

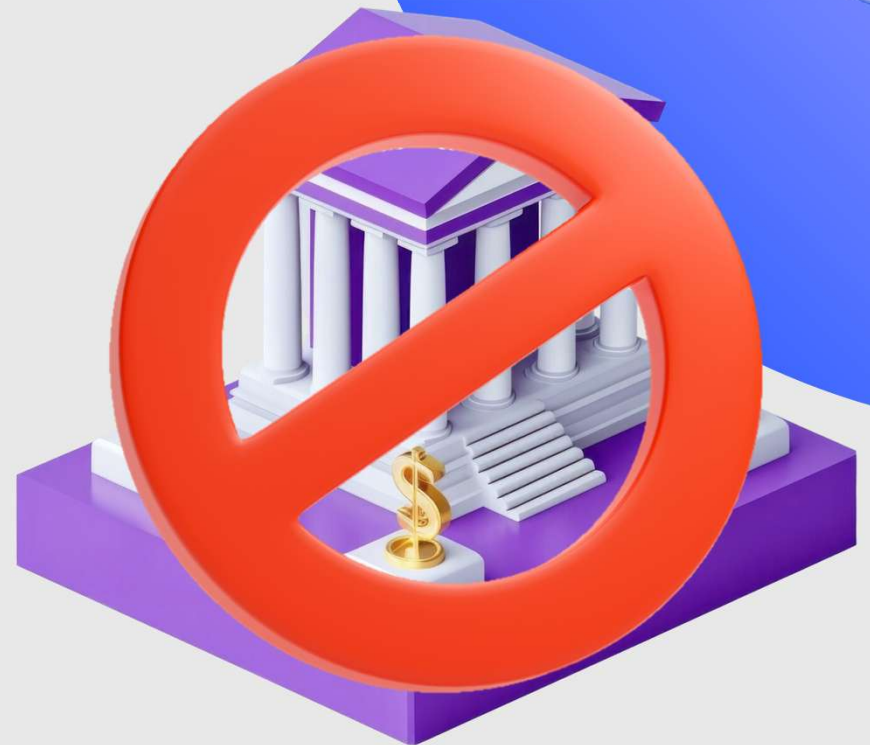


# FundETH

Piattaforma di Crowdfunding Decentralizzato su  
Blockchain

# Il Problema del Crowdfunding Tradizionale

- **Centralizzazione:** Dipendenza da intermediari (es. Kickstarter, GoFundMe) che trattengono commissioni elevate.
- **Mancaanza di Trasparenza:** Difficoltà nel tracciare l'effettivo utilizzo dei fondi raccolti.
- **Rischio di Censura:** Possibilità di blocco dei fondi o delle campagne da parte dell'ente gestore.
- **Tempi di Attesa:** Ritardi nel rilascio dei fondi ai creatori.



# Soluzione: Decentralizzazione e Smart Contract

- **Eliminazione Intermediari:** Connessione diretta P2P tra Creatore e Donatore.
- **Trustless:** La fiducia è garantita dal codice (Smart Contract), non da un ente.
- **Sicurezza dei Fondi:** Sistema di Escrow automatico (i fondi sono bloccati nel contratto fino al target).
- **Immutabilità:** Le regole della campagna (scadenza, target) non possono essere alterate.



# Come funziona FundETH



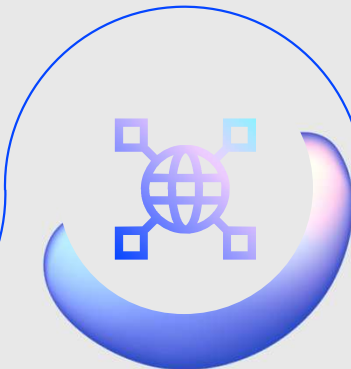
## Creazione

L'utente crea la campagna; l'immagine viene caricata su IPFS.



## Donazione

I sostenitori inviano ETH direttamente allo Smart Contract.



## Risoluzione

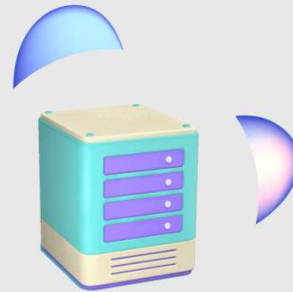
**Successo:** Il creatore preleva i fondi (Withdraw).

**Fallimento:** I donatori richiedono il rimborso (Refund).

# Componenti Principali

## Smart Contract (Solidity)

Gestione logica finanziaria,  
regole di business e custodia  
fondi (su Ethereum).



## Storage Decentralizzato (IPFS)

Archiviazione efficiente delle  
immagini delle campagne (Off-  
chain).

## Frontend (React + Vite)

Interfaccia utente per interagire  
con il contratto tramite Wallet.



# CrowdFunding.sol

- **Architettura Monolitica:** Un unico contratto per ottimizzare i costi di gas e semplificare l'audit.
- **Struttura Dati:**
  - struct Campaign: Contiene Owner, Target, Deadline, Saldo e Hash Immagine.
  - mapping campaigns: Registro di tutte le raccolte fondi.
- **Funzioni Core:** createCampaign, donateToCampaign, withdrawFunds, refund.



# Pattern di Sicurezza Implementati & Gestione Sicura dei Fondi

## Guard Check Pattern

Utilizzo di `require()` per validare input e stati



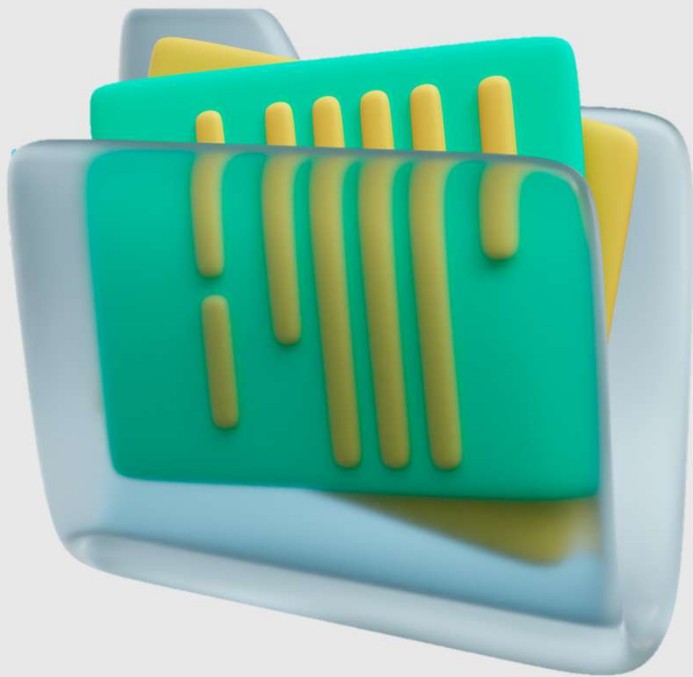
## Checks-Effects-Interactions

Prevenire attacchi di Reentrancy aggiornando lo stato prima di inviare gli ETH

## Pull over Push

Il contratto non invia automaticamente i rimborsi a tutti, ma ogni donatore deve chiamare attivamente la funzione `refund` per ritirare i propri fondi.

# Ottimizzazione Storage con IPFS



- **Problema:** Salvare immagini su Blockchain è troppo costoso.
- **Soluzione:** Caricamento immagine su IPFS (InterPlanetary File System).
- **Collegamento:** Solo il CID (Content Identifier/Hash) viene salvato nello Smart Contract.
- **Risultato:** Efficienza dei costi e decentralizzazione dei dati multimediali.



# Frontend & Interazione



## React (Vite)

Per un'interfaccia reattiva e veloce.



## Ethers.js

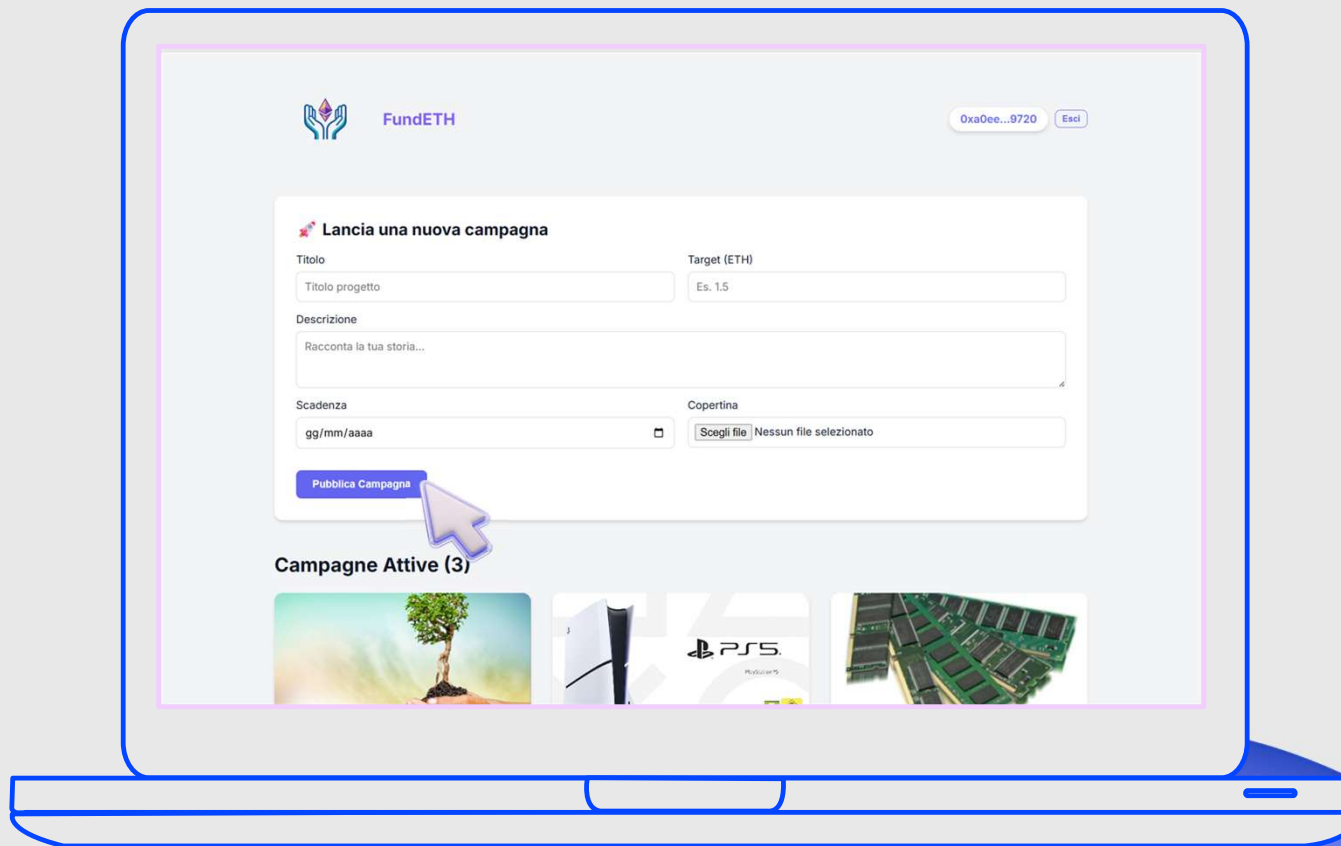
Libreria per collegare il frontend alla Blockchain Ethereum.



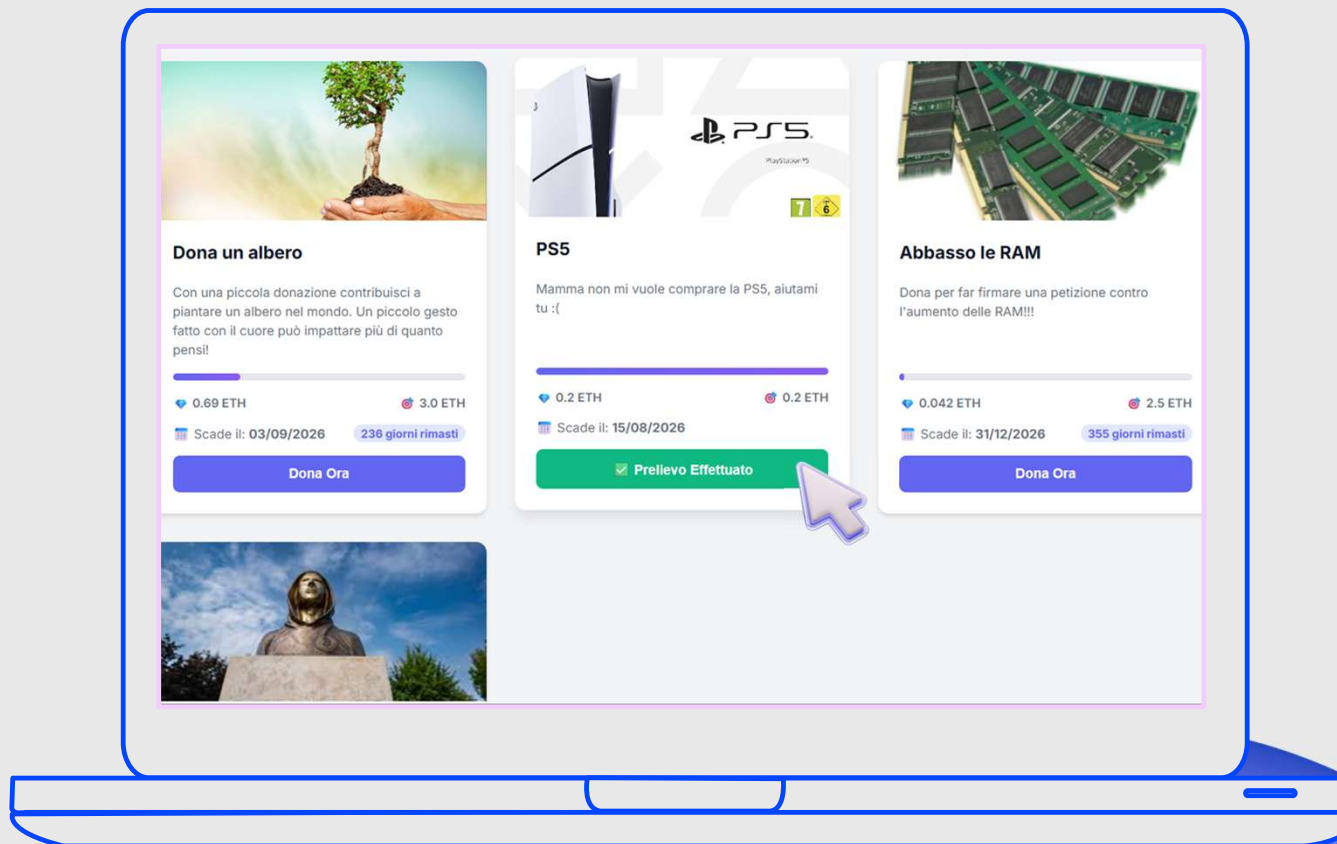
## MetaMask

Wallet per l'autenticazione utente e la firma delle transazioni.

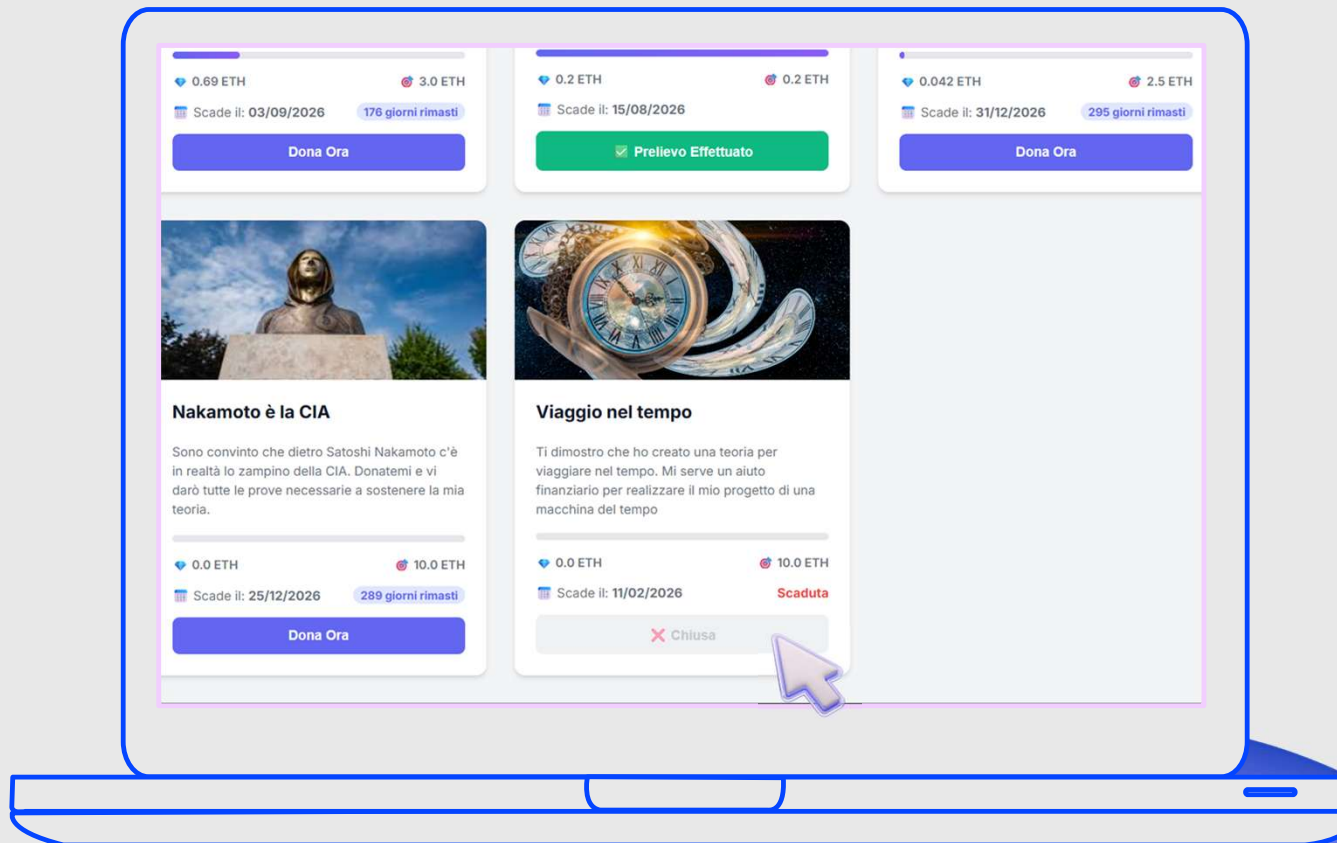
# User Experience



# User Experience



# User Experience





# Conclusioni & Sviluppi Futuri

## *Conclusioni*

- FundETH dimostra la fattibilità di un crowdfunding senza intermediari.
- Sicurezza garantita matematicamente dagli Smart Contract.

## *Sviluppi Futuri*

- Integrazione Stablecoin (USDC/USDT) per evitare la volatilità.
- Deploy su Layer 2 (Polygon/Arbitrum) per ridurre le Gas Fee.
- Governance DAO per campagne comunitarie.

# GRAZIE

Mattia Gallucci - NF22500127

 <https://github.com/MattiaGallucci/FundETH>

