D4D4D4;--ch-t-background: #1E1E1E;--ch-t-lighter-inlineBackground: #1e1e1ee6;--ch-t-editor-background: #1E1E1E;--ch-t-editor-foreground: #D4D4D4;--ch-t-editor-rangeHighlightBackground: #ffffff0b;--ch-t-editor-infoForeground: #3794FF;--ch-t-editor-selectionBackground: #264F78;--ch-t-focusBorder: #007FD4;--ch-t-tab-activeBackground: #1E1E1E;--ch-t-tab-activeForeground: #ffffff;--ch-t-tab-inactiveBackground: #2D2D2D;--ch-t-tab-inactiveForeground: #ffffff80;--ch-t-tab-border: #252526;--ch-t-tab-activeBorder: #1E1E1E;--ch-t-editorGroup-border: #444444;--ch-t-editorGroupHeader-tabsBackground: #252526;--ch-t-editorLineNumber-foreground: #858585;--ch-t-input-background: #3C3C3C;--ch-t-input-foreground: #D4D4D4;--ch-t-icon-foreground: #C5C5C5;--ch-t-sideBar-background: #252526;--ch-t-sideBar-foreground: #D4D4D4;--ch-t-sideBar-border: #252526;--ch-t-list-activeSelectionBackground: #094771;--ch-t-list-activeSelectionForeground: #fffffe;--ch-t-list-hoverBackground: #2A2D2E; }

Send User Operations

In this guide, you will learn how to create Safe user operations , sign them, collect the signatures from the different owners, and execute them.

For more detailed information, see the Starter Kit Reference .

Pimlico(opens in a new tab) is used in this guide as the service provider, but any other provider compatible with the ERC-4337 can be used.

Prerequisites

- Node.js and npm(opens in a new tab)
- A Pimlico account(opens in a new tab)
- and an API key.

Install dependencies

First, you need to install some dependencies.

_10 pnpm add @safe-global/starter-kit

Steps

Imports

Here are all the necessary imports for this guide.

_10 import { _10 createSafeClient, _10 safeOperations, _10 BundlerOptions _10 } from '@safe-global/sdk-starter-kit'

Create a signer

Firstly, you need to get a signer, which will be the owner of a Safe account after it's deployed.

This example uses a private key, but any way to get an EIP-1193 compatible signer can be used.

_10 const SIGNER_ADDRESS = // ... _10 const SIGNER_PRIVATE_KEY = // ... _10 const RPC_URL = 'https://rpc.ankr.com/eth_sepolia'

Initialize theSafeClient

New Safe account Existing Safe account When deploying a new Safe account, you need to pass the configuration of the Safe in thesafeOptions property. The Safe account is configured with your signer as the only owner in this case.

_10 const safeClient = await createSafeClient({ _10 provider: RPC_URL, _10 signer: SIGNER_PRIVATE_KEY, _10 safeOptions: { _10 owners: [SIGNER_ADDRESS], _10 threshold: 1 _10 } _10 }) As this guide is related with ERC-4337 user operations, you need to add thesafeOperations extension to thesafeClient instance to support this functionality.

_12 const bundlerOptions: BundlerOptions = { _12 bundlerUrl:https://api.pimlico.io/v1/sepolia/rpc?apikey={PIMLICO_API_KEY} _12 } _12 _12 const paymasterOptions: PaymasterOptions = { _12 isSponsored: true, _12 paymasterUrl: https://api.pimlico.io/v2/sepolia/rpc?apikey={PIMLICO_API_KEY} _12 } _12 _12 const safeClientWithSafeOperation = await safeClient.extend(_12 safeOperations(bundlerOptions, paymasterOptions) _12) ThesafeClientWithSafeOperation instance has now support for managing user operations.

Create a Safe transaction

Create an array of Safe transactions to execute.

_10 const transactions = [{ _10 to: '0x...', _10 data: '0x', _10 value: '0' _10 }]

Send the Safe operation

If you configured your Safe withthreshold equal to1 , calling thesendSafeOperation method will execute the user operation. However, if thethreshold is greater than1 the other owners of the Safe will need to confirm the user operation until the required number of signatures are collected.

_10 const safeOperationResult = await safeClientWithSafeOperation.sendSafeOperation({ _10 transactions _10 }) _10 _10 const safeOperationHash = safeOperationResult.safeOperations?.safeOperationHash

Confirm the Safe operations

If the user operation needs to be confirmed by other Safe owners, call theconfirmSafeOperation method from a newSafeClient instance initialized with each of the signers that need to confirm it.

_14 const newSafeClient = await createSafeClient({ _14 provider: RPC_URL, _14 signer, _14 safeAddress: '0x...' _14 }) _14 _14 const newSafeClientWithSafeOperation = await newSafeClient.extend(_14 safeOperations({ _14 bundlerUrl: https://api.pimlico.io/v1/sepolia/rpc?apikey={PIMLICO_API_KEY} _14 }, { _14 isSponsored: true, _14 paymasterUrl: https://api.pimlico.io/v2/sepolia/rpc?apikey={PIMLICO_API_KEY} _14 }) _14) Finally, retrieve all the pending user operations from the Safe Transaction Service and confirm the one you just created with each missing owner.

_11 const pendingSafeOperations = _11 await newSafeClientWithSafeOperation.getPendingSafeOperations() _11 _11 for (const safeOperation of pendingSafeOperations.results) { _11 if (safeOperation.safeOperationHash !== safeOperationHash) { _11 return _11 } _11 _11 const safeOperationResult = _11 await newSafeClientWithSafeOperation.confirmSafeOperation({ safeOperationHash }) _11 }

Recap and further reading

After following this guide, you are able to deploy new Safe accounts and create, sign, and submit user operations with the Starter Kit.

Send Transactions Reference Was this page helpful?

Report issue