

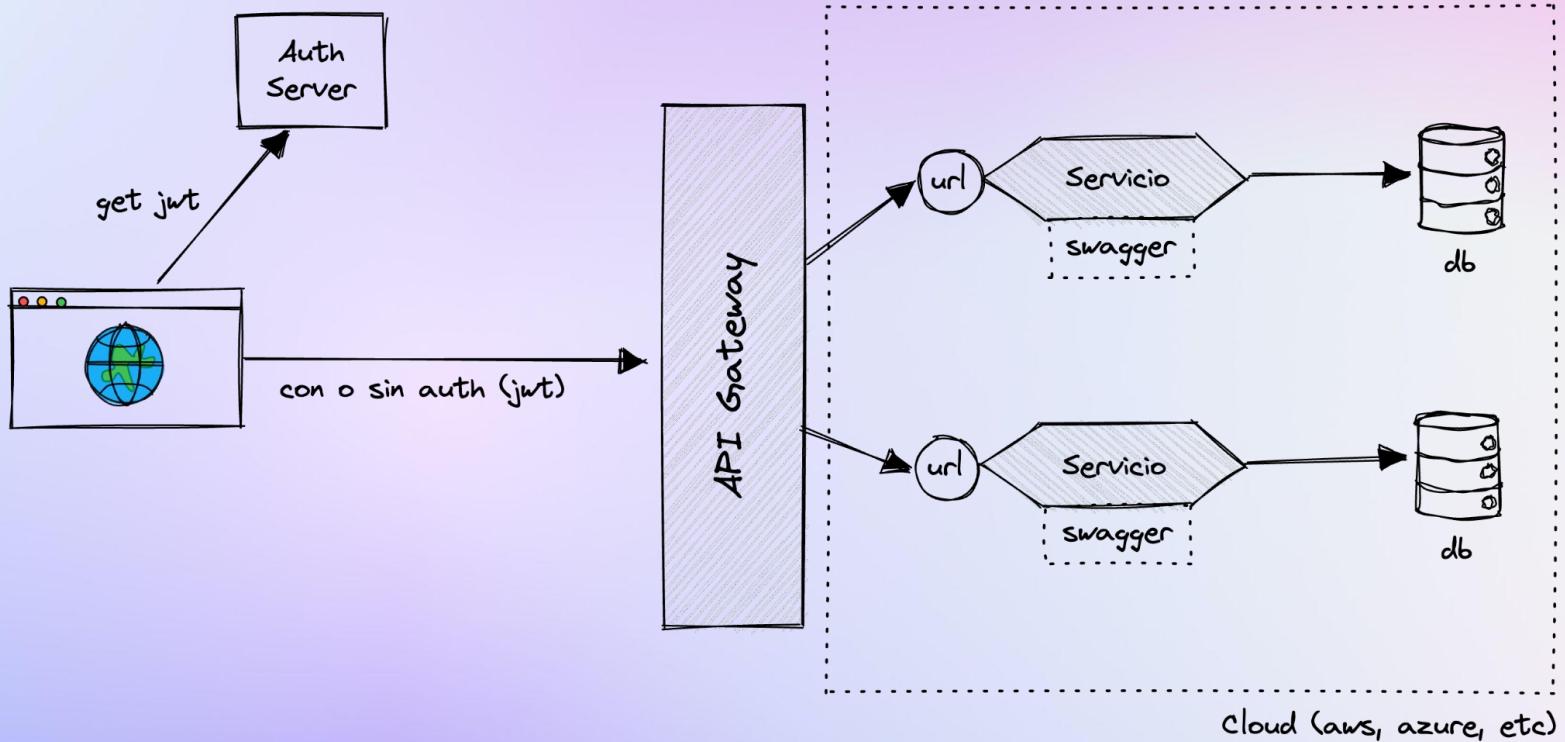
Intro al desarrollo frontend para DApps con React

⌚ UNDERSCOPE

React Buenos Aires | Octubre 2022

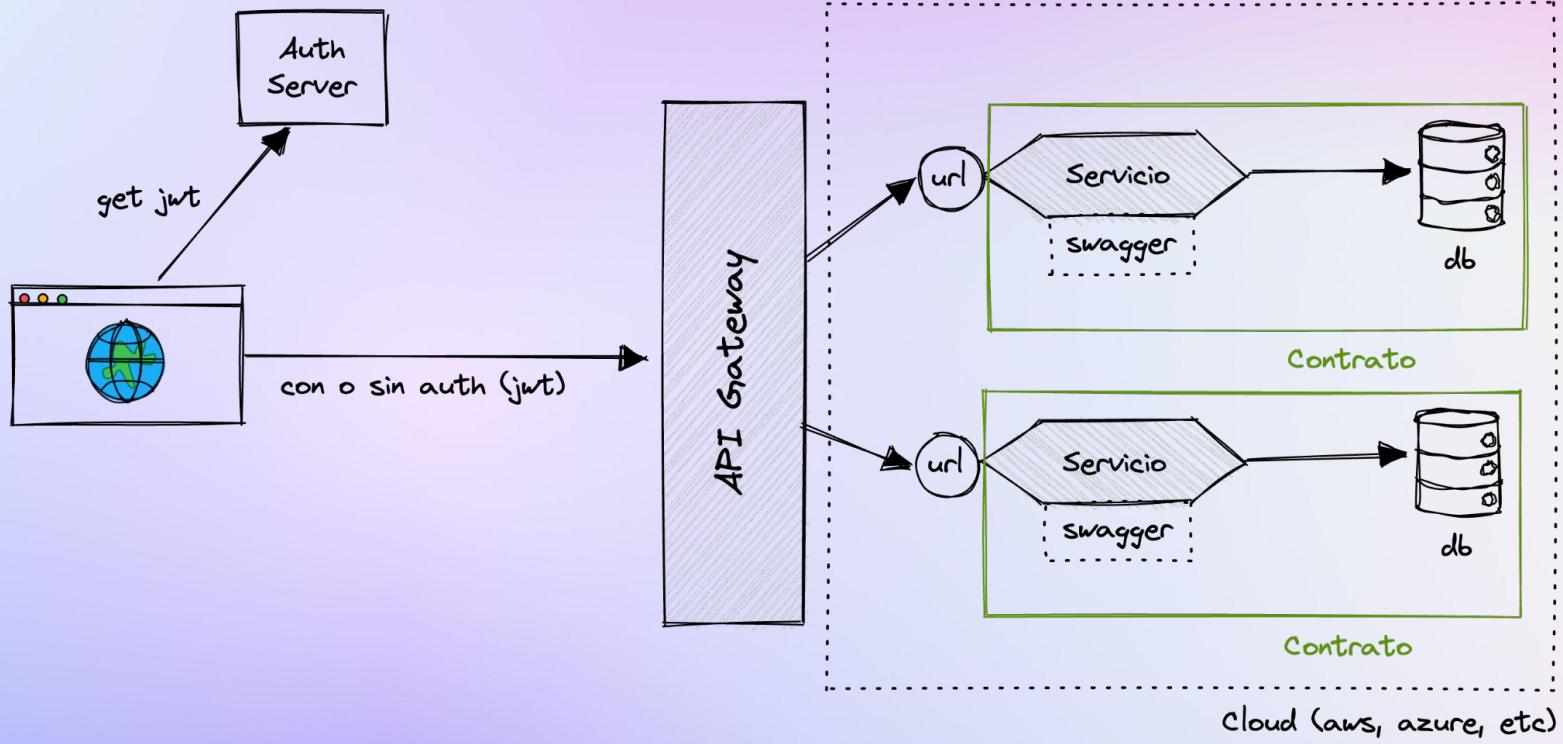


¿Podemos anclarnos en lo que
sabemos de Web2
para **entender el desarrollo**
frontend Web3?



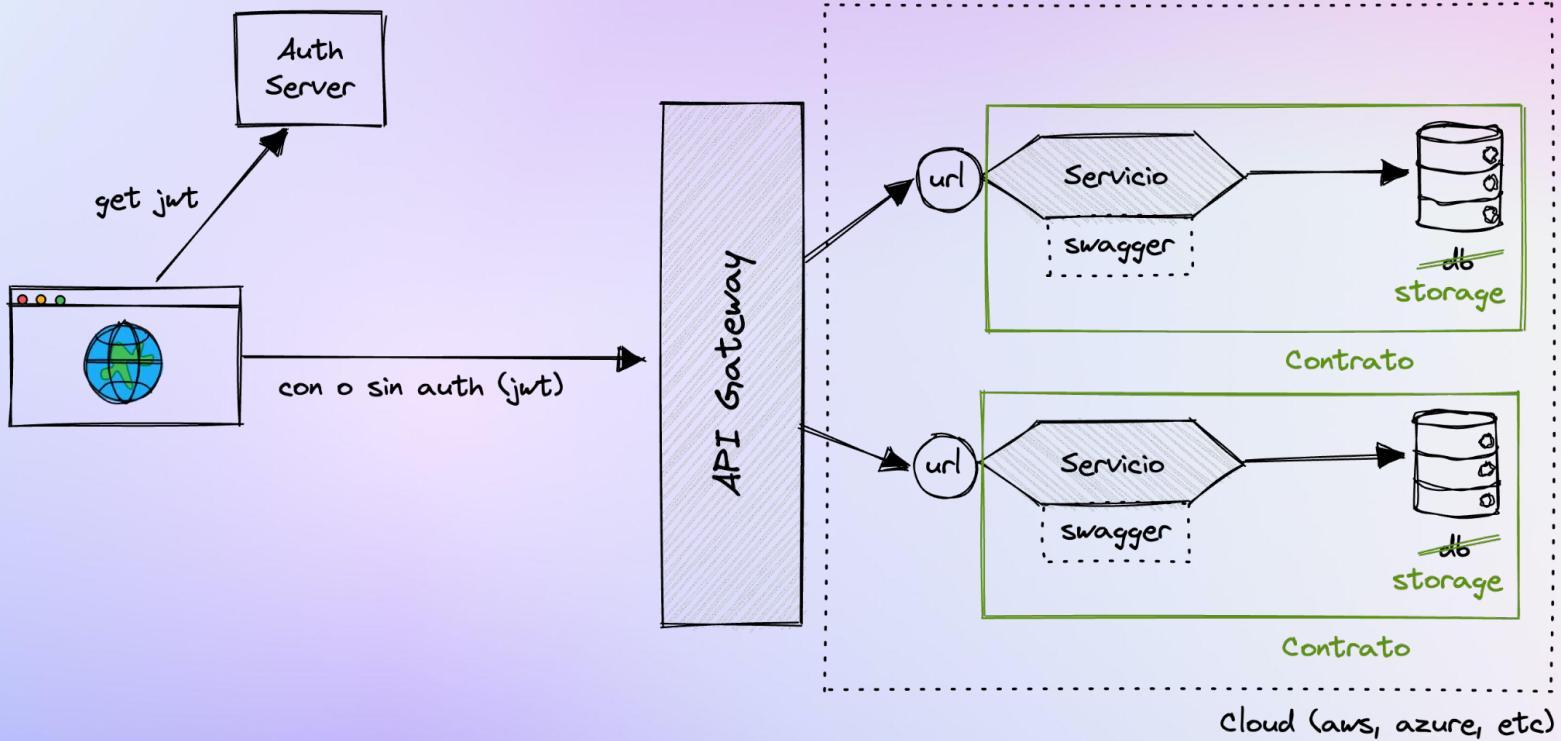
Típica arquitectura Web2

Simplificada y basada en microservicios

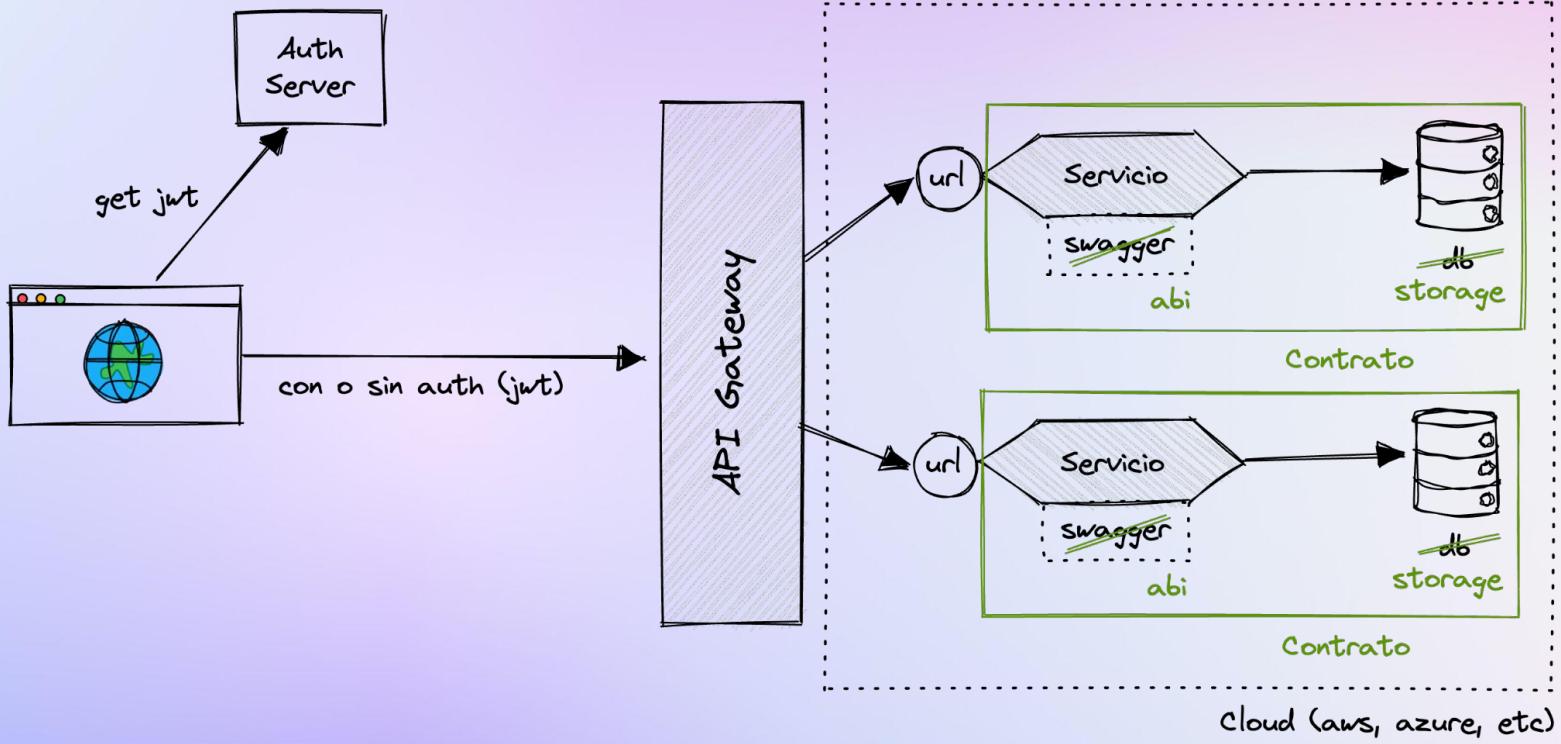


Analogía de arquitectura

Web3 vs Web2

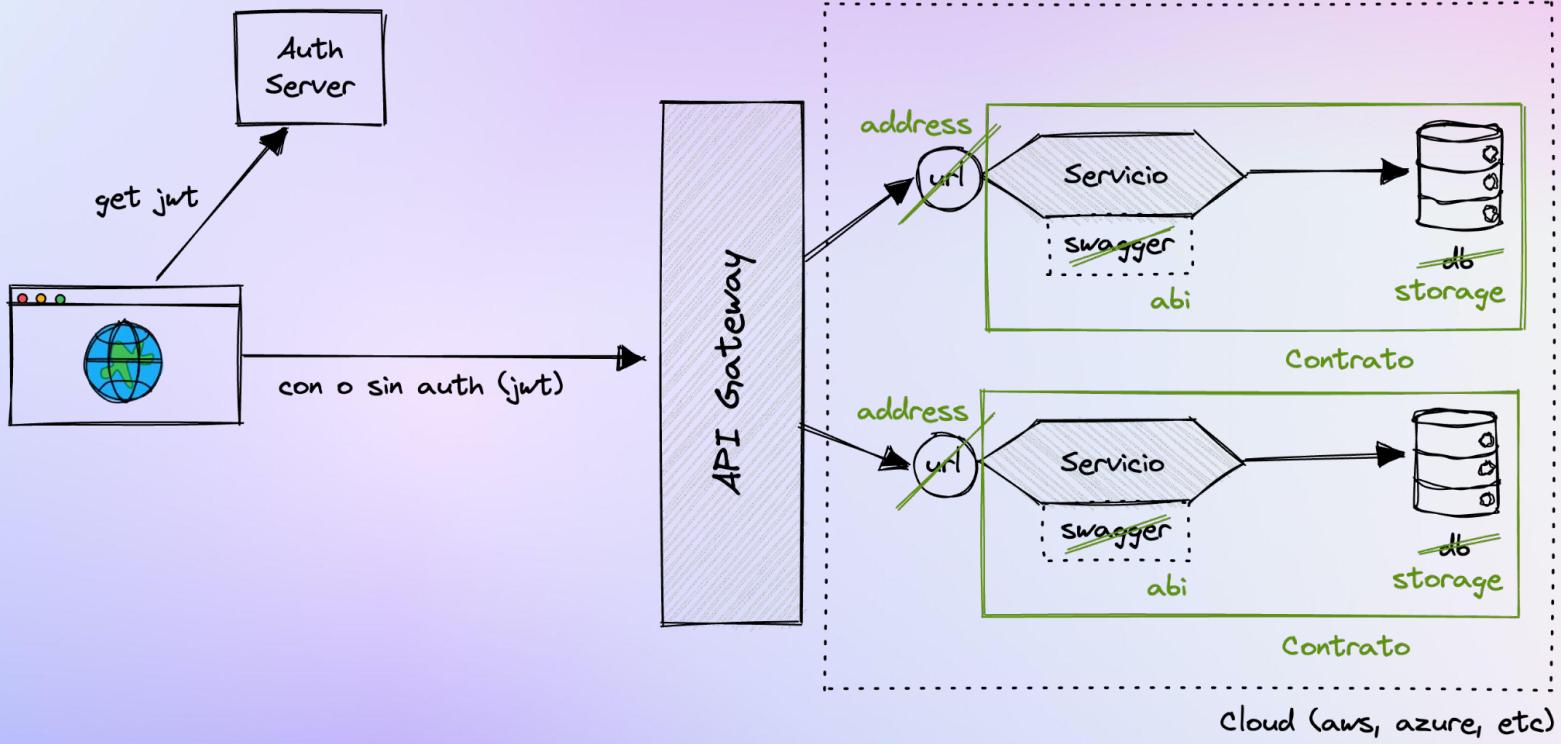


Analogía de arquitectura Web3 vs Web2



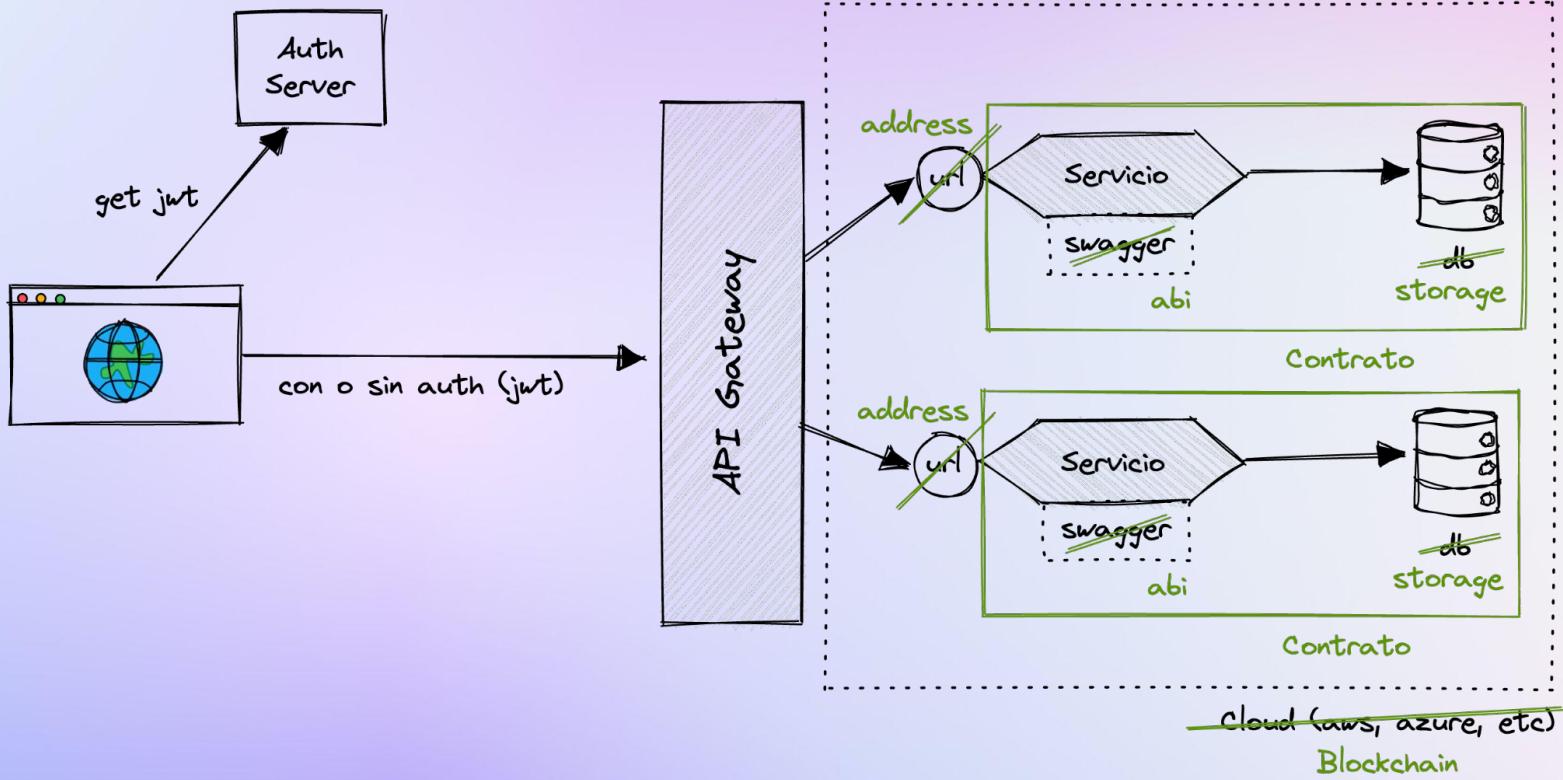
Analogía de arquitectura

Web3 vs Web2



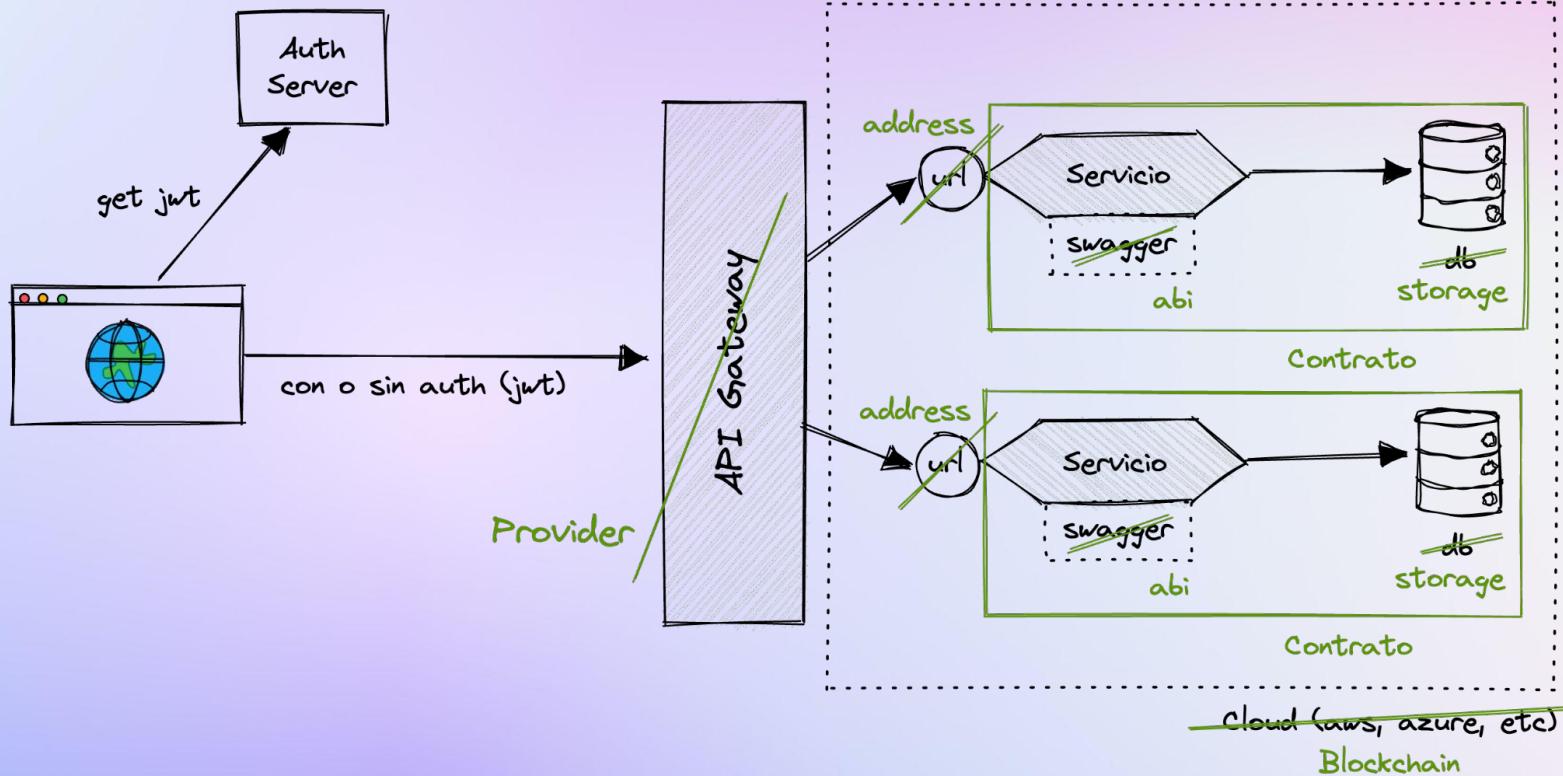
Analogía de arquitectura

Web3 vs Web2



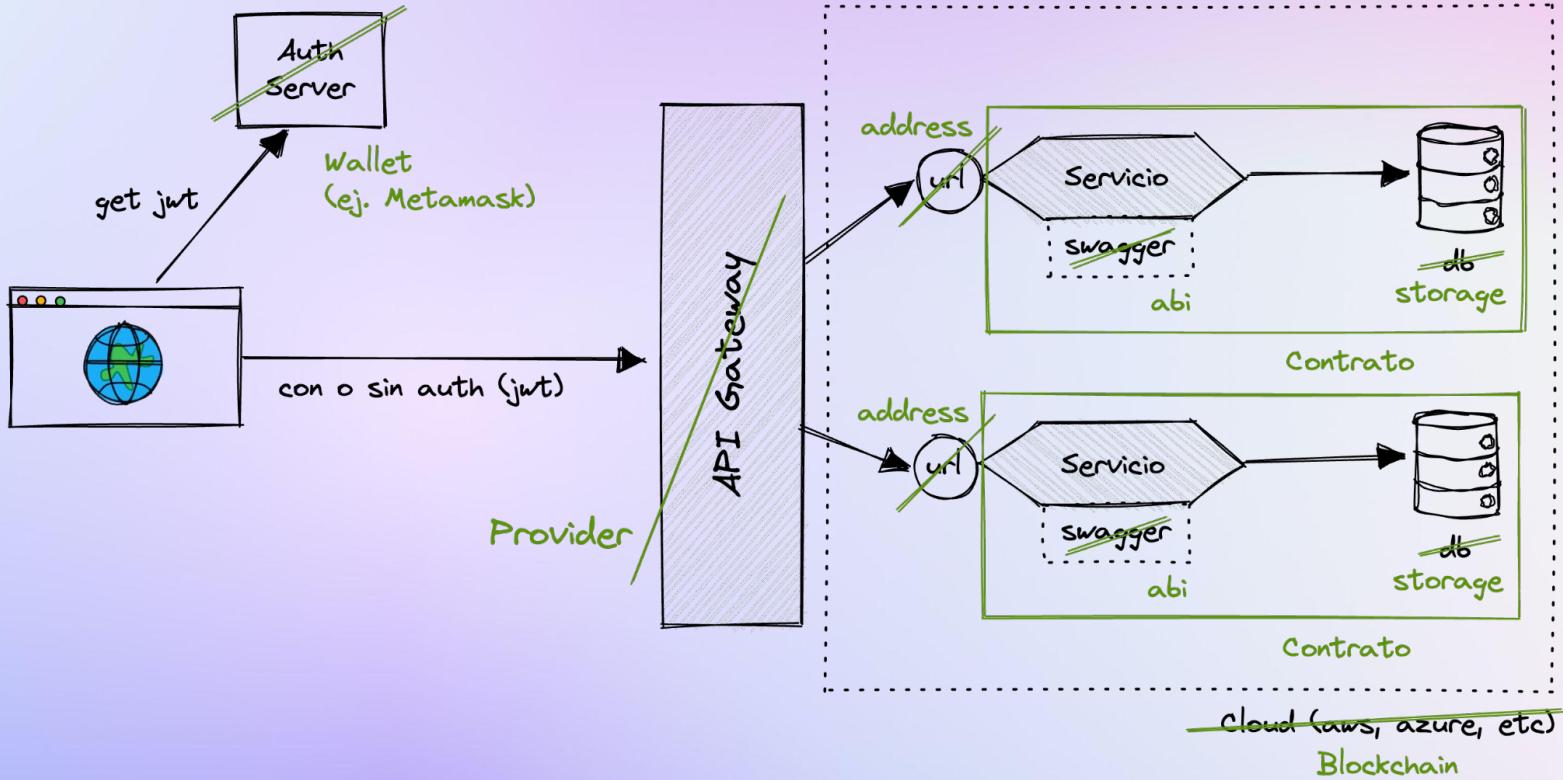
Analogía de arquitectura

Web3 vs Web2



Analogía de arquitectura

Web3 vs Web2



Analogía de arquitectura

Web3 vs Web2

Ejemplos de Código



React hooks para Ethereum



Lib para interactuar con la
Blockchain de Ethereum

- **Setup**
- Read
- Connect
- Write

```
// Setup

import { WagmiConfig, createClient } from 'wagmi'
import { getDefaultProvider } from 'ethers'

const NETWORK_ID = 5 // Goerli

const client = createClient({
  autoConnect: true,
  provider: getDefaultProvider(NETWORK_ID),
})

function App() {
  return (
    <WagmiConfig client={client}>
      <Content />
    </WagmiConfig>
  )
}
```

- Setup
- **Read**
- Connect
- Write

```
// Read

import { useContractRead } from 'wagmi'
import { formatUnits } from 'ethers/lib/utils'

function ReadContractExample() {
  const { data } = useContractRead({
    addressOrName: DAI.address,
    contractInterface: DAI.abi,
    functionName: 'balanceOf',
    args: '0xD2e2B135BCA466271069c394f655e0c70535C2dd',
  })

  return <div>Balance: {data ? formatUnits(data) : '...'}</div>
}
```

- Setup
- Read
- **Connect**
- Write

```
// Connect

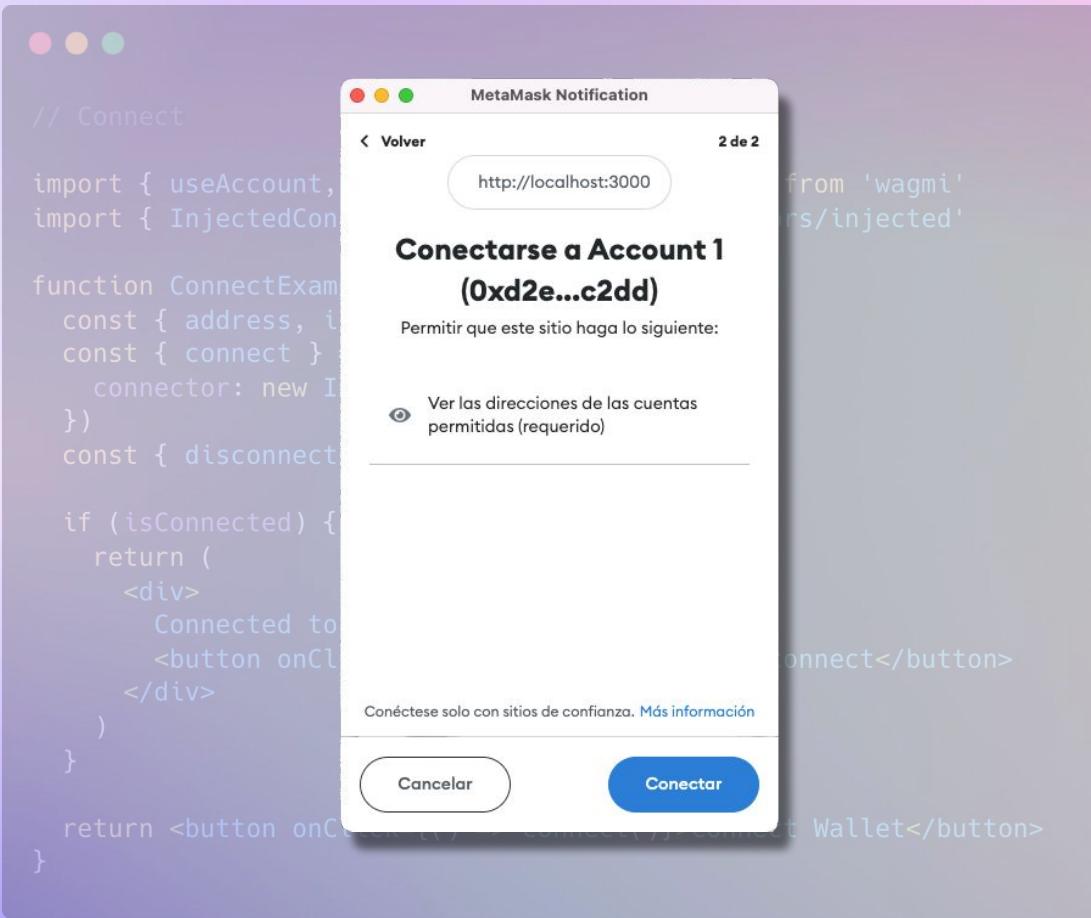
import { useAccount, useConnect, useDisconnect } from 'wagmi'
import { InjectedConnector } from 'wagmi/connectors/injected'

function ConnectExample() {
  const { address, isConnected } = useAccount()
  const { connect } = useConnect({
    connector: new InjectedConnector(),
  })
  const { disconnect } = useDisconnect()

  if (isConnected) {
    return (
      <div>
        Connected to {address}
        <button onClick={() => disconnect()}>Disconnect</button>
      </div>
    )
  }

  return <button onClick={() => connect()}>Connect Wallet</button>
}
```

- Setup
- Read
- **Connect**
- Write



- Setup
- Read
- Connect
- **Write**

```
// Write

import { usePrepareContractWrite, useContractWrite } from 'wagmi'

function WriteContractExample() {
  const { config } = usePrepareContractWrite({
    addressOrName: CDAI.address,
    contractInterface: CDAI.abi,
    functionName: 'mint',
    args: '0.01',
  })

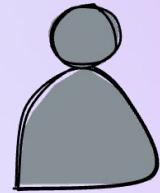
  const { write } = useContractWrite(config)

  return <button onClick={() => write()}>Mint</button>
}
```

Casos de Uso

Consultar saldo de una cuenta | DAI

Operación Sólo-lectura

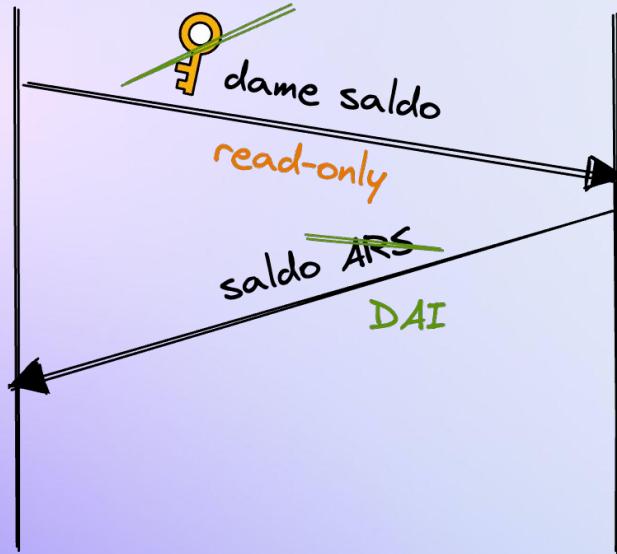


Usuario



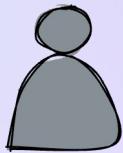
~~API Banco~~

Contrato
DAI



Invertir en un Fondo Común de Inversión | Compound

Operación de Escritura

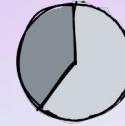


Usuario



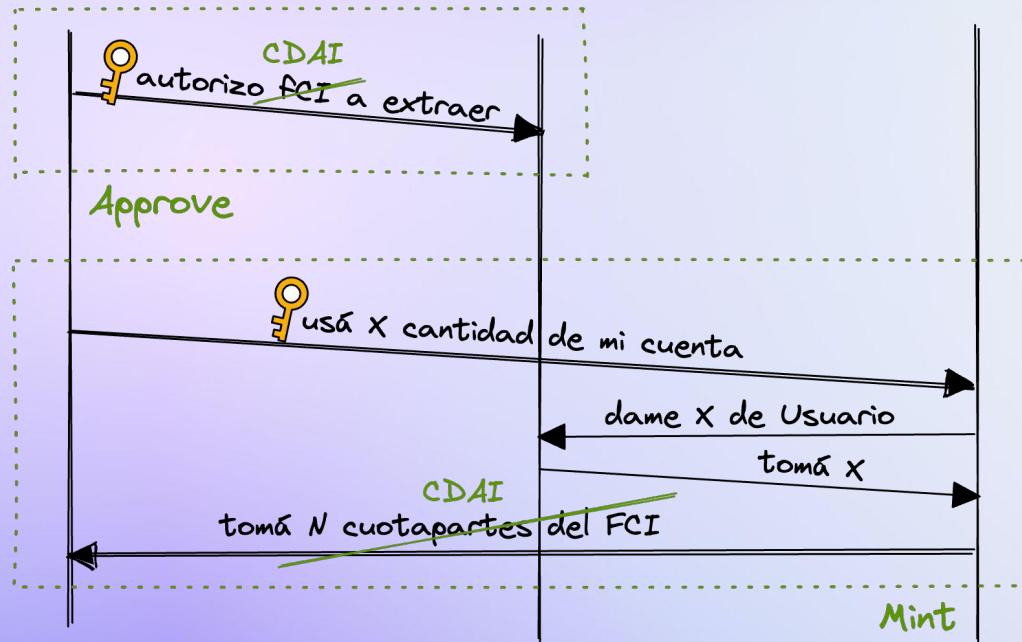
API Banco

Contrato
DAI



FCI

Compound /
Contrato CDAI



Demo

Algunos **tips** y **herramientas** para
tener en cuenta al desarrollar

- **No operes con números de JavaScript**
Usá siempre BigNumber
- **Pensá tu UI considerando TX asincrónicas**
Podés escuchar su estado para comunicarle al usuario
(spinners, banners, etc)

- **Testnets y Faucets**
No es necesario tener ETH para desarrollar
- **Block Explorer**
Toda la info de lo que pasa en la Blockchain
- **The Graph**
Consultá info compleja sobre TXs usando GraphQL

¡Gracias!

◎ UNDERSCOPE

React Buenos Aires | Octubre 2022

