FundETH
Requirement Analysis Document

Introduzione

Contesto e Dominio

Il progetto si inserisce nel contesto della **Finanza Decentralizzata (DeFi)** e del crowdfunding. Attualmente, la raccolta fondi per cause benefiche, startup o progetti personali è dominata da piattaforme centralizzate che agiscono come intermediari fiduciari. Questo progetto mira a sviluppare una piattaforma decentralizzata (DApp) che sfrutta la tecnologia blockchain (specificamente Ethereum) per permettere a chiunque di creare campagne di raccolta fondi e ricevere donazioni in criptovaluta. L'adozione della blockchain garantisce che le transazioni siano pubbliche, immutabili e resistenti alla censura, eliminando la necessità di un'autorità centrale per la gestione dei fondi e aumentando la fiducia tra creatori e donatori.

Obiettivo del Progetto

L'obiettivo principale è la creazione di una piattaforma di crowdfunding trasparente e sicura che connetta direttamente i creatori di progetti con i sostenitori. Gli obiettivi specifici includono:

- **Decentralizzazione:** Eliminare gli intermediari finanziari tradizionali per ridurre le commissioni e i tempi di attesa per l'accesso ai fondi raccolti.
- **Trasparenza:** Registrare ogni donazione su un registro pubblico (Blockchain) in modo che i donatori possano verificare esattamente l'importo raccolto e la destinazione dei fondi.
- **Accessibilità Globale:** Permettere a chiunque, ovunque nel mondo, di avviare una campagna o donare senza restrizioni geografiche o bancarie, utilizzando un wallet digitale (es. MetaMask).
- **Automazione e Fiducia:** Utilizzare **Smart Contracts** per gestire la logica di donazione, assicurando che i fondi vadano direttamente al proprietario della campagna una volta inviati.

Sistema corrente e Sistema proposto

Sistema corrente Attualmente, la raccolta fondi avviene tramite piattaforme centralizzate (es. GoFundMe, Kickstarter).Questi sistemi trattengono una percentuale significativa delle donazioni come commissione di servizio. I fondi sono custoditi dall'intermediario e il rilascio può richiedere giorni o settimane. Esiste il rischio che la piattaforma blocchi una campagna o congeli i fondi in base a politiche interne arbitrarie. I donatori devono fidarsi ciecamente che la piattaforma gestisca correttamente il denaro prima che arrivi al destinatario.

Sistema proposto Il sistema proposto è una DApp che interagisce direttamente con la Blockchain di Ethereum.

- **Eliminazione Intermediari:** Non esiste un ente centrale che detiene i fondi; il denaro passa direttamente dal wallet del donatore allo Smart Contract o al wallet del creatore della campagna.

- **Costi Ridotti:** Le uniche spese sono le "gas fees" della rete blockchain, eliminando le alte commissioni delle piattaforme tradizionali.
- **Immutabilità:** Una volta che una campagna è creata e le donazioni sono inviate, nessuno può alterare i dati o sequestrare i fondi, garantendo la sovranità finanziaria degli utenti.
- **Verificabilità:** Chiunque può auditare il codice dello Smart Contract e vedere lo storico delle transazioni.

Descrizione del Sistema

Requisiti Funzionali

FR1 – Creazione della campagna: La funzionalità primaria è la **Creazione della Campagna**. Il sistema deve permettere a qualsiasi utente autenticato tramite wallet di inizializzare una nuova raccolta fondi. Questo processo richiede l'inserimento di metadati essenziali quali il titolo, una descrizione dettagliata del progetto, l'importo target da raggiungere e la data di scadenza. Per garantire efficienza e ridurre i costi di transazione sulla blockchain, i dati voluminosi come le immagini descrittive non verranno salvati direttamente on-chain, ma su un sistema di file system decentralizzato (IPFS), memorizzando nello Smart Contract solamente il riferimento (hash) a tali dati.

FR2 – Donazione: Una volta attiva, la piattaforma deve garantire la gestione delle **Donazioni**. Gli utenti devono poter inviare fondi in Ether a una specifica campagna. Il sistema deve registrare l'indirizzo del donatore e l'importo versato, aggiornando in tempo reale il totale raccolto visibile pubblicamente. È fondamentale che il sistema accetti donazioni solo finché la campagna è attiva e non scaduta.

FR3 – Prelievo dei fondi: Nel caso di successo, ovvero se il target viene raggiunto o superato entro la scadenza, deve essere attiva la funzione di **Prelievo dei Fondi**, accessibile esclusivamente dal creatore della campagna, che trasferisce il saldo accumulato dallo Smart Contract al wallet del proprietario.

FR4 – Rimborso: Nel caso in cui la scadenza venga raggiunta senza aver ottenuto l'importo richiesto, il sistema deve abilitare la funzionalità di **Rimborso**. In questo scenario, ogni donatore deve avere la possibilità di interagire con il contratto per recuperare esattamente l'importo versato, garantendo che nessun fondo rimanga bloccato indefinitamente.

Requisiti Non Funzionali

NFR1 - Sicurezza e Affidabilità: Il requisito cardine è la **Sicurezza e Affidabilità**. Poiché il sistema gestisce transazioni finanziarie reali, il codice dello Smart Contract deve essere esente da vulnerabilità critiche che potrebbero permettere il furto o il blocco dei fondi. L'autenticazione deve avvenire esclusivamente tramite firma crittografica (es. MetaMask), eliminando la gestione di password centralizzate e riducendo il rischio di furto di credenziali.

NFR2 – Trasparenza: Parallelamente, il sistema deve garantire la **Trasparenza**. Ogni transazione, dalla creazione della campagna alla singola donazione, deve essere registrata

sulla blockchain pubblica di Ethereum. Questo permette a qualsiasi stakeholder di verificare indipendentemente la veridicità dei dati mostrati dall'interfaccia web, assicurando che non ci siano discrepanze tra quanto dichiarato e quanto effettivamente raccolto.

NFR3 – Usabilità: Un altro aspetto critico è l'**Usabilità**. Nonostante la complessità tecnologica sottostante, l'interfaccia utente (Frontend) deve astrarre le difficoltà tecniche della blockchain, presentando un'esperienza d'uso pulita e intuitiva, simile a quella delle applicazioni web tradizionali. L'utente deve essere guidato chiaramente nel processo di connessione del wallet e nella conferma delle transazioni.

NFR4 – Efficienza dei costi: Il sistema deve considerare l'**Efficienza dei Costi**. Dato che ogni operazione di scrittura sulla blockchain comporta un costo (gas fee), l'architettura deve essere ottimizzata per minimizzare le operazioni on-chain. L'uso di IPFS per lo storage dei contenuti multimediali risponde proprio a questa esigenza, mantenendo la piattaforma economicamente accessibile per i creatori.

Architettura del Sistema

Panoramica

Il sistema FundETH è progettato secondo un'architettura decentralizzata (DApp) basata sulla blockchain di Ethereum. A differenza delle architetture web tradizionali client-server, questa piattaforma adotta un approccio Web3 che elimina la necessità di un database centralizzato per la gestione delle transazioni finanziarie e della logica di business. L'architettura prevede una suddivisione logica in tre livelli principali che cooperano per garantire la sicurezza e la trasparenza richieste.

Il **Livello di Presentazione** costituisce l'interfaccia utente web, accessibile tramite browser, che funge da punto di accesso per creatori e donatori. Questo livello non gestisce direttamente i fondi, ma facilita l'interazione con la blockchain attraverso l'integrazione con wallet digitali.

Il **Livello Applicativo** è interamente gestito dagli Smart Contracts distribuiti sulla blockchain di Ethereum. Questo componente rappresenta il "motore" del sistema, responsabile dell'esecuzione automatica delle regole definite nei requisiti funzionali, come la convalida delle scadenze e la gestione dei prelievi o rimborsi.

Infine, il **Livello Dati** adotta un approccio ibrido per massimizzare l'efficienza. Mentre le transazioni e i saldi sono memorizzati in modo immutabile sulla blockchain, i contenuti voluminosi come le immagini delle campagne sono archiviati su IPFS (InterPlanetary File System), un sistema di file system distribuito, riducendo così i costi di gas on-chain.

Componenti Principali

Frontend Web Application La piattaforma web è sviluppata utilizzando moderne tecnologie React (Vite) per fornire un'interfaccia reattiva e user-friendly, rispondendo al requisito di usabilità. Essa permette agli utenti di visualizzare le campagne attive, monitorare

il progresso delle donazioni in tempo reale e interagire con i moduli di creazione. Il frontend non memorizza chiavi private; delega invece tutte le operazioni di firma e autorizzazione al wallet dell'utente.

Smart Contract su Ethereum Scritto in linguaggio Solidity, lo Smart Contract è il cuore trustless del sistema. Esso incapsula la logica descritta nei requisiti funzionali: gestisce la creazione delle campagne registrando gli hash dei metadati, accetta donazioni in Ether, e blocca i fondi in un contratto di escrow. Lo Smart Contract applica rigidamente le condizioni per il prelievo da parte del creatore o il rimborso ai donatori, garantendo che nessuna terza parte possa manipolare i flussi finanziari.

Storage Decentralizzato (IPFS) Per rispondere all'esigenza di efficienza dei costi, il sistema utilizza IPFS per l'hosting dei media. Quando un utente crea una campagna, l'immagine descrittiva viene caricata su questo network distribuito. IPFS restituisce un identificativo univoco (CID o Hash) che viene successivamente memorizzato nello Smart Contract. Questo garantisce che i dati siano accessibili in modo decentralizzato senza appesantire la blockchain.

Wallet Ethereum Strumenti come MetaMask fungono da ponte tra l'utente e la blockchain. Il wallet è essenziale per l'autenticazione crittografica e per la firma delle transazioni, come l'invio di una donazione o la richiesta di prelievo. Senza un wallet connesso, l'utente può operare solo in modalità "sola lettura".

Flusso di Dati e Interazioni

Il flusso operativo del sistema segue un percorso lineare che attraversa i vari componenti architetturali per soddisfare i requisiti definiti.

Nel processo di **Creazione della Campagna**, l'utente compila i dettagli nell'interfaccia web. Prima di interagire con la blockchain, il frontend carica l'immagine su IPFS ricevendo in cambio un hash. Successivamente, il wallet dell'utente richiede la firma di una transazione che invia questo hash, insieme al titolo, alla descrizione, al target e alla scadenza, allo Smart Contract. Una volta minata la transazione, la campagna è ufficialmente attiva e immutabile.

Per quanto riguarda la **Donazione**, il flusso inizia quando un donatore seleziona una campagna e specifica l'importo. Il frontend invoca la funzione di pagamento dello Smart Contract tramite il wallet. I fondi vengono trasferiti direttamente dall'indirizzo del donatore al saldo interno dello Smart Contract, che aggiorna contestualmente il contatore dei fondi raccolti e registra il donatore nel registro pubblico.

Infine, le operazioni di **Prelievo o Rimborso** sono attivate dall'interazione dell'utente con il frontend, che verifica preliminarmente lo stato della campagna (scaduta, target raggiunto o fallito). Se le condizioni sono soddisfatte, viene inviata una richiesta allo Smart Contract che, dopo aver verificato internamente la logica temporale e finanziaria, sblocca i fondi inviandoli al wallet del richiedente.

Diagrammi Architetturali

Di seguito viene descritta la struttura logica dei diagrammi che rappresentano il sistema.

Diagramma Generale del Sistema Il flusso parte dall'Utente che interagisce con il Frontend Web. Il Frontend comunica in due direzioni: invia e riceve dati pesanti (immagini) dal nodo IPFS e invia transazioni e richieste di lettura allo Smart Contract sulla rete Ethereum tramite il Provider Web3 (Wallet). Lo Smart Contract detiene lo stato e i fondi.

Diagramma di Creazione e Storage Creatore -> Caricamento Immagine -> IPFS -> Restituzione Hash (CID) -> Frontend -> Transazione con Metadati + CID -> Smart Contract -> Evento Campagna Creata.

Modellazione del Sistema

Scenari di Utilizzo

Per comprendere appieno come **FundETH** si inserisce nelle dinamiche quotidiane dei suoi utenti, è utile descrivere alcuni scenari narrativi che illustrano l'interazione pratica con la piattaforma. Questi scenari coprono le funzionalità critiche identificate nei requisiti.

Scenario 1: Lancio di un progetto innovativo Immaginiamo Alice, un'imprenditrice che desidera finanziare lo sviluppo di un nuovo dispositivo ecologico. Alice accede a FundETH e connette il suo wallet MetaMask. Naviga nella sezione di creazione e inserisce i dettagli del suo progetto, caricando un'immagine accattivante che il sistema salva automaticamente su IPFS per garantire efficienza. Imposta un target di 5 ETH e una scadenza di 30 giorni. Dopo aver confermato la transazione sul suo wallet, la campagna di Alice viene registrata sulla blockchain ed è immediatamente visibile nella vetrina pubblica della DApp, pronta a ricevere contributi da tutto il mondo.

Scenario 2: Sostegno e Donazione Sicura Bob, un appassionato di tecnologia, visita FundETH alla ricerca di nuovi progetti. Viene colpito dalla campagna di Alice e decide di contribuire. Senza dover creare un account tradizionale, connette semplicemente il suo wallet e decide di donare 0.5 ETH. Il sistema gli mostra chiaramente le commissioni di rete e, una volta confermato, i fondi vengono trasferiti direttamente nello Smart Contract di garanzia. Bob vede immediatamente aggiornarsi la barra di progresso della campagna e il suo indirizzo apparire nella lista pubblica dei sostenitori, avendo la certezza matematica che il suo contributo è stato registrato.

Scenario 3: Il meccanismo di garanzia (Rimborso) Supponiamo purtroppo che la campagna di Alice non riesca a raggiungere l'obiettivo di 5 ETH entro i 30 giorni previsti. In un sistema tradizionale, Bob dovrebbe preoccuparsi di contattare il supporto o sperare nella buona fede di Alice. Su FundETH, invece, Bob visita nuovamente la pagina della campagna scaduta. Il sistema riconosce automaticamente che l'obiettivo non è stato raggiunto e che Bob è un donatore attivo. L'interfaccia gli propone un pulsante per il rimborso. Con un semplice click e una conferma di transazione, lo Smart Contract restituisce l'importo donato (0.5 ETH) direttamente al wallet di Bob, chiudendo il cerchio senza interventi esterni.

Use Cases

A livello più tecnico, le interazioni descritte sopra vengono formalizzate nei seguenti Casi d'Uso, che descrivono puntualmente come il sistema reagisce alle azioni degli attori.

UC1 - Creazione della Campagna L'attore principale è il **Creatore**. L'obiettivo è pubblicare una nuova richiesta di fondi sulla blockchain. Il processo inizia quando il Creatore, dopo essersi autenticato, compila il modulo di lancio. Il sistema ha il compito cruciale di caricare i media su IPFS ottenendo un hash identificativo (CID). Successivamente, il Creatore firma una transazione che invia questo hash, insieme ai dati finanziari (target) e temporali (scadenza), allo Smart Contract.

- **Pre-condizioni:** L'utente deve possedere un wallet Ethereum con saldo sufficiente per le gas fee.
- **Post-condizioni:** La campagna è immutabile, attiva e possiede un ID univoco sulla blockchain.

UC2 - Donazione L'attore è il **Donatore**. L'obiettivo è trasferire valore finanziario a sostegno di una causa. Il Donatore seleziona una campagna attiva e specifica l'importo in ETH. Il sistema verifica che la campagna non sia scaduta e che l'importo sia valido. Alla conferma, lo Smart Contract accetta i fondi, aggiorna il saldo totale della campagna e registra l'indirizzo del donatore in un registro interno per eventuali rimborsi futuri o diritti di prelievo.

- **Pre-condizioni:** La campagna deve essere in corso e non scaduta.
- **Post-condizioni:** Il saldo della campagna aumenta e il wallet del donatore viene addebitato.

UC3 - Prelievo dei Fondi L'attore è il **Creatore**. Questo caso d'uso rappresenta il successo della raccolta. Se la campagna ha raggiunto o superato il target entro la scadenza, il Creatore può invocare questa funzione. Lo Smart Contract effettua un controllo rigoroso: verifica l'identità del richiedente (deve essere il proprietario), controlla che il target sia stato raggiunto e che i fondi non siano già stati prelevati. Se tutte le condizioni sono vere, trasferisce l'intero saldo accumulato al wallet del Creatore.

- **Pre-condizioni:** Target raggiunto e richiedente corrispondente al proprietario della campagna.
- **Post-condizioni:** Il saldo della campagna nello Smart Contract si azzera (o viene marcato come riscosso) e i fondi sono disponibili al Creatore.

UC4 - Richiesta di Rimborso L'attore è il **Donatore**. Questo caso d'uso gestisce lo scenario di fallimento della raccolta, proteggendo gli investitori. Se una campagna scade senza raggiungere il target, il Donatore può interagire con il contratto. Il sistema verifica che l'utente sia effettivamente un donatore registrato e che abbia un saldo positivo versato in quella specifica campagna. In caso affermativo, lo Smart Contract restituisce l'esatto importo versato al wallet del richiedente e azzera il suo saldo nel registro della campagna per evitare doppi rimborsi.

- **Pre-condizioni:** Campagna scaduta, target non raggiunto e utente presente nel registro donatori con saldo positivo.
- **Post-condizioni:** L'utente riottiene i propri fondi e il suo credito verso la campagna viene annullato.

Piano di Sviluppo

Strumenti di Sviluppo

Per la realizzazione di FundETH è stato selezionato uno stack tecnologico moderno e specifico per lo sviluppo Web3, scelto per garantire robustezza e facilità di integrazione.

Per la logica decentralizzata, il linguaggio di riferimento è **Solidity**. È lo standard industriale per la scrittura di Smart Contracts sulla Ethereum Virtual Machine (EVM) e permette di definire in modo preciso e immutabile le regole di gestione dei fondi della piattaforma.

L'ambiente di sviluppo e testing è gestito tramite **Hardhat**. Questo strumento è essenziale per il ciclo di vita del progetto: consente non solo la compilazione dei contratti, ma anche la creazione di una blockchain locale per i test rapidi. Hardhat è particolarmente prezioso per FundETH grazie alla sua capacità di simulare il passaggio del tempo (Time Travel), permettendo di testare funzionalità legate alle scadenze delle campagne, come i rimborsi, senza dover attendere giorni reali.

Per il frontend, la scelta è ricaduta su **React (con Vite)** anziché su semplice HTML/JavaScript. Questa libreria permette di creare un'interfaccia utente reattiva e modulare, gestendo in modo efficiente lo stato dell'applicazione, come l'aggiornamento in tempo reale della barra di progresso delle donazioni.

L'autenticazione e l'interazione con la blockchain sono affidate a **MetaMask**. Questo wallet funge da ponte tra il browser dell'utente e la rete Ethereum, gestendo la firma sicura delle transazioni per donazioni e prelievi senza che la piattaforma debba mai accedere alle chiavi private dell'utente.

Infine, per la gestione dei contenuti multimediali, si utilizza **IPFS** (InterPlanetary File System), spesso interfacciato tramite servizi come Pinata. Questo garantisce che le immagini delle campagne siano memorizzate in modo decentralizzato e sicuro, mantenendo leggeri gli Smart Contract e riducendo i costi di gas per gli utenti.