

Documentación protocolo REST - MensajerO

Recursos disponibles

- Login
- Conversacion
- Usuario

Formato de contenido: json

Recurso - Login

Acción: loguear usuario en el sistema

URI: server/Login

Método: POST

Body: { "user": <str:usuario>,
 "pwd": <str:password>
 }

Restricciones sobre identificador de usuario: alfanumérico, máximo 12 caracteres.

Reply

En caso de usuario y contraseña válidos:

- **Response code:** 201 Created
- **Body:** { "token": <str:login_token> }

En caso de usuario o contraseña inválido:

- **Response code:** 401 Unauthorized request
-

Recurso - Usuario

Acción: solicitar lista de usuarios

URI: server/Usuario

Método: GET

Body: { "token" : <str:login_token> }

Reply

En caso de token inválido:

- **Response code:** 401 Unauthorized request

En case de token valido:

- **Response code:** 200 OK
 - **Body:** { "users": [<str:user1>, <str:user2>, ...] }
-

Acción: solicitar perfil de usuario

URI: server/Usuario/<str:user_id>

Metodo: GET

Body: { "token" : <str:login_token> }

Reply

En caso de token inválido:

- **Response code:** 401 Unauthorized request

En caso de token válido:

Usuario inválido:

- **Response code:** 401 Unauthorized request

Usuario válido:

- **Response code:** 200 OK
-

Acción: modificar o crear usuario

URI: server/Usuario/<str:user_id>

Metodo: PUT

Body: { "token": <str:login_token>,
"nombre": <str:nombre>,
"foto": <str:foto>,
"ubicación": <str:ubicacion>
"conectado": <bool:conectado>
}

(Confirmar luego)

Formato `ubicacion`: "<str:float_latitud>,<str:float_longitud>" ? otra versión: <str:nombre custom>

Formato `foto`: base64 JPG?

Reply

En caso de token inválido:

- **Response code:** 401 Unauthorized request

En caso de token válido:

Usuario inválido:

- **Response code:** 401 Unauthorized request

Usuario válido y atributos inválidos:

- **Response code:** 400 Bad request
- **Body:** { "error" : <str:descripcion_error> }

Usuario válido y atributos válidos:

- **Response code:** 201 Created
-

Recurso - Conversación

Acción: solicitar conversación anterior entre usuario <user1> y <user2>

URI: server/Conversacion/<str:user1>.<str:user2>

Nota: el orden de los argumentos es irrelevante, ambas URI se resuelven por el mismo controlador.

Metodo: GET

Body: { "token" : <str:login_token> }

Reply

Nota: los mensajes se devuelven en orden ascendente según la fecha y hora de su recepción.

En caso de token inválido:

- **Response code:** 401 Unauthorized request

En caso de token válido:

Usuario(s) inválidos:

- **Response code:** 401 Unauthorized request

Usuario válido y atributos inválidos:

- **Response code:** 400 Bad request
- **Body:** { "error" : <str:descripcion_error> }

Usuario válido y atributos válidos:

- **Response code:** 200 OK
 - **Body:** { "mensajes": [<str:msg1>, <str:msg2>, ...] }
- Format de msg: <timestamp>.<usuario>.<mensaje>*

Acción: enviar mensaje de usuario <user1> a <user2>

URI: server/Conversacion/<str:user1>.<str:user2>

Nota: <user1> es quien envía el mensaje, <user2> el destinatario.

Metodo: POST

Body: { "token" : <str:login_token>,
 "mensaje" : <str:mensaje>
 }

Reply

En caso de token inválido:

- **Response code:** 401 Unauthorized request

En caso de token válido:

Usuario(s) inválidos:

- **Response code:** 401 Unauthorized request

Usuario válido y mensaje válidos:

- **Response code:** 201 Created
-