**Project Details**

To run it :

* Run “npx hardhat node” ->this will start hardhat blockchain network
* Run “npx hardhat run --network localhost scripts/deploy.js” -> this will deploy the smartcontract mentioned in script/deploy.js file

And store the artifacts in cleient/src folder

* Run “npm start” in client folder -> this will run react application

**Hardhat:**

Configuration:

module.exports = {

solidity: "0.8.9",

networks: {

hardhat: {

chainId: 1337,

},

},

paths: {

artifacts: "./client/src/artifacts",

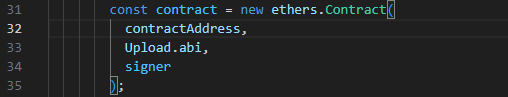
},

};

By default when we start hardhat , it start on <http://127.0.0.1:8545/>

To interact with deployed contract we are using hardhat inbuilt ether library.

Then we create the deployed contract object by passing the contract address, ABI, and signer to the ethers.Contract constructor."



Now the contract is used to call the deployed functions

View type: if only calling value not write in contract

Pure type: have to write or modify variable in contract

If in the contract if function is view type:



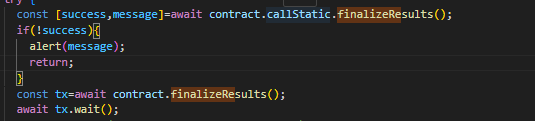
Call the function and right side variable show return variable mention in contract

const [isCommitZero,rtokens,data, contractSuccess,eventId]=await contract.getResults();

But in case of pure(not view):



We have to call the function using callstatic to get return variable then call the function again to do txn



Metamask:

To connect metamask with our application below code is used and here provide is metamask

const provider = new ethers.providers.Web3Provider(window.ethereum);

const signer = provider.getSigner();

the signer here is the same that we use while creating contract object, when we call the contract function the signer ask the metamask to do the txn and then it is prompt to user to complete it.