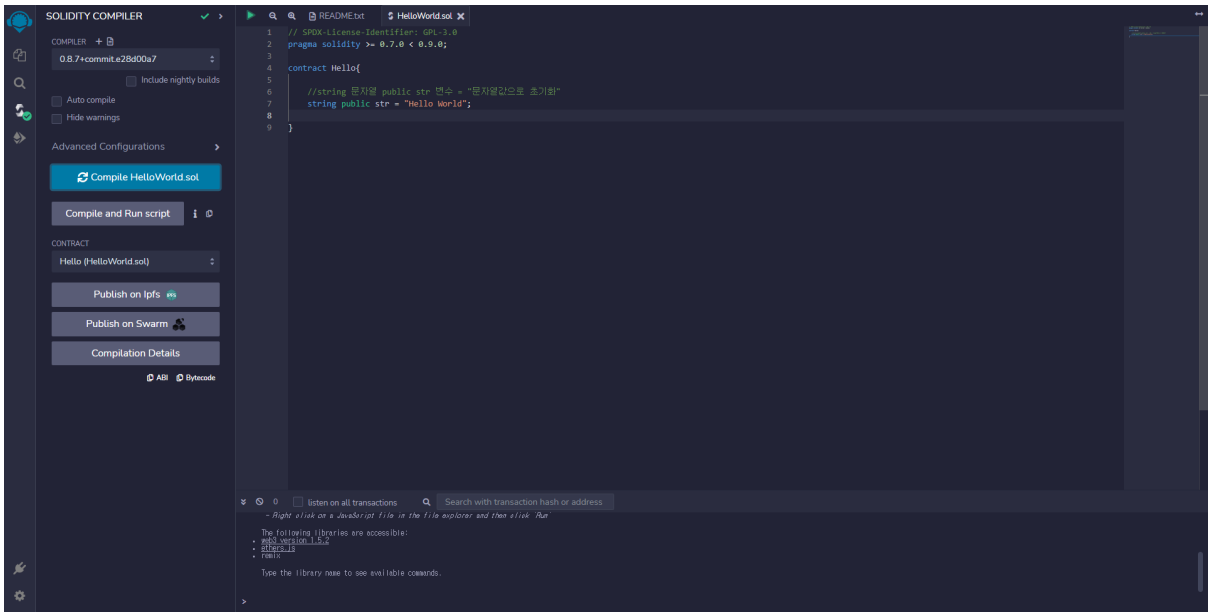
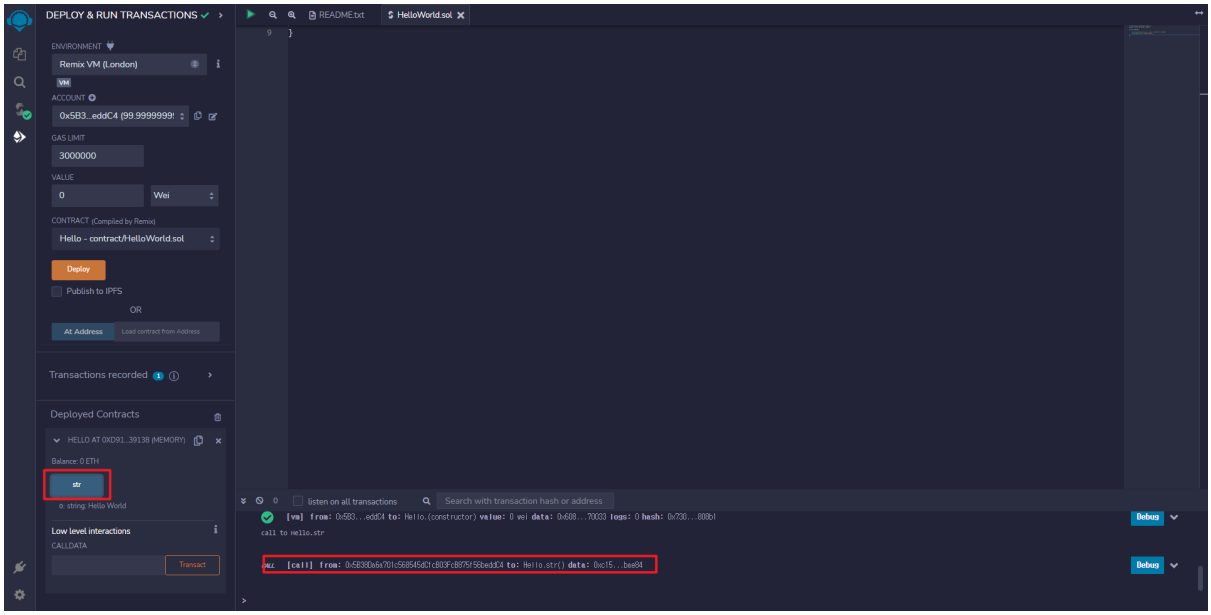
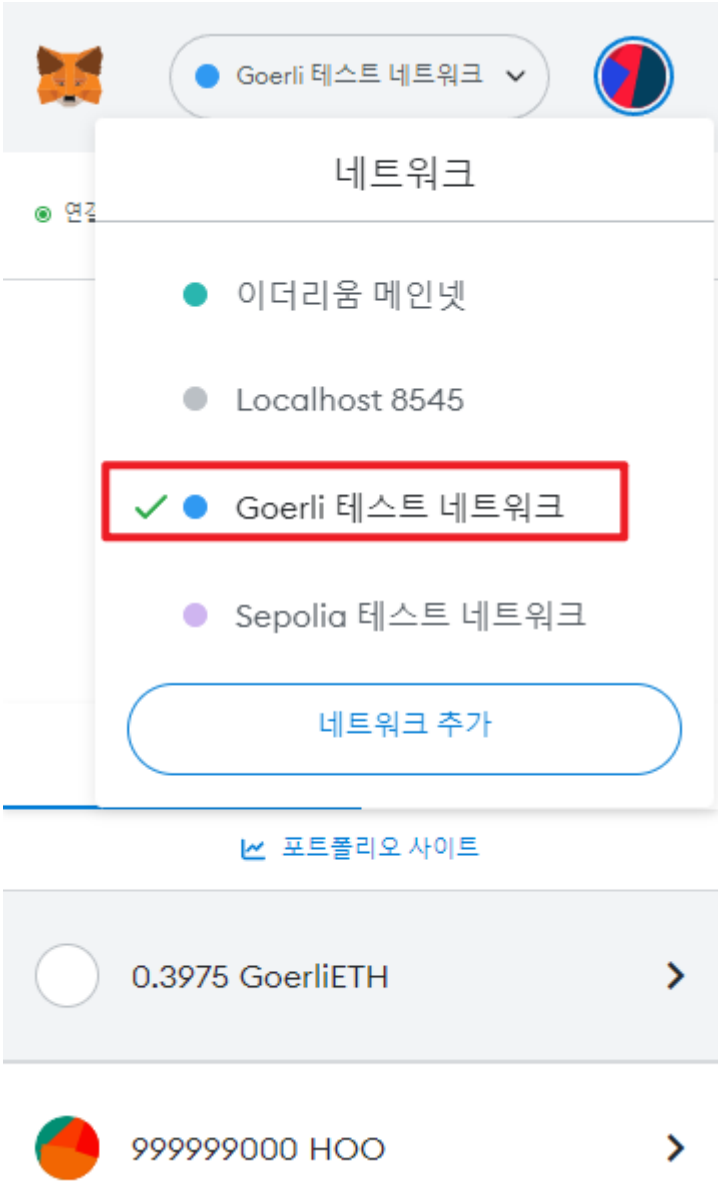
day66-sol

태그	
날짜	@2023년 1월 2일

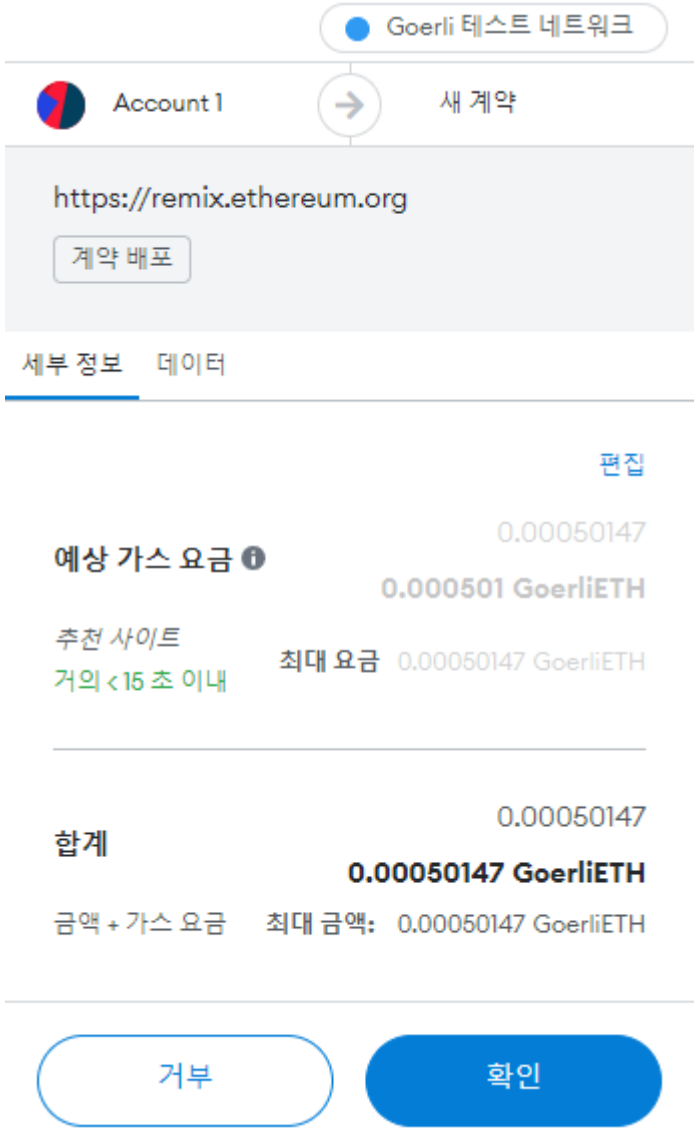
```
// SPDX-License-Identifier: GPL-3.0
pragma solidity >= 0.7.0 < 0.9.0;

contract Hello{

    //string 문자열 public str 변수 = "문자열값으로 초기화"
    string public str = "Hello World";

}
```

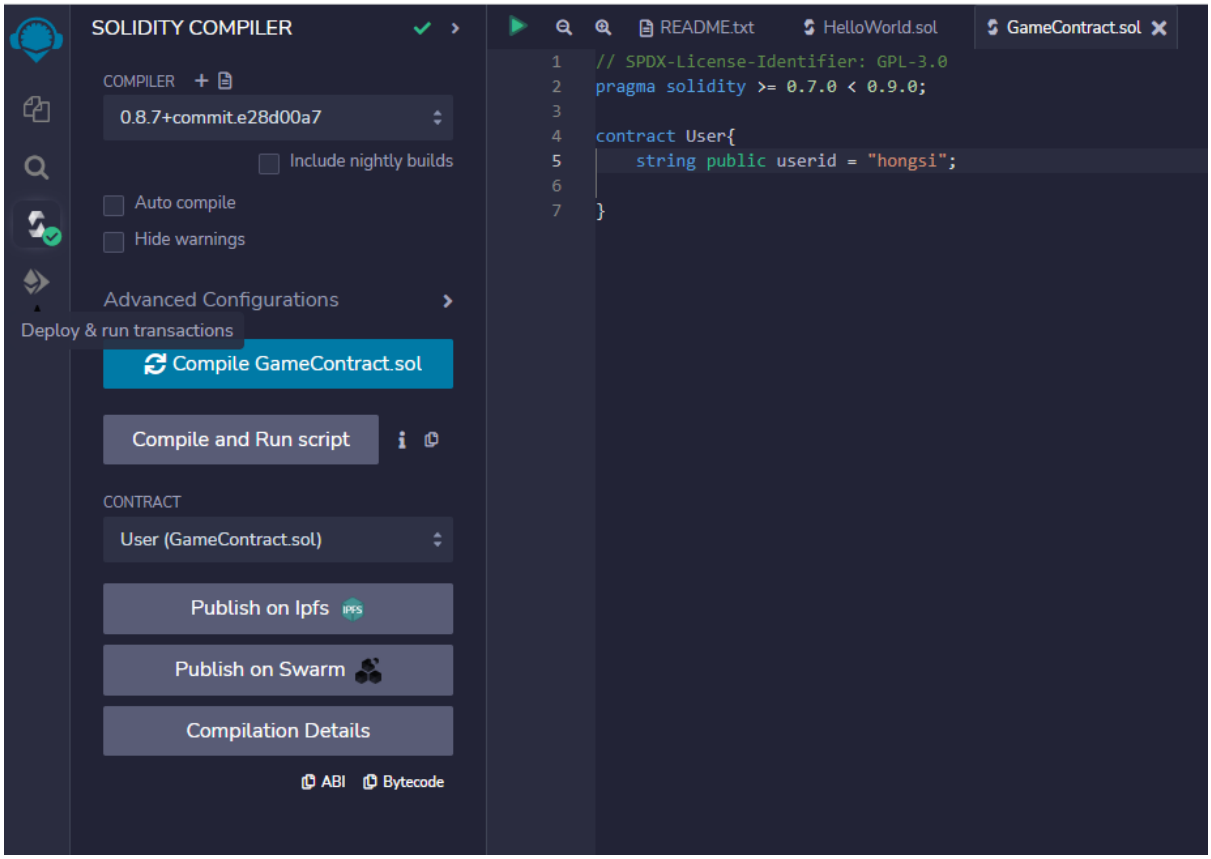



DEPLOY & RUN TRANSACTIONS
ENVIRONMENT
Remix VM (London)
ACCOUNT
GAS LIMIT
3000000
VALUE
0
Wei
CONTRACT (Compiled by Remix)
Hello - contract/HelloWorld.sol
Deploy
Publish to IPFS
OR
At Address
Transactions recorded
Deployed Contracts
Balance: 0 ETH
str
Low level interactions
CALLDATA
Transact
README.txt
HelloWorld.sol
listen on all transactions
Search with transaction hash or address
Debug

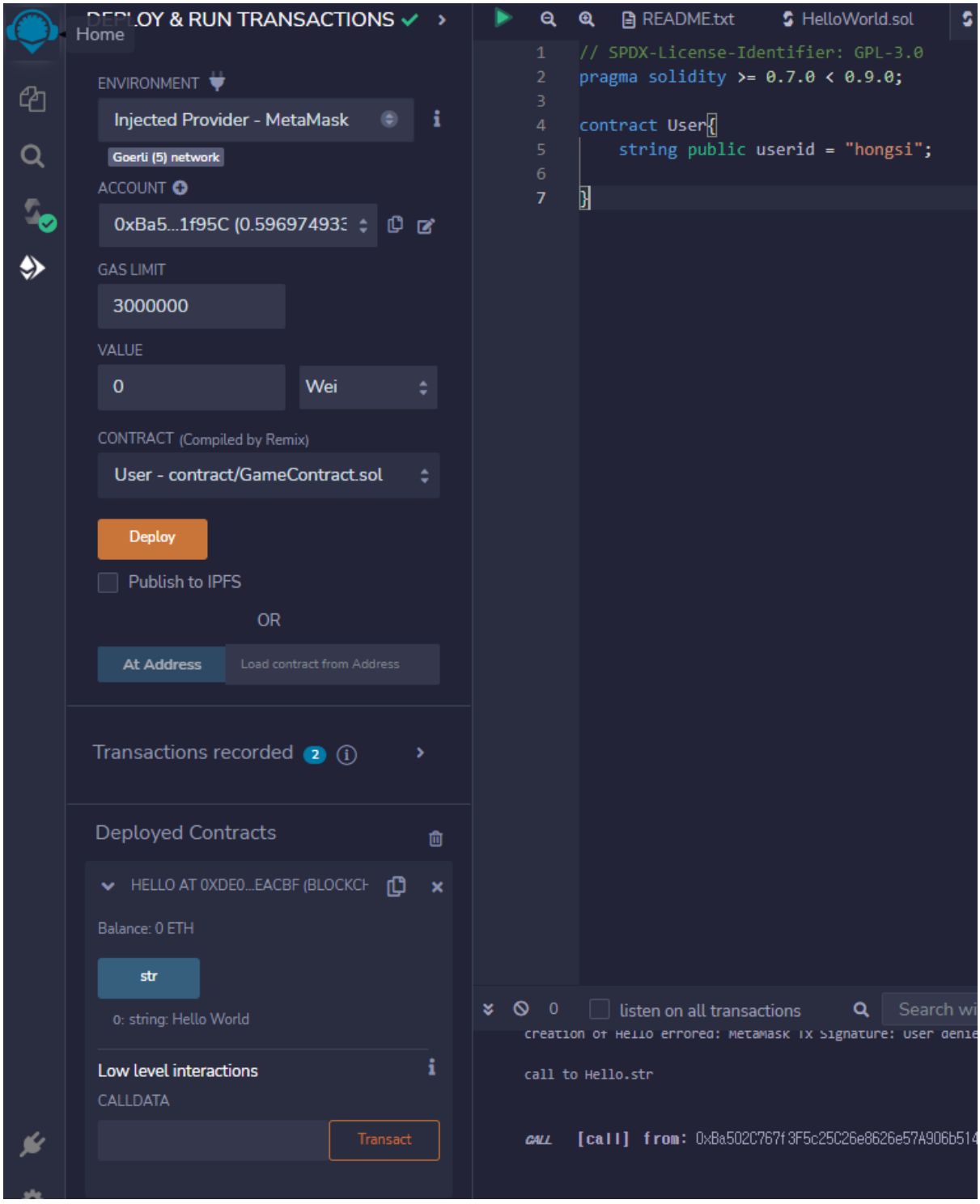

Goerli 테스트 네트워크
네트워크
이더리움 메인넷
Localhost 8545
Goerli 테스트 네트워크
Sepolia 테스트 네트워크
네트워크 추가
포트폴리오 사이트
0.3975 GoerliETH
999999000 HOO

deplay 버튼 →

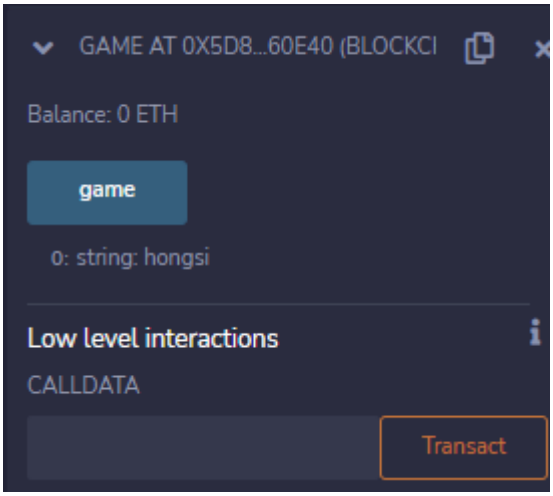

Goerli 테스트 네트워크
Account 1
새 계약
https://remix.ethereum.org
계약 배포
세부 정보
데이터
편집
예상 가스 요금
0.00050147
0.000501 GoerliETH
추천 사이트
거의 < 15 초 이내
최대 요금 0.00050147 GoerliETH
합계
0.00050147
0.00050147 GoerliETH
금액 + 가스 요금
최대 금액: 0.00050147 GoerliETH
거부
확인


SOLIDITY COMPILER
COMPILER
0.8.7+commit.e28d00a7
Include nightly builds
Auto compile
Hide warnings
Advanced Configurations
Deploy & run transactions
Compile GameContract.sol
Compile and Run script
CONTRACT
User (GameContract.sol)
Publish on Ipfs
Publish on Swarm
Compilation Details
ABI
Bytecode
README.txt
HelloWorld.sol
GameContract.sol
1 // SPDX-License-Identifier: GPL-3.0
2 pragma solidity >= 0.7.0 < 0.9.0;
3
4 contract User{
5 string public userid = "hongsi";
6
7 }

코드 작성 → 컴파일 → 네번째

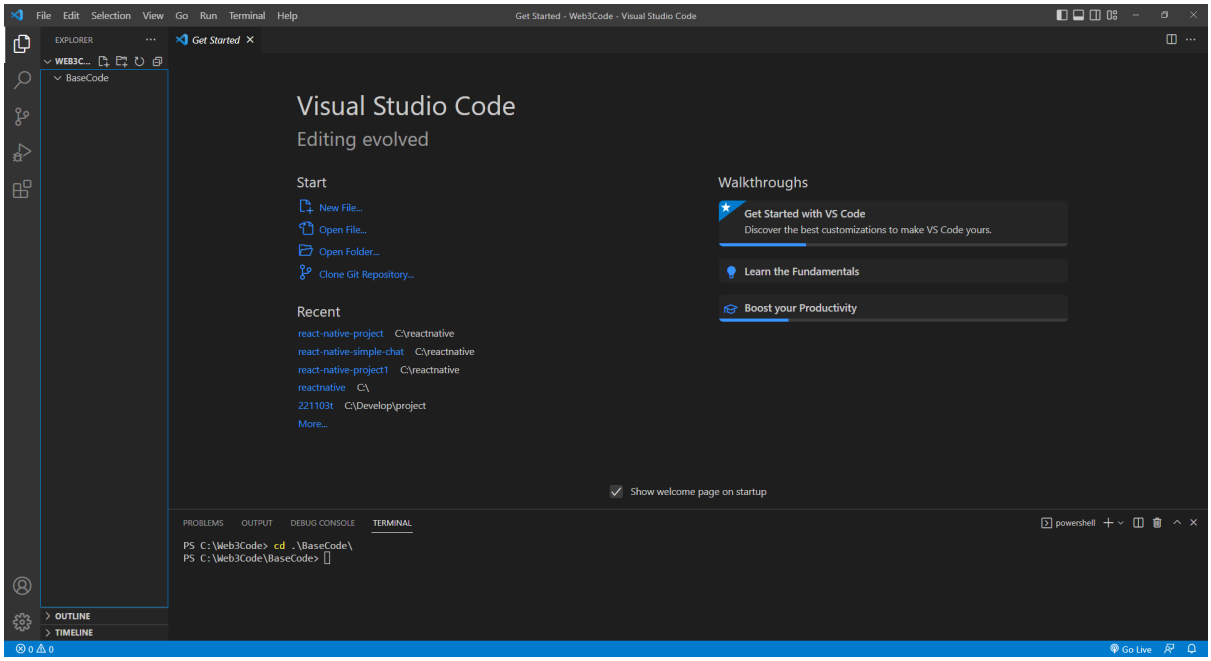


메타마스크 설정 확인 후 deploy

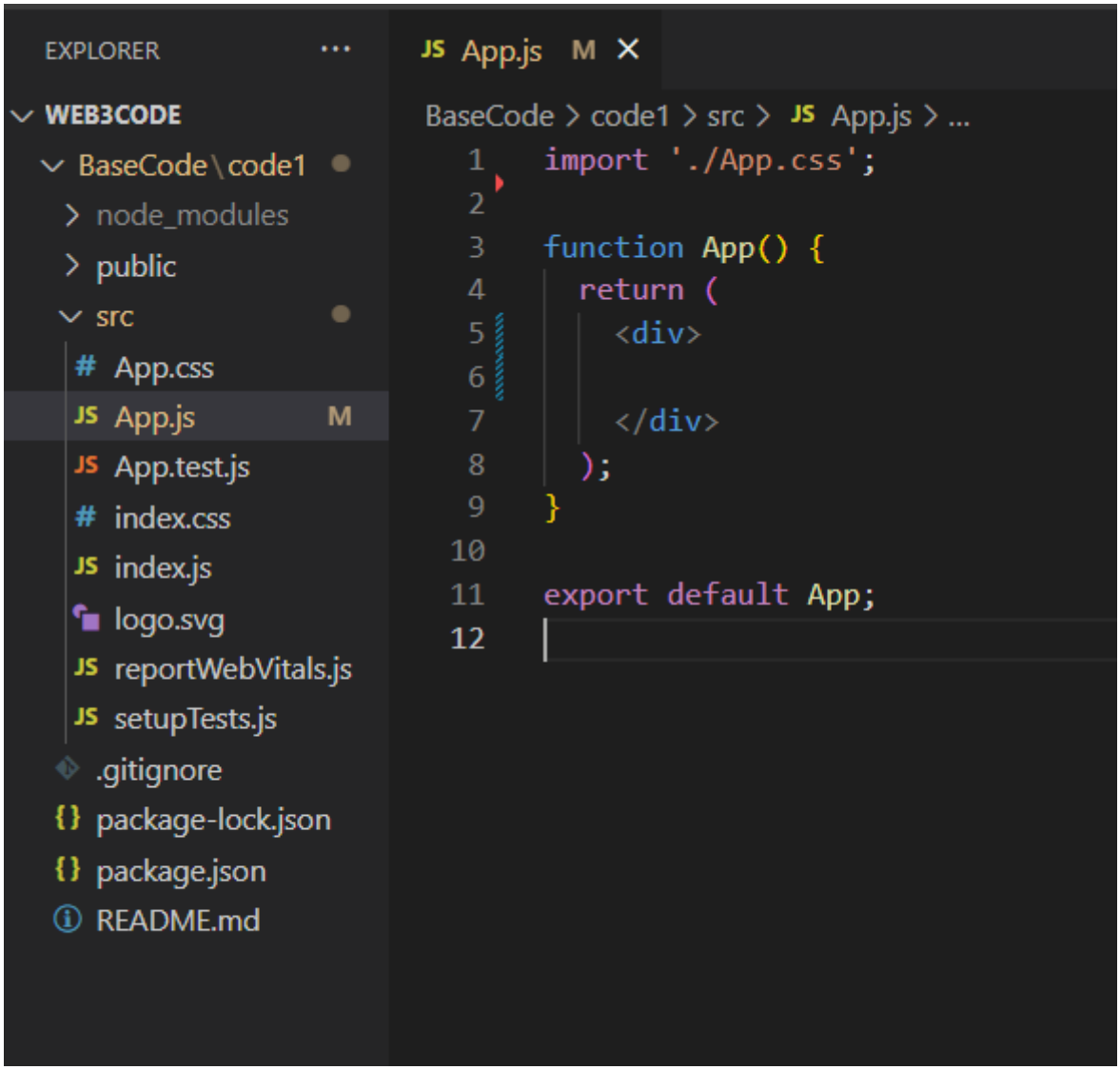


블록체인 트랜젝션 → 리액트 연결

web3로 사용할 폴더 생성 → vscode 열기 →

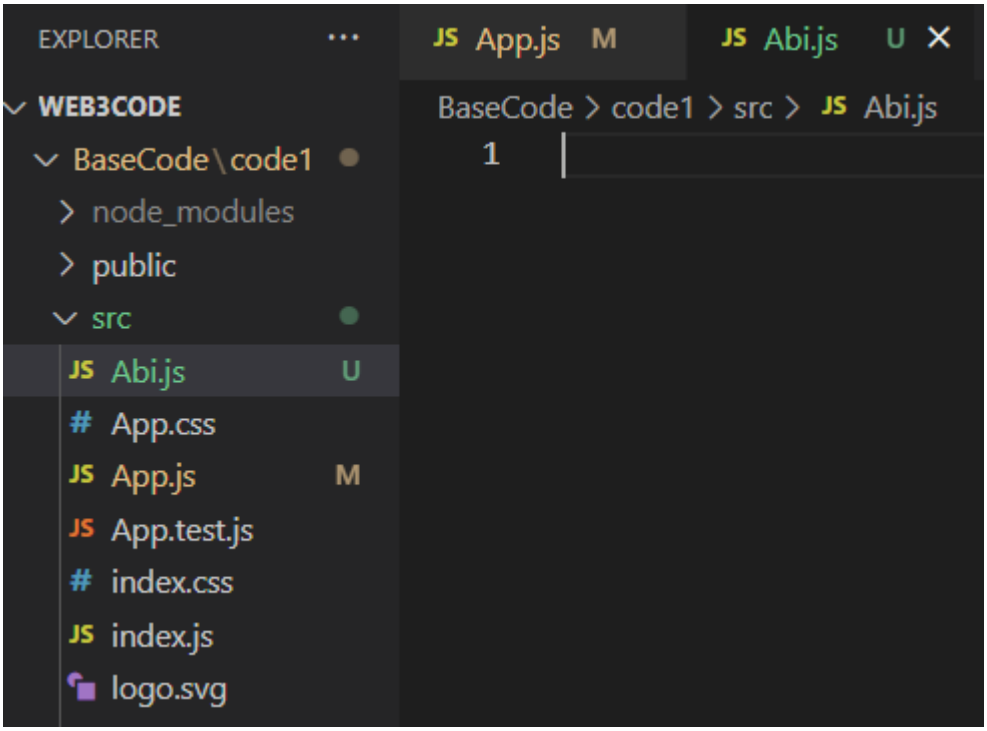


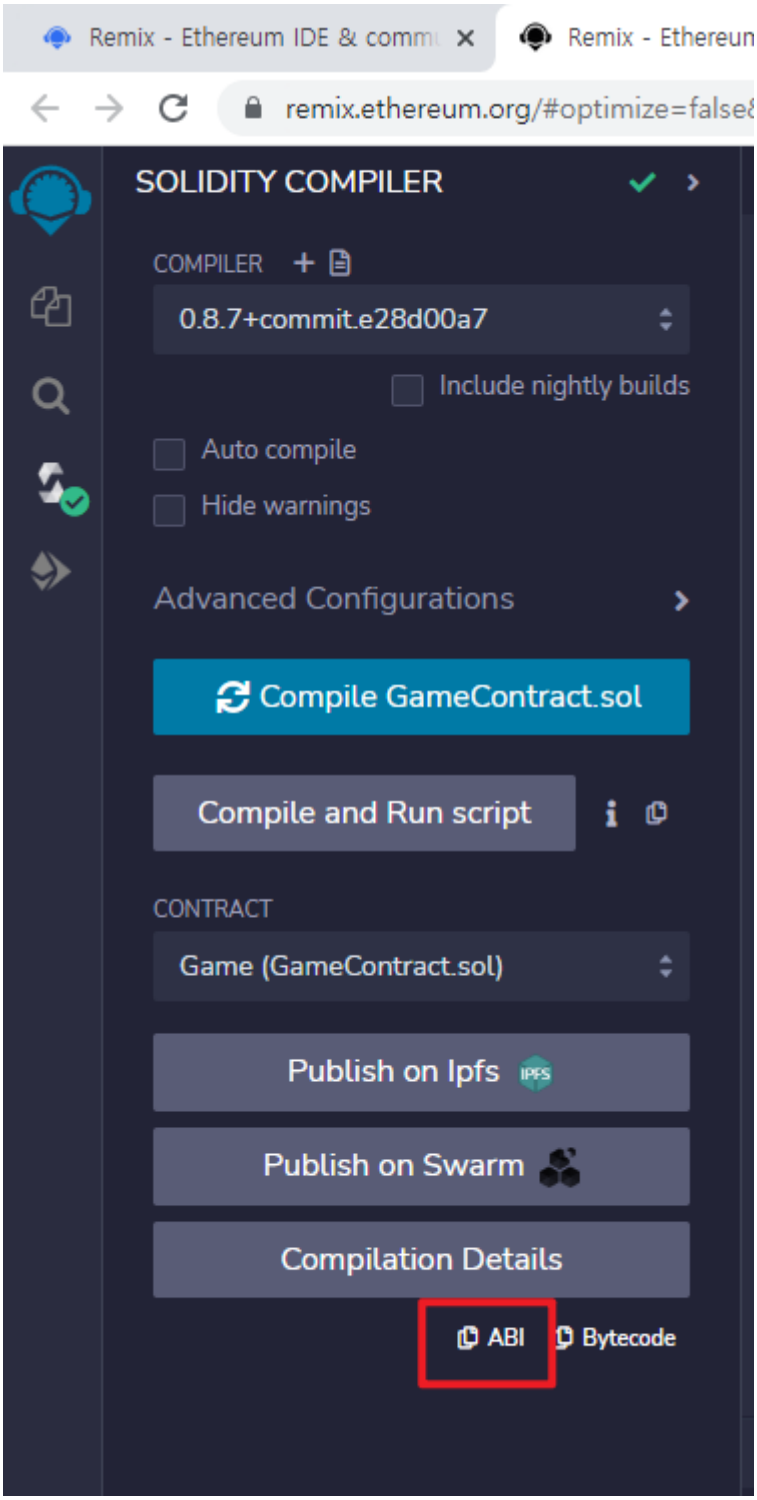
- npx create-react-app code1



- npm install web3

※web3 연결 주의점 abi가져오기


EXPLORER
App.js M
Abi.js U
WEB3CODE
BaseCode > code1 > src > Abi.js
BaseCode\code1
1
node_modules
public
src
Abi.js U
App.css
App.js M
App.test.js
index.css
index.js
logo.svg


Remix - Ethereum IDE & comm
Remix - Ethereun
remix.ethereum.org/#optimize=false
SOLIDITY COMPILER
COMPILER
0.8.7+commit.e28d00a7
Include nightly builds
Auto compile
Hide warnings
Advanced Configurations
Compile GameContract.sol
Compile and Run script
CONTRACT
Game (GameContract.sol)
Publish on Ipfs
Publish on Swarm
Compilation Details
ABI
Bytecode

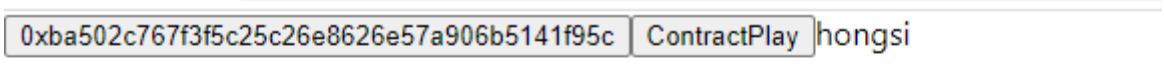
JS App.js M JS Abi.js U

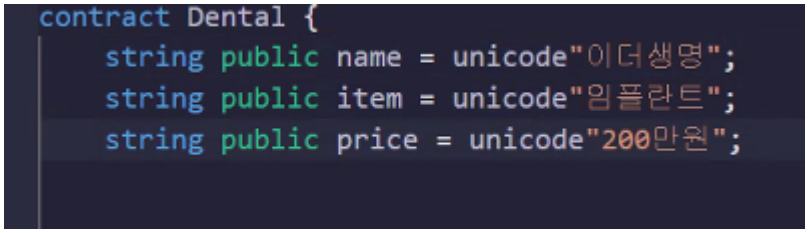
BaseCode > code1 > src > JS Abi.js > ...

```
const Abi = [
    {
        "inputs": [],
        "name": "game",
        "outputs": [
            {
                "internalType": "string",
                "name": "",
                "type": "string"
            }
        ],
        "stateMutability": "view",
        "type": "function"
    }
]

export default Abi;
```

```
import Web3 from 'web3';
import React,{useState, useEffect} from 'react';
import './App.css';
import Abi from './Abi';


function App() {
  const [web3, setWeb3] = useState();
  const [account, setAccount] = useState();
  const [pressStart, setPressStart] = useState();
  const [game, setGame] = useState();

  useEffect(()=> {
    if(typeof window.ethereum != "undefined") {
      try{
        const web = new Web3(window.ethereum);
        setWeb3(web);
      }catch(err){
        console.log(err);
      }
    }
```

```
  },[]);

  //메타마스크로부터 계정을 연결, 계정 주소를 저장
  const connectWallet = async()=>{
    const accounts = await window.ethereum.request({
      method : "eth_requestAccounts",
    })
    setAccount(accounts[0]);
  }
  // 컨트렉트 실행 call get / set
  const ContractPlay = async()=>{
    //컨트렉트 주소
    const ContractAddress = '0x5d899c6680432CdcF7814C30003E04D9d4960E40';
    //컨트렉트
    const Contract = await new web3.eth.Contract(Abi, ContractAddress);

    const game = await Contract.methods.game().call();
    setGame(game);
  }

  return (
    <div>
      <button
        onClick={()=>{
          connectWallet();
          setPressStart(true);
        }}
      >
        {pressStart ? account : "Connect Wallet"}
      </button>

      <button onClick = {ContractPlay}>ContractPlay</button>
      {game}

    </div>
  );
}

export default App;
```





한글 사용 : unicode

---

1. 불린 boolean type

true / false

1 0

2. 문자열 “”;
3. 정수타입 Integer

int - +

uint 양수 +

int : 기호있는 integer

```
bool public b1 = !false; // true
bool public b2 = false || true; // true
bool public b3 = false == true; // false
bool public b4 = false != false; //false
bool public b5 = flase && true; // false

//byte 1~32
bytes4 public bt = 0x12345678;
bytes public bt2 = "String";

//address type
address public addr = 0xd123123123451fd24511123D2541E25E11231ba;

int8 public value = 1;
uint8 public value2 = 123;
uint256 public value3 = 123;
```

솔리디티 함수

- 기본형태

function 이름() publice }

//내용

}