

useVerifyExternalUser

Overview

The `useVerifyExternalUser` hook allows you to verify an external user by passing a securely signed JWT containing the user's details. This function ensures that the user is authenticated and authorized by your system.

Usage Example

```
import { useVerifyExternalUser } from "@replyke/react-js";

function VerifyUser() {
  const verifyExternalUser = useVerifyExternalUser();

  const handleVerify = async () => {
    try {
      const result = await verifyExternalUser({
        userJwt: "securely-signed-jwt", // Replace with the actual JWT
      });

      console.log("User verification successful:", result);
    } catch (error) {
      console.error("User verification failed:", error.message);
    }
  };

  return <button onClick={handleVerify}>Verify User</button>;
}
```

Parameters & Returns

Parameters

The hook returns a function that accepts an object with the following fields:

Parameter	Type	Required	Description
<code>userJwt</code>	<code>string</code>	Yes	A securely signed JWT containing the external user's details.

Returns

The returned function resolves with an object containing:

Property	Type	Description
<code>accessToken</code>	<code>string</code>	The access token for the newly created user. It should be stored in memory.
<code>refreshToken</code>	<code>string</code>	The refresh token for the user session. React Native apps must store this securely, whereas React apps do not need to store it manually as it is managed via cookies.
<code>user</code>	<code>object</code>	The registered user's details.

Last updated on May 7, 2025