

Installing PnP React Native SDK

Selecting your Workflow[^]

In React Native, you have the choice to use one of the following workflows:

Expo Managed Workflow[^]

Your Expo app is built on your Expo's cloud, so you don't have control over the native modules used in the app. Developers build managed workflow apps using `expo-cli` on their computers and a development client on their mobile devices. Managed workflow apps typically use one or more Expo services, such as push notifications, builds, and updates.

Warning Web3Auth SDKs are not compatible with "Expo Go" app. They are compatible only with Custom Dev Client and EAS builds. Please refer to the troubleshooting section for more on this.

Please run `pnpx expo prebuild` to generate native code based on the version of expo a project has installed, before moving forward.

Bare React Native Workflow[^]

Your Bare React Native app is entirely built on your machine. In this workflow, the developer has complete control, along with the complexity that comes with that. Configuration with `app.json` / `app.config.js` is mostly not supported in this context; instead, you will need to configure each native project directly with Swift/Kotlin native modules. Check out the [troubleshooting section](#) for fixing common issues.

tip You can read more about different workflows in the [Expo documentation](#).

Installation[^]

[@web3auth/react-native-sdk](#)

[^]

- npm
- Yarn
- pnpm

```
npm install --save @web3auth/react-native-sdk yarn add @web3auth/react-native-sdk pnpm add @web3auth/react-native-sdk
```

Adding a Web Browser Module[^]

We will also require a `WebBrowser` implementation to allow our JS-based SDK to interact with the native APIs, and there are different extra installation steps depending on whether you are using the bare workflow or managed workflow.

- Expo Managed Workflow
- Bare React Native Workflow

Expo Web Browser[^]

When using our SDK with an Expo-based React Native app (aka managed workflow), you have to install the `expo-web-browser` package as a `WebBrowser` implementation.

```
expo install expo-web-browser
```

React Native Web Browser[^]

When using our SDK with a bare workflow React Native app, you have to install a `WebBrowser` implementation made by us.

- npm
- Yarn
- pnpm

```
npm install --save @toruslabs/react-native-web-browser yarn add @toruslabs/react-native-web-browser pnpm add @toruslabs/react-native-web-browser
```

Adding aStorage

Module

Now with v4, we need to pass aStorage parameter to the SDK, which will be used for session management without storing the private keys of the user in the device.

- Expo Managed Workflow
- Bare React Native Workflow

Expo Secure Store

When using our SDK with an Expo-based React Native app (aka managed workflow), you have to install the expo-secure-store package as aStorage implementation.

```
expo install expo-secure-store
```

React Native Encrypted Storage

When using our SDK with a bare workflow React Native app, you have to install aStorage implementation provided by react-native.

- npm
- Yarn
- pnpm

```
npm install --save react-native-encrypted-storage yarn add react-native-encrypted-storage pnpm add react-native-encrypted-storage
```

Configuration

After you have installed the files needed for your workflow, you'll have to configure the SDK with some additional steps to be able to use the SDK properly.

Expo Managed Workflow

- Adding URL scheme to app.json

To allow the Expo-based SDK to work with exported Expo Android apps, you need to add the designated scheme into app.json

app.json { "expo": { "scheme": "web3authexpoexample" // replace with your own scheme } } * For constructing your redirectUrl *, you'll need to use the expo-linking *, which will help you generate a redirectUrl * for your app. Make sure you * register that URL in the [Web3Auth Developer Dashboard](#) *.

```
App.js import
```

```
Constants ,
```

```
{
```

```
AppOwnership
```

```
}
```

```
from
```

```
"expo-constants" ; import
```

```
*
```

```
as
```

```
Linking
```

```
from
```

```
"expo-linking" ;
```

```
const resolvedRedirectUrl = Constants . appOwnership
```

```
==
```

AppOwnership . Expo

||

Constants . appOwnership

==

AppOwnership . Guest ?

Linking . createURL ("web3auth" ,

{ }) :

Linking . createURL ("web3auth" ,

{ scheme : scheme }) ; tip You may refer to [these example apps](#) and try it out yourself.

Bare React Native Workflow

For the bare workflow, you need to perform additional installation steps, alongside specific configurations for Android and iOS separately.

Android

- Make sure that the minimum SDK compile version in build.gradle
- is 31 or more.

android/build.gradle buildToolsVersion = "31.0.0" minSdkVersion = 21 compileSdkVersion = 31 targetSdkVersion = 31 * Add the intent filter with scheme * defined in your AndroidManifest.xml

android/app/src/main/AndroidManifest.xml < intent-filter

< action

android: name = " android.intent.action.VIEW "

/> < category

android: name = " android.intent.category.DEFAULT "

/> < category

android: name = " android.intent.category.BROWSABLE "

/>

replace with your own scheme

< data

android: scheme = " web3authrnextample "

/> </ intent-filter

- SDK version 31 requires you to explicitly define android:exported="true"
- in AndroidManifest.xml
- , and check whether it is correctly present or
- not.

android/app/src/main/AndroidManifest.xml < activity android: name = " .MainActivity " android: label = " @string/app_name " android: configChanges = " keyboard|keyboardHidden|orientation|screenSize|uiMode " android: launchMode = " singleTask " android: windowSoftInputMode = " adjustResize " android: exported = " true "

- Your redirectUrl
- is your defined scheme following some identifier or your choice. For example, if your scheme is web3authrnextample
- , you can
- define your redirectUrl
- as web3authrnextample://openlogin

- . Make sure you register this redirectUrl
- in the [Web3Auth Developer Dashboard](#)
- .

App.js const scheme =

"web3authrnexample" ;

// Or your desired app redirection scheme const resolvedRedirectUrl =

{ scheme } ://openlogin ;

ios

- Make sure that the minimum SDK compile version in Podfile
- is 14 or more.

ios/Podfile platform :ios, '14' * Install the Cocoa Pods within your project directory.

cd ios pod install * In iOS, you don't need to add redirectUrl * as an iOS Custom URL Scheme, however, you may add it to your Info.plist *, but it is not required. Make * sure your redirectUrl * is registered in the [Web3Auth Developer Dashboard](#) * .

tip You may refer to [these example apps](#) and try it out yourself.

Troubleshooting

Bundler Issues: Missing Dependencies

You might face issues mentioning that certain dependencies are missing within the React Native environment. These are node dependencies that need to be polyfilled in your application, to enable Web3Auth functionalities. Furthermore, your bundler needs to be reconfigured to use them while building the app. Please check out our [React Native Troubleshooting Guide](#) [Edit this page](#) [Previous Web3Auth PnP React Native SDK Next Initialize](#)