certificazione e la compravendita di orologi di lusso su blockchain.

Watchain

Documentazione progetto per il corso “Sicurezza dei Dati”

MARIO LEZZI
(m.lezzi@studenti.unisa.it)

UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA

Sommario

1. Watchain

1.1 Introduzione

Watchain è un'applicazione decentralizzata (DApp) progettata per gestire la compravendita di orologi di lusso in modo sicuro, tracciabile e verificabile tramite blockchain. L'obiettivo del sistema è garantire l'autenticità dei prodotti e la trasparenza delle transazioni, riducendo i rischi tipici del mercato secondario, come contraffazioni, frodi e mancanza di informazioni sulla provenienza.

Ogni orologio è rappresentato da un NFT che ne costituisce l'identità digitale e ne registra in modo permanente la storia sulla blockchain. La proprietà del bene viene trasferita insieme al token, rendendo verificabili produzione, certificazione e passaggi di mano. Le regole di scambio sono implementate tramite smart contract, che automatizzano pagamenti, escrow e controlli di autorizzazione, riducendo la necessità di logiche applicative arbitrarie.

Il sistema distingue tre ruoli principali: il Producer, che crea gli asset digitali e gestisce le vendite primarie; i Reseller, che certificano e rivendono i prodotti; i Consumer finali, che acquistano sul mercato secondario. Questa separazione riflette la filiera reale e consente di applicare permessi e responsabilità in modo chiaro.

Dal punto di vista architetturale, Watchain combina una componente on-chain, responsabile della logica di business e della persistenza delle informazioni critiche, con una componente off-chain che fornisce interfaccia utente, autenticazione tramite wallet e gestione delle sessioni. L'utente mantiene il controllo della propria identità firmando una challenge crittografica con il wallet; le operazioni di scrittura verso la blockchain sono orchestrate tramite servizi applicativi e middleware, che gestiscono l'invocazione dei contratti in modo controllato.

Nel complesso, Watchain dimostra come l'integrazione tra smart contract e applicazioni web tradizionali possa essere utilizzata per costruire un marketplace digitale affidabile per beni di alto valore, con tracciabilità on-chain e un'integrazione off-chain orientata a usabilità e governance.

2. Requisiti Non Funzionali (RNF)

2.1 RNF di Sicurezza

ID	Requisito	Descrizione
RNF-S1	Autenticazione crittografica	Il sistema deve consentire l'accesso esclusivamente tramite firma crittografica del wallet dell'utente.
RNF-S2	Prevenzione del riutilizzo delle richieste	Il sistema deve impedire che una richiesta di autenticazione intercettata possa essere riutilizzata da terzi.
RNF-S3	Controllo degli accessi	Il sistema deve consentire l'utilizzo delle funzionalità solo a utenti autorizzati, limitando le operazioni in base al ruolo coperto.
RNF-S4	Integrità e immutabilità dei dati critici	Le informazioni essenziali del mercato devono essere protette da modifiche retroattive e risultare verificabili in ogni momento.
RNF-S5	Protezione delle transazioni	Il sistema deve garantire che i pagamenti avvengano in modo sicuro e coerente, evitando perdite di fondi o stati inconsistenti.

2.2 RNF Architetturali

ID	Requisito	Descrizione
RNF-A1	Separazione delle responsabilità	Il sistema deve mantenere separate la logica di business, l'interfaccia utente e la gestione delle sessioni, riducendo l'impatto di eventuali malfunzionamenti.
RNF-A2	Aggiornamento automatico dello stato	Il sistema deve notificare automaticamente i client quando avvengono eventi rilevanti, così da mantenere l'interfaccia sincronizzata senza interventi manuali.
RNF-A3	Fonte unica di verità	Lo stato critico del mercato deve essere determinato da un'unica fonte affidabile, consultabile da tutti gli attori.

2.3 RNF Operativi

ID	Requisito	Descrizione
RNF-OP1	Sospensione d'emergenza	Il sistema deve permettere di sospendere temporaneamente le operazioni di mercato in caso di anomalie o problemi di sicurezza.
RNF-OP2	Recupero fondi	Il sistema deve consentire agli utenti di recuperare i propri fondi maturati anche in situazioni di errore o interruzione delle vendite.
RNF-OP3	Isolamento delle funzionalità amministrative	Le operazioni amministrative devono essere accessibili solo ai soggetti autorizzati e separate da quelle destinate agli utenti comuni.

3. Requisiti Funzionali (RF)

ID	Attore	Requisito	Descrizione	Priorità
RF_1	Producer	Minting orologio	Il Producer genera un nuovo orologio digitale. Il sistema assegna automaticamente un identificativo univoco e l'orologio viene aggiunto all'inventario del Producer.	Alta
RF_2	Producer	Abilitazione Reseller	Il Producer può autorizzare un indirizzo wallet ad operare come Reseller attivo, abilitandolo alle operazioni di certificazioni e di vendita sul mercato secondario.	Alta
RF_3	Producer	Disabilitazione Reseller	Il Producer può revocare l'operatività di un Reseller, impedendogli nuove operazioni di certificazione e di vendita sul mercato secondario.	Alta
RF_4	Producer	Listing Primario	Il Producer può selezionare un orologio dal proprio inventario e metterlo in vendita sul mercato primario, specificandone il prezzo.	Alta
RF_5	Producer	Blocco Factory	Il Producer può sospendere le operazioni di minting e di certificazione.	Critica
RF_6	Producer	Blocco Marketplace	Il Producer può sospendere le operazioni di vendita e acquisto.	Critica
RF_7	Producer/Reseller	Visualizzazione mercato primario	Il Reseller e il Producer possono visualizzare l'elenco degli orologi disponibili sul mercato primario.	
RF_8	Reseller	Acquisto primario	Il Reseller può acquistare un orologio dal mercato primario. Al termine dell'operazione, la proprietà dell'orologio viene trasferita al Reseller.	Alta
RF_9	Reseller	Certificazione	Il Reseller può certificare l'autenticità di un orologio in suo possesso.	Critica
RF_10	Reseller	Listing secondario	Il Reseller può mettere in vendita un orologio certificato sul mercato secondario.	Alta
RF_11	Producer/Reseller	Annullamento listing	Il venditore può ritirare un orologio dalla vendita.	Media
RF_12	Producer/Reseller	Aggiorna prezzo	Il venditore può aggiornare il prezzo di vendita di un orologio.	

RF_13	Consumer	Visualizzazione mercato secondario	Il Consumer può visualizzare gli orologi certificati disponibili sul mercato secondario.	Media
RF_14	Consumer	Acquisto finale	Il Consumer può acquistare un orologio dal mercato secondario.	Alta
RF_15	Tutti	Visualizzazione inventario	Ogni utente può visualizzare il proprio inventario personale di orologi.	Alta
RF_16	Producer/Reseller	Prelievo fondi	Il Producer e il Reseller possono prelevare manualmente i fondi accumulati dalle vendite.	Alta

4. Architettura del sistema

Watchain è progettato come un sistema a più livelli che combina componenti decentralizzate e componenti applicative tradizionali. Le funzionalità critiche del mercato, come la gestione degli asset digitali, i trasferimenti di proprietà e le regole di compravendita, sono affidate alla blockchain, che garantisce trasparenza e immutabilità dei dati.

Attorno a questo nucleo decentralizzato è presente una componente applicativa off-chain, responsabile della gestione dell'interfaccia utente, dell'autenticazione e della semplificazione delle interazioni con la rete blockchain.

Un livello di integrazione intermedio consente la comunicazione tra applicazione e smart contract, permettendo l'invio di transazioni e la ricezione di eventi in tempo reale.

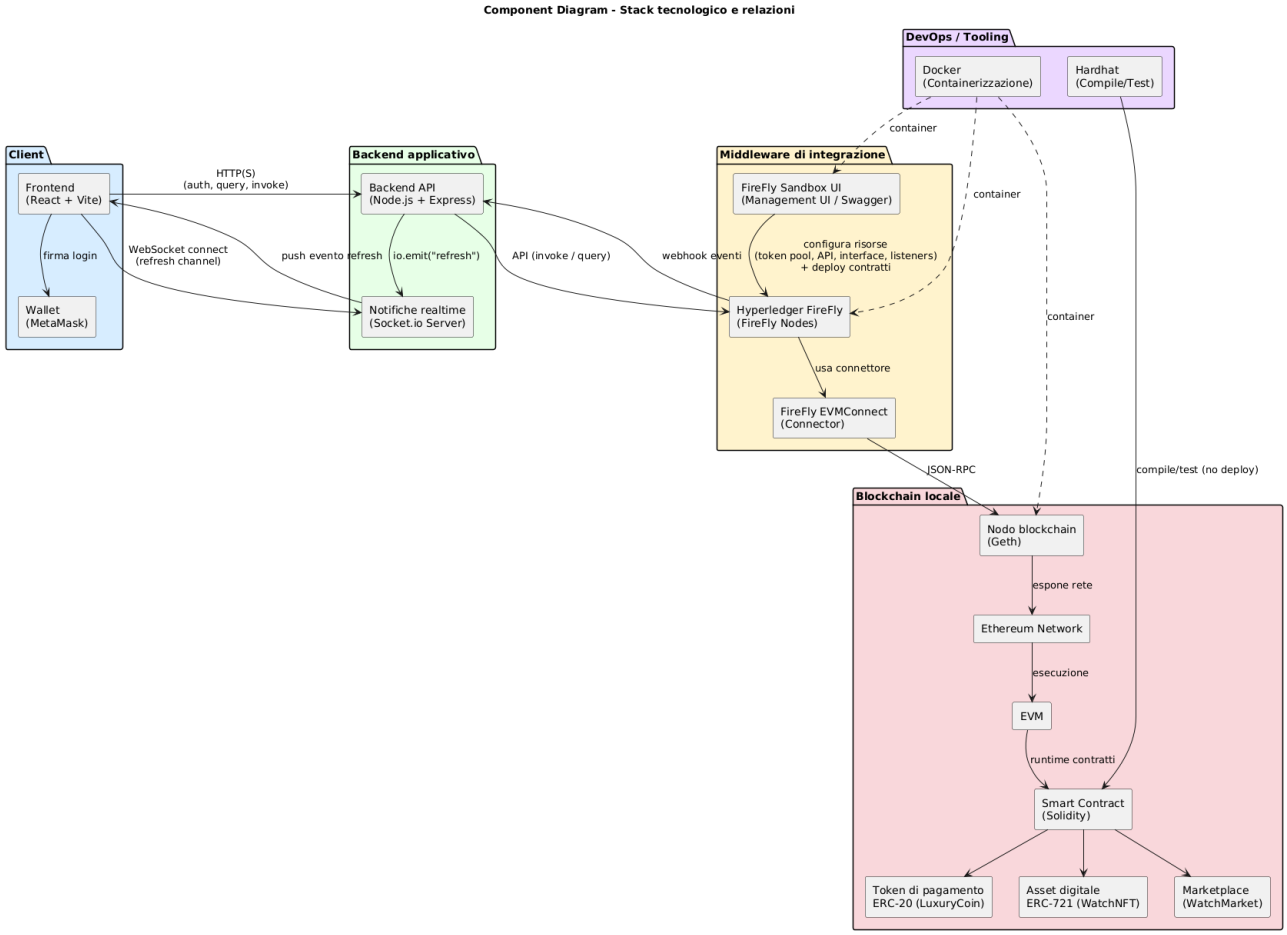
Questa separazione consente di sfruttare i vantaggi della blockchain per la sicurezza e l'affidabilità, mantenendo al contempo l'usabilità e la reattività tipiche delle applicazioni web moderne.

4.1 Stack tecnologico

La seguente tabella riassume le principali tecnologie utilizzate e il loro ruolo all'interno del sistema.

Componente	Tecnologia	Ruolo
Piattaforma blockchain	Ethereum	Rete su cui risiede lo stato del marketplace e vengono eseguiti i contratti
Macchina virtuale	EVM	Ambiente di esecuzione degli smart contract
Nodo blockchain	Geth (ethereum/client-go)	Nodo Ethereum locale utilizzato per esporre l'interfaccia di rete e processare le transazioni
Smart contract	Solidity	Implementano NFT, token di pagamento e regole di mercato on-chain
Token di pagamento	ERC-20 (LuxuryCoin)	Token di pagamento utilizzato per le operazioni economiche del sistema
Asset digitale	ERC-721 (WatchNFT)	Rappresentazione degli orologi come NFT
Middleware di integrazione	Hyperledger FireFly	Strato intermedio che gestisce transazioni, letture e la propagazione degli eventi tra app e blockchain
Connettore blockchain	FireFly EVMConnect	Componente che collega FireFly alla rete EVM
Interfaccia di gestione	FireFly Sandbox UI	Interfaccia utilizzata per configurare e gestire risorse (token pool, API, Interfacce)
Deployment contratti	FireFly Swagger	Deploy sulla rete locale tramite endpoint FireFly.
Backend applicativo	Node.js + Express	Gestione autenticazione, sessione e servizi off-chain
Frontend	React + Vite	Interfaccia utente del marketplace
Wallet	MetaMask	Consente la firma crittografica durante l'autenticazione (login)
Notifiche realtime	Socket.io	Sincronizzazione dell'interfaccia tramite notifiche di aggiornamento
Sviluppo e test contratti	Hardhat	Compilazione, test e generazione ABI dei contratti
Containerizzazione	Docker	Esecuzione isolata dell'ambiente blockchain e dei servizi di integrazione

4.2 Component Diagram



4.3 Infrastruttura blockchain

L'infrastruttura blockchain di Watchain viene eseguita interamente tramite container Docker.

In questo ambiente sono avviati un nodo Ethereum locale basato su Geth e i servizi necessari all'interazione con la rete.

Geth espone JSON-RPC usato da FireFly/EVMConnect; frontend e backend non parlano con Geth ma con FireFly via REST (le write passano dal proxy backend).

Gli smart contract vengono compilati/testati (Hardhat) e la distribuzione sulla rete locale avviene tramite gli endpoint di deploy esposti dal nodo FireFly del Producer (Swagger).

Una volta pubblicati, la loro logica viene eseguita dalla Ethereum Virtual Machine (EVM), che garantisce esecuzione deterministica e persistenza dello stato.

Docker viene utilizzato esclusivamente per ospitare tali servizi infrastrutturali (nodo blockchain e componenti di integrazione), mentre frontend e backend vengono eseguiti esternamente ai container e comunicano con la rete tramite API HTTP.

4.4 Hyperledger FireFly

Per semplificare l'interazione con la blockchain è stato adottato Hyperledger FireFly come livello di integrazione intermedio.

Invece di interagire direttamente con il nodo Ethereum tramite chiamate JSON-RPC, l'applicazione utilizza le API REST fornite da FireFly, che astraggono l'invio delle transazioni, la lettura dello stato e la gestione degli eventi on-chain.

Nel progetto sono stati configurati tre nodi FireFly distinti, uno per ciascun attore logico del sistema (Producer, Reseller e Consumer), così da rappresentare entità operative separate.

Gli smart contract, compilati con Hardhat, vengono distribuiti sulla rete locale richiamando gli endpoint di deploy esposti da FireFly.

Le ABI generate durante la compilazione vengono quindi importate nella Sandbox per registrare le interfacce dei contratti ed esporne i metodi come API applicative.

Attraverso la Sandbox sono state inoltre configurate:

- interfacce per l'invocazione delle funzioni dei contratti,
- token pool per il token fungibile LuxuryCoin (ERC-20),
- token pool per gli asset NFT WatchNFT (ERC-721),
- listener per la ricezione degli eventi blockchain.

Il frontend esegue direttamente chiamate alle API REST di FireFly solo per operazioni di lettura. Le operazioni on-chain che richiedono firma (write/invoke) vengono inoltrate a un proxy off-chain che valida la sessione e invoca FireFly server-to-server.

4.5 Smart Contract

L'infrastruttura on-chain della piattaforma è implementata mediante una suite di Smart Contract sviluppati in *Solidity* (versione ^0.8.20), utilizzando le librerie *OpenZeppelin* per garantire aderenza agli standard e buone pratiche di sicurezza.

Di seguito vengono analizzati i tre contratti principali che costituiscono il core del sistema.

4.5.1 WatchNFT

WatchNFT è il contratto ERC721 che rappresenta l'orologio come NFT. Il suo compito è gestire la vita dell'asset (creazione, ruoli e certificazione), mentre la vendita è delegata al marketplace. Il contratto integra Ownable per le operazioni amministrative e un meccanismo di pausa tramite EmergencyStop, così da poter bloccare le funzioni operative in caso di incidente.

La produzione degli NFT è controllata da un indirizzo immutabile factory, che identifica il Producer: solo questo indirizzo può creare nuovi token. I tokenId sono generati in modo incrementale tramite un contatore (nextId), garantendo unicità e tracciabilità dell'emissione. I

token vengono emessi verso il Producer, che poi li immette sul mercato tramite il marketplace.

Il contratto gestisce i rivenditori con due livelli: reseller identifica l'appartenenza al ruolo, mentre activeReseller rappresenta l'abilitazione operativa corrente. Questa separazione consente di disabilitare un Reseller senza perdere l'informazione che quell'indirizzo risulti comunque classificato come reseller, utile per controlli e policy nel marketplace.

La certificazione è una proprietà associata al token e può essere impostata una sola volta da un reseller autorizzato che possiede l'NFT. La logica impone quindi che la certificazione sia legata sia al ruolo sia alla proprietà effettiva del bene, e che non sia ripetibile.

Le operazioni amministrative principali sono: gestione reseller e attivazione/disattivazione della pausa. Le operazioni operative (produzione e certificazione) sono bloccabili quando il contratto è in stato "paused", secondo l'approccio standard di emergenza per ridurre l'impatto di bug o attacchi.

4.5.2 WatchMarket

WatchMarket è il contratto che gestisce la compravendita dei WatchNFT. Mantiene un registro on-chain delle inserzioni e consente tre operazioni: creare un annuncio di vendita, modificarlo/annullarlo, acquistare.

Il mercato distingue due tipologie di vendita. La vendita primaria è la prima immissione sul mercato: può essere avviata solo dal Producer e può essere acquistata solo da Reseller autorizzati. La vendita secondaria è la rivendita: può essere avviata solo da Reseller autorizzati che possiedono l'NFT, richiede che l'orologio sia certificato, e può essere acquistata solo da Consumer.

Per mettere in vendita un orologio, il proprietario deve autorizzare il marketplace a trasferire i propri NFT. L' autorizzazione può essere rilasciata come approvazione globale, che viene richiesta una sola volta e rimane valida per tutte le future inserzioni finchè non viene revocata. In alternativa, è possibile usare un'approvazione per singolo token. Senza una di queste

autorizzazioni, l'inserzione non può essere creata, perché il marketplace non è abilitato a trasferire l'NFT. Dopo l'autorizzazione, quando un orologio viene messo in vendita, l'NFT viene trasferito al marketplace e rimane custodito lì fino a quando viene acquistato oppure l'inserzione viene annullata; in caso di annullamento l'NFT torna al venditore.

L'acquisto procede solo se esiste un'inserzione attiva per quel tokenId, se l'NFT è effettivamente custodito dal marketplace (quindi può essere consegnato), e se le regole di ammissibilità della vendita sono rispettate (primaria o secondaria, includendo certificazione e vincoli su buyer/seller). Il pagamento avviene tramite token ERC20 (LuxuryCoin) e richiede che il compratore abbia autorizzato il marketplace a spendere almeno l'importo del prezzo tramite allowance; se allowance o saldo non sono sufficienti, il trasferimento fallisce e l'acquisto va in revert. Dopo l'incasso, l'NFT viene trasferito all'acquirente e l'importo dovuto al venditore viene registrato come credito interno, da riscuotere successivamente tramite una chiamata di prelievo.

4.5.3 LuxuryCoin

LuxuryCoin (LUX) è il token fungibile standard ERC20 utilizzato come valuta di pagamento nell'ecosistema._Tutte le transazioni economiche tra gli attori del sistema, sia nelle vendite primarie sia in quelle secondarie, vengono regolate tramite questo token, evitando l'uso diretto della valuta nativa della rete Ethereum.

La fornitura è fissa e definita una sola volta al deploy: 1.000.000 LUX. In fase di creazione il contratto distribuisce una quota iniziale di 10.000 LUX a un indirizzo Reseller e 10.000 LUX a un indirizzo Consumer (se forniti e diversi da zero); il resto della supply viene assegnato al soggetto che effettua il deploy (Producer).

4.5.4 Design Pattern implementati

Per garantire la massima sicurezza dei fondi e la stabilità del sistema, sono stati adottati specifici pattern.

4.5.4.1 Behavioral Patterns

- *Guard Check*: prima di eseguire qualsiasi operazione che modifica lo stato vengono verificate in modo esplicito le precondizioni di correttezza (ruoli autorizzati, stato operativo, coerenza dell'inserzione e disponibilità dell'NFT, requisiti di certificazione e autorizzazioni di pagamento). In caso di mancato soddisfacimento anche di una sola condizione, l'esecuzione viene interrotta e la transazione va in revert, garantendo l'assenza di aggiornamenti parziali dello stato.

4.5.4.2 Security Patterns

- *Access Restriction*: L'indirizzo amministratore (che nel progetto coincide con Producer/Owner/Factory) è l'unico abilitato alle operazioni di governance e alle azioni riservate al Producer, mentre i Reseller autorizzati sono gli unici a poter effettuare le operazioni di certificazione e alla vendita sul mercato secondario; i Consumer possono esclusivamente acquistare nel mercato secondario. Nel marketplace, inoltre, le operazioni su un annuncio (modifica/cancellazione) sono consentite solo al soggetto che lo ha creato, impedendo interventi da parte di terzi non autorizzati.

- *Check-Effects-Interactions:* utilizzato per ridurre il rischio di re-entrancy e, più in generale, di stati incoerenti quando una funzione effettua chiamate esterne. Applicato soprattutto nelle operazioni del marketplace e nei prelievi dei crediti: prima vengono validate le condizioni dell'operazione, poi viene aggiornato lo stato interno, e solo in chiusura avvengono i trasferimenti verso contratti esterni (NFT e token ERC20).

- *Secure Ether Transfer*: I trasferimenti di token avvengono tramite le utility SafeERC20, che gestiscono in modo robusto token non standard e riducono il rischio di esiti ambigui nelle operazioni di trasferimento. I trasferimenti vengono eseguiti solo dopo la validazione dei prerequisiti e l'aggiornamento coerente dello stato interno, con revert in caso di errore.

- *Pull over Push*: Il marketplace non invia automaticamente i proventi al venditore nel momento dell'acquisto. Registra invece un credito a favore del venditore, che viene incassato in un secondo momento con una chiamata separata di prelievo. Questo

evita che un pagamento “in uscita” possa rendere fragile l’acquisto (ad esempio per errori/condizioni del destinatario) e isola meglio la parte di pagamento.

- *EmergencyStop:* Sia WatchNFT sia WatchMarket integrano una modalità di “pausa” per sospendere le operazioni critiche. La scelta è pragmatica: in caso di bug o anomalia si congela l’operatività, mantenendo comunque disponibili azioni di recupero; in particolare, nel marketplace le operazioni di ritiro crediti e di annullamento inserzione non vengono bloccate, così gli utenti non restano “incastrati” con fondi o asset.

4.5.4.3 Economic Pattern

- *Tight Variable Packing*: usato per ridurre il gas occupando meno *storage slot*: variabili di dimensione ridotta vengono affiancate nello stesso slot quando possibile. Nei contratti la scelta e l’ordine dei tipi favoriscono questo impacchettamento, diminuendo letture/scritture su storage e quindi i costi di esecuzione.

4.6 Livello applicativo off-chain

4.6.1 Autenticazione e sessione

Il sistema adotta un accesso senza password basato sul possesso di un wallet. L'utente dimostra la propria identità firmando un messaggio generato dal sistema; la firma viene verificata lato server e, in caso di esito positivo, viene avviata una sessione.

Per ridurre il rischio di riutilizzo della stessa firma, il sistema utilizza un valore casuale monouso (nonce) associato all'indirizzo dell'utente: il client richiede il nonce, lo firma con il wallet e invia la firma al servizio di autenticazione, che ne verifica validità e coerenza.

Dopo il login, la sessione non viene gestita tramite credenziali nel client, ma tramite cookie impostati dal server. In particolare, viene usato un token a breve durata per autenticare le richieste correnti e un token a durata maggiore per rinnovare automaticamente la sessione senza richiedere un nuovo login. I cookie principali non sono accessibili dagli script lato client, riducendo l'esposizione in caso di codice malevolo nella pagina.

Le operazioni di rinnovo e chiusura della sessione adottano un controllo aggiuntivo contro invii indesiderati da contesti esterni: il server accetta tali richieste solo se il client dimostra di essere lo stesso che ha ricevuto la sessione (controllo basato su cookie e valore di conferma inviato in intestazione).

Le rotte che richiedono autenticazione verificano la sessione ad ogni richiesta. Quando l'utente è autenticato, l'identità usata dal sistema (indirizzo wallet) è ricavata dalla sessione e non da parametri liberamente manipolabili nelle richieste. Questo principio viene applicato anche alle invocazioni verso i servizi di integrazione (es. proxy verso FireFly), impedendo a un client di impersonare indirizzi diversi dal proprio. Il routing verso i nodi (Producer/Reseller/Consumer) è un dettaglio applicativo e non costituisce di per sé una garanzia di autorizzazione, che è demandata alle regole on-chain.

4.6.2 Interfaccia utente

L'interfaccia utente è realizzata come applicazione web single-page sviluppata in React e compilata tramite Vite.

Essa rappresenta il punto di accesso al sistema e consente agli utenti di interagire con i servizi applicativi e con le funzionalità blockchain.

MetaMask viene utilizzato esclusivamente come strumento di autenticazione.

Durante il login il client richiede la firma di un nonce per dimostrare il possesso dell'indirizzo wallet. Non viene invece utilizzato per firmare o inviare transazioni on-chain.

Le operazioni applicative sono suddivise in due categorie.

Le funzionalità di autenticazione e gestione della sessione vengono invocate tramite richieste HTTP verso il servizio off-chain.

Le operazioni che coinvolgono gli smart contract vengono eseguite tramite chiamate alle API REST esposte da FireFly, che si occupa internamente della gestione e dell'invio delle transazioni alla rete Ethereum.

Il client mantiene localmente lo stato dell'utente, come indirizzo, ruolo e saldi, aggiornandolo tramite interrogazioni periodiche ai servizi di integrazione.

Gli aggiornamenti generati dagli eventi on-chain vengono inoltre propagati al client per sincronizzare automaticamente l'interfaccia con lo stato della blockchain.

4.6.3 Sincronizzazione event-driven

Per mantenere l'interfaccia allineata allo stato della blockchain, il sistema utilizza un meccanismo di notifica basato sugli eventi generati dagli smart contract.

Gli eventi on-chain vengono intercettati dai listener configurati in FireFly, che inviano una richiesta HTTP verso un endpoint webhook del servizio off-chain.

Alla ricezione della notifica, il servizio:

- verifica l'autenticità della chiamata tramite un secret condiviso,
- risponde immediatamente con un acknowledgement,
- invia un segnale di aggiornamento ai client connessi tramite socket.

Il contenuto dell'evento non viene utilizzato direttamente dal frontend. La notifica viene impiegata esclusivamente come trigger di refresh, inducendo il client a interrogare nuovamente i servizi di integrazione per ottenere lo stato aggiornato.

Questo approccio riduce il polling continuo, semplifica la gestione dei dati e garantisce coerenza tra interfaccia e blockchain senza replicare logica applicativa lato server.

5. Use Cases

UC_01: Autenticazione utente

ID/Nome	UC_01: Autenticazione utente
Attore	Producer, Reseller, Consumer
Entry Condition	• L'utente possiede un wallet associato ad un indirizzo blockchain.
Flusso	1. L'utente richiede l'accesso al sistema. 2. Il sistema genera un messaggio univoco temporaneo, associato all'indirizzo dell'utente. 3. L'utente firma il valore utilizzando il proprio wallet. 4. L'utente invia la firma al sistema. 5. Il sistema verifica la validità della firma e l'associazione con l'indirizzo dell'utente. 6. Il sistema autentica l'utente e avvia una sessione valida.
Exit Condition (on success)	L'utente risulta autenticato e può accedere alle funzionalità consentite dal proprio ruolo.
Exit Condition (on failure)	L'utente non ha accesso al sistema.
Flusso alternativo	• Valore scaduto: il sistema richiede una nuova autenticazione. • Firma non valida: il sistema rifiuta l'autenticazione. • Utente non autorizzato: il sistema limita o nega l'accesso alle funzionalità.

UC_02: Minting Orologio

ID/Nome	UC_02: Minting
Attore	Producer
Entry Condition	• Il Producer è autenticato. • Il sistema non è in stato di emergenza per le operazioni di creazione.
Flusso	1. Il Producer richiede la creazione di un nuovo orologio al sistema, attraverso l'apposito pulsante. 2. Il sistema, dopo aver richiesto conferma all'utente, valida l'autorizzazione del Producer per l'operazione di creazione. 3. Il sistema crea l'orologio assegnandogli un identificativo univoco. 4. Il sistema conferma l'avvenuta creazione.
Exit Condition (on success)	Un nuovo orologio digitale viene creato e registrato sulla blockchain. L'orologio risulta di proprietà del Producer ed è disponibile nel suo inventario.
Exit Condition (on failure)	Il sistema rifiuta l'operazione.
Flusso alternativo	Se le operazioni di creazione sono sospese, il sistema non permette di utilizzare il pulsante di mint.

UC_03: Gestione operatività dei Reseller

ID/Nome	UC_03: Gestione Reseller
Attore	Producer
Entry Condition	• Il Producer è autenticato.
Flusso	1. Il Producer richiede attraverso l'apposita interfaccia, la gestione di un Reseller, indicando l'indirizzo wallet e l'operazione desiderata (abilitazione o revoca). 2. Il sistema richiede conferma. 3. Il sistema aggiorna lo stato del Reseller. 4. Il sistema conferma l'avvenuto aggiornamento.
Exit Condition (on success)	Il Reseller ottiene/perde i permessi.
Exit Condition (on failure)	Aggiornamento non riuscito. Lo stato del Reseller rimane invariato.
Flusso alternativo	//

UC_04: Vendita e acquisto sul mercato primario

ID/Nome	UC_04: Mercato primario
Attore	Producer, Reseller
Entry Condition	• Il Producer è autenticato ed è proprietario di uno o più orologi. • Il Reseller è autenticato ed è abilitato come Reseller attivo. • Le operazioni di mercato primario non sono sospese per emergenza del sistema.
Flusso	1. Il Producer seleziona un orologio dal proprio inventario e richiede di metterlo in vendita sul mercato primario indicando un prezzo. 2. Il sistema richiede conferma all'utente. 3. Il sistema verifica che l'orologio appartenga al Producer e che la vendita sia consentita. 4. Il sistema rende l'orologio disponibile sul mercato primario. 5. Il Reseller seleziona un orologio dal mercato primario e richiede l'acquisto. 6. Il sistema richiede conferma all'utente. 7. Il sistema verifica che il Reseller sia autorizzato ad acquistare sul mercato primario. 8. Il sistema registra l'acquisto e trasferisce la proprietà dell'orologio al Reseller.
Exit Condition (on success)	• L'orologio viene trasferito dal Producer al Reseller. • Il Producer matura un credito corrispondente al prezzo di vendita. • L'orologio non risulta più essere sul mercato primario.
Exit Condition (on failure)	• Il Producer ha inserito un prezzo non valido. Se il prezzo è <= 0, il sistema impedisce il listing. • Errore durante la transazione: il sistema non completa il trasferimento e mantiene lo stato precedente. • Utente non autorizzato: il sistema impedisce l'operazione.
Flusso alternativo	Se il blocco del mercato è attivo, il sistema non permette la compravendita di orologi

UC_05: Certificazione orologio

ID/Nome	UC_05: Certificazione
Attore	Reseller
Entry Condition	• Il Reseller è autenticato. • Il Reseller è autorizzato ad operare come tale. • Il Reseller è proprietario dell'orologio da certificare. • Le operazioni di certificazione non sono sospese per lo stato di emergenza.
Flusso	1. Il Reseller seleziona un orologio dal proprio inventario. 2. Il Reseller richiede la certificazione dell'orologio. 3. Il sistema richiede conferma al Reseller. 4. Il sistema verifica che il Reseller sia autorizzato e proprietario dell'orologio. 5. Il sistema registra la certificazione dell'orologio. 6. Il sistema conferma l'avvenuta certificazione.
Exit Condition (on success)	• L'orologio risulta certificato dal Reseller. • Lo stato di certificazione dell'orologio è aggiornato e visibile.
Exit Condition (on failure)	Il sistema rifiuta la richiesta di certificazione. Lo stato dell'orologio rimane invariato.
Flusso alternativo	• Operazioni di certificazioni sospese: il sistema segnala che la funzionalità non è disponibile. • Il Reseller non è abilitato: il sistema rifiuta l'operazione di certificazione.

UC_06: Vendita e acquisto sul mercato secondario

ID/Nome	UC_06: Mercato secondario
Attore	Reseller, Consumer
Entry Condition	• Il Reseller è autenticato ed è autorizzato ad operare come tale. • Il Reseller è proprietario di uno o più orologi certificati. • Il Consumer è autenticato. • Le operazioni di mercato non sono sospese per lo stato di emergenza del sistema.
Flusso	1. Il Reseller seleziona un orologio certificato dal proprio inventario e richiede di metterlo in vendita sul mercato secondario indicando un prezzo. 2. Il sistema richiede la conferma al Reseller. 3. Il sistema verifica che l'orologio sia certificato e di proprietà del Reseller. 4. Il sistema rende l'orologio disponibile sul mercato secondario. 5. Il Consumer seleziona un orologio dal mercato secondario e richiede l'acquisto. 6. Il sistema richiede la conferma al Consumer. 7. Il sistema verifica che l'operazione rispetti i requisiti di vendita. 8. Il sistema registra l'acquisto e trasferisce la proprietà dell'orologio al Consumer.
Exit Condition (on success)	• Un orologio viene trasferito dal Reseller al Consumer. • Il Reseller matura un credito corrispondente al prezzo di vendita. • L'orologio non risulta più essere sul mercato secondario.
Exit Condition (on failure)	• Il Reseller ha inserito un prezzo non valido. Se il prezzo è <= 0, il sistema impedisce il listing. • Errore durante la transazione: il sistema non completa il trasferimento e mantiene lo stato precedente. • Utente non autorizzato: il sistema impedisce l'operazione.
Flusso alternativo	Se il blocco del mercato è attivo, il sistema non permette la compravendita di orologi.

UC_07: Pagamenti e prelievo dei fondi

ID/Nome	UC_07: Pagamenti e prelievo fondi
Attore	Producer, Reseller
Entry Condition	• L'utente è autenticato. • L'utente ha maturato crediti a seguito di una o più vendite.
Flusso	1. L'utente richiede il prelievo dei fondi maturati. 2. Il sistema richiede la conferma all'utente. 3. Il sistema verifica che l'utente abbia un saldo disponibile. 4. Il sistema trasferisce i fondi all'utente. 5. Il sistema aggiorna il saldo dei crediti.
Exit Condition (on success)	• I fondi maturati vengono trasferiti all'utente. • Il saldo dei crediti dell'utente viene aggiornato.
Exit Condition (on failure)	• Nessun credito disponibile: il sistema non mostra il tasto per il prelievo. • Errore durante il trasferimento: il sistema non aggiorna il saldo e segnala il problema.
Flusso alternativo	//

UC_08: Aggiornamento in tempo reale tramite eventi Blockchain

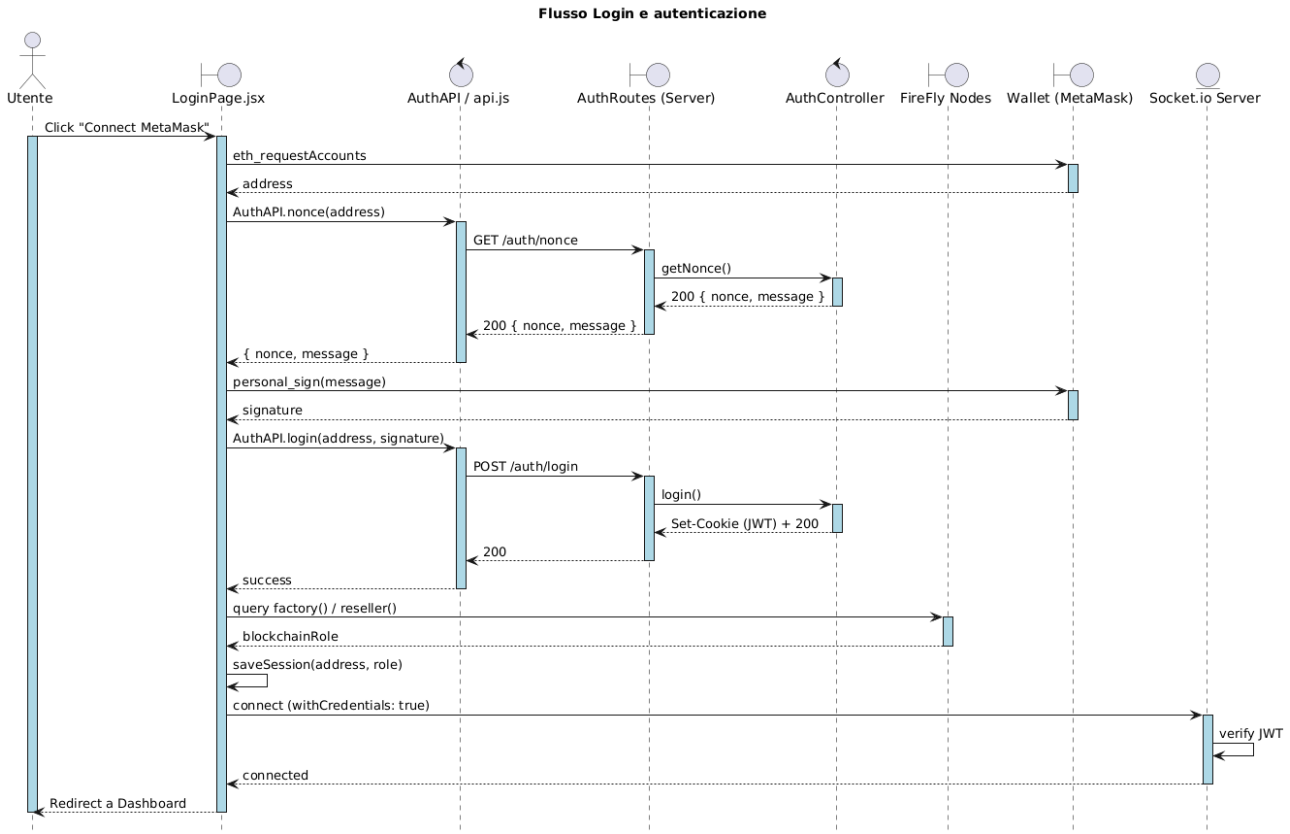
ID/Nome	UC_08: Aggiornamenti in tempo reale
Attore	Producer, Reseller, Consumer
Entry Condition	• Il sistema è in esecuzione e in ascolto degli eventi. • L'utente è autenticato e sta visualizzando una schermata che mostra dati aggiornabili (market, profilo, ecc.).
Flusso	1. L'utente compie un'operazione che può modificare lo stato (es. acquisto, vendita, certificazione). 2. Il sistema rileva il cambiamento di stato. 3. Il sistema notifica agli utenti connessi che è disponibile un aggiornamento. 4. Il sistema aggiorna automaticamente i dati sul dispositivo dell'utente recuperando lo stato più recente. 5. L'utente visualizza lo stato aggiornato.
Exit Condition (on success)	• Lo stato visualizzato dall'Utente è aggiornato.
Exit Condition (on failure)	Il sistema ignora l'evento e non invia notifiche. L'utente non visualizza gli aggiornamenti in tempo reale.
Flusso alternativo	• Evento non valido o incompleto: il sistema ignora l'evento e non invia notifiche. • Sorgente non autorizzata: il sistema rifiuta l'evento e non invia notifiche. • Errore durante l'elaborazione: il sistema non notifica e registra l'errore per diagnosi. • Utente non connesso: l'utente non riceve l'aggiornamento in tempo reale, ma visualizzerà lo stato aggiornato alla successiva consultazione.

UC_09: Gestione Emergenza e Sospensione operazioni

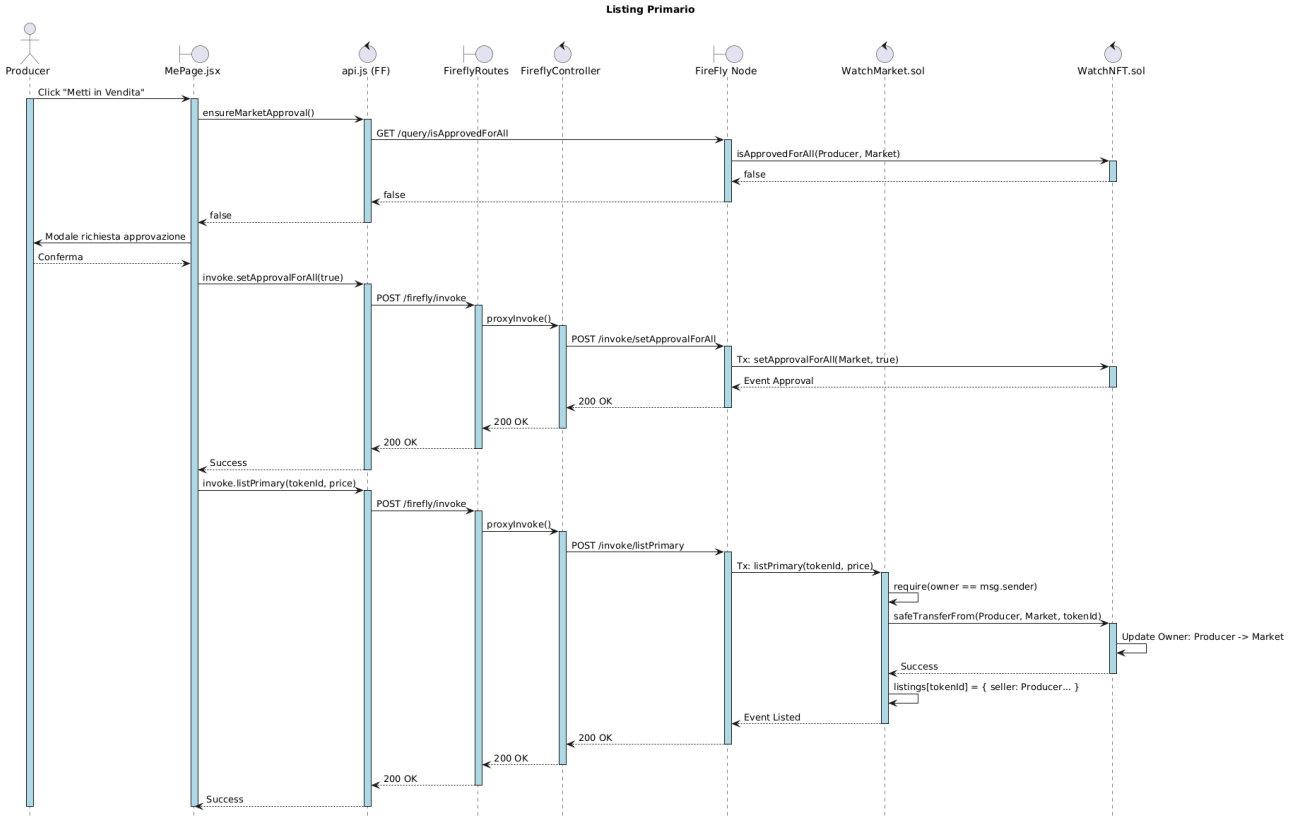
ID/Nome	UC_09: Sospensione operazioni
Attore	Producer
Entry Condition	• Il Producer è autenticato.
Flusso	1. Il Producer accede alla funzionalità di gestione emergenza. 2. Il Producer seleziona l'azione desiderata. 3. Il sistema richiede conferma dell'operazione. 4. Il sistema aggiorna lo stato operativo del sistema, abilitando o bloccando le operazioni interessate. 5. Il sistema conferma l'avvenuto aggiornamento.
Exit Condition (on success)	• Se la sospensione è attivata, le operazioni di creazione e/o compravendita risultano temporaneamente non disponibili. • Se la sospensione è disattivata, le operazioni tornano disponibili secondo le regole di autorizzazione previste.
Exit Condition (on failure)	Lo stato del sistema rimane invariato e l'operazione non viene applicata.
Flusso alternativo	//

6. Sequence Diagrams

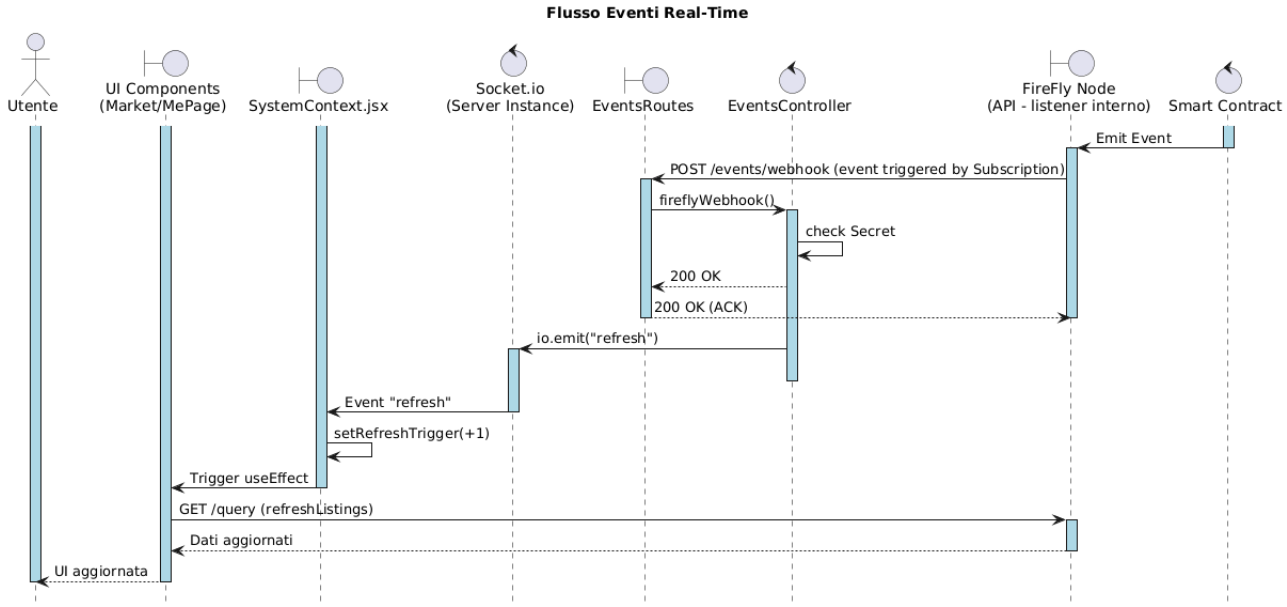
6.1 SD_01: Autenticazione utente



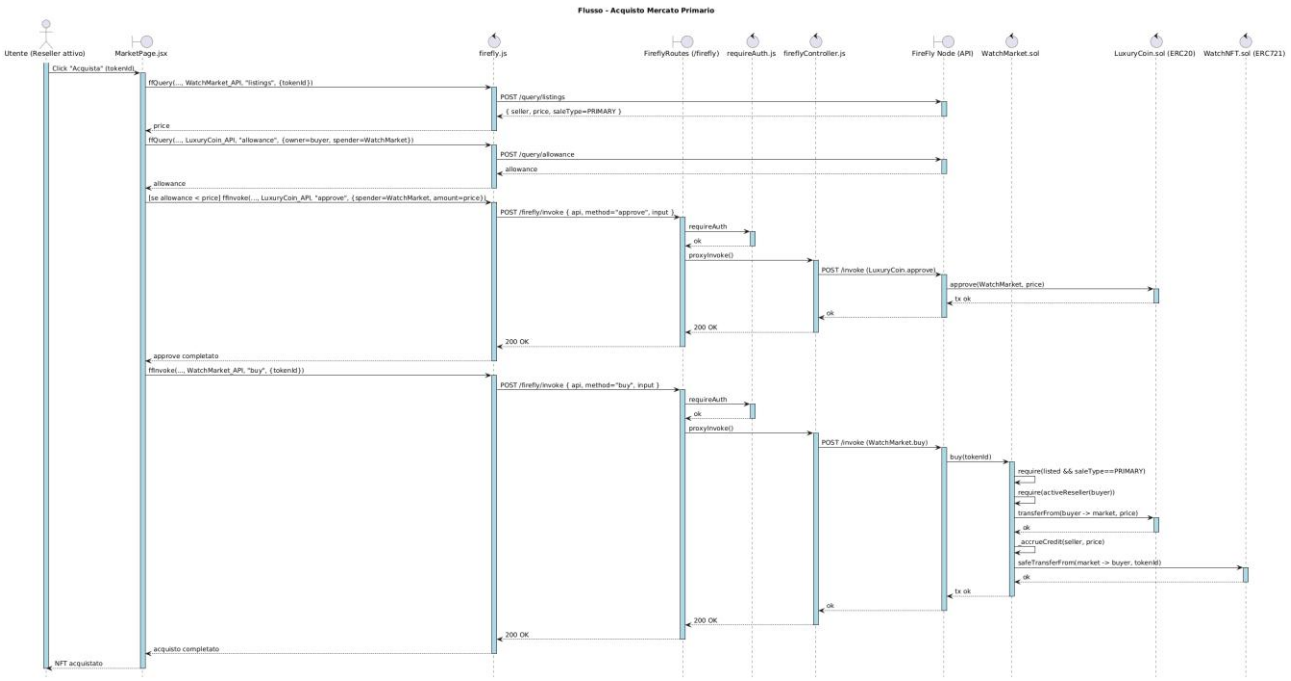
6.2 SD_02: Listing primario



6.3 SD_03: Aggiornamento in tempo reale tramite eventi



6.4 SD_04: Acquisto primario



7. Analisi delle minacce, Misuse e Abuse Cases

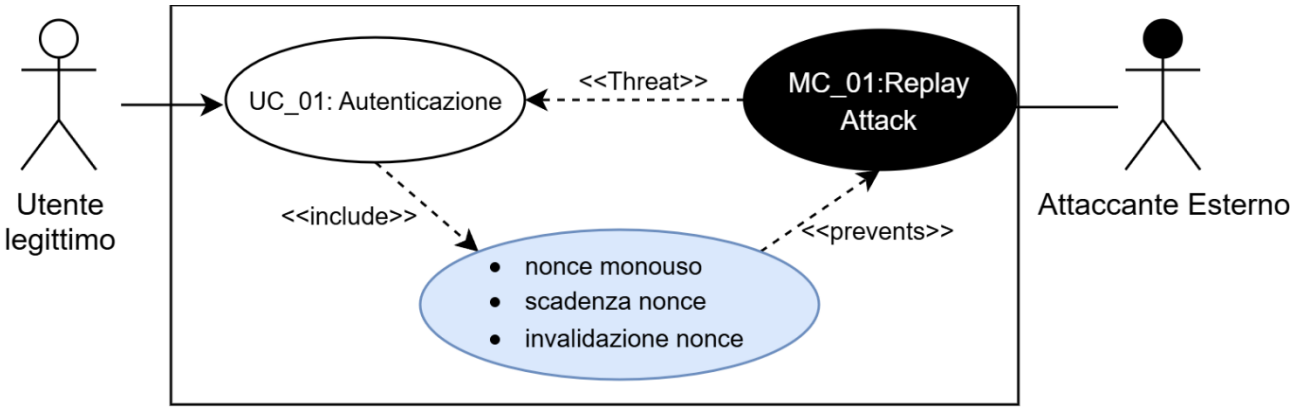
Questo capitolo analizza il sistema dal punto di vista dell'attaccante, modellando i principali scenari di uso malevolo o scorretto delle funzionalità applicative. L'obiettivo non è descrivere nuove funzionalità, ma valutare come le funzionalità esistenti possano essere abusate e quali contromisure progettuali e implementative siano state adottate.

7.1 Misuse Case

I Misuse Case rappresentano azioni indesiderate o malevole in cui un attore, anche se autenticato, tenta di eseguire operazioni non autorizzate, violando i meccanismi di sicurezza o le proprietà del sistema.

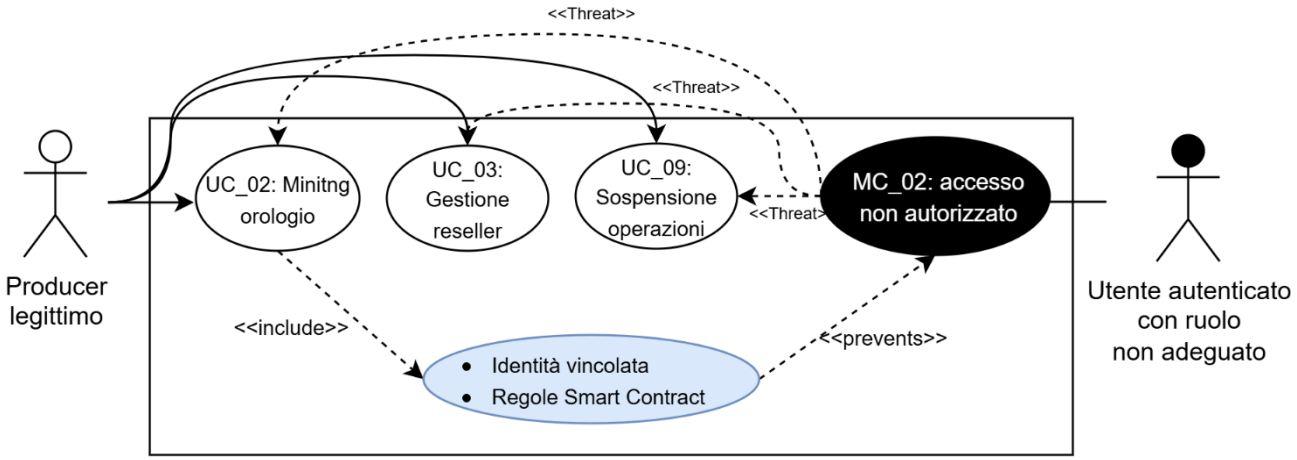
MC_01: Riutilizzo della firma di autenticazione (Replay Attack)

ID/Nome	MC_01: Replay Attack
Scenario	Un attaccante tenta di autenticarsi riutilizzando una firma crittografica precedentemente intercettata.
Misactor	Attaccante esterno
Obiettivo	Accedere al sistema impersonando un utente legittimo.
Funzionalità minacciata	UC_01: Autenticazione utente
Contromisure	• Utilizzo di nonce monouso associati all'indirizzo wallet • Scadenza temporale del nonce • Invalidazione del nonce dopo il primo utilizzo



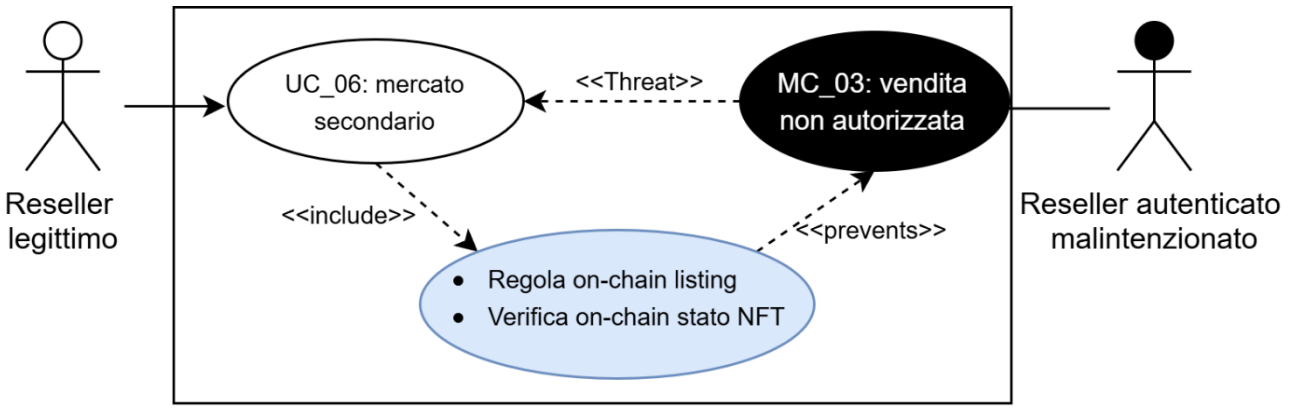
MC_02: Accesso a funzionalità non autorizzate per ruolo

ID/Nome	MC_02: Accesso non autorizzato
Scenario	Un utente autenticato tenta di accedere direttamente a operazioni riservate ad altri ruoli, ad esempio invocando manualmente endpoint o funzioni di smart contract non autorizzate.
Misactor	Utente autenticato con ruolo non adeguato
Obiettivo	Ottenere accesso a operazioni non consentite dal modello di dominio.
Funzionalità minacciate	• UC_02: Minting Orologio • UC_03: Gestione Reseller UC_09: Sospensione operazioni
Contromisure	• Identità vincolata: firma sempre associata all'indirizzo autenticato. • Autorizzazione finale on-chain: operazioni non ammesse falliscono.



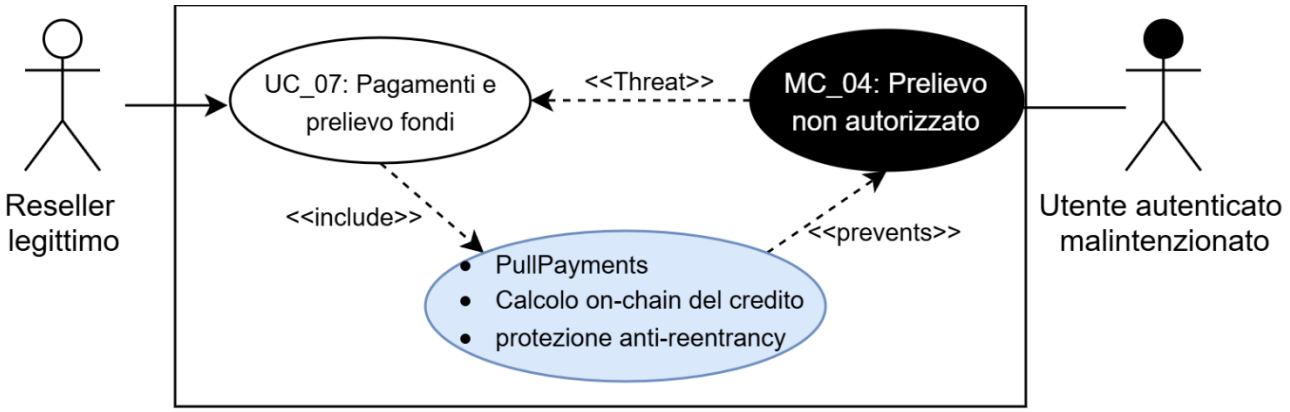
MC_03: Vendita di orologi non certificati

ID/Nome	MC_03: Vendita non autorizzata
Scenario	Un utente tenta di listare sul mercato secondario un orologio non certificato, aggirando i vincoli del sistema.
Misactor	Reseller autenticato malintenzionato
Obiettivo	Vendere un asset non conforme alle regole del sistema.
Funzionalità minacciata	UC_06: Mercato secondario
Contromisure	• Verifica on-chain dello stato di certificazione dell'NFT • Regola on-chain: listing consentito solo per NFT certificati.



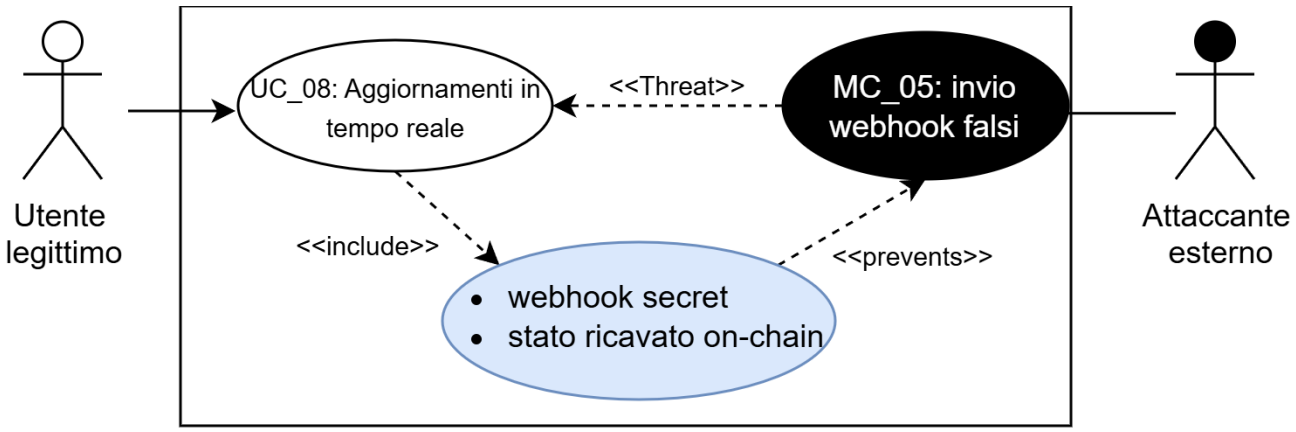
MC_04: Tentativo di prelievo improprio dei fondi

ID/Nome	MC_04: Prelievo non autorizzato
Scenario	Un utente tenta di prelevare fondi non maturati o non di sua competenza.
Misactor	Utente autenticato malintenzionato
Obiettivo	Ottenere un vantaggio economico indebito.
Funzionalità minacciata	UC_07: Pagamenti e prelievo dei fondi
Contromisure	• Meccanismo PullPayments con saldo interno per indirizzo • Calcolo on-chain dei crediti maturati • Protezione contro la reentrancy



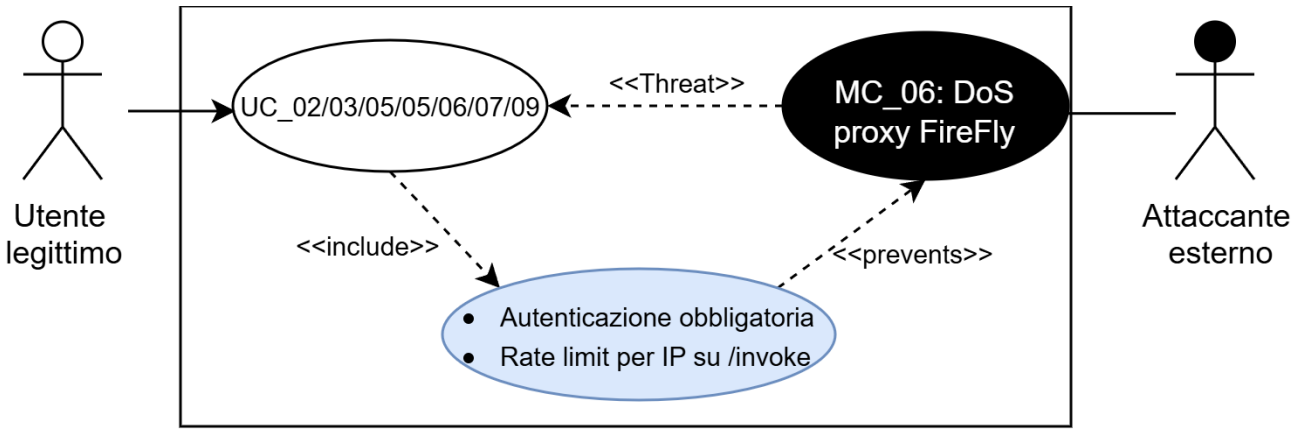
MC_05: Invio di webhook blockchain falsificati

ID/Nome	MC_05: invio webhook falsificati
Scenario	Un attaccante tenta di inviare richieste false all'endpoint webhook per generare notifiche di aggiornamento non legittime.
Misactor	Attaccante esterno
Obiettivo	Generare refresh/notifiche di aggiornamento non legittime (disturbo o spam).
Funzionalità minacciata	UC_08: Aggiornamenti in tempo reale
Contromisure	• webhook protetto da secret (richieste non autorizzate rifiutate) • Verità on-chain: lo stato mostrato è ricavato da dati on-chain (il webhook non può alterarlo)



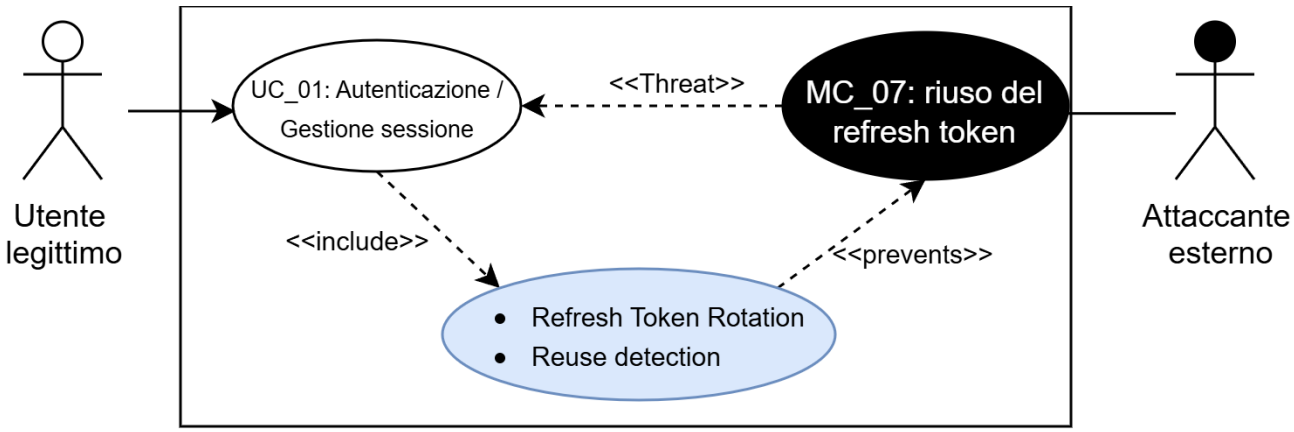
MC_06: DoS dell'endpoint di invocazione

ID/Nome	MC_06: DoS proxy FireFly
Scenario	Un attaccante invia molte richieste a /firefly/invoke per saturare server/FireFly e generare transazioni fallite o carico eccessivo.
Misactor	Attaccante esterno
Obiettivo	Degradare il servizio e disturbare gli utenti.
Funzionalità minacciata	Tutti gli UC che richiedono /firefly/invoke: • UC_02: Minting • UC_03: Gestione Reseller • UC_04: Mercato primario • UC_05: Certificazione • UC_06: Mercato secondario • UC_07: Prelievo dei fondi maturati • UC_09: Sospensione operazioni
Contromisure	• Rate limit per IP sull'endpoint di invocazione • Autenticazione obbligatoria per invocazioni



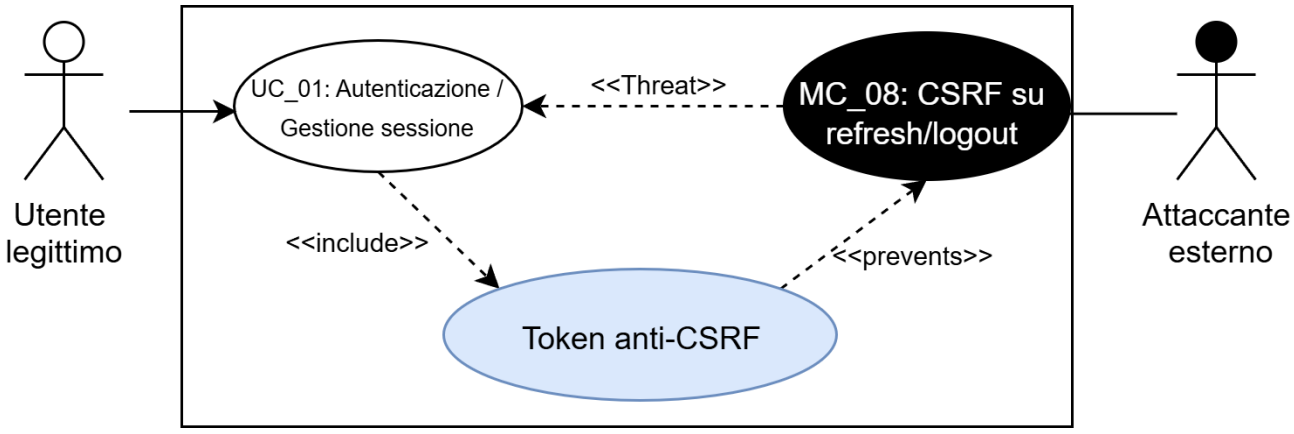
MC_07: Riuso del refresh token

ID/Nome	MC_07: Riuso del Refresh Token
Scenario	Un attaccante ruba un Refresh Token e tenta di riutilizzarlo per mantenere o ripristinare la sessione.
Misactor	Attaccante esterno (in possesso del Refresh Token)
Obiettivo	Prolungare l'accesso non autorizzato al sistema
Funzionalità minacciata	• UC_01: Autenticazione • Gestione della sessione
Contromisure	• Refresh Token Rotation: il token valido è solo l'ultimo emesso • Reuse detection: se un refresh "vecchio" viene riutilizzato, la sessione viene revocata/bloccata lato server



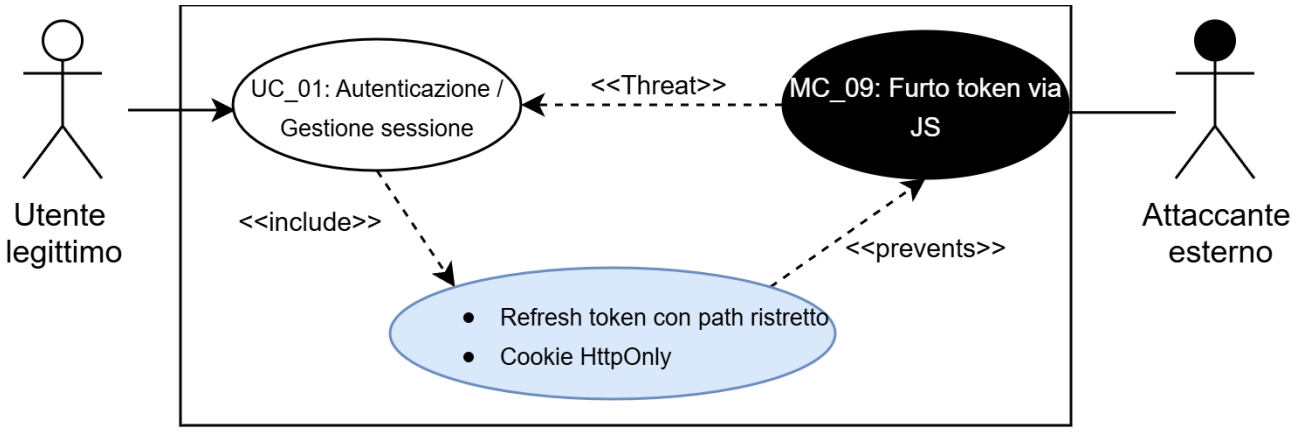
MC_08: Cross-Site Request Forgery su refresh/logout

ID/Nome	MC_08: CSRF su gestione sessione (refresh/logout)
Scenario	Un sito terzo induce il browser della vittima a inviare richieste di refresh/logout sfruttando i cookie di sessione.
Misactor	Attaccante esterno (sito terzo)
Obiettivo	Forzare azioni sulla sessione della vittima senza consenso
Funzionalità minacciata	• Gestione della sessione • UC_01: Autenticazione
Contromisure	• Token anti-CSRF: il server Accetta refresh/logout solo se la richiesta include un token CSRF valido (cookie CSRF + header con lo stesso valore)



MC_09: Furto token via JavaScript (XSS)

ID/Nome	MC_09: Furto token via JavaScript
Scenario	Uno script malevolo (XSS) nella pagina tenta di leggere i token di sessione dal browser per impersonare l'utente.
Misactor	Attaccante esterno (Iniezione script)
Obiettivo	Ottenere token e utilizzare la sessione della vittima
Funzionalità minacciata	• Gestione della sessione • UC_01: Autenticazione
Contromisure	• Token sensibili in cookie HttpOnly (quindi non leggibili da JavaScript) • Refresh token con path ristretto /auth/refresh per ridurre l'esposizione

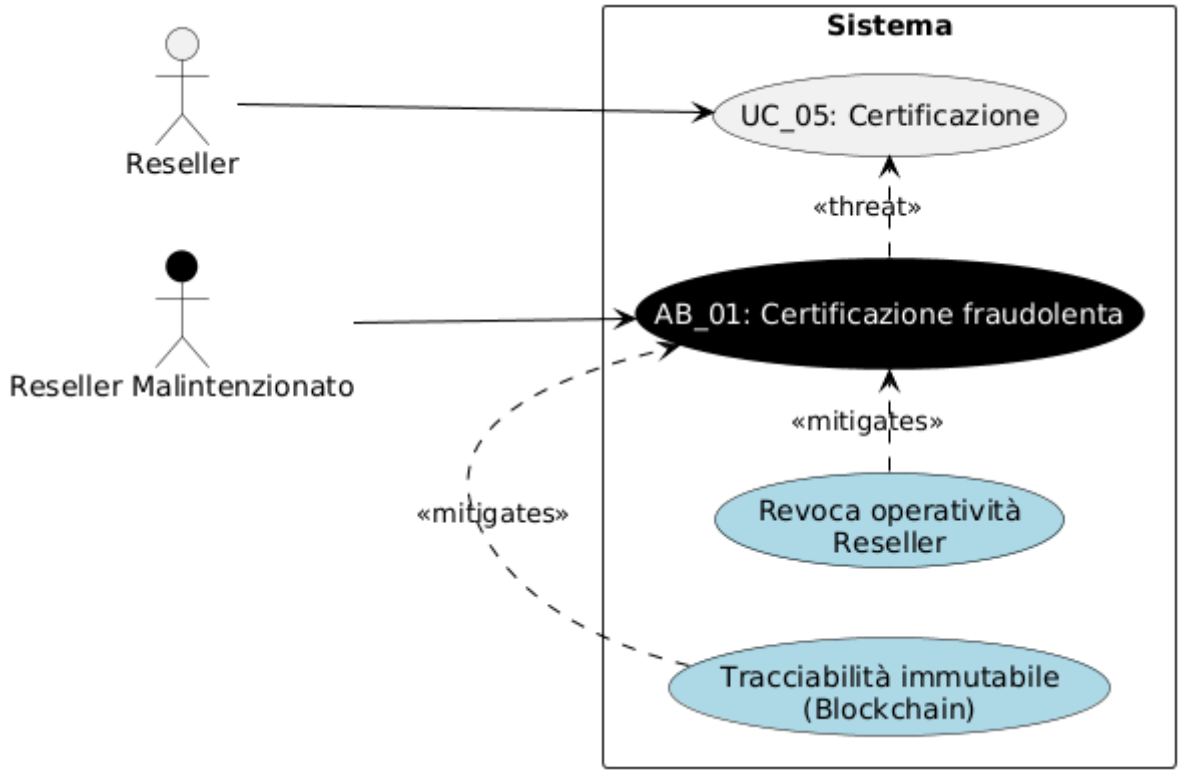


7.2 Abuse Case

Gli Abuse Case descrivono scenari in cui un attore legittimo del sistema, correttamente autenticato e autorizzato, sfrutta le funzionalità disponibili in modo opportunistico o scorretto, compromettendo l'equità o l'affidabilità del sistema senza violarne direttamente i meccanismi di sicurezza.

AB_01: Abuso delle funzionalità di certificazione

ID/Nome	AB_01: Abuso certificazione
Scenario	Un reseller rilascia certificazioni senza aver verificato realmente l'autenticità dell'orologio (certificazione fraudolenta o superficiale).
Actor	Reseller malintenzionato
Obiettivo	Massimizzare profitto compromettendo l'affidabilità del processo di certificazione.
Funzionalità abusata	UC_05: Certificazione
Contromisure	• Revoca operatività Reseller • Tracciabilità immutabile delle operazioni di certificazione



AB_02: Abuso del potere di governance

ID/Nome	AB_02: Abuso potere governance
Scenario	Il Producer utilizza meccanismi di controllo operativo (abilitazione reseller, blocchi operativi) in modo opportunistico per condizionare il funzionamento del mercato.
Actor	Producer malintenzionato
Obiettivo	Ottenere un vantaggio competitivo o influenzare la filiera.
Funzionalità abusata	UC_04: Mercato primario
Contromisure	• Tracciabilità/verificabilità delle azioni di governance