**Hackathon Blockchain**

**Abordaje del Reto**

Creación de una Red Blockchain Privada

Uso de Python: Para simplificar el desarrollo y la demostración, implementamos una red blockchain privada utilizando Python en lugar de plataformas más complejas como Hyperledger Fabric o Ethereum.

Clase Blockchain: Implementamos una clase Blockchain en Python que gestiona la cadena de bloques, permitiendo añadir bloques y transacciones de manera segura y transparente.

Desarrollo de una Aplicación de Donaciones

Flask: Utilizamos Flask, un microframework de Python, para crear una API REST que maneja las operaciones de la blockchain y la interacción con la aplicación de donaciones.

Endpoints: Se crearon varios endpoints para gestionar la cadena de bloques, añadir transacciones, establecer límites de donaciones y registrar nuevas ONGs.

Implementación de Características de Transparencia

Transacciones en Blockchain: Todas las donaciones se registran como transacciones en la blockchain. Cada transacción incluye información del donante, beneficiario y la cantidad donada, asegurando que los fondos sean rastreables y verificables.

Hash y Previous Hash: Cada bloque contiene un hash y un hash del bloque anterior, lo que garantiza la integridad y la inmutabilidad de la cadena.

Provisión de una Solución de Informes

Endpoints para Informes: Se implementaron endpoints que permiten consultar el total de donaciones realizadas y el límite de donaciones establecido. También se creó una funcionalidad para excluir transacciones internas (como las realizadas por la plataforma) en los informes.

Interfaz de Usuario: Utilizando Bootstrap y jQuery, se desarrolló una interfaz de usuario que muestra la cadena de bloques y las transacciones de manera clara y comprensible. Los usuarios pueden ver las transacciones detalladas de cada bloque y el estado actual de las donaciones en comparación con el límite establecido.

**Detalles tecnicos**

Endpoint: **Crear una nueva transacción**

**Ruta**: /transaction/new

**Método**: POST

**Funcionalidad**: Permite que un usuario realice una nueva donación.

**Parámetros Esperados**: Un JSON con los campos donor (el donante), beneficiary (el beneficiario de la donación) y amount (la cantidad donada).

**Respuestas**:

201 Created: La transacción fue añadida exitosamente.

400 Bad Request: Si faltan valores en la solicitud o si la donación excede el límite establecido.

Endpoint: **Obtener la cadena completa**

**Ruta**: /chain

**Método**: GET

**Funcionalidad**: Devuelve toda la cadena de bloques.

**Parámetros Esperados:** Ninguno.

**Respuestas:**

200 OK: Devuelve un JSON que contiene toda la cadena de bloques y su longitud.

Endpoint: **Obtener información de donaciones**

**Ruta**: /donation\_info

**Método**: GET

**Funcionalidad**: Proporciona información sobre la cantidad total de donaciones realizadas, excluyendo las donaciones internas de la plataforma, y el límite de donaciones establecido.

**Parámetros Esperados**: Ninguno.

**Respuestas:**

200 OK: Devuelve un JSON con la cantidad total donada y el límite de donaciones.

Endpoint: **Agregar una nueva ONG**

**Ruta**: /ongs/new

**Método**: POST

**Funcionalidad**: Permite registrar una nueva ONG (organización benéfica).

**Parámetros Esperados**: Un JSON con el campo name (nombre de la ONG).

**Respuestas**:

201 Created: La ONG fue agregada exitosamente.

400 Bad Request: Si falta el nombre de la ONG en la solicitud.

Endpoint: **Establecer el límite de donaciones**

**Ruta**: /limit

**Método**: POST

**Funcionalidad**: Establece un límite para el total de donaciones permitidas.

**Parámetros Esperados**: Un JSON con el campo limit (límite de donaciones).

**Respuestas**:

201 Created: El límite de donaciones fue establecido exitosamente.

400 Bad Request: Si falta el valor del límite en la solicitud.

Endpoint: **Establecer el límite de donaciones**

**Ruta**: /limit

**Método**: POST

**Funcionalidad**: Establece un límite para el total de donaciones permitidas.

**Parámetros Esperados**: Un JSON con el campo limit (límite de donaciones).

**Respuestas**:

201 Created: El límite de donaciones fue establecido exitosamente.

400 Bad Request: Si falta el valor del límite en la solicitud

Endpoint: **Minar un nuevo bloque**

**Ruta**: /mine

**Método**: GET

**Funcionalidad**: Minar un nuevo bloque en la cadena. Esto implica:

Resolver el problema de prueba de trabajo (proof of work).

Crear una nueva transacción de recompensa por la minería.

Crear y añadir el nuevo bloque a la cadena.

**Parámetros Esperados**: Ninguno.

**Respuestas**:

200 OK: Devuelve un JSON con la información del nuevo bloque minado.

Endpoint: **Distribuir las donaciones entre las ONGs**

**Ruta**: /distribute

**Método**: POST

**Funcionalidad**: Distribuir una cantidad especificada de donaciones entre todas las ONGs registradas. Esto implica:

Verificar que la cantidad no exceda el total de donaciones.

Dividir el monto entre todas las ONGs registradas.

Crear una transacción para cada ONG con su parte correspondiente.

Minar un nuevo bloque que incluya estas transacciones.

**Parámetros Esperados**: Un JSON con el campo amount (la cantidad a distribuir).

**Respuestas:**

200 OK: Devuelve un JSON con la información de la distribución y las transacciones incluidas en el nuevo bloque.

400 Bad Request: Si faltan valores en la solicitud, si la cantidad excede el total de donaciones, o si no hay ONGs registradas.

**Uso de la Aplicación**

1. Establece el límite de donaciones mediante formulario de establecer limite que contenga el valor del límite.
2. Registra nuevas ONGs mediante formulario registrar ong
3. Realiza una donación mediante formulario de donaciones que contenga el donante, el beneficiario y la cantidad.
4. Mina un nuevo bloque mediante botón validar donación Esto confirmará todas las transacciones pendientes en la cadena de bloques.
5. Distribuye una cantidad especificada de donaciones entre todas las ONGs registradas mediante el botón distribuir donaciones después se valida la donación.