# Base dApp Template — small app + deploy guide

## File tree

base-dapp-template/  
├─ contracts/  
│ └─ SimpleStorage.sol  
├─ hardhat/  
│ ├─ hardhat.config.js  
│ └─ scripts/  
│ └─ deploy.js  
├─ frontend/  
│ ├─ package.json  
│ ├─ index.html  
│ └─ src/  
│ ├─ main.jsx  
│ └─ App.jsx  
├─ .env.example  
├─ .gitignore  
├─ README.md  
└─ LICENSE

## contracts/SimpleStorage.sol

// SPDX-License-Identifier: MIT  
pragma solidity ^0.8.19;  
  
contract SimpleStorage {  
 uint256 private value;  
 event ValueChanged(uint256 newValue, address changedBy);  
  
 function set(uint256 \_value) external {  
 value = \_value;  
 emit ValueChanged(\_value, msg.sender);  
 }  
  
 function get() external view returns (uint256) {  
 return value;  
 }  
}

## hardhat/hardhat.config.js

require("dotenv").config();  
require("@nomiclabs/hardhat-ethers");  
  
const { PRIVATE\_KEY, BASE\_SEPOLIA\_RPC } = process.env;  
  
module.exports = {  
 solidity: "0.8.19",  
 networks: {  
 baseSepolia: {  
 url: BASE\_SEPOLIA\_RPC || "https://sepolia.base.org",  
 chainId: 84532,  
 accounts: PRIVATE\_KEY ? [PRIVATE\_KEY] : [],  
 },  
 },  
};

## hardhat/scripts/deploy.js

async function main() {  
 const [deployer] = await ethers.getSigners();  
 console.log("Deploying contracts with account:", deployer.address);  
  
 const SimpleStorage = await ethers.getContractFactory("SimpleStorage");  
 const simple = await SimpleStorage.deploy();  
 await simple.deployed();  
  
 console.log("SimpleStorage deployed to:", simple.address);  
}  
  
main().catch((error) => {  
 console.error(error);  
 process.exitCode = 1;  
});

## frontend/package.json

{  
 "name": "base-dapp-frontend",  
 "version": "0.1.0",  
 "private": true,  
 "scripts": {  
 "dev": "vite",  
 "build": "vite build",  
 "preview": "vite preview"  
 },  
 "dependencies": {  
 "ethers": "^6.0.0",  
 "react": "^18.0.0",  
 "react-dom": "^18.0.0"  
 },  
 "devDependencies": {  
 "vite": "^5.0.0"  
 }  
}

## frontend/index.html

<!doctype html>  
<html>  
 <head>  
 <meta charset="utf-8" />  
 <meta name="viewport" content="width=device-width, initial-scale=1.0" />  
 <title>Base dApp</title>  
 </head>  
 <body>  
 <div id="root"></div>  
 <script type="module" src="/src/main.jsx"></script>  
 </body>  
</html>

## frontend/src/main.jsx

import React from "react";  
import { createRoot } from "react-dom/client";  
import App from "./App";  
  
createRoot(document.getElementById("root")).render(<App />);

## frontend/src/App.jsx

import React, { useEffect, useState } from "react";  
import { ethers } from "ethers";  
  
// Paste the contract ABI here (SimpleStorage ABI)  
const SIMPLE\_STORAGE\_ABI = [  
 {"inputs":[{"internalType":"uint256","name":"\_value","type":"uint256"}],"name":"set","outputs":[],"stateMutability":"nonpayable","type":"function"},  
 {"inputs":[],"name":"get","outputs":[{"internalType":"uint256","name":"","type":"uint256"}],"stateMutability":"view","type":"function"},  
 {"anonymous":false,"inputs":[{"indexed":false,"internalType":"uint256","name":"newValue","type":"uint256"},{"indexed":true,"internalType":"address","name":"changedBy","type":"address"}],"name":"ValueChanged","type":"event"}  
];  
  
export default function App() {  
 const [provider, setProvider] = useState(null);  
 const [signer, setSigner] = useState(null);  
 const [contract, setContract] = useState(null);  
 const [contractAddress, setContractAddress] = useState("");  
 const [value, setValue] = useState(0);  
 const [input, setInput] = useState(0);  
  
 useEffect(() => {  
 if (window.ethereum) {  
 const p = new ethers.BrowserProvider(window.ethereum);  
 setProvider(p);  
 }  
 }, []);  
  
 async function connect() {  
 if (!provider) return alert("Install MetaMask or Coinbase Wallet");  
 await provider.send("eth\_requestAccounts", []);  
 const s = await provider.getSigner();  
 setSigner(s);  
 }  
  
 async function attach() {  
 if (!signer || !contractAddress) return alert("Connect + provide contract address");  
 const c = new ethers.Contract(contractAddress, SIMPLE\_STORAGE\_ABI, signer);  
 setContract(c);  
 }  
  
 async function read() {  
 if (!contract) return alert("Attach contract first");  
 const v = await contract.get();  
 setValue(v.toString());  
 }  
  
 async function write() {  
 if (!contract) return alert("Attach contract first");  
 const tx = await contract.set(BigInt(input));  
 await tx.wait();  
 read();  
 }  
  
 return (  
 <div style={{fontFamily: 'system-ui, sans-serif', padding: 20}}>  
 <h1>Base dApp — SimpleStorage</h1>  
 <button onClick={connect}>Connect Wallet</button>  
 <div style={{marginTop: 12}}>  
 <input placeholder="Contract address" value={contractAddress} onChange={(e)=>setContractAddress(e.target.value)} style={{width: 400}} />  
 <button onClick={attach} style={{marginLeft:8}}>Attach</button>  
 </div>  
 <div style={{marginTop: 12}}>  
 <div>Current value: {value}</div>  
 <button onClick={read}>Read</button>  
 </div>  
 <div style={{marginTop: 12}}>  
 <input type="number" value={input} onChange={(e)=>setInput(e.target.value)} />  
 <button onClick={write} style={{marginLeft:8}}>Set</button>  
 </div>  
 </div>  
 );  
}

## .env.example

PRIVATE\_KEY=0xYOUR\_PRIVATE\_KEY\_FOR\_DEPLOYER  
BASE\_SEPOLIA\_RPC=https://sepolia.base.org

## .gitignore

node\_modules  
.env  
dist  
build  
.DS\_Store

## README.md (skeleton)

# Base dApp Template  
  
Small starter for deploying a SimpleStorage contract to Base (Sepolia testnet) and connecting with a React frontend.  
  
## Quickstart  
  
1. Clone repo  
2. Copy `.env.example` to `.env` and fill `PRIVATE\_KEY`  
3. `cd hardhat` → `npm install` → `npx hardhat compile`  
4. `npx hardhat run scripts/deploy.js --network baseSepolia`  
5. Note deployed contract address and open `frontend`:  
 - `cd frontend` → `npm install` → `npm run dev`  
6. In the frontend enter the contract address, connect wallet, and interact.

## LICENSE

MIT License (short text included in the LICENSE file)

## Notes

* This template uses Base Sepolia as testnet (chainId 84532). Replace RPC and chainId as needed for mainnet or other Base networks.
* The frontend uses ethers v6 and assumes the user has a browser wallet (MetaMask or Coinbase Wallet) connected and switched to the Base Sepolia network.

## Next steps

* Add prettier / linters
* Add CI (GitHub Actions) for tests and solidity verification
* Add a deployment verification step for BaseScan

*Document created for you — open the canvas panel (“base-dapp-template”) to copy files and publish to GitHub.*