Build a SUAPP Webapp

This tutorial will show you how to build a SUAPP web application usingsuave-viem , ourtypescript SDK .

info There are two different templates you can use for your SUAPP. One with minimal, TypeScript-only dependencies; and one which uses Next.

eth_sign dependency

1. Confidential Compute Requests on SUAVE do not work with wallets that implement the EIP-1193 Javascript API. Therefore, we use the unsafeeth_sign
2. method to sign CCRs, which does work, but requires that you enable this functionality in wallets like MetaMask.1. To do so in MetaMask, go to "Settings" -> "Advanced" -> scroll to bottom -> switch Eth_sign requests on.
3. Both templates below assume that you arerunning SUAVE locally
4. .

To get a feel for how a SUAVE-enabled web app works, we've provided a couple examples. One is written in Vanilla TS, and one is written withNext .

Vanilla TypeScript Example

This project is written in vanilla TS (built withVite), which means we directly manipulate theDOM Tree in the browser to render the site, rather than using a web framework like React/Next.

This template can be found in thesuave-viem repo underexamples/suave-web-demo .

Setup

Before running the example (in docker or locally), make sure you have aSUAVE devnet running locally .

You'll also need to installFoundry (forge is used to deploy contracts).

Next, clone thesuave-viem repo and deploy the contract we'll be using for this demo to your local SUAVE chain:

clone the repo

git clone https://github.com/flashbots/suave-viem.git

deploy contracts

cd suave-viem/examples/suave ./deployContracts.sh

Run with Docker

We recommend running the Typescript examples inDocker for security, since javascript runtimes have the potential to execute arbitrary code on your machine.

If you're still insuave-viem/examples/suave/ , jump back up tosuave-viem/ :

cd

.. / .. Now we'll build the docker image, which will build the source code and configure the example's environment to connect to your local devnet.

docker build -t suave-web-example . Next, we'll run a docker container with our new image:

docker run -it suave-web-example This will open a bash terminal inside docker, where you'll land inexamples/ .

To run the web example:

cd suave-web-demo bun dev Now your container should be hosting the web app onhttp://172.17.0.2:5173 . If not, look for the correctNetwork address in your terminal output.

Run Locally

system dependencies The following dependencies are required to run this example on your machine:

- foundry
- (forge
- is used to deploy contracts)
- bun
- (JS runtime & package manager) If you're still insuave-viem/examples/suave/ , jump up tosuave-web-demo/ :

cd

.. /suave-web-demo Install the project's dependencies and start the web app:

bun install bun dev This template uses the same MEV-Share example contract we worked with using the Golang SDK in the previous tutorial .

If you're struggling with any of the above, you can also find this pure TypeScript template as a standalone repo here .

Next.js Example

This template comes with a more extensive frontend framework, which uses Next (in TypeScript) and therefore depends on React. You can get it running by first cloning the repo and installing its dependencies.

Make sure you have previously built and symlinked suave-viem for this to work:

git clone git@github.com:andytudhope/build-a-suapp-next-ts.git cd build-a-suapp-next-ts yarn Setupforge to compile your contracts:

cd packages/forge/ forge install forge build Deploy the compiled contracts from the root directory (you need to have SUAVE running locally for this to work):

chmod +x packages/forge/deploy yarn contracts:deploy You can start the frontend with:

yarn fe:dev

Conclusion

You now have two different templates from which to begin building your own SUAPP .

These templates demonstrate how to interact with SUAVE confidentially, both directly and with data from another domain. Follow thenext tutorial to understand how.

Good luck and happy building ⚡ . Edit this page Previous Deploy Contracts Next Confidential Compute Requests