dynamic

Dynamic.xyz is a web3 login solution for everyone, offering straightforward onboarding and embedded wallet options for newcomers and elegant and smart login flows for crypto-native users. We aim to abstract away the complexities of user management, offering powerful developer tools around authentication, authorization and onboarding.

1. Install Dynamic and Biconomy

In your app's repository, install@dynamic-labs/sdk-react-core ,@dynamic-labs/ethereum and@biconomy/account :

yarn add @dynamic-labs/sdk-react-core @biconomy/account tip @dynamic-labs/ethereum will help you to enable all EVM compatible chains including layer 2's i.e. Base as well as Dynamic Embedded Wallets. Learn more about WalletConnectorshere .

2. Initialize & configure Dynamic

2a. Add the DynamicContextProvider

All you need to get started isthe DynamicContextProvider - you will want to wrap your app with this component at the highest possible level, like so:

import

{

DynamicContextProvider

}

from

"@dynamic-labs/sdk-react-core" ; import

{

EthereumWalletConnectors

}

from

"@dynamic-labs/ethereum" ; import

{

EthersExtension

}

from

"@dynamic-labs/ethers-v6" ;

< DynamicContextProvider settings = { { // Find your environment id at https://app.dynamic.xyz/dashboard/developer environmentId :

"REPLACE-WITH-YOUR-ENVIRONMENT-ID" , walletConnectors :

[EthereumWalletConnectors] , walletConnectorExtensions :

[EthersExtension] , } }

{ / *Your app's components* / } </ DynamicContextProvider

; tip We are using Ethers here instead of the default Viem, as Biconomy requires an Ethers signer object. Learn more about switching between Viem and Ethershere . tip To enable embedded wallets for your users, toggle it on along with email/social authin the dashboard .

2b. Access the wallet

Inside any component which is wrapped by DynamicContextProvider, you can access any ofthe provided hooks . While you can access the full context viausedynamiccontext , we are most interested in the users currently connected wallets which we can easily access viauseuserwallets .

tip A user can have branded wallets connected as well as Dynamic embedded wallets, so here we will filter for a users embedded wallet, assuming they had one created when they signed up. import

{ useUserWallets }

from

"@dynamic-labs/sdk-react-core" ;

const userWallets =

useUserWallets ( ) ;

const embeddedWallet = userWallets . find ((wallet)

=> wallet . connector ?. isEmbeddedWallet ===

true) ;

3. Configure your Biconomy bundler and paymaster

Go to theBiconomy Dashboard and configure a Paymaster and a Bundler for your app. Make sure these correspond to the desired network for your user's smart accounts. You can learn more about the dashboardhere

4. Create the smart account

Lastly, using the user's paymaster, bundler, and validation module instances from above, initialize the user's smart account using Biconomy'screateSmartAccountClient method:

...

// Note that we are using the ethers signer here rather than Viem const signer =

await embeddedWallet . connector ?. ethers ?. getSigner ( ) ;

const smartAccount =

await

createSmartAccountClient ({ signer , bundlerUrl :

"" ,

// get from biconomy dashboard https://dashboard.biconomy.io/ biconomyPaymasterApiKey :

"" ,

// Biconomy Paymaster API Key can also be obtained from dashboard rpcUrl :

""

// <-- read about this at https://docs.biconomy.io/Account/methods#createsmartaccountclient }) ;

End to end flow

You can find a complete example of the integration herehttps://github.com/dynamic-labs/dynamic-biconomy-example
Previous WAGMI Next Particle Network