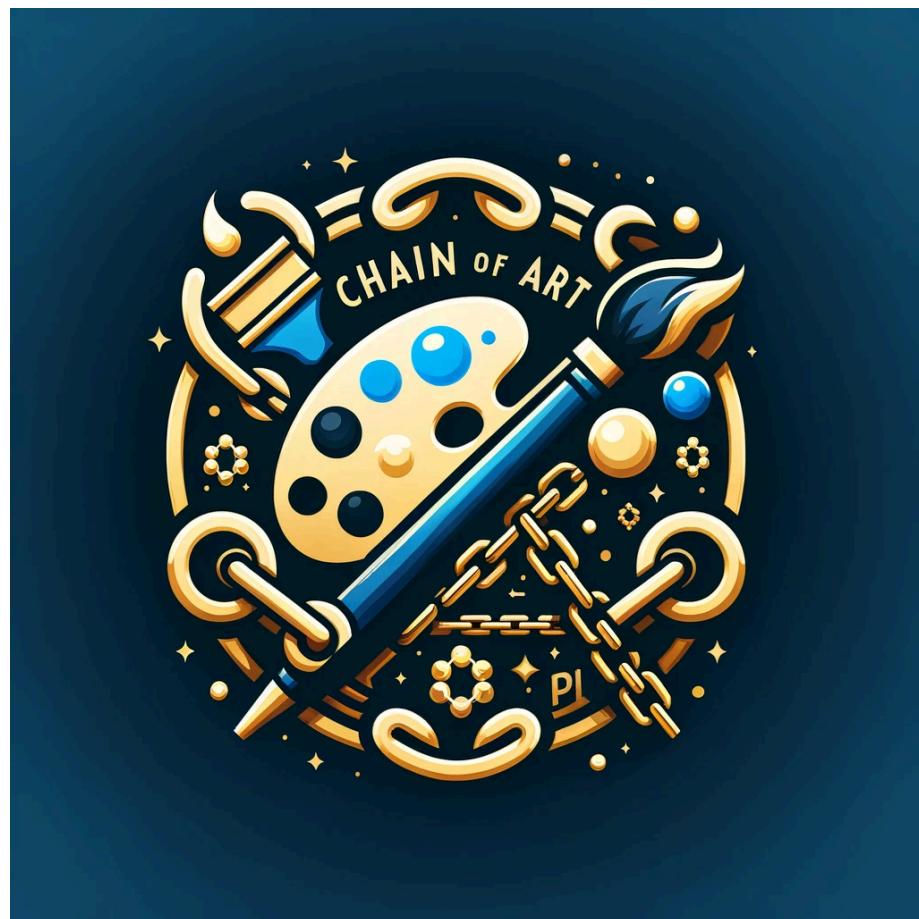




Laurea Magistrale in Informatica - Università degli studi di Salerno
Corso di *Sicurezza dei Dati* -
Professori Alfredo De Santis, Christiancarmine Esposito



Documentazione progetto - Chain of Art

Autori

Ciano Marco
matr. 0522501674
m.ciano4@studenti.unisa.it

De Martino Angela
matr. 0522501589
a.demartino68@studenti.unisa.it



Sommario

1. Introduzione.....	3
2. Tecnologie utilizzate.....	4
3. Fase di progettazione.....	5
3.1 Requisiti funzionali.....	5
3.1.1 RF_1: Visualizzazione delle opere d'arte digitali (Priorità Alta).....	5
3.1.2 RF_2: Acquisto di un'opera d'arte digitale (Priorità Alta).....	5
3.1.3 RF_3: Visualizzazione delle opere d'arte acquistate (Priorità Alta).....	5
3.1.4 RF_4: Autenticazione degli utenti (Priorità Alta).....	5
3.1.5 RF_5: Aggiornamento dello stato dell'opera d'arte (Priorità Media).....	5
3.2 Mock-up.....	6
3.2.1 Sezione degli acquisti.....	6
3.2.2 Sezione di visualizzazione delle opere acquistate.....	7
3.3 Use Case Model.....	8
3.4 Use Cases.....	8
3.4.1 UC_01: Acquisto di un'opera d'arte digitale.....	8
3.4.2 UC_02: Visualizzazione delle opere d'arte acquistate.....	9
3.5 Sequence Diagram.....	10
3.5.1 SD_01: Acquisto di un'opera d'arte digitale.....	10
3.5.2 SD_02: Visualizzazione delle opere d'arte acquistate.....	11
4. Struttura del progetto.....	12
5. Setup e configurazione.....	14
5.1 Prerequisiti.....	14
5.2 Installazione delle Dipendenze.....	14
5.3 Configurazione di Truffle e Ganache.....	14
5.4 Distribuzione dei Contratti Intelligenti.....	14
5.5 Avvio dell'Interfaccia Utente.....	14
5.6 Esecuzione dei test.....	15
6. Focus su smart contract.....	16
7. Front-end.....	18
8. Interazione Web3.....	21
9. Testing.....	22
Considerazioni e conclusioni.....	23
Dizionario.....	24



1. Introduzione

Nel panorama emergente delle applicazioni decentralizzate (Dapp²), "Chain of Art" si posiziona come un progetto pionieristico che fonde l'innovazione tecnologica della blockchain con il mondo dell'arte. Il progetto "Chain of Art" è un ecosistema che unisce l'arte digitale, la tecnologia blockchain e il Web 3.0 per creare una piattaforma innovativa di acquisto e vendita di opere d'arte digitali. Questa combinazione offre agli artisti un modo decentralizzato per esporre e vendere le proprie opere, mentre fornisce agli acquirenti una sicurezza e una trasparenza garantite dalla tecnologia blockchain.

Il sito web interagisce con lo smart contract Ethereum chiamato "PaintContract". Questo contratto gestisce la collezione di quadri digitali, tracciando la disponibilità di ogni opera, gestendo le transazioni di acquisto e mantenendo un registro degli acquisti effettuati da ciascun utente.

Gli utenti possono navigare tra le opere, visualizzare i dettagli dei quadri e acquistarli tramite l'interfaccia del sito web. Le transazioni di acquisto sono eseguite sulla blockchain Ethereum, garantendo trasparenza e immutabilità.

In questo paper, esploreremo come "Chain of Art" sfrutta la tecnologia blockchain per rivoluzionare il mercato dell'arte, creando un nuovo paradigma in cui la bellezza artistica si intreccia con la generosità umanitaria. Analizzeremo l'architettura della piattaforma, il suo modello di business, le strategie di coinvolgimento della comunità artistica e dei benefattori, e l'impatto sociale che il progetto si prefigge di realizzare. Attraverso "Chain of Art", si apre un nuovo capitolo nel mondo dell'arte, dove ogni transazione diventa un atto di sostegno a cause più grandi, dimostrando il potere dell'arte come forza per il cambiamento positivo nella società.



2. Tecnologie utilizzate

Il cuore di questo progetto è alimentato da una suite di strumenti tecnologici avanzati, tra cui Ganache, MetaMask, Truffle, Solidity, Web3.js e Bootstrap, ciascuno svolgendo un ruolo fondamentale nel suo funzionamento e sviluppo.

Ganache è un pezzo chiave del puzzle, fornendo un ambiente di sviluppo blockchain locale per testare smart contracts e transazioni in un contesto sicuro e controllato, simulando la rete Ethereum.

MetaMask agisce come un interfaccia essenziale, un portafoglio criptovalute basato su browser che consente agli utenti di interagire con la DApp in modo sicuro e decentralizzato. È fondamentale per la gestione delle identità degli utenti e per le transazioni di acquisto.

Il framework di sviluppo Truffle offre una suite di strumenti per lo sviluppo, il test e la distribuzione di smart contracts, garantendo che siano robusti, sicuri e ben integrati nel network Ethereum.

Solidity è un linguaggio di programmazione ad alto livello progettato specificamente per lo sviluppo di smart contract sulla piattaforma Ethereum.

Web3.js è una libreria JavaScript che è stata utilizzata per interagire con la blockchain Ethereum. Questa libreria consente di comunicare con nodi Ethereum e interagire con gli smart contract sulla blockchain.

Ed infine, il framework Bootstrap è stato utilizzato per lo sviluppo del front-end, e per definire, quindi, la grafica dell'applicazione.



3. Fase di progettazione

3.1 Requisiti funzionali

3.1.1 RF_1: Visualizzazione delle opere d'arte digitali (Priorità Alta)

Tutti gli utenti potranno visualizzare l'elenco completo di tutte le opere d'arte digitali compresi i loro dettagli come nome dell'opera, artista, prezzo del quadro (in ETH).

3.1.2 RF_2: Acquisto di un'opera d'arte digitale (Priorità Alta)

Gli utenti avranno la possibilità di acquistare l'opera d'arte che preferiscono.

3.1.3 RF_3: Visualizzazione delle opere d'arte acquistate (Priorità Alta)

Gli utenti devono poter accedere ad un elenco delle opere d'arte che hanno acquistato.

3.1.4 RF_4: Autenticazione degli utenti (Priorità Alta)

Prima di poter procedere all'acquisto di un'opera d'arte digitale, gli utenti devono poter autenticarsi attraverso il loro wallet digitale Metamask, garantendo così l'immutabilità delle transazioni legate a delle identità verificate.

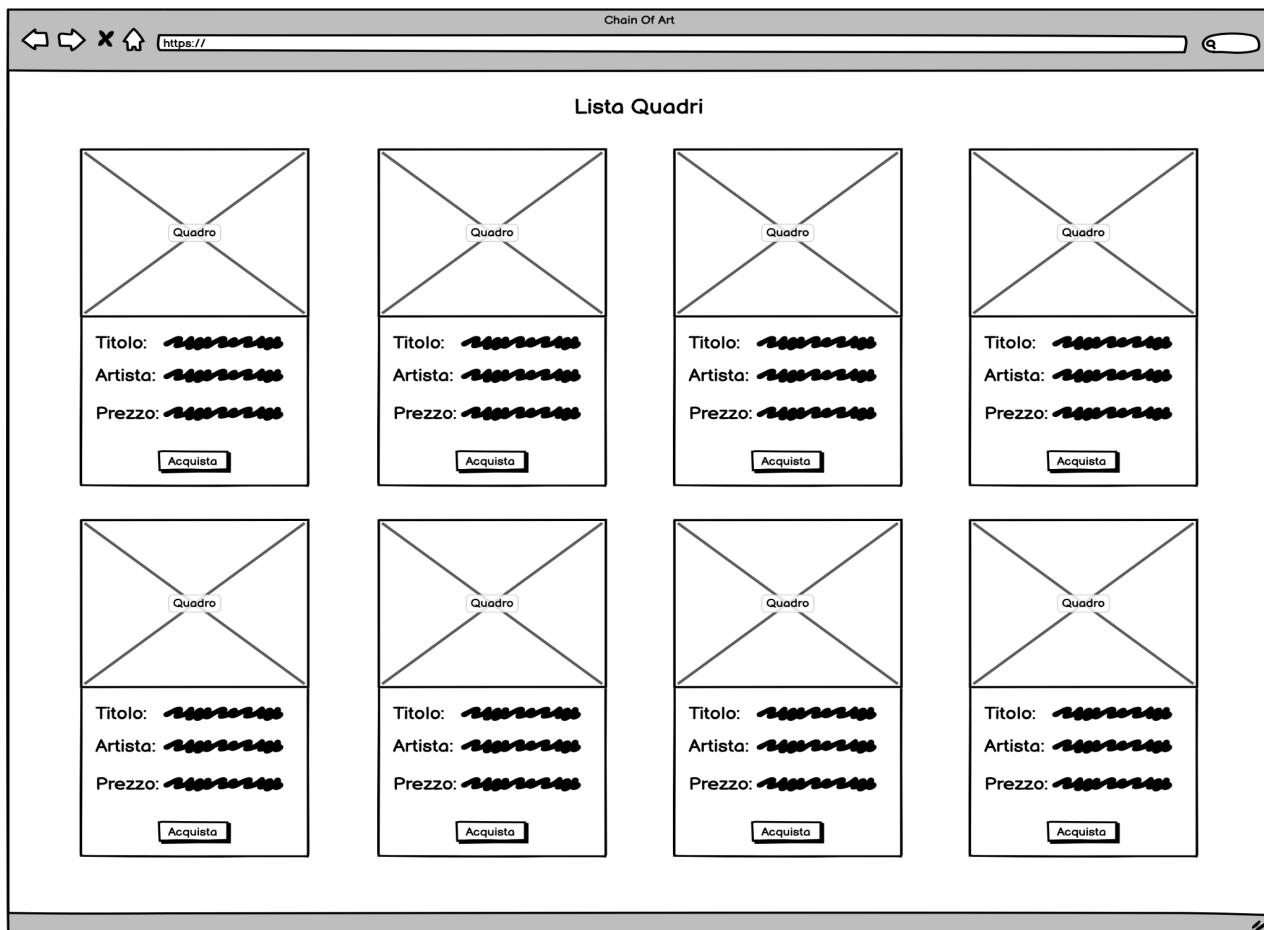
3.1.5 RF_5: Aggiornamento dello stato dell'opera d'arte (Priorità Media)

Il sistema deve aggiornare lo stato del quadro da "Disponibile" ad "Acquistato" dopo l'avvenuto acquisto da parte di un utente.

3.2 Mock-up

Sono stati realizzati i prototipi relativi alla sezione riportante la lista delle opere digitali (dove gli utenti possono effettuare l'acquisto dei quadri) e la sezione relativa alla visualizzazione delle opere acquistate.

3.2.1 Sezione degli acquisti





Laurea Magistrale in Informatica - Università degli studi di Salerno

Corso di *Sicurezza dei Dati* -

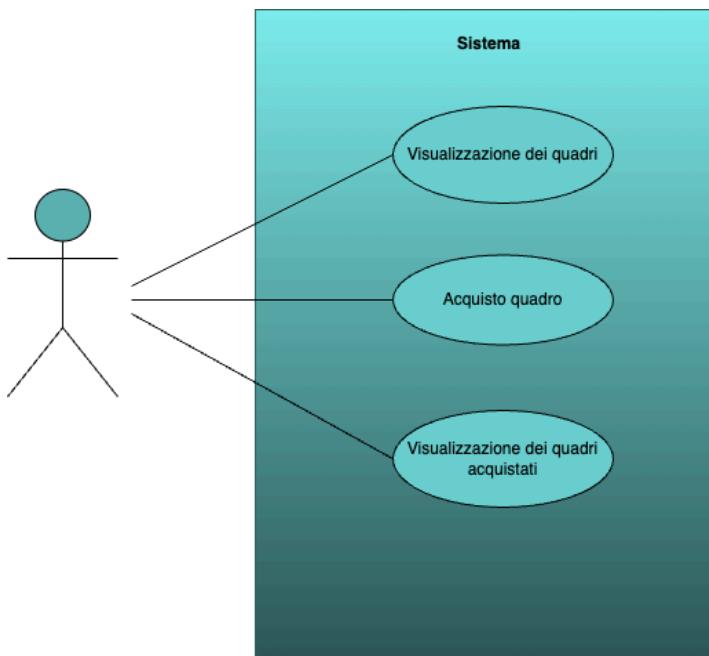
Professori Alfredo De Santis, Christiancarmine Esposito

3.2.2 Sezione di visualizzazione delle opere acquistate

ID	Quadro	Titolo	Artista	Prezzo	Stato
.....		ACQUISTATO
.....		ACQUISTATO

3.3 Use Case Model

Lo Use Case Model riportato qui di seguito è una rappresentazione visuale e testuale che descrive le funzionalità di Chain Of Art da una prospettiva degli utenti.



3.4 Use Cases

3.4.1 UC_01: Acquisto di un'opera d'arte digitale

Attore: Utente

Entry condition: L'utente si trova nella homepage della dApp e si è autenticato al suo wallet digitale Metamask.

Flusso degli eventi:

1. L'utente naviga attraverso l'elenco delle opere d'arte digitali disponibili sulla dApp. Ogni quadro presenta le seguenti informazioni: Immagine del quadro, titolo del quadro, nome e cognome dell'artista e il prezzo.
2. L'utente seleziona il quadro che desidera comprare.
3. L'utente clicca sul pulsante “Acquista” posizionato sotto i dettagli del quadro.
4. Il sistema controlla che l'utente non abbia già acquistato quel quadro.
5. Il sistema mostra una schermata per confermare la transazione.
6. L'utente autorizza la transazione

Exit condition (On Success): Il sistema completa l'acquisto del dipinto e informa l'utente del successo della transazione.

Exit condition (On failure): Il sistema notifica l'utente di un errore nella transazione.



Flussi alternativi:

- Se nel punto 4 il sistema nota che l’utente ha già acquistato il quadro, la transazione viene annullata e viene mostrato all’utente il seguente messaggio di errore “Errore nella transazione”

3.4.2 UC_02: Visualizzazione delle opere d’arte acquistate

Attore: Utente

Entry condition: L’utente si trova nella homepage della dApp e si è autenticato al suo wallet digitale Metamask ed ha effettuato l’acquisto di almeno un quadro.

Flusso degli eventi:

1. L’utente naviga nella homepage verso la sezione “Quadri Acquistati”.
2. L’utente visualizza una tabella riportante i quadri acquistati con i rispettivi dettagli per ogni quadro.

Exit condition (On Success): Il sistema mostra all’utente i quadri acquistati con i relativi dettagli.

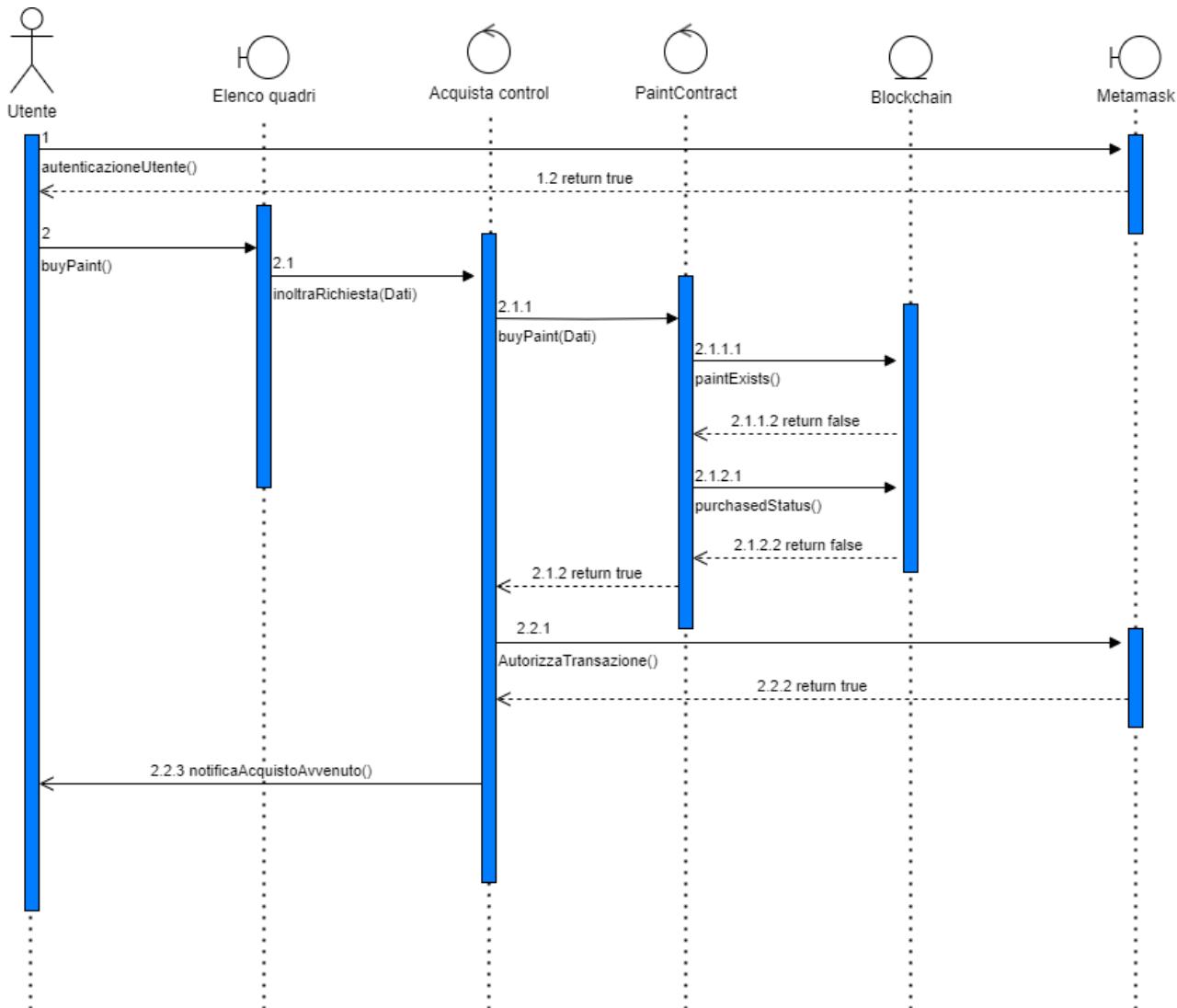
Flussi alternativi:

- Se nel punto 2 l’utente non ha effettuato l’acquisto almeno un quadro, il sistema mostra un’immagine con label di “no data” seguita dal messaggio “Non hai acquistato alcun quadro”.

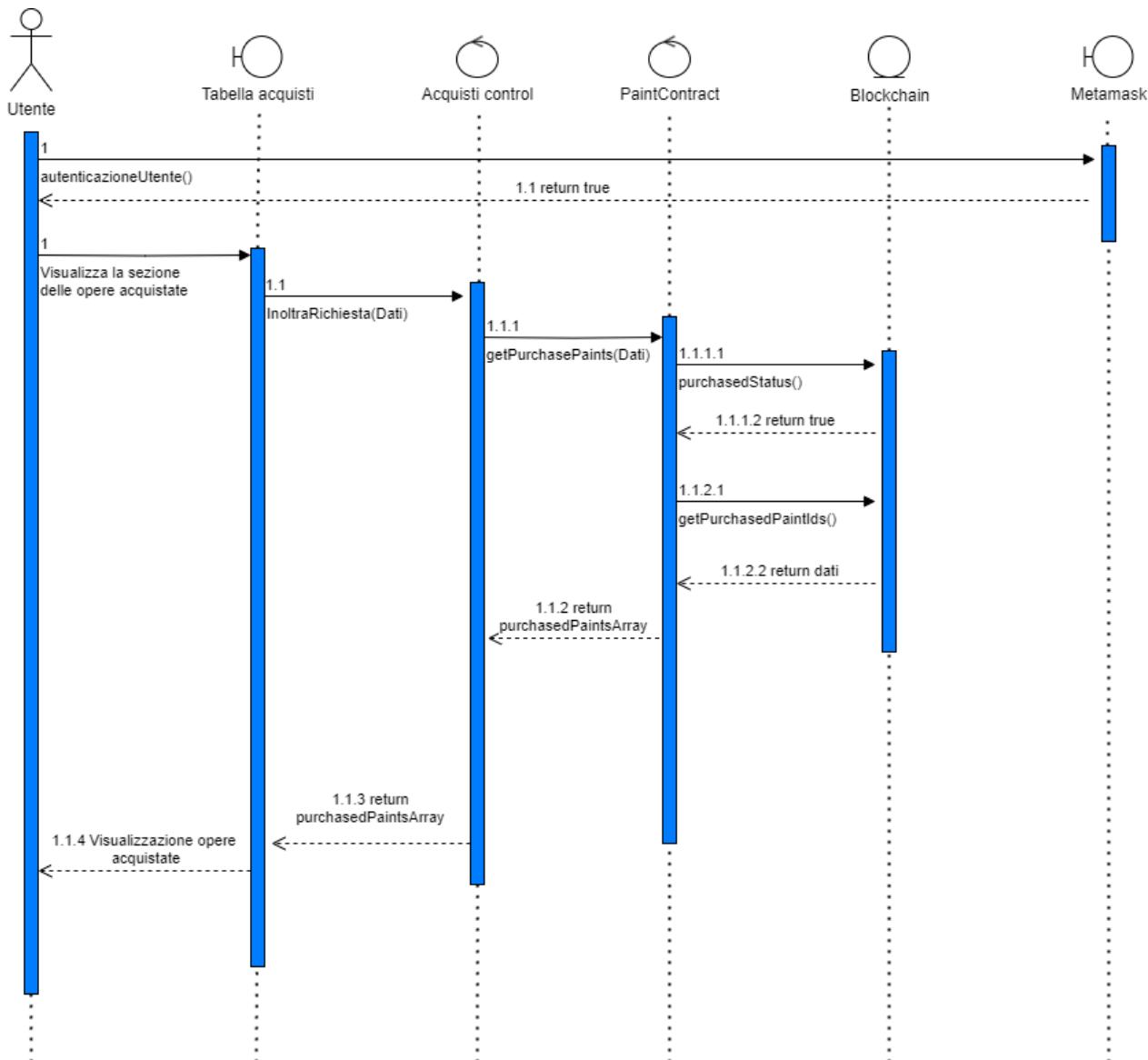
3.5 Sequence Diagram

I sequence diagram che vengono presentati in questa paragrafo descrivono come i diversi oggetti coinvolti nel sistema interagiscono sequenzialmente nel corso del tempo di interazione.

3.5.1 SD_01: Acquisto di un'opera d'arte digitale



3.5.2 SD_02: Visualizzazione delle opere d'arte acquistate





4. Struttura del progetto

<https://bit.ly/3vk7432>

Il progetto “Chain Of Art” è organizzato in una struttura di cartelle e file che facilita lo sviluppo e la manutenzione del codice. Di seguito è fornita una descrizione dettagliata della struttura del progetto:

Directory Principali

build/: Contiene i file compilati del contratto intelligente. Questi file JSON includono l’ABI¹ e l’indirizzo del contratto, essenziali per l’interazione front-end con il contratto.

contracts/: Questa directory ospita i file sorgente dello smart contract scritti in Solidity. Qui risiede la logica di business principale del progetto.

migrations/: Contiene gli script di migrazione usati da Truffle per distribuire i contratti intelligenti sulla blockchain.

node_modules/: Directory generata automaticamente contenente tutte le dipendenze del progetto installate tramite NPM (Node Package Manager).

src/: La cartella più importante per lo sviluppo front-end. Include tutti i file HTML, CSS, JavaScript e le immagini necessarie per l’interfaccia utente.

test/: Contiene gli script di test per i contratti intelligenti, permettendo di eseguire test automatizzati per assicurare la correttezza del codice del contratto.

File Principali

truffle-config.js: Il file di configurazione per Truffle, utilizzato per impostare le reti di blockchain, i compilatori e altre opzioni di configurazione.

package.json e **package-lock.json**: Questi file gestiscono le dipendenze del progetto e assicurano una coerenza nell’ambiente di sviluppo.

Directory src/

All’interno della directory src/, si trovano le seguenti sottodirectory e file:

src/index.html: Il file principale dell’interfaccia utente, che funge da punto di ingresso per l’applicazione web.



src/js/: Contiene i file JavaScript, inclusi app.js (che gestisce la logica principale dell’interfaccia utente e l’interazione con il contratto), web3.min.js (libreria per interagire con Ethereum) e truffle-contract.js (per lavorare con i contratti Truffle).

src/css/: Ospita i fogli di stile CSS per la personalizzazione dell’aspetto dell’interfaccia utente.

src/fonts/ e src/img/: Contengono rispettivamente i font e le immagini utilizzati nell’interfaccia utente.

src/paints.json: Un file JSON che potrebbe essere usato per memorizzare i dati dei quadri, come parte della logica dell’applicazione.

Questa struttura consente una separazione chiara tra la logica del contratto intelligente e l’interfaccia utente, facilitando lo sviluppo e la manutenzione.



5. Setup e configurazione

Per configurare e avviare il progetto “Chain Of Art”, segui i passaggi dettagliati qui sotto.

5.1 Prerequisiti

Prima di procedere, assicurati di avere installato:

- **Node.js e npm**: Necessari per gestire le dipendenze JavaScript del progetto.
- **Truffle**: Utilizzato per lo sviluppo, il test e la distribuzione dei contratti intelligenti Solidity.
- **Ganache**: Una blockchain personale per lo sviluppo Ethereum, utilizzata per testare il progetto in un ambiente locale.

5.2 Installazione delle Dipendenze

1. Clonazione del Progetto: Se il progetto è ospitato su un repository Git, clonalo nel tuo sistema. Altrimenti, assicurati di avere tutti i file del progetto estratti dalla cartella ZIP.
2. Installazione delle Dipendenze Node: Apri un terminale nella directory principale del progetto e esegui il comando **npm install package.json**. Questo comando installerà tutte le dipendenze elencate nel file package.json.

5.3 Configurazione di Truffle e Ganache

1. Avvio di Ganache: Avvia Ganache (GUI o CLI). Prendi nota dell’indirizzo e della porta mostrati nell’interfaccia di Ganache, che di solito è <http://localhost:7545>.
2. Configurazione di Truffle: Assicurati che nel file truffle-config.js le impostazioni di rete siano configurate per corrispondere a quelle di Ganache. Questo garantirà che Truffle possa connettersi alla tua blockchain locale.

5.4 Distribuzione dei Contratti Intelligenti

1. Compilazione dei Contratti: Nella directory principale del progetto, esegui il comando **truffle compile** per compilare i contratti intelligenti.
2. Migrazione dei Contratti: Utilizza il comando **truffle migrate --network development** per distribuire i contratti sulla blockchain Ganache.

5.5 Avvio dell’Interfaccia Utente

1. Visualizzazione dell’Interfaccia Utente: Apri il file src/index.html nel tuo browser per interagire con l’applicazione. Se il progetto include un server di sviluppo (ad esempio, configurato nel package.json), puoi avviarlo eseguendo npm start.



Laurea Magistrale in Informatica - Università degli studi di Salerno

Corso di *Sicurezza dei Dati* -

Professori Alfredo De Santis, Christiancarmine Esposito

5.6 Esecuzione dei test

Nel file PaintContract.test.js vengono definiti i test relativi alla creazione di un nuovo quadro, l’acquisto di un nuovo quadro ed infine la prevenzione dell’acquisto di uno stesso quadro due volte. Attraverso il comando **truffle test** verranno eseguiti tutti gli script di test presenti nel file all’interno della cartella test.



6. Focus su smart contract

Il cuore del progetto “Chain Of Art” è rappresentato dagli smart contract scritti in Solidity. Questi contratti risiedono nella directory **contracts**/ del progetto e sono responsabili della logica di business principale, come la gestione degli acquisti e la proprietà delle opere d’arte digitali. Di seguito è fornita una descrizione dettagliata del funzionamento e delle funzionalità principali dello smart contract.

Descrizione dello Smart Contract

- Nome del Contratto: PaintContract
- Versione Solidity: pragma solidity ^0.8.0;

Funzioni Principali dello Smart Contract

1. **createPaint(uint id, string memory title, string memory img, string memory artist, string memory price)**: Crea un nuovo quadro nella collezione con i dettagli forniti e verifica se il quadro creato con lo stesso id non esiste già.
1. **loadPaints()**: Carica inizialmente una serie di quadri usando la funzione createPaint. Tale funzione viene utilizzata per inizializzare il contratto con un insieme predefinito di quadri.
1. **buyPaint(uint paintId)**: Consente all’utente di acquistare un quadro specifico, Verificando che il quadro sia disponibile e che l’utente non abbia acquistato già lo stesso quadro. Inoltre, aggiorna lo stato del quadro come venduto, registra l’acquisto ed emette un evento.
2. **PurchasedStatus(address buyer, uint paintId)**: Funzione di supporto che verifica se un utente ha già acquistato un determinato quadro.
3. **getPurchasedPaints(address buyer)**: Restituisce un array di strutture Paint che rappresentano i quadri acquistati da un utente specifico.
4. **getPaintDetails(uint paintId)**: Restituisce i dettagli di un quadro dato il suo Id.

Eventi

- **PurchasedPaint**: Questo evento viene emesso quando un’opera d’arte viene acquistata, registrando l’acquirente e l’ID dell’opera.

Sicurezza e Considerazioni

- Validazioni: Il contratto include controlli per evitare acquisti doppi e per assicurarsi che vengano inviati fondi adeguati per l’acquisto.



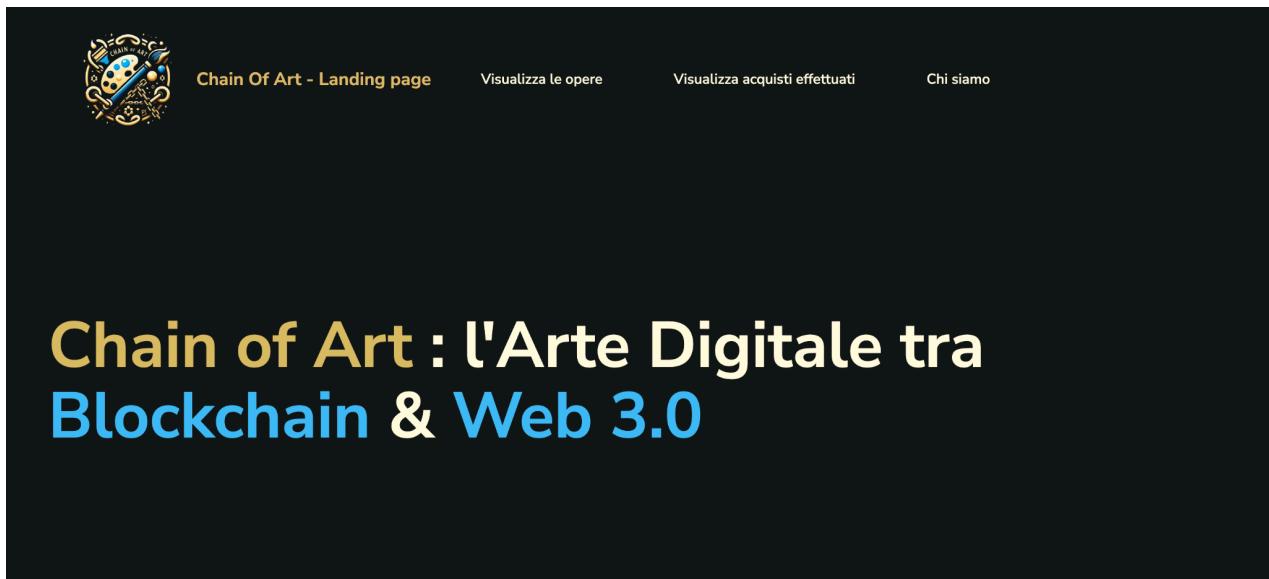
Deploy e Interazione

- Distribuzione: Il contratto viene distribuito sulla blockchain attraverso gli script di migrazione presenti nella cartella migrations/.
- Interazione: L'interazione con il contratto avviene tramite l'interfaccia utente del progetto, che utilizza Web3.js per inviare transazioni e richiamare le funzioni dello smart contract.

7. Front-end

L’interfaccia utente del progetto “Chain Of Art” offre agli utenti la possibilità di interagire con il sistema di acquisto delle opere d’arte. Realizzata con HTML, CSS e JavaScript, l’interfaccia è progettata per essere intuitiva e accessibile, consentendo agli utenti di visualizzare facilmente le opere disponibili e di procedere con gli acquisti.

Qui di seguito è mostrata una sequenza di interazione che mostra l’acquisto di un quadro sulla piattaforma Chain Of Art:



Descrizione Immagine 1: Rappresenta l’header della pagina principale (index.html)



Laurea Magistrale in Informatica - Università degli studi di Salerno

Corso di *Sicurezza dei Dati* -

Professori Alfredo De Santis, Christiancarmine Esposito

Selezione del quadro

Seleziona il quadro che desideri acquistare

Titolo	Autore	Prezzo	Coin	Acquista
Distese Stellate	Alessandro Vibrante	0.33	ETH	Acquista
Sinfonia di Spirali Cosmiche	Marco Tessitore	0.43	ETH	Acquista
Il giorno	Vittorio Formale	0.26	ETH	Acquista
Equilibrio Astratto	Silvio Baratta	0.47	ETH	Acquista

Descrizione immagine 2: L'utente procede alla selezione del quadro che intende acquistare, effettuando lo scrolling verso il basso per visualizzare i quadri disponibili.

Selezione del quadro

Seleziona il quadro che desideri acquistare

MetaMask Notification

Chain-of-Art

Account 3 → 0xFaD5A...307aE

http://localhost:3000

0xFaD5A...307aE : INTERAZIONE CONTRATTO

0.33 ETH

DETtagli HEX

Market >

Gas (estimated) 0.000485 0.000485 ETH
Likely in < 30 seconds Max fee: 0.000485 ETH

Totale 0.330485 0.330485 ETH
Amount + gas fee Max amount: 0.330485 ETH

Annulla Conferma

Titolo	Autore	Prezzo	Coin	Acquista
Distese Stellate	Alessandro Vibrante	0.33	ETH	Acquista
Sinfonia di Spirali Cosmiche	Marco Tessitore	0.43	ETH	Acquista
Il giorno	Vittorio Formale	0.26	ETH	Acquista
Equilibrio Astratto	Silvio Baratta	0.47	ETH	Acquista

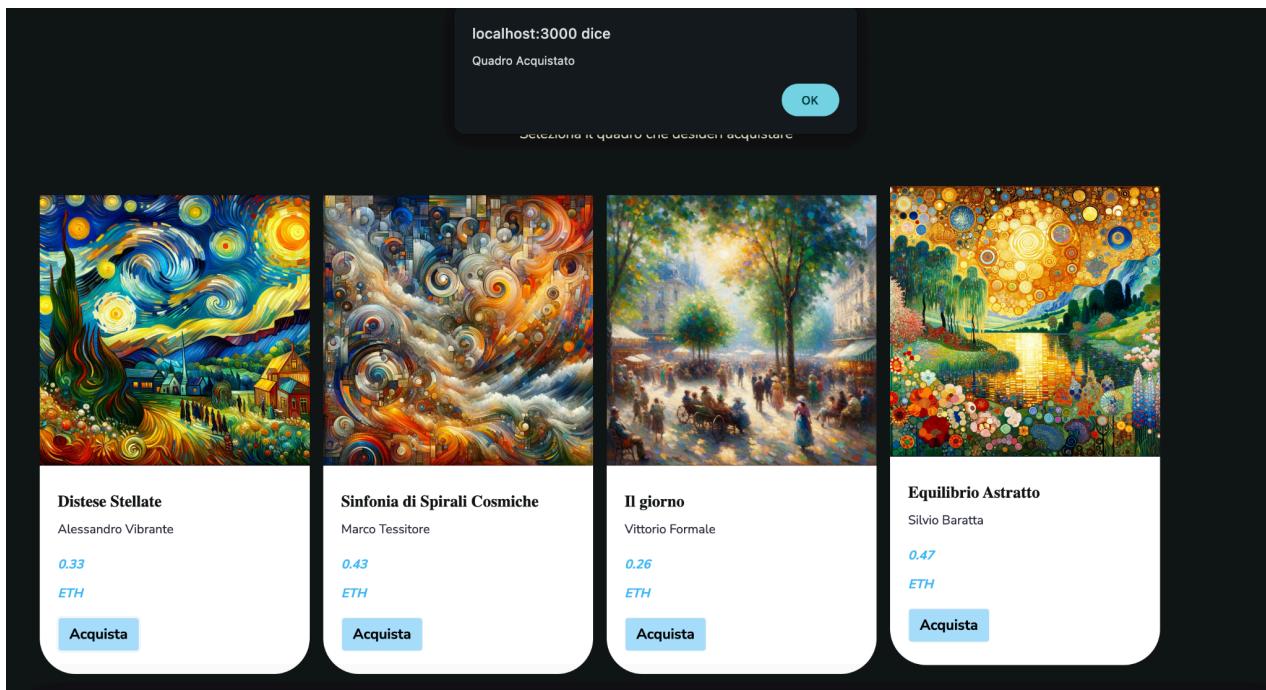
Descrizione immagine 3: Una volta selezionato il quadro da acquistare, verrà avviata la transazione attraverso l'apertura della finestra di dialogo di Metamask, riportante il prezzo del quadro selezionato.



Laurea Magistrale in Informatica - Università degli studi di Salerno

Corso di *Sicurezza dei Dati* -

Professori Alfredo De Santis, Christiancarmine Esposito



Descrizione immagine 4: Effettuata la transazione, l'utente viene notificato dal sistema dell'avvenuto acquisto del quadro.

Quadri Acquistati				
ID	QUADRO	TITOLO QUADRO	ARTISTA	PREZZO
0		Distese Stellate	Alessandro Vibrante	0.33 ETH

Descrizione immagine 5: Effettuando lo scrolling verso il basso, l'utente può visualizzare una tabella riportante tutti i quadri acquistati con i relativi dettagli, senza perdere così la cronologia delle transazioni effettuate.



8. Interazione Web3

Nel progetto “Chain Of Art”, la libreria Web3.js gioca un ruolo fondamentale nell’interazione tra l’interfaccia utente (front-end) e la blockchain Ethereum. Il file web3.min.js, situato nella directory src/js/, è il motore che permette all’applicazione di comunicare con lo smart contract sulla blockchain.

Funzioni Chiave di Web3.js:

Connessione con Ethereum: Web3.js si connette al provider Ethereum, come MetaMask o un nodo locale Ganache. Questo permette all’applicazione di interagire con la blockchain, inviando transazioni e richieste di lettura.

Interazione con lo Smart Contract: Utilizzando l’ABI (Application Binary Interface) fornita dal file PaintContract.json, Web3.js consente all’applicazione di richiamare le funzioni dello smart contract, come l’acquisto di opere d’arte o la verifica della proprietà.

Gestione delle Transazioni: La libreria gestisce l’invio di transazioni alla blockchain, inclusa la gestione del gas e la conversione dei valori in Ether.

Integrazione nel File app.js:

Inizializzazione di Web3: Nel file app.js, Web3.js viene inizializzato e configurato per utilizzare il provider Ethereum disponibile. Questo passaggio è essenziale per garantire che l’applicazione possa effettuare chiamate allo smart contract.

Richiamo delle Funzioni del Contratto: app.js include la logica per interagire con le funzioni dello smart contract tramite Web3.js, consentendo agli utenti di eseguire azioni come l’acquisto di quadri e la visualizzazione delle informazioni di proprietà.



9. Testing

Il testing è un aspetto vitale nel ciclo di vita dello sviluppo dello smart contract nel progetto “Chain Of Art”. I test assicurano che lo smart contract funzioni come previsto e che sia sicuro da eventuali vulnerabilità.

Test dello Smart Contract:

Script di Test: Situati nella directory `test/`, gli script di test sono scritti per verificare vari aspetti e funzionalità dello smart contract, come la corretta assegnazione della proprietà e il funzionamento delle transazioni.

Esecuzione dei Test con Truffle: Utilizzando Truffle, una suite di strumenti per lo sviluppo di Ethereum, i test possono essere eseguiti eseguendo **truffle test** dalla riga di comando. Questo comando esegue tutti gli script di test e fornisce un report dei risultati.

Importanza dei Test: I test aiutano a scoprire bug, problemi di logica o potenziali vulnerabilità nello smart contract. Sono una parte fondamentale per garantire la stabilità e la sicurezza del contratto prima della sua distribuzione su una rete pubblica Ethereum.



Considerazioni e conclusioni

Il progetto “Chain Of Art” unisce arte e tecnologia attraverso l’utilizzo della blockchain, offrendo una piattaforma innovativa per l’acquisto e la vendita di arte digitale. Con un’interfaccia utente intuitiva e un solido smart contract, il progetto ha già dimostrato il suo potenziale nel democratizzare l’accesso all’arte digitale. Guardando al futuro, ci sono diverse aree entusiasmanti per l’espansione e il miglioramento. La piattaforma potrebbe beneficiare dell’introduzione di sistemi di aste e di rivendita, rendendo il mercato dell’arte più dinamico e accessibile. Allo stesso tempo, miglioramenti nell’interfaccia utente e l’implementazione di funzionalità di valutazione e social sharing⁴ potrebbero aumentare significativamente l’engagement degli utenti³ e la visibilità degli artisti. Mantenendo un focus sulla sicurezza e l’innovazione tecnologica, “Chain Of Art” è ben posizionato per continuare a crescere e affermarsi come punto di riferimento nel panorama dell’arte digitale.



Dizionario

1. ABI: (Application Binary Interface): rappresenta l'interfaccia tra due programmi o componenti di software, consentendo loro di comunicare tra loro.
2. DApp: applicazione decentralizzata.
3. Engagement degli utenti: si riferisce al coinvolgimento e all'interazione degli utenti con l'applicazione o il sito web.
4. Valutazione e social sharing: riferito alle possibili funzionalità future della piattaforma "Chain Of Art" in termini di valutazioni delle opere d'arte e condivisione su social media.