COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

PLATAFORMA DE INVERSIÓN EN INFRAESTRUCTURA RENOVABLE

Autor: Santiago Gericke Parga

Director: Atilano Fernández-Pacheco ; José Luis Gahete Díaz

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI ICADE CIHS

Santiago Gericke Parga

Marketplace for Renewable Energies

Madrid

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título Marketplace of Sustainable Projects en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el
curso académico 2019/2020 es de mi autoría, original e inédito y
no ha sido presentado con anterioridad a otros efectos. El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada
de otros documentos está debidamente referenciada.

Fdo.: Santiago Gericke Parga Fecha: 01/07/2020

Santiago Gericke

Autorizada la entrega del proyecto
Atilano Fernández-Pacheco & José Luis Gahete Diaz

Fdo.: Atilano Fernández-Pacheco José Luis Gahete Diaz Fecha: 01/ 07/ 2020

AUTORIZACIÓN PARA LA DIGITALIZACIÓN, DEPÓSITO Y DIVULGACIÓN EN RED DE PROYECTOS FIN DE GRADO, FIN DE MÁSTER, TESINAS O MEMORIAS DE BACHILLERATO

1º. Declaración de la autoría y acreditación de la misma.

El autor D Santiago Gericke Parga

DECLARA ser el titular de los derechos de propiedad intelectual de la obra: Marketplace of Sustainable Projects, que ésta es una obra original, y que ostenta la condición de autor en el sentido que otorga la Ley de Propiedad Intelectual.

2º. Objeto y fines de la cesión.

Con el fin de dar la máxima difusión a la obra citada a través del Repositorio institucional de la Universidad, el autor **CEDE** a la Universidad Pontificia Comillas, de forma gratuita y no exclusiva, por el máximo plazo legal y con ámbito universal, los derechos de digitalización, de archivo, de reproducción, de distribución y de comunicación pública, incluido el derecho de puesta a disposición electrónica, tal y como se describen en la Ley de Propiedad Intelectual. El derecho de transformación se cede a los únicos efectos de lo dispuesto en la letra a) del apartado siguiente.

3º. Condiciones de la cesión y acceso

Sin perjuicio de la titularidad de la obra, que sigue correspondiendo a su autor, la cesión de derechos contemplada en esta licencia habilita para:

a) Transformarla con el fin de adaptarla a cualquier tecnología que permita incorporarla a internet y hacerla accesible; incorporar metadatos para realizar el registro de la obra e incorporar "marcas de agua" o cualquier otro sistema de seguridad o de protección.
b) Reproducirla en un soporte digital para su incorporación a una base de datos electrónica, incluyendo el derecho de reproducir y almacenar la obra en servidores, a los efectos de garantizar su seguridad, conservación y preservar el formato.
c) Comunicarla, por defecto, a través de un archivo institucional abierto, accesible de modo libre y gratuito a través de internet.
d) Cualquier otra forma de acceso (restringido, embargado, cerrado) deberá solicitarse expresamente y obedecer a causas justificadas.
e) Asignar por defecto a estos trabajos una licencia Creative Commons.
f) Asignar por defecto a estos trabajos un HANDLE (URL *persistente)*.

4º. Derechos del autor.

El autor, en tanto que titular de una obra tiene derecho a:

a) Que la Universidad identifique claramente su nombre como autor de la misma
b) Comunicar y dar publicidad a la obra en la versión que ceda y en otras posteriores a través de cualquier medio.
c) Solicitar la retirada de la obra del repositorio por causa justificada.
d) Recibir notificación fehaciente de cualquier reclamación que puedan formular terceras personas en relación con la obra y, en particular, de reclamaciones relativas a los derechos

de propiedad intelectual sobre ella.

5º. Deberes del autor.

El autor se compromete a:

a) Garantizar que el compromiso que adquiere mediante el presente escrito no infringe ningún derecho de terceros, ya sean de propiedad industrial, intelectual o cualquier otro.
b) Garantizar que el contenido de las obras no atenta contra los derechos al honor, a la intimidad y a la imagen de terceros.
c) Asumir toda reclamación o responsabilidad, incluyendo las indemnizaciones por daños, que pudieran ejercitarse contra la Universidad por terceros que vieran infringidos sus derechos e intereses a causa de la cesión.
d) Asumir la responsabilidad en el caso de que las instituciones fueran condenadas por infracción de derechos derivada de las obras objeto de la cesión.

6º. Fines y funcionamiento del Repositorio Institucional.

La obra se pondrá a disposición de los usuarios para que hagan de ella un uso justo y respetuoso con los derechos del autor, según lo permitido por la legislación aplicable, y con fines de estudio, investigación, o cualquier otro fin lícito. Con dicha finalidad, la Universidad asume los siguientes deberes y se reserva las siguientes facultades:

- La Universidad informará a los usuarios del archivo sobre los usos permitidos, y no garantiza ni asume responsabilidad alguna por otras formas en que los usuarios hagan un uso posterior de las obras no conforme con la legislación vigente. El uso posterior, más allá de la copia privada, requerirá que se cite la fuente y se reconozca la autoría, que no se obtenga beneficio comercial, y que no se realicen obras derivadas.
- La Universidad no revisará el contenido de las obras, que en todo caso permanecerá bajo la responsabilidad exclusive del autor y no estará obligada a ejercitar acciones legales en nombre del autor en el supuesto de infracciones a derechos de propiedad intelectual derivados del depósito y archivo de las obras. El autor renuncia a cualquier reclamación frente a la Universidad por las formas no ajustadas a la legislación vigente en que los usuarios hagan uso de las obras.
- La Universidad adoptará las medidas necesarias para la preservación de la obra en un futuro.
- La Universidad se reserva la facultad de retirar la obra, previa notificación al autor, en supuestos suficientemente justificados, o en caso de reclamaciones de terceros.

Madrid, a 01 de Julio de 2020

ACEPTA

Santiago Gericke

Fdo: Santiago Gericke Parga

Motivos para solicitar el acceso restringido, cerrado o embargado del trabajo en el Repositorio Institucional:

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI

GRADO EN INGENIERÍA EN TECNOLOGÍAS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

PLATAFORMA DE INVERSIÓN EN INFRAESTRUCTURA RENOVABLE

Autor: Santiago Gericke Parga

Director: Atilano Fernández-Pacheco ; José Luis Gahete Díaz

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI ICADE CIHS

Santiago Gericke Parga

Marketplace for Renewable Energies

Agradecimientos:

La colaboración de Atilano Fernández-Pacheco y José Luis Gahete ha sido muy importante de cara a desarrollar este proyecto.

MARKETPLACE OF SUSTAINABLE PROJECTS

Autor: Gericke Parga, Santiago.
Director: Fernández-Pacheco, Atlilano; Gahete Díaz, José Luis.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas

RESUMEN DEL PROYECTO

MORE se define como un crowdfunding basado en Blockchain para financiar la instalación de infraestructura renovable en España. La transparencia, trazabilidad y descentralización de la tecnología Blockchain permite aportar seguridad a todo el proceso de pago.

Palabras clave: Energía, Renovables, Blockchain, Inversión, Ethereum, Crowdfunding, Paneles solares

Introducción

España es uno de los países con más horas de sol al año de Europa y con la aprobación de la última regulación sobre la inversión en infraestructura renovable, la financiación de paneles solares se ha convertido en una opción muy interesante tanto para inversores nacionales como internacionales. Sin embargo, la inversión en infraestructura renovable tiene dos principales problemas:

1. Por un lado, requiere grandes inversiones iniciales. Por ello a día de hoy la financiación de infraestructura renovable queda restringida únicamente a fondos de inversión y a empresas punteras del sector eléctrico como Iberdrola o Endesa.
2. Por otro lado, el sistema convencional de crowdfunding no es seguro ya que el dinero invertido va directamente a la cuenta del creador del crowdfunding. Por ello, es muy sencillo que se produzcan fraudes en todo el proceso de pago.

Definición del proyecto

MORE se define como un crowdfunding basado en Blockchain para financiar la instalación de infraestructura renovable en España. La transparencia, trazabilidad y descentralización de la tecnología Blockchain permite aportar seguridad a todo el proceso de pago. Por tanto, la plataforma se centra en

dos tipos de usuarios, el usuario que quiere instalar un panel solar y el usuario que quiere invertir en la instalación de un panel solar. De esta forma, se resuelven los dos problemas que se han mencionado anteriormente. En primer lugar, la inversión inicial que se requiere, se divide entre todos los participantes del crowdfunding. En segundo lugar, todo el proceso de pago se realiza a través de la tecnología Blockchain lo que permite aportar transparencia y seguridad a todo el proceso de pago.

Descripción del modelo/sistema/herramienta

Por tanto, el Proyecto se centra en dos tipos de usuarios, el usuario que quiere instalar un panel solar y el usuario que quiere invertir en la instalación de un panel solar. El diagrama de flujo de la plataforma es el siguiente:

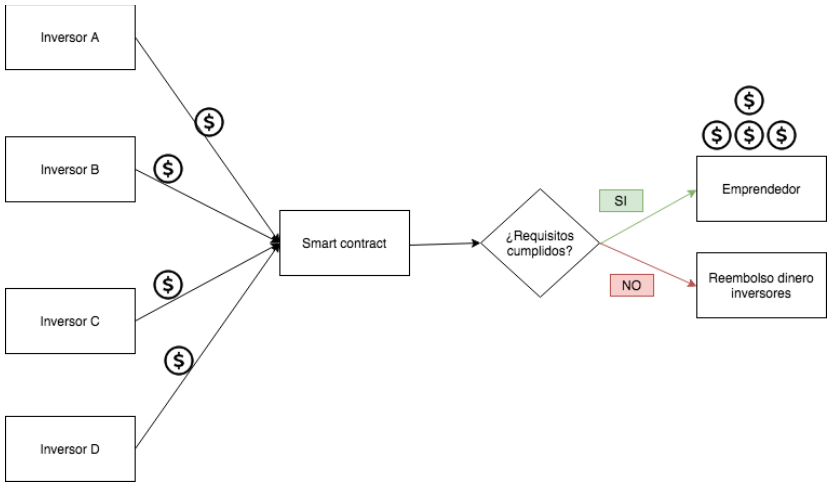


Ilustración 1 – Diagrama de flujo de la plataforma

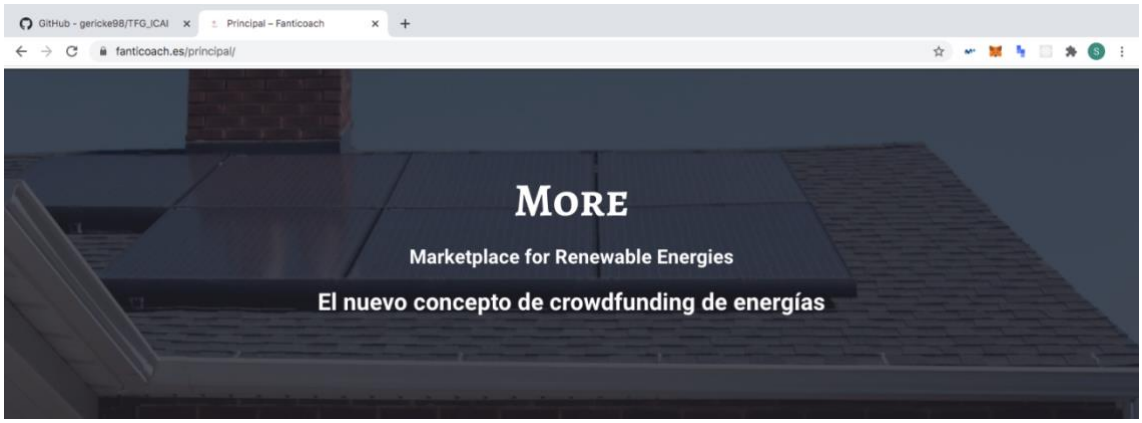
A diferencia de un crowdfunding convencional, el dinero va a un Smart Contract desplegado en la red Blockchain de Ethereum y para sacar dinero del Smart Contract, se deben cumplir una serie de requisitos prestablecidos en el propio Smart Contract.

Por otro lado, en este tipo de inversión, los inversores llevan a cabo la inversión que deriva en un ahorro energético para el consumidor final. En cada oportunidad de inversión se especifica la rentabilidad estimada de la inversión que suele estar en el rango del 6-10 %. Tras finalizar la instalación, los inversores son quienes obtienen la totalidad del ahorro económico provocado por el ahorro. Por otro lado, el consumidor, se hace dueño de la

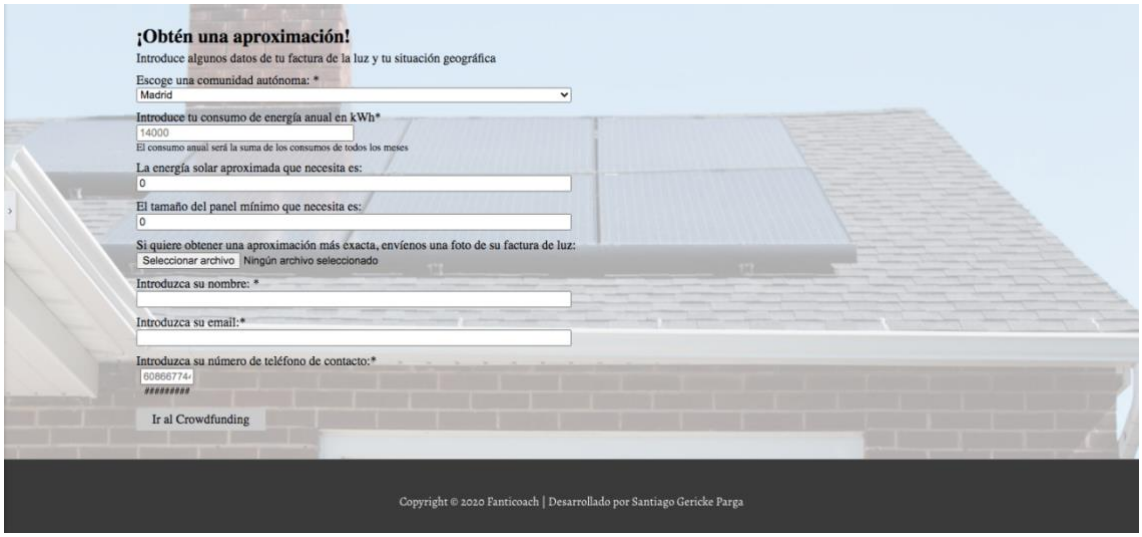
instalación para explotarla hasta el final de su vida útil (aproximadamente 25 años), cuando los inversores han obtenido la rentabilidad deseada. De esta forma el consumidor no lleva a cabo ninguna inversión y empieza a obtiene ahorros una vez los inversores hayan rentabilizado su inversión.

Resultados

Finalmente, se ha desarrollado de forma exitosa una plataforma que tiene la conexión entre el "mundo Blockchain" y el servidor web. En las siguientes ilustraciones se muestran algunos screens de la plataforma desarrollada:


GitHub - gericke98/TFG_ICAI
Principal – Fanticoach
fanticoach.es/principal/
MORE
Marketplace for Renewable Energies
El nuevo concepto de crowdfunding de energías

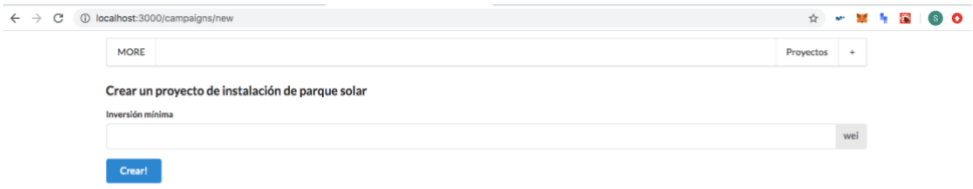

¿ Por qué nosotros ?
INVERSIÓN INICIAL
Invierta en infraestructura renovable sin necesidad de desembolsar grandes cantidades de dinero
RENOVABLES
Únase a la nueva revolución energética y contribuya a la transición energética hacia energías limpias. Además aporte su granito de arena reduciendo la utilización de energías de origen fósil
SEGURIDAD
Invierte en infraestructura renovable con total seguridad y sin riesgo de fraude


¡Obtén una aproximación!
Introduce algunos datos de tu factura de la luz y tu situación geográfica
Escoge una comunidad autónoma: *
Madrid
Introduce tu consumo de energía anual en kWh*
14000
El consumo anual será la suma de los consumos de todos los meses
La energía solar aproximada que necesita es:
0
El tamaño del panel mínimo que necesita es:
0
Si quiere obtener una aproximación más exacta, envíenos una foto de su factura de luz:
Seleccionar archivo
Ningún archivo seleccionado
Introduzca su nombre: *
Introduzca su email:*
Introduzca su número de teléfono de contacto:*
Ir al Crowdfunding
Copyright © 2020 Fanticoach | Desarrollado por Santiago Gericke Parga

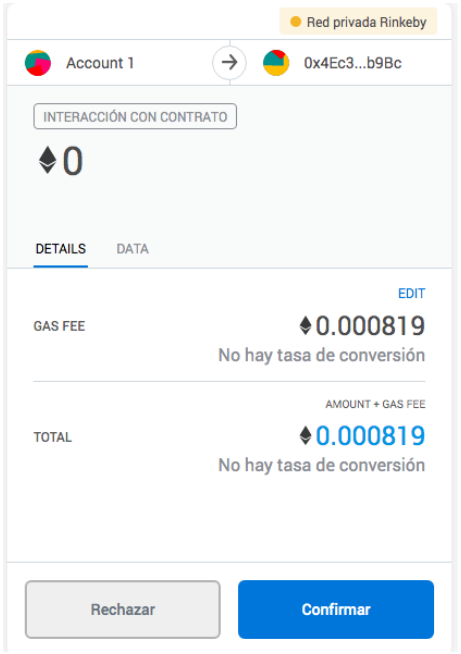
Si el usuario pincha en “Ir al Crowdfunding” se le auto dirige a la página principal del crowdfunding que es la siguiente:

Como se puede observar en la imagen, al entrar en la página principal se muestra una lista de todos los proyectos abiertos. En cada proyecto de la lista se muestra la address del contrato del proyecto y un botón para obtener más información del proyecto. Por otro lado, el usuario también tiene la opción de crear un nuevo proyecto de inversión.

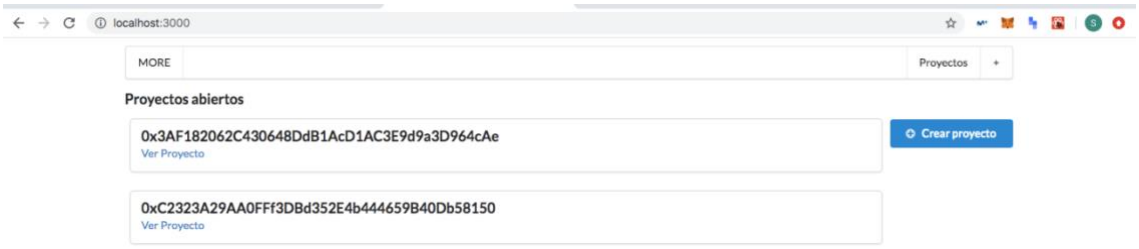
En caso de crear un nuevo proyecto, se le redirige a la siguiente página:



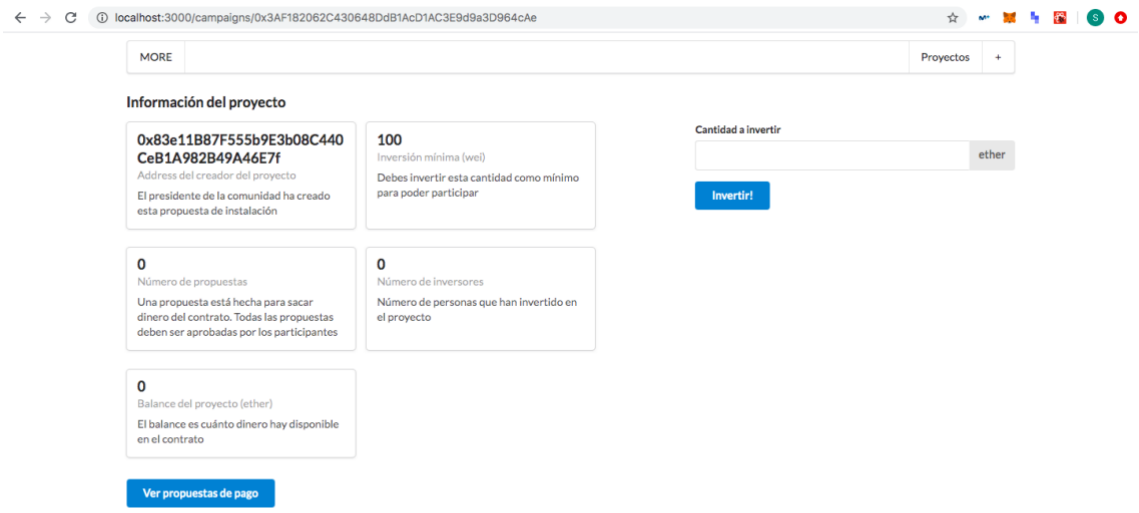
Como se puede observar, el único input que debe introducir el usuario es la inversión mínima que se debe realizar. Una vez introducido el input se crea una transacción como se muestra en la siguiente imagen:



Una vez creada, se añade el nuevo proyecto a la lista de todos los proyectos, como se muestra en la siguiente imagen:

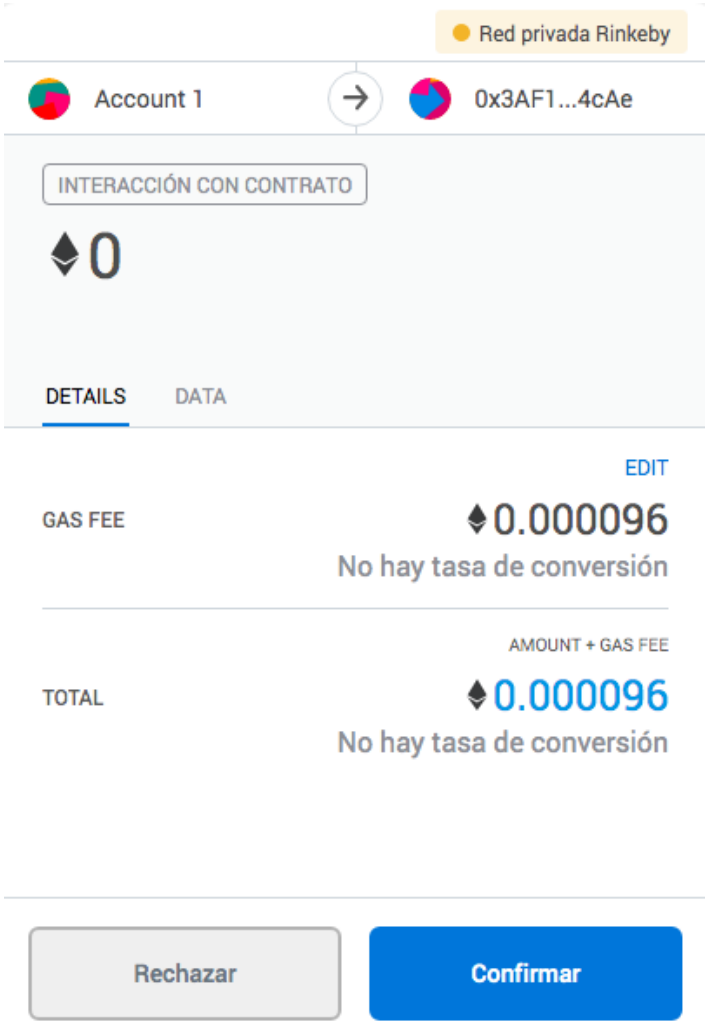


Por otro lado, en caso de que el usuario quiera obtener más información sobre un proyecto de inversión específico se le dirige a la siguiente página:

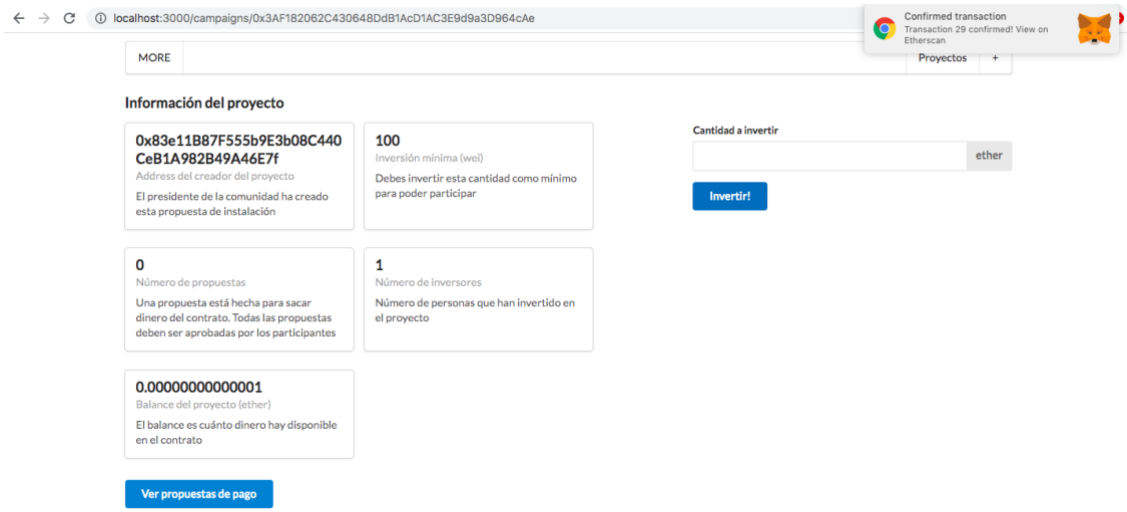


Como se puede observar en la imagen, la información que el usuario puede obtener es la address del manager del contrato, la inversión mínima, el número de propuestas de pago, el número de inversores y el balance del contrato. Además, el usuario tiene la posibilidad de invertir en el proyecto y de ver las propuestas de pago realizadas en dicho proyecto.

En caso de invertir, en primer lugar, el usuario debe confirmar la transacción como se muestra en la siguiente imagen:



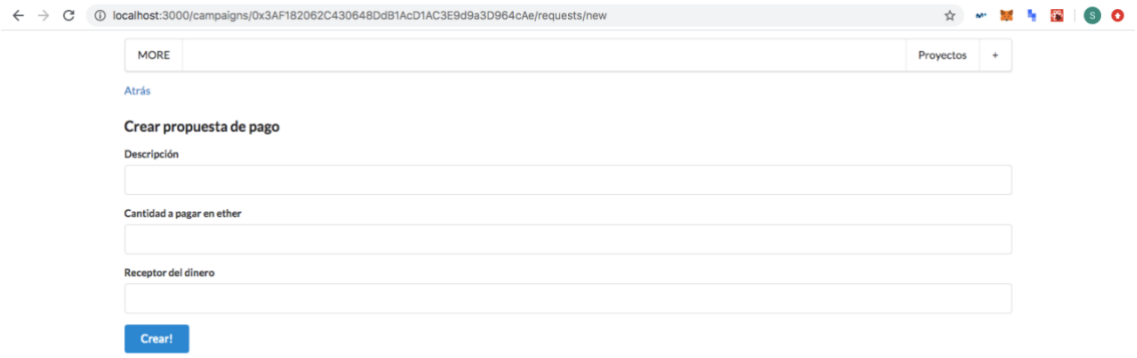
Una vez confirmada, se actualiza la información del proyecto como se muestra en la siguiente figura:



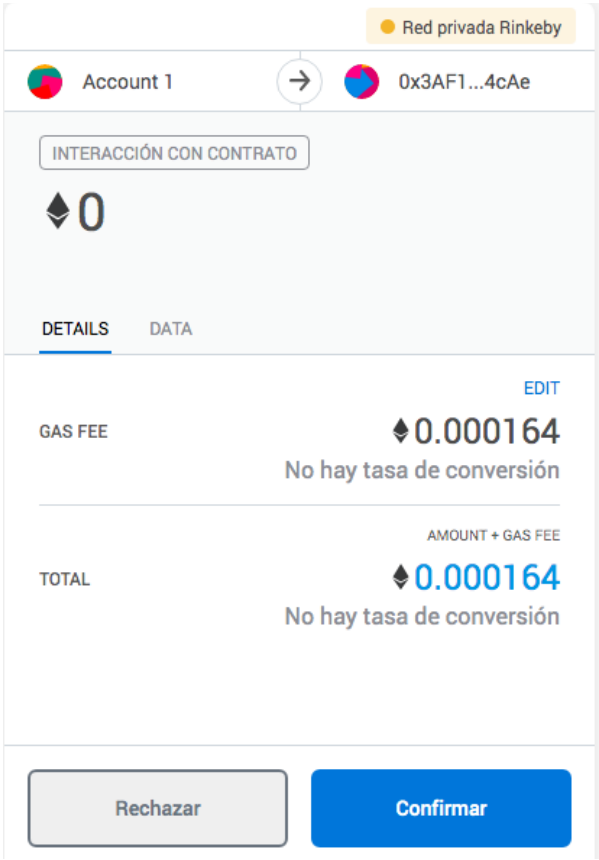
Como se puede observar, se ha actualizado el número de inversores y el balance del contrato.

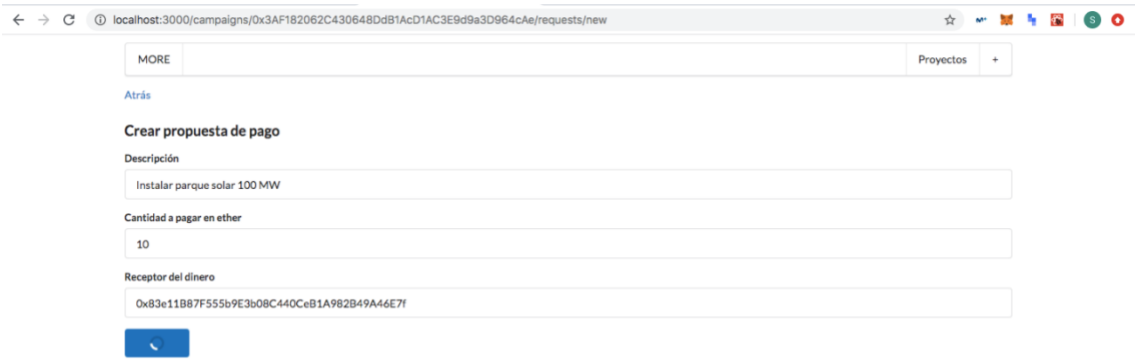
Por otro lado, el usuario tiene la posibilidad de ver las propuestas de pago realizadas en dicho proyecto. En caso de pinchar en ver propuestas de pago, se dirige al usuario a la siguiente página:

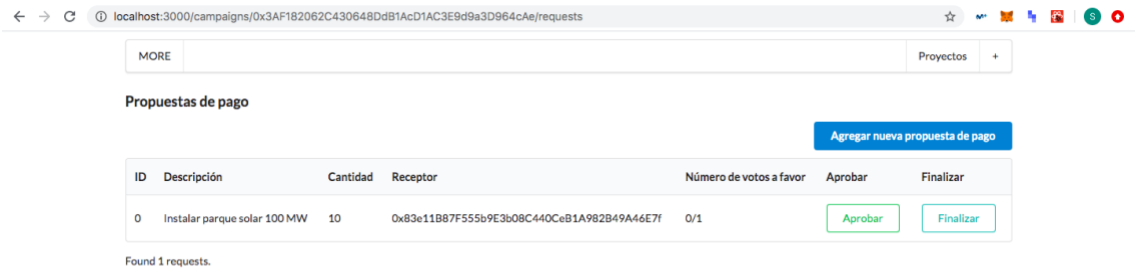
Como se puede observar, el usuario tiene la posibilidad de ver todas las propuestas de pago realizadas en una tabla. Además, también tiene la posibilidad de crear una nueva propuesta de pago. En caso de agregar una nueva propuesta de pago, el usuario debe rellenar el siguiente formulario:



En dicho formulario, el usuario debe introducir la descripción del gasto, la cantidad a pagar en ether y la address del receptor del dinero. Una vez introducidos dichos datos, se crea una nueva transacción y se añade la propuesta de pago a la lista como se muestra en las siguientes figuras:







Como se puede observar en la última imagen, la proposición se ha añadido a la lista. En ella, se incluye una descripción, la cantidad a invertir, la address del receptor del dinero, el número de votos a favor y la posibilidad de aprobar la medida o finalizarla.

Por tanto, se ha desarrollado una plataforma en la que:

1. Cualquier persona puede crear un crowdfunding para instalar paneles solares y convertirse en prosumidor sin necesidad de desembolsar una inversión inicial alta.

2. Cualquier persona puede invertir en infraestructura renovable ya que la inversión mínima es baja.

3. Cualquier persona puede invertir sin preocuparse por posibles fraudes o hackeos. Tras invertir, los inversores siguen teniendo control sobre la inversión total.

MARKETPLACE OF SUSTAINABLE PROJECTS

Author: Gericke Parga, Santiago.
Supervisor: Fernández-Pacheco, Atlilano; Gahete Díaz, José Luis.
Colaborating Entity: ICAI – Universidad Pontificia Comillas

ABSTRACT

MORE is defined as a Blockchain-based crowdfunding to finance the installation of renewable infrastructure in Spain. The transparency, traceability and decentralization of Blockchain technology allows to provide security to the entire payment process.

Keywords: Energy, Renewables, Blockchain, Investment, Ethereum, Crowdfunding, Solar panels

Introduction

Spain is one of the countries with the most hours of sunshine per year in Europe and with the approval of the latest regulation on investment in renewable infrastructure, the financing of solar panels has become a very interesting option for both national and international investors. However, investing in renewable infrastructure has two main problems:

1. On the one hand, it requires large initial investments. Therefore, today the financing of renewable infrastructure is restricted only to investment funds and leading companies in the electricity sector such as Iberdrola or Endesa.

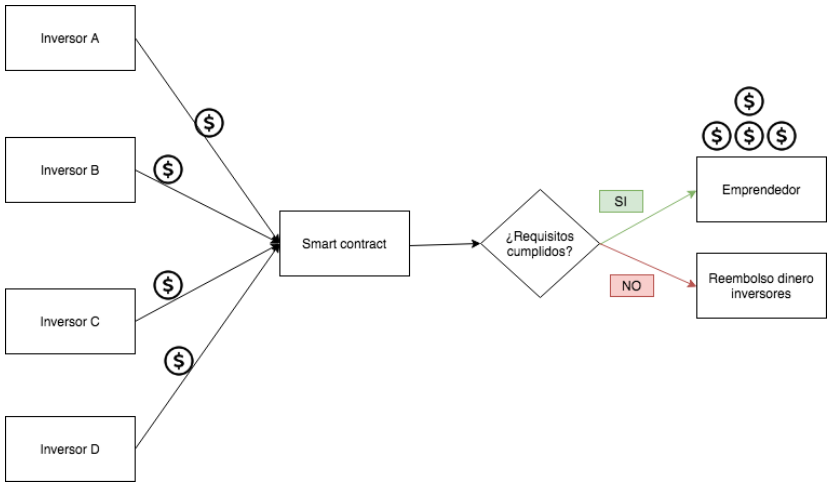
2. On the other hand, the conventional crowdfunding system is not safe since the invested money goes directly to the account of the creator of the crowdfunding. Therefore, it is very easy for fraud to occur throughout the payment process.

Project Definition

MORE is defined as a Blockchain-based crowdfunding to finance the installation of renewable infrastructure in Spain. The transparency, traceability and decentralization of Blockchain technology allows to provide security to the entire payment process. Therefore, the platform focuses on two types of users, the user who wants to install a solar panel and the user who wants to invest in the installation of a solar panel. In this way, the two problems mentioned above are solved. First of all, the initial investment required is divided among all the crowdfunding participants. Secondly, the entire payment process is done through Blockchain technology, which allows transparency and security to be provided to the entire payment process.

Platform description

Therefore, the Project focuses on two types of users, the user who wants to install a solar panel and the user who wants to invest in the installation of a solar panel. The platform flow diagram is as follows:



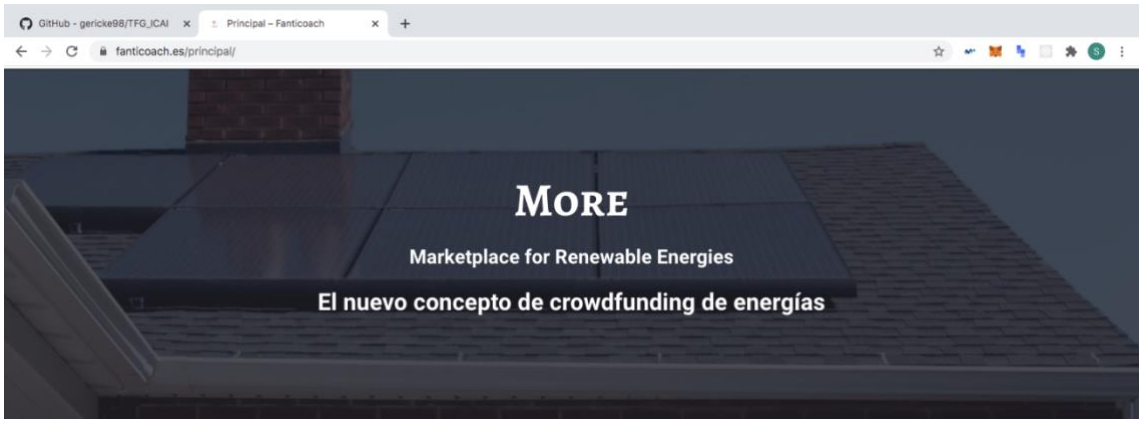
Ilustration 1 – Platform flowchart

Unlike conventional crowdfunding, money goes to a Smart Contract deployed on the Ethereum Blockchain network and to withdraw money from the Smart Contract, a series of pre-established requirements must be met in the Smart Contract itself.

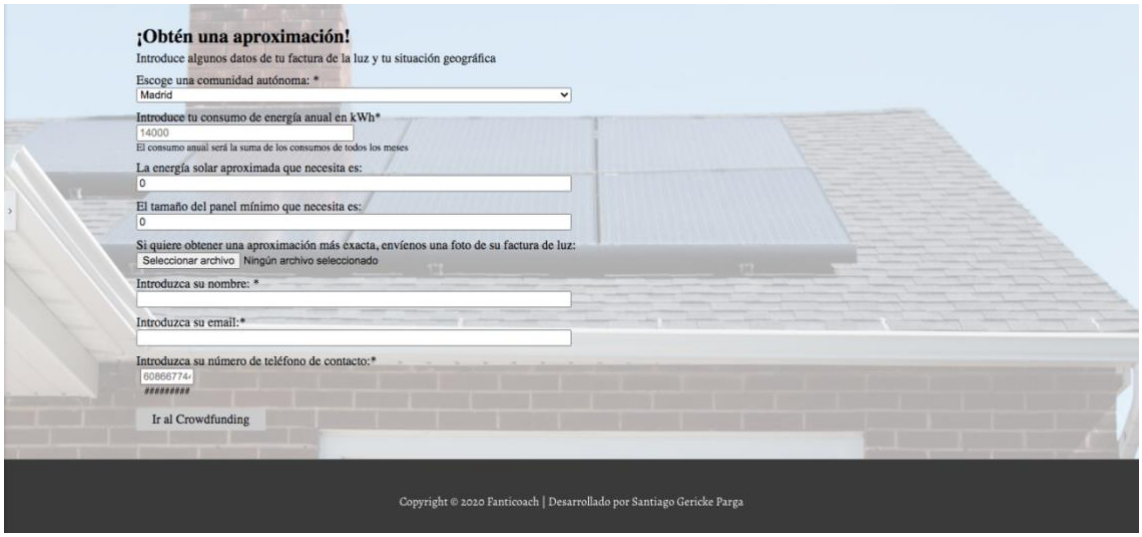
On the other hand, in this type of investment, investors carry out the investment that results in energy savings for the final consumer. In each investment opportunity, the estimated return on investment is specified, which is usually in the range of 6-10%. After completing the installation, the investors are the ones who obtain all the economic savings caused by the savings. On the other hand, the consumer owns the installation to operate it until the end of its useful life (approximately 25 years), when the investors have obtained the desired profitability. In this way the consumer does not carry out any investment and begins to obtain savings once the investors have profited from their investment.

Results

Finally, a platform that has the connection between the “Blockchain world” and the web server has been successfully developed. The following illustrations show some screens of the developed platform.


GitHub - gericke98/TFG_ICAI
Principal – Fanticoach
fanticoach.es/principal/
MORE
Marketplace for Renewable Energies
El nuevo concepto de crowdfunding de energías

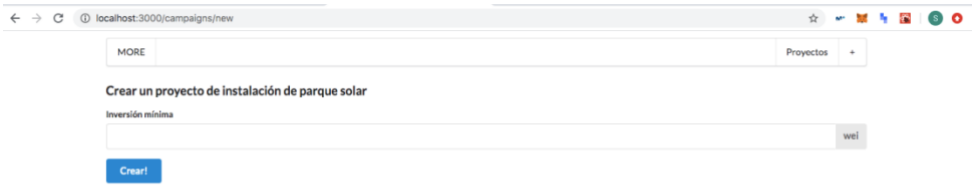

¿ Por qué nosotros ?
INVERSIÓN INICIAL
Invierta en infraestructura renovable sin necesidad de desembolsar grandes cantidades de dinero
RENOVABLES
Únase a la nueva revolución energética y contribuya a la transición energética hacia energías limpias. Además aporte su granito de arena reduciendo la utilización de energías de origen fósil
SEGURIDAD
Invierte en infraestructura renovable con total seguridad y sin riesgo de fraude


¡Obtén una aproximación!
Introduce algunos datos de tu factura de la luz y tu situación geográfica
Escoge una comunidad autónoma: *
Madrid
Introduce tu consumo de energía anual en kWh*
14000
El consumo anual será la suma de los consumos de todos los meses
La energía solar aproximada que necesita es:
0
El tamaño del panel mínimo que necesita es:
0
Si quiere obtener una aproximación más exacta, envíenos una foto de su factura de luz:
Seleccionar archivo
Ningún archivo seleccionado
Introduzca su nombre: *
Introduzca su email:*
Introduzca su número de teléfono de contacto:*
Ir al Crowdfunding
Copyright © 2020 Fanticoach | Desarrollado por Santiago Gericke Parga

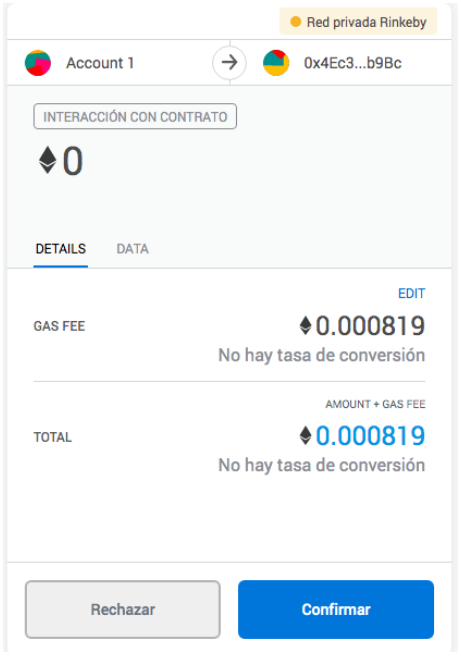
If the user clicks on “Go to Crowdfunding” they will automatically go to the crowdfunding main page, which is the following:

As you can see in the image, when entering the main page a list of all open projects is displayed. The project contract address and a button to get more information about the project are shown in each project in the list. On the other hand, the user also has the option of creating a new investment project.

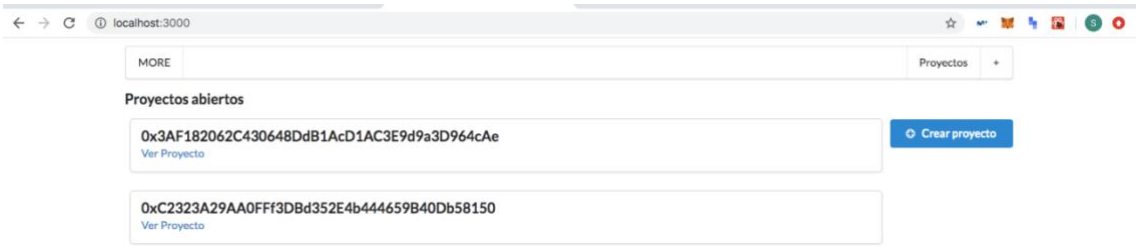
In case of creating a new project, you will be redirected to the following page:



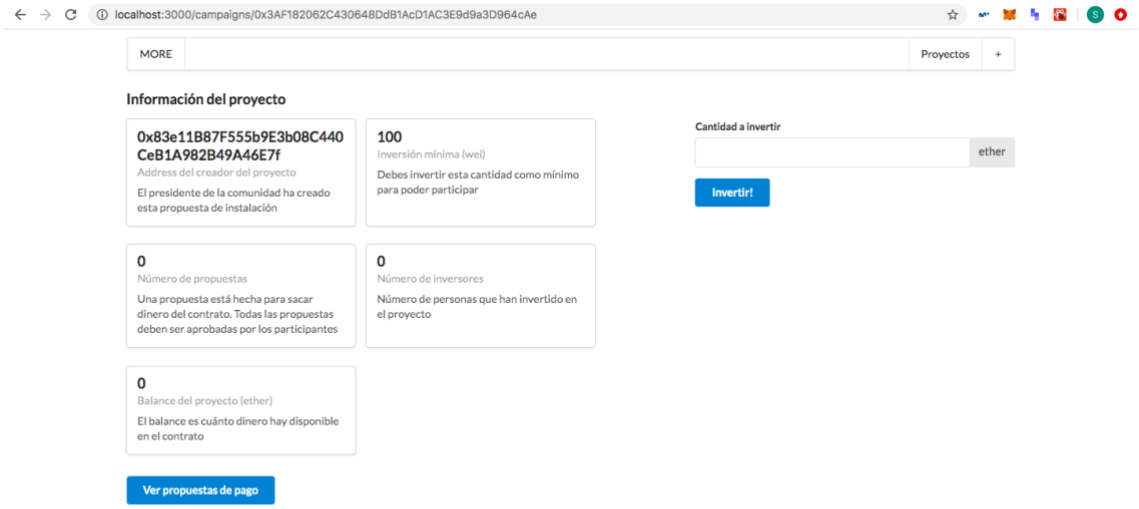
As can be seen, the only input that the user must enter is the minimum investment that must be made. Once the input is entered, a transaction is created as shown in the following image:



Once created, the new project is added to the list of all projects, as shown in the following image:

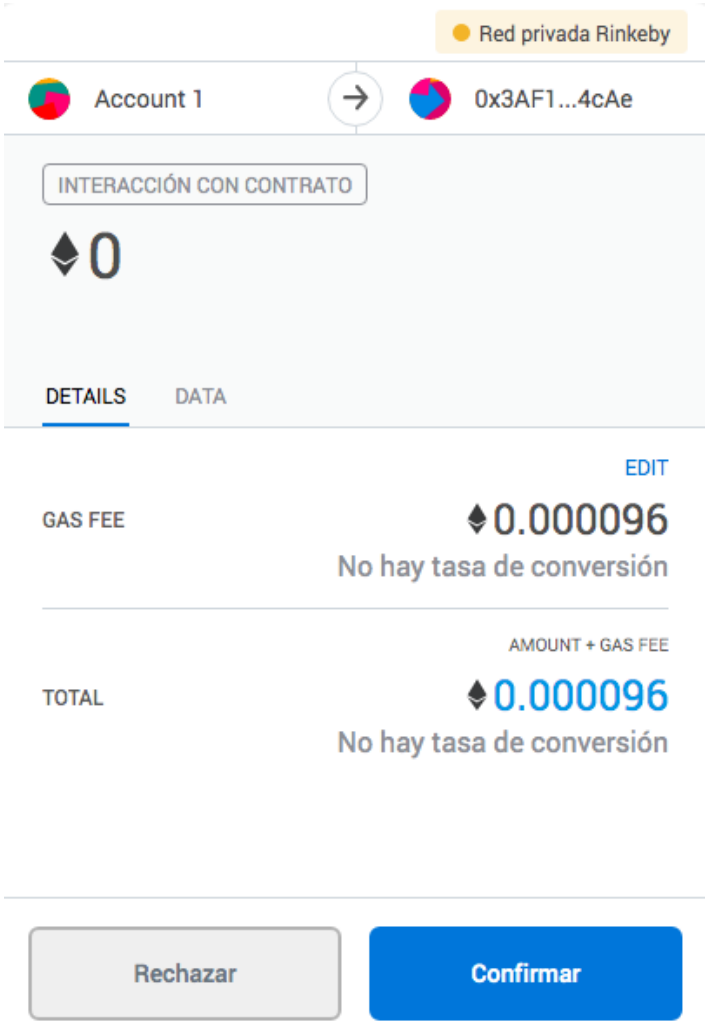


On the other hand, in case the user wants to obtain more information about a specific investment project, they are directed to the following page:

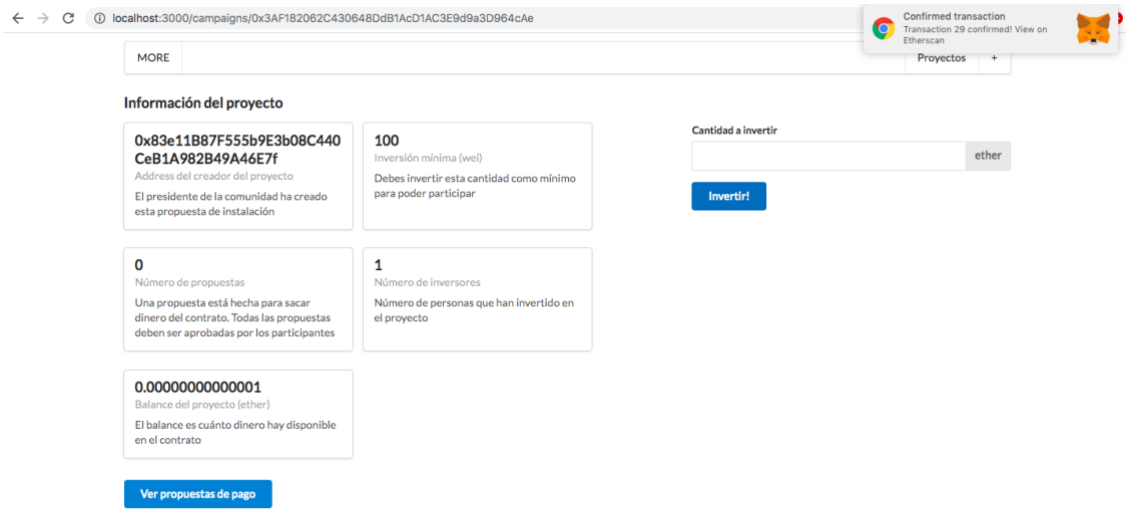


As can be seen in the image, the information that the user can obtain is the address of the contract manager, the minimum investment, the number of payment proposals, the number of investors and the balance of the contract. Furthermore, the user has the possibility to invest in the project and to see

the payment proposals made in said project. In case of investing, first of all, the user must confirm the transaction as shown in the following image:



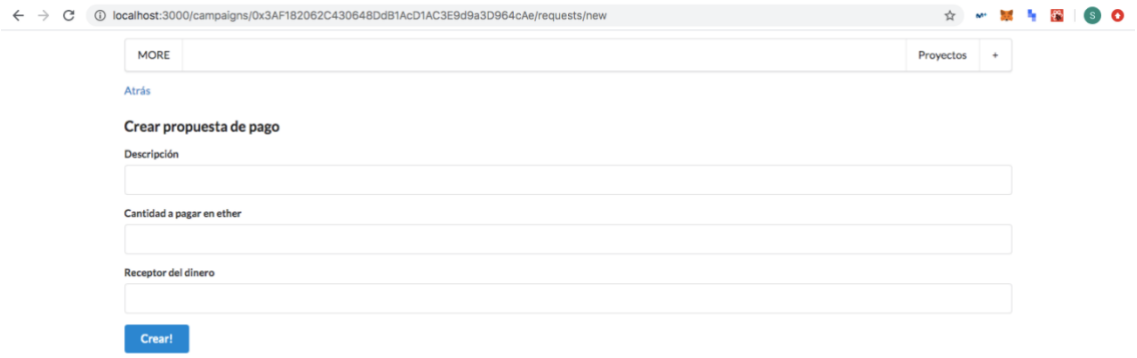
Once confirmed, the project information is updated as shown in the following figure:



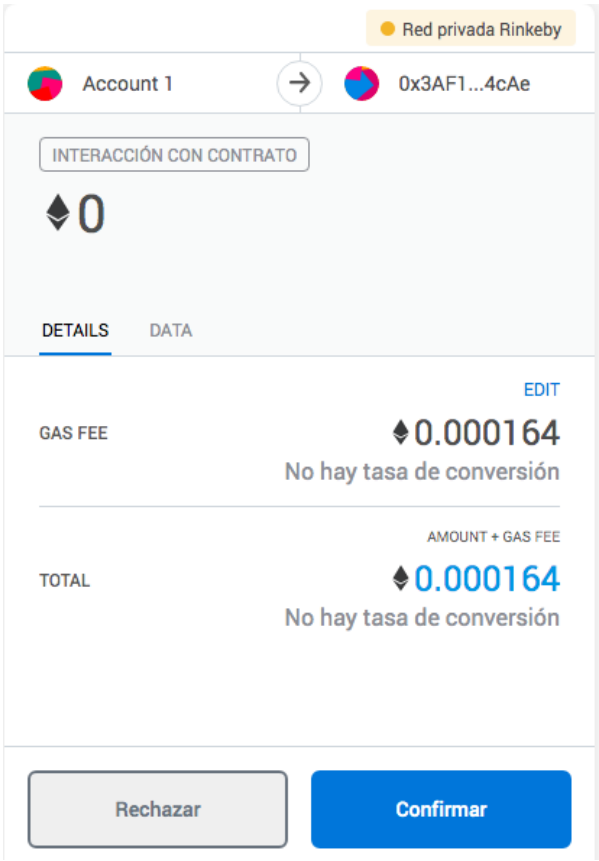
As can be seen, the number of investors and the balance of the contract have been updated.

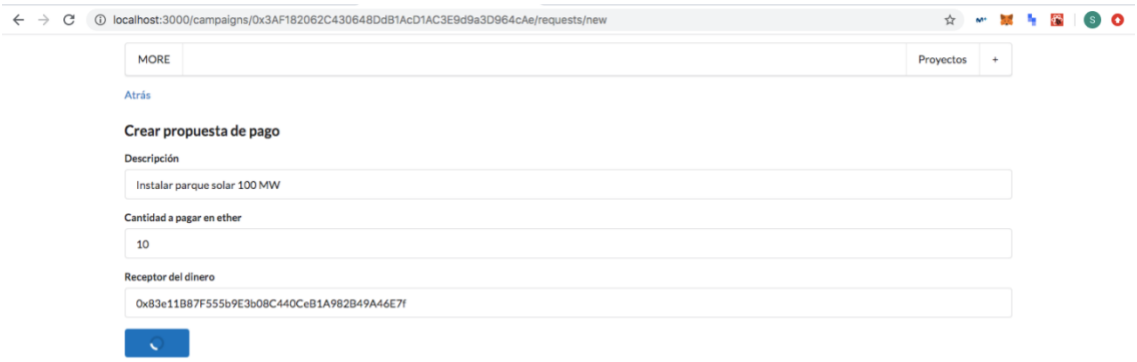
On the other hand, the user has the possibility to see the payment proposals made in said project. In case of clicking on view payment proposals, the user is directed to the following page:

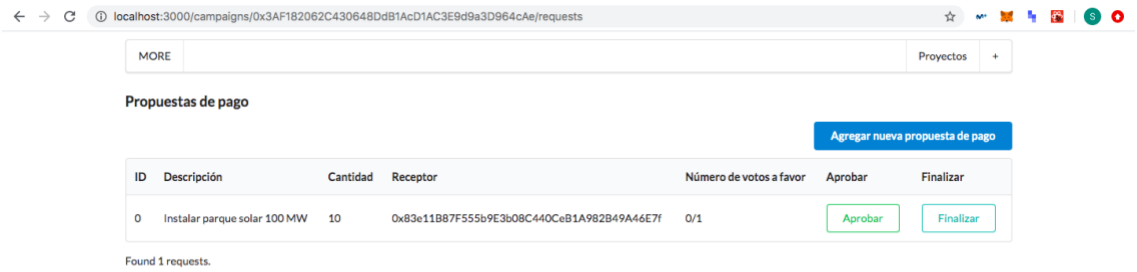
As it can be seen, the user has the possibility to see all the payment proposals made in a table. In addition, you also have the possibility to create a new payment proposal. In case of adding a new payment proposal, the user must fill out the following form:



In this form, the user must enter the description of the expense, the amount to pay in ether and the address of the recipient of the money. Once these data have been entered, a new transaction is created and the payment proposal is added to the list as shown in the following figures:







As you can see in the last image, the proposition has been added to the list. It includes a description, the amount to invest, the address of the recipient of the money, the number of votes in favor and the possibility of approving or ending the measure.

Therefore, a platform has been developed in which:

1. Anyone can create a crowdfunding to install solar panels and become a prosumer without having to pay a high initial investment.

2. Anyone can invest in renewable infrastructure since the minimum investment is low.

3. Anyone can invest without worrying about possible fraud or hacks. After investing, investors continue to have control over the total investment.

1. Introducción

¿Qué se le viene a la cabeza cuando le pregunto por el futuro? Robots, automatización, sensores, 5G… Se aproxima una etapa de grandes cambios en la que la llegada de las nuevas tecnologías, revolucionará la gran mayoría de las industrias. Muchos lo llaman la revolución industrial 4.0 y lo que es seguro es que se van a desarrollar grandes cambios en los próximos años. Todas las industrias tendrán que evolucionar y adaptarse a esta revolución. Entre ellas, la industria energética constituye un núcleo fundamental, ya que es el motor que hace funcionar muchas otras industrias. La revolución energética, se encamina hacia un paradigma de generación distribuida, donde las fuentes renovables tendrán mayor protagonismo, y el autoconsumo y los denominados prosumidores transformarán el sector radicalmente.

Para entender esta transición es importante tener claros tres conceptos: autoconsumo, prosumidor y generación distribuida.

El **autoconsumo** se define como la generación individual de electricidad para su propio consumo, generalmente a partir de fuentes renovables, y en particular, solar fotovoltaica.

El **prosumidor** se refiere al usuario que practica el autoconsumo y además tiene la posibilidad de vender el excedente de energía que produce con otros usuarios a través de la red general.

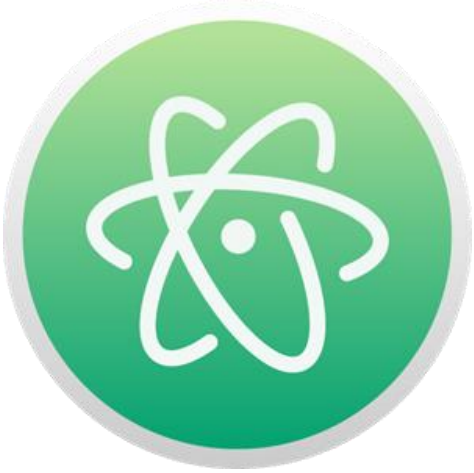
Por último, la **generación distribuida** se define como un sistema alternativo a la generación centralizada hoy vigente, que propone sumar a la red energética las diversas instalaciones de los prosumidores. De esta manera, se descentraliza la generación de energía y se mejora tanto la eficiencia energética como el coste de generar energía.

Históricamente, las energías renovables se inician en España a principios de los años 80. En 1985 el gobierno apuesta fuertemente por las renovables y aprueba el *Real Decreto 916/85* que supone un gran apoyo a las hidráulicas (única fuente de generación renovable de la época). En 1986, España entra en la Unión Europea y los objetivos de mejora medioambiental empiezan a tomar fuerza (en 1997 se establecen una serie de objetivos para 2010, entre los que destaca que el 12% de la energía primaria consumida debe ser generada por energías renovables). En 1998, se introduce el mercado eléctrico en España. Las principales fuentes de generación de energía renovable eran la eólica y la hidráulica. Más tarde, se implantan tres marcos regulatorios para dar estabilidad a largo plazo al sistema (*RD 2818/1998, RD 436/2004, RD 661/2007*). Se establecen una serie de objetivos para el año 2020 entre los que destaca que el 20% de la energía primaria consumida debe ser suministrada por generadores renovables. Pocos años después, se desarrollaron otros métodos de generación de energía renovable como la fotovoltaica o la biomasa.

Por otro lado, la industria eléctrica ha vivido una gran evolución en los últimos años con la llegada de los contadores inteligentes a España. Según la Comisión Nacional de los Mercados y la Competencia, a finales del año 2018 el 99% de los medidores en España estaban digitalizados. Gracias a este gran avance, todos los usuarios tienen la posibilidad de conocer los datos de su consumo eléctrico de los últimos tres años por meses, días e incluso horas.

2. Descripción de las tecnologías:

a. Atom



Atom es un software editor de código fuente, es decir, una plataforma a través de la cual se pueden escribir programas en cualquier lenguaje de programación, incluido Solidity. Además, Atom permite la descarga de diferentes tipos de componentes y extensiones que ayudan a la edición y corrección de errores del código.

Atom destaca por su sencilla e intuitiva interfaz, que permite que pueda ser usado de una manera fácil por el usuario para el desarrollo del código.

Además, este editor de código fuente permite usar Git, el cual permite realizar un control de versiones, del código escrito, permitiendo que sea más fácil saber qué cambios se han realizado y guardarlos.

Por último, cabe destacar que Atom es un software gratis, de libre distribución, de código abierto y que es soportado tanto en Windows, Linux y MacOS.

b. Ganache

Ganache es un simulador de la red BlockChain de Ethereum que permite realizar pruebas de desarrollo sobre el comportamiento de una aplicación BlockChain sin saturar la red Ethereum principal y permitiendo realizar estas pruebas sin ningún tipo de coste de transacción.

Entre las ventajas que Ganache proporciona se pueden destacar las siguientes: muestra el historial de ejecución de BlockChain para poder conocer lo que está sucediendo en la red y poder detectar errores; permite configurar actividades de minado para fijar los bloques de manera más sencilla y adaptada a las necesidades del desarrollo; permite explorar los diferentes bloques que hay en la red de BlockChain; brinda las últimas características de Ethereum para el desarrollo de aplicaciones enfocadas en

esta tecnología.

En último lugar, cabe destacar que es una tecnología completamente gratuita que, además, está disponible para los principales Sistemas Operativo del mercado.

c. Truffle Framework

Truffle es un entorno de desarrollo y un framework de pruebas que permite que la programación para Ethereum sea más fácil. Para ello permite la creación del entorno de programación de Ethereum mediante un comando.

Entre las características que brinda Truffle destacan: permite la prueba del correcto funcionamiento de los Smart Contracts; permite la realización de las migraciones de los Smart Contracts a la red de BlockChain; brinda una consola que permite realizar, con facilidad, gestiones con los Smart Contracts desarrollados.

En último lugar, cabe destacar que es una tecnología completamente gratuita que, además, está disponible para los principales Sistemas Operativo del mercado.

d. Node js

Node es un entorno multiplataforma que permite el desarrollo de la capa del servidor. En este proyecto ha sido empleado para el despliegue del servidor para el desarrollo del interfaz web que conecta la red Ethereum con la plataforma.

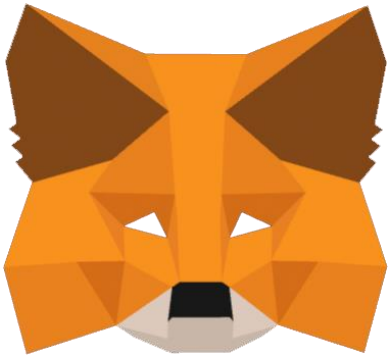
Entre las ventajas por las que ha sido escogido destacan su facilidad de uso, su alto rendimiento y por la facilidad de su aprendizaje. Así mismo es una tecnología gratuita y compatible con cualquiera de los principales Sistemas Operativos del Mercado actual.

e. NPM

Es el gestor de paquetes por defecto de Node, por lo que en este proyecto ha sido empleado junto a Node para el despliegue de los archivos necesarios para el desarrollo del interfaz web y del servidor.

Destaca por ser un software gratuito y, al igual que Node, ser compatible con los principales Sistemas Operativos del Mercado actual.

f. MetaMask



MetaMask es una extensión para el explorador de internet que funciona de conexión entre la red de Blockchain Ethereum y el servidor de una página

web, así como gestionar las cuentas que hay dentro de la red.

Por lo tanto, en este proyecto se ha empleado para realizar la conexión entre la plataforma y la red de pruebas desplegada con Ganache.

Entre sus principales características destaca que es gratuito y compatible con Google Chrome y Firefox.

Finalmente es importante destacar que es muy intuitivo de usar, así como compatible con diferentes redes de BlockChain.

g. Next.js

Next JS es un framework basado en React JS, es decir nos ayuda a crear aplicaciones Reactivas que el servidor se encarga de renderizarla. Es muy ligero y práctico por lo que el desarrollo de aplicaciones con Javascript y React JS se nos hace más rápido y con pocas complicaciones.
React es una librería Javascript focalizada en el desarrollo de interfaces de usuario.

h. Solidity

Solidity es el lenguaje de programación orientado a objetos empleado para el desarrollo de Smart Contracts dentro de la tecnología BlockChain en Ethereum. Por lo tanto, en este proyecto ha sido empleado para el desarrollo de los contratos que permiten el funcionamiento de la plataforma.
Las principales ventajas de este lenguaje es la facilidad para interactuar con funciones de contratos ya subidos a la red Blockchain, permitiendo así que un evento se ejecute cuando suceda una condición.

i. Web3js

Web3JS es un conjunto de librerías de JavaScript que permite la conexión entre el interfaz web y la red BlockChain. En este proyecto, por lo tanto, se ha empleado para conectar el interfaz de la plataforma con los contratos desarrollados en Solidity. Por tanto, Web3 es el puente que conecta el mundo BlockChain con el mundo de interfaz de usuario.

Entre las ventajas que brinda Web3JS destacan su facilidad a la hora de usarlo y de implementarlo.

j. Wordpress

WordPress es un sistema de administración de contenido (CMS) gratuito y de código abierto escrito en PHP y emparejado con una base de datos MySQL o MariaDB. Las características incluyen una arquitectura de complementos y un sistema de plantillas, denominado dentro de WordPress como Temas. WordPress se creó originalmente como un sistema de publicación de blogs, pero ha evolucionado para admitir otros tipos de contenido web, incluidas listas de correo y foros más tradicionales, galerías de medios, sitios de membresía, sistemas de gestión de aprendizaje (LMS) y tiendas en línea.

k. Trello

Trello es una plataforma que permite la planificación y organización de proyectos, de manera colaborativa y de metodología ágil mediante la creación de los diferentes sprints que deben de realizarse durante el transcurso del proyecto.

Trello fue la plataforma elegida para realizar el seguimiento del proyecto, junto a los directores del mismo, y para realizar la planificación para el transcurso del mismo.

Entre sus ventajas destacan que es una plataforma gratuita y con un interfaz muy sencillo de usar.

I. Ethereum

Ethereum es una plataforma que permite la creación de contratos inteligentes, de manera descentralizada, mediante el uso de la metodología

BlockChain. Ethereum ha sido la plataforma seleccionada para albergar los contratos inteligentes en el presente proyecto.

Entre las ventajas que brinda Ethereum destaca que es una plataforma gratuita, de fácil acceso y muy utilizada en el mundo del BlockChain, lo que permite que tenga una comunidad activa y un equipo amplio de desarrollo detrás.

3. Estado de la cuestión

A día de hoy, España es uno de los países de Europa que más depende de otros para comprar energía como petróleo, gas y carbón lo que le supone unos gastos de 25,13 billones de euros[1]. Sin embargo, España se puede convertir en una de las potencias energéticas más importantes de los próximos años. La sociedad se ha concienciado de la importancia del medioambiente y es por ello que ya no solo se busca rentabilidad en los proyectos, sino que también se busca sostenibilidad es sus tres facetas (social, económica y medioambiental). Por ello, se está realizando una gran transición de energías como el petróleo, el carbón y el gas natural a energías renovables y

[1] *Fuente: Acciona*

sostenibles. España tiene una de las mejores infraestructuras eólicas del mundo y además es uno de los países más soleados de Europa. Con la última regulación que ha aprobado el gobierno para incentivar a los prosumidores, todos los actuales consumidores se pueden convertir en prosumidores y beneficiarse del gran crecimiento que va a vivir este mercado a la vez que contribuyen a que España se convierta en una potencia de renovables. ¿Cuál es el problema? Son cinco las razones principales por las cuales no está aumentando el número de prosumidores en España:

1. **Inversión inicial**: Uno de los problemas más importante a día de hoy es que se requieren grandes inversiones iniciales para construir parques eólicos o parques fotovoltaicos. Por esta razón, el mercado eléctrico siempre ha estado restringido a grandes compañías o fondos de inversión. Para avanzar en esta línea, además de los cambios regulatorios que se están aprobando en España, ya se están desarrollando varias iniciativas entre las que destaca Fundeen, que se define como un crowdfunding de energías renovables.

2. **Trámites**: En muchos casos, los numerosos trámites que se tienen que llevar a cabo para la instalación de paneles solares provocan que el usuario tanto si quiere instalar paneles para un domicilio particular o para una comunidad de vecinos, abandone el intento de instalar renovables.

3. **Trazabilidad:** En la generación distribuida hacia la que nos dirigimos, es fundamental la certificación y trazabilidad de la energía para

identificar tanto el emisor como el receptor de la energía. En esta línea, hay varias empresas y startups, entre las que predomina Iberdrola, que ya han realizado proyectos piloto exitosos en los que utilizan la tecnología Blockchain para trazar la energía.

4. **Desinformación - Regulación:** Uno de los problemas fundamentales en España. Hay una gran incertidumbre en cuanto a la regulación ya que en España se están aprobando decretos para fomentar e incentivar a los prosumidores, pero existe una gran controversia y desinformación por parte de los ciudadanos respecto a este tema.

   Por otro lado, existe también una gran desinformación en cuanto a la rentabilidad de instalar renovables, el tipo de panel que se debe instalar y el tamaño del panel a instalar.

5. **Liquidez:** Como en cualquier inversión, uno de los principales objetivos de la inversión es obtener un buen rendimiento. A día de hoy, las energías renovables son muy buenas opciones de inversión ya que ofrecen rendimientos de entre el 5 y el 18 %, a la vez que contribuyen al cambio climático y a reducir la huella de carbono. Es por ello que además de ser inversiones con buena rentabilidad, tienen deducciones y bonificaciones fiscales que varían según el municipio con un valor medio del 20% en el IRPF (*Impuesto sobre la Renta de las Personas Físicas*), de hasta el 50 % en el IBI (*Impuesto de Bienes Inmuebles*) y de hasta el 95 % en el ICIO (*Impuesto sobre Construcciones, Instalaciones y Obras*). Sin embargo, uno de los

principales problemas de esta inversión es la falta de liquidez para poder desinvertir en un momento dado.

España es uno de los países de Europa con mayor número de horas de sol. Este factor, emparejado a los compromisos europeos de asentamiento de energías renovables, así como el objetivo estratégico de terminar con la dependencia energética de otros países y agrandar la independencia energética ha provocado que la producción de energía solar en España sea especialmente atractiva.

El primer parque solar en España se remonta a 1984. Fue en ese año que Iberdrola instaló en San Agustín de Guadalix la primera placa fotovoltaica conectada a la red. Esta conexión, de 100 kWp, fue la única con la que contó la península durante casi 10 años. Nueva años después, en 1993 se unieron cuatro sistemas, cada uno de 2,7 kWp, instalados por ATERSA en unas viviendas particulares de Pozuelo de Alarcón. Estas instalaciones dieron representación a un directorio de proyectos que cumplían más adecuadamente un papel demostrativo: 42 kWp en una academia en Menorca, 13,5 kWp en el Instituto de Energía Solar de la Universidad Politécnica de Madrid, 53 kWp en la Biblioteca de Mataró, e incluso una instalación de 1 MW en Toledo que tras su inauguración, el 7 de junio de 1994, se proclamó como la mayor central fotovoltaica de Europa. A finales de 1995, esta tecnología permanecía en el recorrido de la investigación, sin que se regulase en el contexto ascendiente del sistema eléctrico.

Con la regulación del RD 2818/1998 se establecieron primas de 60 y 30 pesetas por kWh inyectado a la red para sistemas con potencia nominal inferior y superior a 5 kWp respectivamente. Más tarde, con el RD 1663/2000, se establecieron condiciones técnicas y administrativas que representaron la verdadera apertura para la tecnología fotovoltaica en el sistema eléctrico español.

Sin embargo, a pesar de todos los incentivos, en 2004 la fotovoltaica no tuvo el desarrollo que se esperaba de ella, ya que suponía un 6.5 % del consumo de energía primaria en España.

Ante la insuficiente madurez de las renovables, la legislación cambió varias veces en poco tiempo. En 2004 se pasó del sistema de primas al abono de un porcentaje sobre la Tarifa Media de Referencia (TMR), y en 2007, se cambió de nuevo para fijar unas primas y tarifas reguladas fijas. Con este último cambio, las grandes instalaciones fotovoltaicas resultaron muy beneficiadas.

El futuro de la energía fotovoltaica en España está garantizado. Así lo demuestra el Plan Nacional Integrado de Energía y Clima (PNIEC), que traza la meta de un sector eléctrico 100% renovable en 2050, con una etapa intermedia del 74% en 2030. Precisamente para lograr esa meta de 2030 se prevé para entonces una potencia total instalada de 44 GW de energía solar, de los cuales 37 GW serán de fotovoltaica. Esto la convertirá en la tecnología de generación renovable de mayor crecimiento en los próximos 10 años. Por ello, se estima un fuerte crecimiento en los próximos años de la energía solar fotovoltaica.

Sin embargo, es importante rebajar la inversión mínima para participar en este tipo de instalaciones. Como se ha mencionado anteriormente, la mayor parte de inversión renovable en España está siendo liderada por las grandes empresas eléctricas españolas y grandes fondos de inversión. Dicho problema se puede solucionar con la creación de un crowdfunding para invertir en infraestructura renovable. De esta manera se divide la inversión inicial entre todos los participantes y cualquier usuario puede participar en la instalación renovable.

Por otro lado, con el crecimiento de la inversión en renovables también crecerá el número de fraudes. Como el aumento del interés en infraestructura renovable también crecerán los proyectos fraudulentos de supuestas instalaciones renovables. Por ello, es muy importante aportar seguridad y confianza a todos los procesos de instalación renovable en España. De esta manera, se aumentará la confianza en esta tecnología y cada vez más personas se animarán a conocer de primera mano las ventajas de invertir en renovables en España. Siguiendo esta línea el crowdfunding de infraestructuras renovables no consigue solucionar el problema de la seguridad. Sin embargo, si el crowdfunding se realiza a través de la tecnología Blockchain, se consiguen solucionar ambos problemas.

¿Qué es Blockchain?

La tecnología *Blockchain* nace en 2009 siendo la primera implementación conocida de la moneda digital o criptomoneda *Bitcoin*. Blockchain nace con el objetivo de revolucionar el sector financiero, pero a día de hoy está revolucionando prácticamente todos los sectores, incluyendo el sector eléctrico.

En las últimas dos décadas, se ha vivido una revolución de la información con el auge de plataformas que permiten la comunicación y transferencia de entre usuarios con independencia de su ubicación. Por tanto, cualquier persona se puede comunicar en tiempo real con otra persona que se encuentra al otro lado del mundo. Además, esta revolución también permite la transmisión de información (documentos, imágenes, videos entre otros) de forma instantánea a cualquier parte del mundo. La tecnología Blockchain supone la revolución del valor. Técnicamente, Blockchain es una tecnología basada en una base de datos pública, descentralizada y distribuida en la que se registran de forma segura las transacciones que se van realizando. Las transacciones se almacenan en bloques y los bloques están conectados entre sí formando una cadena.

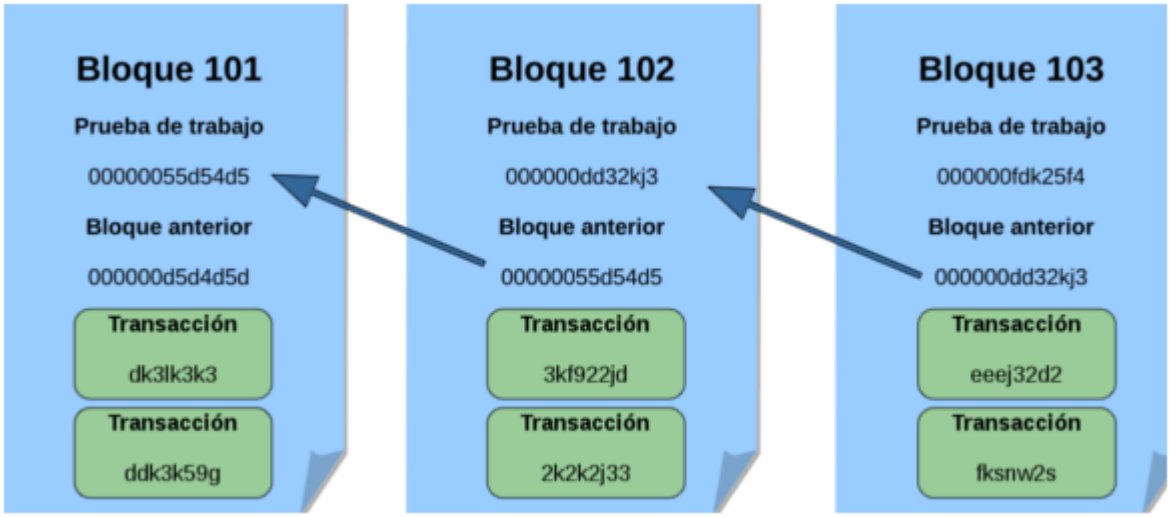
Por tanto, se trata de un libro de registro digital, que solo se puede actualizar cuando se alcanza la mayoría de consenso entre los participantes

de la red. Una vez introducida información en la “base de datos” distribuida, se crea una marca de tiempo y un enlace al bloque anterior. Por tanto, los datos introducidos en la cadena son verificables e inmutables.

Sin embargo, la utilidad de esta tecnología va mucho más allá del sector financiero, ya que a día de hoy se puede digitalizar cualquier activo. Una vez digitalizado, se puede transferir de forma descentralizada y segura sin necesidad de intermediarios a cualquier usuario de la red Blockchain.

Para entender el funcionamiento de la tecnología Blockchain hay que tener claros una serie de conceptos:

¿Qué es un Bloque?



Como se puede observar en la Figura 1, los bloques están formados por una serie de transacciones y están conectados entre sí. Cada bloque tiene una cabecera con información sobre el bloque y una serie de datos, que contiene todas la información y las transacciones que el bloque contiene. En toda red Blockchain, habrá un bloque raíz o primer bloque al que se irán uniendo los bloques consecutivos conforme se van creando nuevas transacciones. Las partes que forman un bloque son las siguientes:

1. Código alfanumérico o *hash* que sirve de enlace con el bloque anterior (Bloque anterior en la Figura 1)
2. El número de transacciones que existen en el interior del bloque
3. Otro código alfanumérico o *hash* que sirve de enlace para el bloque siguiente (Prueba de trabajo en la Figura 1). Dicho código se debe desencriptar con una serie de cálculos para que sea válido y los mineros son los encargados de realizar las operaciones para ir creando nuevos bloques.
4. Nonce: Aunque no aparezca en la Figura 1, cada bloque de la cadena tiene también un código alfanumérico o hash que se denomina nonce, y se explicará más adelante.

Por tanto, las transacciones que se incorporan a un bloque conectado a los bloques anteriores, son transacciones validadas. La labor de validación de las transacciones es la de los mineros. El bloque inicial o bloque raíz fue creado en 2009 con la creación del Bitcoin por Satoshi Nakamoto (Satoshi Nakamoto es el pseudónimo que usó para identificarse y a día de hoy no se conoce la identidad real de Satoshi Nakamoto).

¿Qué son los mineros?

Son los “usuarios” encargados de desencriptar los códigos alfanuméricos descritos anteriormente. Para descifrar los códigos alfanuméricos se necesitan una serie de ordenadores que cuentan con el software y el hardware necesario para realizar millones de cálculos por segundo y descifrar el código alfanumérico.

Cada vez que un minero descifra el código alfanumérico de un bloque, se verifican las transacciones de un bloque y dicho bloque se conecta a la cadena de bloques. Además, los mineros cuentan con el incentivo de que, si consiguen descifrar el código alfanumérico, se llevan una serie de criptomonedas como “comisión por el servicio”. Es decir, cada transacción lleva incorporada una comisión para el minero que la decide el creador de dicha transacción. Cuando el minero descifra el código alfanumérico del bloque, se lleva todas las comisiones de las transacciones incluidas en dicho bloque. Los mineros son piezas clave para el funcionamiento de la tecnología Blockchain ya que comprueban la veracidad de las transacciones a la vez que “crean” nuevas criptomonedas. Cabe destacar que según la red Blockchain en la que se está trabajando las criptomonedas serán distintas. Por tanto, si un minero verifica transacciones en la red pública de Bitcoin,

las criptomonedas con las que se trabajará en dicha red serán los bitcoins. Sin embargo, si un minero verifica transacciones en la red pública de Ethereum, que es la red de este proyecto, las criptomonedas con las que se trabajará serán los ethers.

¿Qué es el Nonce?

Como se ha mencionado anteriormente, uno de los códigos alfanuméricos o hash incorporados en cada bloque se denomina Nonce, que se conoce por sus siglas en inglés “Number that can be only used once”.

Por tanto, el nonce es un número completamente aleatorio que solo se puede utilizar una vez y es destinado a la autenticación de la transferencia de datos entre dos o más partes. El nonce implica también una marca de tiempo o *timestamp* en el bloque.

El cálculo del nonce se realiza de manera forzada, es decir, se requieren grandes cantidades de recursos de cómputo y de tiempo, por lo que conseguir este valor se necesita realizar una prueba de trabajo o *Proof of Work.*

¿Qué es Proof Of Work?

Prueba de trabajo es un tipo de algoritmo de consenso que se utiliza en Blockchain para alcanzar el consenso en la red. Existen otros algoritmos de consensor como *Proof of Stake* pero uno de los más completos y seguros es *Proof of Work*. Como se ha mencionado anteriormente, la creación de los bloques requiere de un gran cómputo computacional para resolver un acertijo matemático, que se resuelve mediante prueba y error. Dicho acertijo, es resuelto por los mineros descritos anteriormente que realizan

millones de intentos hasta resolver el acertijo, crear el bloque de transacciones y confirmar las transacciones involucradas en dicho bloque.

¿Qué son los nodos?

Por último, además de los mineros y de los bloques, un aspecto importante de la tecnología Blockchain son los nodos. Los nodos de Blockchain se definen como ordenadores conectado a la red de Blockchain, que se encargan de guardar y distribuir una copia actual de cada bloque.
Es decir, todos los nodos de la red tienen una copia de toda la información disponible en la cadena de bloques de la red. Por tanto, en caso de que un ataque cibernético ataque a uno o varios nodos de la red, la seguridad de la red se mantiene porque el resto de los nodos sigue manteniendo la copia de la cadena de bloques. Por ello, se dice que la tecnología Blockchain es mucho más segura que otras tecnologías ya que hasta antes de la aparición de la tecnología Blockchain, toda la información se almacenaba de forma centralizada en un punto o nodo. Por ello, a pesar de que dicho nodo sea muy seguro, siempre existe la posibilidad de que sea hackeado. Sin embargo, si toda la información de las transacciones está distribuida entre varios nodos, la seguridad se incrementa notablemente ya que el hacker o ataque cibernético tendría que atacar a todos los nodos de la red, que es prácticamente imposible. Además, la identidad de todos los nodos de la red, es anónima y las transacciones se realizan a través de la criptografía asimétrica, que aporta aún más seguridad a las transacciones.

¿Qué es la criptografía asimétrica?

La criptografía asimétrica es un método de desencriptación de datos compuesto por dos claves: una pública y una privada. La clave pública la puede tener cualquier persona mientras que la clave privada es intransferible para que la comunicación no se vea comprometida, es decir, solo puede tenerla uno de los extremos de la comunicación.

Para entender de manera más fácil su funcionamiento, se va a explicar mediante el ejemplo de una cuenta bancaria:

Una cuenta bancaria siempre tiene un código identificador o IBAN para que en el caso de que se quiera realizar una transferencia a esa cuenta bancaria, se pueda identificar. Por tanto, cualquier persona puede enviar dinero a una cuenta bancaria con el IBAN de la cuenta de destino.

Sin embargo, para acceder a los fondos de la cuenta bancaria, se necesita un PIN de 4 dígitos que se asocia a la clave privada de la criptografía asimétrica.

Como en la cuenta bancaria, en las redes Blockchain, cada usuario o nodo tiene una cartera (*wallet*) con una clave pública y una clave privada. La combinación de ambas claves permite la transmisión de información de forma segura entre dos partes. La clave es la encriptación y desencriptación del mensaje o transacción. Para encriptar un mensaje o transacción, se utiliza la llave pública del receptor que es conocida y la privada del emisor. Para desencriptarlo se utiliza la llave pública del emisor que es conocida y la llave privada del receptor. La llave privada es secreta y es la única que permite descifrar los mensajes. Lo que se consigue con este sistema es que ninguna tercera parte pueda descifrar la información en caso de que lograse interceptar la transacción.

¿Qué son las transacciones?

Una transacción se refiere a cualquier envío de un activo digital entre dos usuarios. Dicho activo digital puede ser cualquier activo desde dinero hasta un certificado. Una vez realizada la transacción, se almacena en un "pool" temporal de transacciones hasta que un minero verifique dicha transacción y la introduzca en un bloque de la cadena. Como se ha mencionado anteriormente, al realizar la transacción el usuario tiene que introducir la comisión que se lleva el minero por realizar la transacción. Es importante que dicha comisión no sea muy baja ya que, si es demasiado baja, la transacción tardará más tiempo en ser verificada por un minero.

4. Motivación

La motivación de este proyecto se centra en aprovechar la situación actual y resolver todos los problemas que se han mencionado anteriormente en una única plataforma.

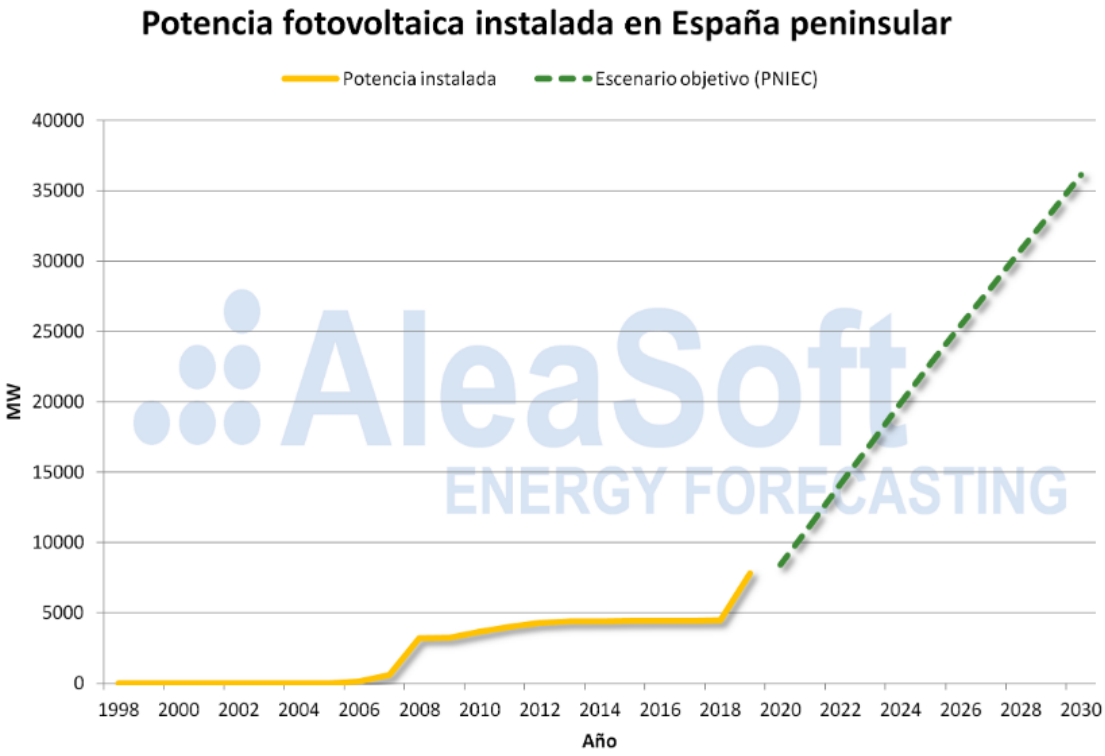
En primer lugar, aprovechar el momento porque el mercado eléctrico va a tener un gran aumento de demanda en los próximos años debido al desarrollo de tecnologías como Big Data, IA, Blockchain o los vehículos eléctricos, que requieren grandes cantidades de energía eléctrica para funcionar.

Además, los gobiernos están aprobando cada vez más medidas para fomentar e incentivar la inversión en instalaciones renovables y los ciudadanos están cada vez más concienciados con el cambio climático y la huella de carbono. Por ello, en los próximos años la generación de renovables va a tener un gran crecimiento y el objetivo de esta plataforma es aprovechar dicho crecimiento a la vez que contribuye a crear una sociedad más sostenible económicamente, socialmente y medioambientalmente. El creciente interés en este sector se puede observar en la creciente emisión de los bonos verdes en España, que se definen como activos de renta fija alineados con los ODS. Según El Economista, se emitieron más de 2000 millones de bonos verdes en el primer trimestre de 2019 en España. Aunque su rentabilidad sea baja (0,46 % según El Economista), hay un gran interés en España por este tipo de activo de deuda.

Como se ha mencionado anteriormente, el 99% de los usuarios en España tienen contadores inteligentes, por lo que pueden conocer su consumo de energía eléctrica por días o incluso por horas. Dicho conocimiento, unido a la generación propia de energía eléctrica a través de paneles solares puede contribuir a reducir drásticamente la factura de luz mediante la utilización de los dispositivos que más consumo producen en casa, como lavadoras o secadoras, a la misma hora que se obtiene el pico de generación solar cada día.

Por otro lado, la instalación de renovables va a tener un gran crecimiento en los próximos años. El gobierno español ya se ha comprometido a ello con la aprobación de la última regulación y la aprobación de planes como el Plan

Integrado de Energía y Clima en el que traza la meta de un sector 100 % renovable en 2050 con una etapa intermedia del 74 % en 2030.



Como se puede observar en la gráfica anterior, se estima un gran crecimiento en la instalación de potencia solar en España.

Sin embargo, toda la instalación que se va a instalar en los próximos años se debe financiar y supone una inversión inicial importante. Este año el mundo ha vivido una pandemia mundial que ha provocado tener que parar gran parte de la economía mundial. España, un país que tiene una gran dependencia de los servicios, se ha visto muy afectada por el virus lo que ha provocado miles de despidos y miles de cierres de empresas. Es por ello que se acerca una gran crisis que se espera que sea mucho peor que la crisis de España en 2008. Por ello, en los próximos años una gran parte de los ciudadanos no serán propensos a invertir en renovables debido a elevada inversión inicial que requiere. Sin embargo, si se divide la inversión inicial

necesaria entre varios participantes a través de un crowdfunding habrá más gente propensa a invertir en infraestructura renovable.

La cara buena de la moneda es que, con la evolución de la pandemia, los ciudadanos se han concienciado más de la importancia del medio ambiente. Por ello, una inversión exitosa a día de hoy no es únicamente una inversión rentable, sino que es una inversión rentable y sostenible en sus tres facetas (económica, social y medioambiental).

Por último, con

Como consecuencia del más que seguro incremento de las inversiones en infraestructuras renovables durante los próximos 10 años, también se puede hablar con certeza de un incremento de los fraudes en dicho mercado. Por ello, es muy importante aportar seguridad, confianza y transparencia a todo el proceso a través de la tecnología Blockchain.

5.Objetivos

El desarrollo de este proyecto busca dos principales objetivos que se centran tanto en el consumidor final del sector energético como en inversores externos en España:

1. **Contribuir al incremento de prosumidores & reducir la huella de carbono:** Con una plataforma sencilla de cara al usuario, se busca acercar a todos los ciudadanos a la inversión en renovables para que descubran en primera persona las ventajas que ofrece la inversión en

infraestructura renovable. De esta forma, se busca también reducir las fuentes de generación de energía no renovable.

2. **Aportar seguridad a las inversiones:** Como se ha mencionado, la inversión en renovables está creciendo y va a tener un gran crecimiento en los próximos años, pero con el crecimiento de este tipo de inversión, también crecerán los fraudes. Por ello, es muy importante conseguir una plataforma segura y trazable de cara al inversor.

3. **Reducir la inversión mínima:** Como se ha mencionado, uno de los principales problemas de la inversión en renovables es la elevada inversión inicial que se necesita. Por ello, uno de los principales objetivo de la plataforma es conseguir que un usuario pueda participar en la instalación de una infraestructura con inversiones de entre 50 y 100 euros.

Por otro lado, el desarrollo de este proyecto busca cumplir con algunos de los 17 Objetivos de Desarrollo Sostenible:

En primer lugar, la plataforma busca fomentar la energía limpia y facilitar el acceso a este tipo de energía a cualquier usuario con independencia de sus recursos. Es decir, la plataforma busca cumplir el ODS 7 garantizando el acceso a una energía asequible, segura, sostenible y moderna para todos.

En segundo lugar, la plataforma promueve la construcción de infraestructura renovable y sostenible, promoviendo así la industrialización inclusiva y sostenible.

Siguiendo esta línea, la plataforma fomenta la construcción de infraestructura sostenible con el objetivo de conseguir la autosuficiencia energética renovable, es decir, que no se requieran fuentes de generación de energía no renovable.

Por último, la plataforma busca combatir el cambio climático reduciendo la generación de energía no renovable y la polución provocada. El carbón y el petróleo son los dos combustibles más utilizados y a la vez son los combustibles más contaminantes. Emiten una gran cantidad de gases de efecto invernadero siendo los principales culpables del efecto invernadero y todas sus consecuencias. Por ello, la plataforma busca fomentar la instalación de parques solares para reducir al máximo la utilización de fuentes de energía contaminantes.

6.Metodología:

La resolución de todos los problemas mencionados anteriormente, se realiza a través de una página web sencilla e intuitiva para financiar la instalación de parques solares y permitir la inversión segura transparente y trazable a cualquier inversor sin necesidad de destinar a ello grandes cantidades de dinero. La plataforma consiste en un crowdfunding basado en Blockchain para financiar la instalación de infraestructura renovable. La plataforma tiene tres principales fases:

1. En primer lugar, informar al usuario del tamaño de panel que necesita. Como se ha mencionado, uno de los principales problemas a la hora de instalar paneles es la desinformación. Por ello, se le facilita al usuario un sencillo formulario para introducir tres datos fáciles de encontrar y obtener una estimación del tamaño del panel necesario. Para averiguarlo, se le solicita al usuario el consumo medio anual, la orientación de su vivienda y la localización de su vivienda. Con estos datos, se le otorga una estimación del tamaño del panel que debe instalar. El cálculo que se realiza en la plataforma, se detalla más adelante.

2. Una vez conocido el panel que se debe instalar, el siguiente paso es crear un crowdfunding para financiar dicha instalación. Dicho crowdfunding se desarrolla a través de la tecnología Blockchain para aportar seguridad, transparencia, trazabilidad e inmutabilidad a todo el proceso de inversión.

3. Por último, como se explica más adelante, la inversión total no va a una cuenta bancaria, sino que va a un contrato que está desplegado en la red Blockchain de Ethereum. Para sacar el dinero de dicho contrato, se tiene que realizar una votación en la que se establece la cantidad de dinero que se va a retirar y lo que se va a pagar con dicho dinero. Además, en dicha votación pueden participar todos los inversores.

Por tanto, el proyecto tiene tres partes fundamentales para su desarrollo:

En primer lugar, requiere un estudio de varios aspectos entre los que destacan la regulación actual relacionada con la generación independiente de energía, el cálculo del panel necesario a instalar y la viabilidad económica y tecnológica de la plataforma.

Por otra parte, el proyecto requiere un desarrollo informático con la creación de la plataforma, la programación de los Smart Contracts correspondientes y el desarrollo del código para crear la plataforma.

Por último, se requiere una última fase de prueba de la plataforma para corregir los errores que se presenten.

Para el desarrollo de las tres fases, es muy importante la planificación, que se ha realizado a través de dos plataformas: Trello para la planificación de objetivos o la planificación a corto plazo y el Diagrama de Gantt para la planificación a largo plazo.

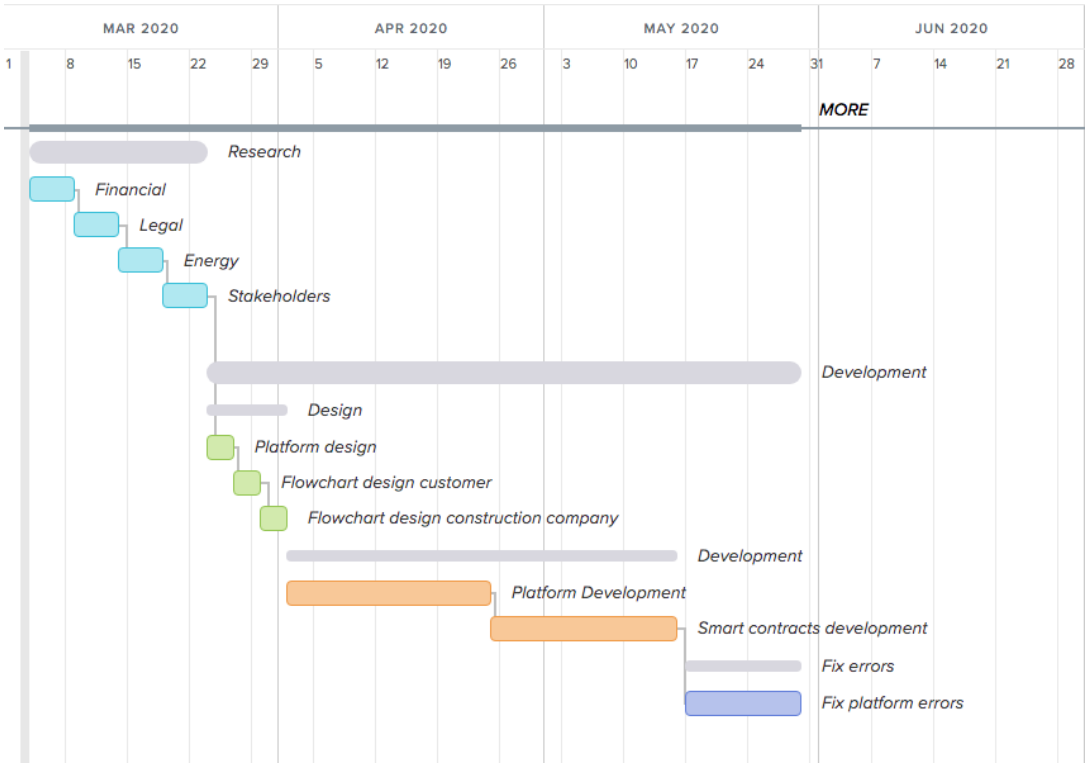


Figura 1: Diagrama de Gantt del proyecto (I)

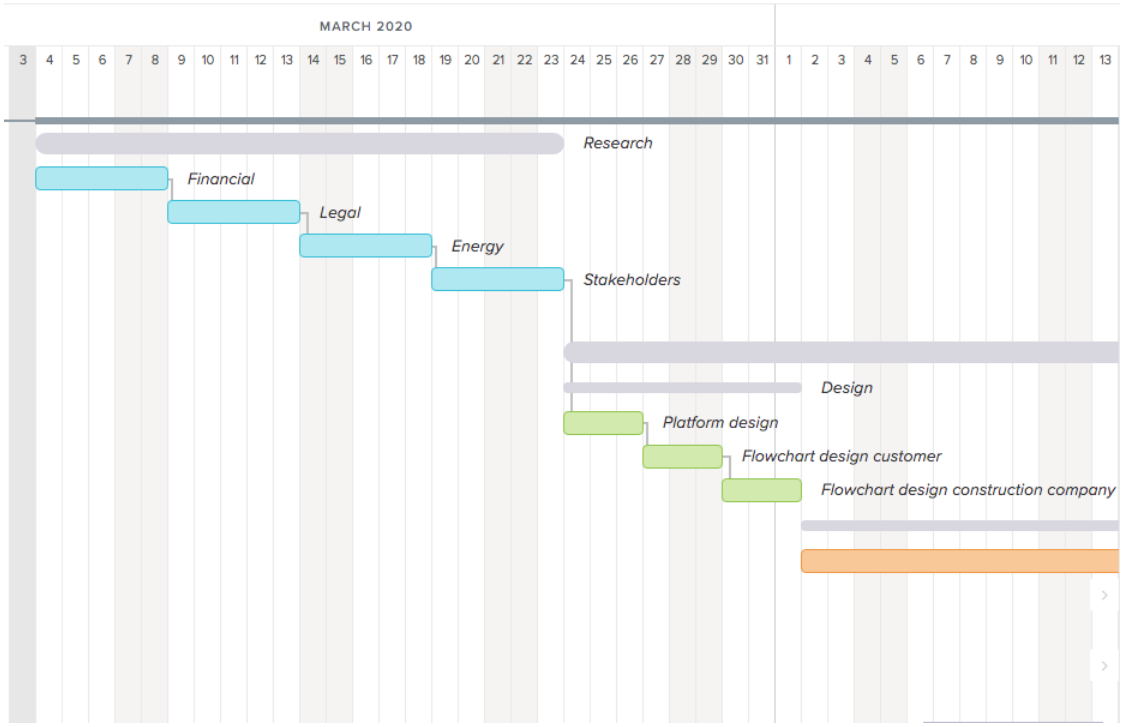


Figura 2: Diagrama de Gantt del proyecto (II)

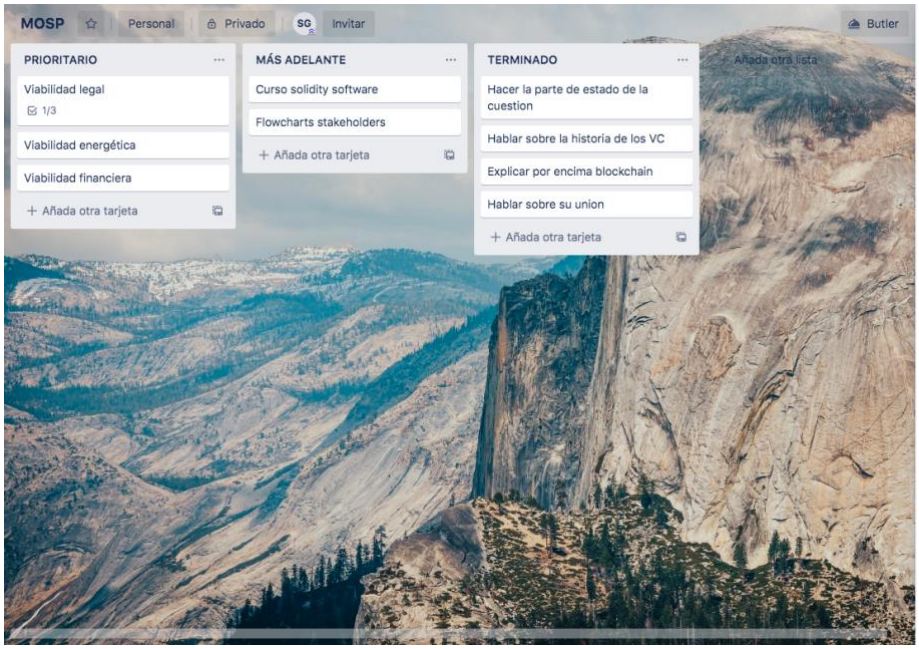


Figura 3: Diagrama de Trello del proyecto (I)

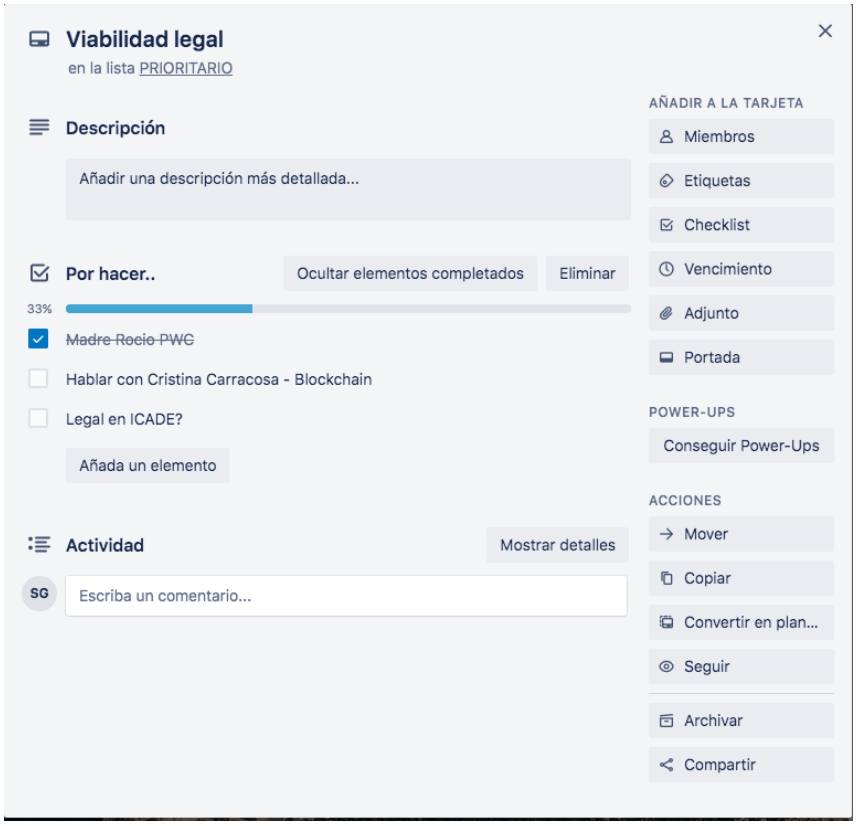


Figura 4: Diagrama de Trello del proyecto (II)

6.1 ¿Por qué Blockchain?

Para el desarrollo de la plataforma se va a utilizar la tecnología Blockchain. Concretamente, la tecnología Blockchain se utiliza para todo el proceso del Crowdfunding y la votación. Las ventajas que aporta la tecnología Blockchain en el proceso, se van a explicar con dos diagramas:

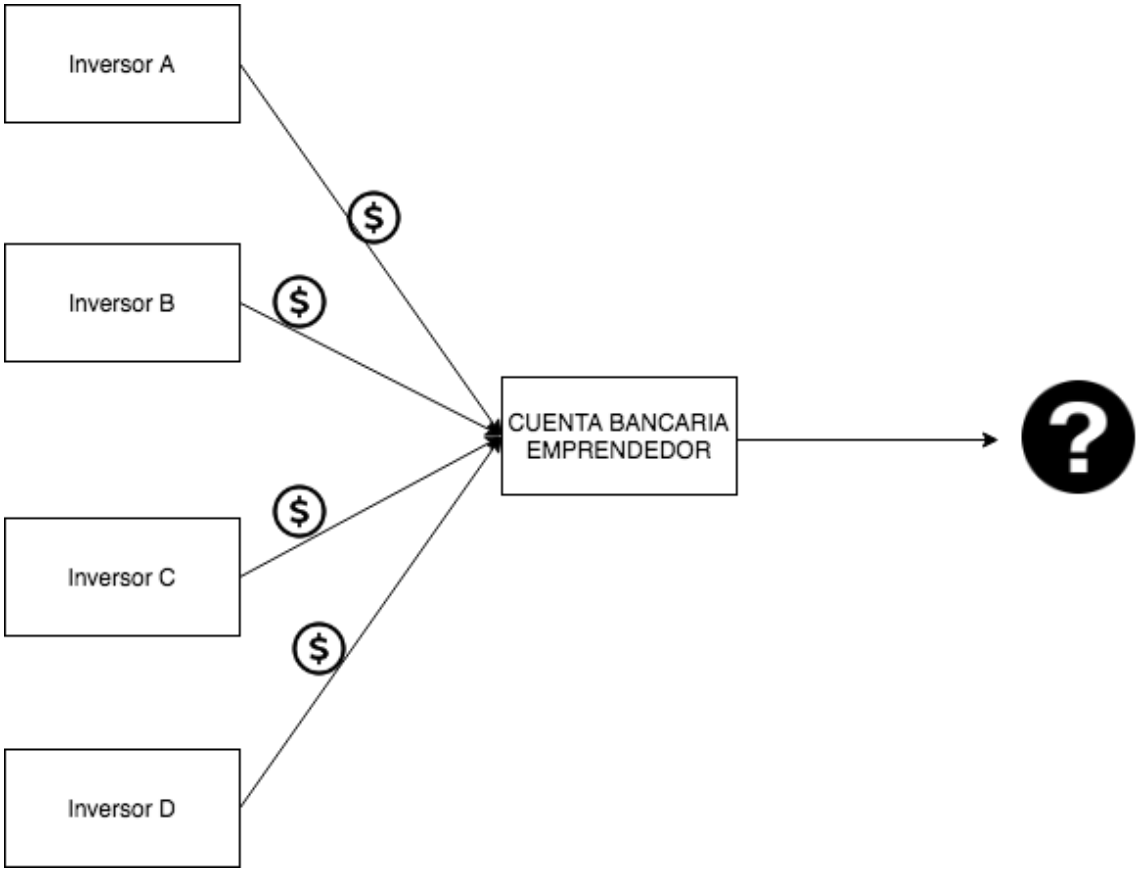


Figura 5: Diagrama de flujo crowdfunding convencional

Como se puede observar en la Figura 5, en un crowdfunding convencional, como puede ser Kickstarter, la metodología es la siguiente:

1. El emprendedor o creador del crowdfunding solicita una cantidad de dinero a cambio de un porcentaje de su compañía. Normalmente, el emprendedor establece lo que se va a realizar con los fondos. Por ejemplo, invertir en marketing o en la compra de materias primas.

2. Una vez creado el crowdfunding, los inversores interesados invierten en el proyecto en un periodo de tiempo prestablecido.

3. En caso de llegar al objetivo de inversión, el emprendedor recibe el dinero en su cuenta bancaria. En este punto el emprendedor puede utilizar dicho dinero para cumplir los objetivos que estableció en el plan de inversión o desaparecer con el dinero. Es por ello que a día de hoy en muchos crowdfundings se tiene la posibilidad de ser estafado.

En cambio, si se utiliza la tecnología Blockchain para el Crowdfunding, el diagrama de flujo es el siguiente:

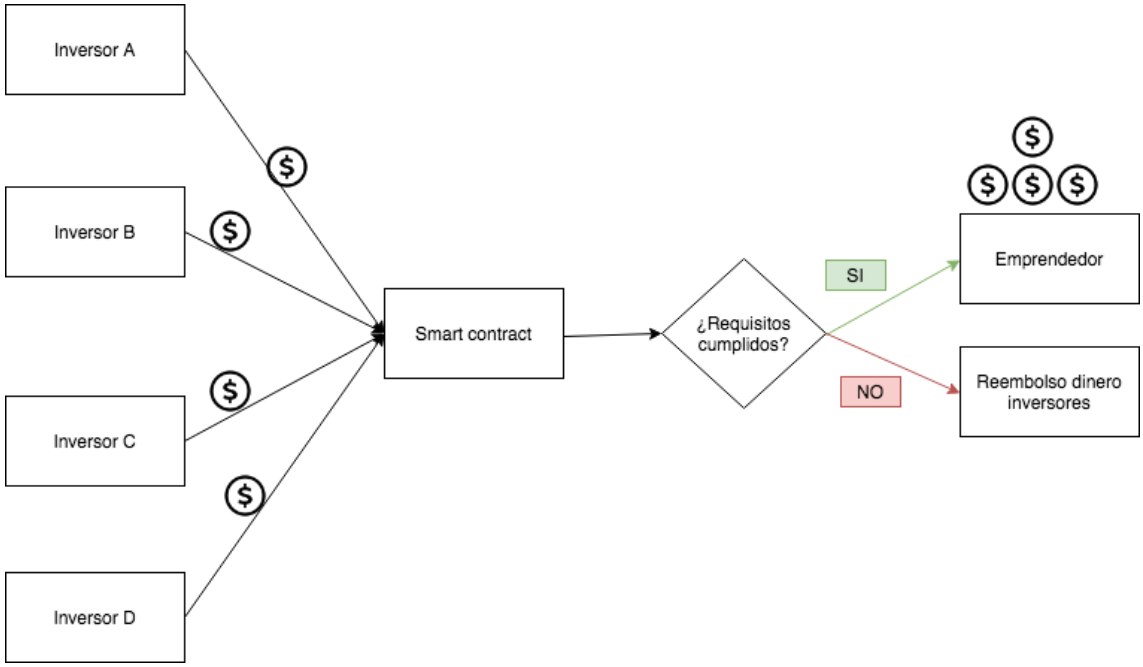


Figura 6: Diagrama de flujo crowdfunding blockchain

Como se puede observar, el dinero de los inversores va a un contrato inteligente no a la cuenta bancaria del emprendedor. En caso de que el emprendedor quiera recibir el dinero del contrato, se tienen que cumplir una serie de requisitos que se habrán prestablecido en el contrato. De esta manera, se aporta más control y transparencia a todo el proceso de pago. Los requisitos pueden ser de cualquier tipo, desde requisitos de tiempo hasta requisitos de votaciones. En este proyecto, el requisito que se ha incluido para realizar un pago es una votación en la que un tercio o más de los inversores, deben aprobar el envío de dinero.

6.2 ¿Por qué es una inversión interesante?

En este tipo de inversión, los inversores llevan a cabo la inversión que deriva en un ahorro energético para el consumidor final. En cada oportunidad de inversión se especifica la rentabilidad estimada de la inversión que suele estar en el rango del 6-10 %. Tras finalizar la instalación, los inversores son quienes obtienen la totalidad del ahorro económico provocado por el ahorro. Por otro lado, el consumidor, se hace dueño de la instalación para explotarla hasta el final de su vida útil (aproximadamente 25 años), cuando los inversores han obtenido la rentabilidad deseada. De esta forma el consumidor no lleva a cabo ninguna inversión y empieza a obtiene ahorros una vez los inversores hayan rentabilizado su inversión.

7.Sistema/Modelo desarrollado:

En este apartado, se explica de forma detallada el desarrollo del proyecto. Como se ha mencionado anteriormente, el desarrollo del proyecto se realizará en tres fases principales: La primera parte se centra en el estudio de la viabilidad del proyecto; la segunda parte se centra en el desarrollo tecnológico del proyecto con el desarrollo de los Smart Contracts y el desarrollo de un prototipo de web; por último, la fase final se centra en el testeo de la plataforma y la corrección de posibles errores.

7.1 Explicación de la plataforma

7.1.1 ¿Cuánta energía necesita cada usuario para autoconsumo?:

En el presente proyecto, se ha desarrollado una plataforma que permite invertir en la instalación de infraestructura renovable. La plataforma persigue aportar seguridad al proceso de inversión en infraestructura renovable a la par que reduce la inversión mínima requerida para participar en la instalación de infraestructura renovable.

Por lo tanto, para el desarrollo de la plataforma, el primer paso se centra en informar el tamaño del panel que el usuario necesita instalar. Cada usuario tiene un consumo de energía diferente y es por ello que el tamaño del panel necesario cambia según parámetros como la ubicación y orientación de la parcela, el tipo de panel que se instalará, la presencia o no de sombras y el uso o no uso de baterías. En este caso, tras realizar un

riguroso estudio, se ha llegado a la conclusión de que la mejor opción a día de hoy es instalar paneles solares sin el uso de baterías.

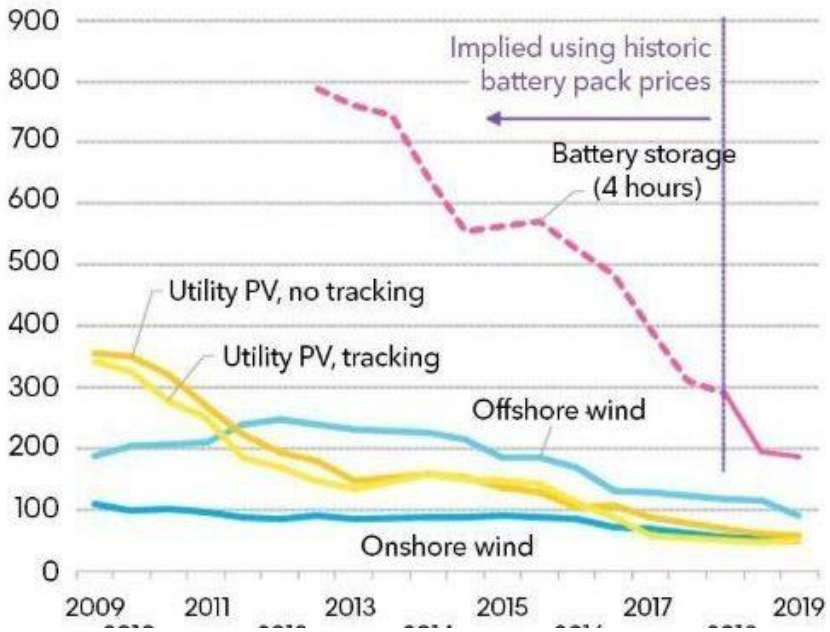


Figura 7: LCOE Energías Renovables

El LCOE (*Levelized Cost of Electricity*) es una medida del coste medio actual neto promedio de generación de electricidad para una planta generadora durante su vida útil. Como se puede observar en la Figura 7, el LCOE de las placas solares ha tenido un gran decrecimiento en los últimos años mientras que las baterías, aunque su LCOE se está reduciendo a pasos agigantados, sigue sin ser rentable. Sin embargo, en los próximos años, empezará a ser rentable unir el potencial de la energía solar fotovoltaica y el almacenamiento de la energía.

Para el cálculo del tamaño mínimo del panel, se ha utilizado la siguiente fórmula, suponiendo que la parcela no tiene sombras:

$$Energia = \frac{Consumomedioanual(kWh)}{365 * HSP}$$

Figura 8: Fórmula energía solar necesaria

En la fórmula de la figura 8 se tiene, por un lado, el consumo medio anual, que representa la media del consumo del usuario en kWh. Este dato se encuentra fácilmente en la factura de la luz del usuario. Por otro lado, se tiene la variable HSP que representa la hora solar pico. La hora solar pico es una unidad que mide la irradiación solar y se define como la energía por unidad de superficie que se recibiría con una hipotética irradiación solar constante de 1000 W/m². Por tanto, una Hora Solar Pico equivale a 1kWh/ m². Cabe destacar que el usuario no tiene que introducir la Hora Solar Pico, sino que tiene que introducir su Comunidad Autónoma y la plataforma en función de la Comunidad Autónoma que se elige, le asigna un valor a la variable HSP automáticamente.

Una vez conocida la cantidad de energía mínima que se necesita, el siguiente paso es calcular el tamaño mínimo del panel necesario para generar el consumo propio. Para realizar el cálculo del tamaño mínimo necesario, se utiliza la siguiente fórmula:

$$Tam = \frac{Energia * 1.15}{1000 * eficiencia}$$

Figura 9: Fórmula tamaño del panel

En la fórmula de la figura 9, se tiene por un lado la energía mínima necesaria calculada en el apartado anterior. Dicha energía se multiplica por un coeficiente de seguridad del 15%. Por otro lado, se tiene el valor de 1000W/m2 y la eficiencia del panel que varía entre el 15 y el 20%. En esta fórmula se escoge un valor de eficiencia del 17%.

7.1.2 Inversión en los paneles:

Una vez conocido el tamaño del panel necesario, se pasa a la siguiente fase que consiste en la financiación de los paneles. Dicha financiación o inversión se realizará a través de un Smart Contract que se explica más adelante.

Para la financiación, se crea un crowdfunding público en la plataforma. El diagrama de flujo del crowdfunding es el siguiente:

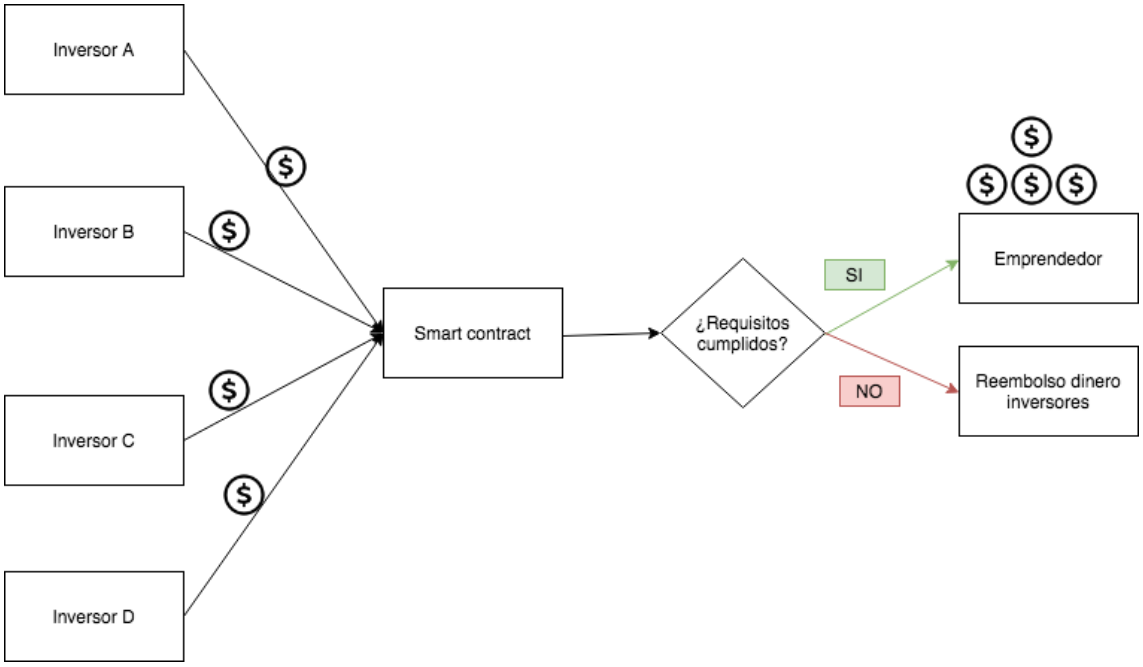


Figura 10: Diagrama de flujo pago crowdfunding

Como se puede observar en la Figura 10, una vez creado el crowdfunding, los inversores interesados invierten la cantidad que estimen oportuna. Si el emprendedor quiere sacar dinero del contrato, debe iniciar una votación en la que establece la cantidad de dinero que tiene que extraer y la utilidad que se le va a dar a dicho dinero. En caso de que 1/3 o más de los inversores voten a favor, el dinero saldrá del contrato y le llegará al emprendedor.

7.1.3 Votación para gastos:

Como se ha mencionado, el principal problema de los crowdfundings convencionales es la posibilidad de fraude. Por ello, se utiliza la tecnología Blockchain para el proceso de pago de los parques solares. Todo el dinero invertido se deposita en un contrato desplegado en la red Blockchain de Ethereum y para que el dinero salga del contrato, se debe aprobar en

votación. La votación está dirigida únicamente a inversores que hayan participado en el proyecto. La votación se realiza a través de una función dentro del Smart Contract que se explica más adelante.

Las votaciones son procesos atrasados que muchas veces suponen dificultades y dolores de cabeza para los integrantes de la votación. Los principales problemas a los que se suelen enfrentar los miembros de una votación son la privacidad o no privacidad de las votaciones y/o la imposibilidad de asistir a la votación. Por ello, la utilización de la tecnología Blockchain para la votación puede resultar muy útil ya que aporta las siguientes ventajas:

- Eficiencia: El proceso tradicional de votación es lento y laborioso. Sin embargo, la naturaleza descentralizada de la tecnología Blockchain aporta transparencia y elimina las dificultades administrativas de procesar los datos.

- Registros: Los resultados de las votaciones se almacenan con garantía de privacidad en un “libro mayor” que a su vez se almacena en todos los nodos de la red pública de Ethereum. Por tanto, los datos no se almacenan en puntos centrales evitándose así posibles ciberataques y pérdidas de datos.

- Seguridad y anonimato: Los resultados de la votación se almacenan en bloques cifrados con marca de tiempo inmutables, es decir, que no se pueden cambiar.

- Accesibilidad: Por último, la votación a través de la tecnología Blockchain permite a las personas emitir su voto desde cualquier parte del mundo. Por ello, los implicados en la votación pueden emitir su voto de manera segura sin necesidad de acudir presencialmente. Tras pasar una pandemia mundial, se puede afirmar que España está tecnológicamente preparada para pasar del mundo presencial al mundo telemático y virtual.

Por tanto, una vez establecido el tamaño del panel que se necesita, se puede crear un nuevo crowdfunding para financiar la instalación. Una vez creado, todos los usuarios tienen la posibilidad de invertir en el proyecto. Cabe destacar que el dinero invertido no va a la cuenta bancaria del manager o creador del crowdfunding, sino que va al contrato desplegado en la red Blockchain. Para sacar o extraer el dinero del contrato, se tiene que aprobar una votación en la que 1/3 o más de los inversores tienen que votar a favor.

7.2 Diagramas de flujo:

7.2.1 Diagrama de flujo plataforma:

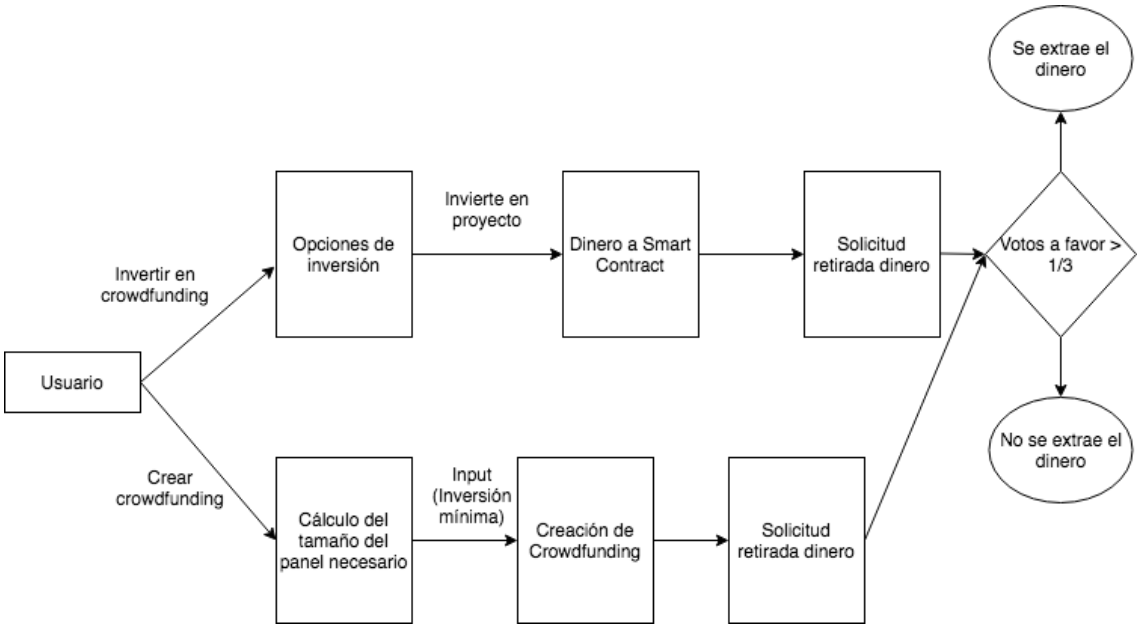


Figura 11: Diagrama de flujo plataforma

Como se puede observar en la Figura 11, el diagrama de flujo de la plataforma depende de si el usuario quiere invertir en infraestructura renovable o crear una nuevo crowdfunding para instalar un parque solar. En caso de que el usuario quiera invertir directamente en un crowdfunding, el único requisito que debe cumplir es que la inversión que realice sea mayor que la inversión mínima prestablecida por el creador del crowdfunding. En caso de invertir se convertirá en "votante" dentro del Smart Contract y si el creador del Crowdfunding quiere retirar dinero del Smart Contract, el usuario debe votar si permite la retirada de dinero o no la permite.

Por otro lado, en caso de ser un usuario que quiera crear un nuevo crowdfunding, en primer lugar, el usuario debe calcular el tamaño del panel que quiera instalar. Una vez calculado, el usuario debe crear un nuevo crowdfunding introduciendo únicamente la inversión mínima para participar en dicho crowdfunding. En caso de que quiera retirar el dinero de dicho Smart Contract, el creador del crowdfunding debe crear una votación estableciendo el dinero que se va a retirar y lo que se va a realizar con dicho dinero. Los participantes de dicha votación son los inversores que hayan invertido en el crowdfunding. En caso de que 1/3 o más de los inversores voten a favor, el creador del crowdfunding recibe su dinero. En caso de que la votación sea desfavorable, el creador del crowdfunding no recibe el dinero y tiene la posibilidad de realizar una nueva propuesta de votación para extraer el dinero.

7.3 Smart contracts de la plataforma:

Como se ha mencionado anteriormente, el primer paso consiste en calcular el tamaño aproximado que cada vecino necesita instalar.

Una vez conocido el tamaño del panel necesario, el siguiente es crear el Crowdfunding e iniciar el proceso de financiación. Todo el proceso de financiación se realiza a través de dos Smart Contracts que se explican más adelante.

7.3.1 Smart Contract Campaign.sol

Como se ha mencionado anteriormente, una vez calculado el tamaño del panel necesario, se debe proceder al pago de dicho panel. Para realizar el pago de la instalación, se crea un Crowdfunding en el que se debe establecer la inversión mínima para participar en el proyecto. El código de dicho contrato es el siguiente:

```
15
16  contract Campaign {
17      struct Request {
18          string description;
19          uint value;
20          address recipient;
21          bool complete;
22          uint approvalCount;
23          mapping(address => bool) approvals;
24      }
25
26      Request[] public requests;
27      address public manager;
28      uint public minimumContribution;
29      mapping(address => bool) public approvers;
30      uint public approversCount;
31
```

Figura 12: Campaign.sol (I)

En primer lugar, se declara la versión del lenguaje Solidity que se va a utilizar. En este caso, es la 0.4.25

Más tarde, se realiza la declaración de variables. En este Smart Contract se trabajará con las siguientes variables:

- Por un lado, se declara la estructura **Request,** que representa cada proposición de pago del Crowdfunding. La estructura es un tipo de

variable que sirve para almacenar distintos tipos de datos en su interior. En este caso, en el interior de la estructura de datos **Request**, se han introducido las siguientes variables:

- Una variable llamada **descripción** del tipo string. En ella, se introduce la descripción de la propuesta de pago.

- Una variable del tipo entero llamada **value**. En ella, se introduce la cantidad de dinero que se va a pagar.

- Una variable del tipo address llamada **recipient**. En ella, se introduce la address del receptor del dinero, es decir, la "identificación" del usuario al que se le va a pagar.

- Una variable del tipo booleana llamada **complete**. En ella, se establece si la request o propuesta de pago ha sido exitosa o no.

- Una variable del tipo entero llamada **approvalCount**. En ella, se introduce la cantidad de inversores que votan a favor de la propuesta de pago.

- Una variable del tipo mapping llamada **approvers.** Las variables del tipo mapping relacionan dos tipos de variables distintas, en este caso, una variable del tipo address y una variable del tipo booleana. Esta variable almacena la información de los inversores que a la vez votan en una propuesta determinada.

- Por otro lado, se declara el array de estructuras de datos Request denominada **requests**. Esta variable almacena todas las proposiciones de pago realizadas en un mismo proyecto.

- Más tarde, se declara la variable **manager**. En Solidity existe un tipo de variable denominada *address* que sirve para identificar a un usuario en la red Blockchain. Esta variable almacena la identidad en Blockchain del creador del Smart Contract que suele coincidir con el manager del contrato inteligente. Por último, se declara como *public* con el objetivo de que, en cualquier momento, cualquier usuario pueda ver la *address* del presidente de la comunidad.

- Seguidamente, se declara la variable **minimumContribution** que es del tipo entero. Esta variable almacena la inversión mínima de cada Crowdfunding.

- Más tarde se declara la variable **approvers** que es del tipo de datos mapping relaciona dos tipos de datos distintos, en este caso una *address* con un booleano. En esta variable se almacenarán todos los votantes e inversores implicados.

- Más tarde, se declara la variable **approversCount** que es del tipo entero, que almacena la cantidad de inversores que aprueban una determinada propuesta de pago.

- Por último, se declara la variable **vecinos** que es del tipo entero para almacenar el número de vecinos de la comunidad.

Tras realizar la declaración de variables, se declaran los modifiers. Los modifiers en Solidity son complementos a las funciones que se declaran más adelante en el Smart Contract. Permiten crear características adicionales o agregar restricciones a las funciones.

En este caso hay un modifier en el Smart Contract que es **restricted**:

```
31
32      modifier restricted() {
33          require(msg.sender == manager);
34          _;
35      }
36
```

Figura 13: Campaign.sol (II)

En este caso, se utiliza un modifier denominado **restricted**, que se utiliza para restringir el acceso a una función únicamente al manager o creador del contrato inteligente.

7.3.1.1 Función constructor

Una vez declaradas las variables y los modifiers, el siguiente y último paso en este caso, es declarar las funciones. En este contrato, hay siete funciones contando con la función constructor. La función constructor es la función que se ejecuta automáticamente cuando se ejecuta el contrato:

```
36
37      function Campaign(uint minimum, address creator) public {
38          manager = creator;
39          minimumContribution = minimum;
40      }
41
```

Figura 14: Función constructor campaign.sol (I)

Como se puede observar en la Figura 14, la función constructora tiene dos inputs, la inversión mínima y la address del creador del contrato inteligente. Una vez dentro de la función, se establece el manager como el usuario que ha creado el contrato y se crea un nuevo crowdfunding. Cabe destacar que la función constructora se puede declarar con el nombre constructor o con el mismo nombre que el Smart Contract. En este caso, se ha utilizado el mismo nombre para el nombre del Smart Contract y el nombre de la función constructora.

7.3.1.2 Función contribute

```
function contribute() public payable {
    require(msg.value > minimumContribution);

    approvers[msg.sender] = true;
    approversCount++;
}
```

Figura 15: Función contribute

Una vez creado el crowdfunding y ejecutado el contrato, una de las funciones públicas (cualquier usuario puede ejecutarlas) es contribute. Esta función no tiene ningún input. Además, tiene la característica payable para permitir el envío de dinero.

Por último, esta función verifica que la cantidad invertida sea mayor que la contribución mínima establecida al crear el Crowdfunding.

7.3.1.3 Función createRequest:

```
function createRequest(string description, uint value, address recipient) public restricted {
    Request memory newRequest = Request({
       description: description,
       value: value,
       recipient: recipient,
       complete: false,
       approvalCount: 0
    });

    requests.push(newRequest);
}
```

Figura 16: Función createRequest

Una vez creada la votación y ejecutado el contrato, otra de las funciones públicas (cualquier usuario puede ejecutarlas) es createRequest. Esta función tiene tres inputs:

- En el primer input se debe introducir un string para introducir la descripción de la propuesta de pago.

- En el segundo input se debe introducir la cantidad de dinero que se va a extraer del contrato

- En el tercer input se debe introducir la address del receptor del dinero, es decir, la identificación en el mundo Blockchain del receptor del dinero.

Cabe destacar que esta función tiene el modifier restricted ya que únicamente el manager del contrato puede realizar una propuesta de pago.

7.3.1.4 Función approveRequest

```
function approveRequest(uint index) public {
    Request storage request = requests[index];

    require(approvers[msg.sender]);
    require(!request.approvals[msg.sender]);

    request.approvals[msg.sender] = true;
    request.approvalCount++;
}
```

Figura 17: Función approveRequest

Una vez creada la votación y ejecutado el contrato, otra de las funciones públicas (cualquier usuario puede ejecutarlas) es approveRequest. Esta función tiene un input, que es el índice de la propuesta que se quiere aprobar. Es decir, en caso de que haya tres propuestas de pago y el usuario quiera aprobar la primera de ellas, debe introducir el índice 0 como input al llamar a la función approveRequest.

Por otro lado, esta función controla que un usuario no haya votado más de una vez y que el usuario haya invertido en el crowdfunding.

7.3.1.5 Función numero_votos_favor

```
function finalizeRequest(uint index) public restricted {
    Request storage request = requests[index];

    require(request.approvalCount > (approversCount / 3));
    require(!request.complete);

    request.recipient.transfer(request.value);
    request.complete = true;
}
```

Figura 18x: Función finalizeRequest

Esta función de carácter público como las anteriores, se utiliza para finalizar la votación de la propuesta de pago. Como en la función anterior, tiene un único input que es el índice de la propuesta de pago que se quiere finalizar.

Además, esta función controla que al menos 1/3 de los inversores hayan votado a favor y que no se haya realizado ya el envío del dinero.

En caso de que se cumplan todas las condiciones, esta función enviará el dinero especificado en la función createRequest a la address del receptor del dinero especificado también en la función createRequest.

7.3.1.6 Función getSummary

```
function getSummary() public view returns (
  uint, uint, uint, uint, address
  ) {
    return (
      minimumContribution,
      this.balance,
      requests.length,
      approversCount,
      manager
    );
}
```

Figura 19: Función getSummary

Esta función se utiliza para obtener información de un determinado crowdfunding. La información que devuelve esta función es la inversión mínima del Crowdfunding, el balance, el número de propuestas de pago, el número de inversores que han votado a favor y la address del manager o creador del contrato. Por tanto, esta función es meramente informativa por lo que lleva el atributo view.

7.3.1.7 Función getRequestsCount

```
    function getRequestsCount() public view returns (uint) {
        return requests.length;
    }
}
```

Figura 20: Función getRequestsCount

Por último, la última función del contrato es getRequestsCount que no tiene inputs ni modifiers. Como en el caso anterior, esta función es meramente informativa y devuelve el número de propuestas de pago de un determinado Crowdfunding.

7.3.2 Smart Contract CampaignFactory.sol:

Para tener una conexión entre todos los Crowdfundings creados, en esta plataforma se crea otro contrato (CampaignFactory) que almacenará todos los Crowdfundings creados. El código de dicho contrato es muy corto ya que su objetivo es contener todos los Crowdfundings creados. El código del Smart Contract CampaignFactory.sol es el siguiente:

```
pragma solidity ^0.4.25;

contract CampaignFactory {
    address[] public deployedCampaigns;

    function createCampaign(uint minimum) public {
        address newCampaign = new Campaign(minimum, msg.sender);
        deployedCampaigns.push(newCampaign);
    }

    function getDeployedCampaigns() public view returns (address[]) {
        return deployedCampaigns;
    }
}
```

Figura 21: CampaignFactory.sol

Como en el contrato anterior, en primer lugar, se declara la versión del lenguaje Solidity que se va a utilizar. En este caso, es la 0.4.25

Más tarde, se realiza la declaración de variables. En este Smart Contract se trabajará con las siguientes variables:

- En primer lugar, se declara la variable **deployedCampaigns** que es un array y almacena todos los Crowdfundings creados

- Seguidamente, se declara la función **createCampaign** que se utiliza para crear un nuevo Crowdfunding. Esta función tiene un único input que es la inversión mínima del Crowdfunding.

- Por último, se declara la función **getDeployedCampaigns** que se utiliza para obtener información de todos los Crowdfundings creados. Cabe destacar que esta función es meramente informativa por lo que no tiene inputs, es de carácter view y devuelve las address de los crowdfundings desplegados.

7.4 Test de la plataforma:

Una vez explicado el código de los Smart Contracts, se procede a realizar el testeo del código de los contratos mediante Mocha. El código utilizado se explica más adelante. Como se ha mencionado anteriormente, los objetivos principales que la plataforma persigue son la seguridad y la inversión mínima. Por ello, para comprobar el correcto funcionamiento de la plataforma se van a realizar seis tests:

```
it('despliega un factory y un crowdfunding', () => {
  assert.ok(factory.options.address);
  assert.ok(campaign.options.address);
});
```

Figura 22: Test despliegue de ambos Smart Contracts

El primer test de la plataforma consiste en comprobar que a cada Smart Contract desplegado en la red Blockchain, se le asigna una *address* correctamente.

```
it('establece al creador del contrato como manager', async () => {
  const manager = await campaign.methods.manager().call();
  assert.equal(accounts[0], manager);
});
```

Figura 23: Test creador del crowdfunding

El segundo test de la plataforma consiste en comprobar que el creador del crowdfunding es asignado como manager o creador del Smart Contract.

```
it('permite a los usuarios invertir y los introduce en el mapping de approvers', async () => {
  await campaign.methods.contribute().send({
    value: '200',
    from: accounts[1]
  });
  const isContributor = await campaign.methods.approvers(accounts[1]).call();
  assert(isContributor);
});
```

Figura 24: Test inversores

El tercer test de la plataforma consiste en comprobar que un usuario puede invertir en un proyecto. Además, comprueba si tras invertir se le introduce al usuario en el mapping de inversores del proyecto.

```
it('te pide inversión mínima', async () => {
  try {
    await campaign.methods.contribute().send({
      value: '5',
      from: accounts[1]
    });
    assert(false);
  } catch (err) {
    assert(err);
  }
});
```

Figura 25: Test Inversión mínima

El cuarto test de la plataforma consiste en comprobar si se cumple el requisito de inversión mínima en el crowdfunding.

```
it('permite al manager hacer proposición de pago', async () => {
  await campaign.methods
    .createRequest('Panel Solar', '100', accounts[1])
    .send({
      from: accounts[0],
      gas: '1000000'
    });
  const request = await campaign.methods.requests(0).call();

  assert.equal('Panel Solar', request.description);
});
```

Figura 26: Test proposición de pago

El quinto test de la plataforma se centra en comprobar que el creador o manager del crowdfunding puede realizar una proposición de pago para sacar dinero del Smart Conctract.

```
  it('Procesa una propuesta de pago', async () => {
    await campaign.methods.contribute().send({
      from: accounts[0],
      value: web3.utils.toWei('10', 'ether')
    });

    await campaign.methods
      .createRequest('A', web3.utils.toWei('5', 'ether'), accounts[1])
      .send({ from: accounts[0], gas: '1000000' });

    await campaign.methods.approveRequest(0).send({
      from: accounts[0],
      gas: '1000000'
    });

    await campaign.methods.finalizeRequest(0).send({
      from: accounts[0],
      gas: '1000000'
    });

    let balance = await web3.eth.getBalance(accounts[1]);
    balance = web3.utils.fromWei(balance, 'ether');
    balance = parseFloat(balance);

    assert(balance > 104);
  });
});
```

Figura 27: Test votación propuesta de pago

Una vez desarrollado el código de los tests, se procede a la realización de los mismos en Terminal. Para ello, se ejecuta el código *npm run test*.

Los resultados obtenidos son los siguientes:

```
MacBook-Air-de-Gericke:TFG_CODIGO Gericke$ npm run test

> tfg_codigo@1.0.0 test /Users/Gericke/Desktop/TFG_CODIGO
> mocha


  Campaigns
    ✓ despliega un factory y un crowdfunding
    ✓ establece al creador del contrato como manager
    ✓ permite a los usuarios invertir y los introduce en el mapping de approvers (123ms)
    ✓ te pide inversión mínima (40ms)
    ✓ permite al manager hacer proposición de pago (136ms)
    ✓ Procesa una propuesta de pago (260ms)


  6 passing (2s)
```

Figura 28: Resultados testeo

Como se puede observar en la Figura 28, todos los tests realizados han funcionado correctamente.

7.5 Servidor de la plataforma:

La plataforma tendrá dos servidores principales, el servidor de Wordpress que se utiliza para la página web y el servidor que realiza la conexión entre la página web y la red de Blockchain (Ethereum). El diagrama de flujo que representa ambos servidores es el siguiente:

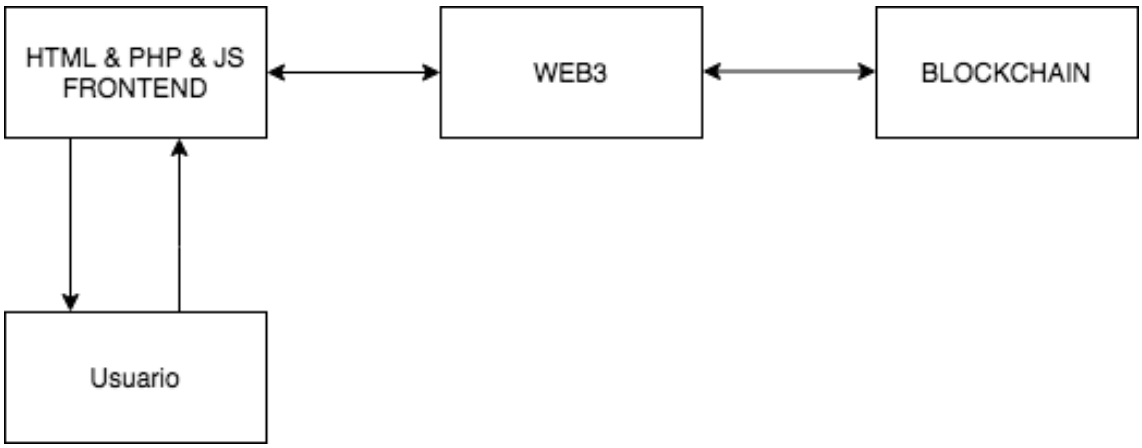


Figura 29: Servidor Web3

Por un lado, se tiene el servidor de Web3. Dicho servidor sirve como conexión entre la web y el mundo Blockchain.

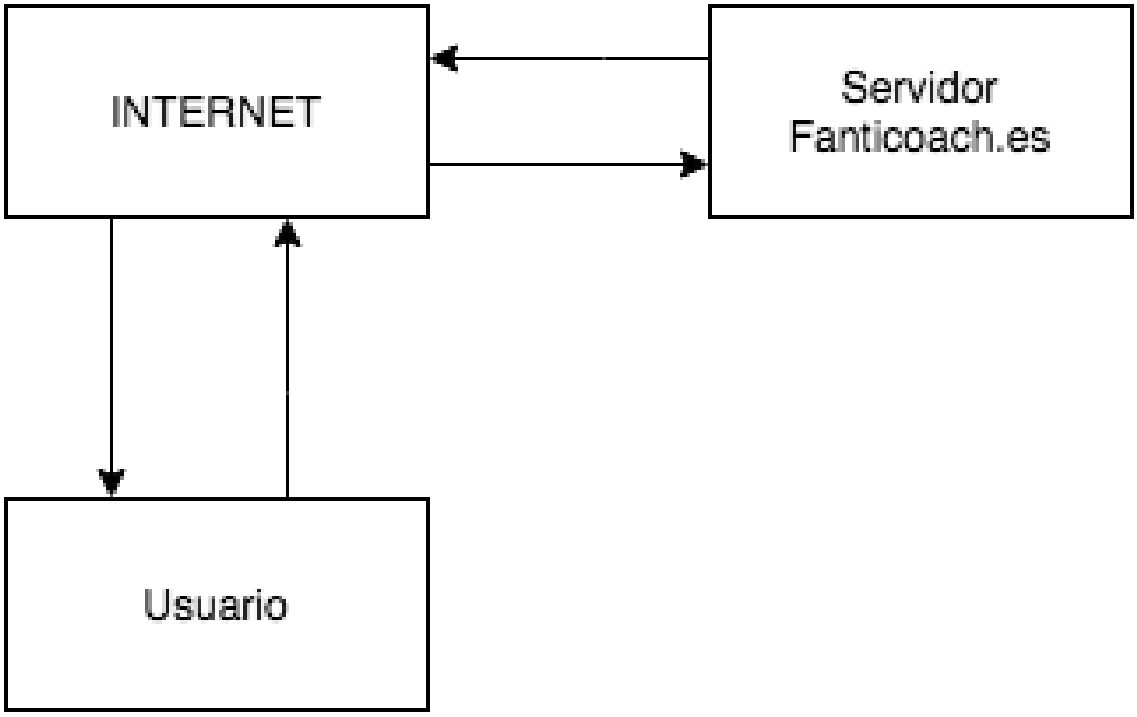


Figura 30: Servidor Web

Por otro lado, se tiene el servidor de la web principal (fanticoach.es/principal). Dicha web se ha desarrollado a través de Wordpress. El funcionamiento es que se refleja en la Figura 30: El usuario envía una serie de inputs que van a internet y son procesados por el servidor web. Una vez recibidos, el servidor devuelve una respuesta que va al usuario.

En primer lugar, los Smart Contracts que se han mencionado anteriormente se tienen que desplegar de forma local. El código utilizado para ello se refleja en la siguiente imagen.

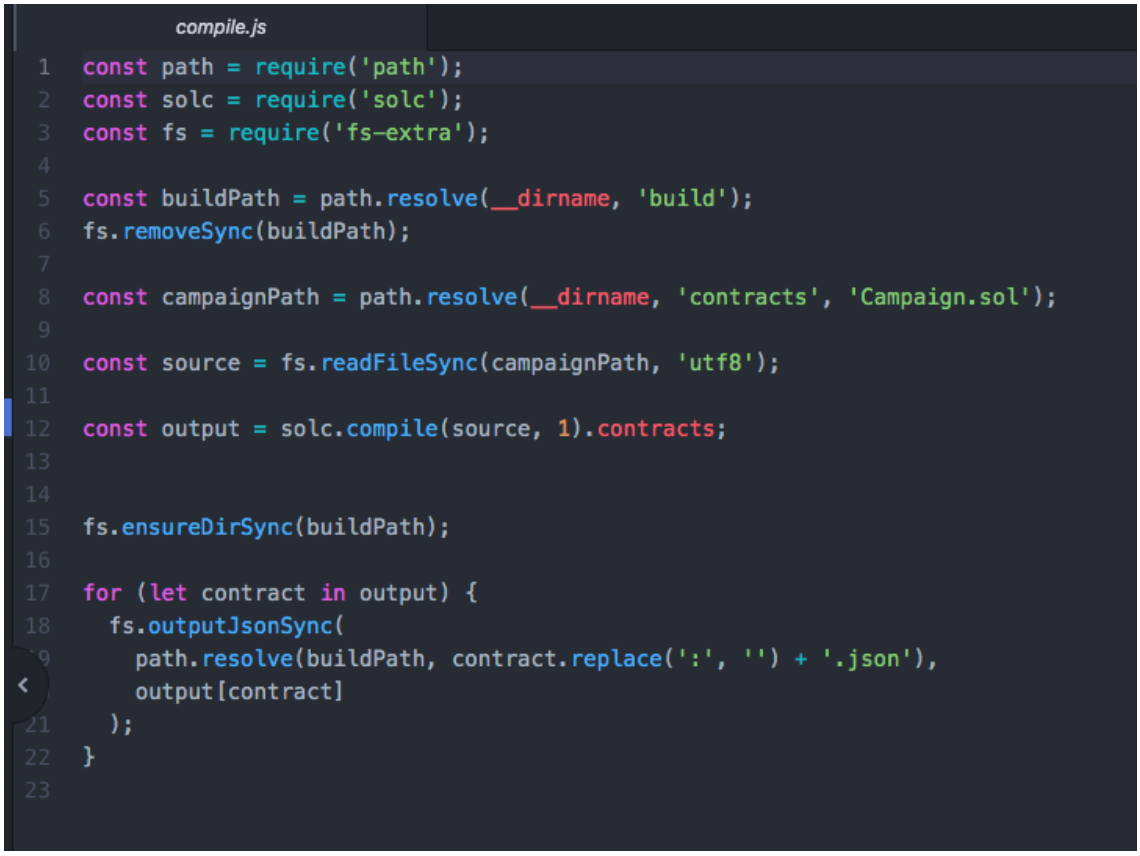
deploy.js

```
const HDWalletProvider = require('truffle-hdwallet-provider');
const Web3 = require('web3');
const compiledFactory = require('./build/CampaignFactory.json');

const provider = new HDWalletProvider(
  'belt swing impose sketch input liquid hen lucky bronze notable pave spread',
  'https://rinkeby.infura.io/v3/38f5b3524c3c4be6a211bdce5408d577'
);
const web3 = new Web3(provider);

const deploy = async () => {
  const accounts = await web3.eth.getAccounts();

  console.log('Attempting to deploy from account', accounts[0]);

  const result = await new web3.eth.Contract(
    JSON.parse(compiledFactory.interface)
  )
    .deploy({ data: compiledFactory.bytecode })
    .send({ gas: '1000000', from: accounts[0] });

  console.log('Contract deployed to', result.options.address);
};
deploy();
```

Figura 31: Servidor Web

Una vez desplegados los Smart Contracts, el siguiente paso es compilarlos.

El código utilizado para ello es el siguiente:



compile.js

```
const path = require('path');
const solc = require('solc');
const fs = require('fs-extra');

const buildPath = path.resolve(__dirname, 'build');
fs.removeSync(buildPath);

const campaignPath = path.resolve(__dirname, 'contracts', 'Campaign.sol');

const source = fs.readFileSync(campaignPath, 'utf8');

const output = solc.compile(source, 1).contracts;


fs.ensureDirSync(buildPath);

for (let contract in output) {
  fs.outputJsonSync(
    path.resolve(buildPath, contract.replace(':', '') + '.json'),
    output[contract]
  );
}
```

Figura 32: Servidor Web

Una vez compilado, el código que se muestra en la Figura 32 devuelve un address que es la identificación del Smart Contract subido a la red Ethereum.

Por último, falta la conexión entre el mundo Blockchain y el front-end del usuario. Como se ha mencionado anteriormente, la conexión se realiza a través de Web3 y el código que se ha utilizado es el siguiente:

web3.js

```
import Web3 from 'web3';

let web3;

if (typeof window !== 'undefined' && typeof window.web3 !== 'undefined') {
  // We are in the browser and metamask is running.
  web3 = new Web3(window.web3.currentProvider);
} else {
  // We are on the server *OR* the user is not running metamask
  const provider = new Web3.providers.HttpProvider(
    'https://rinkeby.infura.io/v3/38f5b3524c3c4be6a211bdce5408d577'
  );
  web3 = new Web3(provider);
}

export default web3;

```

Figura 33: Código Web3

8. Análisis de resultados:

Como se ha mencionado anteriormente, los objetivos del proyecto eran los siguientes:

1. **Contribuir al incremento de prosumidores & reducir la huella de carbono:** Con una plataforma sencilla de cara al usuario, se ha conseguido que el usuario pueda calcular con facilidad una aproximación del panel que debe instalar. Por otro lado, se ha logrado incentivar la inversión en paneles solares reduciendo la inversión mínima y aportando más seguridad al proceso de inversión. Por ello, el desarrollo de esta plataforma ha conseguido de forma indirecta contribuir al auge de los prosumidores y reducir la huella de carbono facilitando los procesos que hay detrás de instalar un parque solar.

2. **Aportar seguridad a las inversiones:** La seguridad es una de las piezas claves de esta plataforma. Este objetivo se ha conseguido con el desarrollo exitoso de una plataforma en la que la inversión no se envía directamente al creador del crowdfunding, sino que se envía a un Smart Contract desplegado en la red Blockchain. De esta forma, para que el dinero salga del Smart Contract, una gran parte de los inversores tienen que dar su aprobación lo que aporta una gran seguridad al proceso.

3. **Reducir la inversión mínima:** Por último, otro de los objetivos principales de la plataforma es acercar la inversión en renovables a los ciudadanos de España. Dicho objetivo se ha conseguido mediante la

creación exitosa de un crowdfunding en el que la inversión inicial se divide entre todos los participantes y la inversión mínima no es alta.

Por tanto, se ha desarrollado una plataforma en la que:

1. Cualquier persona puede crear un crowdfunding para instalar paneles solares y convertirse en prosumidor sin necesidad de desembolsar una inversión inicial alta.

2. Cualquier persona puede invertir en infraestructura renovable ya que la inversión mínima es baja.

3. Cualquier persona puede invertir sin preocuparse por posibles fraudes o hackeos. Tras invertir, los inversores siguen teniendo control sobre la inversión total.

9. Trabajos futuros:

En conclusión, en este proyecto se ha intentado fomentar la instalación de infraestructura renovable en España. El proyecto se ha centrado en el proceso de inversión y creación del Crowdfunding y si tiene suficiente volumen, se pueden agregar ciertas funcionalidades de cara al futuro como:

1. **ONG**: Una de las funcionalidades que sería interesante agregar es la de instalar paneles solares en lugares soleados en los que los ciudadanos tengan pocos recursos, como en África. De esta manera, el inversor tiene la certeza de que está ayudando realmente a gente necesitada gracias a la naturaleza descentralizada y transparente de Blockchain. Además, se contribuye a reducir la pobreza, potenciar la infraestructura del país y a reducir la huella de carbono entre otros Objetivos de Desarrollo Sostenible.

2. **Exchange**: Por otro lado, una funcionalidad que sería muy interesante agregar sería la posibilidad de hacer un Exchange dentro de la Plataforma. Para ello, la plataforma tendría su propio token en el que 1 token representaría 1kWh solar. De esta manera, los inversores de infraestructura renovable podrían liquidar su inversión en cualquier momento a través del Exchange. El Exchange funcionaría como cualquier Exchange Blockchain en el que un usuario cambia dinero Fiat por tokens.

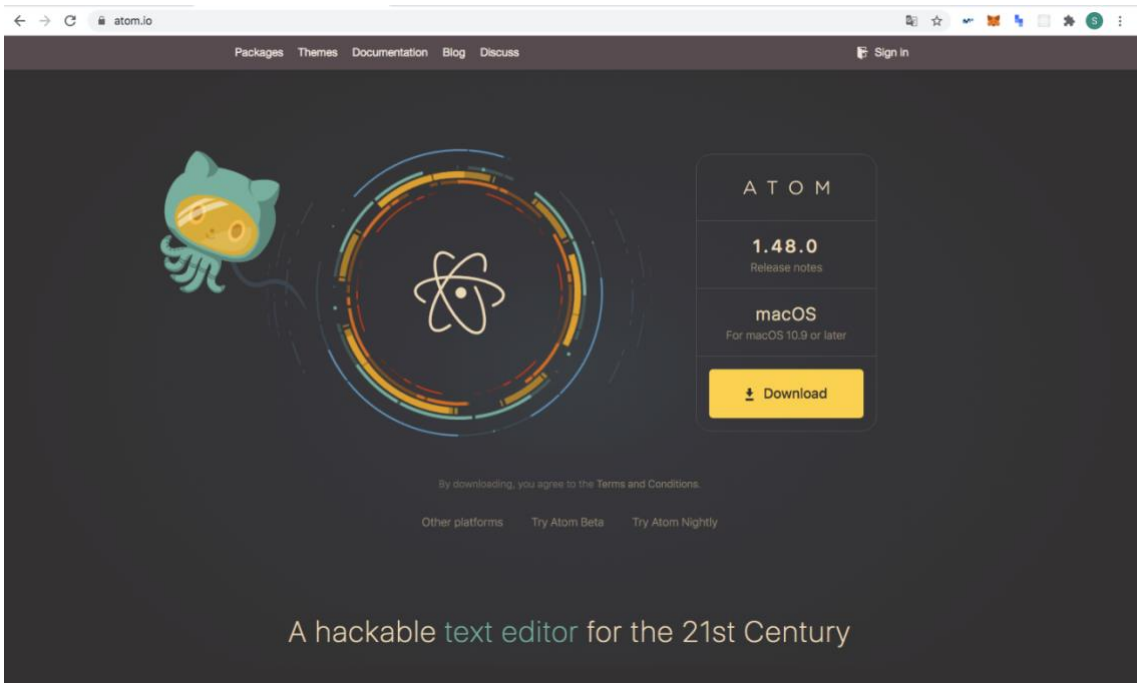
Las dos funcionalidades mencionadas anteriormente son evoluciones que pueden aportar una gran funcionalidad a la plataforma con el fin de potenciar aún más la inversión en infraestructura renovable y reducir la huella de carbono.

ANEXO A: GUÍA DE INSTALACIÓN

En el presente Anexo, se va a explicar el entorno que se debe instalar para poder desarrollar la aplicación correctamente. En primer lugar se explica la instalación de todos los programas y librerías necesarias y seguidamente se explica el código utilizado en el desarrollo de la plataforma.

A. Atom

El primer programa a instalar es el editor de código y la aplicación Atom es la que se ha utilizado en este proyecto. Para descargar Atom se debe acceder en primer lugar a su página web.[2]

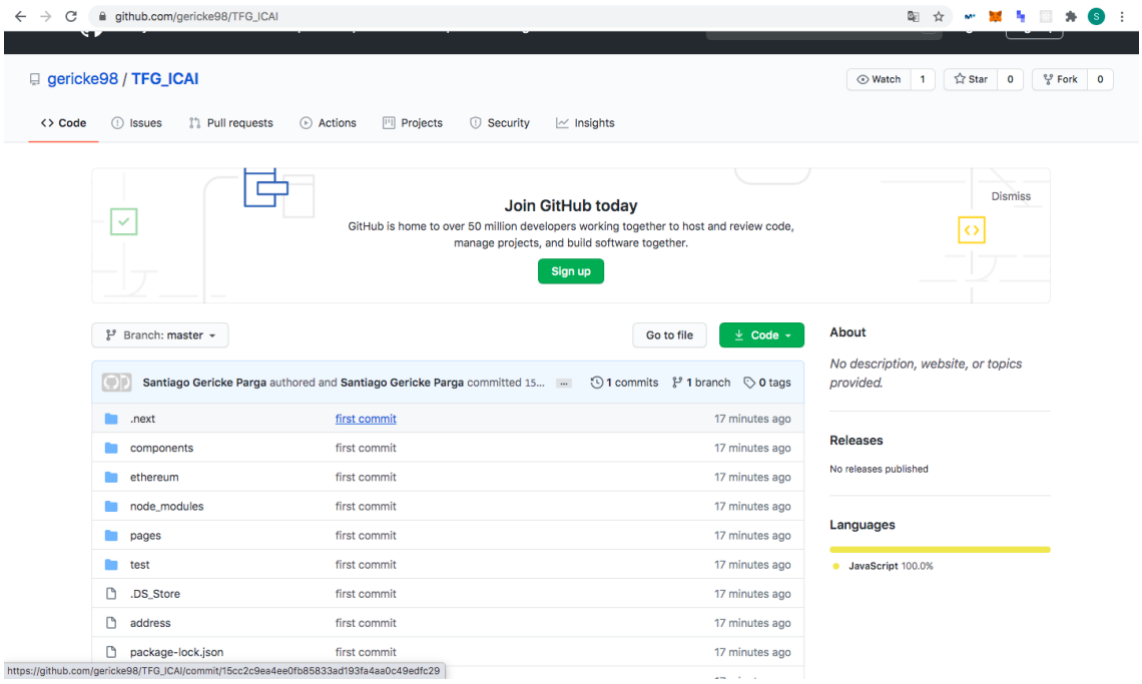


[2] Página oficial de Atom: https://atom.io/

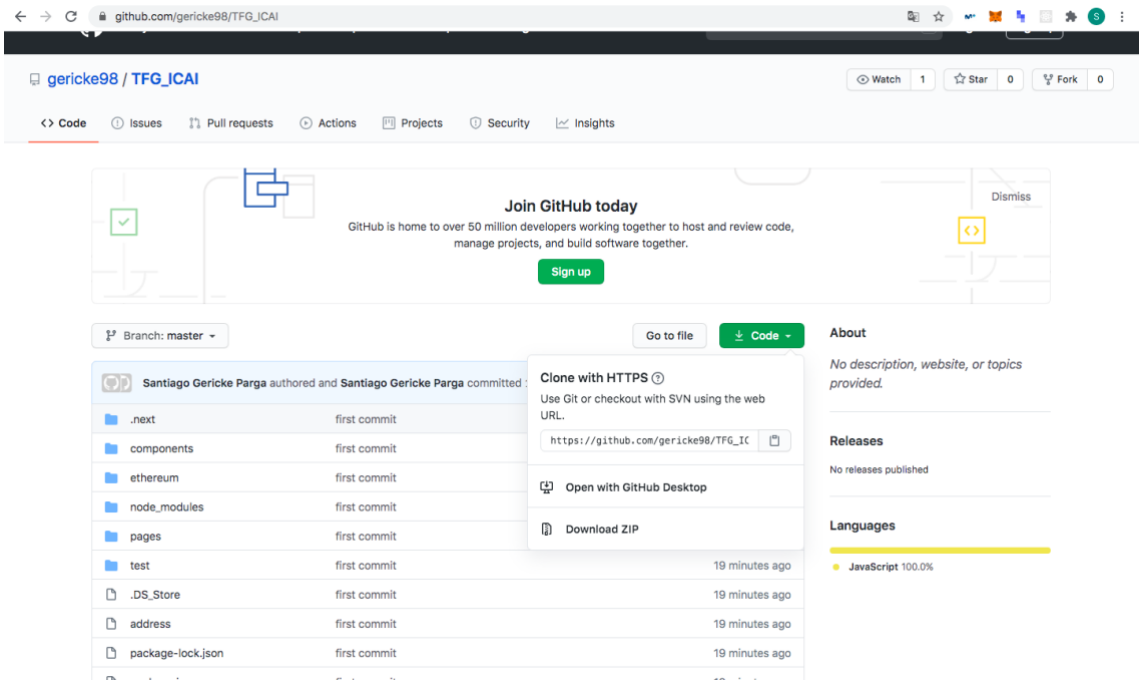
Una vez en ella, se debe pinchar en sistema operativo del ordenador, que en este caso es MacOs. Tras pinchar en el sistema operativo, se descarga un archivo comprimido que, al descomprimirlo, instala el programa en el ordenador.

B. Descarga código:

Tras instalar el editor de código, el siguiente paso es instalar el código necesario para que la plataforma funcióne correctamente. Para ello el primer paso es visitar la siguiente página web: https://github.com/gericke98/TFG_ICAI . Tras visitar dicha página web, aparece la siguiente página:



El siguiente paso consiste en pulsar el botón *Code* y pinchar en *Download Zip,* como se muestra en la siguiente imagen:



Tras descargar el zip, el último paso será descomprimir el archivo y guardar todo el código en una carpeta. En este caso, el código está guardado en la carpeta TFG_CODIGO dentro del escritorio.

C. Instalación npm:

Tras descargar el editor de código y el código, el siguiente paso es instalar las librerías necesarias para el desarrollo de la plataforma.

La primera de ellas es NPM que se instala introduciendo el siguiente código en la Terminal del ordenador:

sudo npm install

Tras introducir el código, el ordenador te solicita tu contraseña. Tras introducir la contraseña correctamente, se termina el proceso de instalación de NPM.

D. Instalación Nodejs:

Otra tecnología que es importante instalar es Nodejs que se instala introduciendo el siguiente código en la Terminal:

sudo nodejs install

Como en el caso anterior, se solicita la contraseña del ordenador y tras introducirla correctamente se termina el proceso de instalación.

E. Instalación librerías:

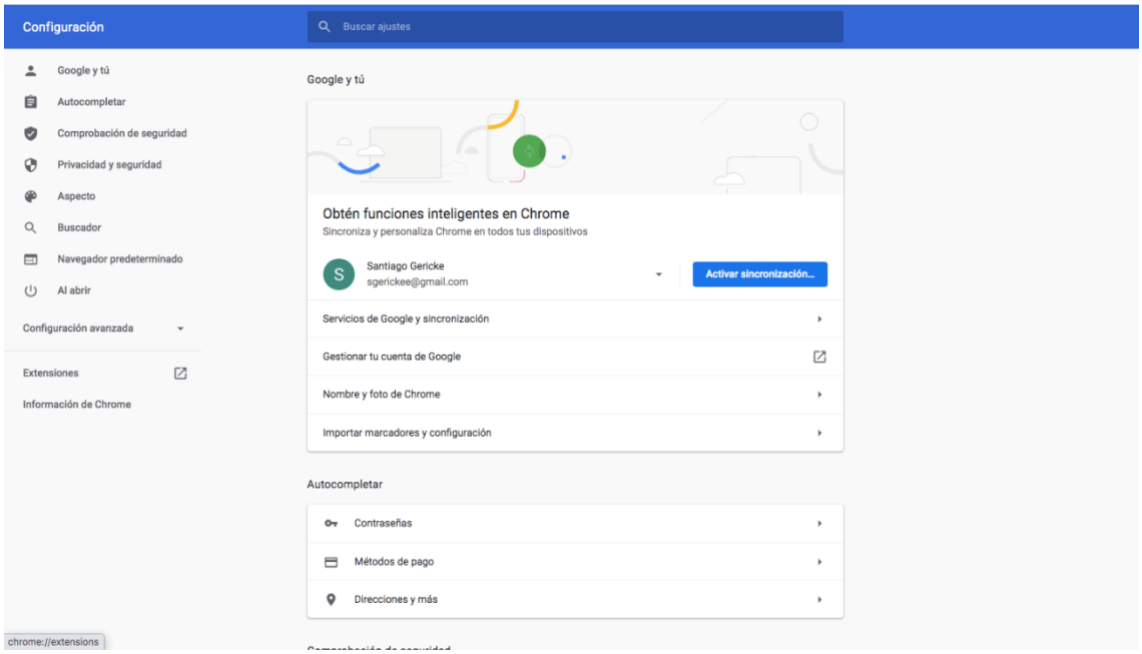
Una vez instalados npm y nodejs el siguiente paso es instalar las librerías restantes ejecutando en la Terminal del ordenador el siguiente código:

npm install --save truffle -g ganache-cli semantic-ui-react solc fs-extra web3@1.0.0-beta.26

Como en el caso anterior, se solicita la contraseña del ordenador y tras introducirla correctamente se termina el proceso de instalación.

F. Instalación MetaMask:

MetaMask es una extensión de Google Chrome que permite tener identidades en el mundo Blockchain. Para instalar dicha extensión, el usuario debe ir a Configuración en Google Chorme (tres puntos verticales de la derecha) y pinchar en Extensiones como se muestra en la siguiente imagen:



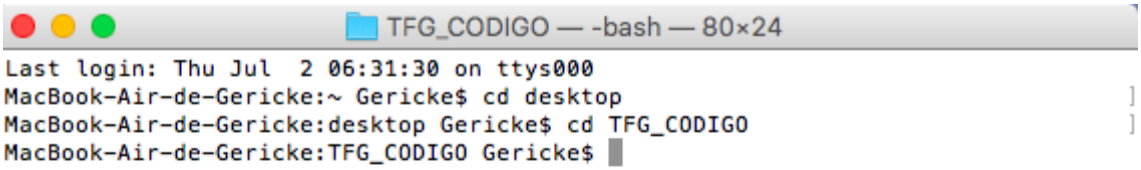
Una vez en la página de Extensiones, se debe buscar MetaMask y pulsar en instalar. El último paso es el de crearse una cuenta introduciendo datos personales.

ANEXO B: MANUAL DE USUARIO

En el presente anexo, se trata de explicar de la forma más detallada posible el funcionamiento de la plataforma.

1. Puesta en marcha

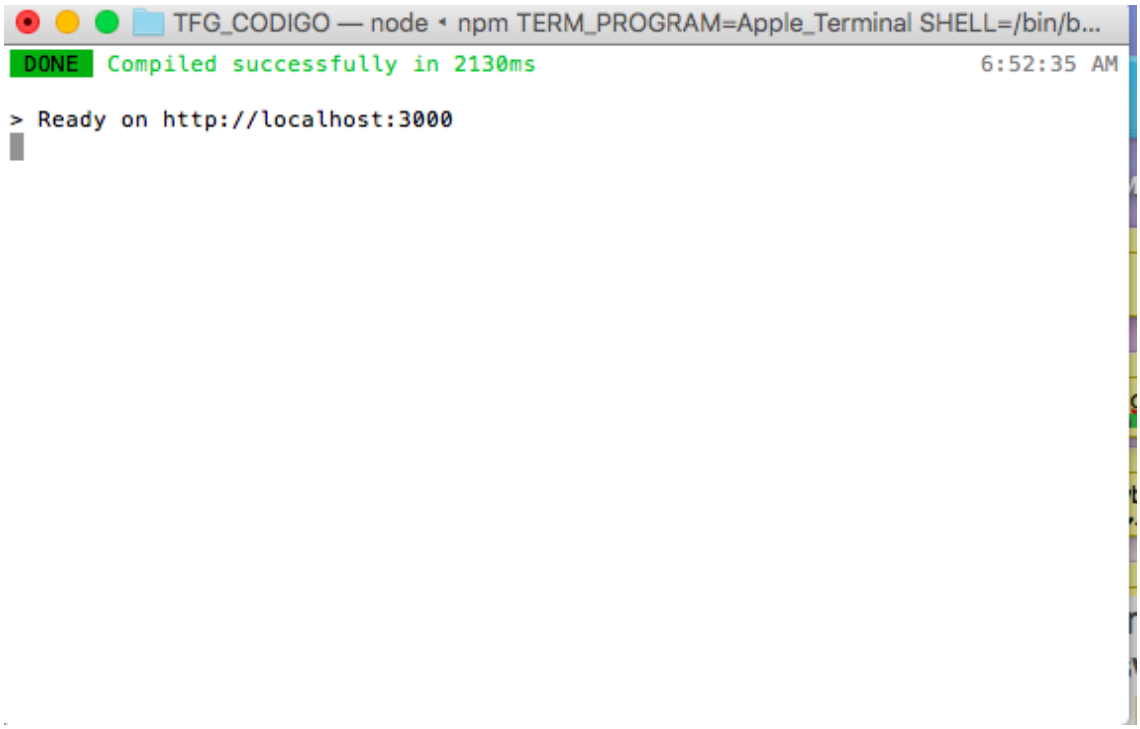
El primer paso consiste en ejecutar el código que se ha explicado anteriormente. Para ello, se abre terminal y se va a la carpeta en la que esté almacenado el código, en este caso, Escritorio/TFG_CODIGO:



Una vez direccionado, se introduce el siguiente código:

npm run dev

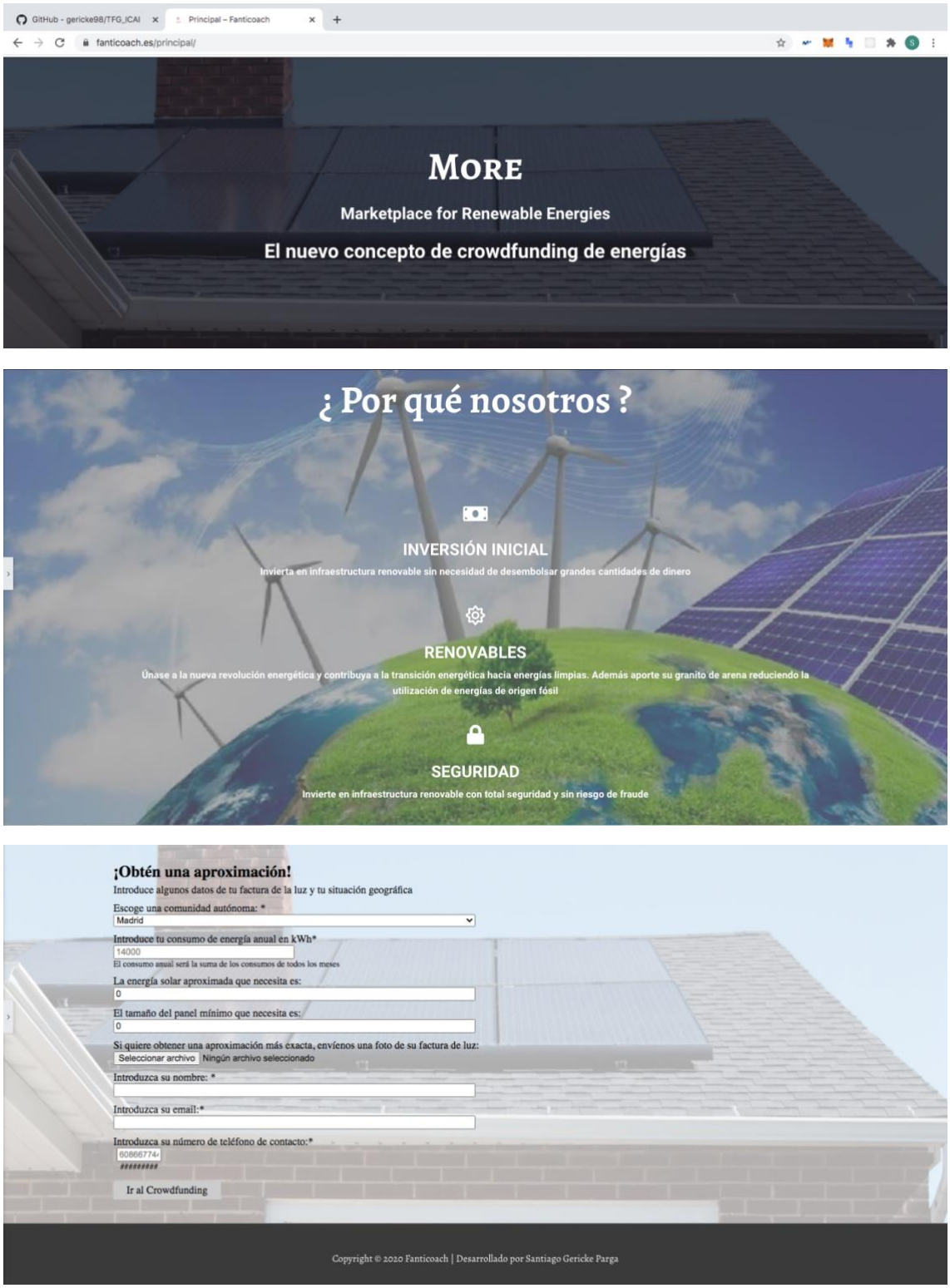
Tras ejecutar el código, se abre la siguiente página en Terminal:



2. Página de inicio

Tras haber realizado la puesta en marcha, el siguiente paso es visitar la página web: https://fanticoach.es/principal

Tras pinchar en el link, la página web que el usuario se encuentra es la siguiente:

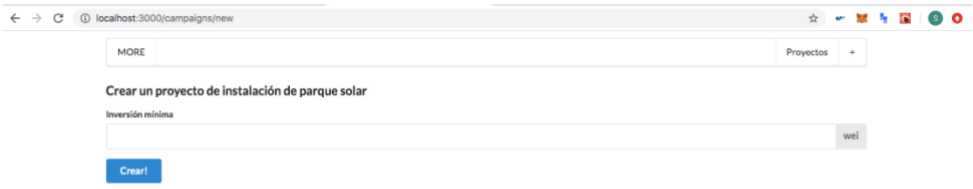


En la página inicial, por tanto, el usuario tiene la posibilidad de calcular el tamaño del panel necesario.

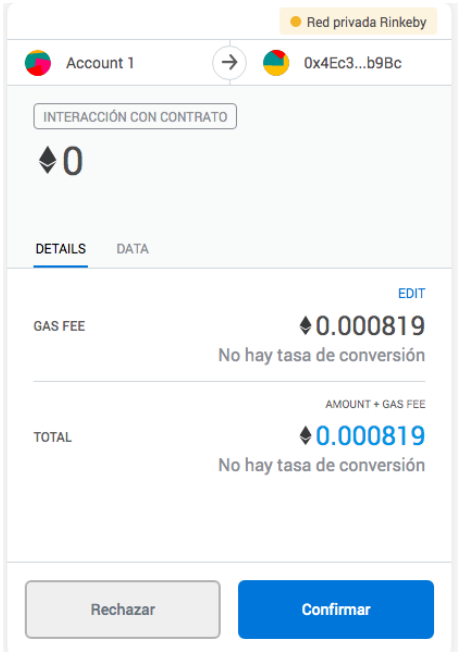
Si el usuario pincha en “Ir al Crowdfunding” se le auto dirige a la página principal del crowdfunding que es la siguiente:

Como se puede observar en la imagen, al entrar en la página principal se muestra una lista de todos los proyectos abiertos. En cada proyecto de la lista se muestra la address del contrato del proyecto y un botón para obtener más información del proyecto. Por otro lado, el usuario también tiene la opción de crear un nuevo proyecto de inversión.

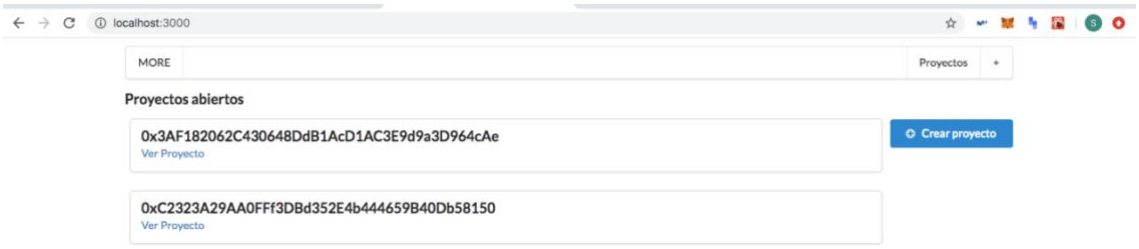
En caso de crear un nuevo proyecto, se le redirige a la siguiente página:



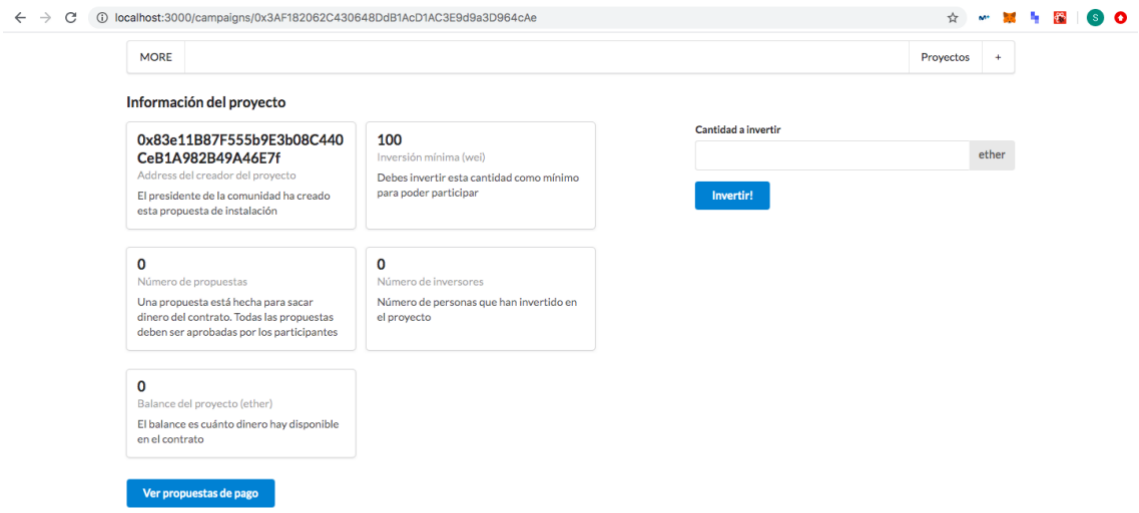
Como se puede observar, el único input que debe introducir el usuario es la inversión mínima que se debe realizar. Una vez introducido el input se crea una transacción como se muestra en la siguiente imagen:



Una vez creada, se añade el nuevo proyecto a la lista de todos los proyectos, como se muestra en la siguiente imagen:

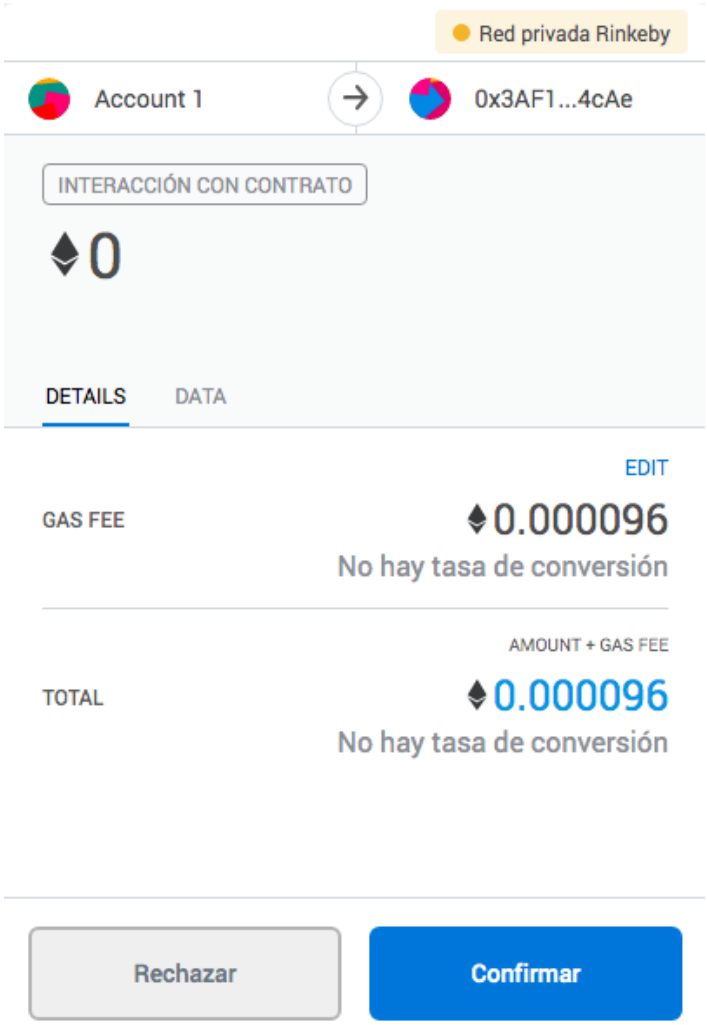


Por otro lado, en caso de que el usuario quiera obtener más información sobre un proyecto de inversión específico se le dirige a la siguiente página:

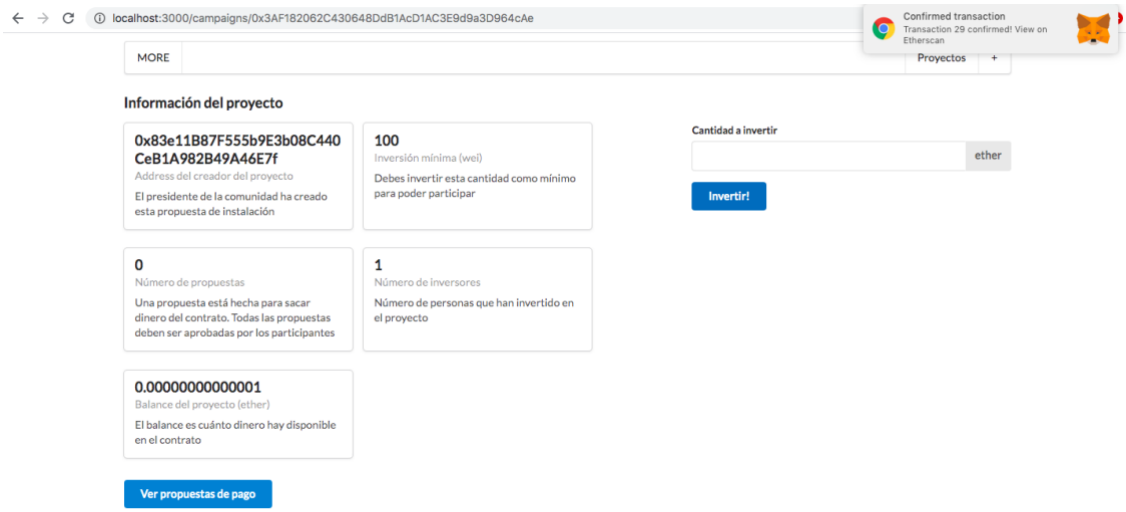


Como se puede observar en la imagen, la información que el usuario puede obtener es la address del manager del contrato, la inversión mínima, el número de propuestas de pago, el número de inversores y el balance del contrato. Además, el usuario tiene la posibilidad de invertir en el proyecto y de ver las propuestas de pago realizadas en dicho proyecto.

En caso de invertir, en primer lugar, el usuario debe confirmar la transacción como se muestra en la siguiente imagen:



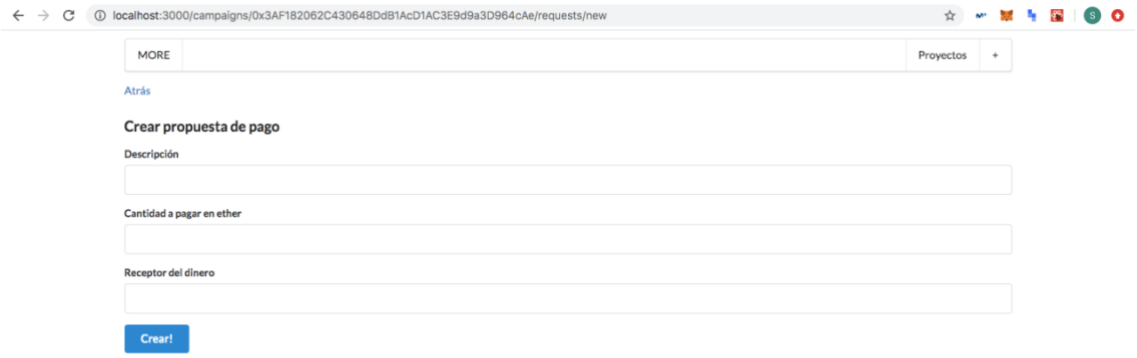
Una vez confirmada, se actualiza la información del proyecto como se muestra en la siguiente figura:



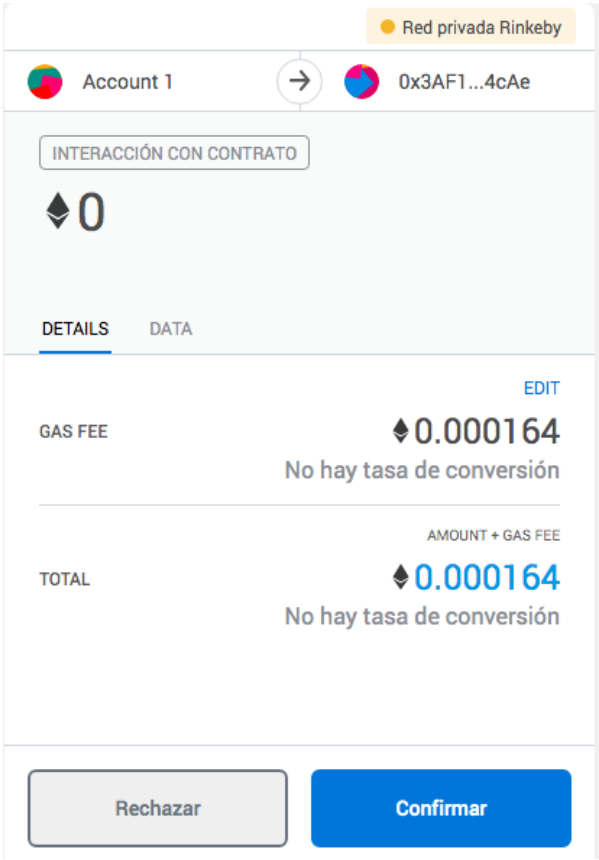
Como se puede observar, se ha actualizado el número de inversores y el balance del contrato.

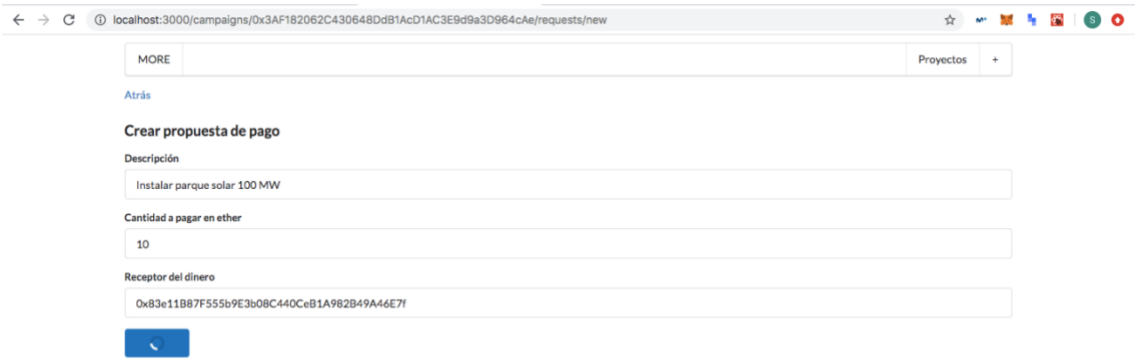
Por otro lado, el usuario tiene la posibilidad de ver las propuestas de pago realizadas en dicho proyecto. En caso de pinchar en ver propuestas de pago, se dirige al usuario a la siguiente página:

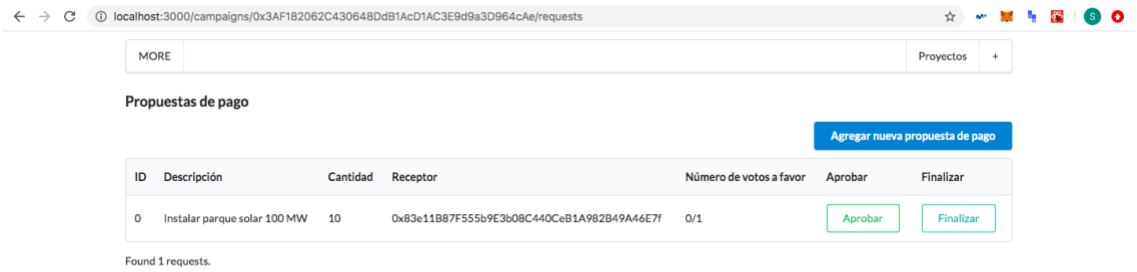
Como se puede observar, el usuario tiene la posibilidad de ver todas las propuestas de pago realizadas en una tabla. Además, también tiene la posibilidad de crear una nueva propuesta de pago. En caso de agregar una nueva propuesta de pago, el usuario debe rellenar el siguiente formulario:



En dicho formulario, el usuario debe introducir la descripción del gasto, la cantidad a pagar en ether y la address del receptor del dinero. Una vez introducidos dichos datos, se crea una nueva transacción y se añade la propuesta de pago a la lista como se muestra en las siguientes figuras:







Como se puede observar en la última imagen, la proposición se ha añadido a la lista. En ella, se incluye una descripción, la cantidad a invertir, la address del receptor del dinero, el número de votos a favor y la posibilidad de aprobar la medida o finalizarla.

ANEXO: CÓDIGO WEB

Una vez desplegado y compilado el contrato en la red pública de Ethereum cuyo código se ha mostrado anteriormente, ya se le pueden realizar llamadas a través de Web3. Seguidamente, se crean dos archivos de importación de librerías y declaración de instancias del contrato:

campaign.js

```
import web3 from './web3';
import Campaign from './build/Campaign.json';

export default address => {
  return new web3.eth.Contract(JSON.parse(Campaign.interface), address);
};

```

factory.js

```
import web3 from './web3';
import VotacionFactory from './build/CampaignFactory.json';

const instance = new web3.eth.Contract(
  JSON.parse(VotacionFactory.interface),
  '0x4Ec317d8d391cCB6dD3856B4615FC1A37927b9Bc'
);

export default instance;

```

Posteriormente, se va a explicar el código referente a las páginas:

Para la página principal, se va a utilizar el archivo index.js:

index.js

```
import React, { Component } from 'react';
import { Card, Button } from 'semantic-ui-react';
import factory from '../ethereum/factory';
import Layout from '../components/Layout';
import { Link } from '../routes';

class CampaignIndex extends Component {
  static async getInitialProps() {
    const campaigns = await factory.methods.getDeployedCampaigns().call();

    return { campaigns };
  }

  renderCampaigns() {
    const items = this.props.campaigns.map(address => {
      return {
        header: address,
        description: (
          <Link route={`/campaigns/${address}`}>
            <a>Ver Proyecto</a>
          </Link>
        ),
        fluid: true
      };
    });

    return <Card.Group items={items} />;
  }

```

```
29
30    render() {
31      return (
32        <Layout>
33          <div>
34            <h3>Proyectos abiertos</h3>
35
36            <Link route="/campaigns/new">
37              <a>
38                <Button
39                  floated="right"
40                  content="Crear proyecto"
41                  icon="add circle"
42                  primary
43                />
44              </a>
45            </Link>
46
47            {this.renderCampaigns()}
48          </div>
49        </Layout>
50      );
51    }
52  }
53
54  export default CampaignIndex;
55
```

Todo lo que se muestra en la página web, es lo que va debajo del looping de render. Por otro lado, se utiliza la librería de Semantic-Ui-React para el CSS de la página web. El código mostrado, es el utilizado para la página principal, en la que se mostrarán una lista de todas las opciones de inversión disponibles.

El código utilizado para el desarrollo de la página para crear un nuevo proyecto es el siguiente:

new.js

```
import React, { Component } from 'react';
import { Form, Button, Input, Message } from 'semantic-ui-react';
import Layout from '../../components/Layout';
import factory from '../../ethereum/factory';
import web3 from '../../ethereum/web3';
import { Router } from '../../routes';

class CampaignNew extends Component {
  state = {
    minimumContribution: '',
    errorMessage: '',
    loading: false
  };

  onSubmit = async event => {
    event.preventDefault();

    this.setState({ loading: true, errorMessage: '' });

    try {
      const accounts = await web3.eth.getAccounts();
      await factory.methods
        .createCampaign(this.state.minimumContribution)
        .send({
          from: accounts[0]
        });

      Router.pushRoute('/');
    } catch (err) {
      this.setState({ errorMessage: err.message });
    }

    this.setState({ loading: false });
  };
```

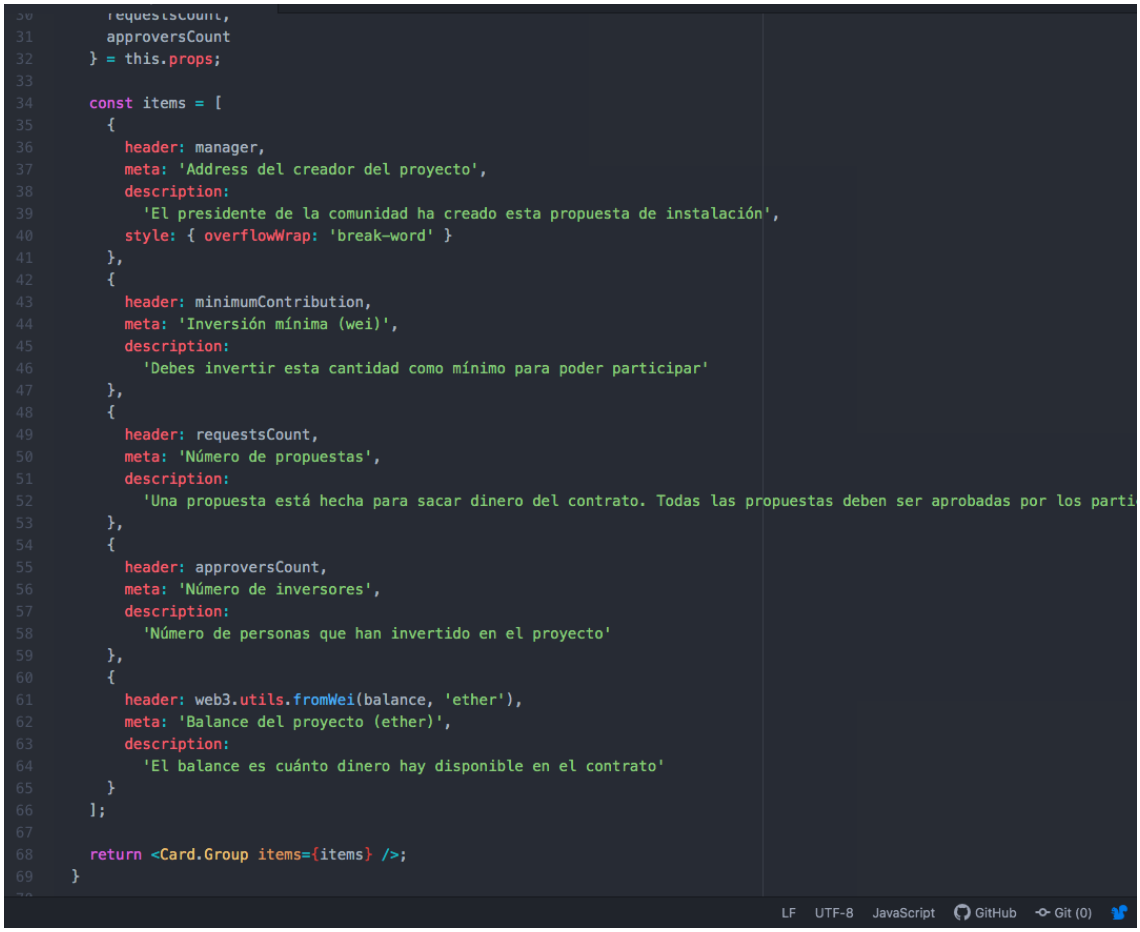
```
  render() {
    return (
      <Layout>
        <h3>Crear un proyecto de instalación de parque solar</h3>

        <Form onSubmit={this.onSubmit} error={!!this.state.errorMessage}>
          <Form.Field>
            <label>Inversión mínima</label>
            <Input
              label="wei"
              labelPosition="right"
              value={this.state.minimumContribution}
              onChange={event =>
                this.setState({ minimumContribution: event.target.value })}
            />
          </Form.Field>

          <Message error header="Oops!" content={this.state.errorMessage} />
          <Button loading={this.state.loading} primary>
            Crear!
          </Button>
        </Form>
      </Layout>
    );
  }
}

export default CampaignNew;
```

Por otro lado, el código utilizado para el desarrollo de la página para obtener más información sobre un proyecto es el siguiente:

show.js

```
import React, { Component } from 'react';
import { Card, Grid, Button } from 'semantic-ui-react';
import Layout from '../../components/Layout';
import Campaign from '../../ethereum/campaign';
import web3 from '../../ethereum/web3';
import ContributeForm from '../../components/ContributeForm';
import { Link } from '../../routes';

class CampaignShow extends Component {
  static async getInitialProps(props) {
    const campaign = Campaign(props.query.address);

    const summary = await campaign.methods.getSummary().call();

    return {
      address: props.query.address,
      minimumContribution: summary[0],
      balance: summary[1],
      requestsCount: summary[2],
      approversCount: summary[3],
      manager: summary[4]
    };
  }

  renderCards() {
    const {
      balance,
      manager,
      minimumContribution,
      requestsCount,
      approversCount
    } = this.props;

    const items = [
      {
        header: manager,
        meta: 'Address del creador del proyecto',
        description:
          'El presidente de la comunidad ha creado esta propuesta de instalación',
        style: { overflowWrap: 'break-word' }
```

LF



```
      requestsCount,
      approversCount
    } = this.props;

    const items = [
      {
        header: manager,
        meta: 'Address del creador del proyecto',
        description:
          'El presidente de la comunidad ha creado esta propuesta de instalación',
        style: { overflowWrap: 'break-word' }
      },
      {
        header: minimumContribution,
        meta: 'Inversión mínima (wei)',
        description:
          'Debes invertir esta cantidad como mínimo para poder participar'
      },
      {
        header: requestsCount,
        meta: 'Número de propuestas',
        description:
          'Una propuesta está hecha para sacar dinero del contrato. Todas las propuestas deben ser aprobadas por los parti
      },
      {
        header: approversCount,
        meta: 'Número de inversores',
        description:
          'Número de personas que han invertido en el proyecto'
      },
      {
        header: web3.utils.fromWei(balance, 'ether'),
        meta: 'Balance del proyecto (ether)',
        description:
          'El balance es cuánto dinero hay disponible en el contrato'
      }
    ];

    return <Card.Group items={items} />;
  }
```

```
  render() {
    return (
      <Layout>
        <h3>Información del proyecto</h3>
        <Grid>
          <Grid.Row>
            <Grid.Column width={10}>{this.renderCards()}</Grid.Column>

            <Grid.Column width={6}>
              <ContributeForm address={this.props.address} />
            </Grid.Column>
          </Grid.Row>

          <Grid.Row>
            <Grid.Column>
              <Link route={`/campaigns/${this.props.address}/requests`}>
                <a>
                  <Button primary>Ver propuestas de pago</Button>
                </a>
              </Link>
            </Grid.Column>
          </Grid.Row>
        </Grid>
      </Layout>
    );
  }
}

export default CampaignShow;
```

Por otro lado, el código utilizado para el desarrollo de la página que tiene la lista de propuesta de pago de un crowdfunding específico es el siguiente:

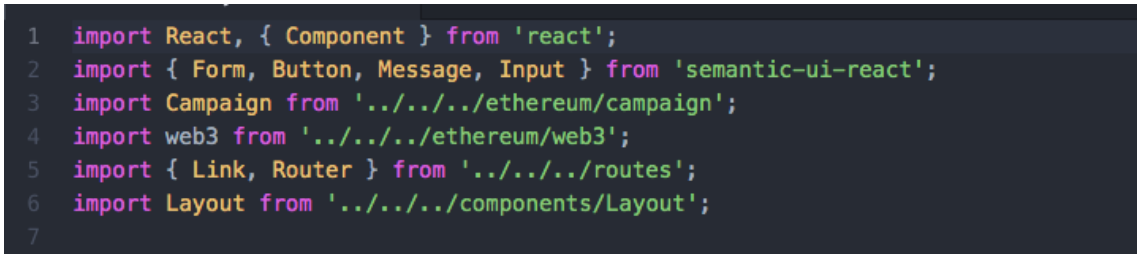
index.js

```
import React, { Component } from 'react';
import { Button, Table } from 'semantic-ui-react';
import { Link } from '../../../routes';
import Layout from '../../../components/Layout';
import Campaign from '../../../ethereum/campaign';
import RequestRow from '../../../components/RequestRow';

class RequestIndex extends Component {
  static async getInitialProps(props) {
    const { address } = props.query;
    const campaign = Campaign(address);
    const requestCount = await campaign.methods.getRequestsCount().call();
    const approversCount = await campaign.methods.approversCount().call();

    const requests = await Promise.all(
      Array(parseInt(requestCount))
        .fill()
        .map((element, index) => {
          return campaign.methods.requests(index).call();
        })
    );

    return { address, requests, requestCount, approversCount };
  }

```

```
  render() {
    const { Header, Row, HeaderCell, Body } = Table;

    return (
      <Layout>
        <h3>Propuestas de pago</h3>
        <Link route={`/campaigns/${this.props.address}/requests/new`}>
          <a>
            <Button primary floated="right" style={{ marginBottom: 10 }}>
              Agregar nueva propuesta de pago
            </Button>
          </a>
        </Link>
        <Table>
          <Header>
            <Row>
              <HeaderCell>ID</HeaderCell>
              <HeaderCell>Descripción</HeaderCell>
              <HeaderCell>Cantidad</HeaderCell>
              <HeaderCell>Receptor</HeaderCell>
              <HeaderCell>Número de votos a favor</HeaderCell>
              <HeaderCell>Aprobar</HeaderCell>
              <HeaderCell>Finalizar</HeaderCell>
            </Row>
          </Header>
          <Body>{this.renderRows()}</Body>
        </Table>
        <div>Found {this.props.requestCount} requests.</div>
      </Layout>
    );
  }
}

export default RequestIndex;

```

```
renderRows() {
  return this.props.requests.map((request, index) => {
    return (
      <RequestRow
        key={index}
        id={index}
        request={request}
        address={this.props.address}
        approversCount={this.props.approversCount}
      />
    );
  });
}
```

Como se ha mencionado anteriormente, el código de index.js es el que se utiliza para mostrar la lista con todas las proposiciones de pago. Por otro lado, el código que se utiliza para crear una nueva proposición de pago es el siguiente:



```
1  import React, { Component } from 'react';
2  import { Form, Button, Message, Input } from 'semantic-ui-react';
3  import Campaign from '../../../ethereum/campaign';
4  import web3 from '../../../ethereum/web3';
5  import { Link, Router } from '../../../routes';
6  import Layout from '../../../components/Layout';
7
```

```
45    render() {
46      return (
47        <Layout>
48          <Link route={`/campaigns/${this.props.address}/requests`}>
49            <a>Atrás</a>
50          </Link>
51          <h3>Crear propuesta de pago</h3>
52          <Form onSubmit={this.onSubmit} error={!!this.state.errorMessage}>
53            <Form.Field>
54              <label>Descripción</label>
55              <Input
56                value={this.state.description}
57                onChange={event =>
58                  this.setState({ description: event.target.value })}
59              />
60            </Form.Field>
61
62            <Form.Field>
63              <label>Cantidad a pagar en ether</label>
64              <Input
65                value={this.state.value}
66                onChange={event => this.setState({ value: event.target.value })}
67              />
68            </Form.Field>
69
70            <Form.Field>
71              <label>Receptor del dinero</label>
72              <Input
73                value={this.state.recipient}
74                onChange={event =>
75                  this.setState({ recipient: event.target.value })}
76              />
77            </Form.Field>
78
79            <Message error header="Oops!" content={this.state.errorMessage} />
80            <Button primary loading={this.state.loading}>
81              Crear!
82            </Button>
83          </Form>
84        </Layout>
```

```
            <Message error header="Oops!" content={this.state.errorMessage} />
            <Button primary loading={this.state.loading}>
              Crear!
            </Button>
          </Form>
        </Layout>
      );
    }
  }

export default RequestNew;

```