datos necesarios

*Join A Chat*

Room Types

Public

Name Room

My Room....

Join A Room

Tras hacer este paso podrás chatear con gente...

**Tecno**

Hola que tal todos?  
16:25 Laura

Hola  
16:26 JgamesV

Bien la verdad  
16:27 JgamesV

Aqui de que se habla ?  
16:28 Laura

Message

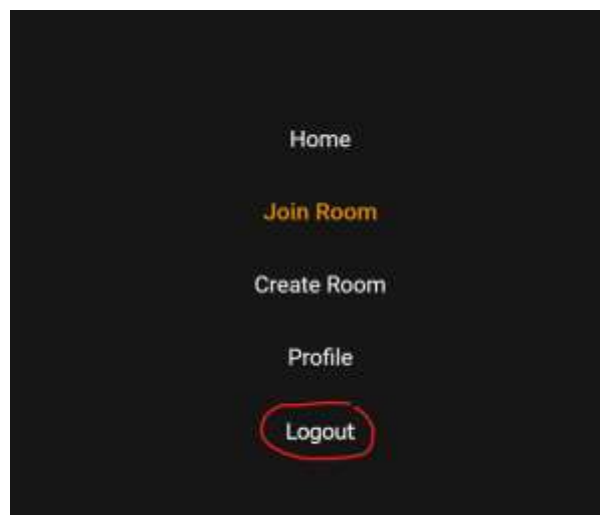
➤ ➡

- Modificar los datos del usuario:  
El usuario puede modificar sus datos de manera fácil a través del siguiente formulario.



The screenshot shows a web application window titled "OnlineChat". The main content area displays a form titled "Modify User". The form contains four input fields: "User Name" (with the value "John"), "Email" (with the value "example@email.es"), "Password", and "Password" (repeated). Below the fields is a "Send" button. The footer of the application window shows "By Jorge Valencia" and some social media icons.

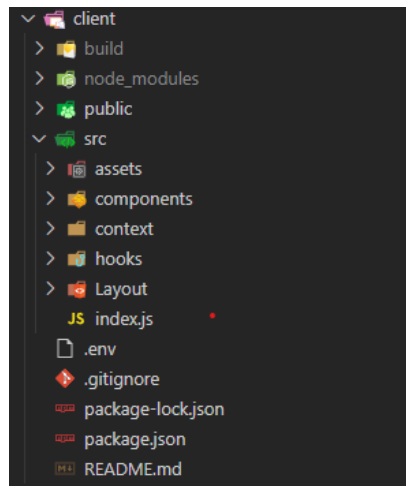
- Cerrar sesión:  
Para cerrar sesión habrá que ir al menú y dar a la opción de "Log Out".



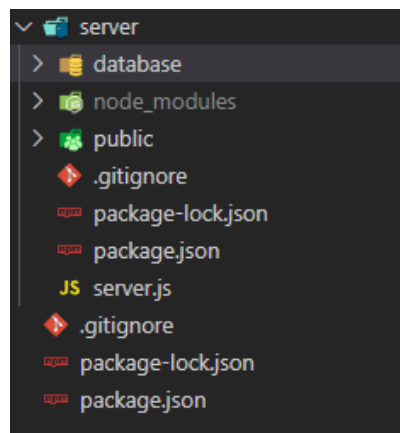
## 4. Estructuración de directorios:

La estructura de directorios está dividida en 2 carpetas:

- Carpeta cliente donde se guarda la parte de cliente de react.



- Carpeta servidor donde se guarda el servidor de node.



## 5. Conclusiones

Gracias a este proyecto he aprendido diferentes herramientas muy útiles para crear aplicaciones web y además me ha ayudado a la hora de buscarme la vida para solucionar los errores que me han ido surgiendo en la fase de desarrollo.

También he podido comprender que realmente es difícil programar una aplicación con unas tecnologías que al principio no tenía un gran conocimiento de ellas y a base de errores he ido aprendiendo, como la vida misma. Podría decir que si hubiera tenido menos errores no habría aprendido sobre las tecnologías.

Aparte de este proyecto, mi compañero y yo hemos tenido que hacer un proyecto como este pero con otras tecnologías y se podía decir que he aprendido el doble que muchos de mis compañeros, aunque a lo mejor no he podido añadir tanta funcionalidad como me hubiera gustado en los dos proyectos, estoy contento con mi empeño.

## 6. Bibliografía

Node documentación:

<https://nodejs.org/es/docs/>

React documentación:

<https://es.reactjs.org/docs/getting-started.html>

PostgreSQL documentación:

<https://www.postgresql.org/docs>

Socket.io documentación:

<https://socket.io/docs/v4/>

JSON Web Token documentación:

<https://jwt.io/introduction>