# Implement session keys

Let's go through the steps on how you could enable sessions in your dApps with Argent.

Pre-requisites

- pnpm add starknetkit@2.2.7
- (or newer versions)
-
    - can be installed withpnpm add starknetkit@next
-
    - required in order to get the deployment data for an account
- pnpm add @argent/x-sessions@next

Connect using starknetkit

Using starknetkit modal, with all default connectors

```

Copy const{wallet,connectorData,connector}=awaitconnect({ webWalletUrl:ARGENT_WEBWALLET_URL,// env var for webwallet url argentMobileOptions:{ dappName:"Starknetkit example dapp", url:window.location.hostname, chainId:constants.NetworkName.SN_MAIN, icons:[], }, })

```

Using starknetkit modal, with all specific connectors

```

Copy const{wallet,connectorData,connector}=awaitconnect({ connectors:[ newInjectedConnector({ options:{ id:"argentX"} }), newInjectedConnector({ options:{ id:"braavos"} }), newArgentMobileConnector({ dappName:"Example dapp", chainId:CHAIN_ID, }), newWebWalletConnector({ url:ARGENT_WEBWALLET_URL}) ] })

```

ARGENT_WEBWALLET_URL

- Mainnet:https://web.argent.xyz/
- Testnet: please reach-out to us if you wish to use web wallet on testnet
-

Get deployment data for an account

When connecting with starknetkit, use thewallet object returned byconnect .

If an account is already deployed, this will throw an exception (so it will need to be managed on client side)

```

Copy const{wallet,connectorData}=res; awaitwallet.request({ type:"wallet_deploymentData", })

```

Create a session

Before the user can execute transactions, a session needs to be created.

The dapp will need to create adappKey , both public and private will be needed.

```

Copy constprivateKey=ec.starkCurve.utils.randomPrivateKey(); constdappKey:DappKey={ privateKey, publicKey:ec.starkCurve.getStarkKey(privateKey), };

```

When creating a session, 2 values needs to be stored:

- sessionRequest, created usingcreateSessionRequest
- from@argent/x-sessions

- sessionSignature, that requires signing a message and created usingopenSession
- from@argent/x-sessions
-

```

Copy constsessionParams:SessionParams={ allowedMethods:[ { "Contract Address":ETHTokenAddress, selector:"transfer", }, ], expiry:Math.floor((Date.now()+1000*60*60*24)/1000)asany;, metaData:{ projectID:"test-dapp", txFees:[ { tokenAddress:ETHTokenAddress, maxAmount:parseUnits("0.1",18).value.toString(), }, ], }, publicDappKey:dappKey.publicKey, };

constsessionRequest=createSessionRequest(allowedMethods,expiry,metaData(isStarkFeeToken),dappKey.publicKey); setSessionRequest(sessionRequest);

constaccountSessionSignature=awaitopenSession({ chainId:awaitprovider.getChainId(), wallet:walletasany, sessionParams, });

setAccountSessionSignature(accountSessionSignature);

```

Execute a tx using session keys

In order to execute a tx, a session account is needed.

To create this account,sessionRequest andaccountSessionSignature created on the step before a required.

```

Copy constsessionAccount=awaitbuildSessionAccount({ accountSessionSignature:stark.formatSignature(accountSessionSignature), sessionRequest, provider, chainId, address, dappKey, argentSessionServiceBaseUrl:ARGENT_SESSION_SERVICE_BASE_URL, });

```

You can now use this account to submit transactions.

ARGENT_SESSION_SERVICE_BASE_URL

- Mainnet:https://cloud.argent-api.com/v1
- Testnet: please reach-out to us if you wish to use sessions on testnet
-

Previous Session Keys Next Session keys with meta-transactions

Last updated3 months ago