**ANALISI DEI REQUISITI**

**Registro delle modifiche**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **versione** | **data** | **nominativo** | **ruolo** | **descrizione** |
|  |  |  |  |  |
|  |  |  |  |  |
| 0.0.2 | 29-11-21 | Chen Xida,  Casazza Domenico | Analista,  Verificatore | Stesura §1, §2 e verifica |
| 0.0.1 | 26-11-21 | Chen Xida,  Casazza  Domenico | Analista,  Verificatore | Aggiunto §1, §2 e verifica |

**Indice**

[1 Introduzione 2](#_Toc653570522)

[1.1 Scopo documento 2](#_Toc265615813)

[2 Descrizione Generale 2](#_Toc1811484704)

[2.1 Obiettivi del Prodotto 2](#_Toc1092810349)

[2.3 Stack front-end 3](#_Toc464621001)

[2.4 Decisione del framework front-end 3](#_Toc1232320053)

[2.5 Stack back-end 3](#_Toc1990863885)

[3 Casi d’uso 4](#_Toc979853458)

[3.1 UC01 Landing Page 4](#_Toc1639762254)

[3.1.1 UC01.1 Accesso al wallett 4](#_Toc1953978603)

[3.1.2 UC01.2 Visualizzazione importo transazione 4](#_Toc974515277)

[3.1.1 UC01.3 Visualizzazione importo venditore 5](#_Toc1762730007)

[3.2 UC02 WebApp 5](#_Toc2036874138)

[3.2.1 UC02.2 Visualizzazione dati wallet 5](#_Toc871116535)

[3.2.2 UC02.3 Annullamento ordine 5](#_Toc1062501330)

# **1 Introduzione**

## **1.1 Scopo documento**

Questo documento si pone come obiettivo l’identificazione dei requisiti, sia quelli del proponente che quelli interni al team, al fine di una corretta realizzazione dell’applicazione.

Più precisamente, il documento conterrà il diagramma dei casi d’uso che riporteranno le informazioni degli attori del sistema e le interazioni con essa dove verranno descritte gli scenari principale e quelli alternativi.

# **2 Descrizione Generale**

## **2.1 Obiettivi del Prodotto**

Al giorno d’oggi, la maggior parte dei servizi di e-commerce non offre un sistema che consenta di eseguire transazioni in modo sicuro: l’acquirente può venire truffato dal venditore se, dopo il pagamento, non gli viene consegnato il prodotto o, viceversa, se l’acquirente riceve l’ordine senza pagare.

ShopChain è un applicativo in grado di sostenere un e-commerce dalla fase di pagamento a quella di consegna usando la tecnologia delle blockchain. La blockchain è incaricata di ricevere l’ammontare speso dall’acquirente in criptovaluta, consegnandola al venditore solo alla ricezione del pacco.

Al momento della consegna, l’acquirente dovrà necessariamente inquadrare il codice QR applicato sul collo che ne certifica l’avvenuta consegna. Solo a questo punto verrà effettuato il passaggio della criptovaluta dal wallet della piattaforma al wallet del venditore.

**2.2 Piattaforme di esecuzione**

|  |  |
| --- | --- |
| Prodotto | Piattaforma di esecuzione |
| Applicazione web ShopChain | Chrome, Firefox, Safari e Opera |
| Applicazione mobile ShopChain | Android e iOS |

## **2.3 Stack front-end**

|  |  |
| --- | --- |
| Tecnologia | Descrizione |
| HTML | Linguaggio di markup per la struttura |
| CSS | Linguaggio di markup per lo stile |
| Javascript | Linguaggio di programmazione lato client |
| Vue | Framework per lo sviluppo front-end |
| MetaMask | Plugin per la gestione dei wallet |

## **2.4 Decisione del framework front-end**

Per lo sviluppo dell’applicazione web, il team ha considerato i tre principali framework: Angular, React e Vue. Le principali caratteristiche di questi framework, che servono a facilitare lo sviluppo lato client dell’applicazione, sono riportate nella seguente tabella:

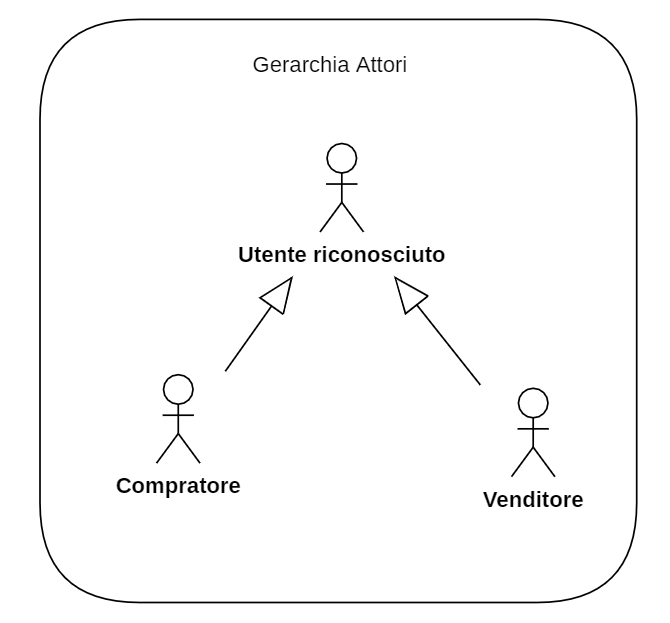
|  |  |  |  |
| --- | --- | --- | --- |
| Caratteristiche | Angular | React | Vue |
| Manipolazione UI/DOM | Sì | Sì | Sì |
| Gestione stati | Sì | Parziale | Sì |
| Routing | Sì | No | Sì |
| Gestione e validazione form | Sì | No | No |
| Http Client | Sì | No | No |

Dopo attente ricerche su queste tecnologie e basandosi anche su quanto emerso dal documento “Verbale\_29\_11\_21”, il team Oberon ha deciso di utilizzare Vue che ha come punti di forza la semplicità e una curva di apprendimento molto meno ripida rispetto a Angular e React, tuttavia riesce a soddisfare pienamente i requisiti tecnici necessari per la buona riuscita della WebApp.

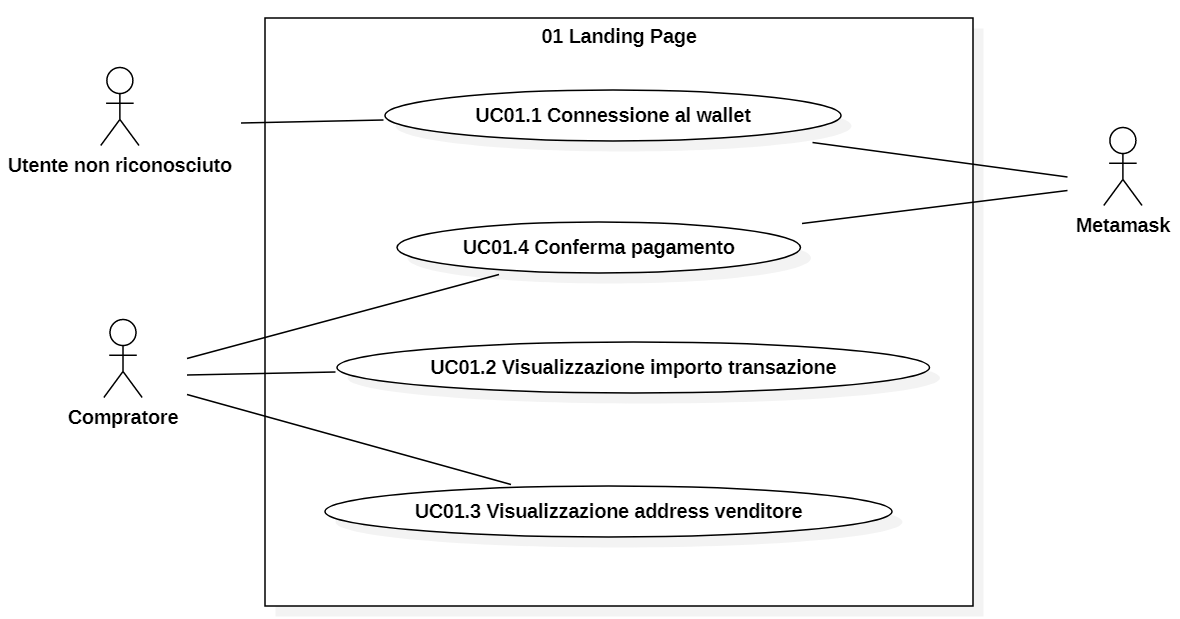
## **2.5 Stack back-end**

|  |  |
| --- | --- |
| Tecnologia | Descrizione |
| Java | Linguaggio di programmazione lato server |
| Spring | Framework per lo sviluppo back-end |
| Solidity | Linguaggio di programmazione per gli SmartContract |
| Avalanche | Blockchain |
| PostgreSQL | Database |

# **3 Casi d’uso**



## **3.1 UC01 Landing Page**



**Titolo:** UC01 Landing Page.

**Attori primari:** Utente non riconosciuto,Compratore.

**Attore secondario:** Metamask.

**Precondizione:** Il compratore ha iniziato la fase di pagamento nella piattaforma e-commerce.

**Postcondizione:**

**Scenario Principale:** Il compratore ha scelto di pagare usando le criptovalute nel sito dell’e-commerce. Dopo aver confermato tale scelta è arrivato alla nostra Landing Page e seguirà i seguenti passi per il pagamento:

1) Connessione al wallet (UC01.1).

2) Conferma pagamento (UC01.4).

3) Visualizzazione dell’importo della transazione e dell’address del venditore come informazioni di riepilogo sulla Landing Page (UC01.2 & UC01.3).

### **3.1.1 UC01.1 Connessione al wallet**

**Titolo:** UC01.1 Connessione al wallet.

**Attore primario:** Utente non riconosciuto.

**Attore secondario:** Metamask.

**Precondizione: I**l compratore non ha effettuato il collegamento tra wallet e smart contract.

**Postcondizione: I**l compratore ha effettuato il collegamento tra wallet e smart contract e può interagire con esso.

**Scenario Principale:** Il compratore vuole interagire con lo smart contract di ShopChain, quindi deve collegare il wallet ad esso per poter autorizzare successivamente le transazioni. Il compratore, quindi, cliccherà su “Connetti wallet” e si aprirà il pop-up di Metamask, dove darà il permesso allo smart contract di interagire con il proprio wallet.

### **3.1.2 UC01.2 Visualizzazione importo transazione**

**Titolo:** UC01.2 Visualizzazione importo transazione.

**Attore primario:** Compratore.

**Attore secondario:** Metamask.

**Precondizione:** Il compratore ha iniziato la fase di pagamento nella piattaforma e-commerce.

**Postcondizione:** Il compratore ha visualizzato l’importo della transazione.

**Scenario Principale:** Il compratore visualizza nella Landing Page l’importo corrispondente al/i prodotto/i selezionato/i, con eventualmente sommato il costo di consegna e le fees stimate (medie).

### **3.1.1 UC01.3 Visualizzazione address venditore**

**Titolo:** UC01.3 Visualizzazione address venditore.

**Attore primario:** Compratore.

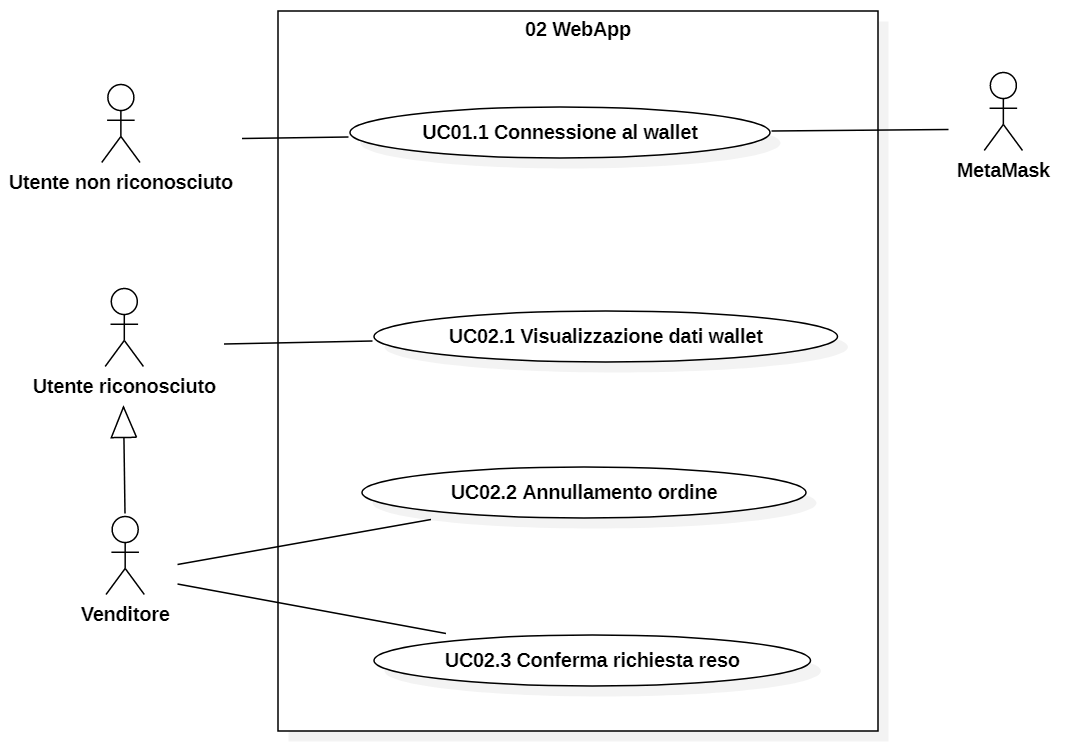
**Attore secondario:** Metamask.

**Precondizione:** Il compratore ha iniziato la fase di pagamento nella piattaforma e-commerce.

**Postcondizione:** Il compratore ha visualizzato correttamente l’address del venditore.

**Scenario Principale:** Il compratore visualizza nella Landing Page l’address del venditore nella blockchain Avalanche, in formato esadecimale (esempio: 0x7949635E2877ef8ca37B8526507AC214B0423Ebf).

## **3.2 02 WebApp**



### **3.2.1 UC02.1 Visualizzazione dati wallet**

**Titolo:** UC02.1 Visualizzazione dati wallet.

**Attore primario:** Utente riconosciuto.

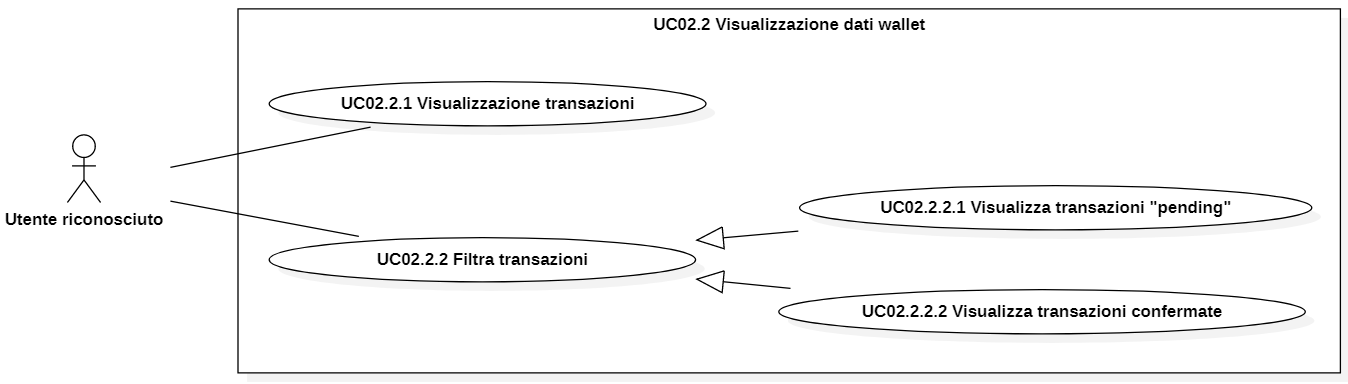
**Precondizione:** L’utente è stato riconosciuto tramite la connessione al suo wallet.

**Postcondizione:** L’utente ha visualizzato le transazioni relative al wallett collegato.

**Scenario Principale:** L’utente può:

1) visualizzare tutte le transazioni relative al wallett con cui è connesso (UC02.2.1);

2) filtrare le transazioni (UC02.2.2) per visualizzare le transazioni "pending” (UC02.2.2.1), cioè non confermate, e le transazioni confermate (UC02.2.2.2).



### **3.2.1.2 UC02.2.1 Visualizzazione transazioni**

**Titolo:** UC02.2.1 Visualizzazione transazioni

**Attore primario:** Utente riconosciuto.

**Precondizione:** L’utente è stato riconosciuto tramite la connessione al suo wallet.

**Postcondizione:** L’utente ha visualizzato le transazioni relative al wallet collegato.

**Scenario Principale:** L’utente accede alla pagina di visualizzazione di tutte le transazioni relative al wallett con cui è collegato.

### **3.2.1.2 UC02.2.2 Filtra transazioni**

**Titolo:** UC02.2.2 Filtra transazioni

**Attore primario:** Utente riconosciuto.

**Precondizione:** L’utente riconosciuto visualizza tutte le transazioni del suo wallet.

**Postcondizione:** L’utente riconosciuto visualizza le transazioni filtrate.

**Scenario Principale:** L’utente riconosciuto può decidere se:

1) Filtrare e visualizzare le transazioni “pending” (UC02.2.2.1)

2) Filtrare e visualizzare le transazioni “confermate” (UC02.2.2.2)

### **3.2.1.2 UC02.2.2.1 Visualizzazione transazioni “pending”**

**Titolo:** UC02.2.2.1 Visualizzazione transazioni “pending”

**Attore primario:** Utente riconosciuto.

**Precondizione:** L'utente riconosciuto ha scelto di visualizzare le transazioni “pending”.

**Postcondizione:** L’utente riconosciuto visualizza solo le transazioni in stato di “pending”.

**Scenario Principale:** L’utente riconosciuto vuole visualizzare solo le transazioni in stato di “pending” e clicca quindi il pulsante che ha la funzionalità di filtrare le transazioni che hanno questo stato.

### **3.2.1.2 UC02.2.2.2 Visualizzazione transazioni “confermate”**

**Titolo:** UC02.2.2.2 Visualizzazione transazioni “confermate”

**Attore primario:** Utente riconosciuto.

**Precondizione:** L'utente riconosciuto ha scelto di visualizzare le transazioni “confermate”.

**Postcondizione:** L’utente riconosciuto visualizza solo le transazioni in stato di “confermato”.

**Scenario Principale:** l’utente riconosciuto vuole visualizzare solo le transazioni in stato di “confermato” e clicca quindi il pulsante che ha la funzionalità di filtrare le transazioni che hanno questo stato.

### **3.2.2 UC02.3 Annullamento ordine**

**Titolo:** UC02.3 Annullamento ordine.

**Attore primario:** Venditore.

**Precondizione:** Il Venditore desidera annullare uno degli acquisti sul proprio e-commerce.

**Postcondizione:** I soldi precedentemente depositati dal Compratore nello smart contract vengono restituiti al compratore.

**Scenario Principale:**

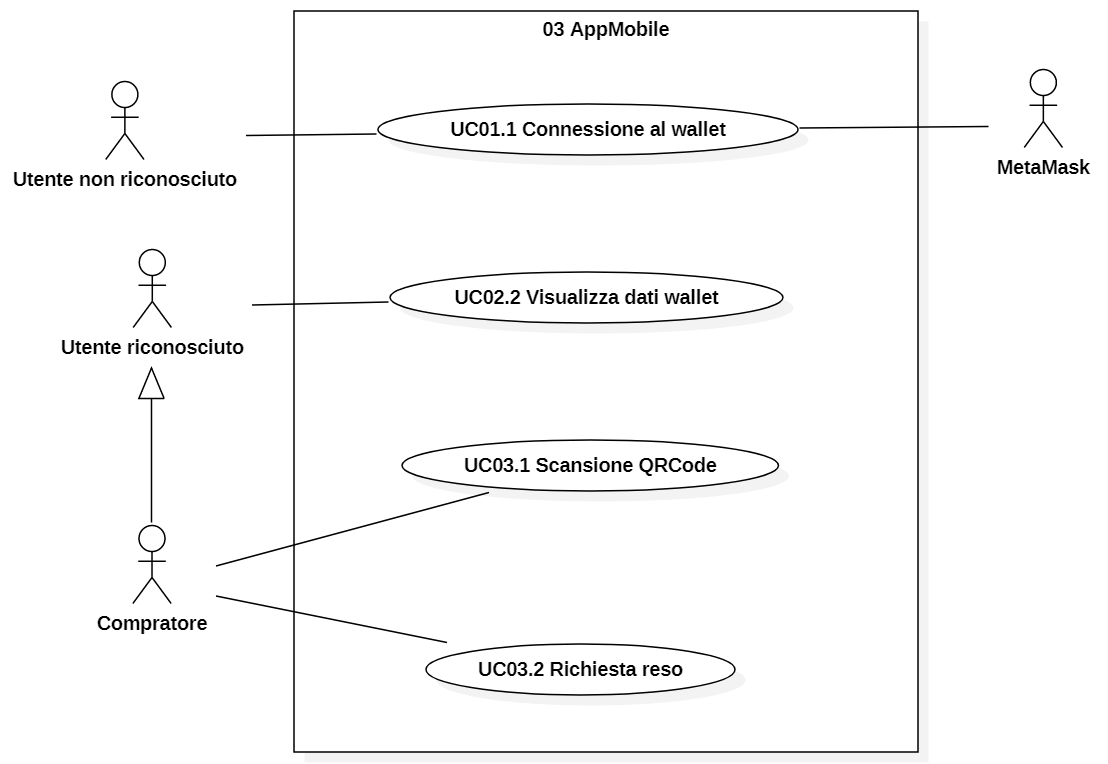
1) Il venditore visualizza le transazioni attive e individua quella da annullare;

2) Clicca sul bottone annulla e conferma la scelta;

3) La WebApp attiva un metodo dello smart contract che procederà a rendere il deposito al wallet del compratore;

4) Il Compratore viene notificato via App del fatto che la transazione è stata vanificata e la merce non arriverà.

# **3.3 03 App Mobile**



### **3.3.1 UC03.1 Scansione QRCode**

**Titolo:** UC03.1 Scansione QRCode.

**Attore primario:** Compratore.

**Precondizione:** Il Compratore è già connesso al wallet.

**Postcondizione:** Il Compratore ha sbloccato le cryptovalute depositate nello smart contract.

**Scenario Principale:**

1. Il pacco con QR raggiunge l’abitazione del compratore;
2. l’utente accede all’app e seleziona “conferma transazione”;
3. dopo aver aperto la fotocamera inquadra il codice ed attende che l’app lo identifichi;
4. con le informazioni contenute l’App ritrova la transazione attiva e invoca un metodo dello smart contract che procede all’inoltro delle cryptovalute nel wallet del venditore;
5. La transazione “pending" diventa “completata” nella web app del venditore.

**3.3.2 UC03.2 Richiesta reso**

**Titolo:** UC03.2 Richiesta reso.

**Attore primario:** Compratore.

**Precondizione:** Il Compratore è già connesso al wallet ed ha effettuato acquisti su ShopChain in passato.

**Postcondizione:** Il Compratore ha effettuato una richiesta di reso.

**Scenario Principale:** Il Compratore visualizza le transazioni effettuate in passato. Da qui può cliccare su un pulsate apposito tramite il quale potrà inizializzare la procedura di reso.