UNIVERSIDAD DE MONTEVIDEO

FACULTAD DE INGENIERÍA

Blockchain - Plataforma de Crowdfunding

Juan Pedro Grille

Juan Martín Staricco

Joaquin Fernandez

1. Descripción del problema

El objetivo de este proyecto es diseñar e implementar una plataforma de crowdfunding basada en blockchain, que permita a los usuarios crear y financiar proyectos utilizando criptomonedas seleccionadas. Esta plataforma sirve como un puente entre los emprendedores que buscan financiamiento y los inversores que están dispuestos a apoyar sus ideas innovadoras.

Un elemento central de la plataforma es su capacidad para permitir la creación de proyectos por parte de cualquier usuario. Los proyectos contendrán una variedad de detalles, como nombre, descripción, objetivo de financiamiento, fecha límite para la financiación y el mínimo a recaudar. Los propietarios de proyectos también tendrán la oportunidad de inyectar fondos en la plataforma como recompensas para los inversores.

En cuanto a las transacciones, los usuarios podrán contribuir a los proyectos utilizando las criptomonedas aceptadas, que serán definidas por el administrador de la plataforma. Las contribuciones deben llegar a un mínimo requerido para ser aceptadas. Esta comisión se puede configurar por el administrador de la plataforma. El administrador de un proyecto tiene la capacidad de retirar el monto recaudado al llegar la deadline y al objetivo monetario del mismo.

Los inversores, por otro lado, tendrán la opción de retirar sus fondos en dos situaciones: cuando el proyecto genera recompensas o cuando el proyecto no alcanza el mínimo requerido en su fecha límite. En este último caso, los inversores pueden reclamar los activos que iban a contribuir.

La plataforma también proporcionará un listado completo de los proyectos para facilitar la búsqueda y elección de proyectos a financiar por parte de los inversores.

Para facilitar la interacción, fue desarrollada una aplicación descentralizada (dApp) que proporcionará una interfaz fácil de usar para los usuarios a modo de que estos puedan interactuar con la plataforma de crowdfunding.

2. Soluciones Evaluadas

2.1. Cantidad de contratos

Dos Contratos (CrowdfundingContract.sol y CrowdfundingProject.sol)

En este enfoque, se utiliza un contrato principal, CrowdfundingContract.sol, que sirve como fábrica para la creación de contratos CrowdfundingProject.sol individuales. Cada vez que un usuario desea iniciar un nuevo proyecto de crowdfunding, se crea un nuevo contrato CrowdfundingProject.sol y se registra en el contrato principal.

El contrato CrowdfundingContract.sol maneja las operaciones a nivel de plataforma, como la creación de nuevos proyectos y la gestión de un registro de todos los proyectos existentes.

El contrato CrowdfundingProject.sol, por otro lado, maneja las operaciones a nivel de proyecto. Esto incluye lógicas como realizar y rastrear contribuciones, verificar si un proyecto ha alcanzado su objetivo, y distribuir recompensas a los contribuyentes.

Un Contrato Único (CrowdfundingContract.sol)

En este enfoque, se decide consolidar toda la lógica de crowdfunding dentro de un solo contrato.

CrowdfundingContract.sol ahora es responsable no solo de las operaciones a nivel de plataforma, sino también de las operaciones a nivel de proyecto. En lugar de crear un nuevo contrato para cada proyecto, ahora se crea una nueva estructura de datos Project dentro del mismo contrato.

Esto significa que todas las funciones relacionadas con los proyectos de crowdfunding, como hacer contribuciones, comprobar si se ha alcanzado un objetivo y distribuir recompensas, se realizan ahora directamente dentro del contrato CrowdfundingContract.sol.

Además, este contrato también sigue manteniendo un registro de todos los proyectos existentes, al igual que en la primera solución, pero sin la necesidad de desplegar contratos adicionales.

2.2. Optimización de la Distribución de Recompensas

En el escenario de nuestro crowdfunding, una de las principales preocupaciones es la eficiente distribución de recompensas a los inversores o contribuyentes. Este problema se vuelve más complejo y costoso en términos de gas cuando el número de contribuyentes es grande.

Este análisis se centra en dos enfoques propuestos para abordar este problema: una solución iterativa que se planteó en una primera instancia y una basada en retiros individuales que surgio ante el problema identificado que describimos más adelante. Se busca proporcionar una solución eficiente y escalable que minimice los costos de gas y garantice una distribución justa de las recompensas.

Solución iterativa
La primera solución implicaba el uso de un bucle for para iterar a través de cada uno de los contribuyentes del proyecto. Para cada contribuyente, calculamos su recompensa proporcional y la añadimos a su balance de recompensas. Aquí está el código relevante:

```
for (address contributor : p.contributors) {
    uint256 reward = (p.contributions[contributor] * _rewardAmount) /
```

```
p.totalContributions;
    p.rewards[contributor] += reward;
}
```

Solución basada en retiros individuales

La segunda solución propuesta evita la necesidad de un bucle al requerir que cada contribuyente retire sus propias recompensas. En lugar de calcular todas las recompensas en el momento de la inyección de fondos, registramos la cantidad total de recompensas y permitimos que cada contribuyente calcule y retire su parte cuando decida hacerlo. Aquí está el código relevante:

```
p.totalReward = _rewardAmount;
```

Luego, cuando un contribuyente desea retirar su recompensa:

```
uint256 reward = (p.contributions[msg.sender] * p.totalReward) /
p.totalContributions;
p.totalReward -= reward;
p.ERCToken.transfer(msg.sender, reward);
```

3. Solución Seleccionada

3.1. Cantidad de Contratos

Después de analizar este enfoque, se decidió consolidar ambos contratos en un único contrato: CrowdfundingContract.sol. Este contrato ahora maneja la creación y gestión de todos los proyectos dentro de su propia lógica.

Hubo varias razones detrás de esta decisión:

Eficiencia de Gas: En Ethereum, cada transacción y despliegue de contrato cuesta "gas", que es esencialmente una medida de la cantidad de trabajo computacional requerido. Desplegar múltiples contratos CrowdfundingProject puede ser costoso en términos de gas. Al consolidar la lógica en un solo contrato, se pueden realizar muchas operaciones en un solo paso, ahorrando gas.
Mantenimiento y Actualización: Tener la lógica dividida entre dos contratos separados puede complicar el mantenimiento y la actualización del código. Con un solo contrato, es más fácil realizar cambios y garantizar que todas las partes del código estén sincronizadas.

Interacción entre contratos: Con dos contratos separados, la comunicación entre ellos puede ser más compleja y podría presentar riesgos de seguridad si no se maneja correctamente. Un solo contrato evita este problema.

Por estas razones, se decidió utilizar un solo contrato, CrowdfundingContract.sol, para manejar la lógica del crowdfunding. Aunque este cambio requería una reescritura significativa del código, los beneficios en términos de eficiencia de gas, mantenimiento, seguridad y organización hicieron que valiera la pena el esfuerzo adicional.

3.2. Optimización de la Distribución de Recompensas

La principal preocupación con el enfoque iterativo es que puede ser muy costoso en términos de gas, especialmente para proyectos con un gran número de contribuyentes. En Ethereum, el coste del gas aumenta con la complejidad de las operaciones, y un bucle que requiere varias operaciones puede resultar en costes de transacción muy altos. Además, existe un límite de gas para las transacciones en Ethereum, lo que significa que esta función podría fallar si el número de contribuyentes es demasiado alto.

Este enfoque es mucho más eficiente en términos de gas, ya que elimina la necesidad de iterar a través de un conjunto potencialmente grande de contribuyentes. Además, es más escalable, ya que el coste del gas no depende del número de contribuyentes.

Por lo tanto, optamos por la segunda solución debido a sus ventajas en términos de eficiencia y escalabilidad. Aunque pone un poco más de responsabilidad en los contribuyentes (ya que ahora tienen que retirar sus propias recompensas), los beneficios en términos de ahorro de gas superan este inconveniente.

4. Por qué blockchain

Blockchain es particularmente útil para solucionar nuestro problema ya que nos aporta las siguientes ventajas:

Transparencia: Todas las transacciones en la blockchain son públicas y verificables, lo que proporciona una capa adicional de confianza y seguridad para los inversores de proyectos. Ellos pueden rastrear y verificar el estado de su inversión y las recompensas correspondientes.

Seguridad: La blockchain es altamente segura. Los datos una vez registrados en la cadena de bloques no pueden modificarse, lo que impide cualquier tipo de manipulación o fraude por parte de inversores, administradores de un proyecto o administradores de la plataforma.

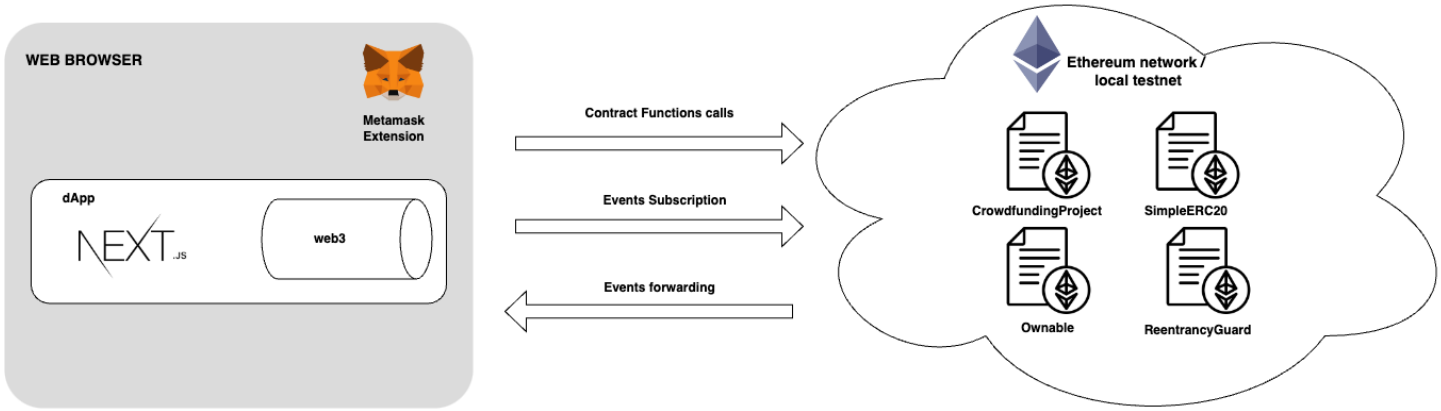
Descentralización: Blockchain elimina la necesidad de intermediarios en este problema y permite a los usuarios interactuar directamente entre ellos. En el caso de la plataforma de crowdfunding, los emprendedores pueden recibir fondos directamente de los inversores sin la necesidad de un tercero.

Automatización con Smart Contracts: Los smart contracts permiten la automatización de procesos y acciones basados en condiciones predefinidas, como la distribución de recompensas o la devolución de fondos en caso de que no se alcance el objetivo de financiamiento. Esto ahorra tiempo y reduce la posibilidad de errores.

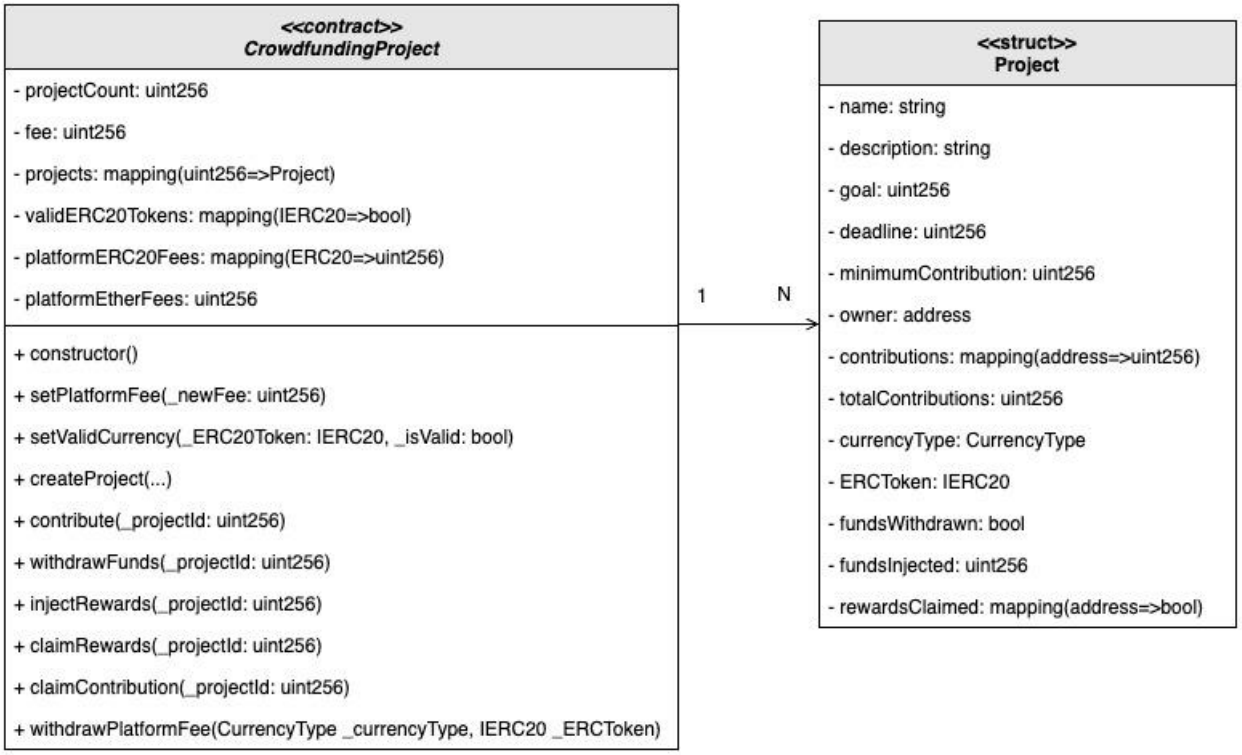
Trazabilidad: Cada transacción en la blockchain puede ser rastreada. Esto permite a los inversores seguir el flujo de sus fondos, garantizar que se utilicen de la manera prevista y verificar que se les devuelva lo que corresponde.

Acceso Global: Como red global, la blockchain permite a cualquier persona con acceso a internet participar en la plataforma de crowdfunding, ampliando el alcance de posibles inversores y emprendedores con proyectos.

5. Arquitectura



6. Diagrama de clases



7. Problemas conocidos de nuestro smart contract

algo de que cuando claimeablas una reward sos pollo si se inyectan mas rewards despues no? No gordo, en realidad el project owner puede inyectar mas de una vez rewards, pero una sola vez se puede hacer el claim del proyecto.

Hay ciertos aspectos del smart contract que sin dudas pueden ser mejorados. Algunos de estos son los siguientes:

Falta de la funcionalidad de pausa: El contrato no tiene una funcionalidad de pausa, que podría ser útil en caso de que se descubra un error o vulnerabilidad.

No hay comprobación de overflows/underflows: En las operaciones matemáticas, no hay ninguna protección contra overflows o underflows. Podrías considerar el uso de la biblioteca SafeMath de OpenZeppelin para evitar estos problemas.

No hay un método para recuperar tokens enviados por error: Si un usuario envía tokens ERC20 al contrato por error, esos tokens estarían atrapados en el contrato para siempre. Sería útil tener un método que permita al propietario del contrato recuperar tokens enviados por error.

Falta la comprobación de dirección cero: En varias funciones, no hay comprobaciones para asegurar que la dirección de los tokens ERC20 no es la dirección cero.

No hay eventos emitidos en algunos casos: Los eventos son útiles para el seguimiento de las acciones que ocurren en el contrato. Podrías considerar emitir eventos en funciones como setPlatformFee y setValidCurrency.

La contribución se realiza en un solo paso: En el caso de la contribución con ERC20, la contribución se realiza en un solo paso, lo que podría resultar en una falla si el usuario no ha aprobado previamente la transferencia.

El contrato no se detiene cuando se alcanza la meta: Cuando el proyecto alcanza su objetivo, las contribuciones pueden seguir llegando. Podría ser útil implementar alguna lógica para detener las contribuciones una vez que se alcanza el objetivo.

Retirada de fondos antes de que el proyecto esté financiado: El método withdrawFunds permite retirar fondos incluso antes de que el proyecto esté completamente financiado.

8. Gobernanza de la plataforma más descentralizada

De cara a mejorar la gobernanza de la plataforma podríamos incorporar un token de gobernanza. El token de gobernanza podría ser, por ejemplo, un token ERC20 y podría permitir a las personas que lo tienen de tokens participar en la toma de decisiones de la plataforma.

Al existir el token los propietarios de este podrían votar sobre diferentes aspectos de la plataforma, como las comisiones, los proyectos que se admiten/rechazan en la plataforma, y cualquier futuro cambio en las reglas de la plataforma. De esta manera, no hay una única entidad o propietario que controle todas las decisiones.

Si fuésemos a desarrollar una solución para esto debemos asegurarnos que el diseño que implementemos evite las “ballenas” del token. Es decir, los propietarios con grandes cantidades que tengan un poder de decisión desproporcionado en las decisiones lo que podría llegar a decisiones injustas, malintencionadas u otros tipos de manipulación de la plataforma.

Otra cosa que se nos ocurre es establecer un fondo comunitario. Es decir un fondo financiado por una pequeña parte de las comisiones cobradas por la plataforma gestionado por la comunidad que se use para financiar mejoras en la plataforma, recompensar a los miembros que contribuyen al desarrollo de la plataforma, o incluso para financiar proyectos de interés para la comunidad.

Finalmente, otra cosa que podríamos hacer para hacer la plataforma más descentralizadas es realizar la descentralización del almacenamiento de datos. Es decir, podríamos almacenar los datos de la plataforma en un sistema descentralizado de almacenamiento de archivos como IPFS (InterPlanetary File System). Esto proporcionaría una mayor resistencia a la censura y garantiza que la plataforma no dependa de un solo servidor o proveedor de

servicios. Sería de nuestro interés, por ejemplo, almacenar en IPFS la información detallada de los proyectos: descripción, detalles, objetivos, deadline, etc.

9. Uso de reputación para tener mayores beneficios

Para plantear cómo podríamos solucionar esto utilizaremos un enfoque de Self Sovereign Identity (SSI) por lo tanto en primer lugar definiremos qué es SSI.
SSI es un enfoque para la gestión de identidades digitales en el que la identidad del usuario es independiente y está completamente bajo su control. A diferencia de los métodos de autenticación y autorización centralizados, como cuentas basadas en servicios de terceros (por ejemplo, "Iniciar sesión con Google" o "Iniciar sesión con Facebook"), SSI permite a los usuarios poseer y controlar sus credenciales personales sin la necesidad de un intermediario.
Esto le aporta al usuario ciertas características claves. Algunas de estas son:

- **Control**: Los usuarios tienen el control total de sus propias identidades, incluyendo la gestión y almacenamiento de datos de identidad.
- **Interoperabilidad**: La identidad digital puede ser utilizada a través de diferentes dominios y servicios, permitiendo la portabilidad de la identidad.
- **Acceso**: Los usuarios pueden acceder y utilizar su identidad digital en cualquier momento.
- **Consentimiento**: Los datos de identidad sólo pueden ser compartidos con el consentimiento del usuario.
- **Protección de la privacidad**: SSI reduce el riesgo de violaciones de datos ya que la información de identidad se encuentra descentralizada.

De cara a mejorar nuestra plataforma de crowdfunding podriamos utilizar SSI para implementar las siguientes features:

- **Verificación de la identidad**

Uno de los aspectos clave de la SSI es que permite la verificación de la identidad. Esto podría utilizarse en la plataforma para verificar la identidad de los dueños de los proyectos y/o inversores. Esto podría ayudar a aumentar la confianza en la plataforma y reducir el fraude. En este aspecto se basan los siguientes.

- **Calificaciones de proyecto**

Se podría permitir que los inversores califiquen los proyectos en los que han invertido. Estas calificaciones podrían ir más allá de simplemente si les gustó el proyecto o no y podrían incluir aspectos como la comunicación del dueño del proyecto o la entrega a tiempo de las recompensas prometidas. De esta forma, los nuevos inversores verían las calificaciones de un proyecto antes de decidir invertir en él.

- **Calificaciones de inversor**

Similar al caso anterior, se podría permitir que los dueños de los proyectos califiquen a los inversores. Esto podría basarse en su interacción con el dueño del proyecto y/o la puntualidad de su inversión. Esto permitiría a los dueños de los proyectos tener una idea de con quién están tratando.

- **Beneficios basados en la reputación**

Finalmente, y como punto más destacado a nuestro parecer, se podría implementar un sistema donde los inversores con alta reputación obtengan ciertos beneficios. Por ejemplo, podrían tener acceso a proyectos antes de que sean públicos o recibir una tarifa reducida en las transacciones. De igual manera, los proyectos con alta reputación podrían ser destacados en la plataforma o recibir tarifas reducidas.

Este conjunto de features haría sin duda que la plataforma sea más atractiva tanto para los inversores como para los dueños de los proyectos