1. Create folder structure
2. Install Vite Boilerplate and run your default app

<https://vitejs.dev/guide/#scaffolding-your-first-vite-project>

sandeeprajbhar@Sandeeps-MacBook-Pro client % npm init vite@latest

Need to install the following packages:

create-vite@latest

Ok to proceed? (y)

✔ Project name: … ./

✔ Package name: … solidity

✔ Select a framework: › react

✔ Select a variant: › react

Scaffolding project in /Users/sandeeprajbhar/Documents/MyProject/POC/Blockchain Development/Web3.0/Solidity/MyFirstSolidityApp/client/...

Done. Now run:

cd

npm install

npm run dev

npm notice

npm notice New major version of npm available! 7.24.0 -> 8.4.1

npm notice Changelog: https://github.com/npm/cli/releases/tag/v8.4.1

npm notice Run npm install -g npm@8.4.1 to update!

npm notice

sandeeprajbhar@Sandeeps-MacBook-Pro client % npm install

sandeeprajbhar@Sandeeps-MacBook-Pro client % npm run dev

Your local app wiil run.

1. Setup Tailwindcss to reduce effort on CSS dev

https://tailwindcss.com/docs/guides/vite

That’s it for react front end

Lets setup a base code for Smart Contract

1. Initialize empty json package : npm init -y
2. Install react icon and ethers in client to interact with Blockchain and smart contract

npm install react-icons ethers

1. You need to install these dependencies to run the sample project use Hardhat

npm install --save-dev @nomiclabs/hardhat-waffle ethereum-waffle chai @nomiclabs/hardhat-ethers ethers

1. Setting up basic Smart contract project via Hardhat
2. Npx hardhat (create simple project and till set ignore file)
3. Run test cmd to test if everything is ok : npx hardhat test
4. Setup for deploy and testing smart contract
5. Change the script file to deploy and right as per new arrow syntaxt
6. Create a factory class to deploy and get address
7. Add some test ETH for test deployment on testnet

<https://faucet.egorfine.com/>

1. Helper to BC development and deploy to bc

<https://alchemy.com/> (Krishna gmail) – Create app and view the transaction.

10) Deploy app to BC via Ropsen testnet

CMD:

npx hardhat run scripts/deploy.js --network ropsten

A screenshot of a computer

Description automatically generated

It will generate a Transection address and ABI. ABI is used to interact with ETH blockchain.

1. Create a context class to pass a data throughout app and interact with Blockchain. , Setup a Context API of React native in main jsx.
2. Copy the transection address and paste in constant
3. Copy the Transactions.json from Artifact and paste in Client Utils.
4. Implement method to check wallet, connect wallet and first transection,

Graphical user interface, text, application, chat or text message

Description automatically generated

Graphical user interface, text, application, chat or text message

Description automatically generated

A picture containing text, screenshot, computer, computer

Description automatically generated

First transection done from account 1st to 2nd.

Graphical user interface, text, application

Description automatically generated

1. Now let’s see services.

For Dummy data

1. Transection Class for transection list
2. Dummy data
3. Show dummy transaction with GIPHY as dynamic image
4. Use real transection data
5. To Query dynamic gif by key word <https://developers.giphy.com/dashboard/> (krishan8@ / I\*\*\*\*\*4)
6. Create custom hooks for Async image load
7. Update the Contract:

npm init -y

npm install --save-dev @nomiclabs/hardhat-waffle ethereum-waffle chai @nomiclabs/hardhat-ethers ethers

Setting up basic Smart contract project via Hardhat

1. Npx hardhat (create simple project and till set ignore file) – it will automatically generate project
2. Run test cmd to test if everything is ok : npx hardhat test

Create Transaction Contract

Create deploy script from sample script