

Tesina Finale di

**Data Security and Blockchain**

Corso di Laurea Magistrale in Ing. Informatica e Robotica – A.A. 2024-2025

DIPARTIMENTO DI INGEGNERIA

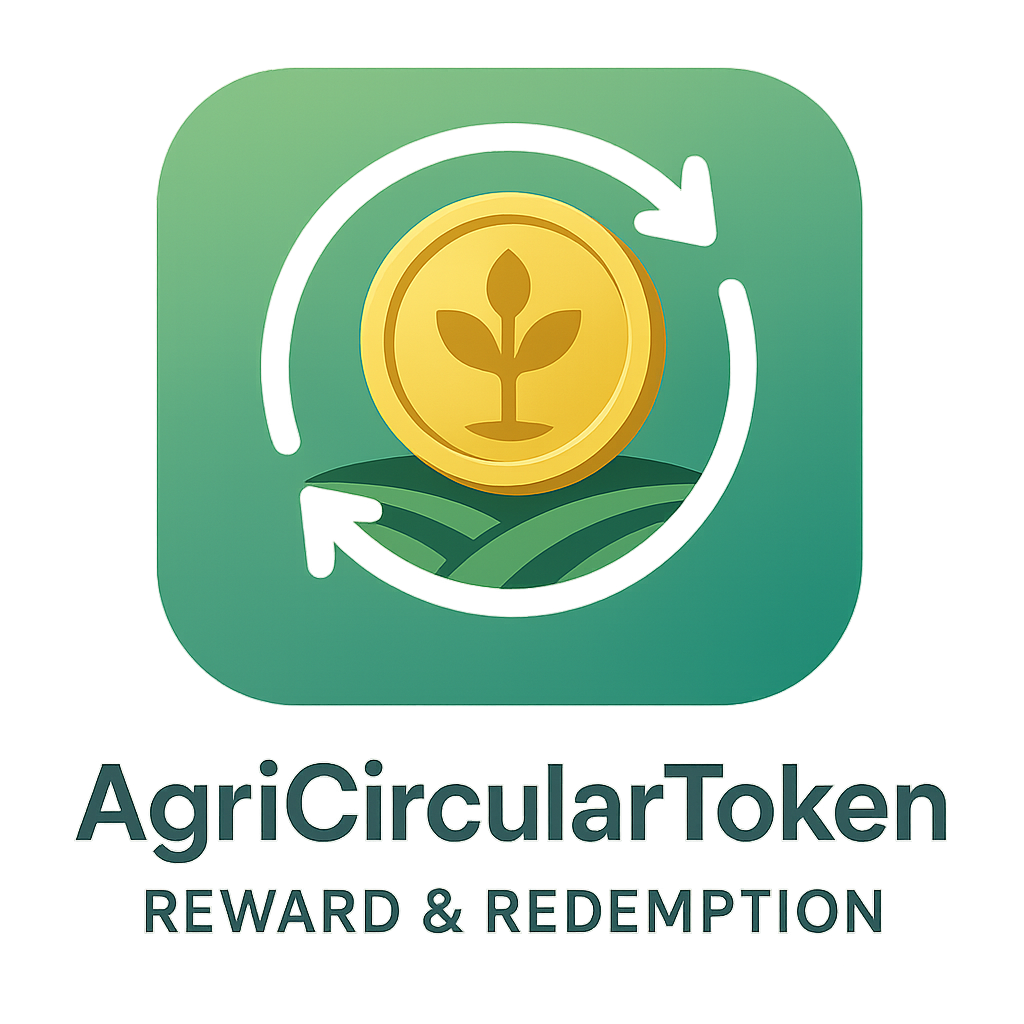
docente

Prof. Luca GRILLI

**AgriCircularToken**

Web DApp eseguita sulla Testnet Sepolia Ethereum

Tecnologie: HTML/CSS/JavaScript/Web3.js



Studente

|  |  |  |  |
| --- | --- | --- | --- |
| 374101 | **Valentina** | **Milighetti** | valentina.milighetti@studenti.unipg.it |

Data ultimo aggiornamento: 12 giugno 2025

**Sommario**

[1. Descrizione del Problema 3](#_Toc200615313)

[1.1 La circolarità nei sistemi agroalimentari 3](#_Toc200615314)

[1.2 Lo standard ERC-20 4](#_Toc200615315)

[2. Specifica dei Requisiti 6](#_Toc200615316)

[3. La Web Dapp AgriCircularToken 8](#_Toc200615317)

[3.1 Architettura del sistema Software 8](#_Toc200615318)

[3.2 Smart Contract 9](#_Toc200615319)

[3.5 Logica di acquisto dei token ACT 11](#_Toc200615320)

[3.5 Problemi riscontrati 11](#_Toc200615321)

[4. Realizzazione 12](#_Toc200615322)

[5. Conclusioni e Sviluppi Futuri 13](#_Toc200615323)

[5. Bibliografia 14](#_Toc200615324)

# 1. Descrizione del Problema

L’obiettivo di questo lavoro è lo sviluppo di una Web DApp, denominata AgriCircularToken, finalizzata alla creazione di token fungibili aderenti allo standard ERC- 20 [1] della blockchain Ethereum. La caratteristica principale di AgriCircularToken è quella di permettere agli utenti di acquistare token per aderire ad attività nell’ambito della circolarità dei sistemi agroalimentari, che siano questi dei badge, corsi o progetti.

Per iniziare, basta dichiarare il proprio impegno nella circolarità, specificando la tipologia e l’entità dell’attività, e di e acquistare i token corrispondenti. I token sono acquistabili con la criptovaluta ETH.

La Web DApp AgriCircularToken sarà eseguibile interamente nel browser, senza l’impiego di un server backend, fatto eccezione l’interfacciamento con un endpoint della blockchain. L’applicazione sarà implementata facendo uso della tecnologia HTML/CSS/JavaScript, mentre sarà impiegata la libreria Web3.js [4] per gestire l’interazione con la blockchain. Lo smart contract della DApp sarà rilasciato nella rete di test pubblica Sepolia Ethereum.

1.1 La circolarità nei sistemi agroalimentari

L'economia circolare nel settore agroalimentare rappresenta un approccio innovativo per affrontare le sfide ambientali e sociali legate alla produzione e al consumo di cibo. Questo modello si basa sui principi delle "3R": riduzione, riutilizzo e riciclo, con l'obiettivo di mantenere il massimo valore dei prodotti, componenti e materiali, trasformando gli scarti in risorse preziose.

Nel 2021 il sistema alimentare europeo ha generato oltre 58 milioni di tonnellate di rifiuti alimentari all'anno, con il 54% proveniente dalle famiglie. Questo spreco è responsabile dell'8-10% delle emissioni globali di gas serra, paragonabili a quelle del trasporto su strada. Per contrastare questo fenomeno, l'Unione Europea ha adottato l'Obiettivo di Sviluppo Sostenibile 12.3, che mira a dimezzare lo spreco alimentare pro capite entro il 2030.

La circolarità può essere applicata in tutte le fasi della filiera agroalimentare: dalla produzione alla trasformazione, fino al consumo e alla gestione degli scarti. Ad esempio, gli scarti alimentari possono essere utilizzati per produrre bioplastiche, fertilizzanti o energia rinnovabile. L'adozione di pratiche agricole sostenibili e l'ottimizzazione del packaging contribuiscono a ridurre l'impatto ambientale del settore.

Per promuovere la circolarità, è fondamentale anche la diffusione di buone pratiche e la collaborazione tra diversi attori della filiera.

In sintesi, l'economia circolare offre un'opportunità concreta per rendere la filiera agroalimentare più sostenibile, contribuendo alla riduzione degli sprechi, alla valorizzazione delle risorse e alla riduzione degli impatti ambientali.

Nel contesto della transizione verso sistemi agroalimentari più circolari, la blockchain può giocare un ruolo chiave nel valorizzare e incentivare i comportamenti virtuosi. Infatti, attraverso un sistema distribuito e verificabile, è possibile:

* Tracciare l’impegno concreto nella circolarità, come il recupero degli scarti, l’adozione di packaging sostenibili o la ridistribuzione delle eccedenze.
* Certificare pubblicamente queste azioni, offrendo visibilità e credibilità a chi adotta pratiche sostenibili.
* Incentivare le buone pratiche mediante ricompense, crediti o riconoscimenti digitali assegnati a chi dimostra comportamenti coerenti con i principi dell’economia circolare.
* Coinvolgere la comunità, che può accedere e verificare i dati in tempo reale, rafforzando il meccanismo di fiducia e controllo diffuso.

Questa combinazione di trasparenza, tracciabilità e incentivazione può accelerare l’adozione della circolarità nella filiera agroalimentare, rendendo misurabili gli sforzi dei singoli attori e premiando chi contribuisce attivamente alla sostenibilità del sistema.

1.2 Lo standard ERC-20

Lo standard ERC-20 (Ethereum Request for Comments) è utilizzato per i token fungibili sulla blockchain Ethereum e definisce un insieme di specifiche ed eventi che un token deve seguire per essere compatibile con ERC-20.

L’utilizzo dello standard ERC-20 garantisce l’uniformità sulla creazione e il trasferimento dei token su Ethereum, rendendoli più sicuri, trasparenti e interoperabili con wallet e altri sistemi senza bisogno di adattamenti specifici.

Le funzioni principali definite dallo standard ERC-20 sono:

* totalSupply(): restituisce il numero totale di token esistenti.
* balanceOf(address account): restituisce il saldo di token dell’indirizzo *account*.
* transfer(address to, uint256 value): trasferisce una quantità *value* di token dal conto del chiamante a quello del destinatario *to*. Restituisce un valore booleano che indica se l'operazione è riuscita.
* allowance(address owner, address spender): restituisce il numero rimanente di token che *spender* sarà autorizzato a spendere per conto di *owner* tramite transferFrom().
* approve(address spender, uint256 amount): autorizza *spender* ad utiliz zare una quantità *value* di token dal saldo di chi effettua la transazione
* transferFrom(address sender, address recipient, uint256 amount): trasferisce token da un indirizzo a un altro, utilizzando il meccanismo di approvazione.

Gli smart contract emettono eventi sulla blockchain che permettono di notificare azioni o cambiamenti di stato. Lo standard ERC-20 individua due eventi principali che servono a tracciare i trasferimenti e le approvazioni sui token:

* Transfer(address indexed from, address indexed to, uint256 value): emesso quando i token vengono trasferiti da un account all’altro
* Approval(address indexed owner, address indexed spender, uint256 value): emesso quando un account *owner* autorizza un altro account *spender* a spendere un certo valore di token per suo conto

# 2. Specifica dei Requisiti

L’applicazione AgriCircularToken dovrà soddisfare i seguenti requisiti:

1. Connessione al wallet Metamask
2. Deploy dello Smart Contract sulla rete Sepolia

* definisce i token ERC-20 “ACT” specifici dell’applicazione

1. Registrazione di attività circolari attraverso un apposito form
2. Calcolo dei token ERC-20 equivalenti alle attività registrate
3. Acquisto di token ERC-20:

* mint dei token equivalenti a un’attività registrata

1. Acquisto di badge che certificano l’impegno nella circolarità:
   * sono suddivisi per livello crescente
   * un utente può acquistare un badge di livello utilizzando token ERC-20 solo se possiede un badge di livello
   * i badge possono essere creati solo dall’*Owner* dello smart contract, che specifica il nome del badge, la descrizione, il livello e il numero di token necessari per riscattarlo
   * i badge disponibili devono essere visualizzabili all’interno di un’area dedicata
2. Acquisto di corsi informativi sulla circolarità nei sistemi agroalimentari:

* un utente può acquistare un corso solo se non l’ha già acquistato in precedenza
* i token corrispondenti a un corso vengono trasferiti dall’utente al proprietario del corso
* i corsi acquistabili devono essere visualizzabili all’interno di un’area dedicata

1. Creazione di corsi informativi, specificando:

* Nome
* Descrizione
* Promotore (società o ente)
* Costo del corso in token

1. Contribuzione a progetti ecologici:

* l’utente specifica il numero di token che vuole trasferire al proprietario del progetto
* i progetti attivi devono essere visualizzabili all’interno di un’area dedicata
* i token vengono trasferiti al proprietario del progetto quando il numero target dei token viene raggiunto

1. Creazione di progetti, specificando:

* nome
* descrizione
* posizione geografica
* numero di token da raccogliere

1. Area personale in cui poter visualizzare:

* saldo dei token ERC-20 posseduti dall’utente
* totale dei token acquistati
* badge posseduti
* corsi acquistati
* progetti contribuiti
* lista delle transazioni con link alla piattaforma etherscan.io

1. Architettura dell’applicazione *frontend-only*
2. Interfaccia utente semplice e intuitiva

* Deve mostrare dei messaggi quando si effettuano le operazioni principali, oltre ad avvisi di eventuali errori

3. La Web Dapp AgriCircularToken

AgriCircularToken è una Web Dapp incentrata sull’acquisto di token ERC-20 con simbolo ACT, con la possibilità di scambiarli per ottenere dei badge, acquistare dei corsi informativi o contribuire a dei progetti inerenti alla circolarità nella filiera agroalimentare. L’obiettivo principale è quello di incentivare gli utenti ad adottare delle pratiche sostenibili, in quanto la piattaforma può rappresentare una vetrina che contribuisce alla loro reputazione, mostrando il loro impegno in maniera verificabile e trasparente.

La realizzazione della Dapp prevede il deploy dello smart contract sulla testnet Sepolia e l’implementazione dell’interfaccia web che comunica con la blockchain.

Le operazioni ammesse dall’applicazione sono semplici da realizzare da parte dell’utente, in quanto sono previsti dei semplici form html o dei bottoni per inviare le transazioni. Inoltre, la comunicazione tra l’interfaccia web e lo smart contract è svolta con l’impiego dell’estensione MetaMask.

3.1 Architettura del sistema Software

L’architettura del sistema è progettata per essere *frontend-only*, per cui tutte le operazioni vengono eseguite direttamente nel browser web dell’utente, tranne l’interfacciamento con l’endpoint della blockchain. L’interfaccia utente comunica con la testnet Sepolia tramite l’endpoint dedicato, utilizzando la libreria Web3.js che offre un’interazione semplice ed efficiente con la blockchain.

L’applicazione è accessibile dalla pagina html *index.html*, che contiene tutte le componenti per effettuare le operazioni previste e lo script per il caricamento dello smart contract e il collegamento del wallet. Il file *style.css* gestisce tutta la formattazione delle componenti html. Le funzionalità del software sono gestite da 5 file JavaScript:

* *dashboard.js:* gestisce il caricamento dell’area personale dell’utente
* *buyTokens.js:* permette di comprare token ACT registrando delle attività
* *badges.js:* se il wallet è collegato, carica tutti i badge disponibili per l’utente e gestisce l’acquisto degli stessi. Se il wallet collegato è quello del proprietario dello smart contract, permette la creazione di nuovi badge
* *courses.js:* se il wallet è collegato, carica tutti i corsi disponibili per l’utente e gestisce l’acquisto degli stessi. Permette di creare nuovi corsi
* *projects.js:* se il wallet è collegato, carica tutti i progetti disponibili per l’utente e gestisce la contribuzione degli stessi, prendendo in input la cifra che l’utente intende devolvere. Permette di creare nuovi progetti

3.2 Smart Contract

Lo smart contract AgriCircularToken.sol definisce la struttura del token ACT e le operazioni permesse. Il deploy del contratto è stato realizzato attraverso l’IDE web Remix e il compilatore Sepolia nella sua versione 0.8.20.

Il contratto eredita il comportamento dello standard ERC-20 e di Ownable e definisce tre strutture di dati, rispettivamente relative a badge, corsi e progetti:

**struct Course** {

  string name;                // Nome del progetto

    string description;         // Descrizione del progetto

  string promoter;            // Promotore del progetto

    uint256 requiredTokens;     // Numero di token per acquistare il corso

    address courseOwner;        // Indirizzo del creatore del corso

}

**struct EcologicProject** {

string name;                // Nome del progetto

string description;         // Descrizione del progetto

    string location;            // Posizione del progetto

      uint256 requiredTokens;     // token richiesti

bool active;                // Stato del progetto

    uint256 totalContributed;   // Totale dei token già contribuiti

    address projectOwner;       // Indirizzo del creatore del progetto

}

**struct CircularityBadge** {

string name;                // Nome del progetto

string description;         // Descrizione del progetto

uint256 level;              // livello di importanza

uint256 requiredTokens;     // Numero di token per acquistare il badge

}

Gli eventi emessi dal contratto sono i seguenti:

* event TokensMinted(address indexed recipient, uint256 amount): emesso quando un utente registra un’attività e acquista token ACT
* event ActivityRecorded(address indexed user, string activity, uint256 quantity): emesso quando un utente registra un’attività e acquista token ACT
* event ProjectCreated(uint256 projectId, string name): emesso dopo la creazione di un progetto
* event ProjectCompensated(uint256 projectId, address indexed user, uint256 tokens): emesso quando un utente devolve token ACT a un progetto
* event ProjectCompleted(uint256 projectId, address indexed projectOwner, uint256 tokens): emesso quando un progetto viene completato
* event BadgeCreated(uint256 badgeId, string name): emesso alla creazione di un badge
* event badgeRedeemed(uint256 badgeId, address indexed user): emesso quando un utente acquista un badge
* event CourseCreated(uint256 courseId, string name): emesso alla creazione di un progetto
* event courseCompleted(uint256 courseId, address indexed projectOwner, address indexed user): emesso quando un utente acquista un progetto

Seguono le funzioni pubbliche che regolano le operazioni principali.

* **buyTokens(uint256 tokenAmount, string memory activity, uint256 quantity) public payable**

Controlla che ci siano ETH necessari per effettuare l’operazione e procede con il mint dei token ACT richiesti. Incrementa la variabile globale che tiene traccia del totale degli ACT creati ed emette gli eventi correlati alla registrazione di attività e il mint dei token.

* **createBadge(string memory name, string memory description, uint256 level, uint256 requiredTokens) public onlyOwner**

La funzione è invocabile solo dall’owner del contratto e crea un nuovo badge, specificando nome, descrizione, livello e token richiesti. Inoltre, incrementa il contatore degli id dei badge e associa un oggetto CircularityBadge all’array dei badge del contratto. Emette l’evento BadgeCreated().

* Le funzioni createProject() e createCourse() sono analoghe a createBadge(), a meno del fatto che possono essere chiamate specificando l’indirizzo di qualsiasi utente.
* **compensateProject(uint256 projectId, uint256 tokenAmount) public**

Controlla che l’id del progetto sia valido e che il progetto sia attivo, oltre a verificare che l’indirizzo che chiama la funzione possieda i token che intende spendere. TokenAmount deve anche essere maggiore di zero e non eccedere i token rimanenti del progetto specificato. Successivamente trasferisce gli ACT dall’utente al proprietario del progetto e aggiorna totalContributed. Se a questo punto totalContributed è uguale a requiredTokens, imposta il progetto come inattivo, trasferisce gli ACT al proprietario del progetto ed emette l’evento di progetto completato, altrimenti emette soltanto l’evento ProjectCompensated.

* **buyBadge(uint256 badgeId) public**

Controlla che l’id passato sia valido, che l’utente possieda abbastanza ACT per riscattare il badge e che non lo abbia già riscattato in precedenza. Verifica poi che l’utente possieda altri badge di livello immediatamente inferiore (se il livello del badge è maggiore di 1), per poi trasferire i token dall’utente al wallet del proprietario del progetto ed emettere l’evento badgeRedeemed.

* **buyCourse(uint256 courseId) public**

Controlla che l’id passato sia valido, che l’utente possieda abbastanza ACT per riscattare il corso e che non lo abbia già acquistato in precedenza. Inoltre, verifica che l’indirizzo di chi acquista il corso non sia lo stesso del proprietario del progetto, prima di procedere con il trasferimento degli ACT dall’utente al proprietario del progetto e all’emissione dell’evento courseCompleted.

3.5 Logica di acquisto dei token ACT

Sono state definite delle attività specifiche che l’utente può registrare per ottenere ACT. La logica di equivalenza tra attività-quantità e token corrispondenti è gestita lato software e non lato Smart Contract per permettere più flessibilità e prevede le seguenti regole, per ciascuna attività:

* **compostaggio**: 2 ACT per kg
* **utilizzo di energia rinnovabile**: 10 ACT per kWh
* **risparmio idrico**: 3 ACT per kWh
* **risparmio di plastica nel packaging**: 1.5 ACT per kg

4. Realizzazione

La pagina iniziale si presenta come segue:

Immagine che contiene testo, schermata, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

Figura : pagina iniziale della web Dapp

La pagina presenta vari bottoni, ciascuno per ogni attività che si può realizzare e uno che mostra se il wallet MetaMask è stato collegato o meno.

Una volta collegato il wallet, è possibile registrare una pratica per ottenere ACT, nella sezione “record activity”:

Immagine che contiene testo, schermata, software, Carattere

Il contenuto generato dall'IA potrebbe non essere corretto.

5. Conclusioni e Sviluppi Futuri

Interdum et malesuada fames ac ante ipsum primis in faucibus. Donec eget velit aliquet, dapibus nunc feugiat, vulputate neque. Donec tincidunt et leo non mattis. Quisque lacinia eu nisi scelerisque vestibulum. Nullam dolor tortor, varius ut pretium placerat, suscipit facilisis ex. Morbi eu dolor nec sem fermentum accumsan at semper ipsum. Cras et nibh a sem mollis eleifend eu tincidunt est. Vivamus pretium congue quam. Nullam suscipit lorem tincidunt metus interdum, vel fringilla nunc mollis. Morbi tempus vehicula eros. Praesent ultricies porta tortor. Nulla tincidunt sed ipsum at vulputate. Nulla rutrum lectus eu risus aliquet pretium. Vestibulum ac pulvinar risus.

# 5. Bibliografia

1. F. Vogelsteller and V. Buterin. ERC-20: Token Standard | eips.ethereum.org, 2015. [Online <https://eips.ethereum.org/EIPS/eip-20>; Consultato il 21-maggio-2025]
2. Ethereum Foundation. Remix- ethereum ide, 2025. [Online <https://remix.ethereum.org/>; Consultato il: 5-maggio-2025]
3. L. v. Wassenaer, C. Verdouw, A, Kassahun, M. v. Hilten, K. v. d. Meij, B. Tekinerdogan. Tokenizing circularity in agri-food systems: A conceptual framework and exploratory study, 2023. [Online <https://www.sciencedirect.com/science/article/pii/S0959652623016852>; Consultato il 27-aprile-2025]
4. ChainSafe Systems. Web3.js — Javascript Ethereum API, 2025. [Online <https://web3js.org/>; Consultato il 3-giugno-2024]
5. sepolia.etherscan.io. Testnet Sepolia (ETH) Blockchain Explorer, 2025. [Online <https://sepolia.etherscan.io/>; Consultato il 5-maggio-2025]
6. Ente Italiano di Normazione. Circolarità nella filiera agro-alimentare, 2023. [Online <https://www.uni.com/circolarita-nella-filiera-agro-alimentare/>; Consultato il 4-giugno-2025]
7. B. Anzà. L’economia circolare nel contesto dei sistemi alimentari, 2024. [Online <https://www.italiaclima.org/leconomia-circolare-nel-contesto-dei-sistemi-alimentari/>; Consultato il 4-giugno-2025]
8. PinkUnicorn80. The Power of Giving — The Food Token Project, 2024. [Online <https://medium.com/@PinkUnicorn80/the-power-of-giving-the-food-token-project-26983b34fd87>; Consultato il 4-giugno-2025]