

Documentación de la API

API

Creador : Mateo Roca

Introducción

Esta API permite administrar la información de los usuarios y controlar su acceso. Además, facilita un sistema de intercambio de mensajes entre usuarios registrados, con la opción de aceptar o rechazar propuestas de chat antes de iniciar la conversación.

URL Base :

`http://localhost:3000/`

Autenticación

De momento la api no cuenta con un key para su uso.

Endpoints

A continuación, se enumeran los endpoints disponibles en la API junto con una descripción de cada uno, los parámetros requeridos y opcionales, y los métodos HTTP admitidos (GET, POST, etc.).



endPoint	método	Descripción	Parámetros de consulta	Respuesta exitosa	Respuestas de error
/sessionHandler/login	POST	este endpoint válida si el userName y el password son correctos y de ser así retorna el userId y un token de sesión	userID y Password	{ status: true, userId: userId, Token: response.token, message: "User logged in", }	{ status: false, message: response.message }
/sessionHandler/signup	POST	este endpoint crea un nuevo usuario en el sistema con los datos recibidos	username , password, name , surname, email.	{ state: true, message: "Registered user successfully", }	{ state: false, message: "error empty data" }
/sessionHandler/logout	GET	Este endpoint elimina el token de sesión del caché del servidor y el id asociado.	sesión Token y userID	{ status: true, message: "session close" }	{ status: false, message: "error user id no valido" }
/sessionHandler/getActiveUsers	GET	este usuario retorna un array de los userId de los usuarios activos	nada	{ status: true, message: "success to get	{ status: false, message: "error to get Active Users" }

				Active Users", data: activeUsersIds, }	
/groupHandler/getgroupsdata	GET	este endpoint muestra todos los nombres de los grupos presentes en el sistema	nada	{ groups: groups }	{ message: "Internal Server Error" }
/groupHandler/addusertogroup	POST	este endpoint incluye a determinado usuario a un grupo seleccionado	userName y groupName	{ status: state, message: `success to add \${userName} to group : \${userGroup}`, }	{ state: false, message: "error empty data" }
/userHandler/getusers	GET	este endpoint retorna un array de userName presentes en el sistema	nada	{ status: true, message: "success to get users", data: userData, }	{ status: false, message: "Error interno del servidor" }
/chatmessagehandler/newchatmessage	POST	este endpoint crea un nuevo mensaje en un chat determinado	chatId , userOriginId , userTargetId , messageBody, timeStampSended	{ status: true, message: `success to send "\${MessageBody}" to userId : \${userTargetId}`, }	{ status: false, message: `Error to save message`, }
/chatmessagehandler/getchatmessages	POST	este endpoint retorna los mensajes de determinado chat	chatID	{ status: true, arrayOfMessages: messages }	{ status: false, message: "Internal Server Error" }
/chatprop	POST	este endpoint retorna	userId	{ state: true, data:	{ state: false,

osalhandler/getchatproposal		las chat proposals hechas a determinado user		proposals }	message: "No chat proposals found" }
/chatHandler/getchats	POST	este endpoint retorna para determinado userId los chat que estan disponibles	UserOrigin ID	{ status: true, message: "success to get chats", data: chatsFiltered, } }	{ status: false, message: "error to get chats" }
/chatproposalhandler/newchatproposal	POST	este endpoint crea una nueva propuesta de chat en base al user que propone y al que va dirigida la propuesta	userOriginId , userTargetID	{ state: true, message: "success to create chat Proposal" }	{ status: false, message: "error to create chat Proposal" }
/chatproposalhandler/confirmchatproposal	POST	este endpoint cambia de estado ha confirmado a determinada propuesta de chat	chatProposal ID	{ state: true, message: "Success to confirm Chat Proposal", } }	{ state: false, message: "Error: Chat Proposal not found", } }
/chatproposalhandler/rejectchatproposal	POST	este endpoint cambia de estado la propuesta de chat a denegada	Chat Proposal ID	{ state: true, message: "Success to reject Chat Proposal", } }	{ state: false, message: "Error: Chat Proposal not found", } }

Ejemplos de uso

se registra uno como usuario :

Home Sign Up Log In

Sign Up

User Name

Email

Name

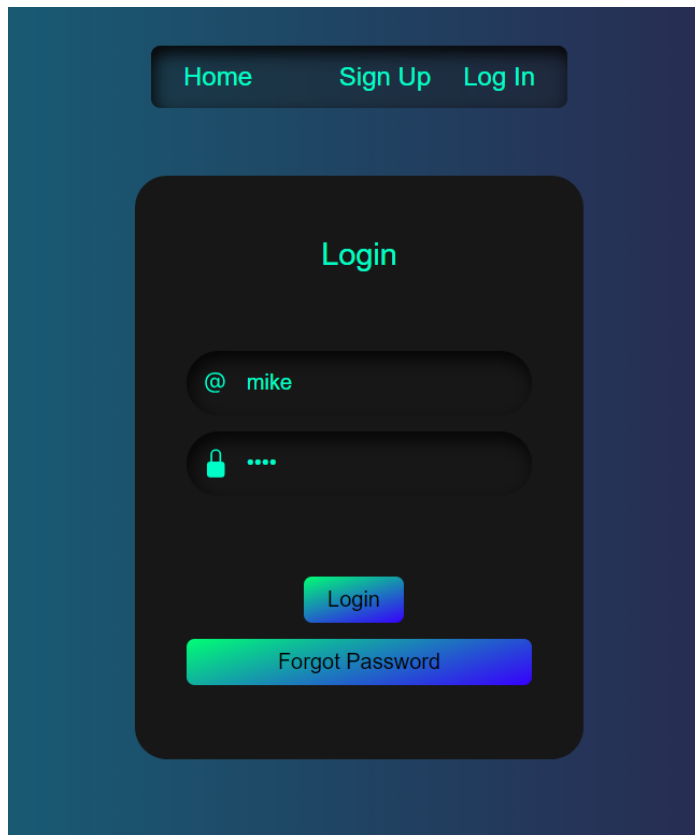
Surname

Password

Check Password

SignUp

se loguea en el sistema :



The image shows a login form with a dark blue gradient background. At the top, there is a navigation bar with three links: "Home", "Sign Up", and "Log In". Below this, the word "Login" is displayed in a light blue font. The form consists of two input fields: the first is for the email, labeled with an "@" icon and the text "mike"; the second is for the password, labeled with a lock icon and four dots. Below the input fields, there is a "Login" button and a "Forgot Password" link, both with a blue-to-purple gradient.

Home Sign Up Log In

Login

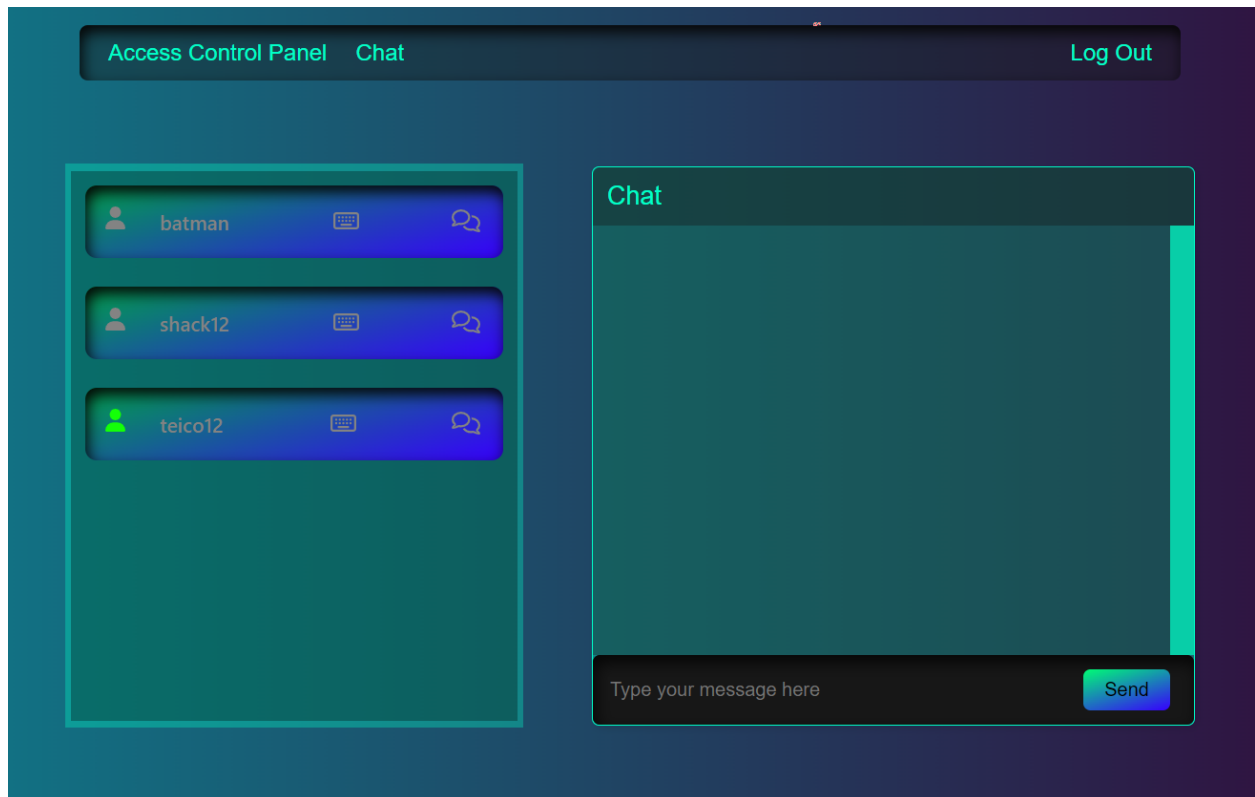
@ mike

lock

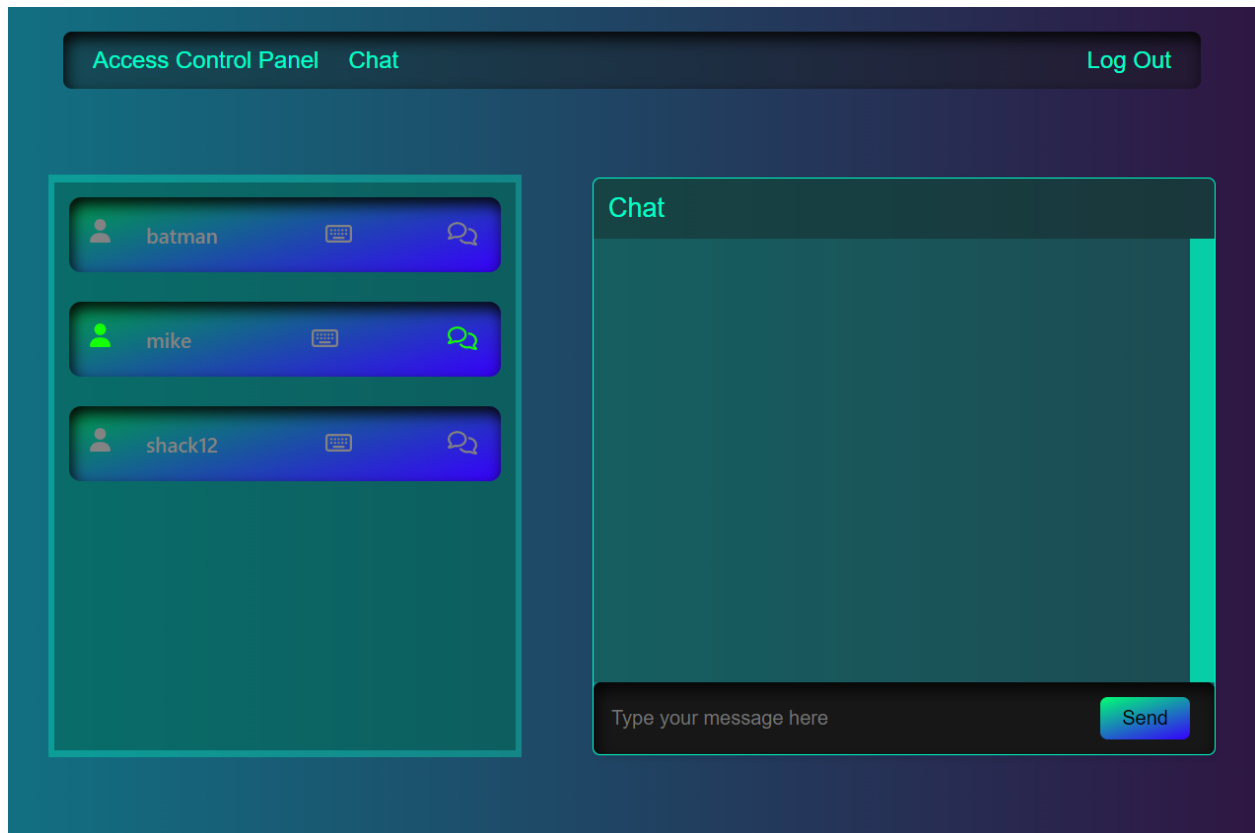
Login

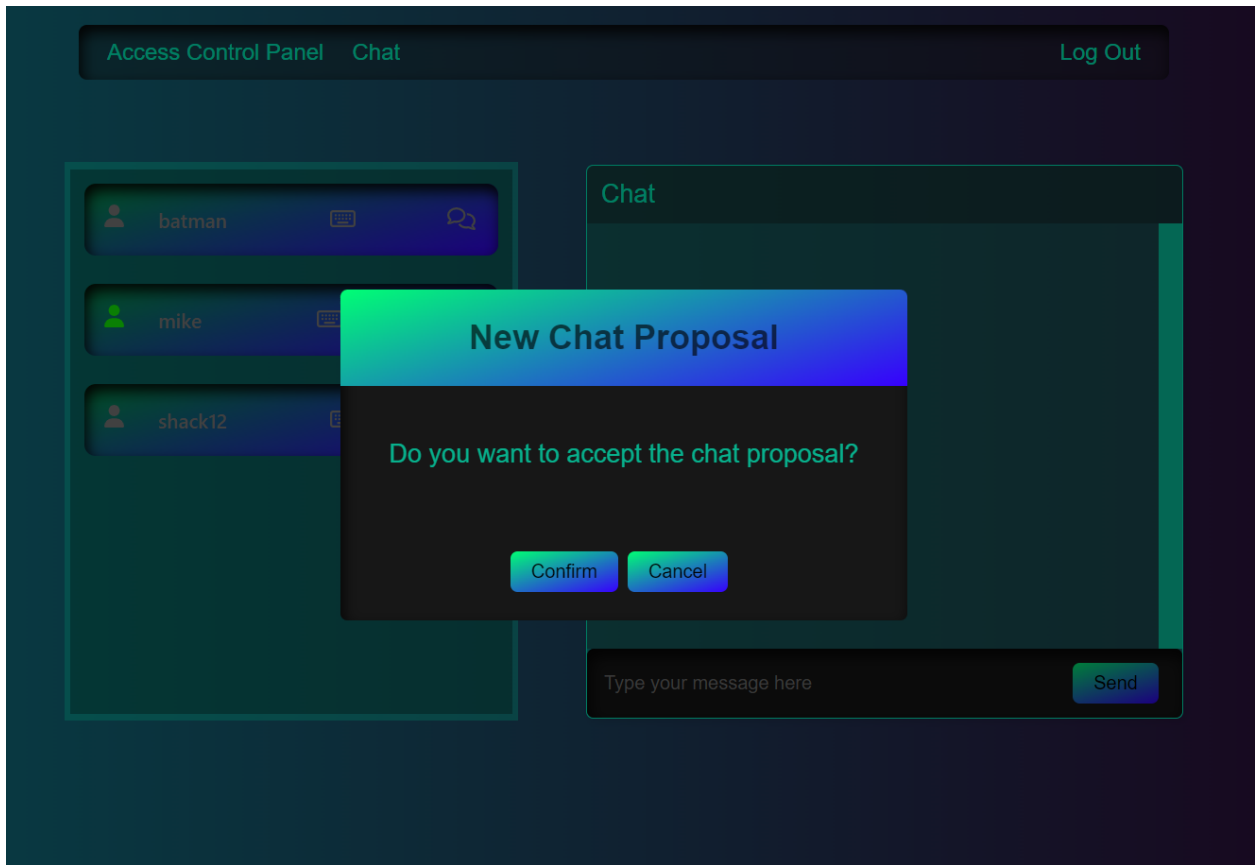
Forgot Password

se propone una nueva conversación clickeando el username que se desee :

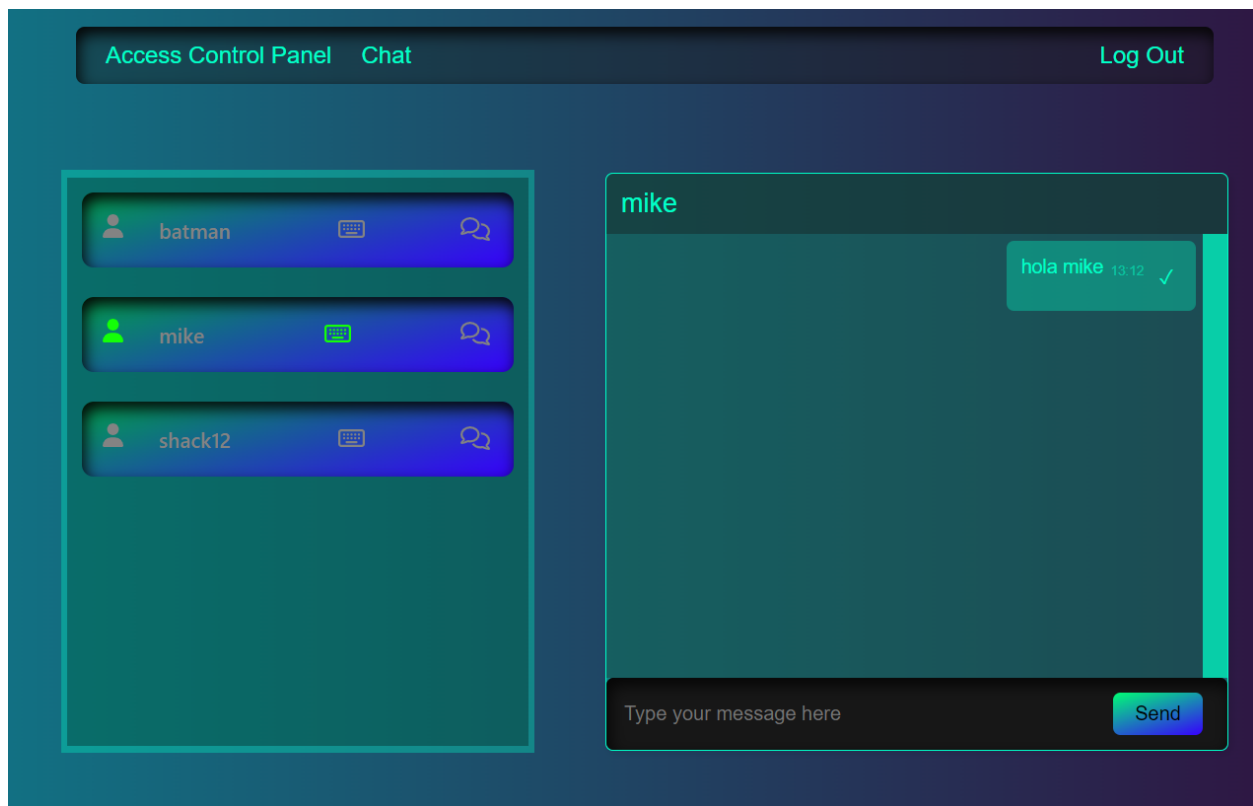


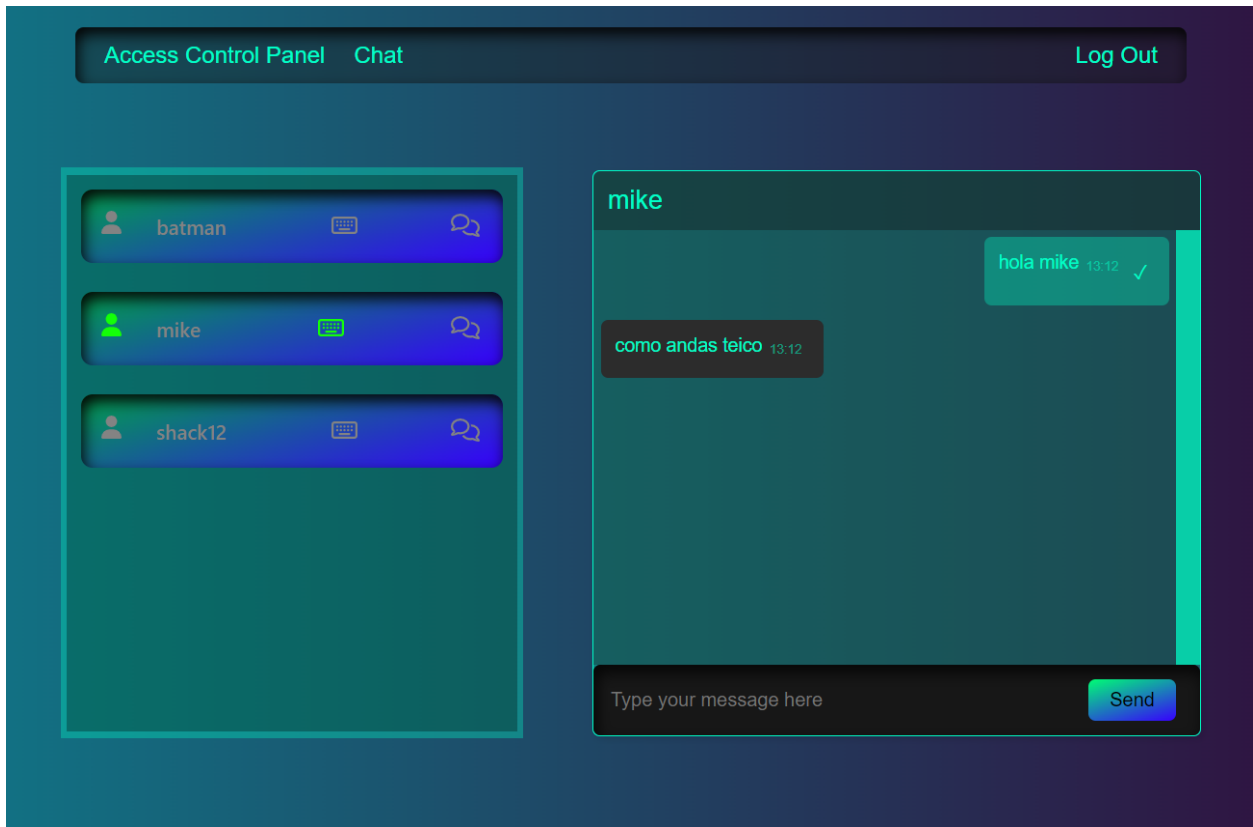
Luego se puede aceptar o rechazar la propuesta :

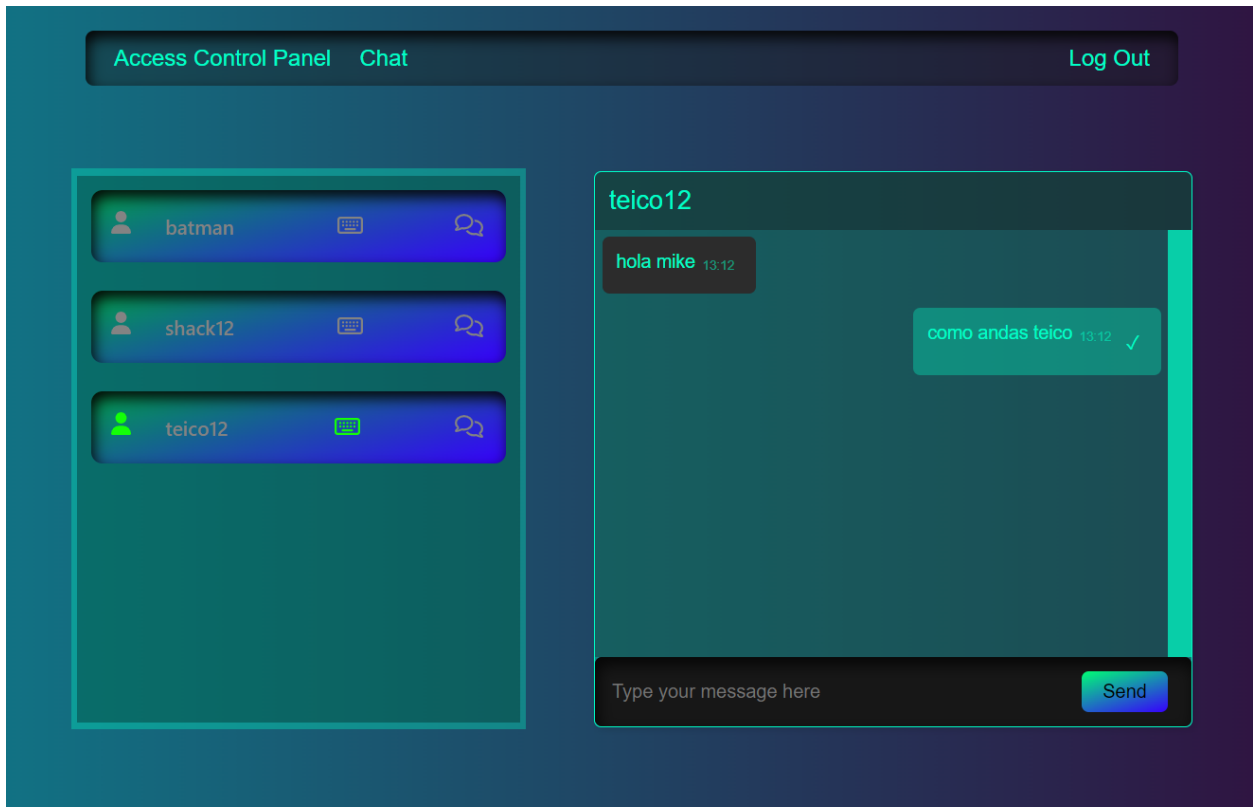




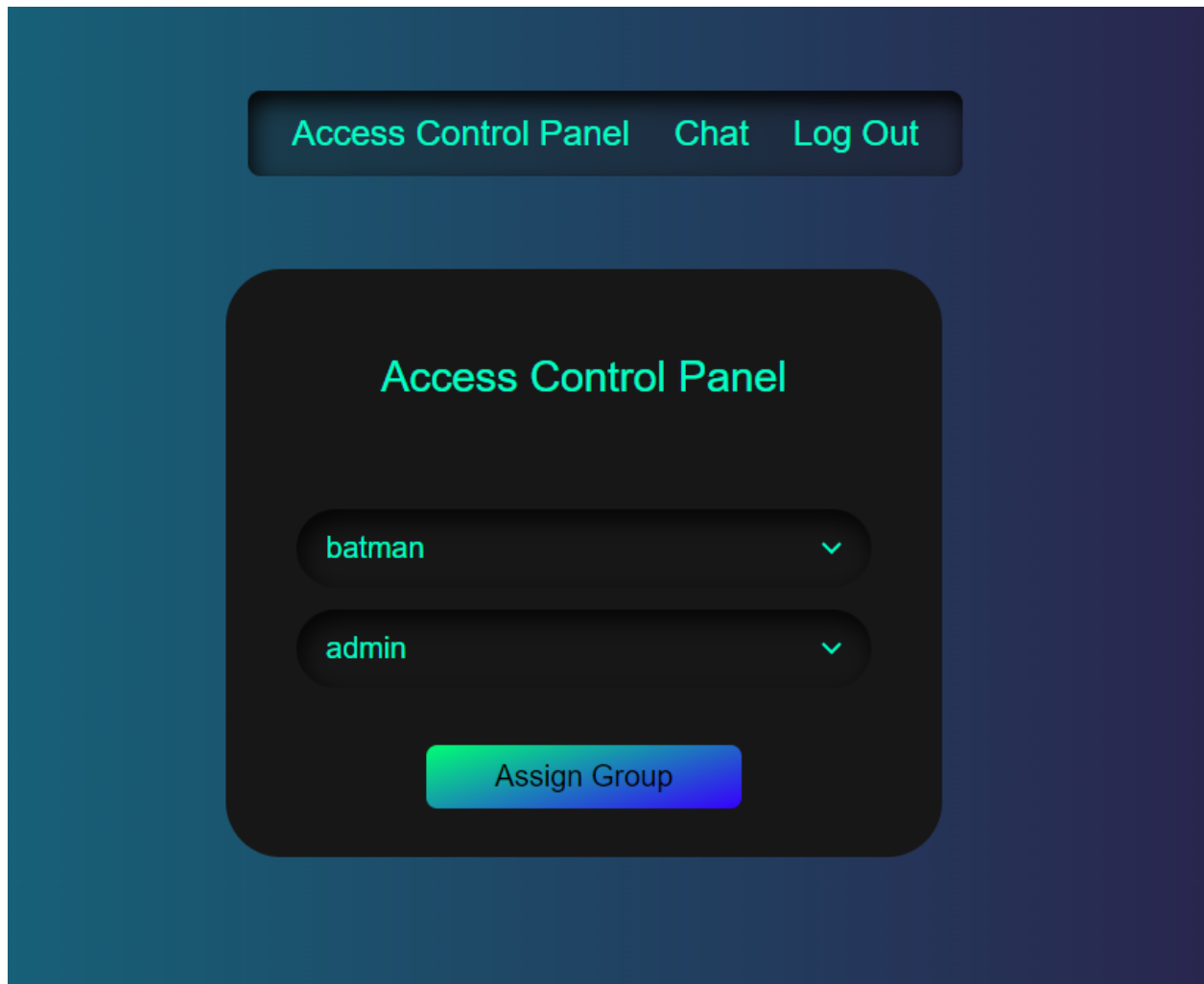
Luego se puede aceptar o rechazar la propuesta , en caso de ser aceptada ya se puede empezar a intercambiar mensajes seleccionando el chat cuando esta en verde:







El sistema incluye un panel de control de accesos que permite al administrador del sistema cambiar a un usuario a diferentes grupos, otorgándole más o menos acceso al sistema según sea necesario:



Y finalmente puedes hacer el Logout del sistema :

