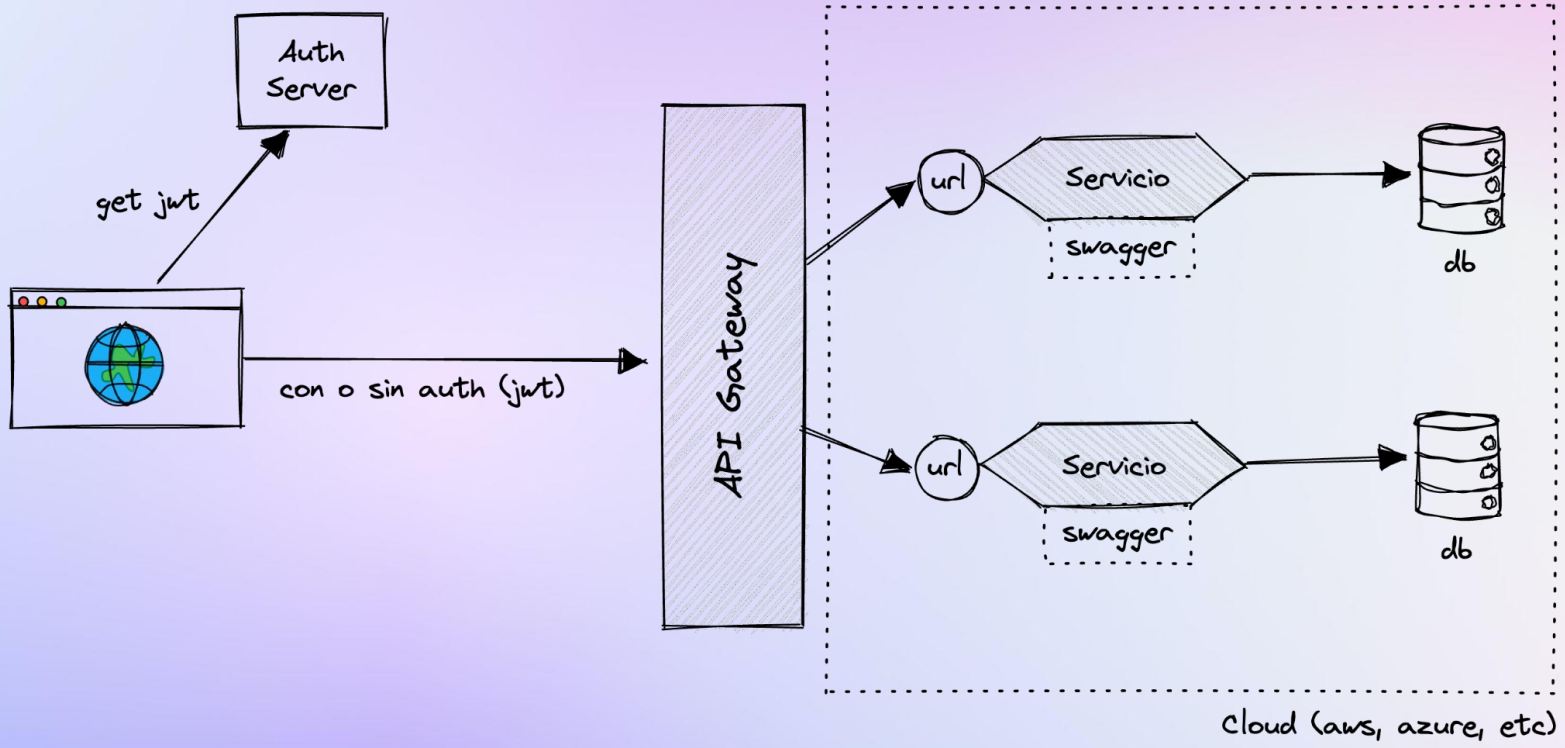
Intro al desarrollo frontend para DApps con React

UNDERSCOPE

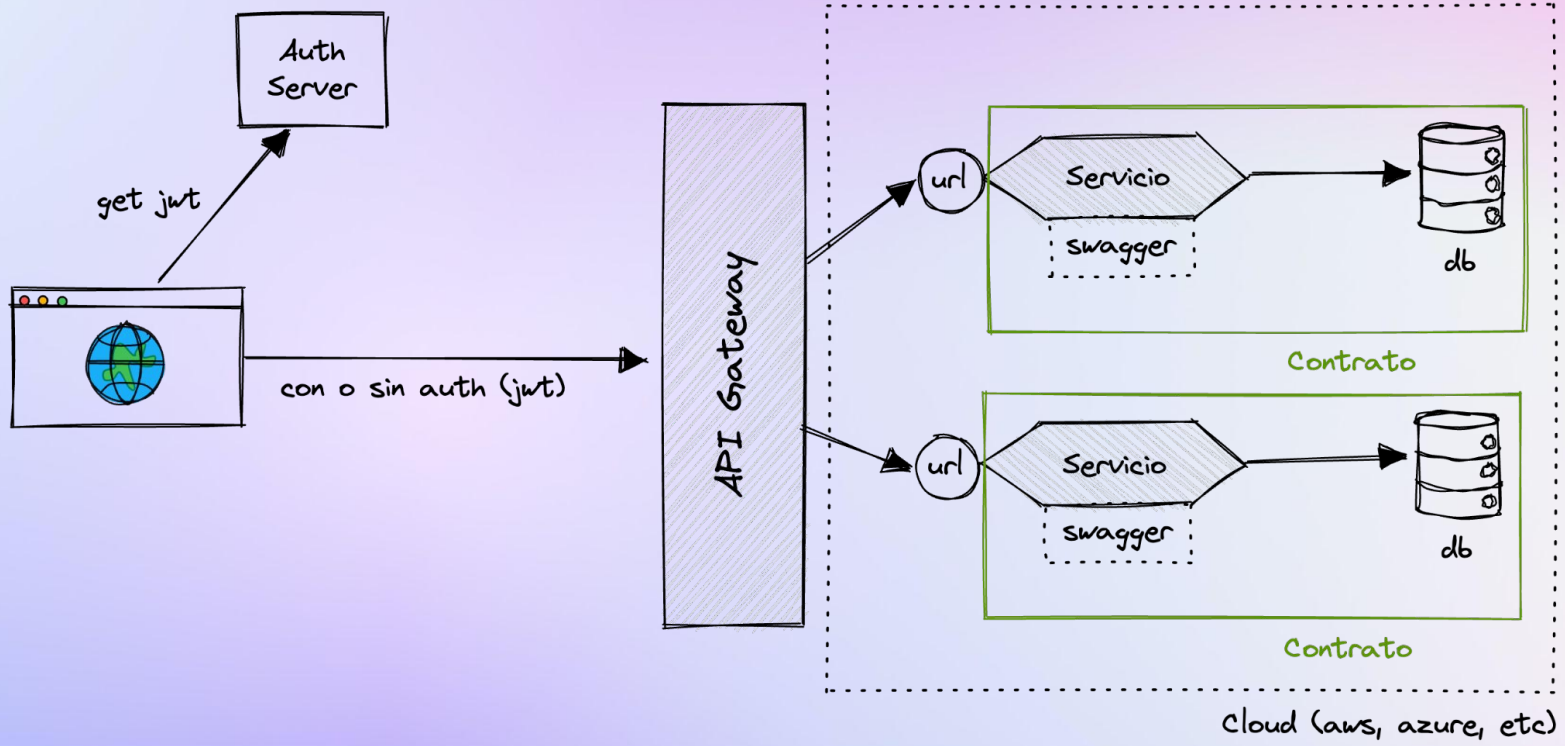
LABITCONF | Buenos Aires 2022

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



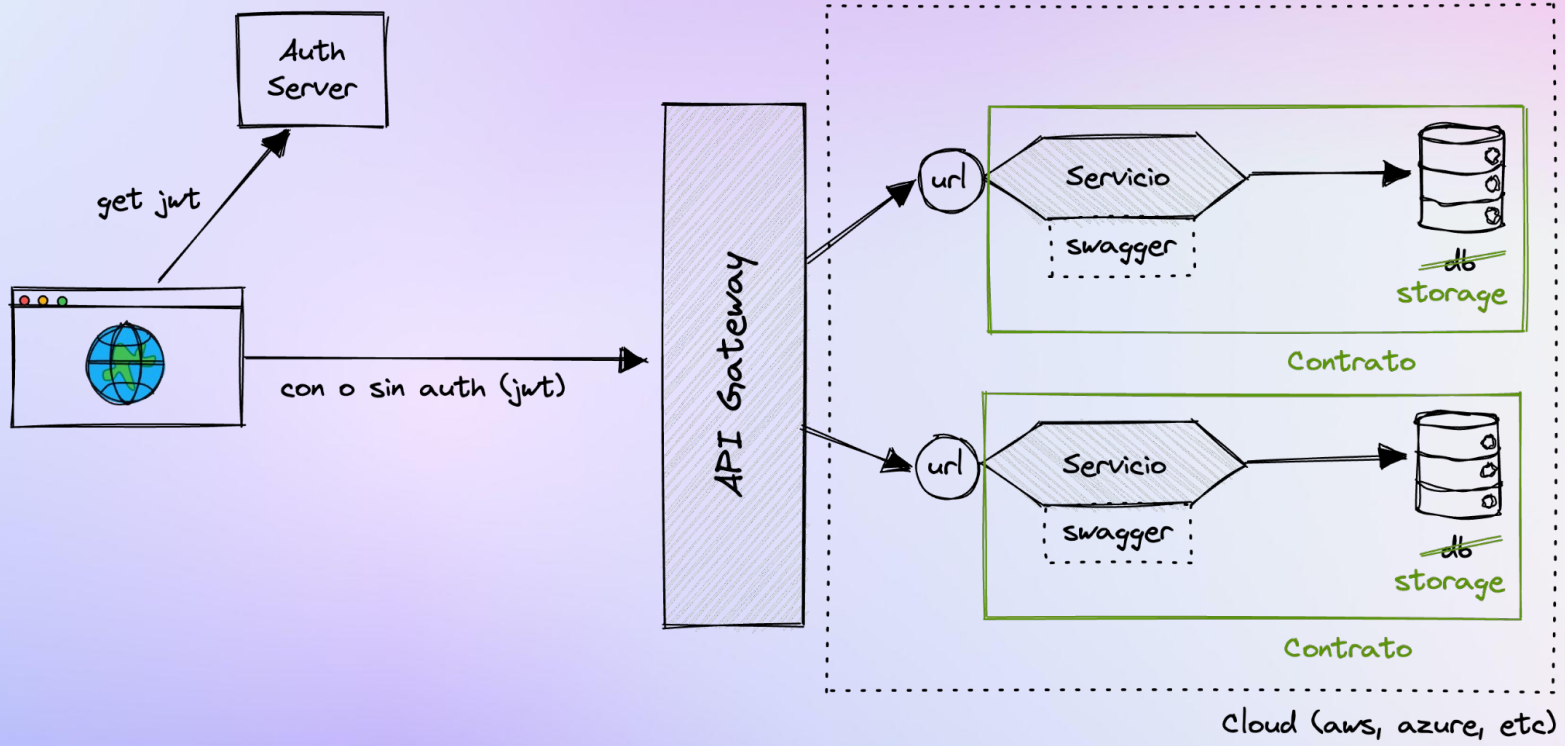
Típica arquitectura Web2

Simplificada y basada en microservicios



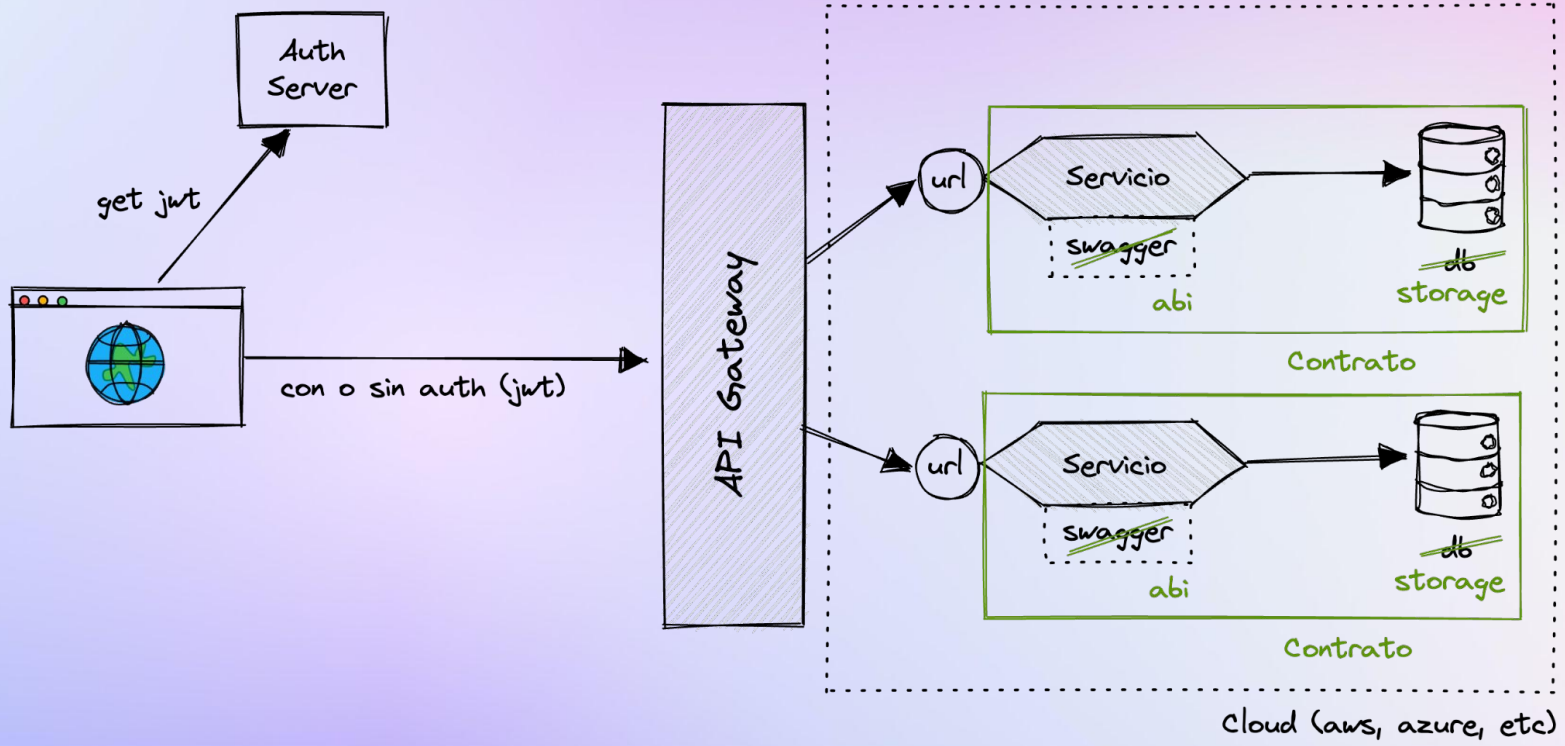
Analogía de arquitectura

Web3 vs Web2



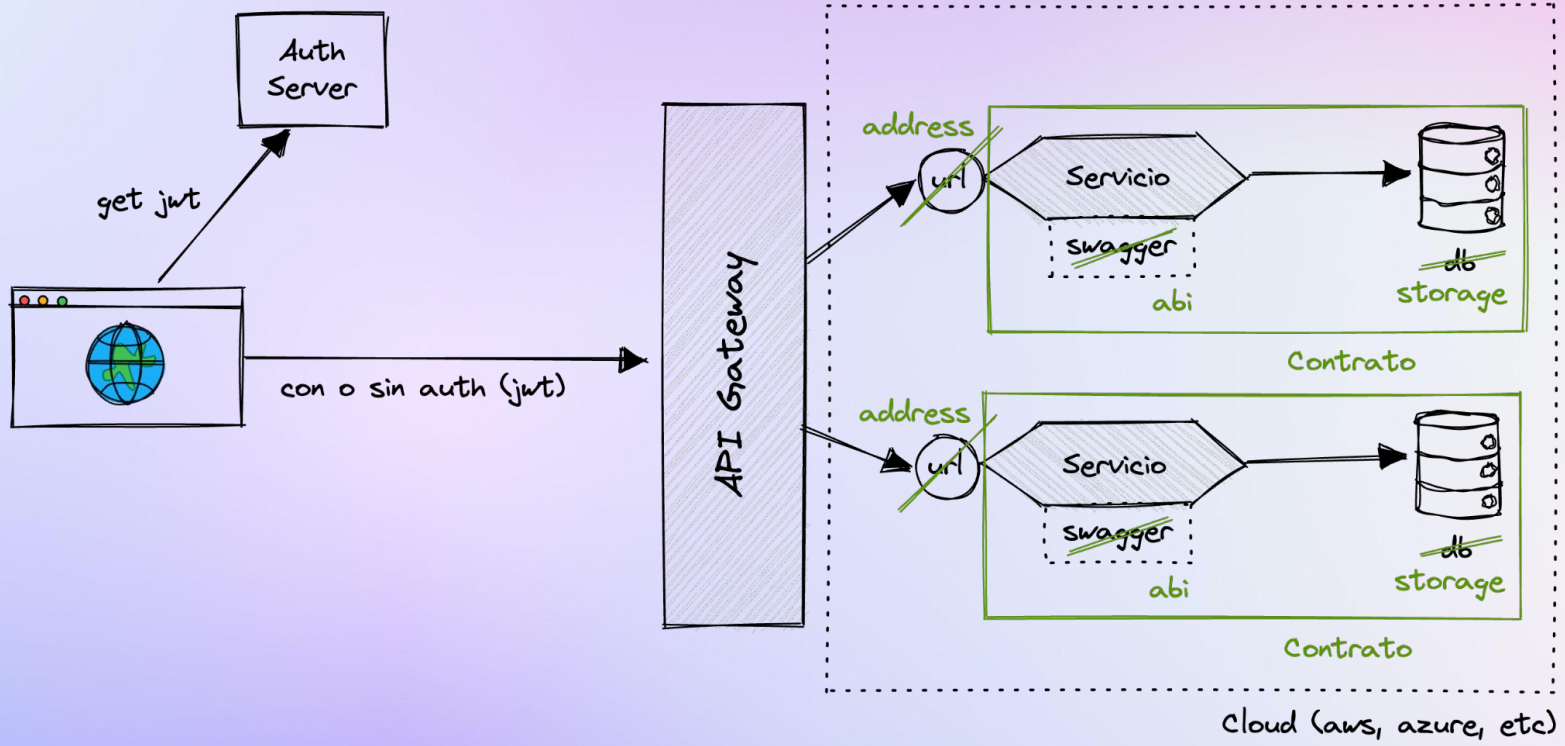
Analogía de arquitectura

Web3 vs Web2



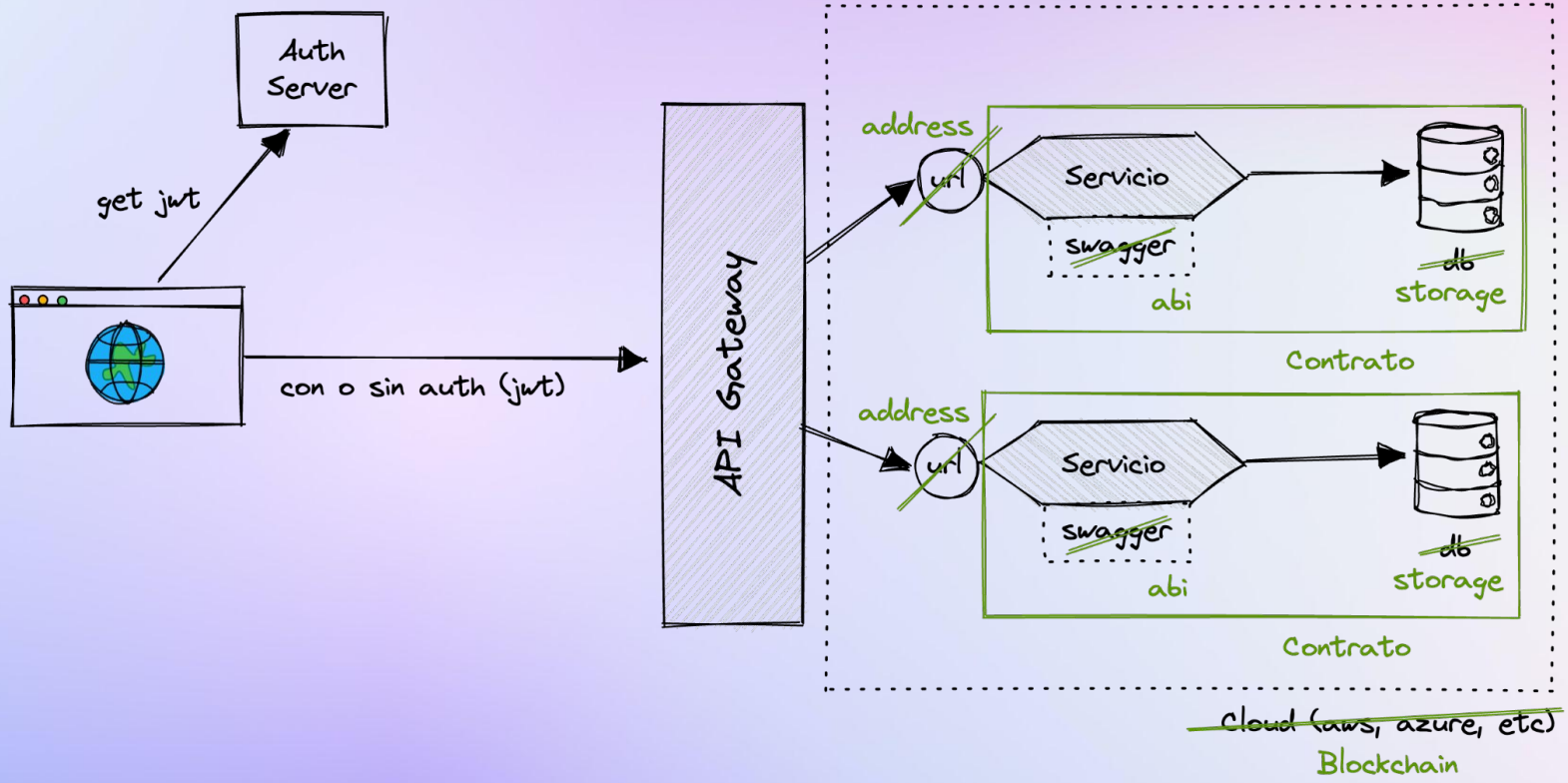
Analogía de arquitectura

Web3 vs Web2



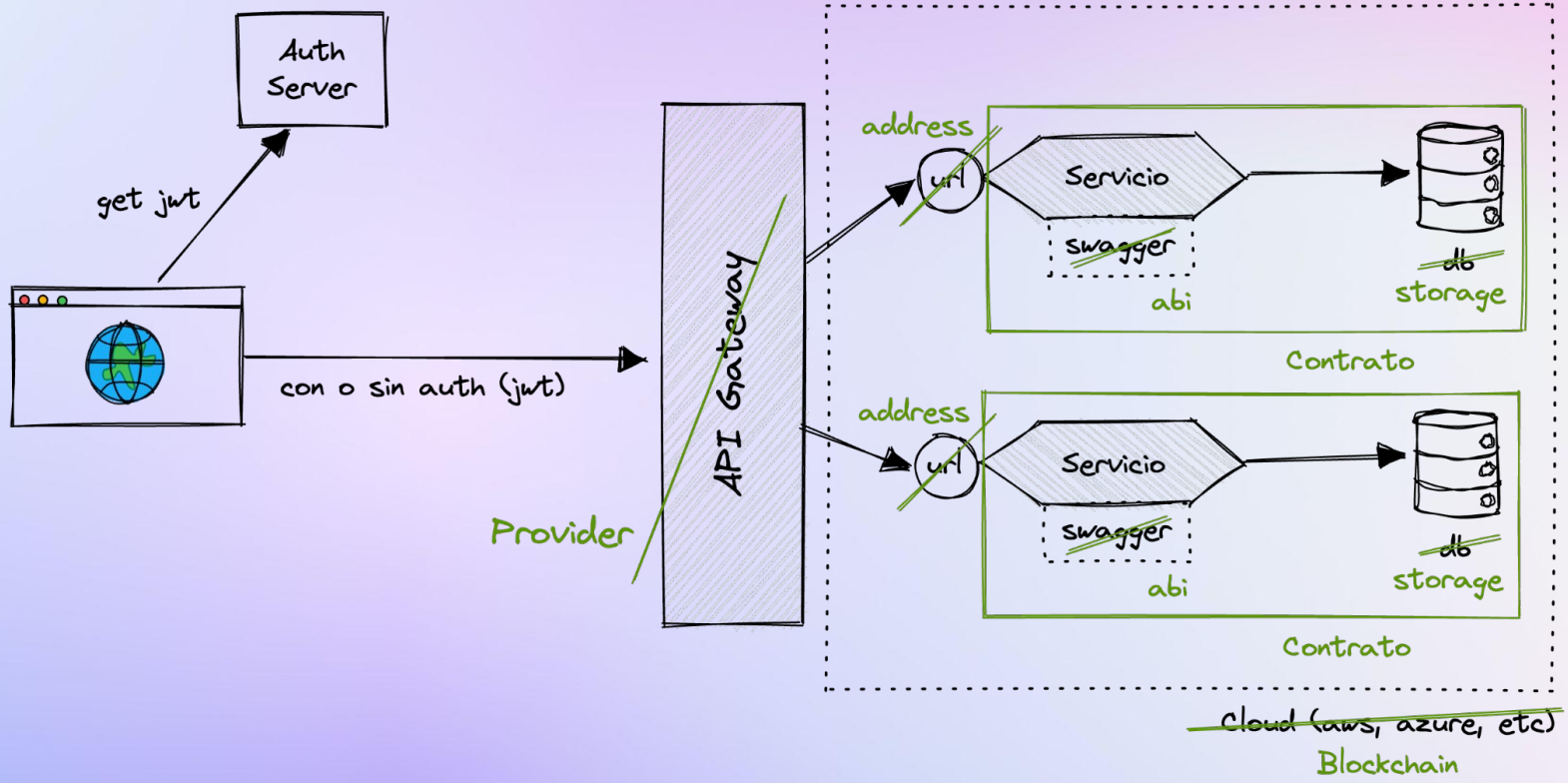
Analogía de arquitectura

Web3 vs Web2



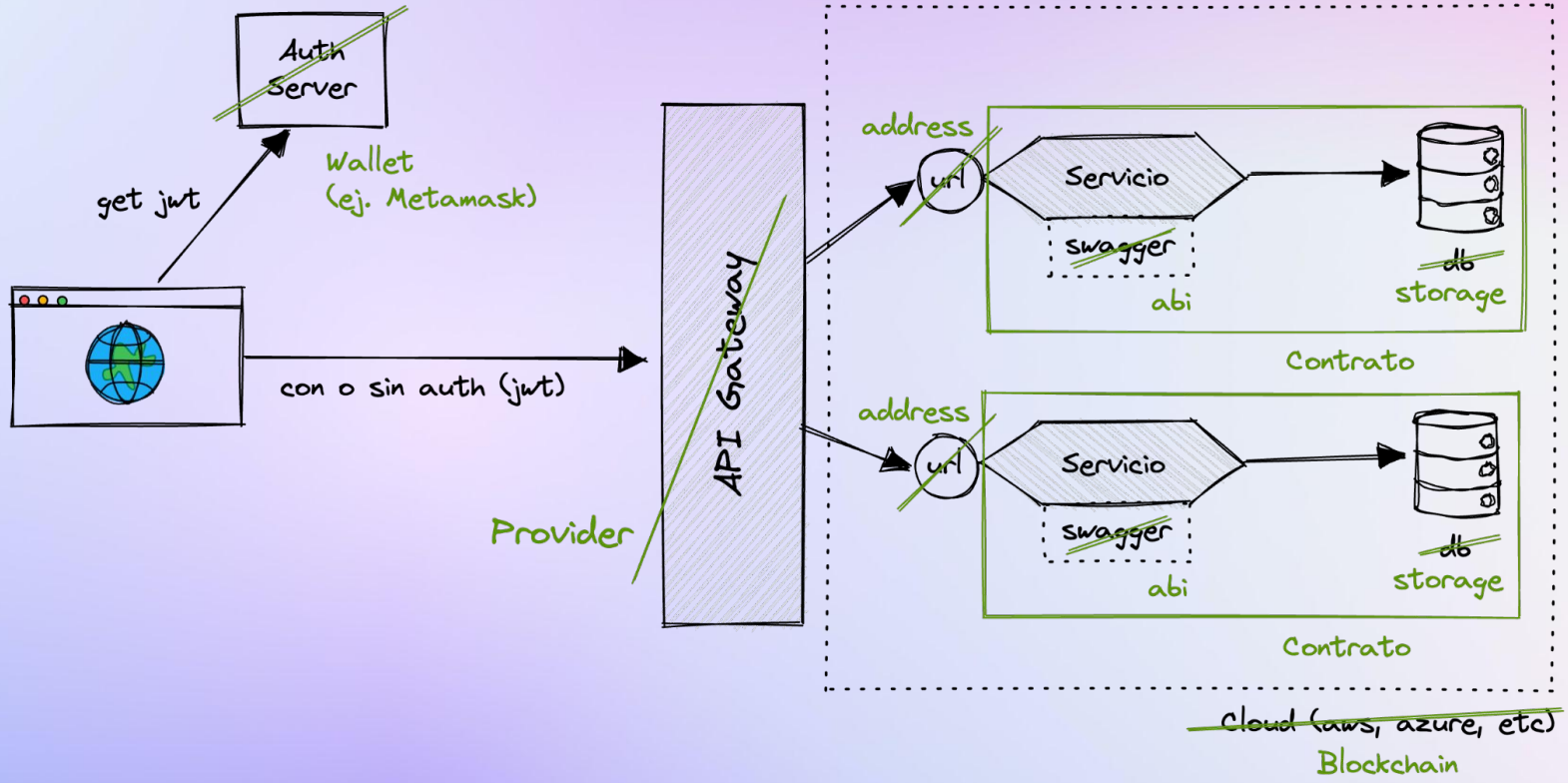
Analogía de arquitectura

Web3 vs Web2



Analogía de arquitectura

Web3 vs Web2



Analogía de arquitectura

Web3 vs Web2

Ejemplos de **Código**

wagmi

React hooks para Ethereum

ethers.js

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>
  )
}
```

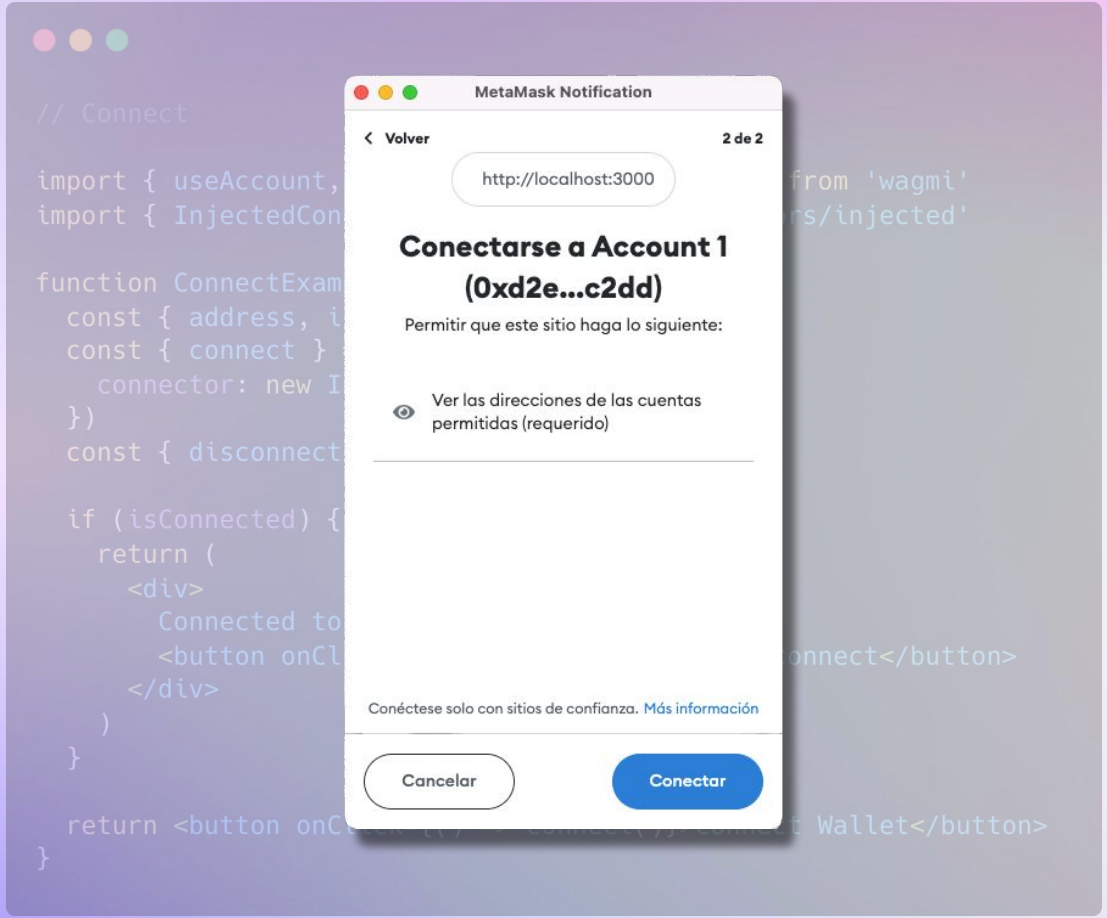
- Setup
- **Read**
- Connect
- Write

```
// 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

```tsx
// 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



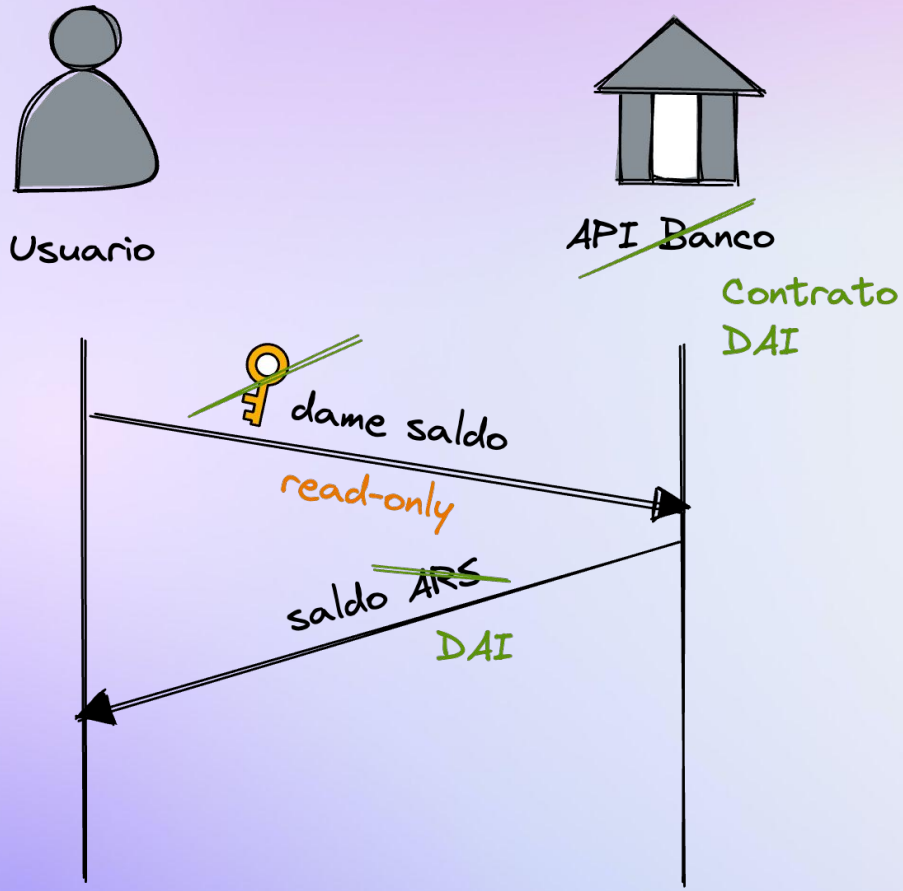
- Setup
- Read
- Connect
- **Write**

```
// 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**

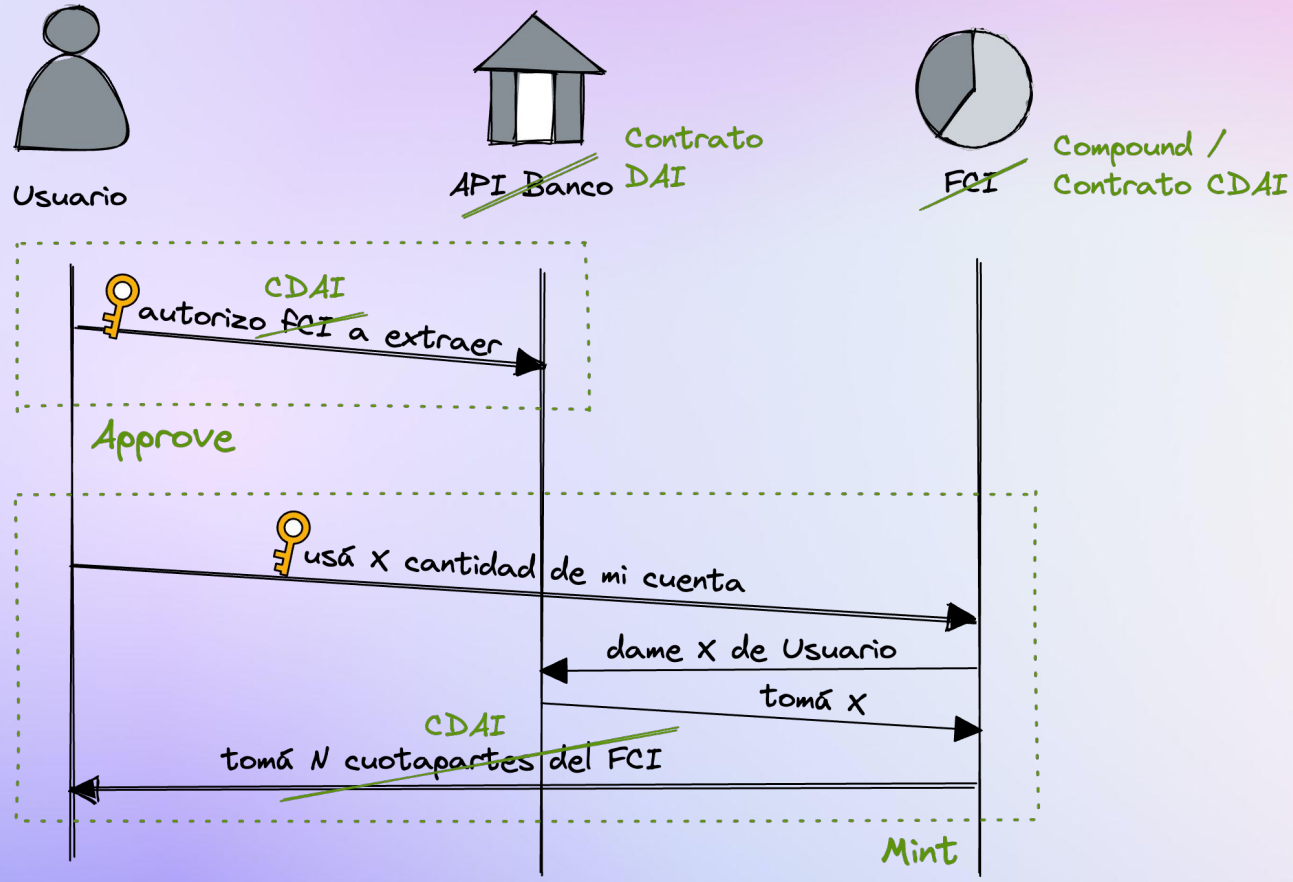
Consultar saldo de una cuenta | DAI

Operación Sólo-lectura

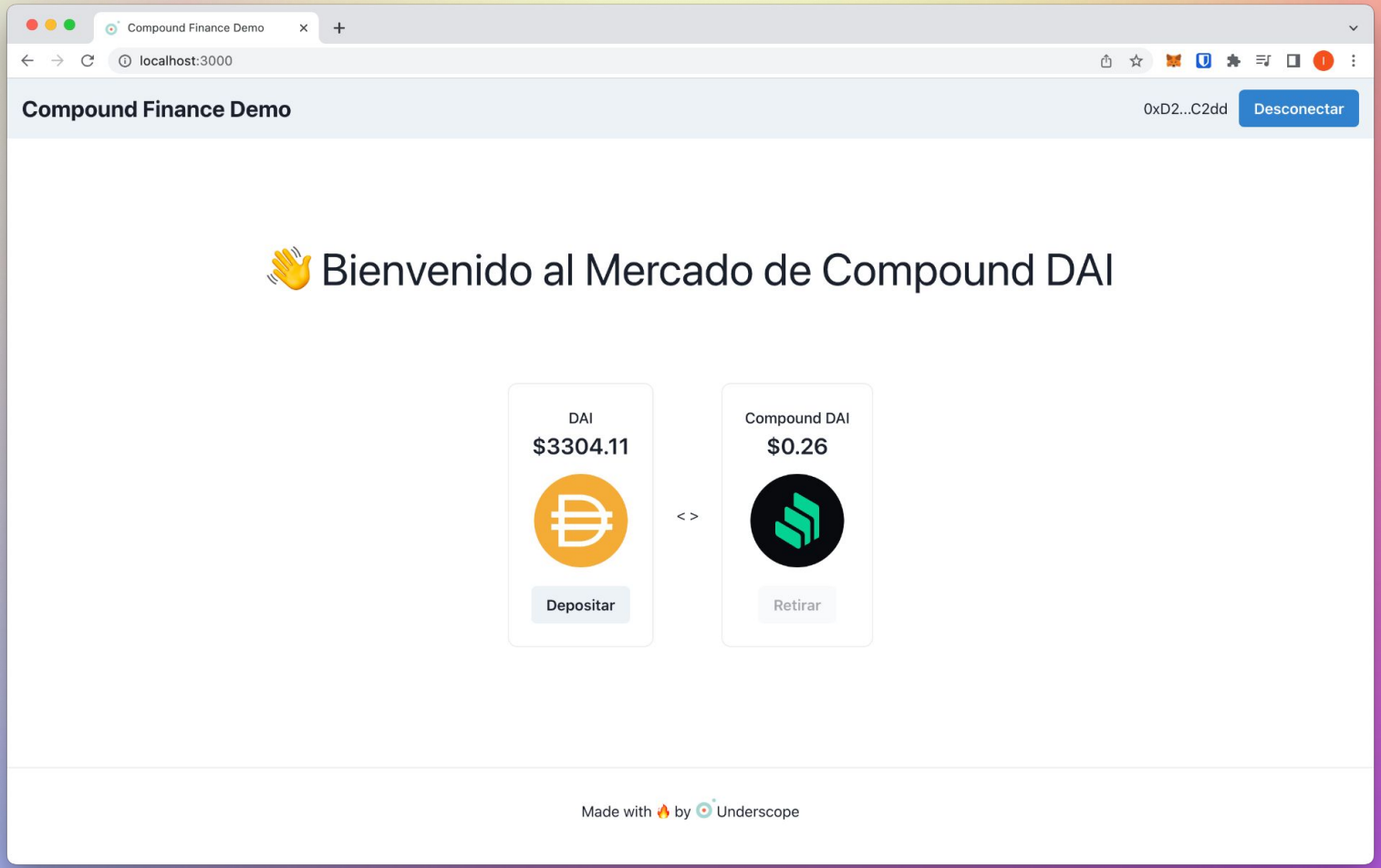

Usuario
API Banco
Contrato
DAI
dame saldo
read-only
saldo ARS
DAI

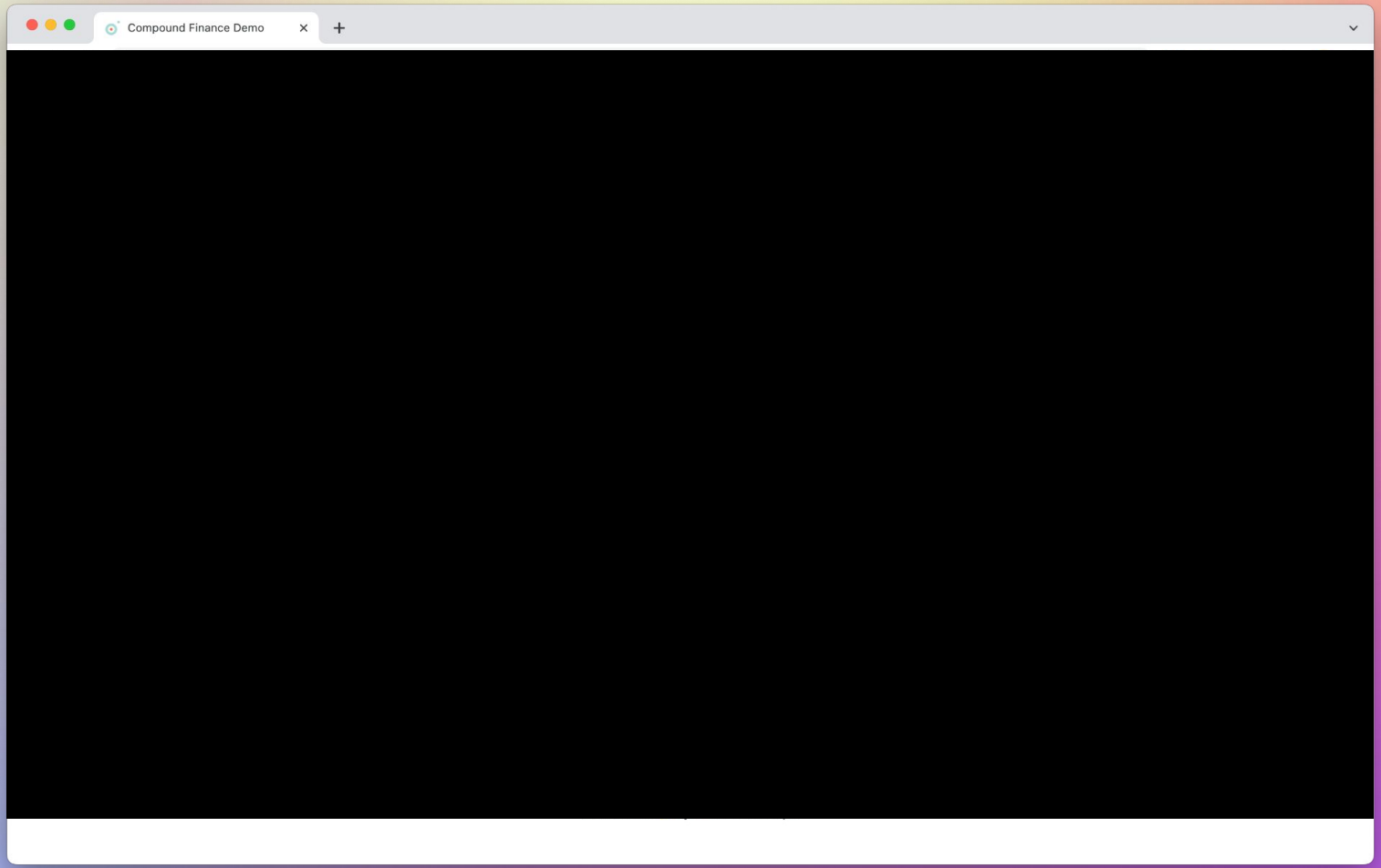
Invertir en un Fondo Común de Inversión | Compound

Operación de Escritura


Usuario
API Banco
Contrato DAI
FCI
Compound / Contrato CDAI
CDAI
autorizo FCI a extraer
Approve
usá X cantidad de mi cuenta
dame X de Usuario
tomá X
CDAI
tomá N cuotapartes del FCI
Mint

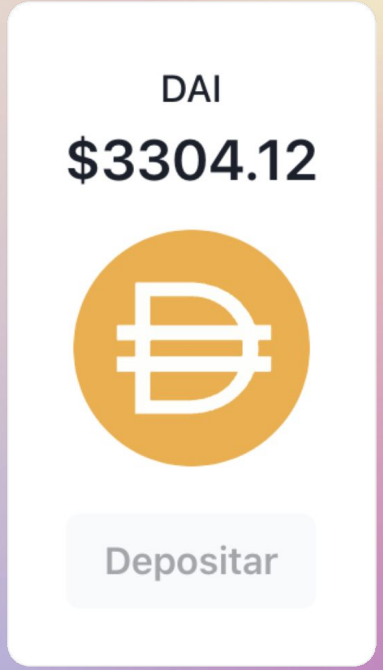
Demo


Compound Finance Demo
localhost:3000
Compound Finance Demo
0xD2...C2dd
Desconectar
👋 Bienvenido al Mercado de Compound DAI
DAI
$3304.11
Depositar
< >
Compound DAI
$0.26
Retirar
Made with 🔥 by Underscope


Compound Finance Demo

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

function MyApp({ Component, pageProps }: AppProps) {
  const isMounted = useIsMounted()

  if (!isMounted) {
    return null
  }

  return (
    <WagmiConfig client={wagmiClient}>
      <Component {...pageProps} />
    </WagmiConfig>
  )
}
```

> Setup > Balances > Conectar > Autorizar > Depositar > Refrescar




```
const useDaiBalance = (address?: string) => {
  const res = useContractRead({
    addressOrName: DAI.address,
    contractInterface: DAI.abi,
    functionName: 'balanceOf',
    args: address,
  })

  return {
    ...res,
    formatted: formatBigNumber(res.data, 2),
  }
}
```

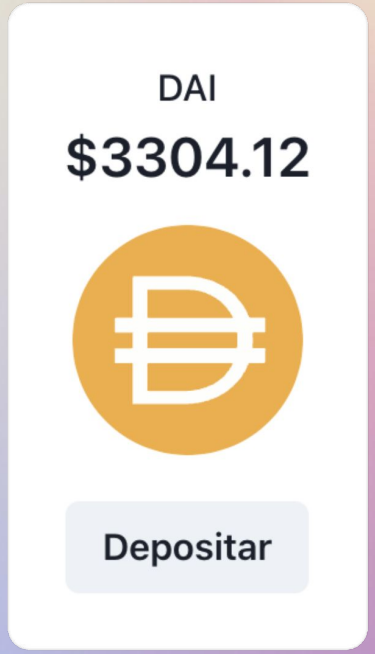
> Setup **> Balances** > Conectar > Autorizar > Depositar > Refrescar


```tsx
const Home: NextPage = () => {
  // Connection
  const { connect } = useConnect({
    connector: new InjectedConnector(),
  })
  const { disconnect } = useDisconnect()
  const { address, isConnected } = useAccount()

  // Read
  const daiBalance = useDaiBalance(address)
  const cdaiBalance = useCdaiUnderlyingBalance(address)
  const cdaiAllowance = useCdaiAllowance(address)

  return (...)
}
```

> Setup > Balances **> Conectar** > Autorizar > Depositar > Refrescar

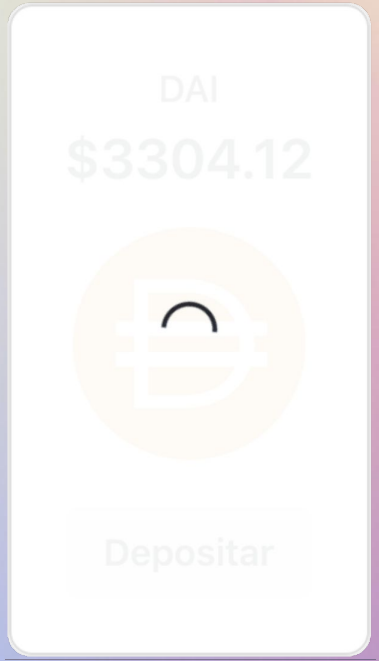

```
const useCdaiApprove = () => {
  const { config } = usePrepareContractWrite({
    addressOrName: DAI.address,
    contractInterface: DAI.abi,
    functionName: 'approve',
    args: [CDAI.address, constants.MaxInt256],
  })

  return useContractWrite(config)
}
```

> Setup > Balances > Conectar **> Autorizar** > Depositar > Refrescar




```
const useCdaiMint = () => {
  const { config } = usePrepareContractWrite({
    addressOrName: CDAI.address,
    contractInterface: CDAI.abi,
    functionName: 'mint',
    args: parseUnits('0.01'),
  })

  return useContractWrite(config)
}
```

> Setup > Balances > Conectar > Autorizar **> Depositar** > Refrescar




```
// Read
const daiBalance = useDaiBalance(address)
const cdaiBalance = useCdaiUnderlyingBalance(address)

// Write
const mint = useCdaiMint()

// Tx info
const mintTx = useWaitForTransaction({ hash: mint.data?.hash })

useEffect(() => {
  if (mintTx.isSuccess) {
    daiBalance.refetch()
    cdaiBalance.refetch()
  }
}, [mintTx.isSuccess])
```

> Setup > Balances > Conectar > Autorizar > Depositar **> Refrescar**

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

- **TheGraph.com**
  Consultá info compleja sobre TXs usando GraphQL

¡Gracias!

UNDERSCOPE

LABITCONF | Buenos Aires 2022