

# WATCHCHAIN

**Mario Lezzi**

matr. 0522501840

Sicurezza dei Dati 25/26



<https://github.com/MarioLezzi92/Watchchain>

# Problema

## Contraffazione

Il mercato secondario degli orologi è invaso da falsi di alta qualità e orologi con parti interne sostituite e non originali.

---

## Scarsa tracciabilità

Chi acquista non ha strumenti per verificare la storia dell'orologio e deve fidarsi ciecamente della parola del venditore.

---

## Limiti del cartaceo

I documenti cartacei di autenticità si deteriorano, si perdono o possono essere facilmente falsificati, rendendo impossibile provare la proprietà.

# Obiettivi



Certificazione dell'autenticità  
tramite identità digitale  
(NFT).



Tracciabilità completa e  
immutabile di provenienza e  
passaggi di proprietà.



Riduzione di frodi e asimmetrie  
informative nel mercato  
secondario.



Automazione e sicurezza delle  
transazioni tramite smart contract.



# Requisiti non funzionali

## 1. RNF Sicurezza

ID	REQUISITO	DESCRIZIONE
RNF-S1	Autenticazione crittografica	Il sistema deve consentire l'accesso esclusivamente tramite firma crittografica del wallet dell'utente.
RNF-S2	Previsione 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 immunità 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. 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 malfunzionamento
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

### 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 anomalia 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

# Idea di soluzione

## 1. Filiera a ruoli

Producer → Reseller → Consumer, con permessi e responsabilità definite.

## 2. Identità digitale dell'orologio

Ogni orologio è rappresentato da un NFT che ne attesta identità e proprietà

## 3. Tracciabilità e trasparenza

Provenienza, certificazione e passaggi di mano registrati in modo immutabile on-chain.

## 4. Regole automatiche di scambio

Smart contract per vendite con escrow e pagamenti automatizzati

# Attori

## Producer

Crea l'NFT e lo vende sul mercato secondario ai Reseller.

---

## Reseller

Acquista l'orologio sul primario, lo certifica e lo rivende sul mercato secondario.

---

## Consumer

Acquista l'orologio sul mercato secondario.

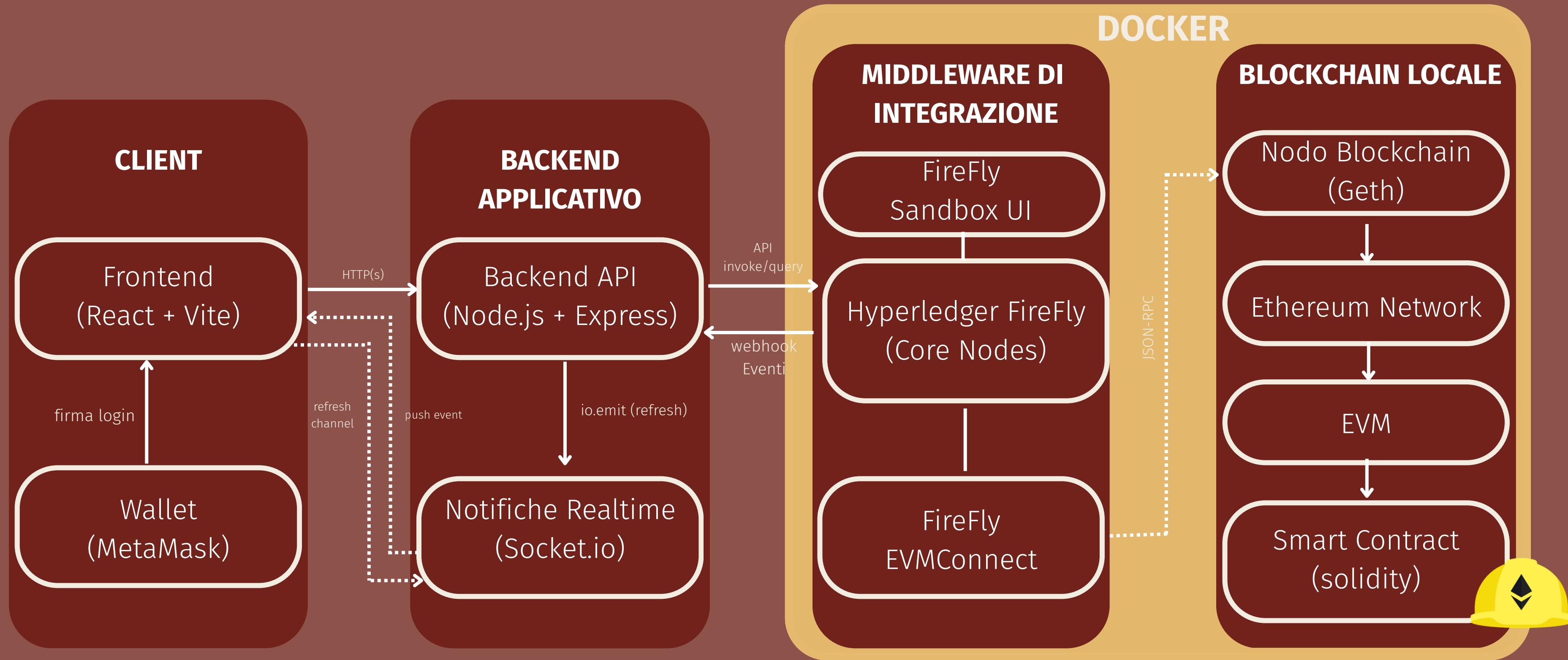
# Requisiti Funzionali (RF)

ID	Attore	Requisito
RF_1	Producer	Minting orologio
RF_2	Producer	Abilitazione Reseller
RF_3	Producer	Disabilitazione Reseller
RF_4	Producer	Listing Primario
RF_5	Producer	Blocco Factory
RF_6	Producer	Blocco Marketplace
RF_7	Producer / Reseller	Visualizzazione mercato primario

# Requisiti Funzionali (RF)

RF_8	Reseller	Acquisto primario
RF_9	Reseller	Certificazione
RF_10	Reseller	Listing secondario
RF_11	Producer/reseller	Annullamento listing
RF_12	Producer/reseller	Aggiorna prezzo
RF_13	Consumer	Visualizzazione mercato secondario
RF_14	Consumer	Acquisto finale
RF_15	Tutti	Visualizzazione inventario
RF_16	Producer/Reseller	Prelievo fondi

# Component Diagram



# Infrastruttura blockchain

1. Infrastruttura blockchain locale eseguita interamente in Docker, con un nodo Ethereum (Geth) e i servizi di integrazione necessari.
2. L'applicazione non comunica direttamente con Geth: interagisce con FireFly via API REST. Le operazioni di scrittura sono mediate dal backend (proxy + controlli).
3. Gli smart contract vengono compilati e testati con Hardhat e poi deployati sulla rete tramite API REST del nodo FireFly del Producer.
4. Dopo il deploy, i contratti sono eseguiti dalla EVM; frontend e backend restano esterni ai container e comunicano via HTTP.

# Hyperledger FireFly

Accesso alla blockchain  
via API REST di FireFly  
(no chiamate dirette  
JSON-RPC a Geth)

The screenshot shows the Swagger UI interface for the WatchNFT\_IFACE API. At the top, it displays the URL `http://127.0.0.1:5000/api/v1/namespaces/default/apis/WatchNFT_API/api/openapi.yaml`. Below this, the title "WatchNFT\_IFACE" is shown with version 1.1 and OAS 3.0 status. A "Servers" dropdown is set to `http://127.0.0.1:5000/api/v1/namespaces/default/apis/WatchNFT_API`. The main area lists several API endpoints:

- GET /interface
- POST /invoke/activeReseller
- POST /invoke/approve
- POST /invoke/balanceOf
- POST /invoke/certified
- POST /invoke/certify



# Hyperledger FireFly

Tre nodi FireFly, ciascuno associato ad un attore:  
Producer, Reseller,  
Consumer.

Network <b>Nodes</b>		
NAME	NODE ID	ORG OWNER
ProducerNode	31f4c...0b29c	did:firefly:node/ProducerNode
ResellerNode	e5a03...09ee1	did:firefly:node/ResellerNode
ConsumerNode	81da3...9557d	did:firefly:node/ConsumerNode



# Hyperledger FireFly

ABI importate in FireFly Sandbox per creare le contract interfaces e rendere disponibili le API di invocazione e listeners eventi.

**Register a Contract API**  
Generates an interactive HTTP API from your contract

Contract Interface \*  
WatchMarket\_IFACE - 1.1

Name \*  
WatchMarket\_API

Address \*  
0xcf7ed3acca5a467e9e704c703e8d87f634fb0fc9

The contract address on your blockchain.

Publish  
Controls whether to automatically publish the contract API to the network.

**Define a Contract Interface**  
Defines all methods and events of your contract to FireFly

Interface Format \*  
FFI - FireFly Interface  
Either FFI or Solidity ABI format

Schema \*

```
{
  "name": "my-contract",
  "version": "1.0",
  "methods": [
    {
      "name": "method1"
    }
  ],
  "events": [
    {
      "name": "event1"
    }
  ]
}
```

**Register a Contract Listener**  
Instructs FireFly to listen for and index events from your contract

Contract API \*  
WatchMarket\_API

Events \*  
Paused

Topic \*  
app-events

A FireFly identifier to group events across multiple listeners.

Name  
A custom name for your listener

First Event  
newest

The first event for this listener to index. Valid options are 'newest', 'oldest', or a specific block number.



# Hyperledger FireFly

Token pool per il tracciamento e la gestione degli Orologi (NFT) e token LUX (pagamenti)

Token Pools / **watchnft**

ID	367b34b3-94c4-4dfe-a694-53539d88dea2
Standard	ERC721 (nonfungible)
Connector	erc20_erc721
Locator	address=0x546a240ffbe4f6f0a4de7eca5d13da5591572b00&schema=ERC721NoData&typ...
Transaction ID	868a9ef5-b396-4afe-9bca-8e92334b31ab
Message ID	025370e0-c359-4490-aa5c-48565839936e
State	Active
Created	01/24/2026 10:52:06,533 AM (13 days ago)

**watchnft**

**Pool Info**

```
{
  "address": "0x546a240ffbe4f6f0a4de7eca5d13da5591572b00",
  "name": "WatchNFT",
  "schema": "ERC721NoData"
}
```

**Accounts in Pool**

KEY	BALANCE
0x005...cf921	1
0x0eb...602b9	1
0x0eb...602b9	1
0x0eb...602b9	1
0xe5c...bd1cf	0
0xe5c...bd1cf	0
0x0eb...602b9	1
0x0eb...602b9	0
0x005...cf921	1
0xe5c...bd1cf	0
0x0eb...602b9	0

**Transfers In Pool**

ACTIVITY	FROM	TO	AMOUNT	BLOCKCHAIN EVENT	SIGNING KEY	TIMESTAMP
Transfer	0xe5c...bd1cf	0x005...cf921	1	d3a83...de67f	0x005...cf921	2 hours ago

Balances for 0x0eb...602b9

- watchnft (Non-Fungible) Total: 6 (#1, #2, #3, #4, #5, #6)
- luxurycoin (Fungible) --- Total: 980013

Create a Token Pool

Defines a new set of tokens for FireFly to index

Type \* **Fungible**

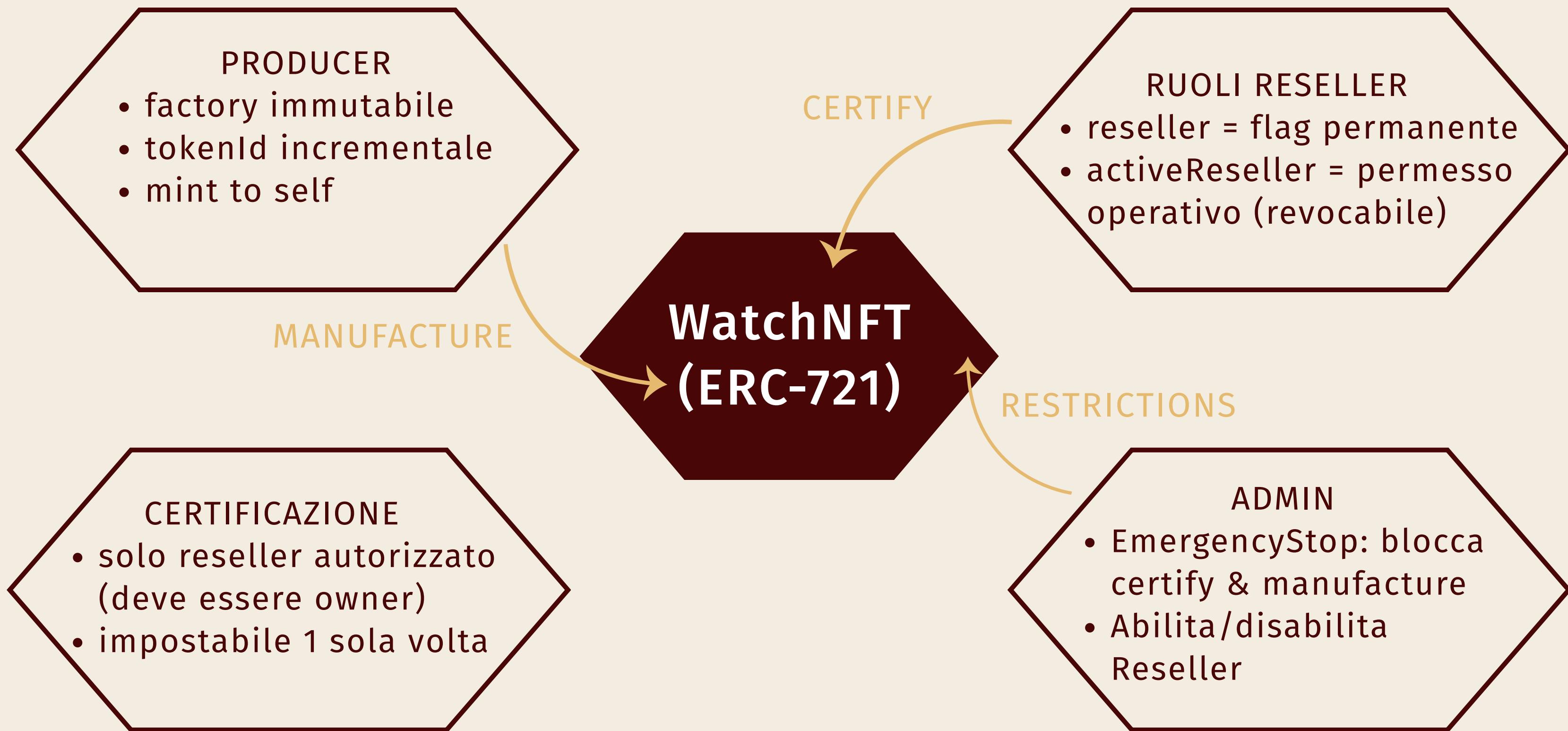
Contract Address



# Smart Contract



# WatchNFT



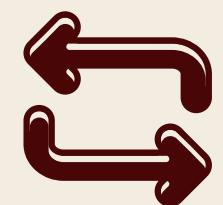
# WatchMarket

Gestisce la compravendita dei WatchNFT con un registro on-chain delle inserzioni.

- Modalità:
1. Vendita primaria (solo Producer vende, solo Reseller autorizzati acquistano)
  2. Vendita secondaria (solo Reseller autorizzato e proprietario vende, NFT deve essere certificato, solo Consumer acquistano).



Per pubblicare un annuncio serve l'autorizzazione al trasferimento (approve globale)



In fase di listing l'NFT viene trasferito in escrow al marketplace e resta custodito fino a vendita o annullamento (ritorno al venditore)



Pagamento in LuxuryCoin (ERC-20) con allowance e saldo sufficienti; in caso contrario la transazione va in revert



Dopo il pagamento l'NFT passa all'acquirente; il ricavo del venditore viene registrato come credito interno e riscosso con una funzione di prelievo



# LuxuryCoin (LUX)

Token fungibile ERC-20 usato come valuta interna di Watchain per regolare tutte le transazioni economiche

---

Evita l'uso diretto della valuta nativa di Ethereum, rendendo i pagamenti uniformi e gestibili tramite allowance ERC-20

---

Supply fissa definita al deploy: 1.000.000 LUX

---

Distribuzione iniziale: 10.000 LUX a un Reseller e 10.000 LUX a un Consumer; il resto assegnato al Producer



# Design pattern

PATTERN	SCOPO	APPLICAZIONE
<b>Guard Check</b>	Validare prerequisiti	Tutte le funzioni state-changing
<b>Access Restriction</b>	Permessi per ruolo	Producer/Reseller/Consumer + owner annuncio
<b>Check-Effects-Interactions</b>	Anti re-entrancy	Buy, transfer, withdraw
<b>Pull Payments</b>	Payout separato	Credito seller → withdraw
<b>Emergency Stop</b>	Pausa operazioni critiche	WatchNFT + WatchMarket (con “exit” consentite)
<b>SafeERC20</b>	Transfer robusti	Pagamenti LUX / allowance
<b>Tight Packing</b>	Gas optimization	Storage layout



# PullPayments

## Problema:

I trasferimenti automatici durante la vendita sono rischiosi. Se il wallet del destinatario rifiuta i fondi, l'intera transazione di acquisto fallisce.

## SOLUZIONE

Separazione netta tra vendita e incasso.  
Il contratto non invia denaro, ma aggiorna solo un saldo virtuale interno associato all'utente.

## IMPLEMENTAZIONE

La vendita aggiorna solo il credito contabile. Il venditore deve chiamare withdraw() per incassare.

## SICUREZZA

Il saldo viene azzerato prima del trasferimento per evitare attacchi di Re-entrancy.

## Il tuo Profilo

### RUOLO

Producer

### INDIRIZZO

0x0eb57516109f4a7e9e5706a1b45344be1d602b9

### SALDO

**980013 LUX**

### VENDITE DA INCASSARE

**8 LUX**

PRELEVA ORA



# EmergencyStop

## OBIETTIVO

Bloccare le attività critiche in caso di bug o attacchi



**SOLUZIONE**  
congelamento delle operazioni critiche

**IMPLEMENTAZIONE**  
modificatore whenNotPaused

**EXIT STRATEGY**  
in emergenza gli utenti possono recuperare gli NFT in escrow e prelevare i fondi accumulati



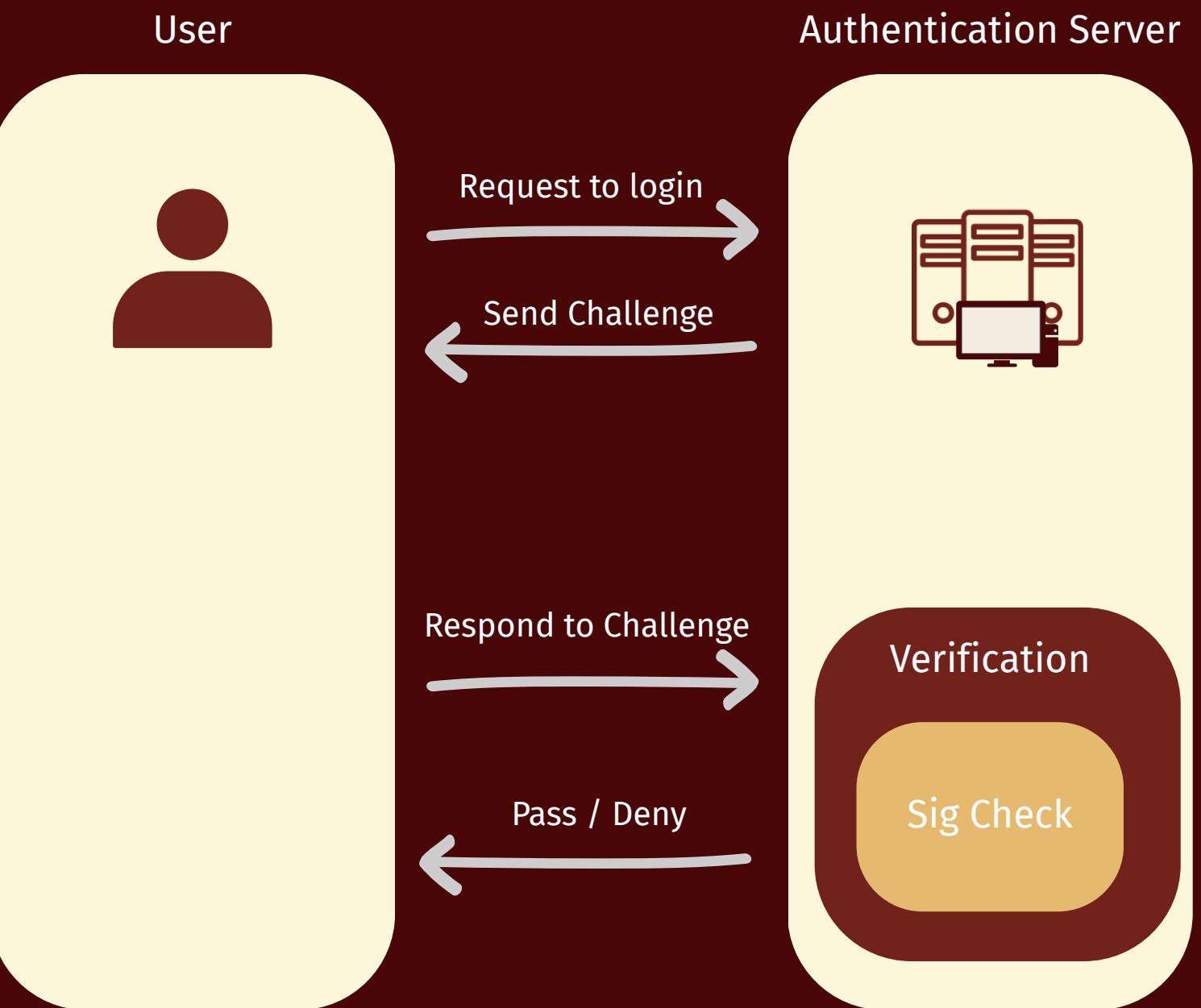
# Autenticazione

**Login passwordless:** No DB per lo storage di credenziali. Identità garantita dal possesso della chiave privata del wallet MetaMask.

**Challenge anti-replay:** Il server genera un Nonce casuale monouso con TTL e un messaggio.

**Firma off-chain:** l'utente firma nonce + messaggio crittograficamente, attraverso il proprio wallet.

Il server **verifica** la validità della firma per autenticare l'accesso. Se valida, rilascia i Cookie HttpOnly.



# Gestione sessione

## Architettura Stateless

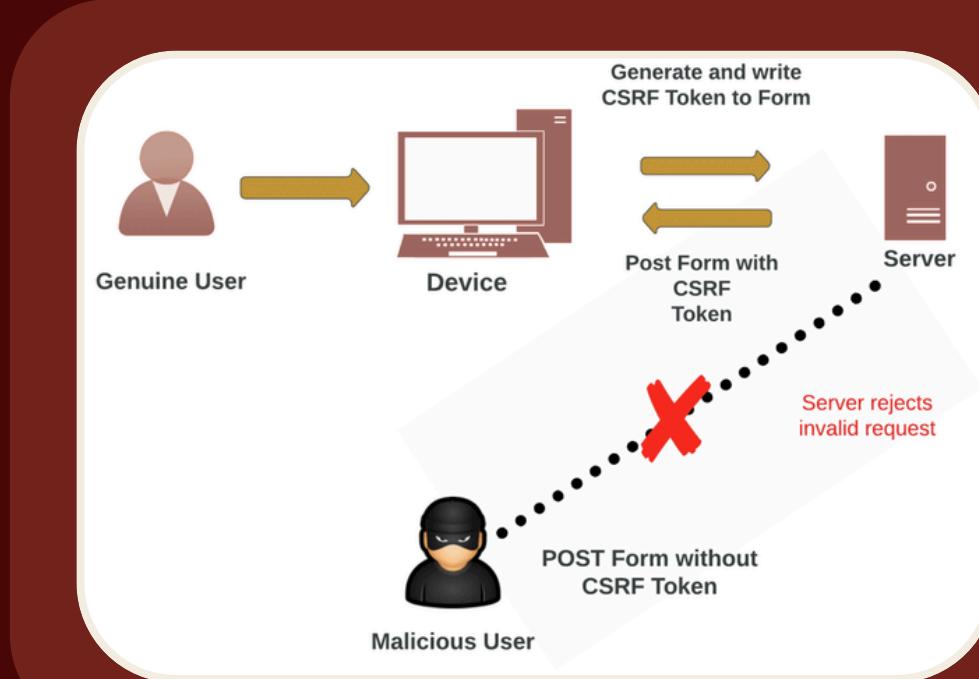
- JWT Stateless
- Garantisce scalabilità orizzontale
- Validazione della firma crittografica ad ogni richiesta.

## Sicurezza HttpOnly

- Mitigazione XSS: I token sono incapsulati in Cookie HttpOnly, invisibili al motore JavaScript del browser.

## Dual Token

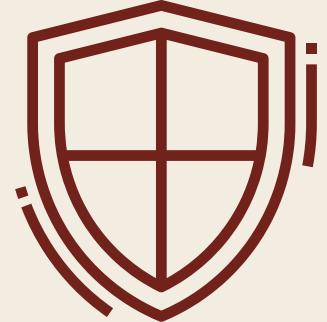
- Access Token (15m): finestra d'attacco ridotta.
- Refresh Token (14gg) + rotation: rinnovo sessione senza richiedere firma all'utente.



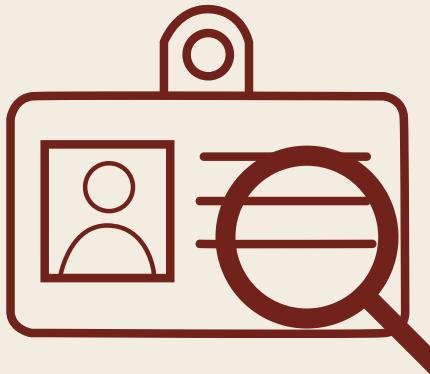
## Protezione CSRF

- Frontend legge il token dal Cookie
- Copia il token nell'header X-CSRF-Token
- Server verifica se coincidono
- Se diversi blocca la richiesta

# Autorizzazione



**Trusted Gateway:** Il Backend gestisce tutte le interazioni verso la blockchain, valida le transazioni prima dell'invio.



**Anti-Spoofing:** Il server ignora l'identità dichiarata dal client nel body della richiesta. L'identità viene estratta forzatamente dal Token JWT firmato, prevenendo l'impersonificazione.



**RBAC Immutabile:** La gestione dei ruoli è demandata agli Smart Contract. Regole di accesso scritte su Blockchain.

# Sincronizzazione Event-Driven

## 1. FireFly Listener

Gli eventi emessi dagli Smart Contract vengono intercettati da FireFly che invia una notifica al backend.

## 3. Notifica real-time

Il server propaga un segnale di "refresh" ai client connessi tramite WebSocket, garantendo reattività immediata.

## 2. Controllo

Il server riceve la notifica, ne verifica l'autenticità (Shared Secret) e risponde immediatamente per confermare la ricezione.

## 4. Sincronizzazione

L'app riceve il segnale di "refresh" e interroga FireFly per scaricare i dati ufficiali aggiornati. L'interfaccia mostra sempre la verità on-chain.

# Use Case

## UC\_01: Autenticazione utente

ID/Nome	UC_01: Autenticazione utente
Attore	Producer, Reseller, Consumer
Entry Condition	<ul style="list-style-type: none"><li>L'utente possiede un wallet associato ad un indirizzo blockchain.</li></ul>
Flusso	<ol style="list-style-type: none"><li>L'utente richiede l'accesso al sistema.</li><li>Il sistema genera un messaggio univoco temporaneo, associato all'indirizzo dell'utente.</li><li>L'utente firma il valore utilizzando il proprio wallet.</li><li>L'utente invia la firma al sistema.</li><li>Il sistema verifica la validità della firma e l'associazione con l'indirizzo dell'utente.</li><li>Il sistema autentica l'utente e avvia una sessione valida.</li></ol>
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	<ul style="list-style-type: none"><li>Valore scaduto: il sistema richiede una nuova autenticazione.</li><li>Firma non valida: il sistema rifiuta l'autenticazione.</li><li>Utente non autorizzato: il sistema limita o nega l'accesso alle funzionalità.</li></ul>

## UC\_02: Minting Orologio

ID/Nome	UC_02: Minting
Attore	Producer
Entry Condition	<ul style="list-style-type: none"><li>• Il Producer è autenticato.</li><li>• Il sistema non è in stato di emergenza per le operazioni di creazione.</li></ul>
Flusso	<ol style="list-style-type: none"><li>1. Il Producer richiede la creazione di un nuovo orologio al sistema, attraverso l'apposito pulsante.</li><li>2. Il sistema, dopo aver richiesto conferma all'utente, valida l'autorizzazione del Producer per l'operazione di creazione.</li><li>3. Il sistema crea l'orologio assegnandogli un identificativo univoco.</li><li>4. Il sistema conferma l'avvenuta creazione.</li></ol>
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	<ul style="list-style-type: none"><li>• Il Producer è autenticato.</li></ul>
Flusso	<ol style="list-style-type: none"><li>1. Il Producer richiede attraverso l'apposita interfaccia, la gestione di un Reseller, indicando l'indirizzo wallet e l'operazione desiderata (abilitazione o revoca).</li><li>2. Il sistema richiede conferma.</li><li>3. Il sistema aggiorna lo stato del Reseller.</li><li>4. Il sistema conferma l'avvenuto aggiornamento.</li></ol>
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	<ul style="list-style-type: none"><li>• Il Producer è autenticato ed è proprietario di uno o più orologi.</li><li>• Il Reseller è autenticato ed è abilitato come Reseller attivo.</li><li>• Le operazioni di mercato primario non sono sospese per emergenza del sistema.</li></ul>
Flusso	<ol style="list-style-type: none"><li>1. Il Producer seleziona un orologio dal proprio inventario e richiede di metterlo in vendita sul mercato primario indicando un prezzo.</li><li>2. Il sistema richiede conferma all'utente.</li><li>3. Il sistema verifica che l'orologio appartenga al Producer e che la vendita sia consentita.</li><li>4. Il sistema rende l'orologio disponibile sul mercato primario.</li><li>5. Il Reseller seleziona un orologio dal mercato primario e richiede l'acquisto.</li><li>6. Il sistema richiede conferma all'utente.</li><li>7. Il sistema verifica che il Reseller sia autorizzato ad acquistare sul mercato primario.</li><li>8. Il sistema registra l'acquisto e trasferisce la proprietà dell'orologio al Reseller.</li></ol>
Exit Condition (on success)	<ul style="list-style-type: none"><li>• L'orologio viene trasferito dal Producer al Reseller.</li><li>• Il Producer matura un credito corrispondente al prezzo di vendita.</li><li>• L'orologio non risulta più essere sul mercato primario.</li></ul>
Exit Condition (on failure)	<ul style="list-style-type: none"><li>• Il Producer ha inserito un prezzo non valido. Se il prezzo è <math>\leq 0</math>, il sistema impedisce il listing.</li><li>• Errore durante la transazione: il sistema non completa il trasferimento e mantiene lo stato precedente.</li><li>• Utente non autorizzato: il sistema impedisce l'operazione.</li></ul>
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	<ul style="list-style-type: none"><li>Il Reseller è autenticato.</li><li>Il Reseller è autorizzato ad operare come tale.</li><li>Il Reseller è proprietario dell'orologio da certificare.</li><li>Le operazioni di certificazione non sono sospese per lo stato di emergenza.</li></ul>
Flusso	<ol style="list-style-type: none"><li>Il Reseller seleziona un orologio dal proprio inventario.</li><li>Il Reseller richiede la certificazione dell'orologio.</li><li>Il sistema richiede conferma al Reseller.</li><li>Il sistema verifica che il Reseller sia autorizzato e proprietario dell'orologio.</li><li>Il sistema registra la certificazione dell'orologio.</li><li>Il sistema conferma l'avvenuta certificazione.</li></ol>
Exit Condition (on success)	<ul style="list-style-type: none"><li>L'orologio risulta certificato dal Reseller.</li><li>Lo stato di certificazione dell'orologio è aggiornato e visibile.</li></ul>
Exit Condition (on failure)	Il sistema rifiuta la richiesta di certificazione. Lo stato dell'orologio rimane invariato.
Flusso alternativo	<ul style="list-style-type: none"><li>Operazioni di certificazioni sospese: il sistema segnala che la funzionalità non è disponibile.</li><li>Il Reseller non è abilitato: il sistema rifiuta l'operazione di certificazione.</li></ul>

ID/Nome	UC_06: Mercato secondario
Attore	Reseller, Consumer
Entry Condition	<ul style="list-style-type: none"> <li>Il Reseller è autenticato ed è autorizzato ad operare come tale.</li> <li>Il Reseller è proprietario di uno o più orologi certificati.</li> <li>Il Consumer è autenticato.</li> <li>Le operazioni di mercato non sono sospese per lo stato di emergenza del sistema.</li> </ul>
Flusso	<ol style="list-style-type: none"> <li>Il Reseller seleziona un orologio certificato dal proprio inventario e richiede di metterlo in vendita sul mercato secondario indicando un prezzo.</li> <li>Il sistema richiede la conferma al Reseller.</li> <li>Il sistema verifica che l'orologio sia certificato e di proprietà del Reseller.</li> <li>Il sistema rende l'orologio disponibile sul mercato secondario.</li> <li>Il Consumer seleziona un orologio dal mercato secondario e richiede l'acquisto.</li> <li>Il sistema richiede la conferma al Consumer.</li> <li>Il sistema verifica che l'operazione rispetti i requisiti di vendita.</li> <li>Il sistema registra l'acquisto e trasferisce la proprietà dell'orologio al Consumer.</li> </ol>
Exit Condition (on success)	<ul style="list-style-type: none"> <li>Un orologio viene trasferito dal Reseller al Consumer.</li> <li>Il Reseller matura un credito corrispondente al prezzo di vendita.</li> <li>L'orologio non risulta più essere sul mercato secondario.</li> </ul>
Exit Condition (on failure)	<ul style="list-style-type: none"> <li>Il Reseller ha inserito un prezzo non valido. Se il prezzo è <math>\leq 0</math>, il sistema impedisce il listing.</li> <li>Errore durante la transazione: il sistema non completa il trasferimento e mantiene lo stato precedente.</li> <li>Utente non autorizzato: il sistema impedisce l'operazione.</li> </ul>
Flusso alternativo	Se il blocco del mercato è attivo, il sistema non permette la compravendita di orologi.

## UC\_06: Vendita e acquisto sul mercato secondario

## UC\_07: Pagamenti e prelievo dei fondi

ID/Nome	UC_07: Pagamenti e prelievo fondi
Attore	Producer, Reseller
Entry Condition	<ul style="list-style-type: none"><li>L'utente è autenticato.</li><li>L'utente ha maturato crediti a seguito di una o più vendite.</li></ul>
Flusso	<ol style="list-style-type: none"><li>L'utente richiede il prelievo dei fondi maturati.</li><li>Il sistema richiede la conferma all'utente.</li><li>Il sistema verifica che l'utente abbia un saldo disponibile.</li><li>Il sistema trasferisce i fondi all'utente.</li><li>Il sistema aggiorna il saldo dei crediti.</li></ol>
Exit Condition (on success)	<ul style="list-style-type: none"><li>I fondi maturati vengono trasferiti all'utente.</li><li>Il saldo dei crediti dell'utente viene aggiornato.</li></ul>
Exit Condition (on failure)	<ul style="list-style-type: none"><li>Nessun credito disponibile: il sistema non mostra il tasto per il prelievo.</li><li>Errore durante il trasferimento: il sistema non aggiorna il saldo e segnala il problema.</li></ul>
Flusso alternativo	//

# UC\_08: Aggiornamento in tempo reale tramite eventi Blockchain

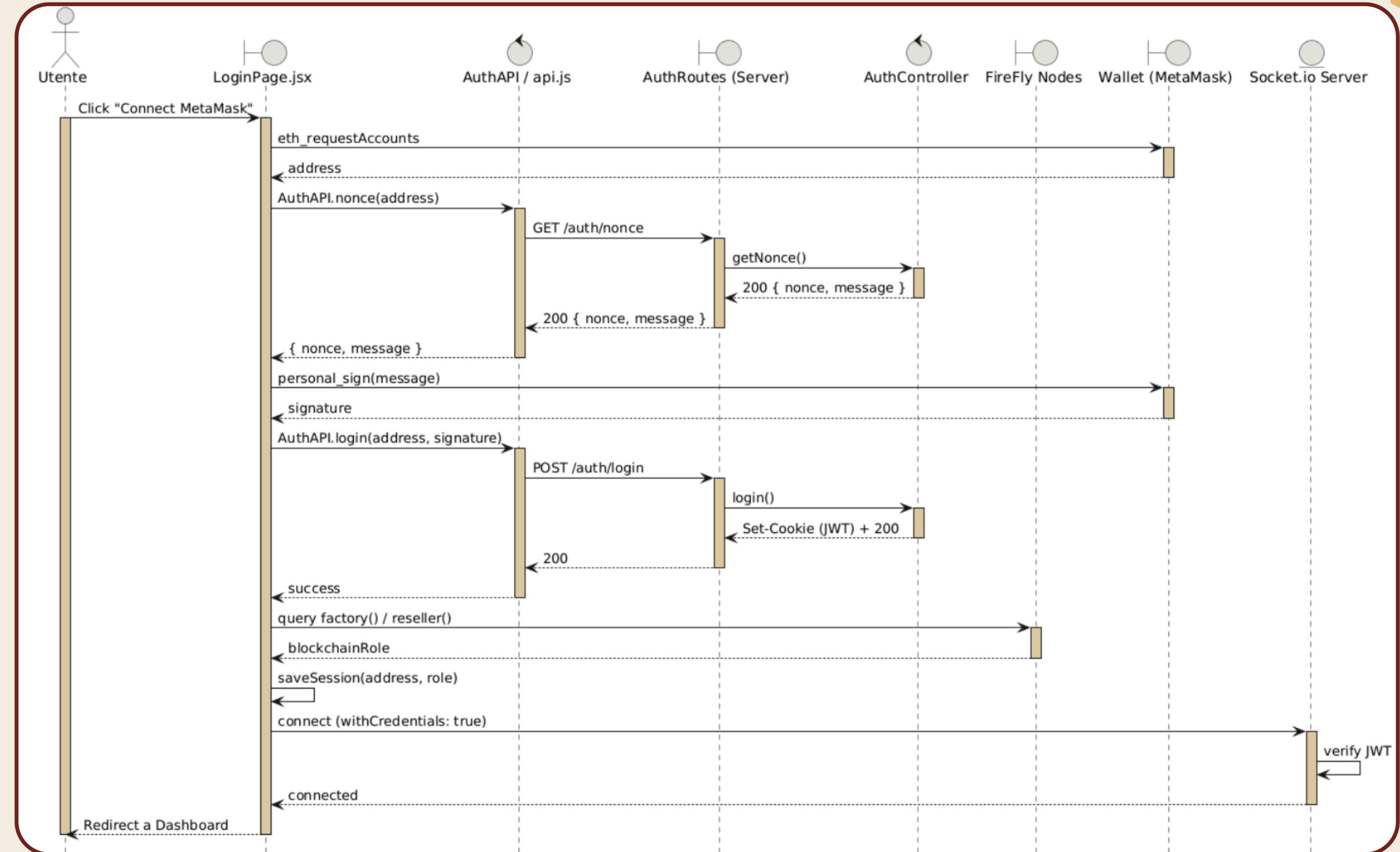
ID/Nome	UC_08: Aggiornamenti in tempo reale
Attore	Producer, Reseller, Consumer
Entry Condition	<ul style="list-style-type: none"><li>• Il sistema è in esecuzione e in ascolto degli eventi.</li><li>• L'utente è autenticato e sta visualizzando una schermata che mostra dati aggiornabili (market, profilo, ecc.).</li></ul>
Flusso	<ol style="list-style-type: none"><li>1. L'utente compie un'operazione che può modificare lo stato (es. acquisto, vendita, certificazione).</li><li>2. Il sistema rileva il cambiamento di stato.</li><li>3. Il sistema notifica agli utenti connessi che è disponibile un aggiornamento.</li><li>4. Il sistema aggiorna automaticamente i dati sul dispositivo dell'utente recuperando lo stato più recente.</li><li>5. L'utente visualizza lo stato aggiornato.</li></ol>
Exit Condition (on success)	<ul style="list-style-type: none"><li>• Lo stato visualizzato dall'Utente è aggiornato.</li></ul>
Exit Condition (on failure)	Il sistema ignora l'evento e non invia notifiche. L'utente non visualizza gli aggiornamenti in tempo reale.
Flusso alternativo	<ul style="list-style-type: none"><li>• Evento non valido o incompleto: il sistema ignora l'evento e non invia notifiche.</li><li>• Sorgente non autorizzata: il sistema rifiuta l'evento e non invia notifiche.</li><li>• Errore durante l'elaborazione: il sistema non notifica e registra l'errore per diagnosi.</li><li>• Utente non connesso: l'utente non riceve l'aggiornamento in tempo reale, ma visualizzerà lo stato aggiornato alla successiva consultazione.</li></ul>

## UC\_09: Gestione Emergenza e Sospensione operazioni

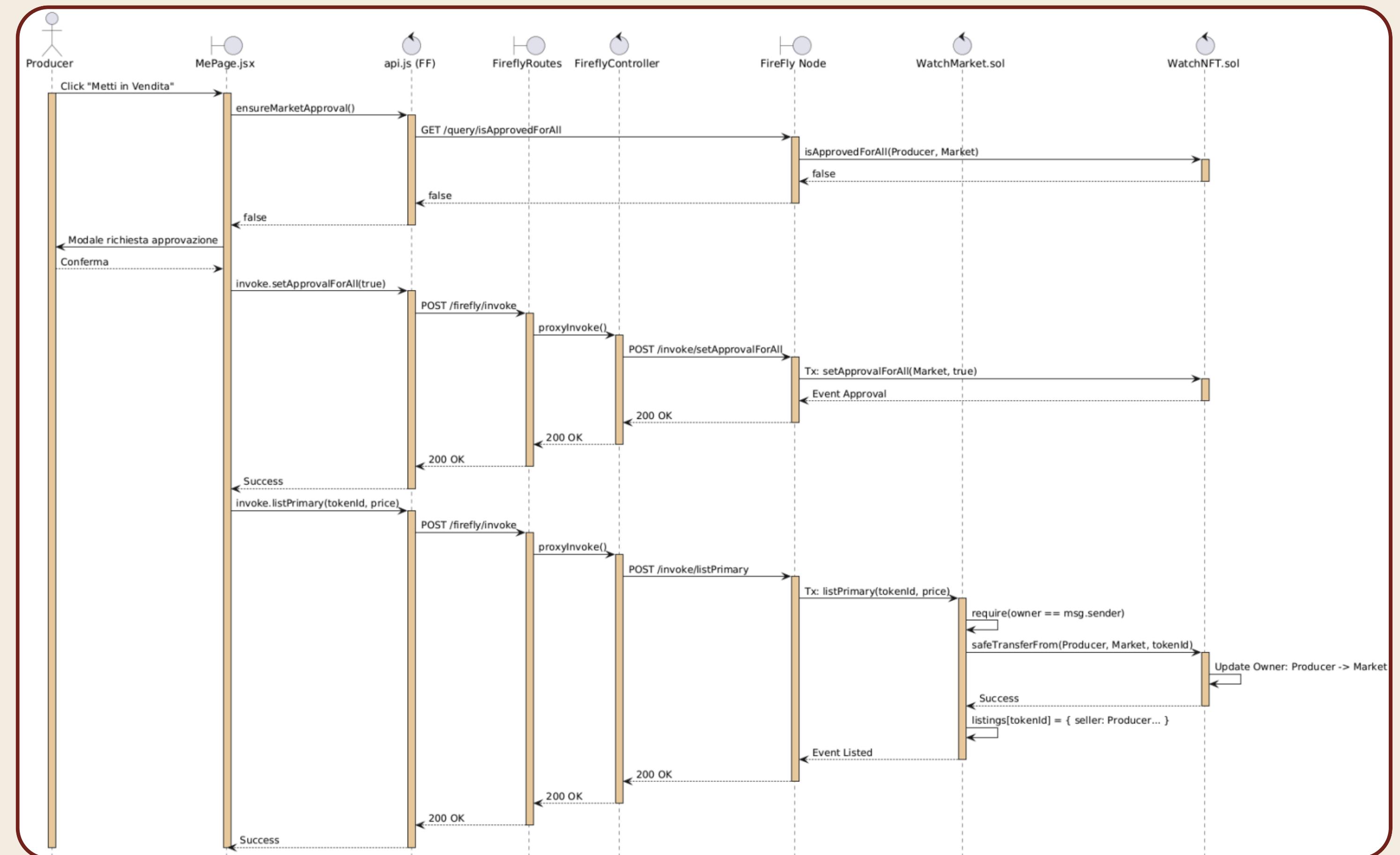
ID/Nome	UC_09: Sospensione operazioni
Attore	Producer
Entry Condition	<ul style="list-style-type: none"> <li>• Il Producer è autenticato.</li> </ul>
Flusso	<ol style="list-style-type: none"> <li>1. Il Producer accede alla funzionalità di gestione emergenza.</li> <li>2. Il Producer seleziona l'azione desiderata.</li> <li>3. Il sistema richiede conferma dell'operazione.</li> <li>4. Il sistema aggiorna lo stato operativo del sistema, abilitando o bloccando le operazioni interessate.</li> <li>5. Il sistema conferma l'avvenuto aggiornamento.</li> </ol>
Exit Condition (on success)	<ul style="list-style-type: none"> <li>• Se la sospensione è attivata, le operazioni di creazione e/o compravendita risultano temporaneamente non disponibili.</li> <li>• Se la sospensione è disattivata, le operazioni tornano disponibili secondo le regole di autorizzazione previste.</li> </ul>
Exit Condition (on failure)	Lo stato del sistema rimane invariato e l'operazione non viene applicata.
Flusso alternativo	//

# Sequence Diagrams

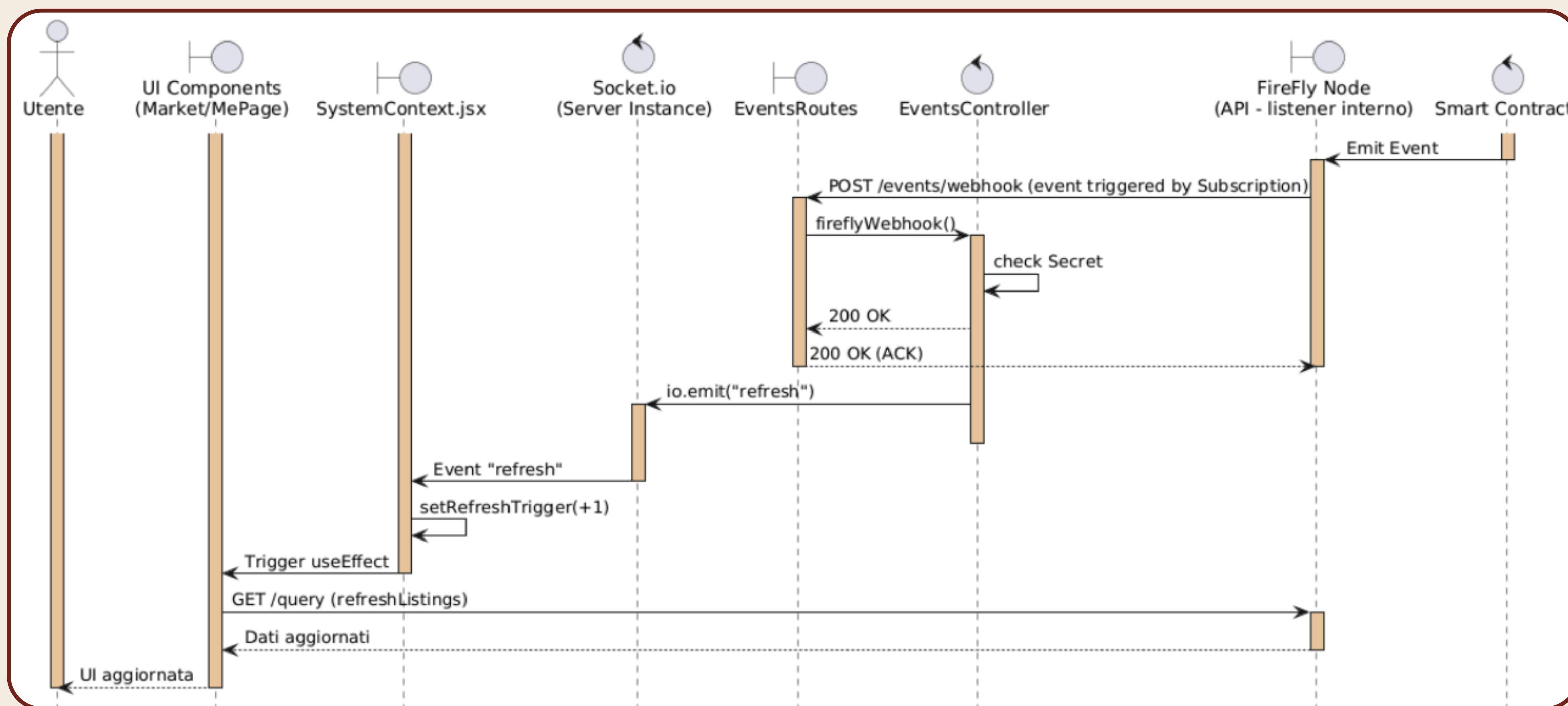
## SD\_01: Autenticazione utente



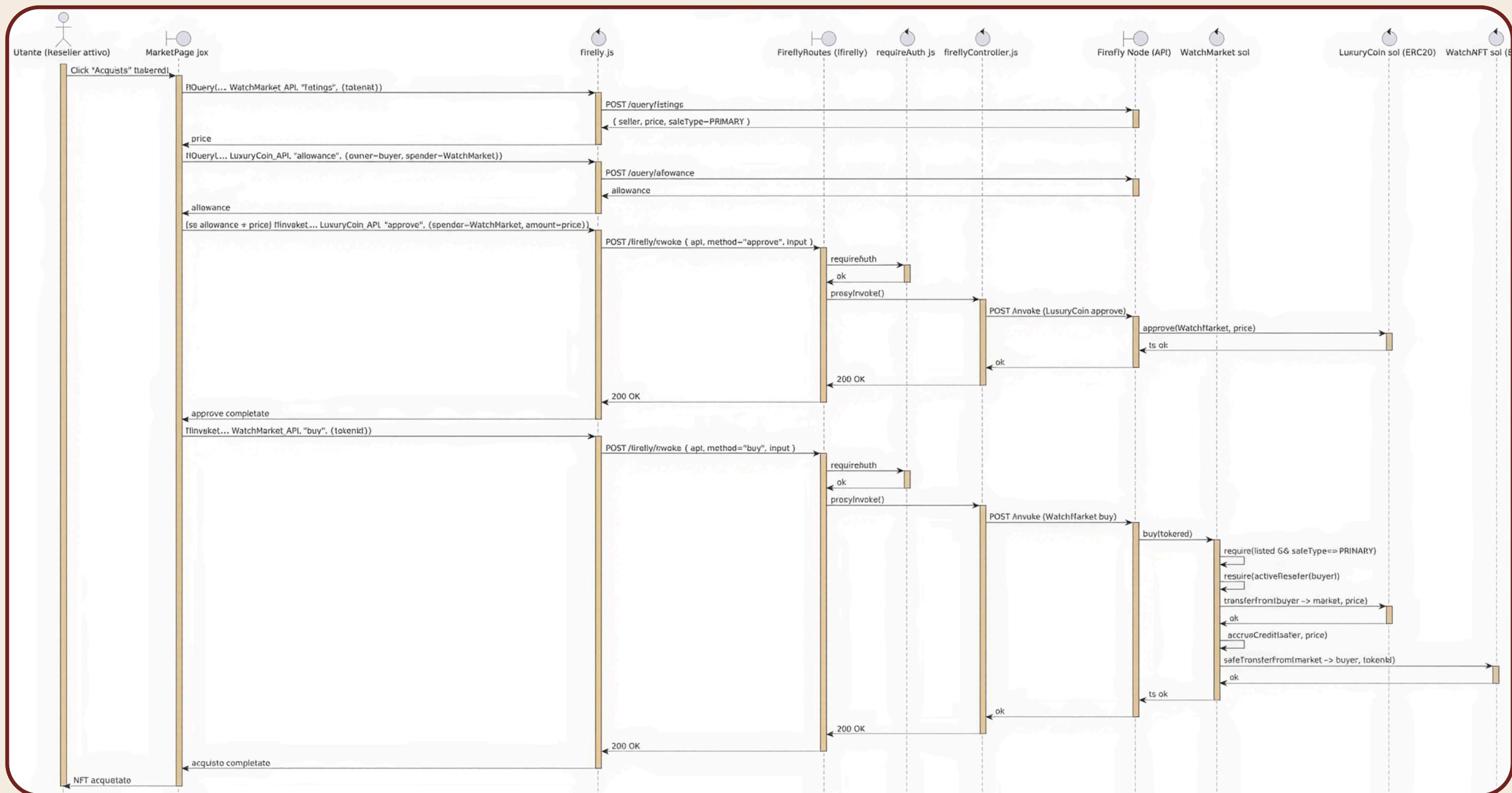
## SD\_02: Listing primario



## SD\_03: Aggiornamento in tempo reale tramite eventi



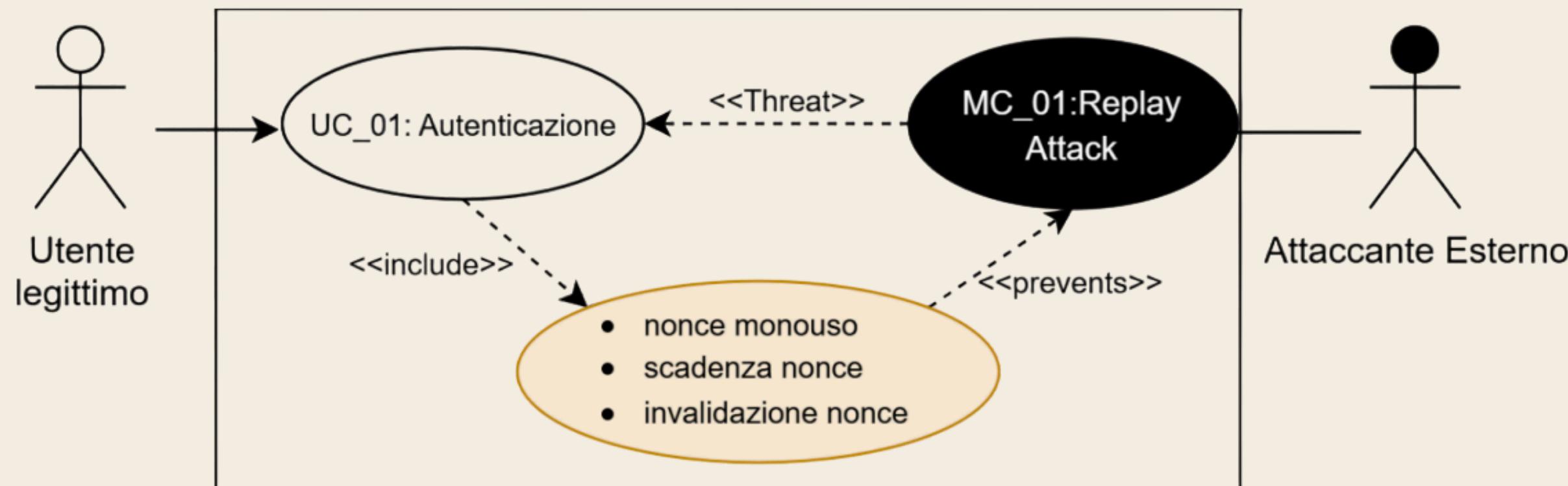
## SD\_04: Acquisto primario



# Misure Case

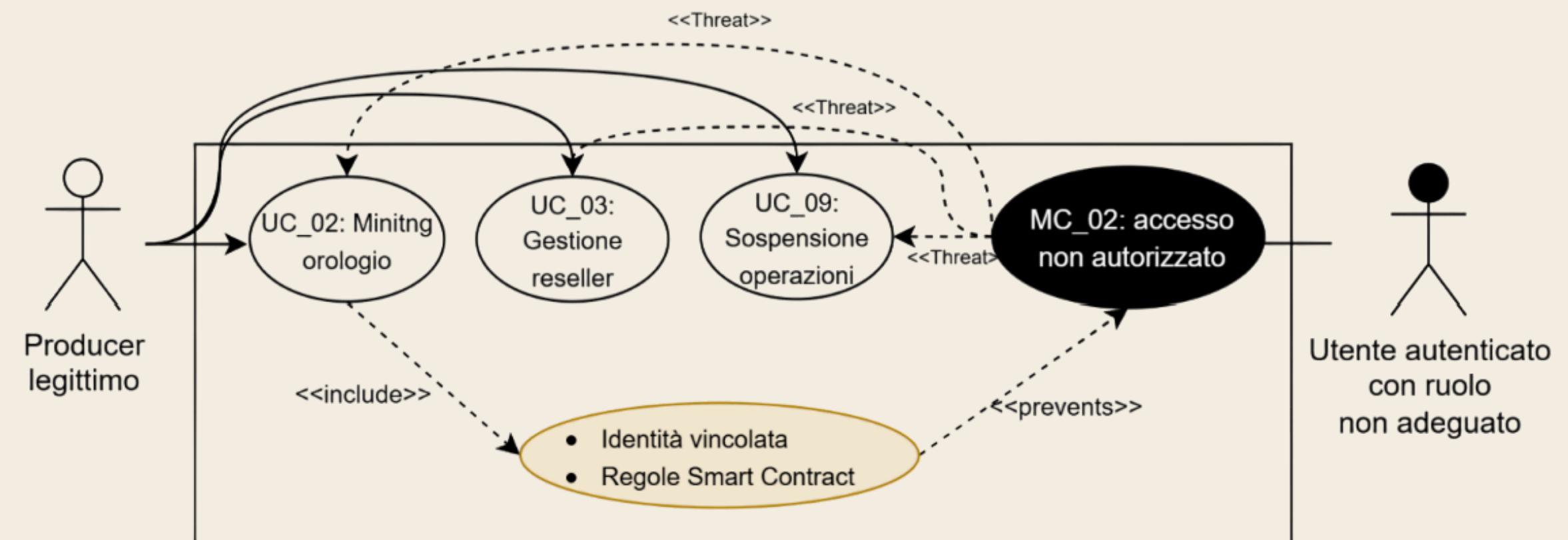
## MC\_01: Riutilizzo della firma di autenticazione (Replay Attack)

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	<ul style="list-style-type: none"><li>Utilizzo di nonce monouso associati all'indirizzo wallet</li><li>Scadenza temporale del nonce</li><li>Invalidazione del nonce dopo il primo utilizzo</li></ul>



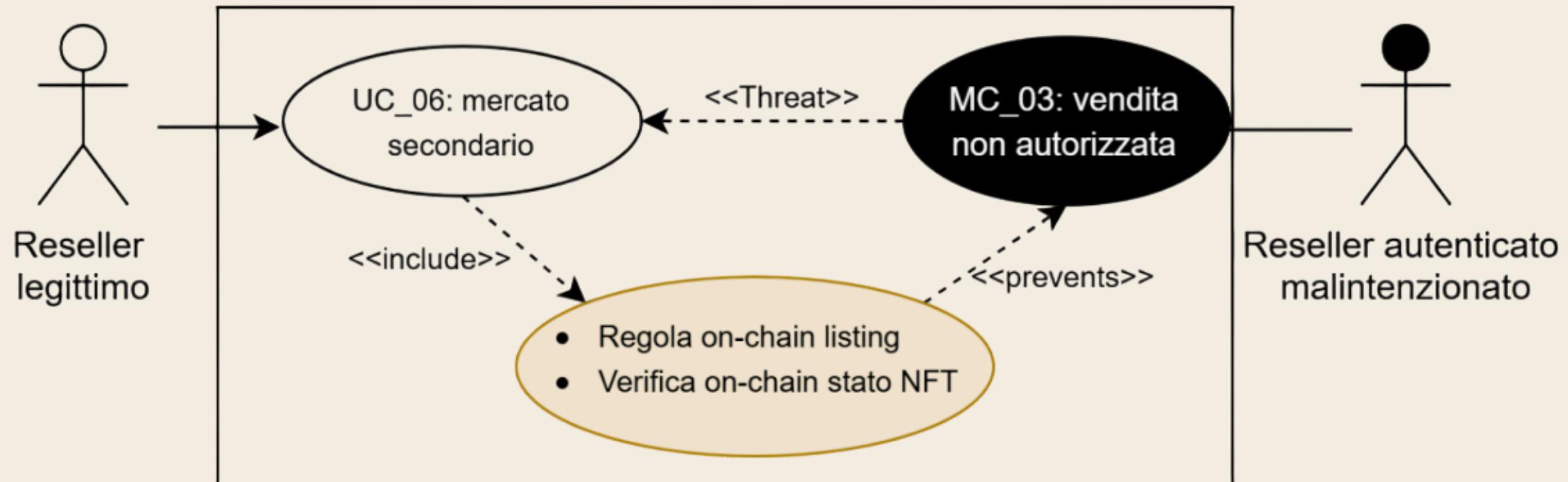
## MC\_02: Accesso a funzionalità non autorizzate per ruolo

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	<ul style="list-style-type: none"> <li>• UC_02: Minting Orologio</li> <li>• UC_03: Gestione Reseller</li> <li>• UC_09: Sospensione operazioni</li> </ul>
Contromisure	<ul style="list-style-type: none"> <li>• Identità vincolata: firma sempre associata all'indirizzo autenticato.</li> <li>• Autorizzazione finale on-chain: operazioni non ammesse falliscono.</li> </ul>



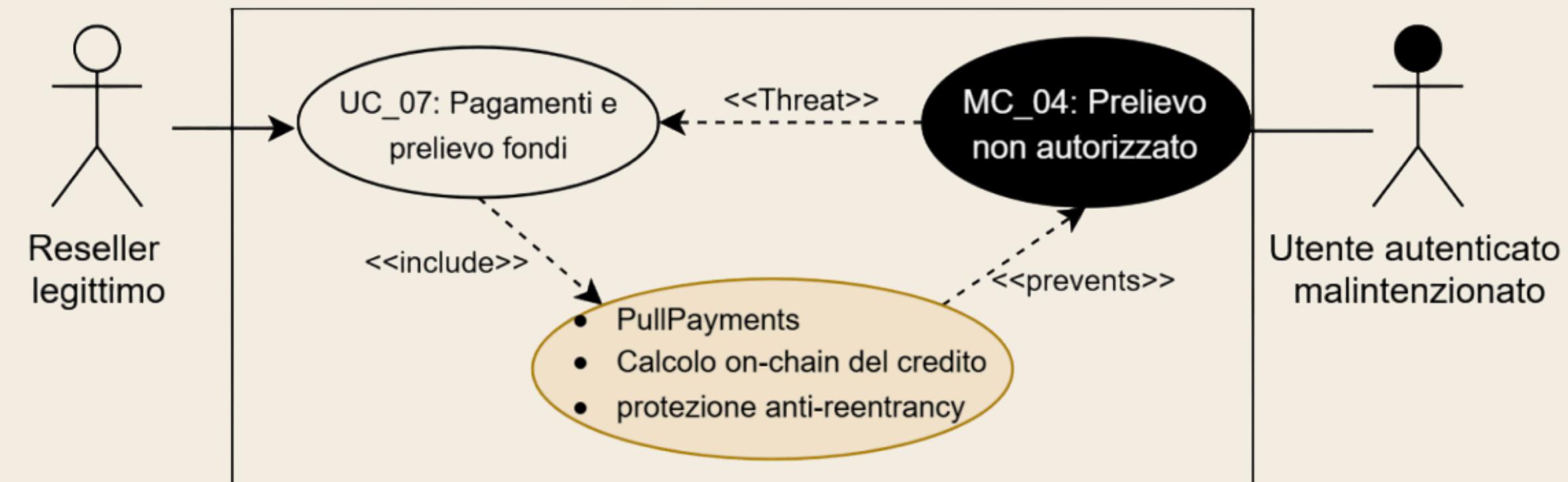
## MC\_03: Vendita di orologi non certificati

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	<ul style="list-style-type: none"> <li>• Verifica on-chain dello stato di certificazione dell'NFT</li> <li>• Regola on-chain: listing consentito solo per NFT certificati.</li> </ul>



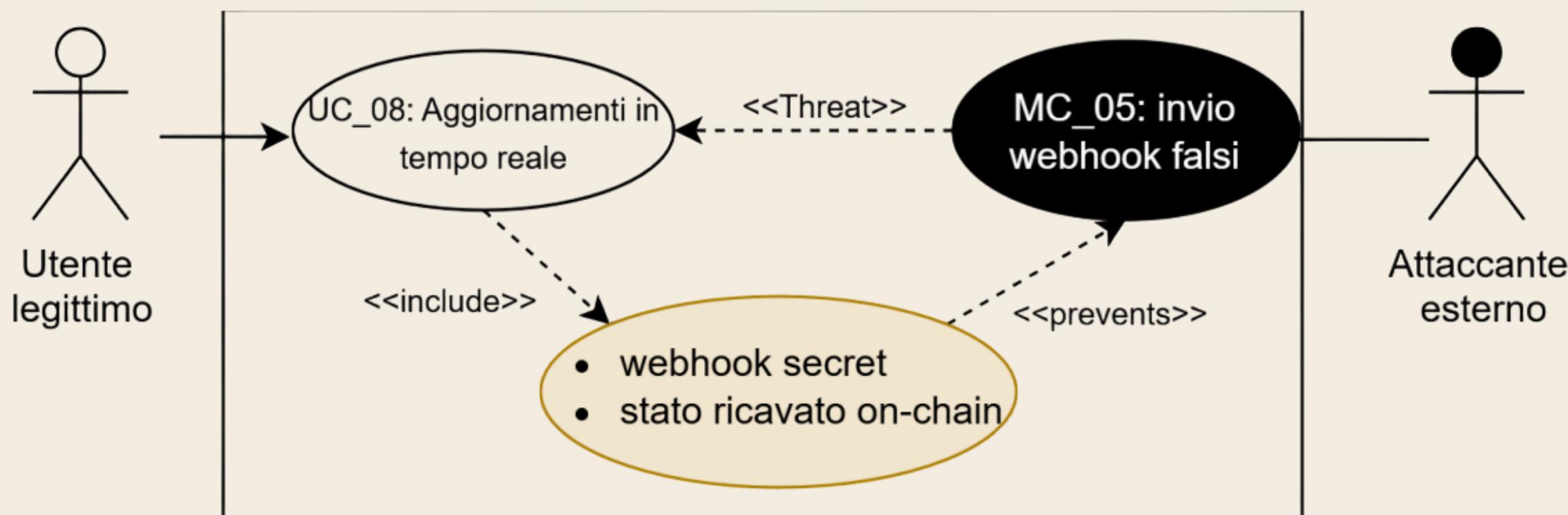
## MC\_04: Tentativo di prelievo improprio dei fondi

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	Ottenerne un vantaggio economico indebito.
Funzionalità minacciata	UC_07: Pagamenti e prelievo dei fondi
Contromisure	<ul style="list-style-type: none"> <li>• Meccanismo PullPayments con saldo interno per indirizzo</li> <li>• Calcolo on-chain dei crediti maturati</li> <li>• Protezione contro la reentrancy</li> </ul>



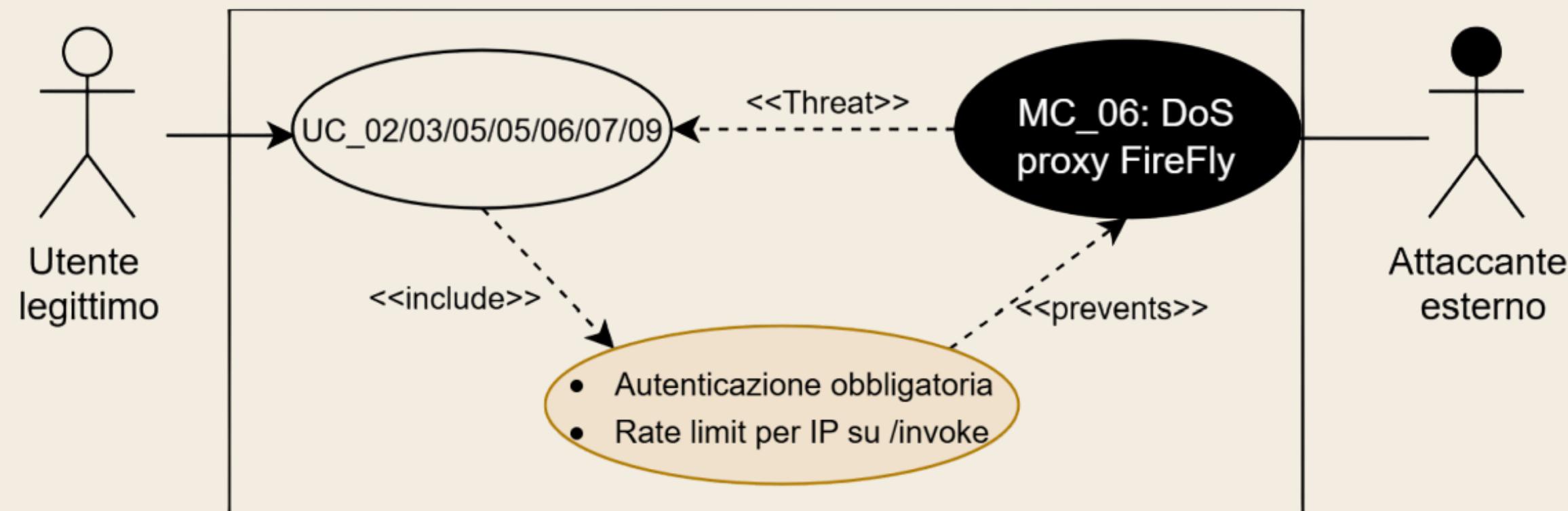
ID/Nome	MC_05: invio webhook falsificati
Scenario	Un attaccante tenta di inviare richieste false all'endpoint webhook per generare notifiche di aggiornamento non legittime.
Misactor	Attaccante esterno
Obiettivo	Generare refresh/notifiche di aggiornamento non legittime (disturbo o spam).
Funzionalità minacciata	UC_08: Aggiornamenti in tempo reale
Contromisure	<ul style="list-style-type: none"> <li>• webhook protetto da secret (richieste non autorizzate rifiutate)</li> <li>• Verità on-chain: lo stato mostrato è ricavato da dati on-chain (il webhook non può alterarlo)</li> </ul>

## MC\_05: Invio di webhook blockchain falsificati



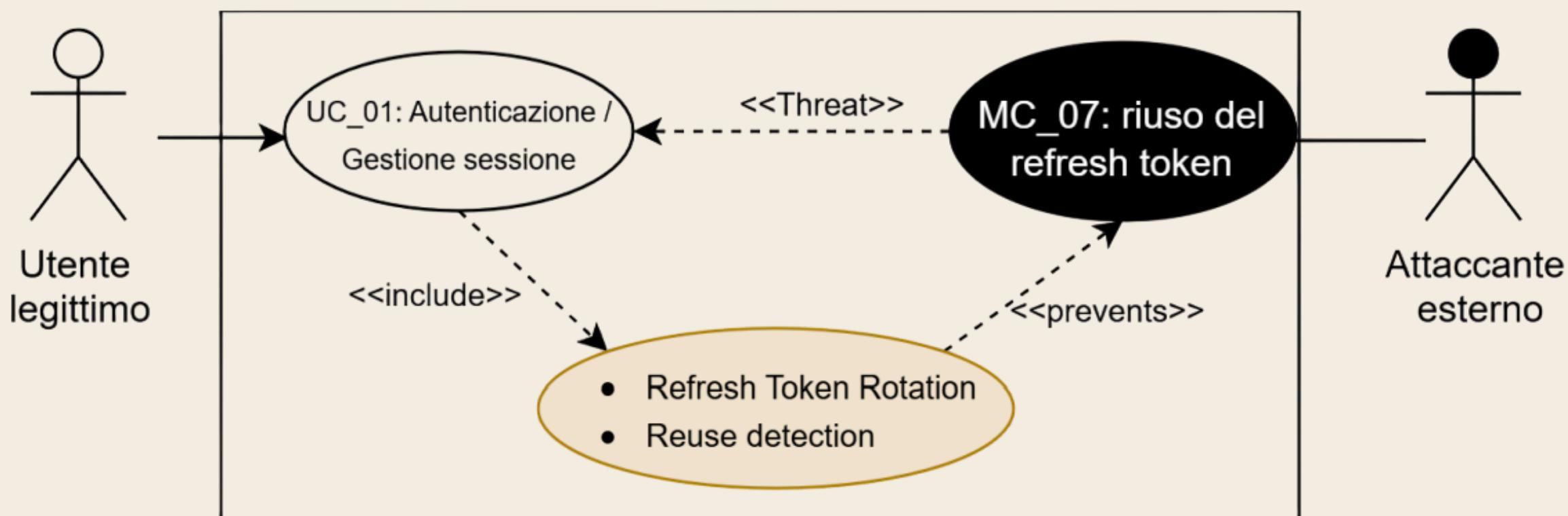
## MC\_06: DoS dell'endpoint di invocazione

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	<p>Tutti gli UC che richiedono /firefly/invoke:</p> <ul style="list-style-type: none"> <li>• UC_02: Minting</li> <li>• UC_03: Gestione Reseller</li> <li>• UC_04: Mercato primario</li> <li>• UC_05: Certificazione</li> <li>• UC_06: Mercato secondario</li> <li>• UC_07: Prelievo dei fondi maturati</li> <li>• UC_09: Sospensione operazioni</li> </ul>
Contromisure	<ul style="list-style-type: none"> <li>• Rate limit per IP sull'endpoint di invocazione</li> <li>• Autenticazione obbligatoria per invocazioni</li> </ul>



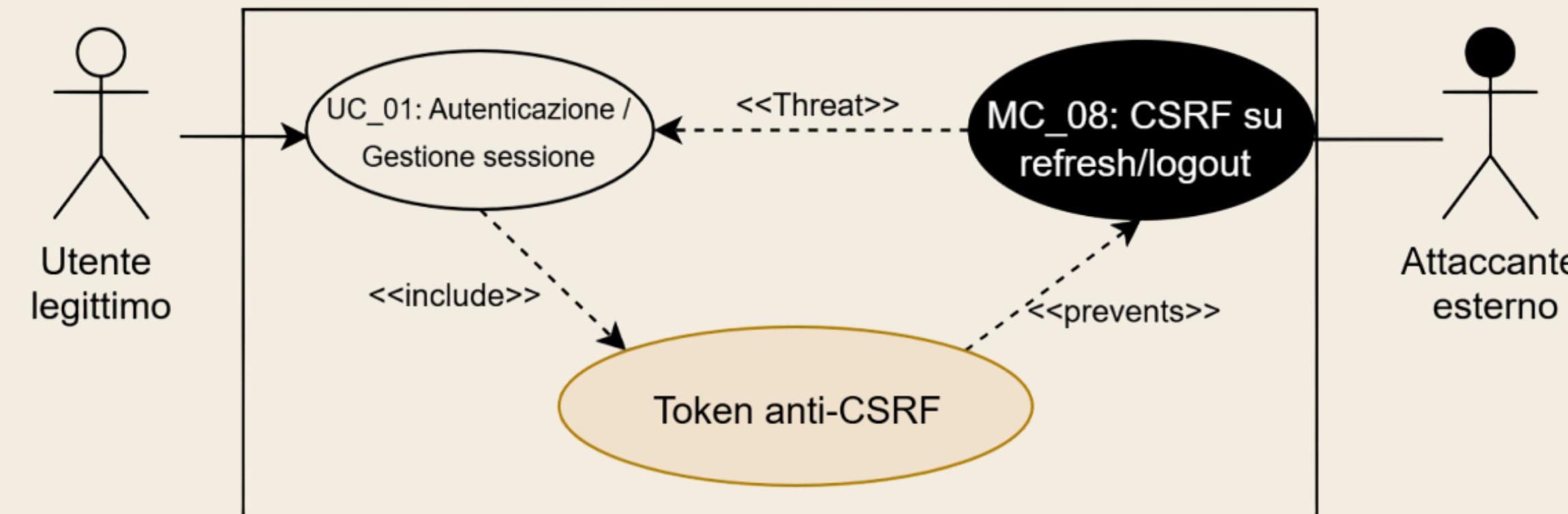
## MC\_07: Riuso del refresh token

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	<ul style="list-style-type: none"> <li>• UC_01: Autenticazione</li> <li>• Gestione della sessione</li> </ul>
Contromisure	<ul style="list-style-type: none"> <li>• Refresh Token Rotation: il token valido è solo l'ultimo emesso</li> <li>• Reuse detection: se un refresh "vecchio" viene riutilizzato, la sessione viene revocata/bloccata lato server</li> </ul>



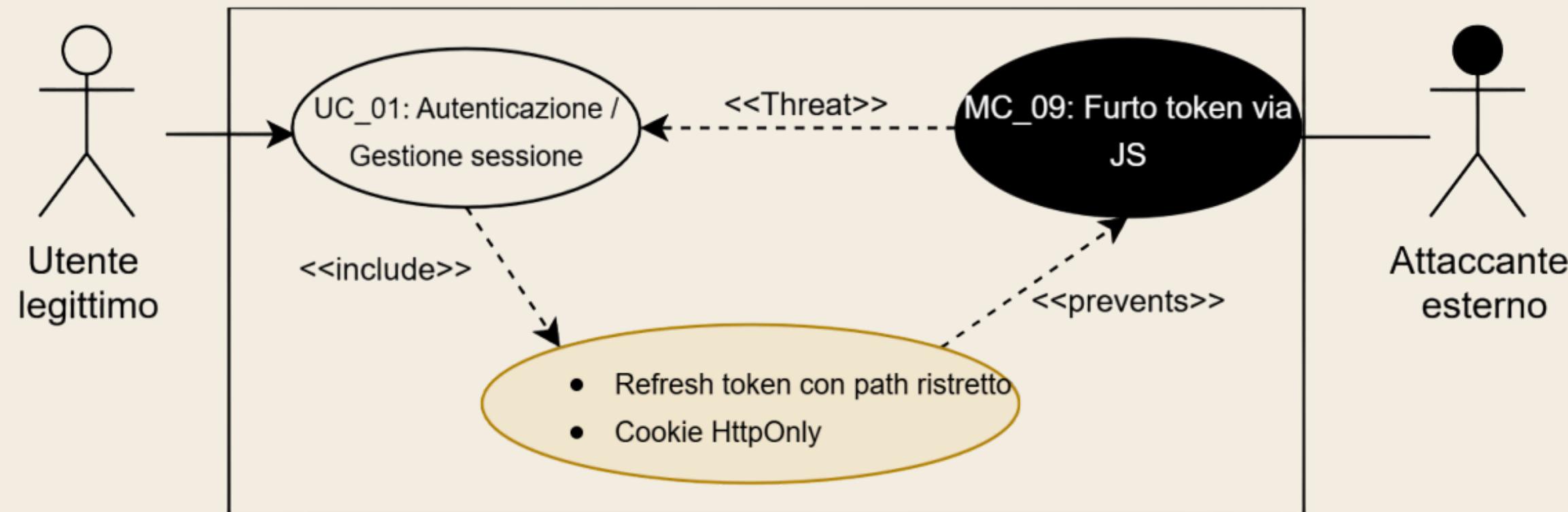
# MC\_08: Cross-Site Request Forgery su refresh/logout

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	<ul style="list-style-type: none"><li>Gestione della sessione</li><li>UC_01: Autenticazione</li></ul>
Contromisure	<ul style="list-style-type: none"><li>Token anti-CSRF: il server Accetta refresh/logout solo se la richiesta include un token CSRF valido (cookie CSRF + header con lo stesso valore)</li></ul>



## MC\_09: Furto token via JavaScript (XSS)

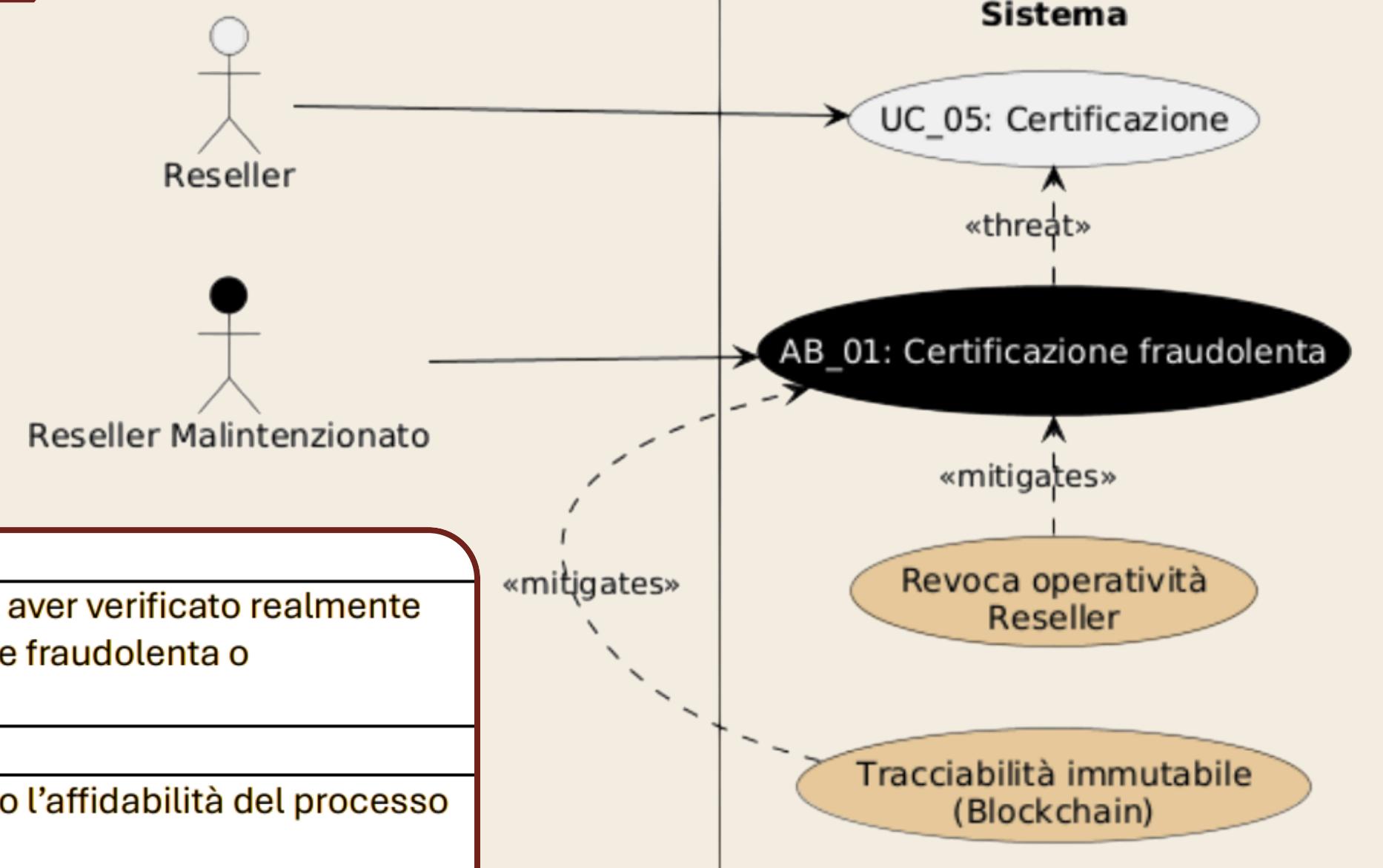
ID/Nome	MC_09: Furto token via JavaScript
Scenario	Uno script malevolo (XSS) nella pagina tenta di leggere i token di sessione dal browser per impersonare l'utente.
Misactor	Attaccante esterno (Iniezione script)
Obiettivo	Ottenerne token e utilizzarne la sessione della vittima
Funzionalità minacciata	<ul style="list-style-type: none"> <li>Gestione della sessione</li> <li>UC_01: Autenticazione</li> </ul>
Contromisure	<ul style="list-style-type: none"> <li>Token sensibili in cookie HttpOnly (quindi non leggibili da JavaScript)</li> <li>Refresh token con path ristretto /auth/refresh per ridurre l'esposizione</li> </ul>



# Abuse Case

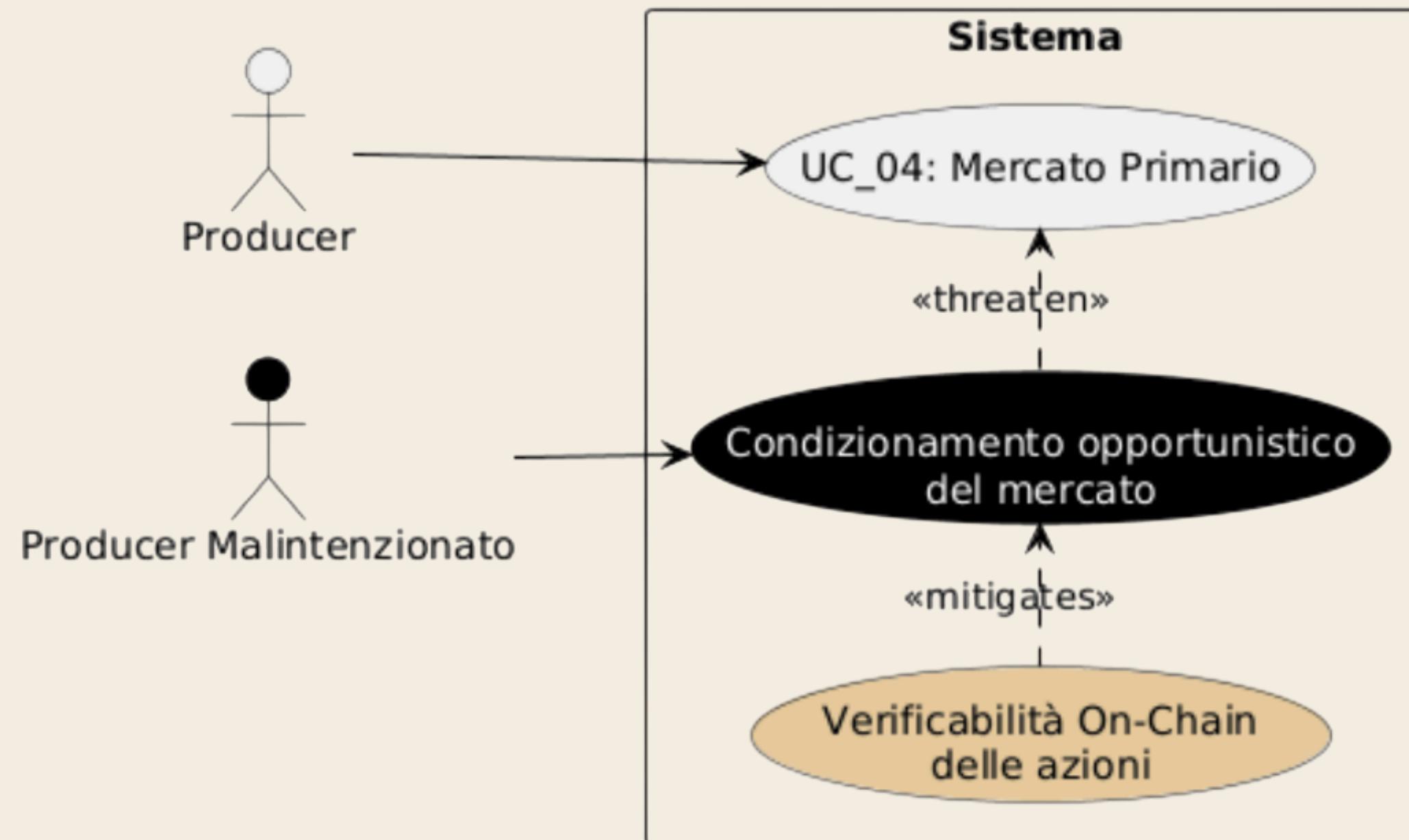
## AB\_01: Abuso delle funzionalità di certificazione

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	<ul style="list-style-type: none"><li>• Revoca operatività Reseller</li><li>• Tracciabilità immutabile delle operazioni di certificazione</li></ul>



## 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	Ottenerne un vantaggio competitivo o influenzare la filiera.
Funzionalità abusata	UC_04: Mercato primario
Contromisure	<ul style="list-style-type: none"><li>• Tracciabilità/verificabilità delle azioni di governance</li></ul>



# Limitazioni

## Firma Custodial

L'utente firma solo il login, le transazioni Blockchain sono firmate dai nodi FireFly.

Trade-off: Si sacrifica la decentralizzazione totale per abbattere i costi (Gas Fee) e semplificare l'esperienza.

---

## Infrastruttura Statica

Attualmente la rete FireFly è preconfigurata per supportare solo tre attori (1 Nodo = 1 Attore).

L'architettura non scala dinamicamente.

# Sviluppi futuri

- Superare la rigidità dei nodi statici.
- Permettere l'ingresso di nuovi partecipanti senza dover riconfigurare l'infrastruttura dei nodi FireFly.
- Evoluzione UI con pagine dedicate agli storici acquisti/vendite



Grazie per  
l'attenzione!



<https://github.com/MarioLezzi92/Watchchain>