**LAB: XÂY DƯNG DAPP TRÊN MÔT ETHEREUM BLOCKCHAIN**

**1. Nội dung thực hành**

**-** Thiết lập một local ethereum blockchain

- Triển khai một smart contract trên ethereum

- Triển khai một Dapp tương tác với smart contract

**2. Cài đặt**

**a)** **Private Ethereum blockchain:** Sử dụng Ganache

- Mạng ảo này sẽ không có node nào và không có mining time.

- Nó cũng tạo ra 10 địa chỉ Ethereum ảo.

(Cài đặt NodeJS sau đó thực hiện các bước sau)

Bước 1: Run ***cmd***

Bước 2: Gõ lệnh sau

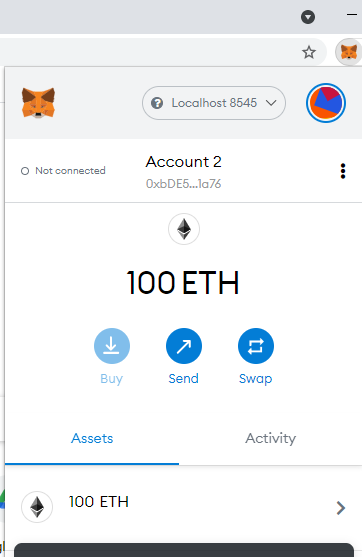
*npm install -g ganache-cli*

*ganache-cli*

Bước 3: Cài MetaMask (là một ví Ethereum (Wallet Ethereum))

*<https://chrome.google.com/webstore/detail/metamask/nkbihfbeogaeaoehlefnkodbefgpgknn>*

Bước 4: Import 1 private key từ một trong 10 account của ethereum vào MetaMask

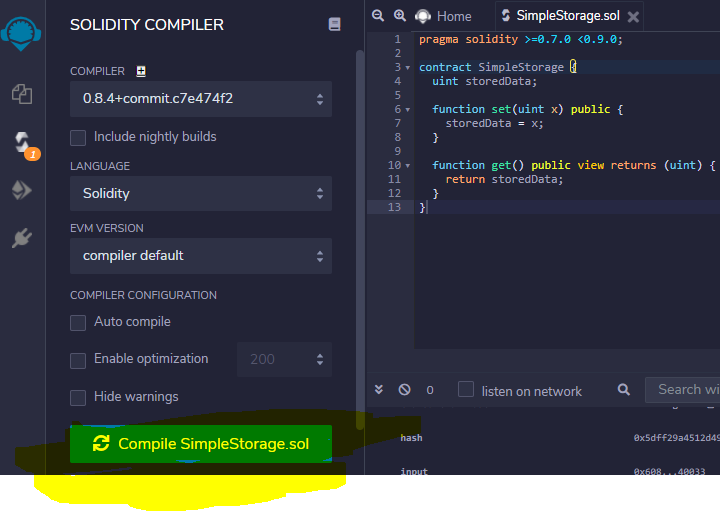


**b) Triển khai một smart contract trên Ethereum**

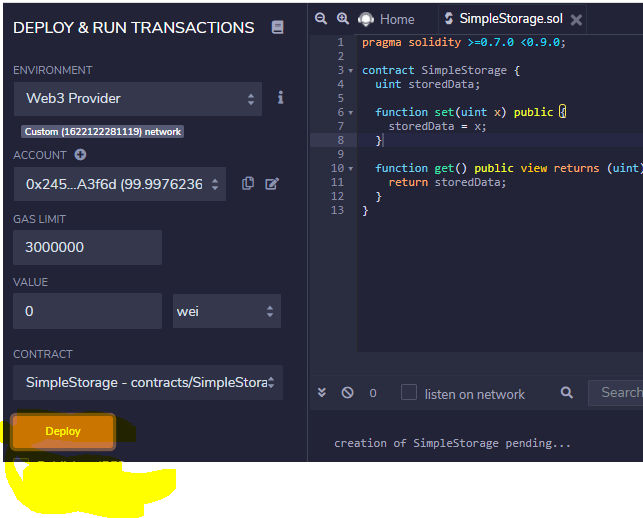
- Viết smart contract trên remix: <https://remix.ethereum.org/>

|  |
| --- |
| *pragma solidity >=0.7.0 <0.9.0;*  *contract SimpleStorage {*  *uint storedData;*  *function set(uint x) public {*  *storedData = x;*  *}*  *function get() public view returns (uint) {*  *return storedData;*  *}*  *}* |

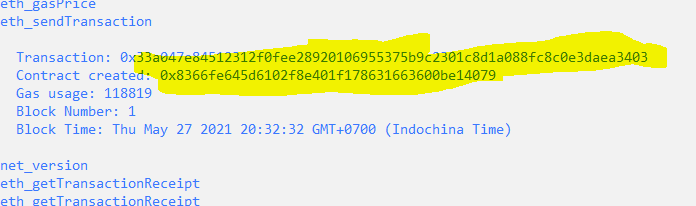
* Biên dịch



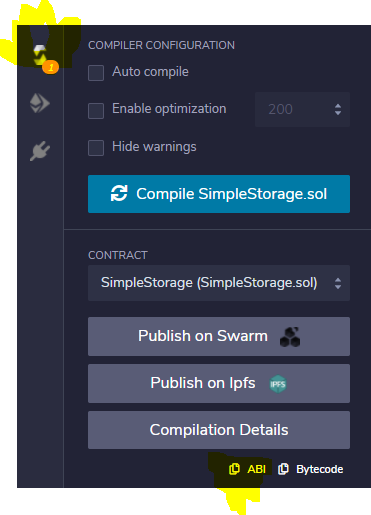
* Liên kết giữa ứng dụng với private Ethereum sử dụng thư viện Web3 trên Web Browser và Deploy smart contract trên Ethereum



* Quan sát kết quả trên ganache



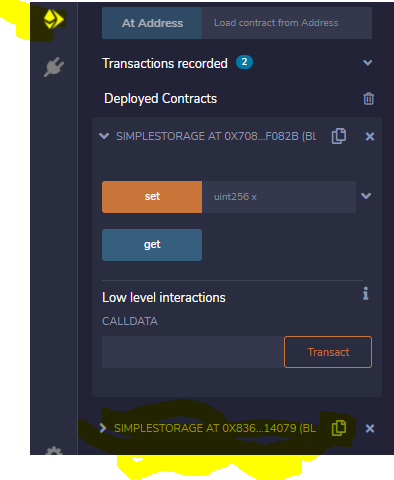
* Xem ABI của smart contract



|  |
| --- |
| [  {  "inputs": [  {  "internalType": "uint256",  "name": "x",  "type": "uint256"  }  ],  "name": "set",  "outputs": [],  "stateMutability": "nonpayable",  "type": "function"  },  {  "inputs": [],  "name": "get",  "outputs": [  {  "internalType": "uint256",  "name": "",  "type": "uint256"  }  ],  "stateMutability": "view",  "type": "function"  }  ] |

* **Địa chỉ của smart contract trên Ethereum**

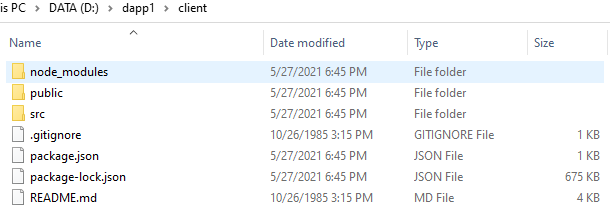
*0x8366fe645d6102F8E401F178631663600Be14079*



**c) Viết Dapp**

***- Tạo Fontend***

*npx create-react-app client*



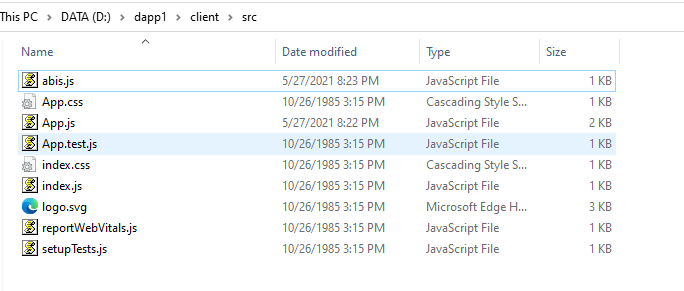
*cd client*

*npm install web3*

Chỉnh sửa file D:\dapp1\client\src\App.js

|  |
| --- |
| import logo from './logo.svg';  import './App.css';  import React, { useState } from 'react';  import Web3 from 'web3';  import { simpleStorageAbi } from './abis';  const web3 = new Web3(Web3.givenProvider);  // contract address is provided by Truffle migration  const contractAddr = '*0x8366fe645d6102F8E401F178631663600Be14079*';  const SimpleContract = new web3.eth.Contract(simpleStorageAbi, contractAddr);  function App() {  const [number, setNumber] = useState(0);  const [getNumber, setGetNumber] = useState('0x00');    const handleGet = async (e) => {  e.preventDefault();  const result = await SimpleContract.methods.get().call();  setGetNumber(result);  console.log(result);  }    const handleSet = async (e) => {  e.preventDefault();  const accounts = await window.ethereum.enable();  const account = accounts[0];  const gas = await SimpleContract.methods.set(number)  .estimateGas();  const result = await SimpleContract.methods.set(number).send({  from: account,  gas  })  console.log(result);  }  return (  <div className="App">  <header className="App-header">  <form onSubmit={handleSet}>  <label>  Set Number:  <input  type="text"  name="name"  value={number}  onChange={ e => setNumber(e.target.value) } />  </label>  <input type="submit" value="Set Number" />  </form>  <br/>  <button  onClick={handleGet}  type="button" >  Get Number  </button>  { getNumber }  </header>  </div>  );}  export default App; |

* Tạo file *abis.js*



Nội dung File abis.js chưa ABI của smart contract

|  |
| --- |
| // abis.js  export const simpleStorageAbi = [  {  "inputs": [  {  "internalType": "uint256",  "name": "x",  "type": "uint256"  }  ],  "name": "set",  "outputs": [],  "stateMutability": "nonpayable",  "type": "function"  },  {  "inputs": [],  "name": "get",  "outputs": [  {  "internalType": "uint256",  "name": "",  "type": "uint256"  }  ],  "stateMutability": "view",  "type": "function"  }  ] |

* Chạy React



