

Using Custom Authentication in PnP Unity SDK

Custom Authentication is a way to authenticate users with your own custom authentication service. For example, while authenticating with Google, you have the ability to use your own Google Client ID and Dashboard to authenticate users directly. To login using your own custom JWT issuers like Auth0, AWS Cognito, or Firebase, you can add the your configuration to the loginConfig field of the Web3AuthOptions class.

The loginConfig field is a key value map. The key should be one of the Web3AuthProvider in its string form, and the value should be a LoginConfigItem struct instance.

First, configure your own verifier in the Web3Auth Dashboard to use custom authentication.

tip Check out how to create [Custom Verifier](#) on Web3Auth Dashboard. Then, you should specify the details of your verifier in the loginConfigItem struct, the details of this struct are as follows.

note This is a paid feature and the minimum [pricing plan](#) to use this SDK in a production environment is the Growth Plan . You can use this feature in the development environment for free.

Arguments^a

LoginConfigItem

^a

- Table
- Interface

Parameter Description verifier The name of the verifier that you have registered on the Web3Auth Dashboard. It's a mandatory field, and accepts string as a value. typeOfLogin Type of login of this verifier, this value will affect the login flow that is adapted. For example, if you choose google , a Google sign-in flow will be used. If you choose jwt , you should be providing your own JWT token, no sign-in flow will be presented. It's a mandatory field, and accepts TypeOfLogin as a value. clientId Client id provided by your login provider used for custom verifier. e.g. Google's Client ID or Web3Auth's client Id if using 'jwt' as TypeOfLogin. It's a mandatory field, and accepts string as a value. name? Display name for the verifier. If null, the default name is used. It accepts string as a value. description? Description for the button. If provided, it renders as a full length button. else, icon button. It accepts string as a value. verifierSubIdentifier? The field in JWT token which maps to verifier id. Please make sure you selected correct JWT verifier id in the developer dashboard. It accepts string as a value. logoHover? Logo to be shown on mouse hover. It accepts string as a value. logoLight? Light logo for dark background. It accepts string as a value. logoDark? Dark logo for light background. It accepts string as a value. mainOption? Show login button on the main list. It accepts bool as a value. Default value is false. showOnModal? Whether to show the login button on modal or not. Default value is true. showOnDesktop? Whether to show the login button on desktop. Default value is true. showOnMobile? Whether to show the login button on mobile. Default value is true. public

class

LoginConfigItem

```
{ public
```

```
string verifier {
```

```
get ;
```

```
set ;
```

```
} public
```

```
TypeOfLogin typeOfLogin {
```

```
get ;
```

```
set ;
```

```
} public
```

```
string name {
```

```
get ;
```

```
set ;
```

```
} public
string description {
    get ;
    set ;
} public
string clientId {
    get ;
    set ;
} public
string verifierSubIdentifier {
    get ;
    set ;
} public
string logoHover {
    get ;
    set ;
} public
string logoLight {
    get ;
    set ;
} public
string logoDark {
    get ;
    set ;
} public
bool mainOption {
    get ;
    set ;
}
=
false ; public
bool showOnModal {
    get ;
    set ;
}
=
true ; public
```

```

bool showOnDesktop {
get ;
set ;
}
=
true ; public
bool showOnMobile {
get ;
set ;
}
=
true ; }

```

TypeOfLogin

[â](#)

```

public
enum
TypeOfLogin { [ EnumMember ( Value =
"google" ) ] GOOGLE , [ EnumMember ( Value =
"facebook" ) ] FACEBOOK , [ EnumMember ( Value =
"reddit" ) ] REDDIT , [ EnumMember ( Value =
"discord" ) ] DISCORD , [ EnumMember ( Value =
"twitch" ) ] TWITCH , [ EnumMember ( Value =
"apple" ) ] APPLE , [ EnumMember ( Value =
"line" ) ] LINE , [ EnumMember ( Value =
"github" ) ] GITHUB , [ EnumMember ( Value =
"kakao" ) ] KAKAO , [ EnumMember ( Value =
"linkedin" ) ] LINKEDIN , [ EnumMember ( Value =
"twitter" ) ] TWITTER , [ EnumMember ( Value =
"weibo" ) ] WEIBO , [ EnumMember ( Value =
"wechat" ) ] WECHAT , [ EnumMember ( Value =
"email_passwordless" ) ] EMAIL_PASSWORDLESS , [ EnumMember ( Value =
"email_password" ) ] EMAIL_PASSWORD , [ EnumMember ( Value =
"jwt" ) ] JWT }

```

Example[â](#)

- Google
- Facebook

```

void

```

```

Start ( ) { web3Auth =
GetComponent < Web3Auth
    ( ) ; var loginConfigItem =
new
LoginConfigItem ( ) { verifier =
"verifier-name" ,
// get it from web3auth dashboard typeOfLogin = TypeOfLogin . GOOGLE , clientId =
getString ( R . string . web3auth_google_client_id )
// google's client id } web3Auth . setOptions ( new
Web3AuthOptions ( ) { redirectUrl =
new
Uri ( "torusapp://com.torus.Web3AuthUnity/auth" ) , clientId =
"BAwFgL-r7wzQKmtcdiz2uHJKNZdK7gzEf2q-m55xfzSZOw8jLOyli4AVvvzaEQO5nv2dFLEmf9LBkF8kaq3aErg" , network =
Web3Auth . Network . TESTNET , // Optional loginConfig object loginConfig =
new
Dictionary < string , LoginConfigItem
    { { "google" , loginConfigItem } } } ) ; } void
Start ( ) { web3Auth =
GetComponent < Web3Auth
    ( ) ; var loginConfigItem =
new
LoginConfigItem ( ) { verifier =
"verifier-name" ,
// get it from web3auth dashboard typeOfLogin = TypeOfLogin . FACEBOOK , clientId =
getString ( R . string . web3auth_facebook_client_id )
// facebook's client id } web3Auth . setOptions ( new
Web3AuthOptions ( ) { redirectUrl =
new
Uri ( "torusapp://com.torus.Web3AuthUnity/auth" ) , clientId =
"BAwFgL-r7wzQKmtcdiz2uHJKNZdK7gzEf2q-m55xfzSZOw8jLOyli4AVvvzaEQO5nv2dFLEmf9LBkF8kaq3aErg" , network =
Web3Auth . Network . TESTNET , // Optional loginConfig object loginConfig =
new
Dictionary < string , LoginConfigItem
    { { "facebook" , loginConfigItem } } } ) ; } note * dApp Share is only returned for the Custom Authentication
    verifiers. * Also, 2FA should be enabled for the account using it. Use mfaLevel = MFALevel.MANDATORY * in
    theLoginParams * during login. See MFA * for more details.

```

ExtraLoginOptions

for special login methods [a](#)

- To use the EMAIL_PASSWORDLESS

- login, you need to put the email into the login_hint
- field of the extraLoginOptions
- .

public

void

login () { var selectedProvider = Provider . EMAIL_PASSWORDLESS ; var options =

new

LoginParams () { loginProvider = selectedProvider , extraLoginOptions =

new

ExtraLoginOptions () { login_hint =

"hello@web3auth.io" } } ; web3Auth . login (options) ; } * To use the JWT * login, you need to put the jwt token into the additionalParams * field of the extraLoginOptions * .

void

Start () { web3Auth =

GetComponent < Web3Auth

() ; var loginConfigItem =

new

LoginConfigItem () { verifier =

"custom-jwt-verifier" ,

// get it from web3auth dashboard typeOfLogin = TypeOfLogin . JWT , } ; web3Auth . setOptions (new

Web3AuthOptions () { redirectUrl =

new

Uri ("torusapp://com.torus.Web3AuthUnity/auth") , clientId =

"BAwFgL-r7wzQKmtcdiz2uHJKNZdK7gzEf2q-m55xfzSZOw8jLOyli4AVvvzaEQO5nv2dFLEmf9LBkF8kaq3aErg" , network = Web3Auth . Network . TESTNET , // Optional loginConfig object loginConfig =

new

Dictionary < string , LoginConfigItem

{ { "jwt" , loginConfigItem } } }) ; } public

void

login () { var selectedProvider = Provider . JWT ; var options =

new

LoginParams () { loginProvider = selectedProvider , extraLoginOptions =

new

ExtraLoginOptions () { verifierIdField =

"sub" , id_token =

"YOUR_JWT_TOKEN" , } } ; web3Auth . login (options) ; [Edit this page](#) [Previous](#) [Whitelabel](#) [Next](#) [Multi Factor Authentication](#)