



# Desarolla Web3 con lo Que Sabes de Web 2 En Hedera

Ed Marquez – Swirlds Labs

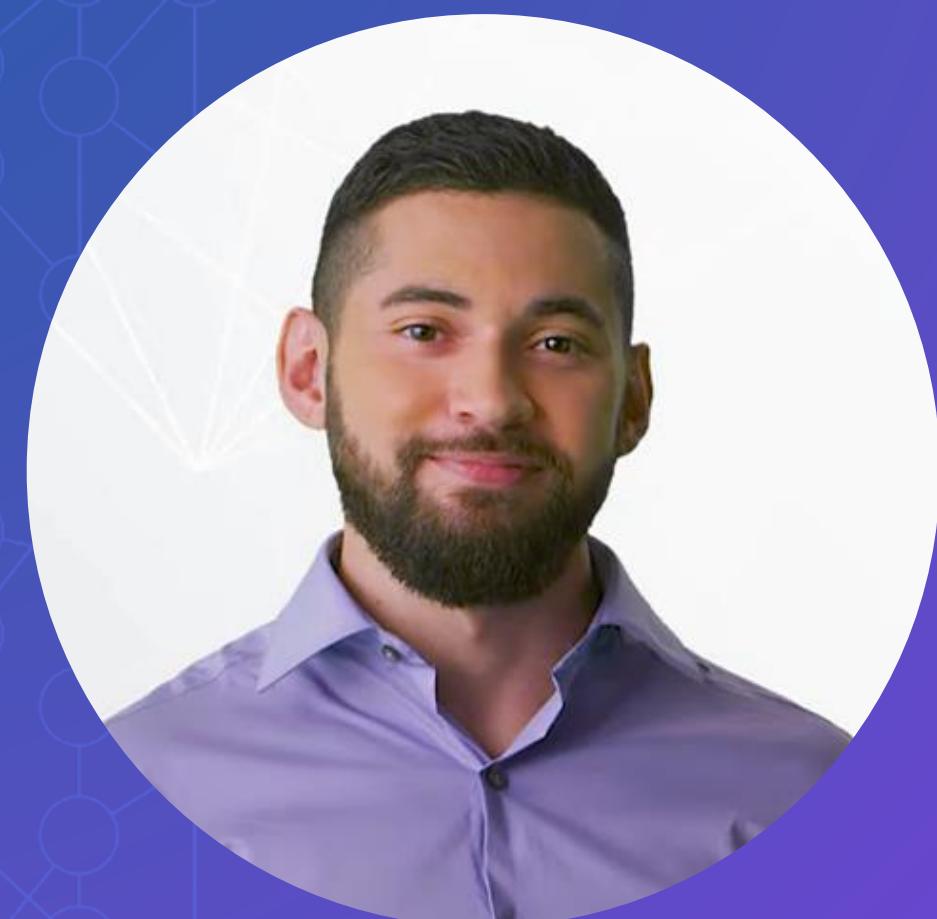


[/ed-marquez](https://www.linkedin.com/in/ed-marquez)



[@ed\\_marquez](https://twitter.com/@ed_marquez)





## Ed Marquez

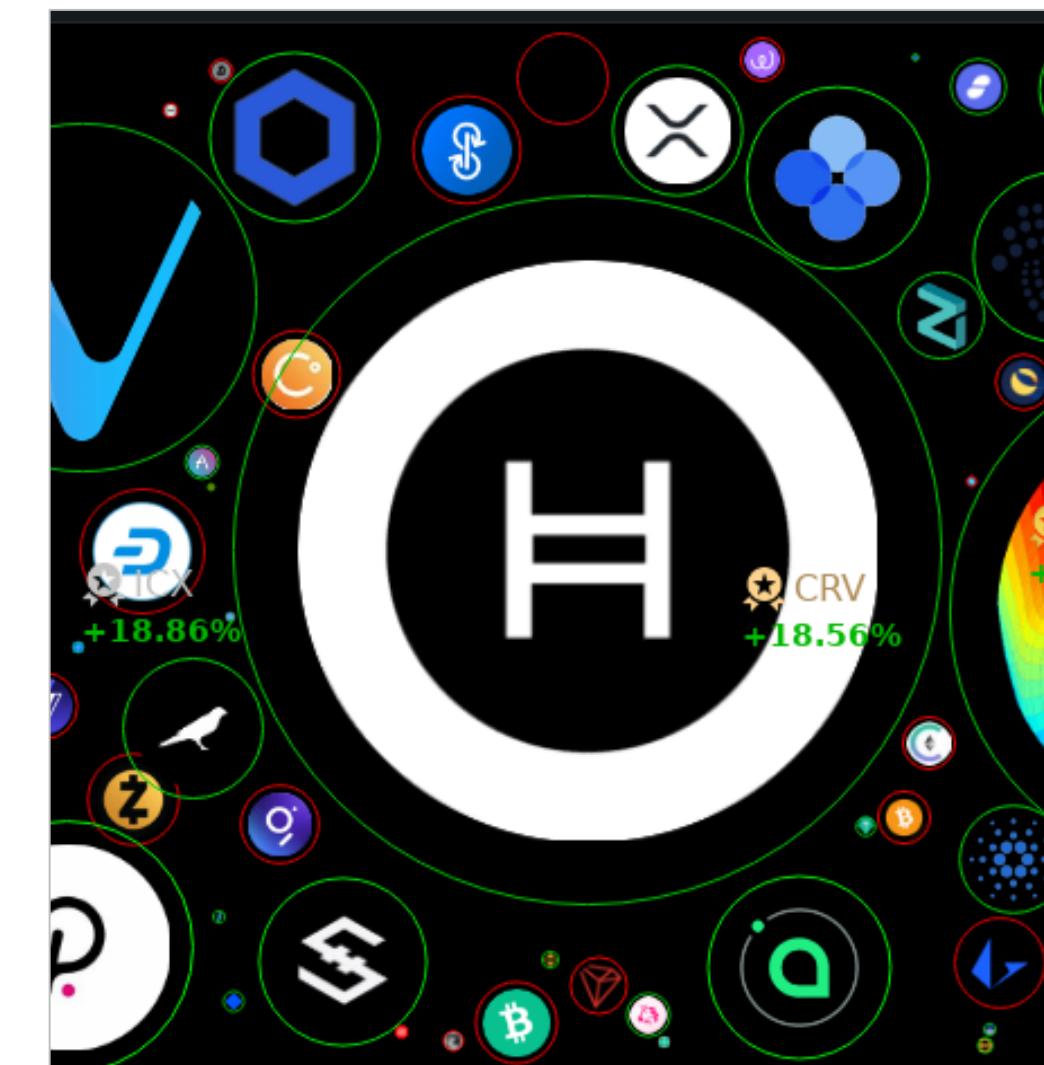
*Developer Relations Engineer  
at Swirls Labs*

### Let's Connect!

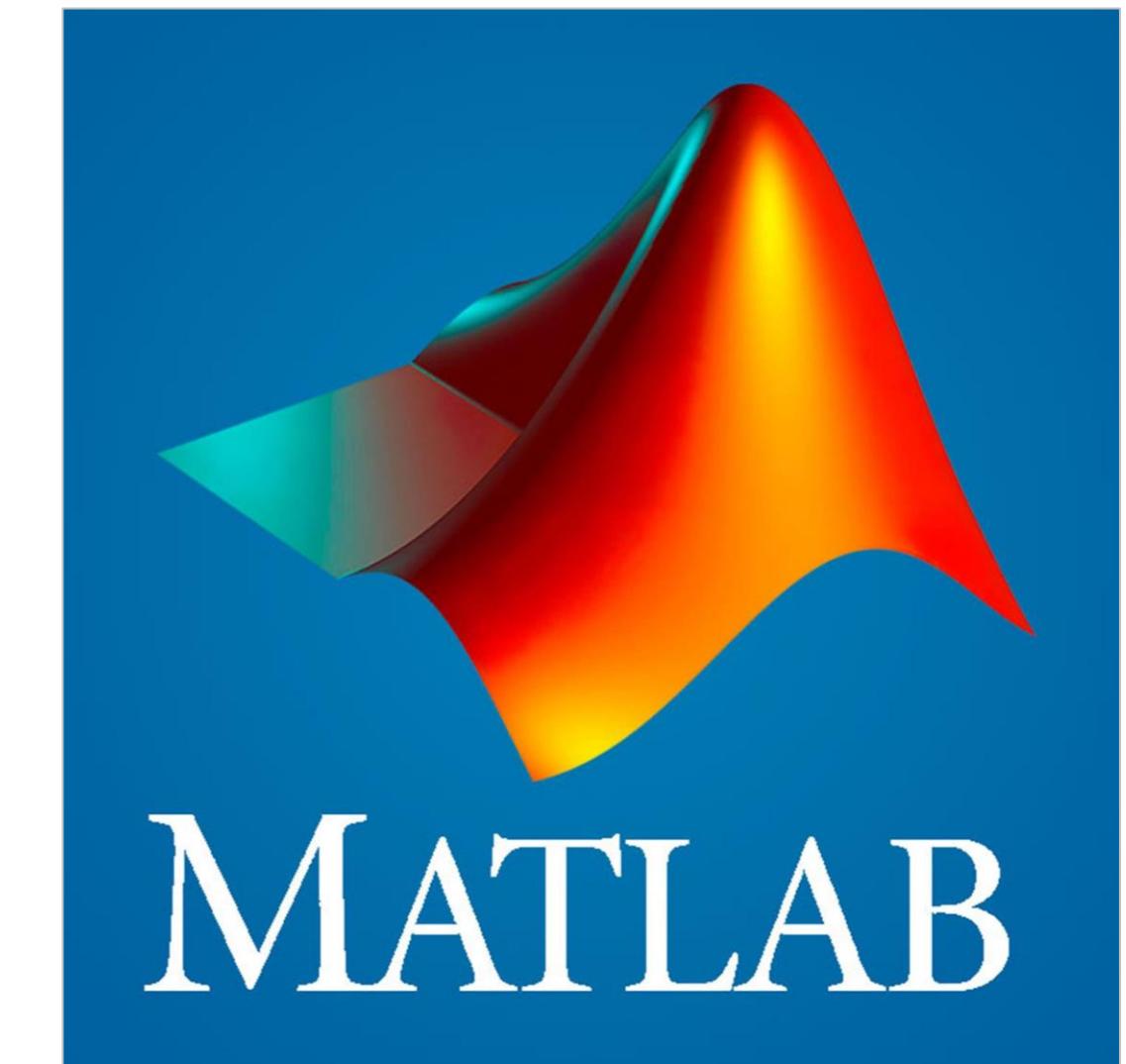
- @ed\_marquez
- /in/ed-marquez



## Ingeniero Mecánico



## Pasión por Web 3



## Trabajé en MathWorks



## Nala & Pepper

# En esta sesión aprenderás sobre **registros distribuidos** y como empezar a desarrollar aplicaciones web 3



Introducción

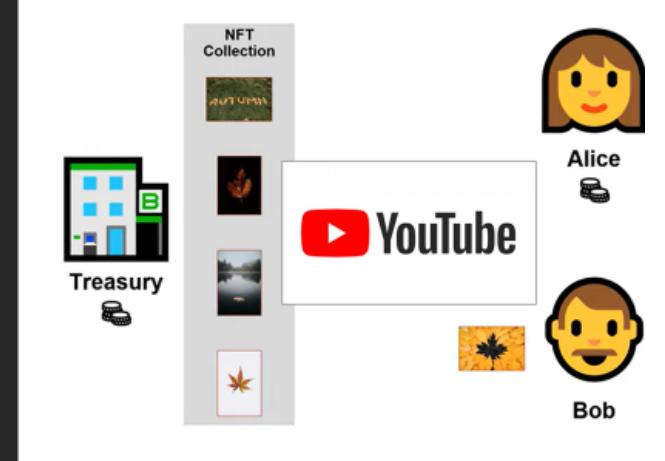
## TokenCreateTransaction() | Non-fungible Token (NFT) Simple

```
{ Client, TokenCreateTransaction, TokenType, TokenSupplyType } = require("@hashgraph/sdk");
client = Client.forTestnet().setOperator(operatorId, operatorKey);

function main() {
  const nftCreate = new TokenCreateTransaction()
    .setTokenName("Fall Collection")
    .setTokenSymbol("LEAF")
    .setTokenType(TokenType.NonFungibleUnique)
    .setInitialSupply(0)
    .setTreasuryAccountId(treasuryId)
    .setMaxSupply(10)
    .setSupplyKey(supplyKey)
    .freezeWith(client);

  const nftCreateTxSign = await nftCreate.sign(treasuryKey);
  const nftCreateSubmit = await nftCreateTxSign.execute(client);
  const nftCreateRx = await nftCreateSubmit.getReceipt(client);
  const tokenId = nftCreateRx.tokenId;
  console.log(`Created NFT with Token ID: ${tokenId}`);
}

)
```



Desarrollo

# En esta sesión aprenderás sobre **registros distribuidos** y como empezar a desarrollar aplicaciones web 3



Introducción

## TokenCreateTransaction() | Non-fungible Token (NFT) Simple

```
{ Client, TokenCreateTransaction, TokenType, TokenSupplyType} = require("@hashgraph/sdk");
client = Client.forTestnet().setOperator(operatorId, operatorKey);

function main() {
  const nftCreate = new TokenCreateTransaction()
    .setTokenName("Fall Collection")
    .setTokenSymbol("LEAF")
    .setTokenType(TokenType.NonFungibleUnique)
    .setInitialSupply(0)
    .setTreasuryAccountId(treasuryId)
    .setMaxSupply(10)
    .setSupplyKey(supplyKey)
    .freezeWith(client);

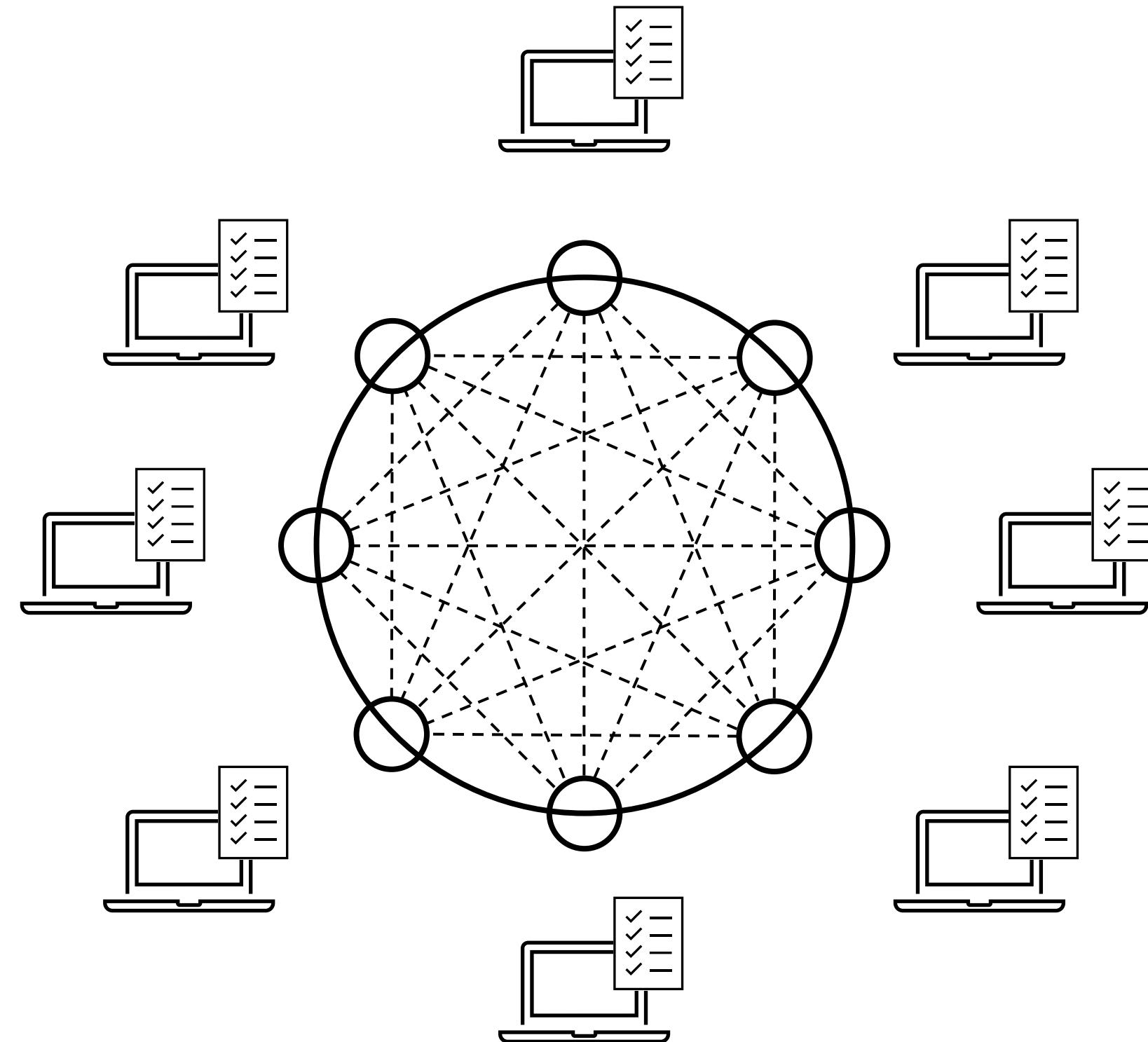
  const nftCreateTxSign = await nftCreate.sign(treasuryKey);
  const nftCreateSubmit = await nftCreateTxSign.execute(client);
  const nftCreateRx = await nftCreateSubmit.getReceipt(client);
  const tokenId = nftCreateRx.tokenId;
  console.log(`Created NFT with Token ID: ${tokenId}`);
}
```

A diagram illustrating the creation of an NFT. It shows a 'Treasury' icon, an 'NFT Collection' icon, and two users, Alice and Bob, interacting with a 'YouTube' video thumbnail. The diagram illustrates how a token is created and stored in a treasury, and then distributed through a collection.

Desarrollo

# Registros distribuidos (DLT) son un componente clave de la Web 3.0 debido a sus cualidades

## REGISTRO DISTRIBUIDO



Toda la red registra y valida cada transacción

## REGISTRO CENTRALIZADO



Autoridad única verifica, registra y ejecuta transacciones

# Intercambio de valor directo – tanto las empresas como las personas realizan transacciones sin intermediarios



# Transparencia – crea aplicaciones más inclusivas sin restricciones de acceso, participación o transacciones



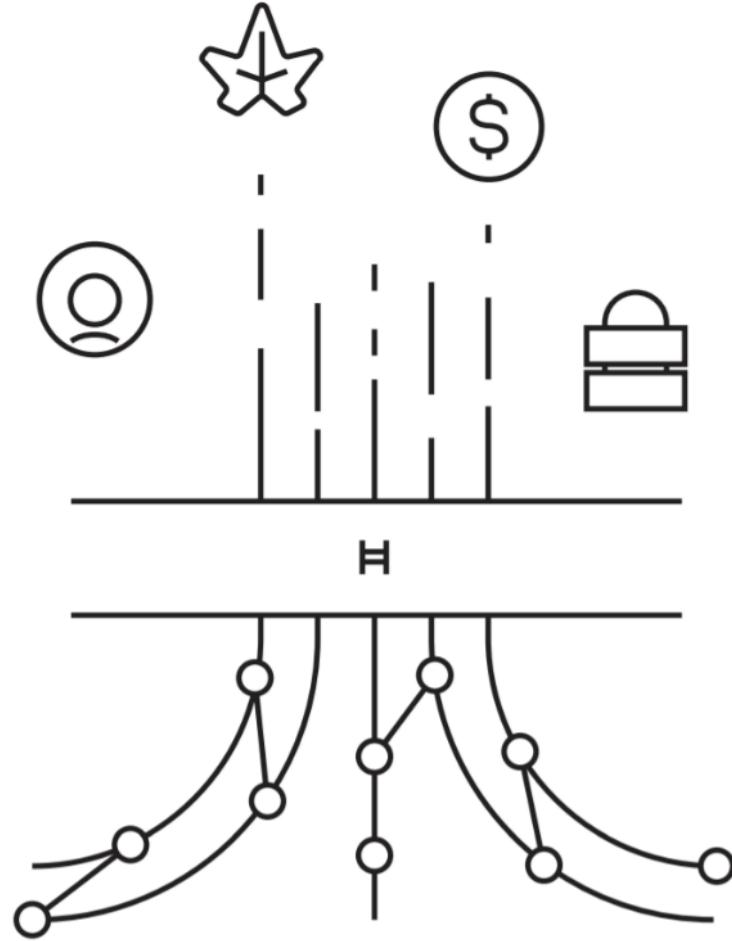
Seguridad – los registros distribuidos son intrínsecamente resistentes a la manipulación porque no hay un punto central de falla para atacar



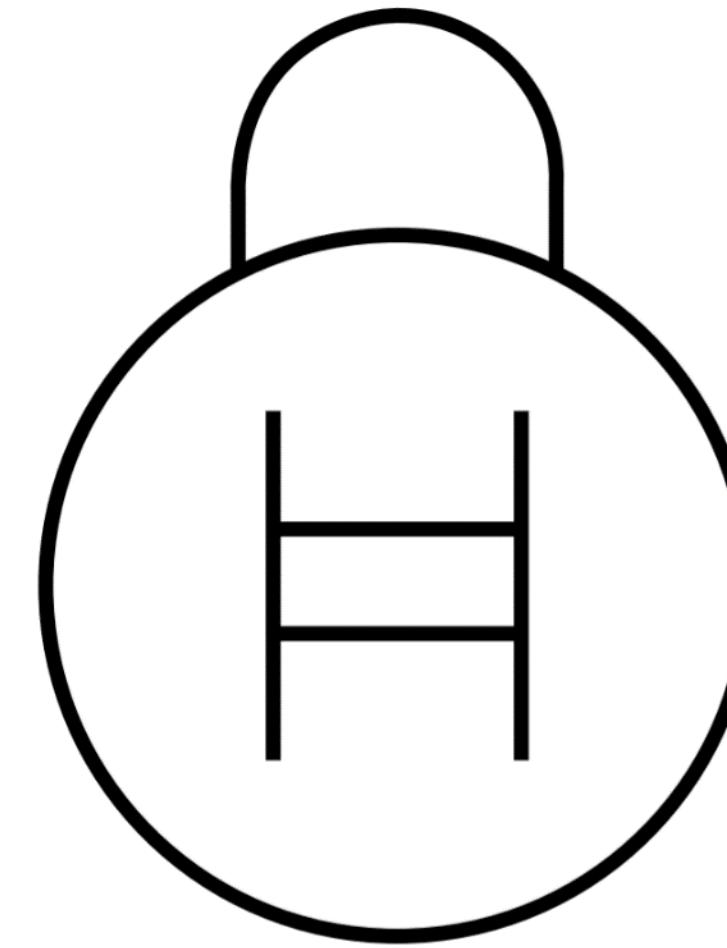
# Los DLTs proporcionan una plataforma de transacciones más rentable, accesible y confiable que los registros centralizados



ELIMINAR AL INTERMEDIARIO  
Y PARTICIPAR EN  
TRANSACCIONES DIRECTAS Y  
DE CONFIANZA



AUMENTAR LA ACCESIBILIDAD  
Y CREAR APLICACIONES  
TRANSPARENTES



PROTEGER EL REGISTRO  
CON MUTABILIDAD  
CONTROLADA Y  
RESISTENCIA A LA  
MANIPULACIÓN

# The open source public ledger for everyone

Powering native web3 ecosystems and  
institutional applications for the next generation  
of the web

[START BUILDING](#)[EXPLORE DOCUMENTATION](#)**0+**

mainnet accounts

**0+**

transactions in the

**0**

seconds to consensus

**\$0**

average cost per

**0**

average kWh per

# Hedera es un registro distribuido público de tercera generación

	1ST GENERACIÓN	2ND GENERACIÓN	3RD GENERACIÓN
TRANSACCIONES POR SEGUNDO	3+ TPS	12+ TPS	10,000+ TPS
COSTO PROMEDIO POR TRANSACCIÓN	\$22.57 USD	\$19.55 USD	\$0.0001 USD
PROMEDIO DE CONFIRMACIÓN A TRANSACCIÓN	10-60 MINUTOS	10-20 SEGUNDOS	3-5 SEGUNDOS (CON FINALIDAD)

• Cryptocurrency transactions. For Hedera, range shown for transactions not requiring a transaction record, but can receive a transaction receipt.

• Avg. Bitcoin tx fee from 6/26/20 - 9/24/20 from <https://blockchair.com/bitcoin/charts/average-transaction-fee-usd?interval=3m>

• Avg. Ethereum tx fee from 6/26/20 - 9/24/20 from <https://blockchair.com/ethereum/charts/average-transaction-fee-usd?interval=3m>

# DLTs públicos de Tercera Generación

	 polygon	 Algorand		 H
Transacciones por segundo	6,500* (claimed)	1,000† (aproximado)	12+	<b>10,000+</b>
Costo promedio (USD)	\$0.0020 (variable)	0.001 Algo (fijo)	\$19.55 (variable)	<b>\$0.0001^ (fijo)</b>
Tiempo de confirmación	5 - 10 segundos (creación de bloque líder)	5 segundos (creación de bloque líder)	10 - 20 segundos (creación de bloque líder)	<b>3-5 segundos (con finalidad)</b>
Use de energía por transacción	90+ kWh	0.00534 kWh	102+ kWh	<b>0.00017 kWh</b>



## 18.000+ DESARROLLADORES

Asistiendo a hackatones globales, reuniones y activos en Discord

## 100+ APLICACIONES

En producción en Hedera Mainnet, desde acceso abierto en septiembre de 2019

## >50.000.000+ TXS/DÍA

Superando por mucho el volumen diario de transacciones de Ethereum

## >2.700.000.000+ TXS A LA FECHA

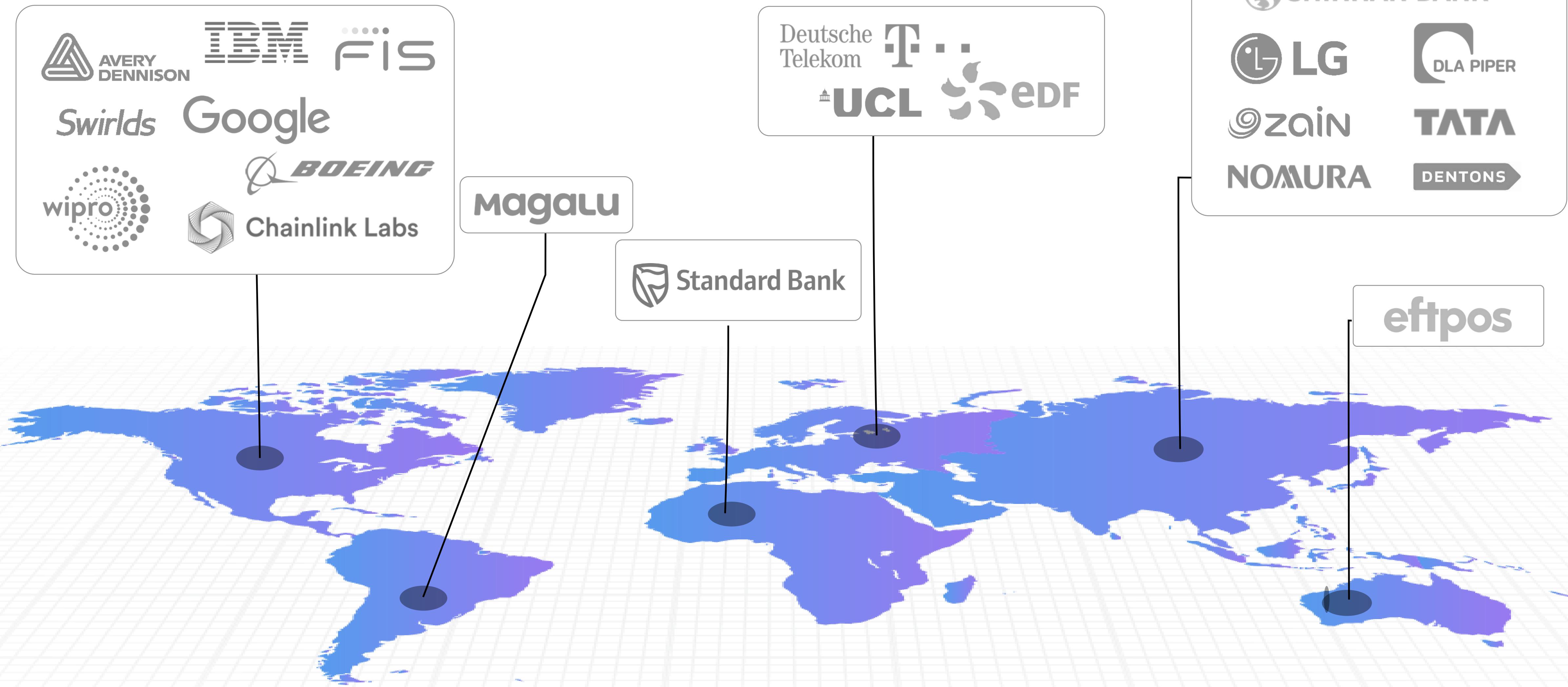
El DLT público más utilizado superando el volumen total de transacciones de Ethereum

## >1.500.000+ TOTAL DE CUENTAS EN MAINNET

Adopción y crecimiento exponencial

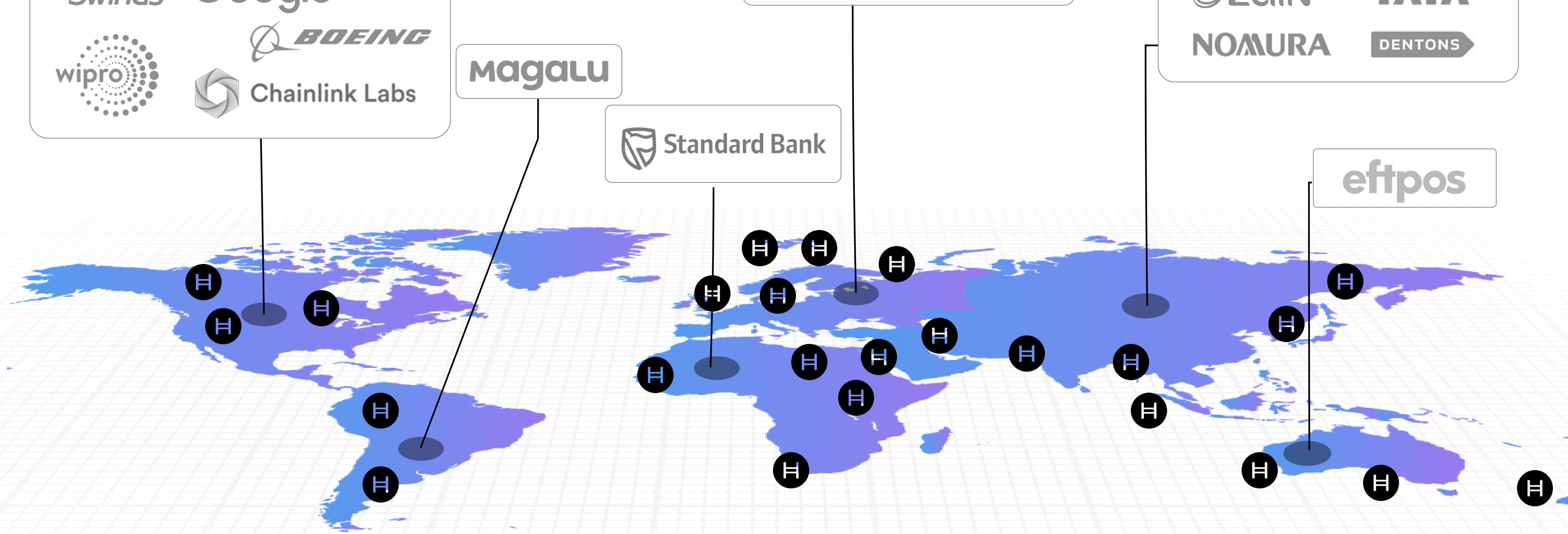
# CRECIMIENTO DE LA RED A LO LARGO DEL TIEMPO

FASE 1: Hasta 39 miembros de consejo



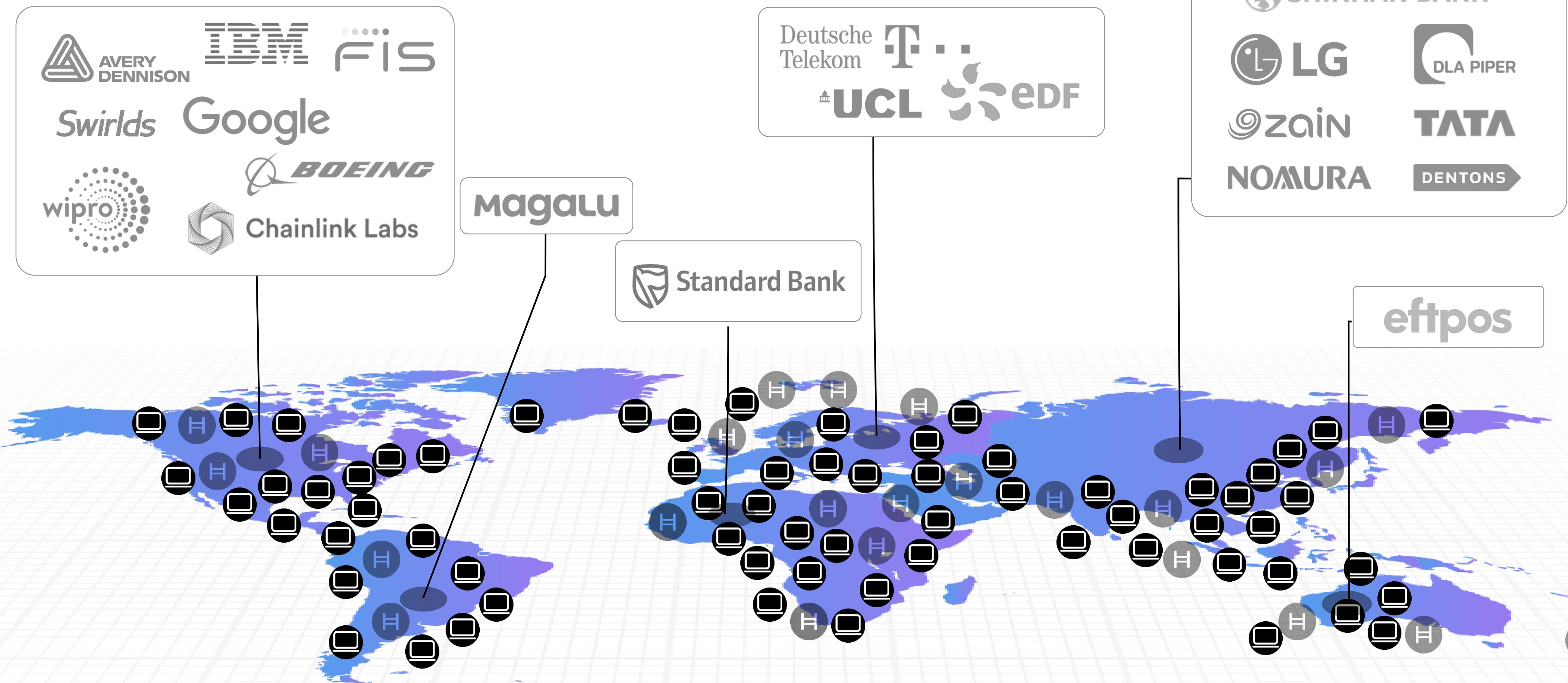
# CRECIMIENTO DE LA RED A LO LARGO DEL TIEMPO

FASE 2: 100s de nodos con permiso y KYC'd

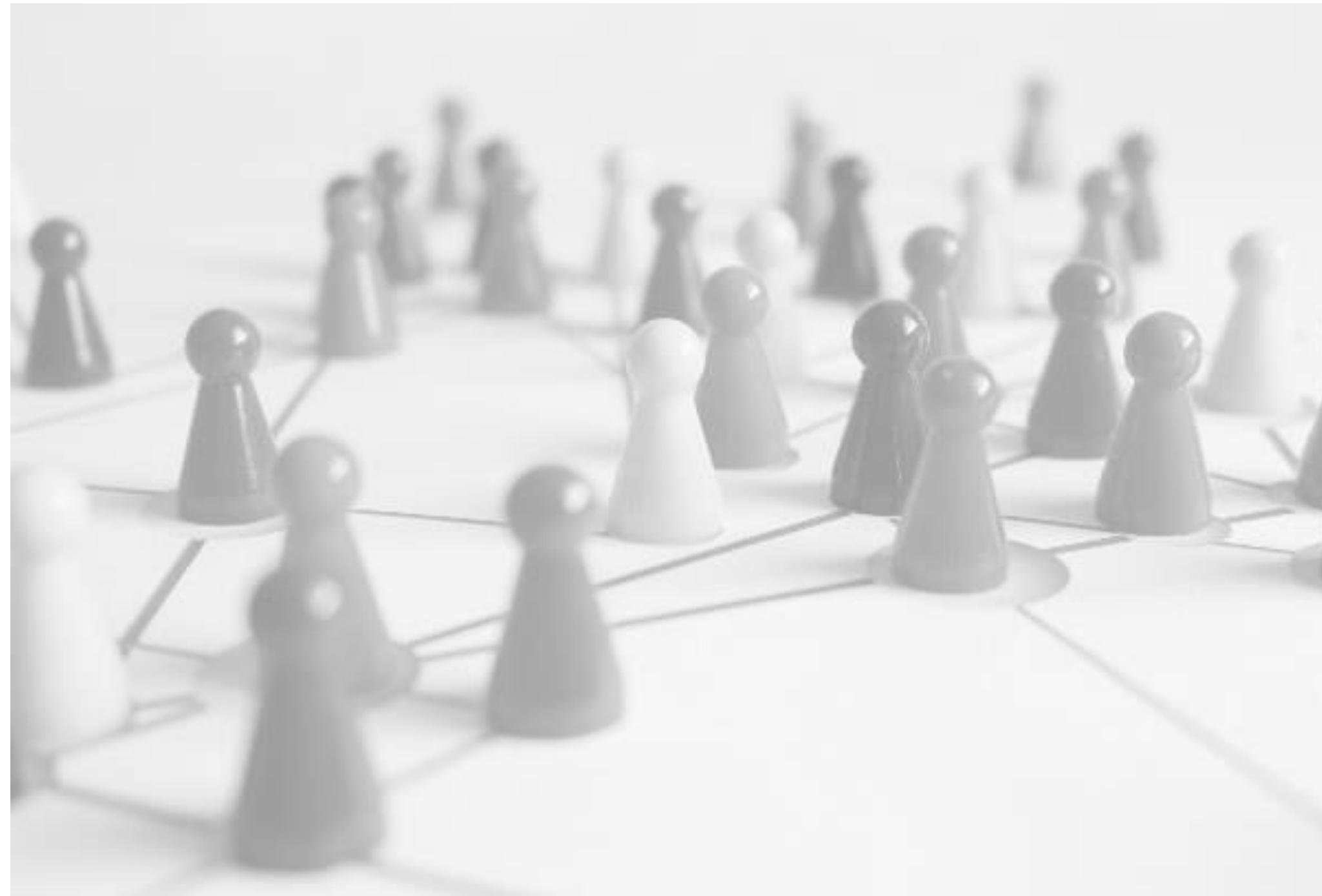


# CRECIMIENTO DE LA RED A LO LARGO DEL TIEMPO

FASE 3: 1000s de nodos sin permiso



# En esta sesión aprenderás sobre **registros distribuidos** y como empezar a desarrollar aplicaciones web 3



Introducción

**TokenCreateTransaction() | Non-fungible Token (NFT) Simple**

```
{ Client, TokenCreateTransaction, TokenType, TokenSupplyType } = require("@hashgraph/sdk");
client = Client.forTestnet().setOperator(operatorId, operatorKey);

function main() {
  const nftCreate = new TokenCreateTransaction()
    .setTokenName("Fall Collection")
    .setTokenSymbol("LEAF")
    .setTokenType(TokenType.NonFungibleUnique)
    .setInitialSupply(0)
    .setTreasuryAccountId(treasuryId)
    .setMaxSupply(10)
    .setSupplyKey(supplyKey)
    .freezeWith(client);

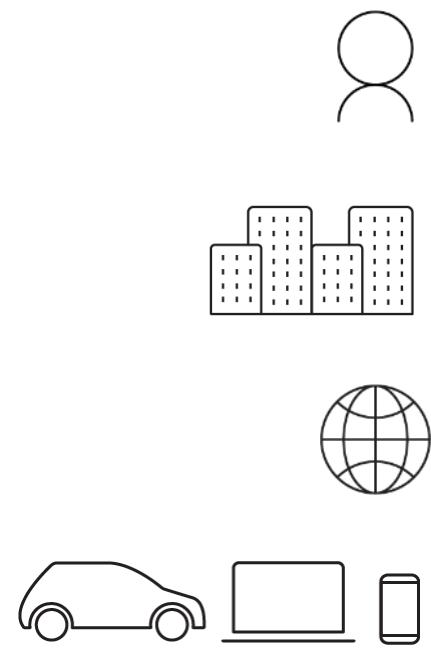
  const nftCreateTxSign = await nftCreate.sign(treasuryKey);
  const nftCreateSubmit = await nftCreateTxSign.execute(client);
  const nftCreateRx = await nftCreateSubmit.getReceipt(client);
  const tokenId = nftCreateRx.tokenId;
  console.log(`Created NFT with Token ID: ${tokenId}`);
}


```

The diagram illustrates the creation of an NFT. It shows a 'Treasury' account containing several NFTs (represented by small icons like a house, a leaf, and a maple leaf). A transaction is being signed by Alice (a woman icon) and submitted to the network. The transaction is then confirmed by Bob (a man icon) on YouTube.

Desarrollo

## USUARIOS



INDIVIDUOS  
EMPRESAS  
GOBIERNOS  
DISPOSITIVOS

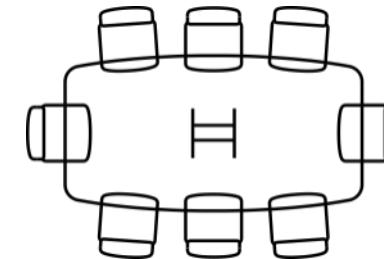
## APLICACIONES Y CASOS DE USO



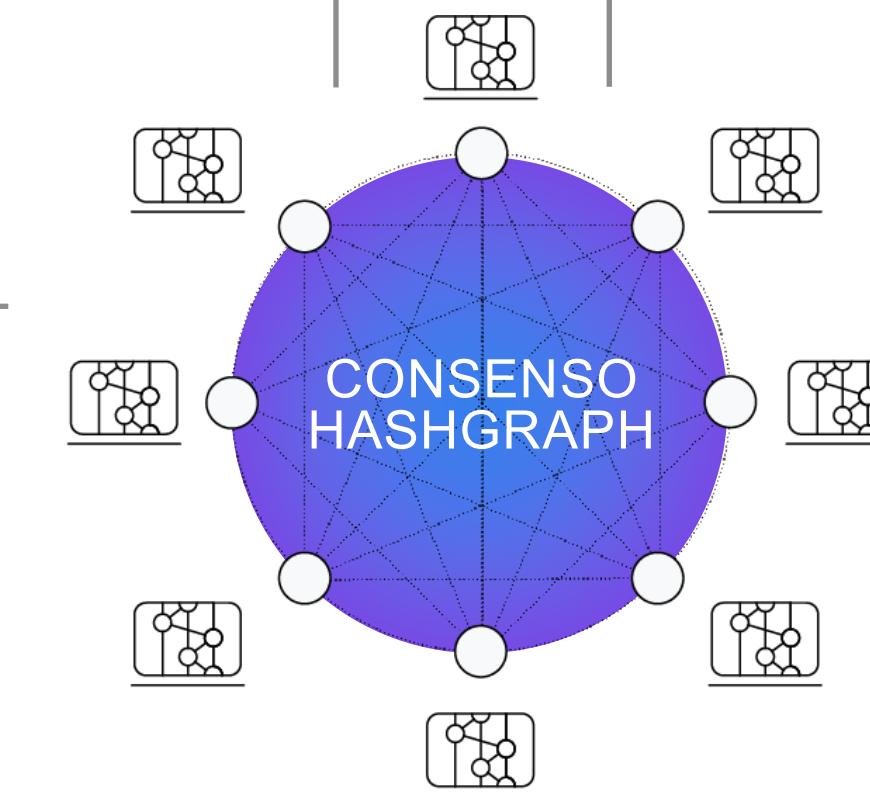
SERVICIO DE CONSENSO

SERVICIO DE TOKENS

CONTRATOS INTELIGENTES

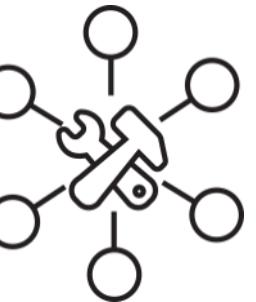


CONSEJO DE GOBIERNO DE HEDERA

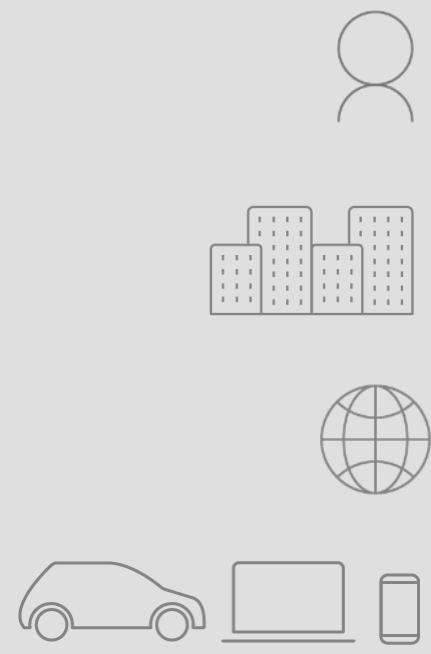


HEDERA MAINNET & NODOS MIRROR

SERVICIOS DE RED PRIMARIOS

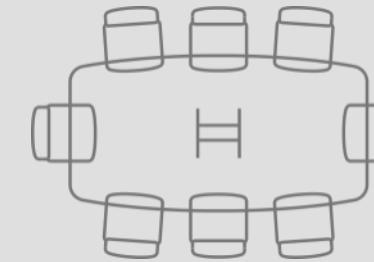
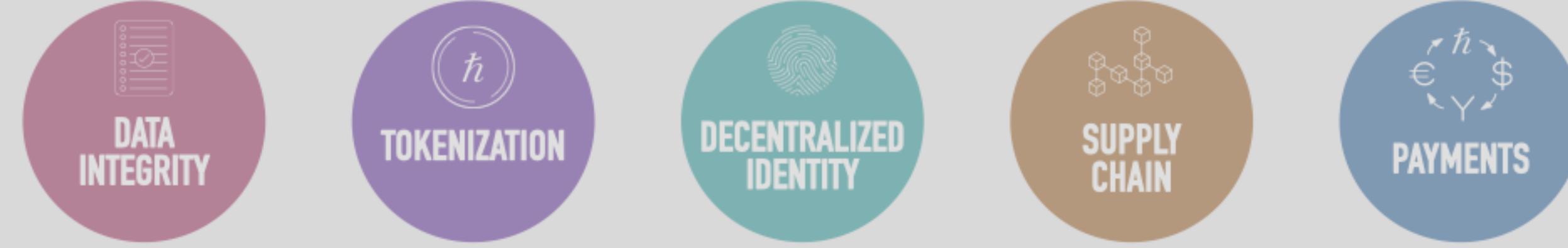


## USUARIOS

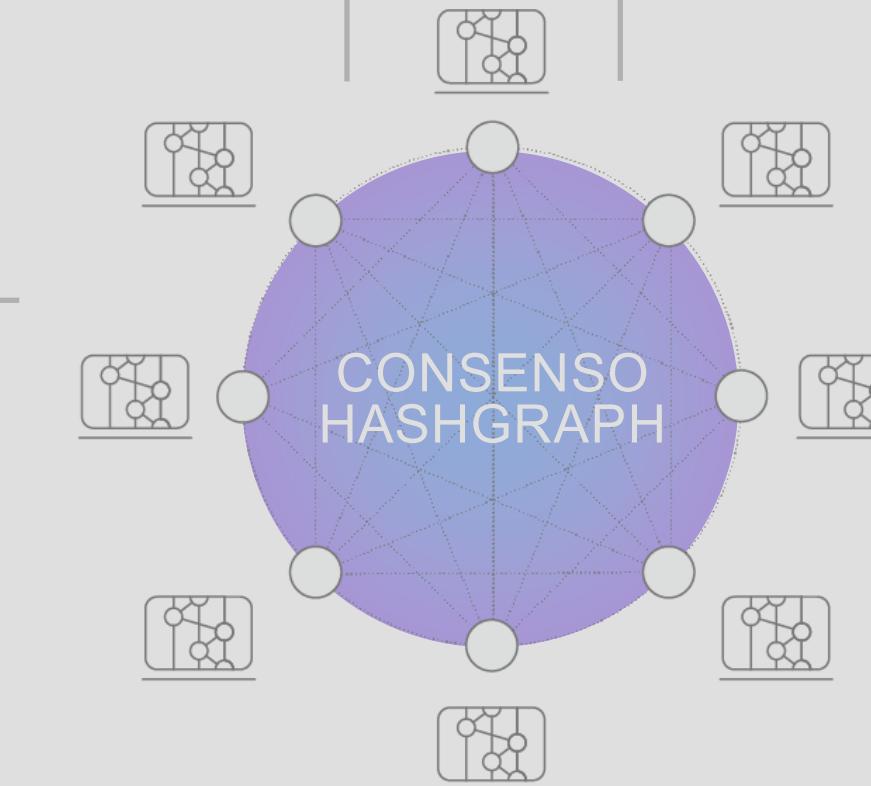


INDIVIDUOS  
EMPRESAS  
GOBIERNOS  
DISPOSITIVOS

## APLICACIONES Y CASOS DE USO



CONSEJO DE GOBIERNO DE HEDERA



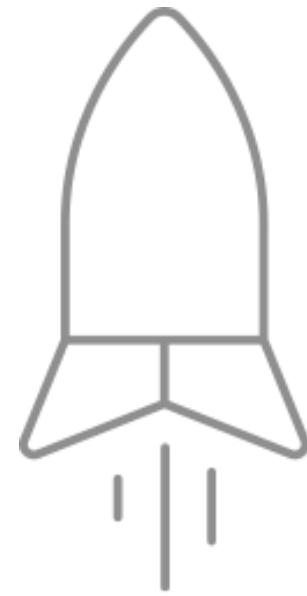
HEDERA MAINNET & NODOS MIRROR

SERVICIOS DE RED PRIMARIOS



¿Por qué construir  
sobre Hedera?

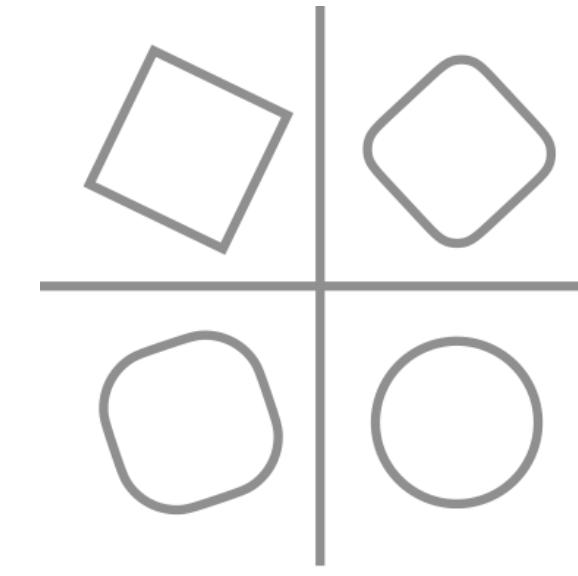
# Hedera te ayuda a cumplir requisitos estrictos en las siguientes áreas...



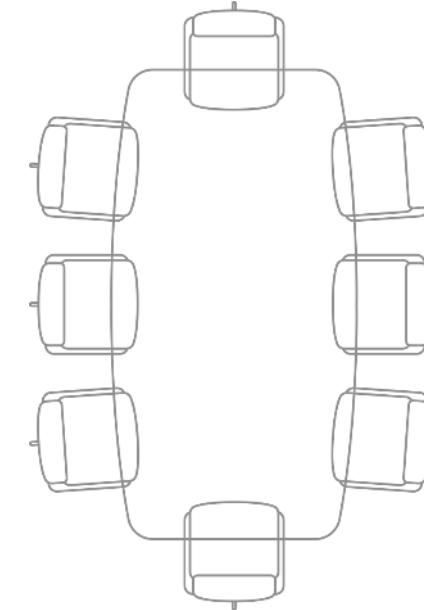
Rendimiento



Seguridad



Estabilidad



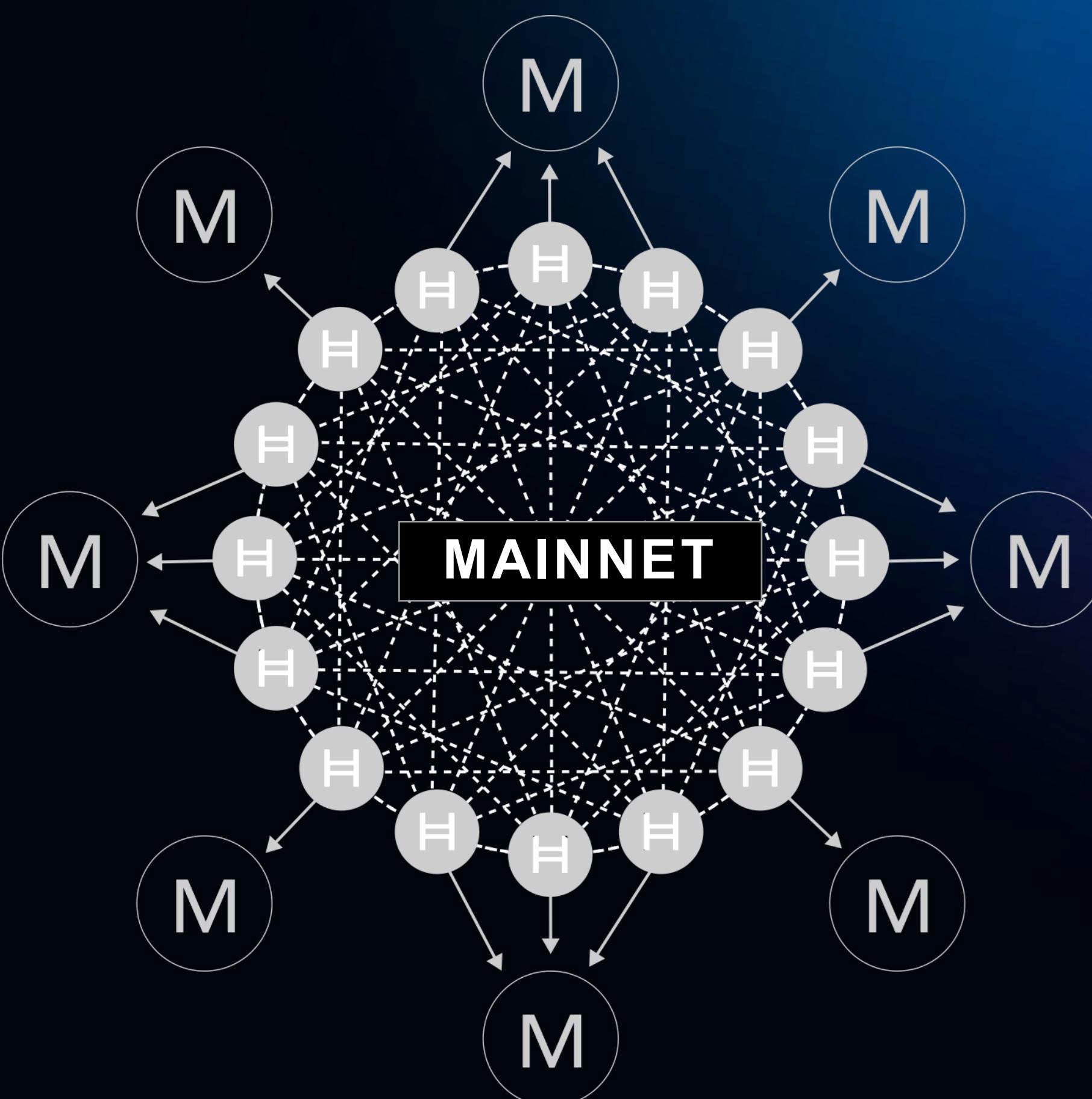
Gobernancia

# HEDERA MAINNET & MIRRORNET



## MAINNET

- Puede enviar transacciones HAPI (Hedera API) a la red Hedera
- Contribuye al consenso sobre las transacciones
- Crea eventos en la red Hedera
- Requiere pago en criptomonedas HBAR para transacciones y consultas



## MIRRORNET

- Mantiene historial del estado parcial o total de la red y un registro de transacciones
- Servicios de valor agregado (nodo de solo lectura administrado, etc.)
- Permite el análisis del estado y transacciones de una aplicación
- Capacidades de publicación y suscripción



Hedera™ Hashgraph



# Cryptocurrency

*“Hedera es la única plataforma que hemos visto que puede hacer frente al volumen de transacciones en fracciones de segundo que deben realizarse.”*

Jiro Olcott | Director | Power Transition

## Transacciones escalables

Finalidad garantizada en 2-5 segundos

10,000 tps en un solo fragment (shard)

## Se utiliza para pagar las tarifas de la red

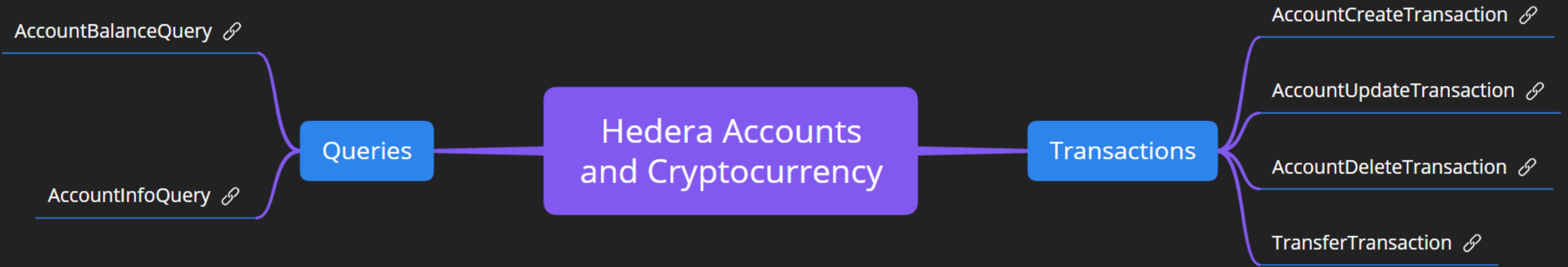
HBAR es la moneda nativa de Hedera

Se utiliza para pagar cada llamada al API

## Mecanismo de seguridad

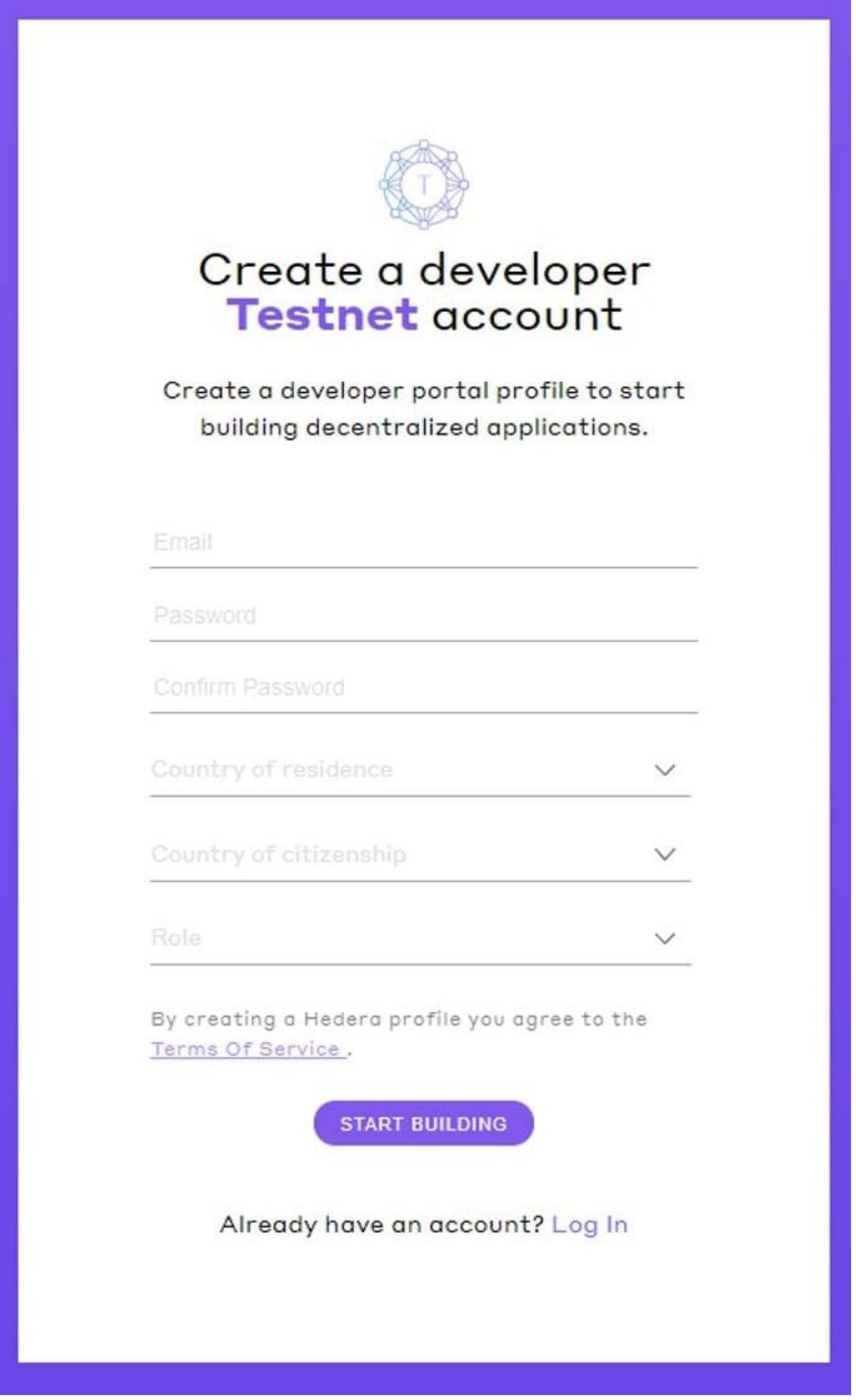
Hedera es una red de prueba de participación  
(Proof of Stake = PoS)

Los actores malintencionados necesitarían controlar 1/3 del suministro total de HBAR para lanzar un ataque

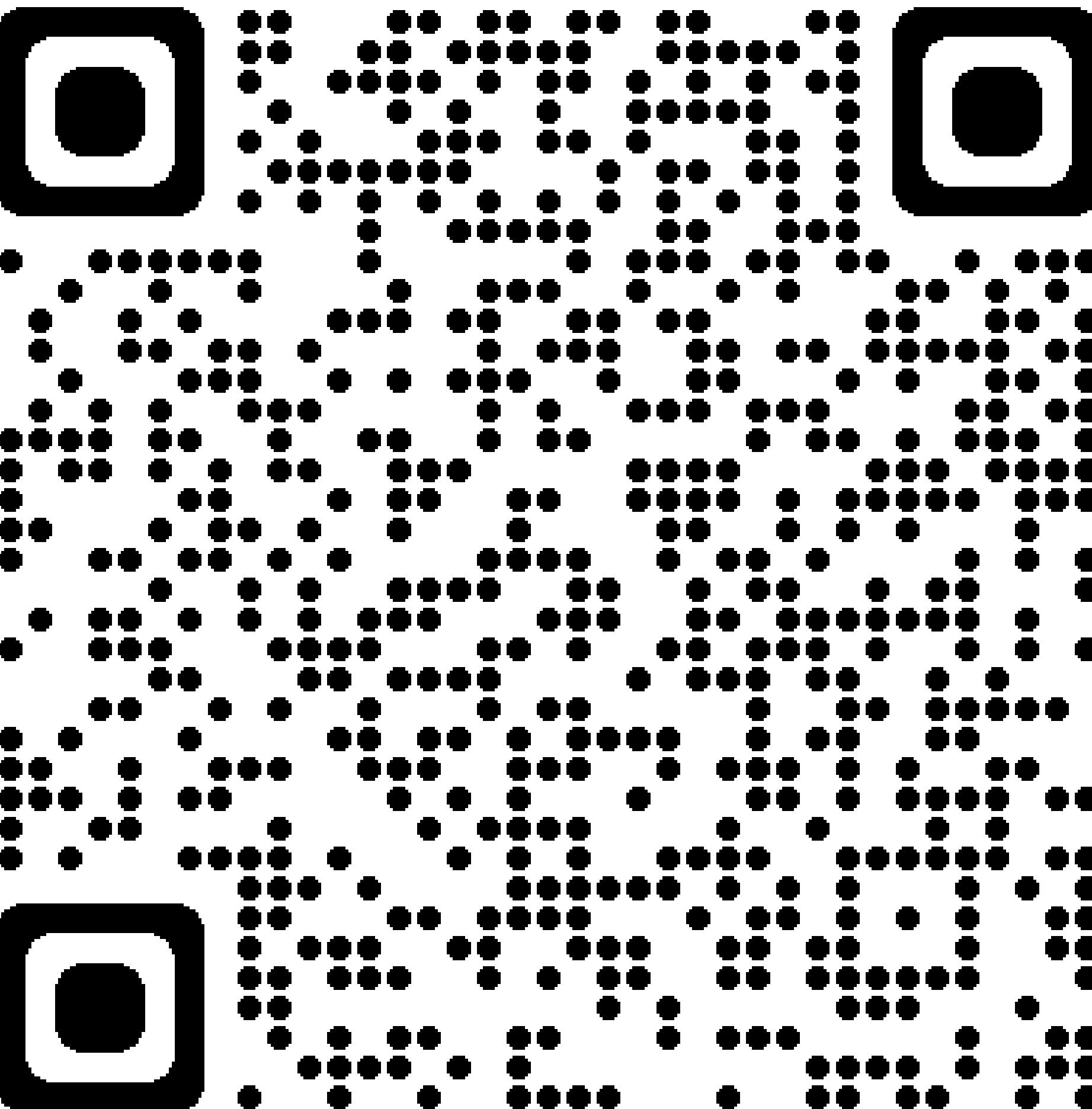


# Conexión a una red

# ¡Abre una cuenta de Testnet! (<https://portal.hedera.com/register>)



The screenshot shows the 'Create a developer Testnet account' page. At the top is a logo consisting of a stylized 'T' inside a circular network graph. Below it is the title 'Create a developer Testnet account'. A sub-instruction reads 'Create a developer portal profile to start building decentralized applications.' The form includes fields for 'Email', 'Password', 'Confirm Password', 'Country of residence' (with a dropdown arrow), 'Country of citizenship' (with a dropdown arrow), and 'Role' (with a dropdown arrow). Below these fields is a note: 'By creating a Hedera profile you agree to the [Terms Of Service](#)'. A purple 'START BUILDING' button is centered below the note. At the bottom, a link says 'Already have an account? [Log In](#)'.



# Conexión a una red (Javascript SDK)...

Paso 1: Crea un archivo .env

```
OPERATOR_ID = 0.0.351...
```

```
OPERATOR_KEY = 302e...
```

Paso 2: Importa las credenciales

```
const operatorId = AccountId.fromString(process.env.OPERATOR_ID);  
const operatorKey = PrivateKey.fromString(process.env.OPERATOR_KEY);
```

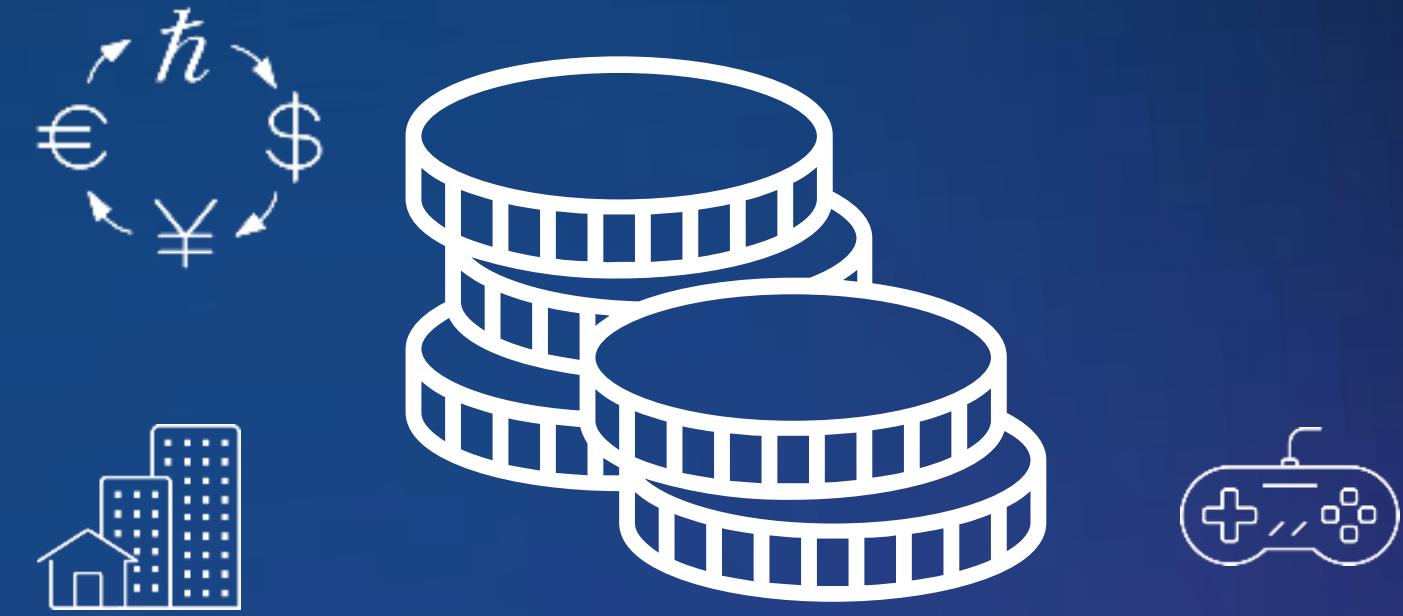
Paso 3: Configura el cliente

```
const client = Client.forTestnet()
```

forMainnet()

forPreviewnet()

```
client.setOperator(operatorId, operatorKey);
```



# Servicio de Tokens

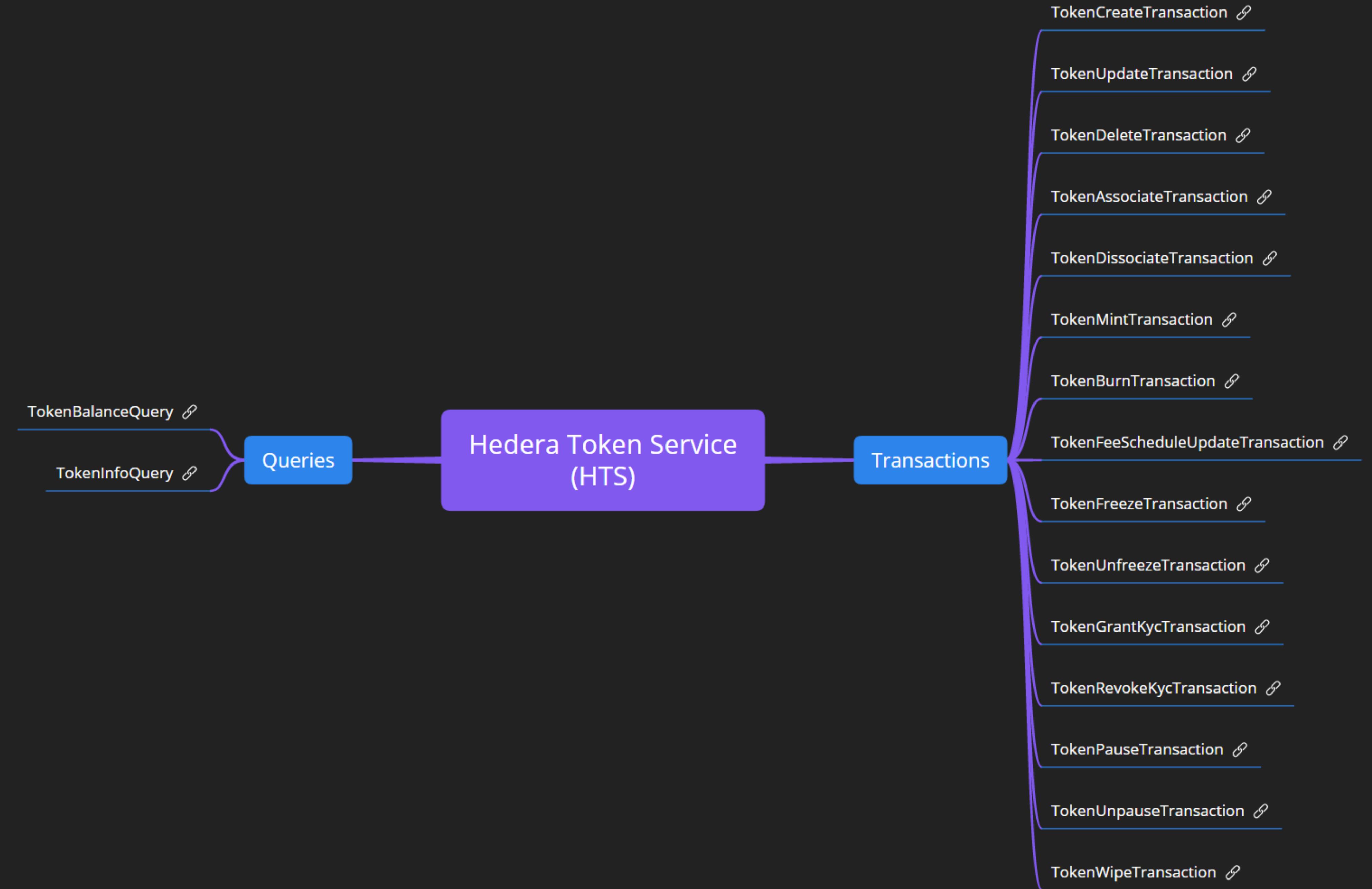
*“para el que ya hemos creado un caso de uso [para nuestros clientes] se trata de la tokenización de activos. El origen de eso proviene de hablar con nuestros clientes comerciales que tienen grandes activos de bienes raíces comerciales que son difíciles de manejar.”*

## Tokenización simplificada

Mintea y administra tokens fungibles y no fungibles (NFT) sin necesidad de un contrato inteligente

## Tokeniza nativamente

Alto rendimiento, configuraciones de conformidad y programabilidad on-chain



(Haga clic en cualquier lugar para abrir en el navegador)

# TokenCreateTransaction() | Non-fungible Token (NFT) Simple

```
const { Client, TokenCreateTransaction, TokenType, TokenSupplyType } = require("@hashgraph/sdk");
const client = Client.forTestnet().setOperator(operatorId, operatorKey);

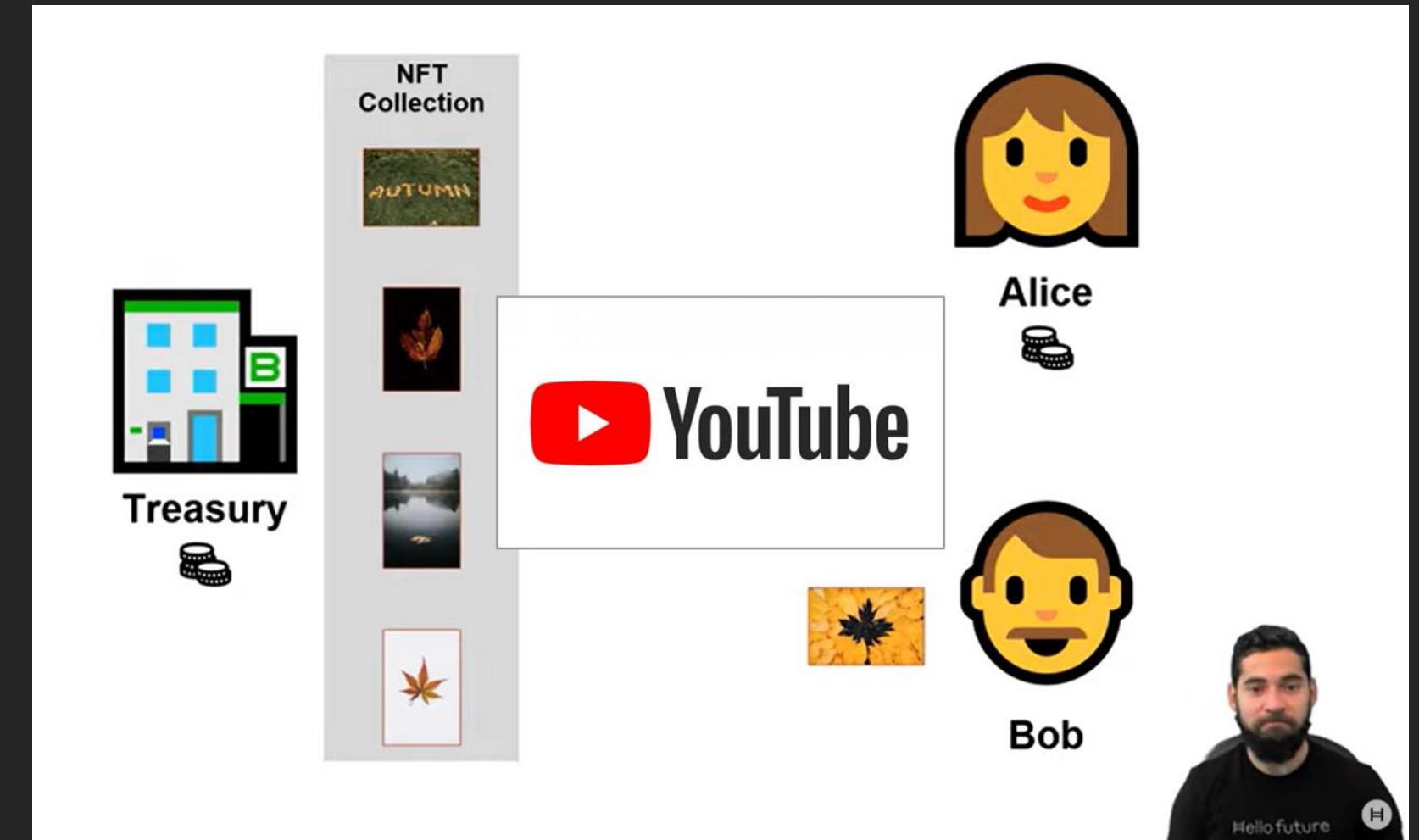
async function main() {

  const nftCreate = new TokenCreateTransaction()
    .setTokenName("Fall Collection")
    .setTokenSymbol("LEAF")
    .setTokenType(TokenType.NonFungibleUnique)
    .setInitialSupply(0)
    .setTreasuryAccountId(treasuryId)
    .setMaxSupply(10)
    .setSupplyKey(supplyKey)
    .freezeWith(client);

  const nftCreateTxSign = await nftCreate.sign(treasuryKey);
  const nftCreateSubmit = await nftCreateTxSign.execute(client);
  const nftCreateRx = await nftCreateSubmit.getReceipt(client);
  const tokenId = nftCreateRx.tokenId;
  console.log(`Created NFT with Token ID: ${tokenId} \n`);

}

main();
```



# TokenCreateTransaction() | Non-fungible Token (NFT) Personalizada

```
const { Client, TokenCreateTransaction, TokenType, TokenSupplyType } = require("@hashgraph/sdk");
client = Client.forTestnet().setOperator(operatorId, operatorKey);

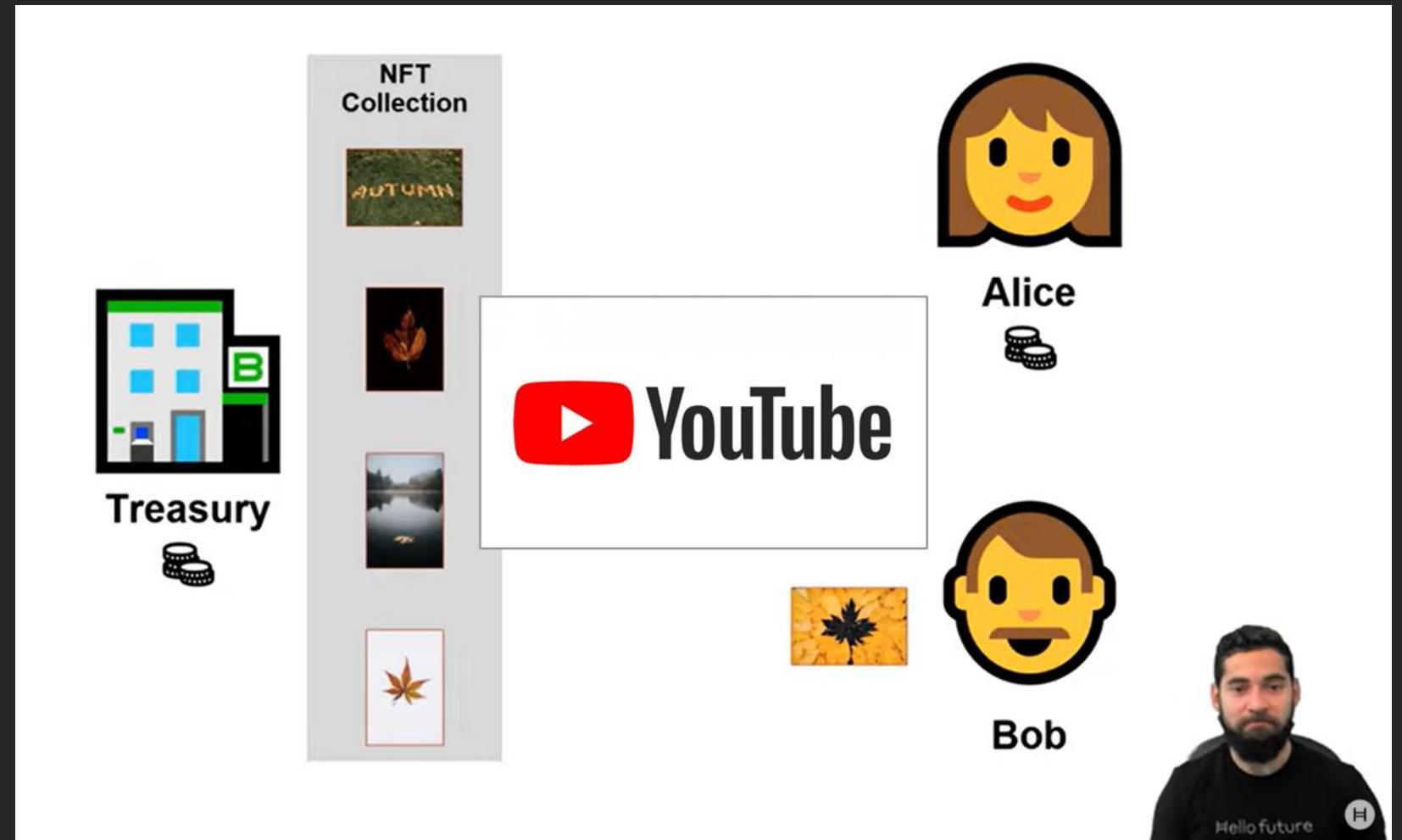
async function main() {

  const nftCreate = await new TokenCreateTransaction()
    .setTokenName("Fall Collection")
    .setTokenSymbol("LEAF")
    .setTokenType(TokenType.NonFungibleUnique)
    .setDecimals(0)
    .setInitialSupply(0)
    .setTreasuryAccountId(treasuryId)
    .setSupplyType(TokenSupplyType.Finite)
    .setMaxSupply(CID.length)
    .setCustomFees([nftCustomFee])
    .setAdminKey(adminKey)
    .setSupplyKey(supplyKey)
    .setPauseKey(pauseKey)
    .setFreezeKey(freezeKey)
    .setWipeKey(wipeKey)
    .freezeWith(client)
    .sign(treasuryKey);

  const nftCreateTxSign = await nftCreate.sign(adminKey);
  const nftCreateSubmit = await nftCreateTxSign.execute(client);
  const nftCreateRx = await nftCreateSubmit.getReceipt(client);
  const tokenId = nftCreateRx.tokenId;
  console.log(`Created NFT with Token ID: ${tokenId} \n`);

}

main();
```





# Contratos Inteligentes

*“Hedera tiene ventajas obvias con respecto a otras blockchains L1, incluyendo la velocidad, la confianza y el costo, y nos permite pensar con bastante franqueza y ser creativos sobre lo que puede ser un DEX y qué tipo de funcionalidad podemos ofrecer a nuestros usuarios.”*

Pete Campbell | Co-Founder & CEO | SaucerSwap

## Construye Usando Solidity

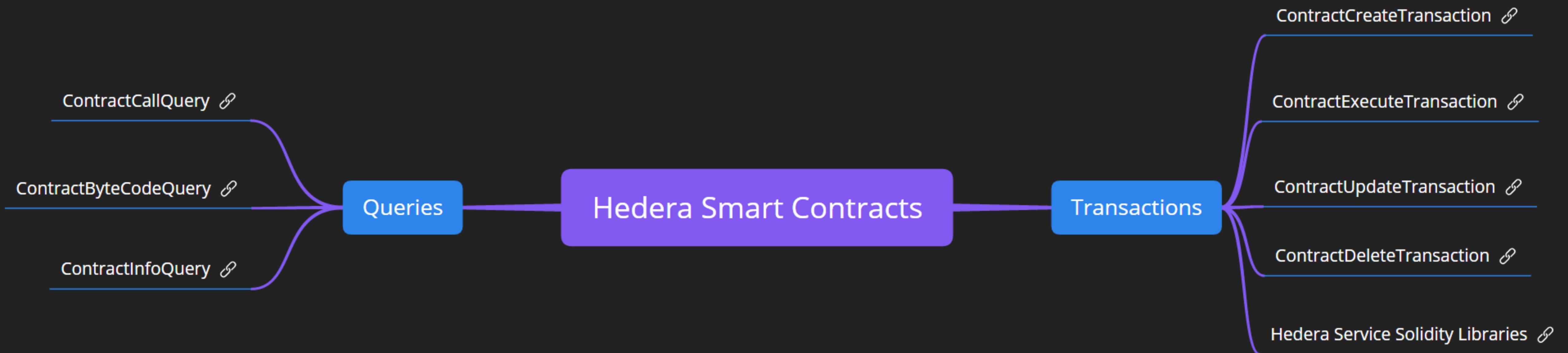
Ejecuta contratos inteligentes de Solidity, sin cambios y usando estándares existentes

## Orden Justo de Transacciones

Como todos los servicios, los contratos inteligentes se ejecutan por orden de recepción, nunca por la cantidad de gas pagada. No es necesario pagar extra para ser incluido antes en la historia

## Privilegios Administrativos

Define transparentemente las llaves administrativas de un contrato, lo que permite a los propietarios realizar modificaciones, a veces drásticamente necesarias, en un contrato que de otro modo sería inmutable... O no, ¡depende de su implementación!

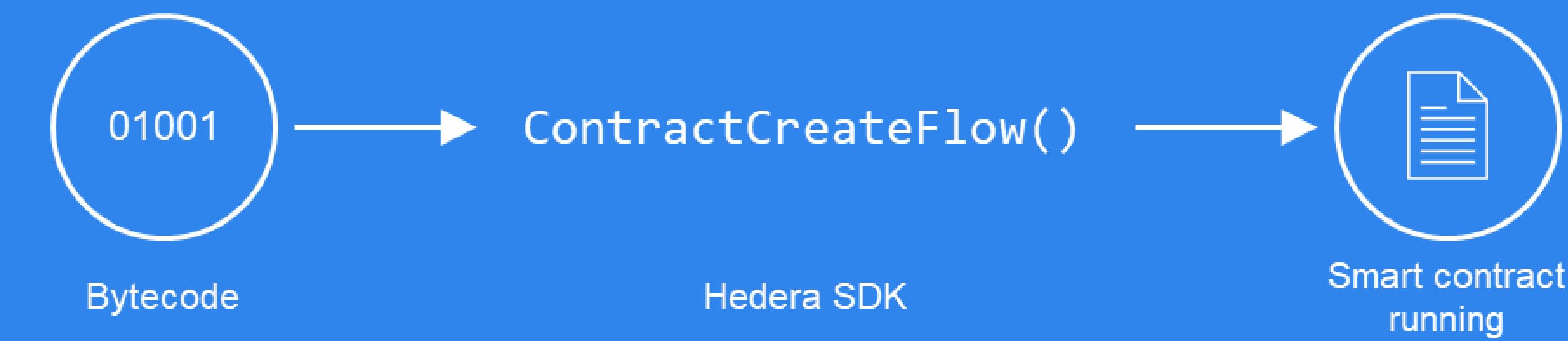


(Haga clic en cualquier lugar para abrir en el navegador)

## 1 Compile to bytecode



## 2 Create smart contract



## 3 Call smart contract





## *LookupContract.sol*

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

contract LookupContract{

mapping (string => uint) public myDirectory;

constructor (string memory _name, uint _mobileNumber) public {
    myDirectory[_name] = _mobileNumber;
}

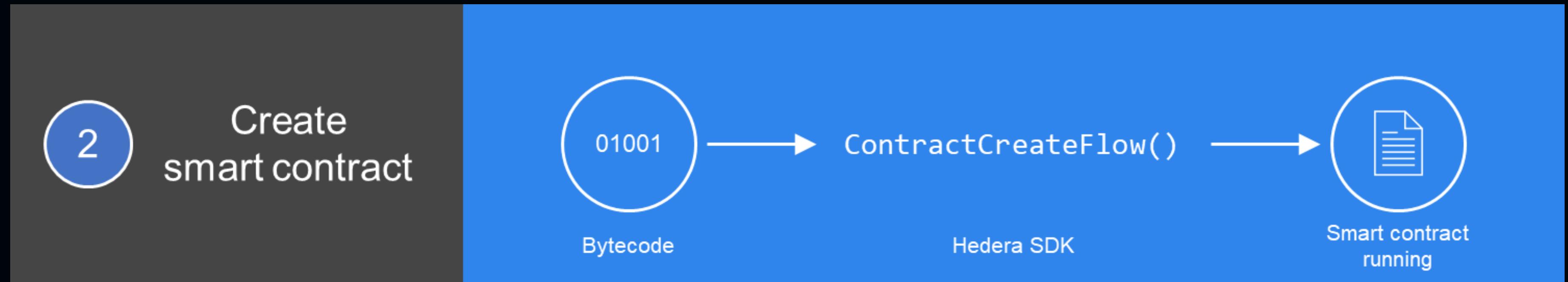
function setMobileNumber(string memory _name, uint _mobileNumber) public{
    myDirectory[_name] = _mobileNumber;
}

function getMobileNumber(string memory _name) public view returns(uint){
    return myDirectory[_name];
}
}
```

## JavaScript SDK

```
// Import the compiled contract bytecode
const contractBytecode = fs.readFileSync("LookupContract_sol_LookupContract.bin");
```





## JavaScript SDK

```
// Import the compiled contract bytecode
const contractBytecode = fs.readFileSync("LookupContract.bin");

// Instantiate the smart contract
const contractInstantiateTx = new ContractCreateFlow()
  .setBytecode(contractBytecode)
  .setGas(100000)
  .setConstructorParameters(
    new ContractFunctionParameters().addString("Alice").addUint256(111111)
  );
const contractInstantiateSubmit = await contractInstantiateTx.execute(client);
const contractInstantiateRx = await contractInstantiateSubmit.getReceipt(client);
const contractId = contractInstantiateRx.contractId;
const contractAddress = contractId.toSolidityAddress();
console.log(`- The smart contract ID is: ${contractId} \n`);
console.log(`- The smart contract ID in Solidity format is: ${contractAddress} \n`);
```



## JavaScript SDK

```
// Query the contract to check changes in state variable
const contractQueryTx = new ContractCallQuery()
  .setContractId(contractId)
  .setGas(100000)
  .setFunction("getMobileNumber", new ContractFunctionParameters().addString("Alice"));
const contractQuerySubmit = await contractQueryTx.execute(client);
const contractQueryResult = contractQuerySubmit.getUint256(0);
console.log(` - Here's the phone number that you asked for: ${contractQueryResult} \n`);
```



```
// Call contract function to update the state variable
const contractExecuteTx = new ContractExecuteTransaction()
  .setContractId(contractId)
  .setGas(100000)
  .setFunction("setMobileNumber", new ContractFunctionParameters().addString("Bob").addUint256(222222));
const contractExecuteSubmit = await contractExecuteTx.execute(client);
const contractExecuteRx = await contractExecuteSubmit.getReceipt(client);
console.log(`- Contract function call status: ${contractExecuteRx.status} \n`);

// Query the contract to check changes in state variable
const contractQueryTx1 = new ContractCallQuery()
  .setContractId(contractId)
  .setGas(100000)
  .setFunction("getMobileNumber", new ContractFunctionParameters().addString("Bob"));
const contractQuerySubmit1 = await contractQueryTx1.execute(client);
const contractQueryResult1 = contractQuerySubmit1.getUint256(0);
console.log(`- Here's the phone number that you asked for: ${contractQueryResult1} \n`);
```

# Console Output

- The smart contract ID is: 0.0.30468614
  - The smart contract ID in Solidity format is: 000000000000000000000000000000001d0ea06

---

  - Here's the phone number that you asked for: 111111
  - Contract function call status: SUCCESS
  - Here's the phone number that you asked for: 222222

# ¡Aprenda más y comienza a construir web 3!

For developers

Integrate Hedera Hashgraph into your applications and microservices.

**Quickstart**  
Get your development environment set up in under 5 minutes with Java, JavaScript, or Go.  
[GET STARTED](#)

**Tutorials**  
Learn more about building on Hedera Hashgraph through a hands-on-tutorial.  
[GET STARTED](#)

**Starter projects**  
Reduce your development time by getting set up in a starter project using familiar frameworks.  
[GET STARTED](#)

**What is Hedera?**

Hedera is the only public distributed ledger that utilizes the fast, fair, and secure hashgraph consensus mechanism. Hedera's governance is fully decentralized, consisting of up to 39 very-limited and highly diversified leading organizations and enterprises.

**Distributed ledger technologies** **Hedera** **Cryptocurrencies**

**Hedera Hashgraph Explained**

Hedera is a public distributed ledger and governing body built from the ground-up to support new and existing applications running at web scale. Developers use distributed ledger technologies to build computational trust directly into their applications. This allows individuals and businesses who might not know or trust each other to quickly and inexpensively collaborate. Public distributed ledgers allow for creating and exchanging value, proving identity, verifying and authenticating important data, and much more.

Hedera is unique in that it achieves the same result as the most ubiquitous public blockchains (such as Bitcoin or Ethereum), but in a way that is faster, fairer, and more energy efficient, stable, and secure – these advantages can be attributed to the underlying hashgraph consensus algorithm and the global enterprise governing body, which owns and operates Hedera today.

AFTER READING THIS, YOU'LL UNDERSTAND:

- The basics of hashgraph consensus
- Network services offered by Hedera
- Decentralized governing council structure and members
- Examples of third-party applications built on Hedera
- Hedera's path to decentralization

RELATED TOPICS:

Logos of various partners and supporters: Avery Dennison, Boeing, Chainlink Labs, DBS, Dentons, Deutsche Telekom, EDF, eftpos, FIS, Google, IBM, IIT Madras, LG, LSE, Magalu, Nomura, ServiceNow.

**Welcome to Hedera**

This is the beginning of this server.

February 3, 2018

**KenTheJr** 02/03/2018

Hashgraph is a data structure and consensus algorithm that is:

- Fast:** With a very high throughput and low-latency consensus
- Secure:** Asynchronous Byzantine fault tolerant (ABFT)
- Fair:** Fairness of access, ordering, and timestamps

These properties enable new decentralized applications such as a stock market, improved collaborative applications, games, and auctions.

**Papers**

- Whitepaper: <http://www.swirls.com/downloads/SWIRLDS-TR-2016-01.pdf>
- How It Works: <http://www.swirls.com/downloads/SWIRLDS-TR-2016-02.pdf>
- Hashgraph Overview: <http://www.swirls.com/downloads/Overview-of-Swirls-Hashgraph.pdf>
- Hashgraph & Sybil Attacks: <http://www.swirls.com/downloads/Swirls-and-Sybil-Attacks.pdf>
- Dictatorships, Democracy, and Blockchain: <http://www.swirls.com/downloads/Dictatorships-Democracy-and-Blockchain.pdf>

[hedera.com/get-started](https://hedera.com/get-started)

[Centro de Aprendizaje  
Hedera](#)

[Únete al Discord de  
desarrolladores](#)

Green Energy / Energy Conscious Dapps

VIEW FULL PLAYLIST

DLT and Blockchain in the Real World

VIEW FULL PLAYLIST

Engineering Insights

VIEW FULL PLAYLIST

Hedera21 Hello Tokenization Virtual Hackathon

VIEW FULL PLAYLIST

Coding With Cooper

VIEW FULL PLAYLIST

Working at Hedera

VIEW FULL PLAYLIST

Community Town Halls

PLAY ALL

Virtual Meetups

VIEW FULL PLAYLIST

Forkless Cafe - Social Community Live Stream

VIEW FULL PLAYLIST

Gossip About Gossip Podcasts

Updated 6 days ago

VIEW FULL PLAYLIST

Hedera Hashgraph Dapp Talk

VIEW FULL PLAYLIST

Hedera Developer Workshop

VIEW FULL PLAYLIST

Hedera Governing Council

VIEW FULL PLAYLIST

Dapp Spotlights

VIEW FULL PLAYLIST

Hedera Hashgraph Webinars

VIEW FULL PLAYLIST

[Canal de YouTube de Hedera](#)

**Hedera Token Service Demo**

ISSUER (0.0.283791) ALICE (0.0.283792) BOB (0.0.283793)

Accounts associated with token DTT2

**Issuer (0.0.283791)**

Token Balance: 9950 hBar Balance: 84.36257751 h

Frozen  KYCd

**Alice (0.0.283792)**

Token Balance: 20 hBar Balance: 108.68696976 h

Frozen  KYCd

**Bob (0.0.283793)**

Token Balance: 30 hBar Balance: 88.71117156 h

Frozen  KYCd

**Marketplace (0.0.283794)**

Token Balance: 0 hBar Balance: 98.71117146 h

Frozen  KYCd

**FREEZE GRANT KYC WIPE**

[Demostraciones de Aplicaciones](#)

# Web 3 está lista para ti, y con Hedera, ¡tú estás list@ para Web 3!

