Create Smart Accounts with WAGMI React Hooks

This section shows how to use Wagmi React Hooks to create a Smart Account with Biconomy. If you would like to simply see a code implementation, one is availablehere which showcases how to create a smart account using Wagmi.

Dependencies

You will need the following dependencies to create a Smart Account this way:

yarn add wagmi viem @alchemy/aa-core @biconomy/account

Set up Wagmi Config

First we'll need to set up the Wagmi config file to wrap the full app. The context below assumes a Next JS application but can be used in any React framework.

Imports

import

{ WagmiConfig , createConfig , configureChains }

from

"wagmi" ; import

{ baseGoerli }

from

"@wagmi/core/chains" ; import

{ alchemyProvider }

from

"wagmi/providers/alchemy" ; import

{ publicProvider }

from

"wagmi/providers/public" ; import

{ MetaMaskConnector }

from

"wagmi/connectors/metaMask" ; import

{ createPublicClient , http }

from

"viem" ;

Config Setup

const

{ chains , webSocketPublicClient }

=

configureChains ([baseGoerli] , [alchemyProvider ({ apiKey :

""

}) ,

publicProvider ( )]) ;

const config =

createConfig ({ autoConnect :

false , publicClient :

createPublicClient ({ chain : baseGoerli , transport :

http ( ) , }) , connectors :

[new

MetaMaskConnector ({ chains })] , webSocketPublicClient , }) ;

export

default

function

App ({ Component , pageProps } : AppProps)

{ return

(<

< WagmiConfig config = { config }

< Component { ... pageProps }

/

< / WagmiConfig

< /

) ; } With the config completed we can now access the Wagmi hooks in our other components.

Hooks Imports

import

{ useConnect , useAccount , useDisconnect , useWalletClient }

from

"wagmi" ; import

{ createSmartAccountClient , createECDSAOwnershipValidationModule , IPaymaster , createPaymaster , IBundler , createBundler , }

from

"@biconomy/account" ; import

{ useState }

from

"react" ;

Create Bundler and Paymaster Instance

To set up the smart account lets instances of our bundler and Paymaster set up. These opitonal values in creating the smart account will be helpful in accessing the full stack of Account Abstraction made available by the Biconomy SDK.

Connect to Users EOA and Create Smart account

const

{ connect , connectors , error , isLoading , pendingConnector }

= useConnect ( ) ; const

{ address , isConnected }

=

useAccount ( ) ; const

{ disconnect }

=

useDisconnect ( ) ; const

{ data : walletClient }

=

useWalletClient ( ) ; const

[smartAccountAddress , setSmartAccountAddress]

=

useState ( ) ;

const

createSmartAccount

=

async

( )

=>

{ if

(! walletClient)

return ;

const biconomySmartAccount =

await

createSmartAccountClient ({ signer : walletClient , bundlerUrl :

"" ,

// <-- Read about this at https://docs.biconomy.io/dashboard#bundler-url biconomyPaymasterApiKey :

"" ,

// <-- Read about at https://docs.biconomy.io/dashboard/paymaster }) ; console . log ({ biconomySmartAccount }) ; const saAddress =

await biconomySmartAccount . getAccountAddress ( ) ; setSmartAccountAddress (saAddress) ; } ; See a basic implementation in the UI below:

return

(<

< Head

< title

Biconomy x WAGMI < / title

```
        < meta name = "description" content = "WAGMI Hooks With Biconomy"
/
        < meta name = "viewport" content = "width=device-width, initial-scale=1"
/
        < link rel = "icon" href = "/favicon.ico"
/
        < / Head
        < main className = { styles . main }
        < h1
        Biconomy x WAGMI Example < / h1
        { address &&
< h2
        EOA :
{ address } < / h2
        } { smartAccountAddress &&
< h2
        Smart Account :
{ smartAccountAddress } < / h2
        } { connectors . map ( ( connector )
=>
( < button key = { connector . id } onClick = { ( )
=>
connect ( { connector } ) }
        { connector . name } { isLoading && connector . id === pendingConnector ?. id && " (connecting)" } < / button
        ) ) }
{ error &&
< div
        { error . message } < / div
        } { isConnected &&
< button onClick = { disconnect }
        Disconnect < / button
        } { isConnected &&
( < button onClick = { createSmartAccount }
        Create Smart Account < / button
        ) } < / main
        < /
```

) ; You are now ready to get started using WAGMI with Biconomy. For a full code implementation check out this example repo . Previous Viem Next Dynamic