**Negawatt Trading-Project Details**

**Steps to run the NegawattIncentiveCurtailment contract**

1. Open VS code
2. Select File Menu🡪‘Open Folder’
3. Select Project Folder “Blockchain-Txn-Both” (browse particular folder from system)
4. Select Terminal🡪 New Terminal
5. Select Split Terminal🡪 command prompt
6. Select Split Terminal 🡪 command prompt (2 times)
7. To run hard hat blockchain network use “***npx hardhat node***” at command prompt window (**First**)
8. It starts “Started HTTP and WebSocket JSON-RPC server at http://127.0.0.1:8545/” and it display some accounts with their private keys
9. Then run “***npx hardhat run --network localhost scripts/deploy.js***” in **second** command prompt window-> this will deploy the smart contract mentioned in script/deploy.js file
10. Change location to “**cd client**” >
11. Then > npm start (Run “***npm start***” in client folder -> this will run the react application)
12. It will open the “Negawatt Energy Trading” application in the local web after connecting the MetaMask account
13. (**InputData.js** code available in **components** and **NegawattIncentiveCurtailment** contarct code available in **contracts**)

**Hardhat:**

Configuration:

module.exports = {

solidity: "0.8.9",

networks: {

hardhat: {

chainId: 1337,

},

},

paths: {

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

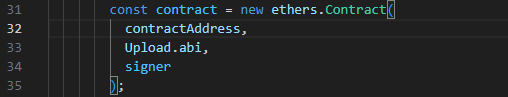
},

};

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:



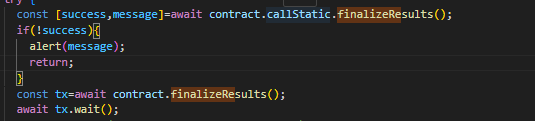
Call the function and right side variable show return variable mention in contract

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.

**Hard Hat configuration**

**require("@nomicfoundation/hardhat-toolbox");**

**/\*\* @type import('hardhat/config').HardhatUserConfig \*/**

**module.exports = {**

**solidity: "0.8.9",**

**networks: {**

**hardhat: {**

**chainId: 1337,**

**},**

**},**

**paths: {**

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

**},**

**};**

**Hard hat dependence package**

**{**

**"devDependencies": {**

**"@nomicfoundation/hardhat-toolbox": "^2.0.2",**

**"hardhat": "^2.23.0"**

**},**

**"dependencies": {**

**"client": "^0.0.1",**

**"create-react-app": "^5.1.0"**

**}**

**}**

**Deployment**

**const hre = require("hardhat");**

**async function main() {**

**const Upload = await hre.ethers.getContractFactory("NegawattIncentiveCurtailment");**

**const upload = await Upload.deploy();**

**await upload.deployed();**

**console.log("Library deployed to:", upload.address);**

**}**

**main().catch((error) => {**

**console.error(error);**

**process.exitCode = 1;**

**});**

**Package.json Client**

**{**

**"name": "client",**

**"version": "0.1.0",**

**"private": true,**

**"dependencies": {**

**"@testing-library/dom": "^10.4.0",**

**"@testing-library/jest-dom": "^6.6.3",**

**"@testing-library/react": "^16.3.0",**

**"@testing-library/user-event": "^13.5.0",**

**"react": "^19.1.0",**

**"react-dom": "^19.1.0",**

**"react-scripts": "5.0.1",**

**"web-vitals": "^2.1.4"**

**},**

**"scripts": {**

**"start": "react-scripts start",**

**"build": "react-scripts build",**

**"test": "react-scripts test",**

**"eject": "react-scripts eject"**

**},**

**"eslintConfig": {**

**"extends": [**

**"react-app",**

**"react-app/jest"**

**]**

**},**

**"browserslist": {**

**"production": [**

**">0.2%",**

**"not dead",**

**"not op\_mini all"**

**],**

**"development": [**

**"last 1 chrome version",**

**"last 1 firefox version",**

**"last 1 safari version"**

**]**

**},**

**"devDependencies": {**

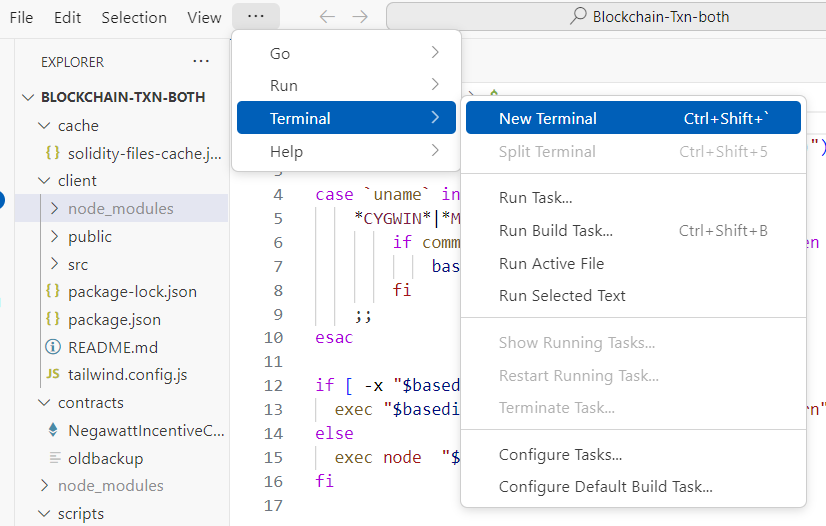
**"tailwindcss": "^3.4.17"**

**}**

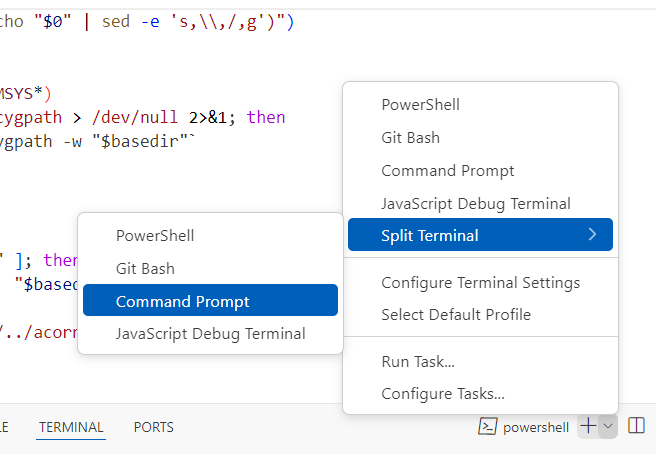
**}**

**Steps to run the NegawattIncentiveCurtailment contract**

1. Open VS code
2. Select File Menu🡪‘Open Folder’
3. Select Project Folder “Blockchain-Txn-Both” (browse particular folder from system)
4. Select Terminal🡪 New Terminal



1. Select Split Terminal🡪 command prompt
2. Select Split Terminal 🡪 command prompt (2 times)

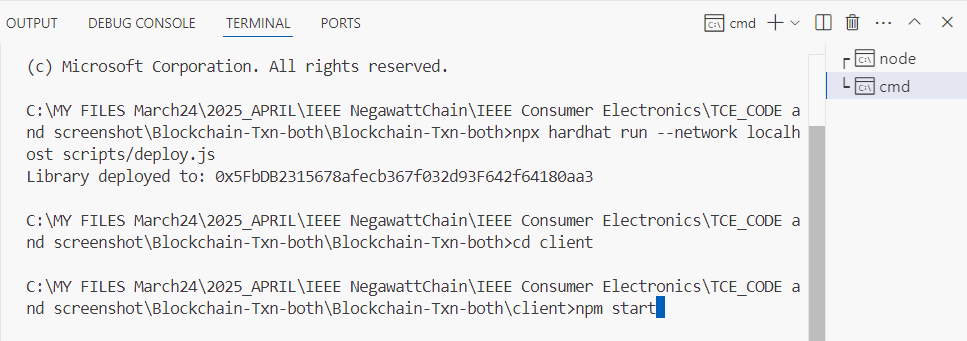




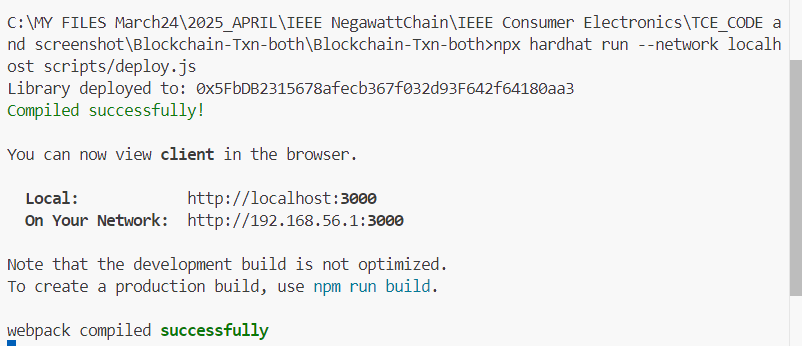
1. To run hard hat blockchain network use “npx hardhat node” at command prompt window (First)
2. It starts “Started HTTP and WebSocket JSON-RPC server at <http://127.0.0.1:8545/>” and it display some accounts with their private keys



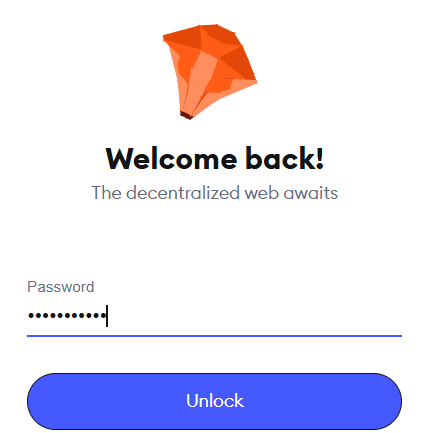
1. Then run “npx hardhat run --network localhost scripts/deploy.js” in second command prompt window-> this will deploy the smart contract mentioned in script/deploy.js file

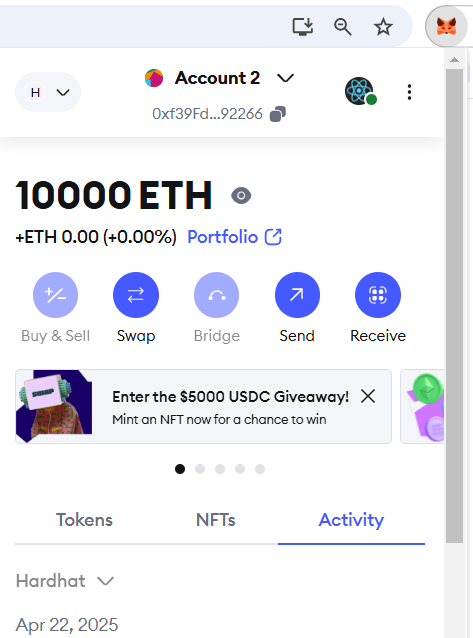
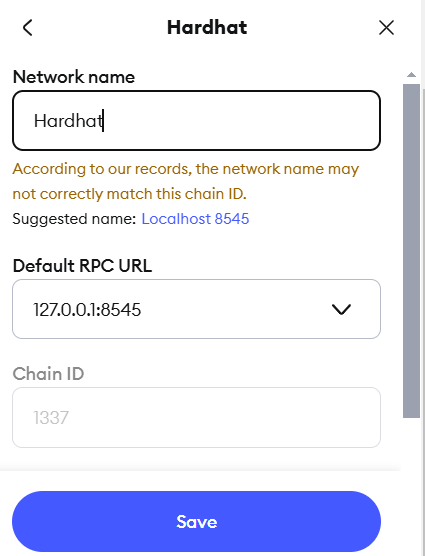


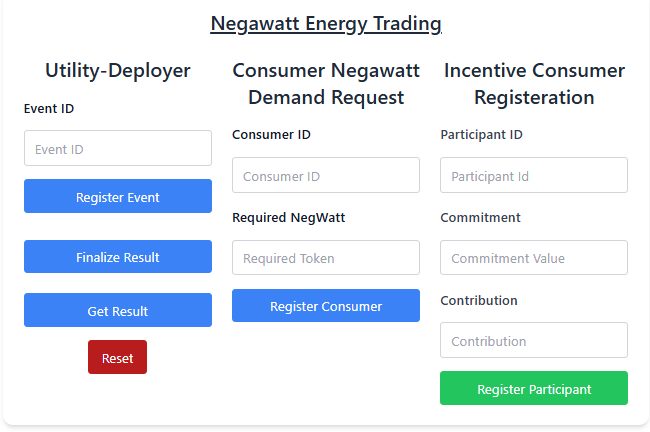
1. Change location to “cd client” > npm start (Run “npm start” in client folder -> this will run the react application)



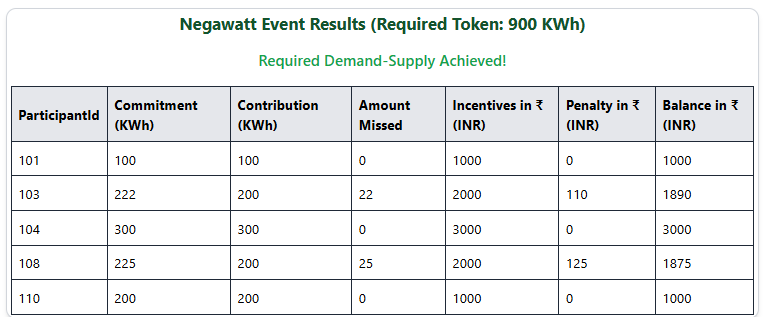
1. It will open the “Negawatt Energy Trading” application in the local web after connecting the MetaMask account

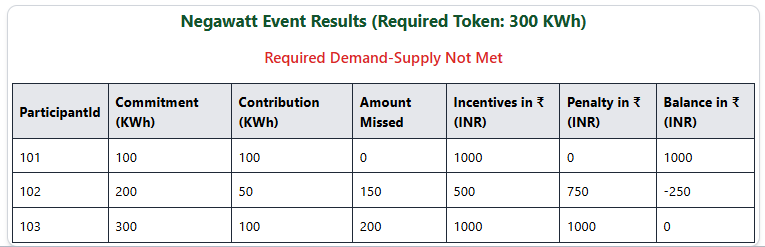




1. Final Results





1. Components🡪 InputData.js (Labels Editing & Input Editing)