

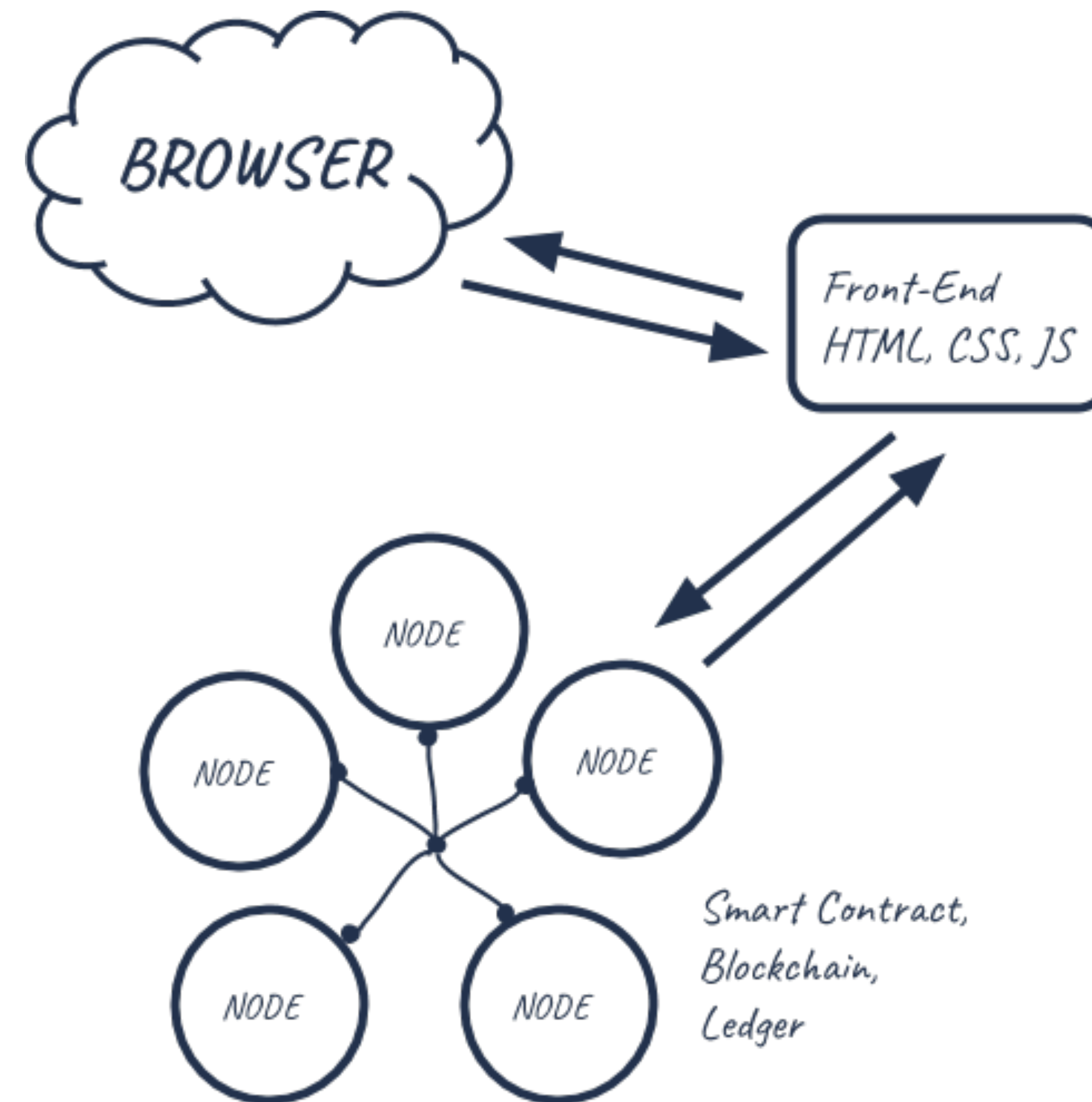


IMD0293

DAPPS

ÐApps

backend distribuído + frontend





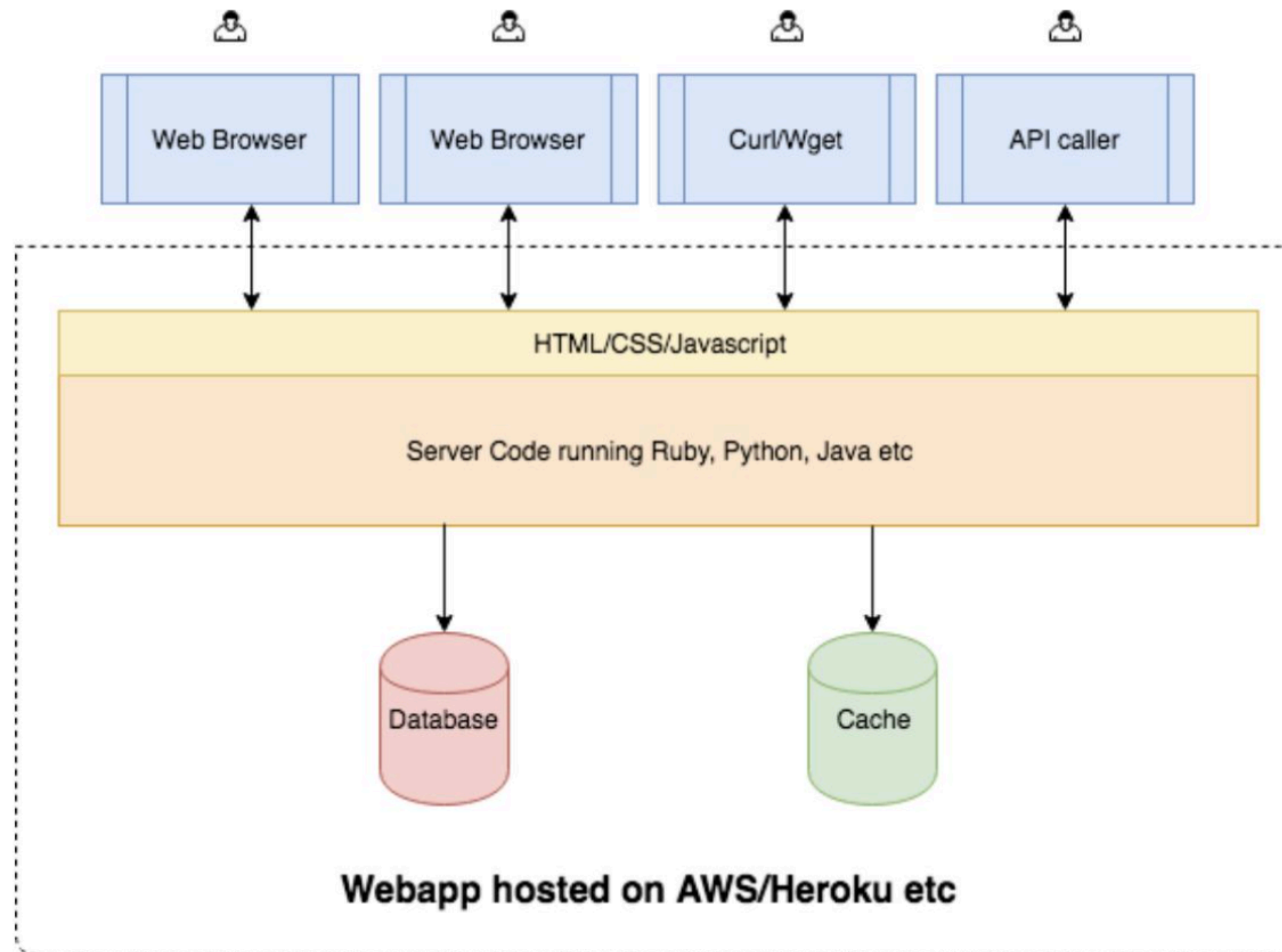
web3.js

Ethereum JavaScript API

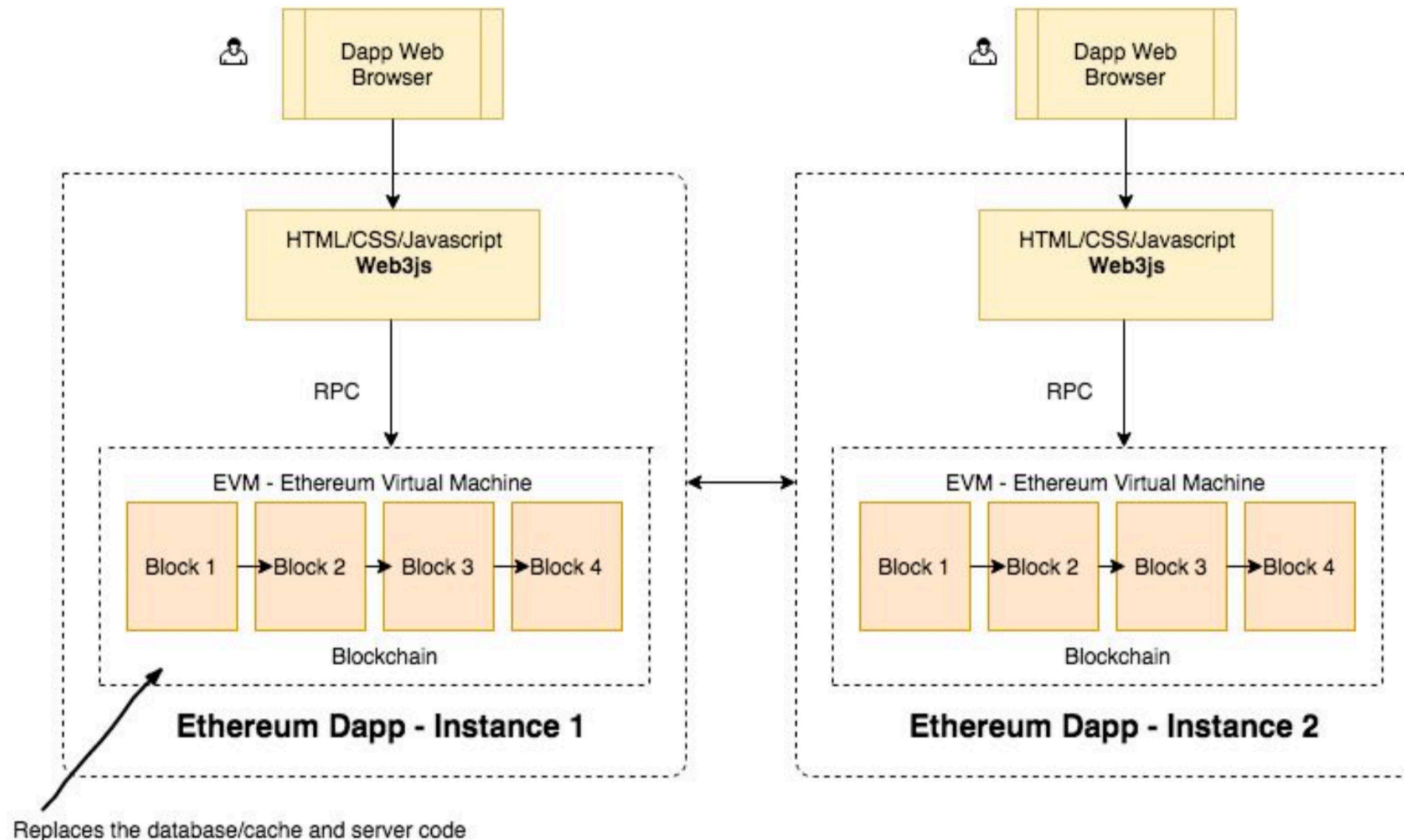
Coleção de bibliotecas que permitem um usuário **interagir** com a plataforma Ethereum e com *Smart Contracts*

<https://web3js.readthedocs.io>

WEB CONVENCIONAL



ONDE ESTAMOS NA PILHA?



com o que o usuário interage

o que transforma a interação do usuário em interações com o “software” e recebe os resultados

a lógica de negócio (*smart contracts*)

WEB3

- ▶ **web3.js** é o ponto de entrada do blockchain Ethereum do lado cliente de um DApp
- ▶ Comunica com os nós na rede utilizando RPC
- ▶ **web3** contém:
 - ▶ **web3.eth** : para interação com o blockchain do Ethereum

ETHEREUM ABI: EXPONDO OS MÉTODOS DO CONTRATO

- ▶ ABI: Application Binary Interface
- ▶ Uma ABI é como você pode chamar funções de um contrato e obter os dados de retorno
 - ▶ Determina como funções são chamadas e em que formato a informação deve ser passada de um componente do programa para outro
- ▶ Por que é necessário?
 - ▶ Você precisa de um jeito para especificar qual função será invocada do contrato e também garantir o tipo de dado que é retornado
- ▶ Não é parte do protocolo Ethereum, você pode definir seu próprio ABI - porém é mais fácil obedecer o formato definido pelo web3.js

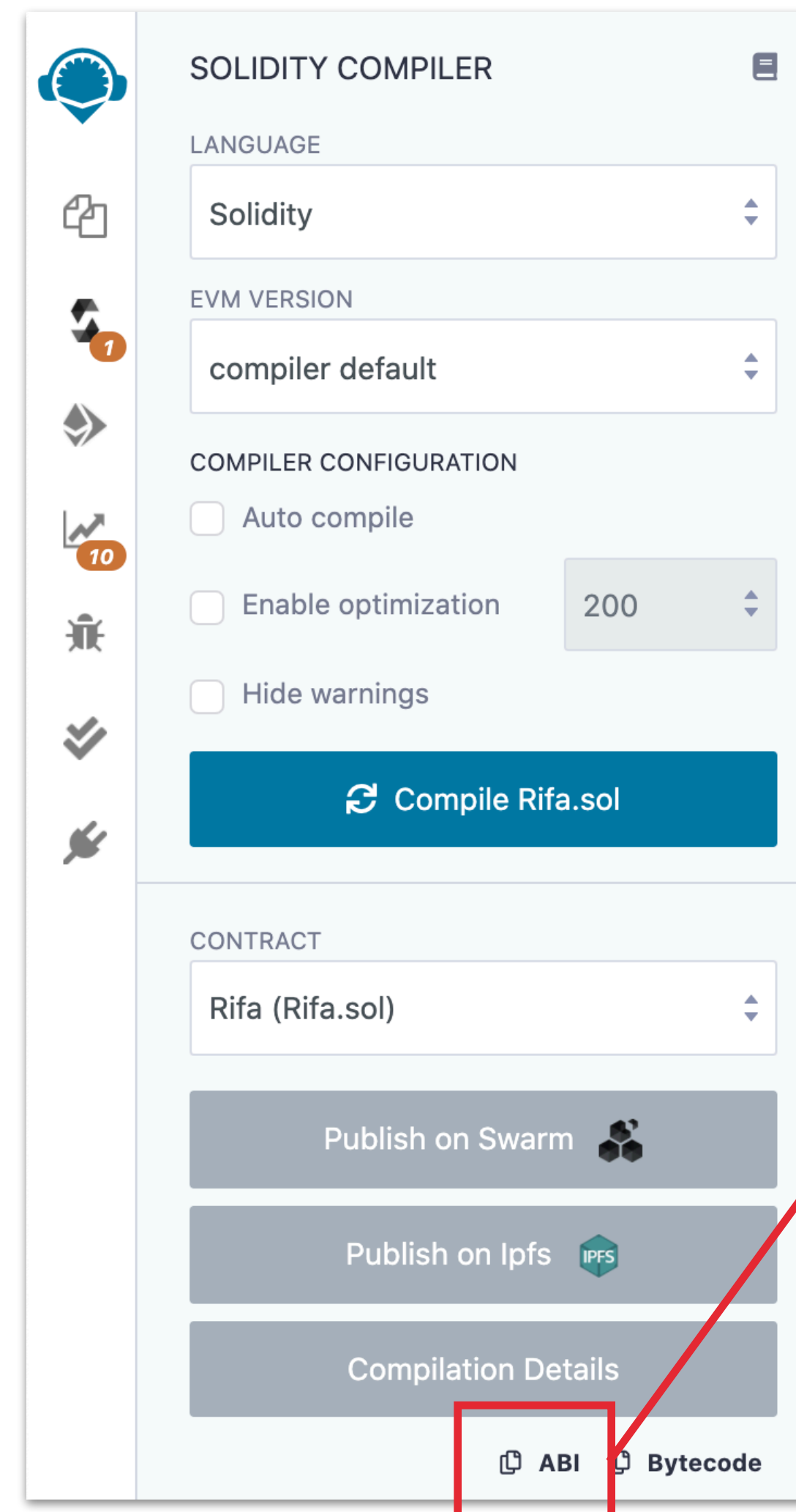
ETHEREUM ABI: EXPONDO OS MÉTODOS DO CONTRATO

```
contract Test {  
    uint a;  
    address d = 0x12345678901234567890123456789012;  
  
    constructor(uint testInt) { a = testInt;}  
  
    event Event(uint indexed b, bytes32 c);  
    event Event2(uint indexed b, bytes32 c);  
  
    function foo(uint b, bytes32 c) returns(address) {  
        Event(b, c);  
        return d;  
    }  
}
```

ETHEREUM ABI: EXPONDO OS MÉTODOS DO CONTRATO

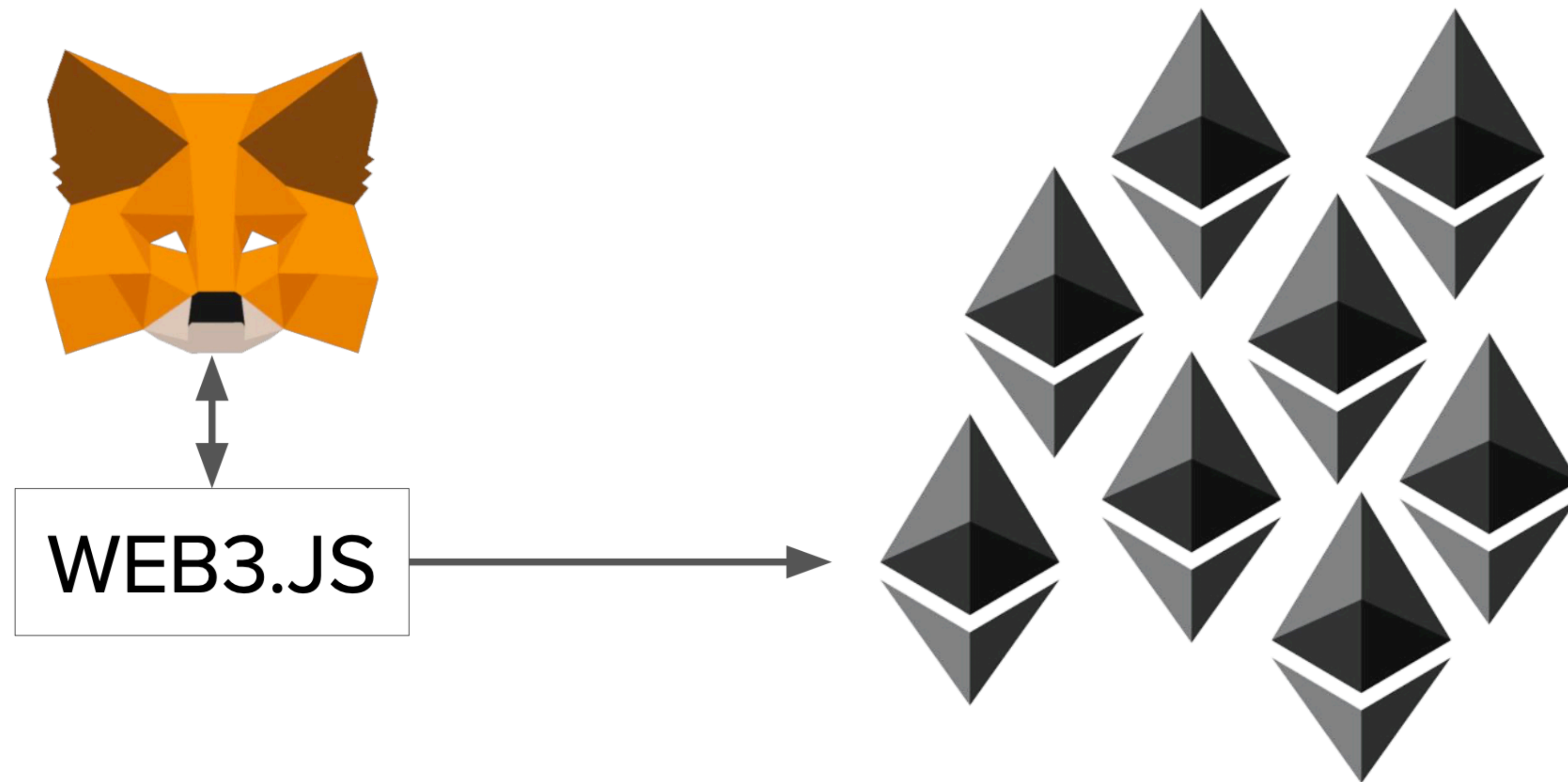
```
[{
  "type": "constructor",
  "payable": false,
  "stateMutability": "nonpayable",
  "inputs": [{"name": "testInt", "type": "uint256"}],
}, {
  "type": "function",
  "name": "foo",
  "constant": false,
  "payable": false,
  "stateMutability": "nonpayable",
  "inputs": [{"name": "b", "type": "uint256"}, {"name": "c", "type": "bytes32"}],
  "outputs": [{"name": "", "type": "address"}]
}, {
  "type": "event",
  "name": "Event",
  "inputs": [{"indexed": true, "name": "b", "type": "uint256"}, {"indexed": false, "name": "c", "type": "bytes32"}],
  "anonymous": false
}, {
  "type": "event",
  "name": "Event2",
  "inputs": [{"indexed": true, "name": "b", "type": "uint256"}, {"indexed": false, "name": "c", "type": "bytes32"}],
  "anonymous": false
}]
```

ABI NO REMIX



```
{
  "inputs": [
    {
      "internalType": "uint256",
      "name": "_quant",
      "type": "uint256"
    }
  ],
  "name": "comprarRifa",
  "outputs": [],
  "stateMutability": "payable",
  "type": "function"
},
{
  "inputs": [],
  "stateMutability": "nonpayable",
  "type": "constructor"
},
{
  "anonymous": false,
  "inputs": [
    {
      "indexed": false,
      "internalType": "address",
      "name": "comprador",
      "type": "address"
    },
    {
      "indexed": false,
      "internalType": "uint256",
      "name": "quant",
      "type": "uint256"
    }
  ],
  "name": "RifaComprada",
  "type": "event"
},
{
  "inputs": [],
  "name": "sacarPremio",
  "outputs": [],
  "stateMutability": "nonpayable",
  "type": "function"
},
{
  "inputs": [],
  "name": "sortearRifa",
  "outputs": [],
  "stateMutability": "nonpayable",
  "type": "function"
}
```

JUNTANDO TUDO...



WEB3.JS

```
1 <!DOCTYPE html>
2 <html lang="en">
3
4 <head>
5     <meta charset="UTF-8">
6     <title>Dapp front-end</title>
7     <script language="javascript" type="text/javascript" src="web3.min.js"></script>
8     <script language="javascript" type="text/javascript" src="abi.js"></script>
9
10 </head>
11
12 <body>
13     <script>
14         window.addEventListener('load', function () {
15             var web3;
16             var account;
17
18             if (typeof window.ethereum !== "undefined") {
19                 try {
20                     const accounts = await window.ethereum.request({
21                         method: "eth_requestAccounts", // Requisita primeiro acesso ao Metamask
22                     });
23                     account = accounts[0];
24                 } catch (error) {
25                     console.error("Usuário negou acesso ao web3!");
26                 }
27                 web3 = new Web3(window.ethereum);
28             } else {
29                 console.error("Instalar MetaMask!");
30             }
31
32             startApp();
33         });
34     </script>
35 </body>
36 </html>
```

ABI

```
1 var abi = [  
2   {  
3     "constant": false,  
4     "inputs": [],  
5     "name": "sortearRifa",  
6     "outputs": [],  
7     "payable": false,  
8     "stateMutability": "nonpayable",  
9     "type": "function"  
10  },  
11  {  
12    "constant": true,  
13    "inputs": [],  
14    "name": "verGanhador",  
15    "outputs": [  
16      {  
17        "name": "",  
18        "type": "address"  
19      }  
20    ],  
21    "payable": false,  
22    "stateMutability": "view",  
23    "type": "function"  
24  },  
25  {  
26    "constant": true,  
27    "inputs": [],  
28    "name": "verTotalDeRifas",  
29    "outputs": [  
30      {  
31        "name": "",  
32        "type": "uint256"  
33      }  
34    ],  
35    "payable": false,  
    "stateMutability": "view",  
    "type": "function"  
  },  
]
```

CALL E SEND

```
1 var enderecoContrato = "0x38996ACE62d2a5C147e04f0bbB36a518be8aa164";
2 var contrato = null;
3
4 function startApp() {
5     contrato = new web3.eth.Contract(abi, enderecoContrato);
6
7     web3.eth.getAccounts().then(function (result) {
8         userAccount = result[0]; // Tudo com web3.js retorna Promises!
9     });
10 }
11
12 function verTotalDeRifas() { // function verTotalDeRifas() public view returns (uint)
13     contrato.methods.verTotalDeRifas().call().then((resposta) => { /* o que você quiser fazer! */ });
14 }
15
16 function comprarRifa() { // function comprarRifa(uint _quant) public payable
17     let quant = document.getElementById("quantidade").value;
18     let preco = 1000000000000000000 * quant;
19     return contrato.methods.comprarRifa(quant).send({ from: account, value: preco }).then(atualizaInterface);
20 }
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
```

Rifas IMD

Exemplo de front-end para um smart contract da rede Ethereum.

Rifa IMD

Preço da Rifa: 0.1 ETH

Minhas Rifas 0x8cb32fec81882d046b95d9e761fc09931e2e8f7b	1
Total de Rifas Total de rifas vendidas	4
Prêmio do Vencedor em ether	0.2 ETH
Ganhador	0x00

Comprar rifas!

COMPRAS DE RIFAS

Endereço	Quantidade	Transação
0x8125A7c0b4B8923961a4Cd7583c5Fff8008ebD67	2	0xba04f3e035a3c41a1e7d046388335f0a6e35a9e9f63ef4c0b482ed41ab1422f5
0x8125A7c0b4B8923961a4Cd7583c5Fff8008ebD67	1	0x6c2605d28136362e68f6ecba8a8b9f29b41f924c213fd1b5c5e1577129f0444e
0x8cb32fEc81882D046b95D9e761fC09931e2E8F7b	1	0xc2611526c389d900922e8c1ba753f49e06d0ec4cb88af12730c547ba6dea2dc4

Rifas IMD

Exemplo de front-end para um smart contract da rede Ethereum.

Rifa IMD

Preço da Rifa: 0.1 ETH

Minhas Rifas	1
0x8cb32fec81882d046b95d9e761fc09931e2e8f7b	
Total de Rifas	4
Total de rifas vendidas	
Prêmio do Vencedor	0.2 ETH
em ether	
Ganhador	0x00

Comprar rifas!

COMPRAS DE RIFAS

Endereço	Quantidade	Transação
0x8125A7c0b4B8923961a4Cd7583c5Fff8008ebD67	2	0xba04f3e035a3c41a1e7d046388335f0a6e35a9e9f63ef4c0b482ed41ab1422f5
0x8125A7c0b4B8923961a4Cd7583c5Fff8008ebD67	1	0x6c2605d28136362e68f6ecba8a8b9f29b41f924c213fd1b5c5e1577129f0444e
0x8cb32fEc81882D046b95D9e761fC09931e2E8F7b	1	0xc2611526c389d900922e8c1ba753f49e06d0ec4cb88af12730c547ba6dea2dc4

```
function verTotalDeRifas() {
  contrato.methods.verTotalDeRifas()
    .call()
    .then((resultado) => { /* */ });
}
```

JAVASCRIPT - FRONTEND

```
function verTotalDeRifas() public view returns (uint) {
  return rifas.length;
}
```

SOLIDITY - EVM - BLOCKCHAIN - BACKEND

Rifas IMD

Exemplo de front-end para um smart contract da rede Ethereum.

Rifa IMD

Preço da Rifa: 0.1 ETH

Minhas Rifas	1
0x8cb32fec81882d046b95d9e761fc09931e2e8f7b	
Total de Rifas	4
Total de rifas vendidas	
Prêmio do Vencedor	0.2 ETH
em ether	
Ganhador	0x00

Comprar rifas!

COMPRAS DE RIFAS

Endereço	Quantidade	Transação
0x8125A7c0b4B8923961a4Cd7583c5Fff8008ebD67	2	0xba04f3e035a3c41a1e7d046388335f0a6e35a9e9f63ef4c0b482ed41ab1422f5
0x8125A7c0b4B8923961a4Cd7583c5Fff8008ebD67	1	0x6c2605d28136362e68f6ecba8a8b9f29b41f924c213fd1b5c5e1577129f0444e
0x8cb32fEc81882D046b95D9e761fC09931e2E8F7b	1	0xc2611526c389d900922e8c1ba753f49e06d0ec4cb88af12730c547ba6dea2dc4

```
function comprarRifa() {
  let quant = document.getElementById("quantidade").value;
  let preco = 100000000000000000 * quant;
  contrato.methods.comprarRifa(quant)
    .send({ from: account, value: preco })
    .then(atualizaInterface);
}
```

JAVASCRIPT - FRONTEND

```
function comprarRifa(uint _quant) public payable {
  require(msg.value == _quant*valorDaRifa);

  for (uint i = 0; i < _quant; i++) {
    rifas.push(msg.sender);
    rifasPorPessoa[msg.sender]++;
  }

  emit RifaComprada(msg.sender, _quant);
}
```

SOLIDITY - EVM - BLOCKCHAIN - BACKEND

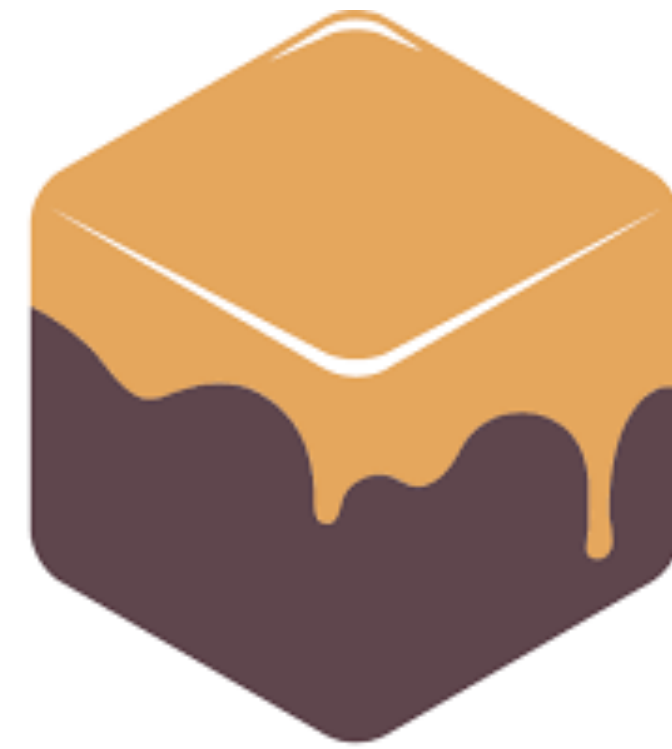


TRUFFLE

OPCIONAL

Framework para desenvolvimento de *Smart Contracts*:

1. Compilação e implantação (*deploy*)
2. Interação com contratos via console
3. Testes automatizados



Ganache

OPCIONAL

Blockchain Ethereum privado

Rodar testes

Executar comandos

Inspecionar estados