Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea Magistrale in Informatica

Realizzazione di una Dapp per la tracciabilità del vino su Ethereum

Relatore:
Prof.
STEFANO FERRETTI

Presentata da:
VITTORIA CONTE

Sessione II
Anno Accademico 2023/2024

Abstract

Negli ultimi anni si è diffuso sempre più, tra le varie aziende, l'utilizzo di metodi di tracciamento dati nella filiera agroalimentare volti a garantire non solo la sicurezza alimentare ma anche la sostenibilità della filiera. Per quanto riguarda il settore vitivinicolo nello specifico tali tecnologie sono state anche utilizzate dalle varie aziende per evitare la contraffazione dei propri prodotti. In questa tesi ci si è posti l'obiettivo di creare un sisitema che contenesse le informazioni di prodotti che provengano da più produttori, quindi che contenesse le informazioni di più tipologie di vino provenienti da diversi produttori. Questo sistema deve essere poi in grado di comparare le informazioni dei vini al suo interno che devono anche essere facilmente reperibili e al sicuro.
L'applicazione sviluppata, tramite blockchain e smartcontract, risponde alla esigenze e presenta le caratteristiche richieste, dimostrando che è possibile comparare diversi vini sulla base di parametri in comune a basso costo. Tale applicazione rappresenta inoltre una base sulla quale poter aggiungere funzionalità in merito al tracciamento dati.

Contents

1 Stato dell'arte **7**
- 1.1 Introduzione . . . 7
- 1.2 DLT . . . 7
  - 1.2.1 Concetto . . . 7
  - 1.2.2 Design . . . 8
  - 1.2.3 Caratteristiche . . . 8
  - 1.2.4 Proprietà . . . 11
- 1.3 Blockchain . . . 11
  - 1.3.1 Sicurezza . . . 12
  - 1.3.2 Validazione . . . 12
  - 1.3.3 Funzioni hash nelle transazioni . . . 12
- 1.4 Smartcontract . . . 13
  - 1.4.1 Sfide . . . 13
- 1.5 Dapp . . . 14
- 1.6 Tracciamento nella filiera agroalimentare . . . 15
  - 1.6.1 Motivazioni . . . 15
  - 1.6.2 Funzionamento della sicurezza . . . 15
  - 1.6.3 Quanto i sisitemi di tracciamento sono utili a svolgere il loro compito . . . 16
- 1.7 Tracciabilità del vino . . . 17
  - 1.7.1 Introduzione . . . 17
  - 1.7.2 Il ruolo della blockchain . . . 17
  - 1.7.3 Certificazioni e blockchain . . . 18
  - 1.7.4 Contraffazione . . . 19
  - 1.7.5 Vantaggi . . . 19
  - 1.7.6 Impatti . . . 20

2 Design **23**
- 2.1 Introduzione . . . 23
- 2.2 Requisiti . . . 23
  - 2.2.1 Introduzione . . . 23

2.2.2 Sistema di tracciabilità 23
2.2.3 Applicazione 24
2.3 Problematiche 24
2.3.1 Comparazione 24
2.4 Architettura 25
2.4.1 Backend 25
2.4.2 Frontend 26

3 Implementazione **27**
3.1 Introduzione 27
3.2 Sviluppo 27
3.2.1 Introduzione 27
3.2.2 Parametri 28
3.2.3 Tecnologie utilizzate 31
3.2.4 Back-end 33
3.2.5 Front-end 39

4 Sperimentazione **49**
4.1 Introduzione 49
4.2 Attivazione e Accesso 50
4.2.1 Attivazione 50
4.2.2 Accesso 51
4.3 Inserimento 52
4.4 Comparazione 53

5 Conclusioni **63**
5.1 Introduzione 63
5.2 Obiettivi 63
5.3 Risultati 64
5.3.1 Blockchain e Smartcontract 64
5.3.2 Transazioni 64
5.3.3 Comparazione 64
5.4 Upgrade possibili 65

6 Ringraziamenti **67**

List of Figures

1.1 Caratteristiche delle blockchain private, pubbliche e di consorzio [10] [3] 19
1.2 Vantaggi e svantaggi di blockchain private, pubbliche e di consorzio [10] [3] 20

2.1 Architettura del progetto 25

4.1 Prima Pagina 54
4.2 Esemipo di inserimento dati 55
4.3 Risultato inserimento 56
4.4 Blocco 57
4.5 Elenco transazioni effettuate correttamente 57
4.6 Dettaglio transazione effettuata correttamente 58
4.7 Esempio di confronto 1 59
4.8 Esempio di confronto 2 59
4.9 Esempio di confronto 3 60
4.10 Esempio di confronto 4 61

Introduzione

L'obiettivo di questa tesi è quello di sviluppare un sistema di tracciamento dati, sicuro e affidabile, che memorizzi le informazioni relative a più tipi di bottiglie di vino per per poter permettere successivamente a un utente che ne sta valutando l'acquisto di poterle paragonare.
Attualmente i sistemi di tracciamento utilizzati per i dati riguardanti il settore vinicolo sono delle blockchain create ad-hoc per la memorizzazione dei dati riguardanti le singole bottiglie di vino nello specifico. Vengono creati degli smartcontract che memorizzano nella blockchain i dati di una determinata bottiglia e che permettono successivamente, prevalentemente tramite qr-code stampato sull'etichetta, di andare a leggere i dati riguardanti i vari passaggi avvenuti per arrivare a produrre quella bottiglia secondo i parametri definiti per quel tipo di bottiglia di vino. I tipi di dato che possono quindi essere visualizzati sono specifici del tipo di lavorazione richiesta per arrivare a produrre quella bottiglia.
Ci si propone, in questa tesi, di creare un unico sistema che possa memorizzare gli stessi dati certificati per più tipologie di bottiglie di vino che un utente possa verificare online, tramite nome univoco, e successivamente comparare con altri. L'utilizzo di un unico sistema prevede l'utilizzo di parametri che siano comuni a tutti i tipi di bottiglie di vino e quindi di individuare i passaggi comuni a tutti i tipi di vino che possono essere inseriti in blockchain. In questo modo l'utente finale potrà esprimere una preferenza su una base dati omogenea e non eterogenea come accadrebbe se il paragone dovesse avvenire tra dati provenienti da due diverse blockchain.

Chapter 1

Stato dell'arte

1.1 Introduzione

L'utilizzo di DLT, in particolare delle blockchain, per il tracciamento nella filiera agroalimentare è una pratica comune negli ultimi anni. Ciò nasce dall'esigenza di permettere ai consumatori finali di tracciare il cibo non solo per la qualità ma anche per la sicurezza alimentare, l'impatto ambientale e la contraffazione.

1.2 DLT

Con il termine DLT (Distributed Ledger Tecnology), si fa riferimento alla tecnologia dei registri distribuiti, ovvero a quella tecnologia che permette a più parti di scambiare valori, informazioni e documenti in modo sicuro e efficiente. I ledger infatti, i "libri mastri", sono dei registri che presentano tutte le transazioni effettuate nel tempo con la garanzia che la diffusione, la verifica e la validazioni di questi sia sicura grazie all'adozione di protocolli come la crittografia end-to-end. I metodi utilizzati per criptare le informazioni devono essere basati su algoritmi che coinvolgano tutti o parte dei partecipanti alla rete per far si che questi possano concordare sull'insieme delle transazioni da validare.

1.2.1 Concetto

Quando si parla ci concetto di DLT si sta descrivendo la struttura e il funzionamento base dei progetti DLT ad un livello di astrazione elevato.

1.2.2 Design

Il design dei progetti di DLT non sono altro che una descrizione astratta dei concetti della Distributed Ledger Tecnology, dei processi e dei valori concreti. Questi presentano, tra di loro, importanti differenze che li rendono adeguati per alcune applicazioni e inadeguati ad altre.

1.2.3 Caratteristiche

Le caratteristiche dei progetti DLT possono essere di natura tecnica o amministrativa. Le caratteristiche tecniche limitano le possibili modifiche future alle caratteristiche amministrative.
Di seguito l'elenco delle caratteristiche:

- **Interoperabilità**: La capacità di interagire tra i registri distribuiti e con altri servizi di dati.
- **Mantenibilità**: Il grado di efficacia ed efficienza con cui un registro distribuito può essere mantenuto operativo.
- **Smartcontract Turing completi**: Il supporto agli smartcontract Turing completi all'interno di un progetto DLT.
- **Supporto token**: I possibili usi dei token all'interno di un registro distribuito.
- **Payload della transazione**: La dimensione del payload in una transazione.
- **Rintracciabilità**: La misura in cui i carichi utili delle transazioni possono essere tracciati cronologicamente in un progetto DLT.
- **Visibilità del contenuto della transazione**: La possibilità di visualizzare il contenuto di una transazione in un design DLT. Ciò implica la difficoltà di mappare mittenti e destinatari nelle transazioni in maniera univoca in quanto non identificabili dall'utente.
- **Controllore del nodo di verifica**: La misura in cui l'identità dei controllori di nodo che convalidano la transazione è verificata prima di unirsi a un registro distribuito.
- **Audibility**: Il grado in cui una terza parte indipendente può valutare la funzionalità di un registro distribuito.
- **Compliance**: L'allineamento di un registro distribuito e il suo funzionamento rispetto alla politica dei requisiti.

- **Grado di Decentramento**: Il grado di decentralizzazione di un registro distribuito si riferisce al numero di controllori indipendenti di convalida dei nodi meno il numero di controllori che controllano più nodi di validazione medi diviso per numero totale di controllori di nodo nella rete DLT.

- **Meccanismo di incentivazione**: Una struttura in creata per motivare il comportamento corretto del nodo, che assicura la fattibilità di una gestione a lungo termine di un registro distribuito.

- **Responsabilità**: L'esistenza di una persona fisica o giuridica che può essere soggetta a contenzioso relativamente al comportamento del registro distribuito.

- **Intervallo di creazione dei blocchi**: Tempo che intercorre tra la creazione di blocchi consecutivi.

- **Limite di dimensione del blocco**: Il valore di una dimensione massima fissa di un blocco.

- **Conferma di latenza**: Il tempo che intercorre tra l'inclusione di un'operazione in un registro e il momento in cui sono state incluse sufficienti operazioni successive nel registro in modo che la probabilità di future manipolazioni dell'iniziale operazione diventi trascurabile.

- **Consumo di risorse**: Gli sforzi computazionali richiesti per gestire un registro distribuito.

- **Ritardo di propagazione**: Il tempo tra la presentazione di una transazione (o blocco) e la sua propagazione a tutti i nodi.

- **Scalabilità**: La capacità di un registro distribuito di gestire in modo efficiente le risorse necessarie, diminuendone o aumentandone la quantità.

- **Tasso di blocco obsoleto**: Il numero di blocchi che sono stati generati in un periodo di tempo, ma non allegati alla catena principale del registro distribuito.

- **Throughput**: Il numero massimo di transazioni che possono essere aggiunte a un registro distribuito in un dato intervallo di tempo.

- **Latenza validazione transazione**: Il tempo necessario per validare una transazione mediante la validazione di nodi.

- **Commissione della transazione**: L'attore che inizia la transazione può o deve pagare per le operazioni commissionate.
- **Facilità di configurazione del nodo**: La facilità di configurare e aggiungere un nuovo nodo al registro distribuito.
- **Facilità d'uso**: La semplicità di accesso e di lavoro con un registro distribuito.
- **Sostegno per dispositivi vincolati**: La misura in cui i dispositivi con capacità di calcolo limitate possono partecipare a un registro distribuito.
- **Atomicity**: Lo stato in cui le transazioni sono completamente eseguite o non eseguite.
- **Autenticità**: Il grado di correttezza dei dati che sono memorizzati su un registro distribuito.
- **Disponibilità**: La probabilità che un registro distribuito funzioni correttamente in qualsiasi momento nel tempo.
- **Resistenza alla censura**: La probabilità che una transazione in un registro distribuito sia intenzionalmente cancellata da un terzo o malintenzionalmente modificata.
- **Riservatezza**: Grado di prevenzione dell'accesso non autorizzato ai dati.
- **Coerenza**: L'assenza di contraddizioni tra gli stati del registro mantenuto da tutti i nodi che partecipano al registro distribuito.
- **Durata**: La proprietà per cui i dati impegnati per il registro distribuito non saranno persi.
- **Tolleranza ai guasti**: La proporzione costante massima di errori, operazioni effettuate da malintenzionati o nodi imprevedibili che un registro distribuito può compensare per funzionare correttamente.
- **Integrità**: Il grado di protezione delle transazioni nel registro distribuito contro la modifica o l'eliminazione non autorizzata o involontaria.
- **Isolamento**: La proprietà tale per cui le transazioni non si influenzano a vicenda durante la loro esecuzione.

- **Non repudiazione**: La difficoltà di negare la partecipazione alle transazioni.
- **Affidabilità**: La capacità di un sistema o componente di svolgere le sue funzioni richieste in condizioni definite per un periodo determinato di tempo.
- **Forza della crittografia**: La difficoltà di rompere gli algoritmi crittografici utilizzati nella progettazione di DLT.

1.2.4 Proprietà

Le proprietà dei registri distribuiti non sono altro che gruppi di caratteristiche condivise da ogni progetto di DLT. Sebbene tutti i progetti DLT coprano tutte le proprietà DLT, i progetti DLT non devono coprire tutte le caratteristiche DLT.
Di seguito le proprietà:

- **Flessibilità**: I gradi di libertà nella distribuzione delle applicazioni su e nella personalizzazione di un registro distribuito.
- **Opacità**: Il grado in cui l'uso e il funzionamento di un registro distribuito non possono essere tracciati.
- **Prestazioni**: Il compimento di un determinato compito su un registro distribuito con l'uso efficiente del calcolo di risorse e di tempo.
- **Policy**: La capacità di guidare e verificare il corretto funzionamento di un registro distribuito.
- **Praticità**: La misura in cui gli utenti di un registro distribuito possono raggiungere i loro obiettivi rispetto ai vincoli socio-tecnici della pratica quotidiana.
- **Sicurezza**: La probabilità che il funzionamento del registro distribuito e dei dati memorizzati non sia compromesso.

1.3 Blockchain

La blockchain è un particolare sistema di DLT che ha i dati delle transazioni che vengono memorizzati in ordine temporale in blocchi collegati e poi inseriti all'interno del registro. Questo fa si che le informazioni archiviate sul registro e quindi validate siano inalterabili in quanto legate allo storico delle

transazioni precedenti. Una successiva manipolazione del dato in questo caso verrebbe infatti subito evidenziata e non ne consentirebbe la successiva validazione.

1.3.1 Sicurezza

Il concetto di blockchain, coniato nel 2008 con la pubblicazione del documento intitolato "Bitcoin: A Peer to Peer Electronic Cash System", ha risolto un problema di sicurezza. [7] Nello specifico, con la blockchain è stato possibile stabilire la fiducia in un sistema distribuito: con essa è possile creare un deposito distribuito di documenti con indicazione temporale in cui nessuna parte può manomettere il contenuto dei dati o i timestamp senza rilevarli.

1.3.2 Validazione

La validazione della blockchain, nota anche come mining, viene generalmente completata utilizzando un algoritmo di consenso che istituisce le regole a cui i nodi devono aderire quando effettuano l'autenticazione dei blocchi. [4] Il consenso richiede che i nodi partecipanti ricevano una risposta positiva da tutti gli altri nodi aderendo all'ordine di transazione prescritto. Questo permette ai nodi di decidere se il blocco sarà aggiunto alla catena. Mentre nessuna autorità centralizzata verifica le transazioni a catena, tutte le transazioni sono completamente sicure e verificate con precisione grazie al protocollo di consenso, una componente chiave di qualsiasi blockchain. Gli algoritmi di consenso richiedono che tutti i peer della rete raggiungano un accordo reciproco riguardo alla condizione del registro distribuito. Questo assicura l'affidabilità blockchain e costruisce la fiducia tra i partecipanti sconosciuti in una rete di calcolo distribuito. In sostanza, gli algoritmi di consenso assicurano che ogni nuovo blocco della catena sia l'unica versione autenticata concordata da tutti i nodi. Molti algoritmi di consenso sono stati creati per l'uso in blockchain. Ognuno ha punti di forza e debolezze uniche, rendendolo adatto a usi specifici.

1.3.3 Funzioni hash nelle transazioni

Le transazioni sono unità di dati contenenti i dettagli della transazione più un timestamp. Entrambi possono essere rappresentati come numeri o stringhe. Una blockchain può essere pensata come una tabella con tre colonne, in cui ogni riga rappresenta una transazione distinta: la prima colonna memorizza il timestamp della transazione, la seconda colonna memorizza i dettagli della

transazione, e la terza colonna memorizza un hash della transazione corrente con i suoi dettagli e l'hash della transazione precedente. Quando un nuovo record viene inserito in una blockchain, l'ultimo hash calcolato viene trasmesso a tutte le parti interessate. Non è necessario che ogni parte conservi una copia dell'intera cronologia delle transazioni, ma è sufficiente che lo facciano alcune parti. Dato che tutti conoscono l'ultimo hash, chiunque può verificare che i dati non siano stati alterati poiché sarebbe impossibile senza ottenere un hash diverso e quindi non valido. L'unico modo per manomettere i dati conservando l'hash sarebbe quello di trovare una collisione nei dati, e questo è computazionalmente impossibile. Richiederebbe così tanta potenza di calcolo che è praticamente antieconomico.
Un hash può essere pensato come una versione crittografata della stringa originale dalla quale è impossibile derivare la stringa originale. Matematicamente, un hash è prodotto da una funzione di hash, f, che deve avere due proprietà importanti: la dimensione dello spazio di input e lo spazio di output deve essere grande; deve essere praticamente impossibile trovare collisioni, cioè due ingressi x1 e x2 che producono lo stesso output f(x1)=f(x2). Esempi di funzioni hash sono gli algoritmi SHA1, SHA128, SHA512, i quali possono prendere qualsiasi stringa come input e produrre sempre una stringa di output che è una rappresentazione esadecimale del numero di uscita della funzione con un numero fisso di cifre.

1.4 Smartcontract

Gli smartcontract sono dei software basati sulla blockchain. Costituiti da codice crittografico, vengono utilizzati per automatizzare l'esecuzione di un accordo offrendo quindi la garanzia che i termini di tale "contratto" vengano rispettati.
Nel caso di questa tesi lo smartcontract deve poter rispondere delle esigenze della Dapp e quindi garantire che le operazioni richieste da essa siano sicure.

1.4.1 Sfide

Le sfide che lo smartcontract deve affrontare attualmente sono le seguenti:

- **Vulnerabilità di reingresso**: Questo problema si verifica quando un attaccante utilizza una funzione di chiamata ricorsiva per effettuare più prelievi ripetitivi. Ciò può comportare comportamenti inaspettati, anche col finire di consumare tutto il gas (la valuta) in possesso di un utilizzatore.

- **Dipendenza dall'ordine delle operazioni**: Si verifica quando diverse operazioni dipendenti che invocano lo stesso contratto sono incluse in un blocco. Sappiamo che i miner possono impostare un ordine arbitrario tra le transazioni, vale a dire quindi che lo stato finale del contratto dipende da come il miner classifica le transazioni. Un avversario può lanciare con successo un attacco se tali transazioni non sono state eseguite nel giusto ordine.

- **Dipendenza dal timestamp**: Generalmente, il timestamp è impostato all'ora corrente del sistema locale dei miner. Tuttavia, il miner può cambiare questo valore mentre altri miner accettano il blocco. Il problema di sicurezza si presenta quando il timestamp viene utilizzato come condizione di attivazione per eseguire azioni specifiche (ad esempio, l'invio di denaro) perché l'attaccante può utilizzare diversi timestamp per manipolare il risultato del contratto. Tale contratto viene comunemente definito come contratto dipendente dal timestamp.

- **Mancanza di fonti attendibili di dati**: Gli smartcontract richiedono talvolta informazioni provenienti da risorse esterne. Tuttavia, l'affidabilità di tali informazioni non può essere garantita.

- **Questioni relative alla privacy**: Poiché tutta la cronologia delle transazioni è memorizzata sulla blockchain ed è visibile a chiunque, è teoricamente possibile ottenere informazioni private degli utenti analizzandone le strutture nei grafici delle transazioni. Tale tipologia di attacco viene definita deanonimizzazione.

1.5 Dapp

Una dapp è un'applicazione decentralizzata ovvero un'applicazione software che funziona su un sistema di calcolo decentralizzato, tipicamente su una blockchain. A differenza delle applicazioni tradizionali che funzionano su server centralizzati, le dapp operano su una rete peer-to-peer, non sono quindi controllate da nessuna singola entità o da un individuo.
Essendo la dapp un'applicazione costruita su una rete decentralizzata essa consiste in una parte di backend, rappresentato da uno smartcontract, e un frontend, rappresentato dall'interfaccia utente. Le dapp sono "senza permessi", il che significa che chiunque è libero di utilizzarle. Di fatto la maggior parte delle dapp includono smartcontract scritti da altri. Sono anche trasparenti e "senza fiducia", il che significa che chiunque può verificarne l'autenticità e la funzionalità.

La maggior parte delle dapp funziona attraverso l'interazione di tre componenti:

- Smartcontract
- Blockchain
- Token: poichè le azioni di una dapp richiedono gas, questo viene pagato tramite token nativo della blockchain. Molte dapp utilizzano vari tipi di criptovalute o asset digitali per eseguire azioni.

1.6 Tracciamento nella filiera agroalimentare

1.6.1 Motivazioni

I sistemi di tracciamento utilizzati nella filiera agroalimentare sono sistemi il cui scopo è garantire la food safety, ovvero la prevenzione di malattie di origine alimentare, e la food security ovvero la sicurezza per un consumatore di comprare un prodotto specifico e non un prodotto contraffatto. Tali sistemi possono però in alcuni casi impattare negativamente sui costi e portare a un aumento del prezzo del prodotto tracciato.

1.6.2 Funzionamento della sicurezza

I dati della filiera alimentare, necessari per la rintracciabilità degli alimenti, in un sistema di tracciamento DLT, sono conservati come registrazioni digitali di transazioni, collegate in tutta la blockchain.
Non è possibile impedire agli attori maligni di cambiare i dati prima della registrazione sulla blockchain, o di utilizzare dati errati, ma il sistema fornisce comunque un potente deterrente per diversi meccanismi di frode tradizionali. Il DLT è infatti un deterrente per i crimini, nella filiera agroalimentare, grazie ai record teoricamente immutabili e aperti a tutti gli attori che hanno permesso di azione sul registro. Tutti gli attori di una catena di approvvigionamento globale possono conseguentemente identificare e segnalare gli attori maligni, mentre i record immutabili forniscono una base dati più sicura. Il DLT aiuta quindi a garantire la "catena delle prove", non sostituisce nessuna delle procedure standard del settore e della regolamentazione richieste e ampiamente adottate per controllare le frodi e le adulterazioni. Impedisce agli attori maligni di nascondere le loro azioni cambiando i record. Gli attori malevoli potrebbero essere rapidamente esposti e sanzionati in un ecosistema di filiere più aperto. Esiste però un rischio reale che parti innocenti possano

essere ingiustamente sanzionate da altri attori della catena di approvvigionamento, o che sanzioni sproporzionate possano essere applicate ai trasgressori. Le accuse non comprovate possono facilmente sorgere in mercati ad alta velocità e volume, dove gli standard di qualità possono essere soggettivi.
Le imprese private non hanno lo stesso potere dei regolatori alimentari: il loro l'accesso ai dati riservati può essere concesso solo con il consenso di tutti gli altri attori della catena, e in un processo di retroazione passo dopo passo verso il basso e in tutta la catena. Su questa base, solo una blockchain chiusa potrebbe essere adatta per il tracciamento globale e complesso di cibo su tutte le catene di approvvigionamento. Questo perché l'accesso ai dati lungo la blockchain può essere concessa solo con il permesso da parte dei proprietari di dati. Le norme e gli orientamenti internazionali sono necessari per concordare quali dati sono memorizzati su un registro blockchain alimentare, il meccanismo di consenso, riservatezza e modalità di accesso da parte di altri.[8]
Perchè la tecnologia della blockchain sia fruibile ed efficace è fondamentale che l'accesso alla tecnologia blockchain sia mantenuto semplice, a bassi costi, e facile da implementare e distribuire. Ciò richiede quindi degli standard globali per la crittografia dei dati, le architetture DLT e l'accesso.

1.6.3 Quanto i sisitemi di tracciamento sono utili a svolgere il loro compito

I sistemi di tracciamento dati, quali i DLT, sono l'unica opzione possibile solo in alcuni casi, come il caso in cui ci sia la necessità di cancellare i dati da parte dell'owner ma non da parte degli stakeholder.
In altri casi i DLT competono per il tracciamento dei dati con altri sistemi come i database centralizzati combinati alle firme digitali.[9]
Inoltre i DLT sono utilizzabili solo per il tracciamento della filiera, non per altri dati di tracciamento come comunicare degli status del prodotto in tempo reale. Questi devono essere utilizzati quindi solo: quando sono tecnologicamente necessari, quando sono l'opzione più semplice o meno costosa.
Nello specifico lo sviluppo dei DLT con smartcontract è altamente dipendente da quanto è facile trasporre la realtà in un contratto che permetta di avere zero errori e la prova delle transazioni.

1.7 Tracciabilità del vino

1.7.1 Introduzione

Tra i settori dell'agri-food, quello vinicolo è uno tra quelli che utilizzano i sistemi di tracciamento, quali le blockchain. Le motivazioni che spingono all'utilizzo di questo sistema per questo settore, oltre a qulle elencate per l'agrifood in generale, riguardano la contraffazione dei vini.

1.7.2 Il ruolo della blockchain

La produzione del vino è un processo complesso che comporta più fasi, dalla coltivazione delle uve all'imbottigliamento. Ciascuna di queste fasi presenta delle sfide che richiedono un'attenta considerazione per garantire la qualità del prodotto finale. Negli ultimi anni si è assistito a una crescita di interesse per la comprensione del processo di produzione del vino e lo scambio di informazioni nel corso dell'intera catena del valore e i fattori che ne influenzano le prestazioni. [6] L'adozione delle teconologie a circuito integrato nel settore agricolo può certamente rafforzare la trasformazione su larga scala, ridurre i costi di produzione e aumentare la crescita degli investimenti. In particolare l'adozione della tecnologia blockchain promuove un'agricoltura elettronica sostenibile. La blockchain può quindi svolgere un ruolo fondamentale e potrebbe avere un ampio campo di applicazione, tenendo conto dell'importanza per i consumatori di conoscere l'origine di un prodotto agroalimentare e dell'utilità di questa tecnologia nella lotta contro la contraffazione e la falsificazione dei prodotti. Di fatto attualmente la blockchain viene utilizzata nel settore agricolo per:

- Ottimizzare i processi nella catena di approvvigionamento
- Migliorare la rintracciabilità, ovvero la capacità di rintracciare e seguire gli alimenti attraverso tutte le fasi
- Migliorare la sicurezza alimentare
- Ridurre i tempi e i costi delle transazioni
- Ridurre le frodi alimentari
- Ridurre i processi inefficienti

La blockchain può inoltre:

- Migliorare i profitti degli agricoltori

- Promuovere le imprese etiche
- Ridurre l'impatto ambientale
- Evitare un uso diffuso di pesticidi e fertilizzanti, che possono causare la presenza di residui pericolosi per la salute umana

La blockchain da quindi la capacità ai fornitori, agricoltori, produttori, rivenditori e governi di identificare e confinare gli elementi contaminati e seguire il suo percorso lungo la catena di fornitura.[1]
Tra le produzioni agroalimentari, l'industria vinicola è l'unica che ha indicato un percorso di crescita per una migliore attuazione delle pratiche sostenibili nella produzione globale e il processo di distribuzione, e per questo motivo l'industria del vino ha un numero elevato di programmi di sostenibilità, processi di certificazione e standard di sostenibilità. Questi programmi differiscono per tipologia e portata. Nello specifico, i programmi di valutazione della sostenibilità sono destinati a pianificare il miglioramento e mitigare gli impatti.
Nonostante ciò, le certificazioni o gli standard di sostenibilità non sono ancora stati comunemente adottati da tutta la catena del vino e le valutazioni degli indicatori non sono diffuse. Gli strumenti di gestione della sostenibilità che consentono la raccolta di dati sulla sostenibilità dei vigneti e delle cantine per confrontarli tra i produttori hanno però un alto grado di adozione, suggerendo che lo sviluppo sostenibile potrebbe essere raggiunto attraverso lo sviluppo di strumenti di gestione interattivi. [5]

1.7.3 Certificazioni e blockchain

Le certificazioni di sostenibilità del vino si concentrano generalmente sulla valutazione e sul miglioramento della sostenibilità a livello di prodotto e cantina. Il sistema di tracciabilità relativamente rapido ed efficace offerto dalla blockchain può essere adottato dalle aziende vinicole e implementato come strumento di supporto alla raccolta dei dati nelle certificazioni dei vini. La blockchain può sostenere la raccolta e l'archiviazione dei dati nella produzione, nel trattamento, nel monitoraggio del materiale. Questa mappatura della catena di approvvigionamento, completata da dati affidabili e verificabili, può supportare la valutazione e la riduzione delle impronte di carbonio, agronomiche e idriche che sono anche al centro del sistema di certificazione dei vini.[5] Di conseguenza, la blockchain potrebbe essere adottata da certificazioni, programmi o standard di sostenibilità del vino come strumento per monitorare le emissioni di gas serra e la gestione delle acque. Il sistema di tracciabilità rapido e conveniente offerto da blockchain mostra anche la

promessa futura nel rilevare i fornitori non etici, pratiche lavorative sleali e prodotti contraffatti.

1.7.4 Contraffazione

La contraffazione del vino è, dall'inizio degli anni '90, rapidamente aumentata. Le frodi sui vini rappresentano quasi il 5% dell'attuale mercato secondario mondiale, che ammonterebbe a 15 miliardi di dollari. Tra le varie frodi vitivinicole vi è la contraffazione e la rietichettatura di vini meno costosi in vini costosi e altamente collezionabili sono i tipi di frode più diffusi. Negli ultimi tempi, l'industria del vino sta prestando maggiore attenzione rispetto alla prevenzione sulla contraffazione dei vini, consentendo la tracciabilità nella catena di approvvigionamento del vino. Tra questi vi è la rintracciabilità, un metodo che permette a chiunque di verificare il processo globale comprese: le materie prime, le condizioni di trasporto e di magazzinaggio, la lavorazione, la distribuzione e la vendita nella catena di approvvigionamento del vino.[6]
Il livello desiderato di protezione dalla contraffazione a monte e a valle che un proprietario di un marchio intende garantire ai clienti attraverso la BC è il fattore chiave da considerare nella progettazione dei sistemi di protezione. Le soluzioni basate su blockchain potrebbero avvantaggiare il settore vinicolo, soprattutto nel campo della prevenzione della contraffazione e dell'autenticità, ma ci sono anche opportunità per soluzioni basate su incentivi e gamification mirate a modelli di business nuovi e innovativi.

	To Read	To Send Transaction	To Participate in Consensus Process	The Mechanism	Other Characteristics
Public "fully decentralized"	anyone	anyone	anyone	PoW (Proof of Work), PoS (Proof of Stake)	Secured by crypto economics; the degree of influence is proportional to the quantity of economic resources
Consortium "partially decentralized"	Anyone/pre-defined nodes	pre-defined nodes	pre-defined nodes	The majority have to sign every block	
Private "fully private"	Anyone/restricted	centralized	centralized		Likely applications include database management, auditing, etc. internal to a single company

Figure 1.1: Caratteristiche delle blockchain private, pubbliche e di consorzio [10] [3]

1.7.5 Vantaggi

La blockchain permette di raccogliere dati e informazioni rilevanti per il monitoraggio e il miglioramento della sostenibilità delle aziende vinicole che già possiedono una certificazione di sostenibilità. Queste hanno però una una scarsa familiarità con le applicazioni blockchain (57,1%) e solo il 14% di queste sostengono l'intenzione di investire in una blockchain nei prossimi

	Advantages	Disadvantages
Public "fully decentralized"	Protects users from developers' influence; Trust of the system (blockchain) Censorship resistance Network effect; Immutability nearly impossible to tamper	Can reduce the block time till 15 s (Ethereum) instead of 2 h (Bitcoin), but still it is more than in the cases of private or consortium blockchains
Consortium "partially decentralized"	Easy changes, revert transaction, modify balances; The validators are known; Cheap transactions; Nodes can be trusted to be very well-connected;	Immutability could be tampered
Private "fully private"	Easy changes, revert transaction, modify balances; The validators are known; Cheap transactions; Nodes can be trusted to be very well-connected; Greater level of privacy if read permissions are restricted.	Immutability could be tampered In some cases, in order to efficiently work the BC, some heterogeneous assets from different industries need to be on the same database, which is difficult to happen in private BCs.

Figure 1.2: Vantaggi e svantaggi di blockchain private, pubbliche e di consorzio [10] [3]

anni. Il miglioramento della tracciabilità e trasparenza lungo la catena di approvvigionamento aumenta la fiducia dei consumatori che si riflette nella crescita delle vendite, e i principali costi sono legati alla complessità nella gestione dei dati.[1] Una piattaforma basata su blockchain introduce dati affidabili comuni, riducendo la duplicazione dei dati e migliorando la visibilità della catena di fornitura. La piattaforma sostiene l'instaurazione di un clima di fiducia tra le parti, ma non sostituisce i requisiti che le organizzazioni devono soddisfare per stabilire una posizione di fiducia. [2]

1.7.6 Impatti

La blockchain rende la catena di approvvigionamento più trasparente e consente la consegna di alimenti di alta qualità con un impatto sociale e ambientale decrescente. Incoraggia un sistema trasparente che avvantaggi le varie parti interessate, in particolare i consumatori, dandogli la possibilità di conoscere tutte le informazioni necessarie sul prodotto. Essa è un'utile strumento per garantire un sistema di tracciabilità e proteggere la produzione da qualsiasi tipo di frode e contaminazione. Pertanto, la capacità di rintracciare e seguire il prodotto dal fornitore al consumatore fa una differenza significativa per le condizioni di salute dei consumatori e anche tra la vita e la morte, di conseguenza, per il successo delle imprese interessate. Inoltre, la blockchain semplifica notevolmente la condivisione delle informazioni tra gli attori lungo la catena di fornitura e digitalizza i processi che danno la possibilità di rintracciare e rintracciare il prodotto in un tempo significativamente breve e con costi bassi. La trasparenza della blockchain e la sua capacità di seguire i prodotti lungo l'intera catena di approvvigionamento consentono quindi di identificare in tempo tutti i prodotti contaminati e, di conseguenza, di richiamare dal mercato non tutta la produzione ma solo quelli pericolosi. Ciò riduce lo spreco alimentare, le esigenze di trasporto verso il mercato e

viceversa e, quindi, l'uso delle risorse naturali in relazione agli effetti ambientali dannosi.[1]
L'adozione della tecnologia blockchain ha anche ripercussioni in termini di impatto sociale grazie alla creazione di nuovi modelli di business, la riorganizzazione dei modelli esistenti e l'introduzione di nuovi sistemi e nuove competenze. L'adozione di una blockchain presuppone, infatti, il coinvolgimento di vari attori operanti lungo la catena di approvvigionamento che opereranno secondo un ciclo virtuoso, impegnandosi in transazioni peer-to-peer, riducendo la corruzione e aumentando la responsabilità, e creare valore per le imprese e per l'insieme delle comunità locali.

Chapter 2

Design

2.1 Introduzione

Prima di poter effettivamente creare un sistema di tracciabilità di dati, sicura e affidabile, per poter paragonare le informazioni di più vini è necessario prima determinare quali tecnologie utilizzare e come utilizzarle per raggiungere l'obiettivo. Ciò impone un'attenta analisi dei requisiti che deve avere il sistema che ci si pone di creare e le problematiche di tale sistema, solo in una fase successiva sarà possibile definire l'architettura finale.

2.2 Requisiti

2.2.1 Introduzione

Per poter creare un sistema di tracciamento dati, riguardanti le bottiglie di vino, che provengono da diversi produttori per far si che un utente, che non fa parte della catena vitivinicola, possa consultare è necessario creare un'applicazione a supporto di tale sisitema.
Sia il sisitema di tracciamento che l'applicazione devono ripondere di più requisiti per far si che l'obiettivo venga raggiunto.

2.2.2 Sistema di tracciabilità

Il sistema di tracciabilità deve poter garantire che i dati non possano essere in alcun modo inseriti o modificati da agenti esterni al sistema. Questo requisito è fondamentale in quanto anche solo un dato errato renderebbe la comparazione tra vini non veritiera e renderebbe quindi inutile tutto il sisitema.

Un altro requisito necessario per il raggiungimento dell'obiettivo è che il sisitema di tracciabilità sia in grado di memorizzare tutte le informazioni necessarie alla tracciabilità in maniera corretta e in modo che si possano facilmente consultare. Se per un qualche motivo non fosse possibile avere a dispisizione tutti i dati necessari alla comparazione, o averli ma in un formato non comprensibile all'utente, renderebbe la comparazione solo parzialmente utile.

2.2.3 Applicazione

L'applicazione deve essere in grado di comunicare in tempo reale con il sistema di traccibilità implementato e deve poter rendere facilmente accessibili all'utente finali tutte le funzionalità che esso può effettuare. L'inserimento e la visualizzazione delle informazioni devono poter essere esguite senza alcun tipo di latenza per permettere sempre una comparazione completa con tutti i vini a sistema. La visualizzazione delle informazioni deve inoltre permettere all'utente di visualizzare le informazioni in maniera che siano facilmente comparabili tra di loro.

2.3 Problematiche

Per poter creare un sisitema di tracciamento e un'applicazione che possano funzionare correttamente devono essere valutate le problematiche emerse dall'analisi.

2.3.1 Comparazione

Per poter comparare correttamente due oggetti deve valere il presupposto che questi possano essere paragonati secondo standard comuni. Nel caso delle bottiglie di vino, questo presupposto viene a mancare nel momento in cui le informazioni in possesso sono ottenute secondo diversi parametri.
Il sistema di tracciamante che si vuole andare a creare, con il fine di comparare successivamente i dati, prende le proprie informazioni da fonti diverse. Non vi è alcuna garanzia infatti che vengano usati gli stessi criteri per rilevare le stesse informazioni o che tutte le informazioni necessarie alla comparazione vengano rilevate.
Per gestire al meglio questa criticità diventa quindi necessario svolgere una selezione dei parametri il più accurata possibile, in modo da minimizzarne le problematiche.

2.4 Architettura

Dopo aver analizzato i requisiti e le criticità del progetto che ci si pone di portare a termine si può delineare l'architettura di esso.
La decisione è ricaduta su un design che preveda l'utilizzo di una parte di backend per la gestione dei dati e del tracciamento della filiera vitivinicola in se, ed una parte di frontend che permetta di interagire con il backend in maniera semplice e senza alcuna conoscenza tecnica.

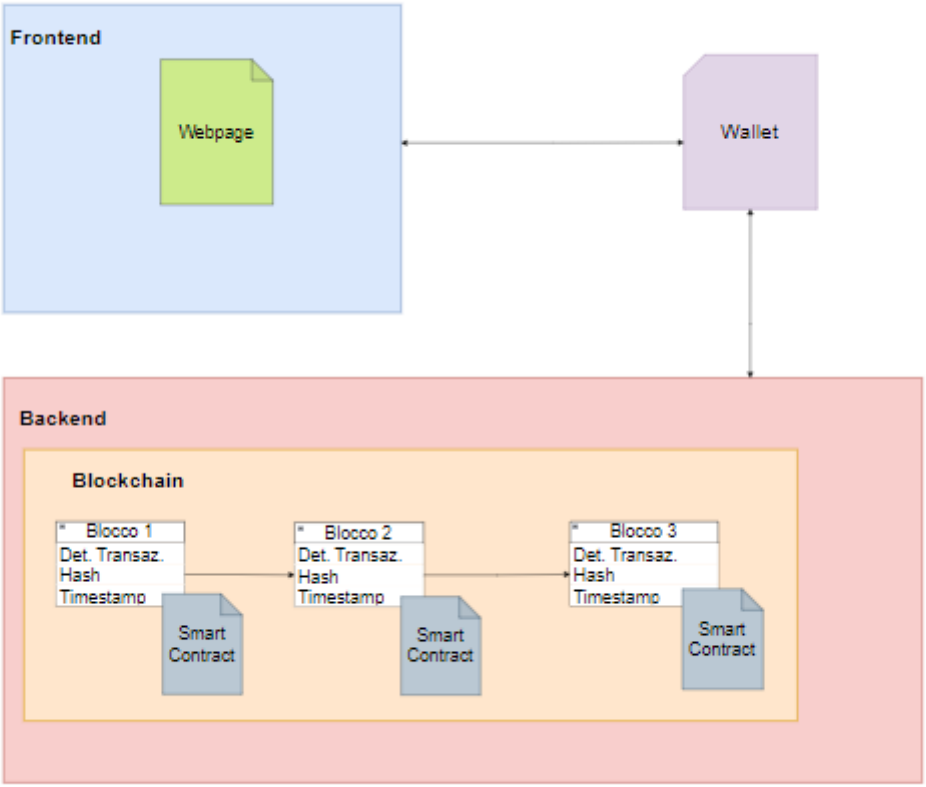

Figure 2.1: Architettura del progetto

2.4.1 Backend

Per questo progetto la parte di backend deve poter prevedere un sistema che permetta di memorizzare, in maniera sicura, i dati da paragonare successivamente. Questo sistema deve poter memorizzare quindi le informazioni

in maniera uniforme e deve poter permettere successivamente di comunicare con il frontend.
A livello tecnologico ci sarà una blockchain, che rappresenterà il sistema che avrà in memoria i dati in maniera sicura e che li renderà disponibili. Su questa blockchain sarà presente uno smartcontract, che rapprensenterà la struttura che permetterà ai dati di essere memorizzati e resi disponibili secondo i canoni che permetteranno poi il confronto.

2.4.2 Frontend

Per questo progetto la parte di frontend deve permettere all'utente di effettuare tutte le operazioni previste sul backend senza mai interagirvi direttamente.
A livello tecnico questo presuppone la costruzione di una Dapp, ovvero di un'applicazione che sia in grado di connettersi alla blockchain e di poter interagire con essa. Per il collegamento con il frontend dovrà essere utilizzato un wallet che permetterà all'utente di connettersi alla blockchain tramite il proprio identificativo.

Chapter 3

Implementazione

3.1 Introduzione

Per poter creare un sistema di tracciamento sicuro delle informazioni, che permetta all'utente finale di paragonare i dati tracciati in maniera chiara e sicura, è stato scelto di utilizzare una Dapp, ovvero un'applicazione simile alle app tradizionali che si appoggia a un server centralizzato sfruttando le piattaforme blockchain e il loro network distribuito.
Per fare ciò è necessario sviluppare un DLT, ovvero un sistema distribuito con libro mastro per memorizzare le informazioni. Tale sistema permette di memorizzare i dati in maniera sicura, attraverso metodi crittografici, e di poterli consultare e modificare indipendentemente su tutti i nodi.
Tra i vari sistemi di DLT vi è la blockchain che, tramite smartcontract, farà da appoggio alla Dapp per poter memorizzare e di seguito visualizzare i dati in modo sicuro.

3.2 Sviluppo

3.2.1 Introduzione

Lo sviluppo della Dapp che risponda alle esigenze poste da questa tesi prevede più fasi:

- Individuare le fasi di produzione del vino dalla vendemmia alla vendita
- Individuare i parametri comuni utilizzabili per il confronto di vini di diverso tipo

- Determinare le caratteristiche necessarie affinchè la Dapp risponda alle esigenze poste inizialmente da questa tesi
- Sviluppare uno smartcontract che corrisponda alle esigenze della Dapp con le adeguate tecnologie (fase back-end)
- Sviluppare un'interfaccia user friendly che implementi le funzionalità richieste con le adeguate tecnologie (fase front-end)

3.2.2 Parametri

Per poter far si che la Dapp possa confrontare le varie bottiglie di vino inserite è necessario determinare i parametri in comune ovvero i passaggi della lavorazione del vino che tutti i vini hanno in comune. I parametri individuati andranno successivamente memorizzati nella blockchain tramite uno smartcontract che dovrà assicurarsi dell'integrità di essi.
Per poter procedere a una selezione è necessario comprendere le fasi di produzione del vino e in cosa consistono. Di seguito i passaggi:

- **Vendemmia**: Momento di raccolta dell'uva, può avvenire in maniera meccanizzata grazie a macchinari specializzati o manualmente. La durata e il periodo di vendemmia incidono sulla qualità delle uve raccolte come il metodo utilizzato per raccoglierle.
- **Selezione**: Momento di selezione delle uve da utilizzare per il vino.
- **Diraspatura**: Operazione che consente di separare gli acini dai raspi per favorire la fuoriuscita del mosto e avviarli poi alla pigiatura.
- **Pigiatura**: Fase durante la quale si estrae il succo dagli acini.
- **Svinatura**: Il processo di svinatura può avvenire per via meccanica, usando impianti appositi detti "svinatori", oppure in maniera più tradizionale, con la tecnica del travaso del vino.
  Durante questo processo il liquido viene delicatamente separato dai sedimenti solidi presenti nel fondo del recipiente di fermentazione.
- **Torchiatura**: Lavorazione compiuta con il torchio, allo scopo di assoggettare a una certa compressione il materiale in lavorazione. Dopo la svinatura le parti solide vengono torchiate, al fine di recuperare la parte di vino rimasta a contatto con le bucce, e tutta la massa liquida viene stoccata in botti, barrique, contenitori in acciaio inossidabile o altri vasi vinari per la fase successiva.

- **Maturazione**: è il termine corretto per quello che viene comunemente definito invecchiamento del vino, quella fase di affinamento che avviene in botte prima del suo imbottigliamento. La maturazione di un vino è il periodo necessario al vino per armonizzare le sue componenti in un insieme armonico. L'affinamento ne è l'ultima fase.

- **Fermentazione**: è un processo chimico naturale che si verifica quando i lieviti presenti sulle bucce d'uva, e quelli aggiunti precedentemente, reagiscono con lo zucchero contenuto nel mosto, lo convertono in alcool e anidride carbonica e danno vita al vino.

- **Travaso**: Il travaso è un'operazione enologica fondamentale per la stabilità chimica e biologica del vino e per la sua longevità. Si realizza spostando il vino da un contenitore all'altro per separarlo dal deposito di vario genere che si forma dopo la fermentazione.

- **Imbottigliamento**: Riempimento delle bottiglie di vino.

- **Conservazione**: Conservazione del vino imbottigliato, che deve avvenire in determinate condizioni a differenza del vino.

Una volta analizzati i passaggi di cui sopra è stato deciso di spezzare le fasi in più parametri per una migliore precisione e per un miglior confronto. Come si può notare nella lista sottostante infatti di molte fasi è specificato l'inizio, la fine, e il luogo. Questo permette di tener traccia dei tempi, poichè per determinati vini una fase come la maturazione può durare mesi, per altri settimane. Si riesce anche a tener traccia di eventuali spostamenti di luogo del prodotto fatti per effettuare i vari passaggi previsti nella filiera, come la fermentazione, in un luogo e eseguire il resto del processo in un altro.
Di seguito i parametri implementati:

- Nome del vino, che deve essere univoco

- Nome della cantina, inteso come il produttore

- Indirizzo della cantina, inteso come sede principale della cantina di produzione

- Indicazione geografica (e.g. D.O.C.G.)

- Particella di coltivazione

- Lotto di produzione

- Uve utilizzate, possono essere di uno o più tipi a differenza del vino

- Inizio della vendemmia, data
- Fine della vendemmia, data
- Luogo di ricezione dell'uva una volta conclusa la vendemmia
- Giorno in cui è stata ricevuta l'uva una volta conclusa la vendemmia
- Giorno e ora di inizio diraspatura delle uve
- Giorno e ora di fine diraspatura delle uve
- Luogo di diraspatura della uve
- Giorno e ora di inizio pigiatura delle uve
- Giorno e ora di fine pigiatura delle uve
- Luogo di pigiatura delle uve
- Giorno e ora di inizio della torchiatura delle uve
- Giorno e ora di fine della torchiatura delle uve
- Luogo della torchiatura delle uve
- Giorno e ora di inizio svinatura
- Giorno e ora di fine svinatura
- Luogo di svinatura
- Giorno e ora di inizio del processo di maturazione e affinamento
- Giorno e ora di fine del processo di maturazione e affinamento
- Luogo del processo di maturazione e affinamento
- Tipo di botte utilizzata per il processo di maturazione e affinamento
- Lieviti utilizzati
- Giorno e ora di inizio della fermentazione del vino
- Giorno e ora di fine della fermentazione del vino
- Luogo di fermentazione del vino
- Giorno e ora di inizio del travaso

- Giorno e ora di fine del travaso
- Luogo del travaso
- Giorno e ora di inizio dell'imbottigliamento
- Giorno e ora di fine dell'imbottigliamento
- Luogo di imbottigliamento
- Giorno e ora di inizio di stoccaggio del vino da parte del produttore
- Giorno e ora di fine di stoccaggio del vino da parte del produttore
- Luogo di stoccaggio del vino da parte del produttore
- Giorno e ora di inizio di stoccaggio del vino da parte del venditore
- Luogo di stoccaggio del vino da parte del venditore

Parametri come il nome del vino, della cantina o l'indicazione geografica non fanno propriamente parte delle fasi della vinificazione ma sono comunque importanti informazioni se si parla del tracciamento delle bottiglie di vino.
Non è stato messo nessun tipo di controllo per il formato di dati inserito e si fa completo affidamento all'owner della blockchain, unico individuo che può inserire dati, per la correttezza di essi.
Sono stati scelti parametri che riguardano il processo di vinificazione escludendo quindi i dati che riguardano le caratteristiche del vino come la gradazione alcolica, il gusto e i tipi di abbinamento culinario.

3.2.3 Tecnologie utilizzate

Per fare la Dapp sono state esaminate diverse tecnologie. È stato scelto di utilizzare come piattaforma Ethereum, scelta che ha portato all'utilizzo di Hardhat come ambiente di sviluppo e Solidity come linguaggio per lo smartcontract.
Inoltre, per la parte di frontend, è stato scelto di utilizzare il linguaggio JavaScript e, nello specifico, la libreria React, poichè creati con lo specifico compito di creare interfacce web.

Ethereum

Ethereum è una piattaforma decentralizzata grazie alla quale è possibile creare e pubblicare smartcontract peer-to-peer. La criptovaluta a esso legata è Ether.

Solidity

Solidity è un linguaggio di programmazione designato per lo sviluppo di smartcontract che funzionino su Ethereum.

Hardhat

Hardhat è un ambiente di sviluppo per Ethereum grazie al quale è possibile compilare smartcontract, testarli tramite log e sviluppare network.

Metamask

Metamask è un portafoglio software di criptovaluta utilizzato per interagire con la blockchain di Ethereum. Consente agli utenti di accedere al proprio portafoglio Ethereum tramite un'estensione del browser o un'app mobile, che può quindi essere utilizzata per interagire con applicazioni decentralizzate.
Per poter far interagire il back-end con il front-end è necessario creare nel proprio portafoglio virtuale un account il cui indirizzo sia presente nel network dove è caricato lo smartcontract.

JavaScript

JavaScript è un linguaggio di programmazione orientato agli eventi e multi paradigma, utilizzato lato client web e lato server per la creazione di RESTful API, applicazioni desktop, siti web e applicazioni web, tramite funzioni di script invocate da eventi innescati a loro volta in vari modi dall'utente sulla pagina web in uso.

React

React è una libreria open-source, front-end, JavaScript per la creazione di interfacce utente.

Node.js

Node.js è un sistema runtime open source multipiattaforma orientato agli eventi per l'esecuzione di codice JavaScript. La maggior parte dei moduli base sono scritti in JavaScript ed è possibile scrivere nuovi moduli nello stesso linguaggio di programmazione.

3.2.4 Back-end

Smartcontract

Una volta definiti i parametri che si vogliono poter paragonare è necessario poterli salvare sulla blockchain non solo in maniera sicura ma anche in modo che siano facilmente comparabili.
La struttura dello smartcontract in questo caso deve far si che i dati siano correttamente memorizzati, accessibili facilmente dall'utente e facilmente comparabili.
Va quindi creato uno smartcontract, con Solidity, che utilizzi una struttura (struct) con le informazioni decise in precedenza.
Di seguito il codice Solidity che crea la struct Bottle utilizzata per memorizzare le informazioni delle singole bottiglie:

```
1  //Definizione dati da avere sulla bottiglia di vino
2  struct Bottle{
3      string nomeVino;
4      string nomeCantina;
5      string indirizzoCantina;
6      address owner;
7      string indicazioneGeografica;
8      string particellaColtivazione;
9      string lottoProduzione;
10     string uveUtilizzate;
11     string vendemmiaInizio;
12     string vendemmiaFine;
13     string ricevimentoUvaLuogo;
14     string ricevimentoUvaGiorno;
15     string diraspaturaInizio;
16     string diraspaturaFine;
17     string diraspaturaLuogo;
18     string pigiaturaInizio;
19     string pigiaturFine;
20     string pigiaturaLuogo;
21     string torchiaturaInizio;
22     string torchiaturaFine;
23     string torchiaturaLuogo;
24     string svinaturaInizio;
25     string svinaturaFine;
26     string svinaturaLuogo;
27     string maturazioneAffinamentoInizio;
28     string maturazioneAffinamentoFine;
29     string maturazioneAffinamentoLuogo;
```

```
30      string maturazioneAffinamentoBotte;
31      string lieviti;
32      string fermentazioneInizio;
33      string fermantazioneFine;
34      string fermantazioneLuogo;
35      string travasiInizio;
36      string travasiFine;
37      string travasiLuogo;
38      string imbottigliamentoInizio;
39      string imbottigliamentoFine;
40      string imbottigliamentoLuogo;
41      string stoccaggioProduttoreInizio;
42      string stoccaggioProduttoreFine;
43      string stoccaggioProduttoreLuogo;
44      string stoccaggioVenditoreInizio;
45      string stoccaggioVenditoreLuogo;
46
47  }
```

Come è possibile vedere dal codice soprastante, la definizione di una struttura prevede la definizione del nome di questa tramite parola chiave struct (in questo caso quindi struct Bottle). Successivamente, tra parenetesi quadre, vanno definite tutte le informazioni presenti al suo interno e la tipologia di esse in modo chiaro ed univoco. Questo permette allo smartcontract di assicurare che una determinata variabile corrisponda a un specifico tipo di dato. Il tipo string, usato per la maggior parte delle informazioni come la variabile chiamata nomeVino, prevede che l'informazione sia formata da caratteri alfanumerici. Il tipo address invece prevede che l'informazione corrisponda a un indirizzo. La variabile chiamata owner è stata definita come variabile address in quanto rappresenterà l'indizzo del prprietario della blockchain (quindi 0xf39fd6e51aad88f6f4ce6ab8827279cfffb92266 nel caso specifico di questo progetto).
La struttura definita come sopra è poi da mappare, quindi da poter identificare, sulla base di un parametro. La scelta è caduta sul nome della bottiglia di vino (nomeVino) in quanto di più facile accesso per l'utente eseguire le operazioni di ricerca su di esso piuttosto che su altri dati salvati. Si assume che tale dato sia formato da il nome del vino e un codice di cui l'utente sia in possesso in quanto vuole fare il confronto.
Di seguito il codice Solidity che permette di identificare gli elementi della struttura sulla base del nome:

```
1  //Mapping della struct
2  mapping(string =>Bottle) bottle;
```

Viene utilizzata la parola chiave del linguagio Solidity mapping per specificare che la struttura definita come Bottle debba essere mappata tramite stringa e richiamata tramite appellativo bottle.
Sono state conseguentemente create le funzioni di inserimento dati nella struttura e le funzioni di riperimento dati in base all'assunto che il nome sia univoco. L'inserimento dei dati deve contenere tutti i dati previsti mentre per il reperimento dati le funzioni sono molteplici. Per facilitare la comparazione all'utente non solo queste funzioni reperiscono i dati in base al nome ma a differenza della funzione verranno reperiti dei dati piuttosto che altri. Ciò faciliterà, in fase di visualizzazione, la comparazione tra dati.
Di seguito la definizione della funzione, in Solidity, che permette l'inserimento di nuove bottiglie di vino:

```
1  //Funzione per l'aggiunta di una bottiglia alla blockchain secondo
       la struct
2  function addBottle(
3  //Vanno definiti tutti i parametri della struct
4     string calldata _nomeVino,
5     string memory _nomeCantina,
6     string memory _indirizzoCantina,
7     address _owner,
8     string memory _indicazioneGeografica,
9     string memory _particellaColtivazione,
10    string memory _lottoProduzione,
11    string memory _uveUtilizzate,
12    string memory _vendemmiaInizio,
13    string memory _vendemmiaFine,
14    string memory _ricevimentoUvaLuogo,
15    string memory _ricevimentoUvaGiorno,
16    string memory _diraspaturaInizio,
17    string memory _diraspaturaFine,
18    string memory _diraspaturaLuogo,
19    string memory _pigiaturaInizio,
20    string memory _pigiaturFine,
21    string memory _pigiaturaLuogo,
22    string memory _torchiaturaInizio,
23    string memory _torchiaturaFine,
24    string memory _torchiaturaLuogo,
25    string memory _svinaturaInizio,
26    string memory _svinaturaFine,
27    string memory _svinaturaLuogo,
28    string memory _maturazioneAffinamentoInizio,
29    string memory _maturazioneAffinamentoFine,
```

```
30      string memory _maturazioneAffinamentoLuogo,
31      string memory _maturazioneAffinamentoBotte,
32      string memory _lieviti,
33      string memory _fermentazioneInizio,
34      string memory _fermantazioneFine,
35      string memory _fermantazioneLuogo,
36      string memory _travasiInizio,
37      string memory _travasiFine,
38      string memory _travasiLuogo,
39      string memory _imbottigliamentoInizio,
40      string memory _imbottigliamentoFine,
41      string memory _imbottigliamentoLuogo,
42      string memory _stoccaggioProduttoreInizio,
43      string memory _stoccaggioProduttoreFine,
44      string memory _stoccaggioProduttoreLuogo,
45      string memory _stoccaggioVenditoreInizio,
46      string memory _stoccaggioVenditoreLuogo
47  ) public {
48  //qui viene specificato che il _nomeVino e' l'identificativo per la
        bottiglia di vino che si sta inserendo
49      bottle[_nomeVino]=Bottle(_nomeVino,
50      _nomeCantina,
51      _indirizzoCantina,
52      _owner,
53      _indicazioneGeografica,
54      _particellaColtivazione,
55      _lottoProduzione,
56      _uveUtilizzate,
57      _vendemmiaInizio,
58      _vendemmiaFine,
59      _ricevimentoUvaLuogo,
60      _ricevimentoUvaGiorno,
61      _diraspaturaInizio,
62      _diraspaturaFine,
63      _diraspaturaLuogo,
64      _pigiaturaInizio,
65      _pigiaturFine,
66      _pigiaturaLuogo,
67      _torchiaturaInizio,
68      _torchiaturaFine,
69      _torchiaturaLuogo,
70      _svinaturaInizio,
71      _svinaturaFine,
```

```
72      _svinaturaLuogo,
73      _maturazioneAffinamentoInizio,
74      _maturazioneAffinamentoFine,
75      _maturazioneAffinamentoLuogo,
76      _maturazioneAffinamentoBotte,
77      _lieviti,
78      _fermentazioneInizio,
79      _fermantazioneFine,
80      _fermantazioneLuogo,
81      _travasiInizio,
82      _travasiFine,
83      _travasiLuogo,
84      _imbottigliamentoInizio,
85      _imbottigliamentoFine,
86      _imbottigliamentoLuogo,
87      _stoccaggioProduttoreInizio,
88      _stoccaggioProduttoreFine,
89      _stoccaggioProduttoreLuogo,
90      _stoccaggioVenditoreInizio,
91      _stoccaggioVenditoreLuogo
92      );
93  }
```

La funzione di inserimento, definita in quanto tale grazie alla parola chiave function, è stata chiamata addBottle in modo da renderne chiaro lo scopo. In seconda battuta sono stati definiti i dati che questa funzione richiede, rendendo quindi indispensabile definire tali informazioni nello stesso modo con cui sono state definite le informazioni della struttura in quanto la funzione necessita di ricevere, quando chiamata, tutti i dati da inserire nella struttura per poterli effettivamente inserire. Inoltre, è stato specificato se questi dati debbano essere memorizzati in uno spazio temporaneo, tramite parola chiave memory, o in uno spazio non modificabile e non persistente tramite parola chiave calldata. Successivamente è stato specificato, con bottle[_nomeVino]=Bottle(..) che i dati esplicitati, quindi quelli che poi verranno inseriti dall'utente, devono essere inseriti nella struttura bottle con identificativo uguale al valore presente nel campo _nomeVino. Le variabili sono infatti poi ripetute nell'ordine di inserimento tra le parentesi tonde. Tali variabili sono state inoltre chiamate allo stesso modo in cui sono state definite le informazioni all'interno della struttura aggiungendo un underscore all'inizio per differenziarle e allo stesso tempo rendere chiaro all'interno del codice il significato e la funzione che esse hanno.
L'inserimento di una bottiglia di vino in blockchain prevede quindi che l'applicazione, connessa alla blockchain, possa accedere alla funzione di inserimento visualiz-

zata sopra. Questa funzione richiede che le variabili che verranno poi inserite nella struttura vengano passate dall'applicazione nell'ordine prestabilito per un corretto inserimento. Nella sezione di frontend è quindi necessario passare tali variabili in questo ordine e far si che all'utente sia chiaro dove inserire quale parametro. Inoltre, come è possibile vedere sopra, in questa fase di inserimento viene esplicitato quanto definito nella fase precedente: la bottiglia di vino viene identificata tramite nome.
È altresì necessario che il frontend possa passare l'indirizzo di chi sta effettuando l'operazione in quanto solo il possessore della blockchain può effettuare questa operazione.
Di seguito il codice Solidity che impone che l'unica utenza che possa apportare modifiche sia l'owner della blockchain:

```
1 //Serve per non far modificare i dati a nessuno ad eccezione dell'
    owner della blockchain
2 modifier onlyOwner {
3       require(msg.sender == owner);
4       _;
5   }
```

Come è possibile vedere sopra, il linguaggio Solidity permette di utilizzare le parole chiave modifier e require per specificare che l'unica entità che può apportare modifiche è l'owner del contratto, che deve quindi essere chi comunica con la blockchain (definito all'interno del linguaggio come msg.sender).
La visualizzazione dei dati presenti in blockchain è stata implementata tramite più funzioni. Tali funzioni richiedono tutte il nome della bottiglia come parametro in quanto identificativo unico senza il quale sarebbe impossibile reperire il dato desiderato. Avendo quindi l'identificativo, le varie funzioni recuperano i dati facendo semplicemente riferimento alla struttura, al nome e al tipo di parametro della struct che si vuole reperire.
La mancanza di tale informazione renderà nulla l'operazione che si vuole effettuare.
Di seguito come esempio due funzioni di reperimento dati definite in Solidity nello smartcontract:

```
1 //Dichiarazione della funzione che serve per prendere i dati base e
    visualizzarli
2 function get_bottleBasics(string memory nomeVino) view public returns (
    string memory){ //Serve avere il nome
3 string memory totalData = string.concat('Nome␣della␣Cantina:␣', bottle[
    nomeVino].nomeCantina,
4   '␣Indirizzo␣della␣Cantina:␣', bottle[nomeVino].indirizzoCantina,
5   '␣Indicazione␣Geografica:␣', bottle[nomeVino].indicazioneGeografica,
6   '␣Particella␣di␣Coltivazione:', bottle[nomeVino].particellaColtivazione,
```

```
7       ' Lotto di Produzione: ', bottle[nomeVino].lottoProduzione,
8       ' Uve Utilizzate: ',  bottle[nomeVino].uveUtilizzate  );
9       return totalData;
10  }
```

```
1   //Funzione per reperire i dati relativi alla vendemmia
2   function get_Vendemmia(string memory nomeVino) view public returns (
        string memory){
3       string memory totalData = string.concat(
4           bottle[nomeVino].vendemmiaInizio,
5           ' - ', bottle[nomeVino].vendemmiaFine);
6           return totalData;
7       }
```

Le funzioni di reperimento dati, come per quella di inserimento, vengono definite con function e denominate in maniera tale che sia comprensibile cosa fanno, nello specifico che tipologia di dati reperiscono. Inoltre, essendo questa una funzione il cui scopo è produrre un risultato che dovrà poi essere reso visibile all'utente, viene definita la tipologia di dati che deve restituire la funzione, in questo caso una stringa. Come evidente dagli esempi soprastanti, l'output presenta già la descrizione del dato che viene visualizzato permettendo uniformità nella visualizzazione tra gli stessi parametri di due bottiglie. Sarà quindi necessario, nella fase di frontend, rendere chiaro all'utente dove inserire il nome delle bottiglie e fornire una zona di visualizzazione dati in modo che i tipi dato scelti per la comparazione siano facilmente comparabili. Entrambe le funzioni sono inoltre definite public, ovvero sono richiamabili al di fuori dello smartcontract. Diversamente non sarebbe possibile utilizzare se non per operazioni da effettuare esclusivamente all'interno di esso.

3.2.5 Front-end

Introduzione

Per il corretto funzionamento della Dapp è necessario sviluppare un'interfaccia web che permetta, da parte dell'utente, di inserire i dati all'interno della blockchain e per tutti gli altri utenti di visualizzare i dati tramite il nome del vino.

Inserimento

L'inserimento dei dati può essere fatto solo ed esclusivamente dall'owner della blockchain. Pertanto, per migliorare la user experience di tutti gli altri utenti, è stato scelto di far apparire e scomparire tale sezione all'accorrenza. Così

facendo chi non deve inserire i dati non dovrà mai vedere tale funzionalità. Di seguito il codice JavaScript che permette al pulsante di rendere visibile la form di inserimento:

```
<div>
  <h4>Se sei il proprietario della blockchain clicca qui per poter aggiungere le bottiglie di vino acquistabili</h4>
  <p></p>
  //Bottone che fa apparire e scomparire la sezione di inserimento
<button className="btn btn-warning"
      type="button"
      onClick={() => setIsVisible(!isVisible)}>
 {isVisible ? 'Hide content' : 'Show content'}
</button>
<p></p>
```

Nello specifico, onClick= {() => setIsVisible(!isVisible)}> isVisible ? 'Hide content' : 'Show content' è la combinazione di parole chiave che permette di rendere visibile e non, al click, il pulsante nel quale è inserito.
Per poter procedere alla scrittura dei dati sulla blockchain è stata prevista quindi una form di inserimento che permette di inserire tutti i dati del vino. Tale form prevede l'inserimento di tutti i dati scelti per la comparazione del vino per poter effettuare correttamente la scrittura del dato tramite oggetti di inserimento opportunamente segnalati.
Di seguito parte del codice JavaScript che definisce la form di inserimento dati:

```
//Form di inserimento
  <div className="form-group">
      <label>Nome Bottiglia</label> //Etichetta dato da inserire
      <input className="form-control" type="text" name="nomeVino " required /> //inserimento obbligatorio
    </div>
    //qui ci sono gli altri dati nella stessa forma
    <div className="form-group">
      <input className="btn btn-primary" type="submit" value="Insert" /> //bottone per confermare l'inserimento
    </div>
```

Come è possibile vedere dal codice soprastante è stata utilizzata la parola chiave JavaScript required per obbligare l'utente a effettuare l'inserimento all'interno dei componenti form dove specificato. Inoltre è specificato il tipo di pulsante come submit, rendendolo il pulsante che fa si che vengano richiamate le funzioni di inserimento con i dati inseriti nel form. Tutto questo fa si

che alla fine della compilazione, solo quando tutti i campi saranno compilati, sarà possibile cliccare sul pulsante di inserimento per richiamare la funzione di addBottle e inserire i dati presenti nella form in quel dato momento.
L'impostazione della form, l'inserimento dei dati e la conferma da parte dell'utente sono funzionalità impostate nel file AddWine. Il codice JavaScript di tale file definisce i vari campi di inserimento dati facendo sì che il nome del dato che viene inserito dall'utente sia specificato sopra, così da permettere all'owner il corretto inserimento. Una volta premuto il pulsante di inserimento verrà richiamata una funzione ad-hoc creata per verificare che tutti i campi siano stati compilati, solo allora tali dati potranno essere usati per l'inserimento. È stato inoltre inserito l'attributo required di JavaScript nel form in tutti i campi di inserimento per far si che l'utente, nel momento in cui non inserisce un dato, possa vedere dove non lo ha inserito.
Una volta ottenuti tutti i dati verrà richiamata la funzione asincrona _addBottleToList che prenderà i valori presi dalla form, l'indirizzo dell'utente che sta effettuando l'operazione e richiamerà la funzione di inserimento definita nello smartcontract per inserire correttamente il dato nella blockchain. Avverrà quindi una transazione all'interno della blockchain tramite lo smartcontract che garantirà che i termini lì definiti vengano rispettati (e.g. se l'utente non dovesse essere il proprietario della blockchain allora la transazione non avverrebbe).
Di seguito parte del codice JavaScript, che utilizza la libreria React, per richiamare la definizione del form di inserimento e la funzione Solidity per effettuare tale operazione:

```
1  { <AddWine
2      //viene definito che la variabile owner e' l'attuale utente
       loggato al wallet
3      owner={this.state.selectedAddress}
4        //richiamo alla funzione per inserire i dati, sono necessari
       tutti i dati richiesti dalla struct prima definita
5        addBottleToList={((nomeVino,
6          nomeCantina,
7          indirizzoCantina,
8          owner,
9          ...
10         ) =>
11         this._addBottleToList(nomeVino,
12           nomeCantina,
13           indirizzoCantina,
14           owner,
15           ...
```

```
16                 )
17       ) }
18         />
19       }
```

Come è possibile vedere nel codice soprastante per creare questa funzionalità viene definita la variabile owner come l'indirizzo di chi sta utilizzando la Dapp. Successivamente viene richiamata la funzione asincrona che permette di richiamare la funzione di inserimento definita nello smartcontract passando come valori i dati inseriti dall'utente, specificati uno a uno con apposita nomenclatura per un inserimento corretto e senza equivoci.
Di seguito il richiamo alla funzione Solidity per l'inserimento della bottiglia, richiamata nel codice sopra stante:

```
1  //richiamo asincrono della funzione di inserimento in attesa di
       essere chiamato
2       async _addBottleToList(nomeVino,
3     nomeCantina,
4     indirizzoCantina,
5     selectedAddress,
6     ...
7     ) {
8
9     try {
10      this._dismissTransactionError();
11    const tx = await this._bottle.addBottle(nomeVino,
12      nomeCantina,
13      indirizzoCantina,
14      selectedAddress,
15      ...
16      );
17      this.setState({ txBeingSent: tx.hash });
18    const receipt = await tx.wait();
19
20      if (receipt.status === 0) {
21            throw new Error("Transaction failed");
22      }
23      console.log(receipt);
24    } catch (error) {
25      if (error.code === ERROR_CODE_TX_REJECTED_BY_USER) {
26        return;
27      }
28      console.error(error);
29      this.setState({ transactionError: error });
```

```
30      } finally {
31        this.setState({ txBeingSent: undefined });
32      }
33    }
```

La funzione di richiamo è asincrona in quanto non è possibile eseguirla in contemporanea rispetto alle altre operazioni. Sono ovviamente specificate le variabili nell'ordine in cui verranno passate nonché con lo stesso ordine in cui sono state definite nella funzione presente nello smartcontract.

Visualizzazione

Per visualizzare i dati è stato scelto di dare la possibilità all'utente di scegliere che tipologia di dati vuole usare per il confronto. Ciò gli consentirà di effettuare più facilemente la comparazione. L'utente potrà quindi inserire il nome dei vini che vuole comparare e selezionare che tipo di confronto vuole fare. Il confronto potrà avvenire per tutti i dati presenti o solo alcuni di essi. Saranno sempre visualizzati i dati base quali Nome del vino, Nome della cantina, Uve utilizzate, Indirizzo della cantina, Indicazione geografica, Particella di coltivazione e Lotto di produzione. I dati sono stati messi in maniera tale da poter essere comparati immediatamente quindi adiacenti tra i due campioni.
Per il reperimento dati è stata creata una sezione, tramite il codice presente nel file GetWine, per l'inserimento del nome dei vini che si vogliono comparare e per la selezione dei dati che si vogliono confrontare. La prima parte del form permette di selezionare, tramite checkbox, di quali dati si vogliono comparare tra quelli presenti in blockchain. Tramite text è possibile identificare quale text fa riferimento a quale dato verrà poi visualizzato.
Di seguito parte del codice JavaScript che definisce le checkbox e le etichette a esse associate:

```
1  //Check box con relative etichette per definire quali funzioni di
       get richiamare
2      <input type="checkbox" name="Vendemmia" id="Vendemmia" value="
       Vendemmia" className="form-group"></input>
3      <label for="Vendemmia" > Inizio Vendemmia</label></p>
4      <p><input type="checkbox" name="RicezioneUva"  id="
       RicezioneUva" value="RicezioneUva"></input>
5      <label for="RicezioneUva"> Ricezione Uva</label></p>
6      <p><input type="checkbox" name="Diraspatura" id="Diraspatura"
        value="Diraspatura"></input>
7      <label for="Diraspatura"> Diraspatura</label></p>
8      //qui la lista continua allo stesso modo come sopra
```

Come si può notare etichette e variabili che indicano i flag delle check box sono state definite utilizzando nomi che facciano riferimento alla funzione che si andrà poi a richiamare. La seconda parte del form permette, in spazi segnalati da una descrizione sopra di essi, di inserire i nomi dei vini che si vogliono inserire. Una volta inseriti entrambi, e solo quando sono stati entrambi inseriti, e si avrà fatto click sul bottone verranno richiamate le funzioni di reperimento dati. Tutto questo avverrà solamente con le funzioni la cui checkbox associata è stata spuntata, altrimenti non verranno richiamate. Di seguito il codice che permette di richiamare le corrette funzioni di get definite sullo smartcontract e che permette di effettuare la comparazione tramite l'inserimento dei nomi delle bottiglie di cui si vogliono visualizzare le informazioni:

```
1    return ( <div>
2     <h4>Informazioni bottiglie da confrontare</h4>
3     <label>Selezionare le informazioni aggiuntive che si vogliono visualizzare</
      label>
4     <p></p>
5     //form di visualizzazione dati
6        <form
7    onSubmit={(event) => {
8      event.preventDefault();
9      const formData = new FormData(event.target);
10     //Visualizzazione nome bottiglie comparate
11     const wine_name1 = formData.get("wine_name1");
12     const wine_name2 = formData.get("wine_name2");
13     //richiami alle funzioni di get dei dati, dopo aver premuto il
      pulsante, a seconda di quali checkbox sono state spuntate
14     if (wine_name1 && wine_name2) {
15       get_bottleBasics(wine_name1, wine_name2);
16      if(document.getElementById("Vendemmia").checked){
17        get_Vendemmia(wine_name1, wine_name2);
18       }
19   }
20   }
21
22   >
23   <div>
24     <p>
25   <input type="checkbox" name="Vendemmia" id="Vendemmia" value="
      Vendemmia" className="form-group"></input>
26    <label for="Vendemmia" > Inizio Vendemmia</label></p>
27    //FOrm di inserimento nomi bottiglie che si vogliono comparare
```

```
28      <div className="form-group">
29        <label>Inserire il nome della prima bottiglia che si vuole comparare </
        label>
30      </div>
31      <div className="form-group">
32        <input className="form-control" type="text" name="
        wine_name1" required />
33      </div>
34      <div className="form-group">
35        <label>Inserire il nome della seconda bottiglia che si vuole comparare</
        label>
36      </div>
37      <div className="form-group">
38        <input className="form-control" type="text" name="
        wine_name2" required />
39      </div>
40      //Pulsante di conferma comparazione
41      <div className="form-group">
42        <input className="btn␣btn-primary" type="submit" value="
        Search" />
43      </div>
44      </div>
45      </form>
46      </div>
47    );
```

Per la visualizzazione dei dati viene nuovamente utilizzata la parola chiave required per far si che non si possano far paragoni senza l'inserimento di nessuna bottiglia o di una sola. Viene inoltre, tramite definizione dell'attributo name, definita quale informazione definita dall'utente corrisponde al primo o al secondo vino da comparare.
Verranno quindi richiamate le funzioni presenti nello smartcontract inviando anche il nome del vino per poter reperire i dati. Conseguentemente la blockchain, una volta reperiti i dati, li invia nuovamente all'utente che li visualizzerà nell'ordine e nel formato impostato precedentemente nello smartcontract.
Di seguito parte del codice JavaScript, che utilizza la libreria React, che richiama le funzioni di get definite nello smartcontract e permette la corretta visualizzazione dei dati reperiti:

```
1             {<GetWine
2             //impostazione valore variabile utente come l'indirizzo
    dell'utente che sta utilizzando l'applicazione
3           utente={this.state.selectedAddress}
```

```
4           //Richiami alle funzioni di get per poter visualizzare i
    dati
5           get_bottleBasics={(wine_name1,wine_name2) =>
6             this._get_bottleBasics(wine_name1,wine_name2)}
7           get_Vendemmia={(wine_name1,wine_name2) =>
8             this._getVendemmia(wine_name1,wine_name2)}
9           />}
10
11          //Qui ci sono gli altri get come sopra
12
13            <div>
14         //Visualizzazione dati
15        <p>
16        Informazioni base della prima bottiglia:</p>
17         <p></p> {this.state.bottleBasics_details_1}
18         <p></p>
19        <p>
20          Informazioni base della seconda bottiglia:  </p>
21        <p></p> {this.state.bottleBasics_details_2}<p></p>
22
23
24        <p>Informazioni sulla vendemmia della prima bottiglia: Inizio – Fine –
     Luogo</p> <p></p> {this.state.Vendemmia_details_1}<p></p>
25        <p>Informazioni sulla vendemmia della seconda bottiglia: Inizio – Fine
    – Luogo</p>  <p></p> {this.state.Vendemmia_details_2}<p></p>
26        <p></p>
```

Come in precedenza viene definito l'utente come l'indirizzo di chi sta effettuando l'operazione. Vengono inoltre richiamate le funzioni asincrone che comunicheranno con lo smartcontract per reperire i dati, prevedendo quindi che vengano inviati i nomi dei vini che vogliono essere paragonati. Il risultato ottenuto viene richiamato tramite this.state, specificando il nome della variabile che contiene il risultato che è stato cercato.
Di seguito la funzione in attesa di essere richiamata che permette di utilizzare le funzioni Solidity get definite in precedenza:

```
1 //Funzioni in attesa di essere richiamate per poter vedere i dati
2     async _get_bottleBasics(wine_name1,wine_name2) {
3    try {
4     const bottleBasics_details_1 = await this._bottle.get_bottleBasics(
      wine_name1)
5     const bottleBasics_details_2 = await this._bottle.get_bottleBasics(
      wine_name2);
6
```

```
7       this.setState({ bottleBasics_details_1 });
8       this.setState({ bottleBasics_details_2 });
9
10
11      } catch (error) {
12        if (error.code === ERROR_CODE_TX_REJECTED_BY_USER) {
13          return;
14        }
15        console.error(error);
16        this.setState({ transactionError: error });
17      } finally {
18        this.setState({ txBeingSent: undefined });
19      }
20    }
```

Sono ovviamente specificate le variabili nell'ordine in cui verranno passate per cui con lo stesso ordine in cui sono state definite nella funzione definita nello smartcontract. I risultati vengono poi inseriti nelle apposite variabili per poterle successivamente visualizzare nell'ordine corretto.

Chapter 4

Sperimentazione

4.1 Introduzione

Una volta completata l'analisi e l'implementazione derivante da essa, si passa alla sperimentazione ovvero a quella fase dove viene effettivamente utilizzata la Dapp e si sperimentano le varie funzionalità verificandone il risultato prodotto. Per effettuare questo passaggio correttamente bisogna tener conto sia della parte di back-end, rappresentata dallo smartcontract implementato nella blockchain, sia della parte di front-end rappresentata dal codice JavaScript che permette all'utente di poter usufruire delle tecnologie implementate in back-end senza consocerne le specifiche.
Per utilizzare la Dapp e sperimentare con essa e poterne poi analizzare i risultati bisogna effettuare tutti i passaggi necessari per lanciare la blockchain con lo smartcontract e attivare l'interfaccia grafica effettuando il collegamento alla blockchain. Solo dopo aver effettuato questi passaggi sarà possibile inserire e confrontare i dati sulla blockchain e vedere se l'obiettivo di questa tesi è stato raggiunto.
I passaggi sopra descritti presuppongono che siano già stati fatti i test preliminari per il funzionamento singolo dei componenti. Nello specifico è possibile testare il funzionamento dello smartcontract senza dover attivare e accedere all'applicativo. Hardhat da infatti la possibilità, tramite file di test appositamente creato, di tesare le funzioni create in Solidity. Dopo la compilazione andata a buon fine dello smartcontract tale file è stato eseguito e l'esito del test è andato a buon fine, garantendo che le funzioni create si comportino effettivamente come previsto. Il file, infatti, è stato creato in modo tale da inserire una bottiglia, con dati di test, all'interno della struttura e di reperirne poi i dati. Il risultato del test risulta positivo se tutte le funzioni sono state eseguite senza errore e se il risultato di esse corrisponde ai valori attesi, i

quali sono specificati nel file. In questo caso tutte le funzioni get create hanno prodotto un risultato uguale a quello atteso prendendo correttamente i dati inseriti nella precedente fase di inserimento. Questo test positivo ha posto le basi per la sperimentazione della Dapp nella sua interezza rispetto agli obiettivi posti inizialmente in questa tesi.

4.2 Attivazione e Accesso

4.2.1 Attivazione

Come primo passaggio per l'utilizzo della Dapp è necessario far girare, quindi attivare, l'istanza del network di Hardhat che possa essere collegata al proprio wallet. Questa fase avviene tramite l'utilizzo di Node.js e di Hardhat che permettono di lanciare localmente il network all'indirizzo specificato nel codice, in questo caso http://127.0.0.1:8545. Tramite il comando npx hardhat node si attiverà il network dove sarà presente la blockchain e gli account che possono operare su di essa, come nell'esempio sotto:

```
Account #0:
0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
(10000 ETH)
Private Key:
0xac0974bec39a17e36ba4a6b4d238ff944bacb478cbed5efcae784d7bf4f2ff80

Account #1:
0x70997970C51812dc3A010C7d01b50e0d17dc79C8
(10000 ETH)
Private Key:
0x59c6995e998f97a5a0044966f0945389dc9e86dae88c7a8412f4603b6b78690d

eth_blockNumber
eth_getBlockByNumber
net_version (2)
```

Successivamente si potrà implementare su di essa il contratto tramite il comando npx hardhat run scripts/deploy.js –network localhost. Di seguito il risultato dell'operazione sul terminal:

```
Deploying the contracts with the account:
0xf39Fd6e51aad88F6F4ce6aB8827279cffFb92266
Account balance:
```

```
100000000000000000000
Bottle address:
0x5FbDB2315678afecb367f032d93F642f64180aa3.
```

È possibile notare che il contratto è stato implementato all'indirizzo 0x5fbdb2315678afecb367f032d93f642f64180aa3, specificato nel codice, tramite l'account 0, il cui indirizzo è 0xf39fd6e51aad88f6f4ce6ab8827279cfffb92266. Pertanto tale account sarà l'owner del contratto nonché l'unico a poter inserire dati.
Di seguito la linea di codice che specifica dove implementare il contratto Bottle:

```
    {
  "Bottle": "0x5FbDB2315678afecb367f032d93F642f64180aa3
   "
}
```

Una volta attivato il network dove è presente la blockchain e implementato lo smartcontract è necessario avviare la parte di front-end dell'app tramite comando npm run. In questo modo sarà possibile utilizzare l'app all'indirizzo http://localhost:3000.

4.2.2 Accesso

Una volta eseguiti tutti i passaggi precedenti sarà possibile andare all'indirizzo locale della Dapp per accedere alle varie funzionalità. Nella prima schermata sarà richiesto l'accesso alla blockchain tramite identificativo wallet ovvero sarà necessario accedere, tramite Metamask, con il proprio indirizzo alla blockchain. Effettuato correttamente l'accesso al proprio portafoglio virtuale si verrà indirizzati alla pagina principale, dove è possibile eseguire le operazioni di inserimento e comparazione. Questa schermata è visibile in figura 4.1. Per poter effettuare correttamente questa operazione è necessario aver creato nel proprio wallet un account con uno degli indirizzi presenti nella fase di attivazione della blockchain. Così facendo quando si effettua l'accesso si potrà legare quell'account, riconosciuto dalla blockchain, all'indirizzo web.
Gli indirizzi web presenti nella fase di attivazione rappresentano gli attori che possono interagire con la blockchain in quanto riconosciuti da essa, pertanto se si vuole testare come diversi utenti riconosciuti, in quanto solo chi è riconosciuto può interagire con la blockchain, possano influire sul risultato finale è necessario creare più account con tali indirizzi.

4.3 Inserimento

Per inserire i dati dalla pagina principale è necessario cliccare sull'apposito pulsante per poter vedere il form di inserimento appositamente creato. Qui, specificati con apposita etichetta sopra stante, è possibile riempire i campi per l'inserimento dei dati in blockchain. Non è possibile confermare l'inserimento con bottone senza aver prima compilato tutti i campi: in caso contrario l'applicazione segnalerà il campo non compilato. Un esempio è presente in figura 4.2. L'unico dato che non è possibile compilare manualmente è l'indirizzo dell'owner in quanto si riferisce all'indirizzo del proprietario del contratto. Questo dato viene preso automaticamente durante la transazione prendendo l'indirizzo di chi sta eseguendo l'operazione, questo consentirà al sistema di verificare che solo l'owner della blockchain possa effettuare l'inserimento, come specificato nello smartcontract. Al click del pulsante apparirà una finestra che chiederà la conferma della transazione che, come da figura 4.3, ha un costo in GO. Non è quindi possibile effettuare tale operazione se non si è l'owner del contratto ovvero se non si è l'utente 0xf39fd6e51aad88f6f4ce6ab8827279cfffb92266. Si deve quindi accedere al wallet tramite l'account creato collegato a quell'indirizzo, come da figura.
Sono stati effettuati diversi inserimenti per verificare non solo il funzionamento ma anche il costo delle transazioni. Di seguito la tabella con i valori registrati:

Transazione	**Costo in GO**
Inserimento 1	0.00273413
Inserimento 2	0.00039509
Inserimento 3	0.00181409
Inserimento 4	0.00173438
Inserimento 5	0.00159212

Ognuno di questi inserimenti ha prodotto un nuovo blocco all'interno della blockchain, visualizzabile come in figura 4.4. Metamask rende inoltre disponibile un elenco di tutte le transazioni effettuate correttamente con i relativi dettagli per un'ulteriore verifica delle operazioni andate a buon fine, come in figura 4.5 e 4.6. La conferma dell'avvenuta transazione tramite Metamask, con i dettagli annessi e la visualizzazione della creazione di un nuovo blocco, tramite terminal, sono la prova che le operazioni di inserimento sono state correttamente effettuate. Per la verifica dei dati inseriti, ovvero che tutti i campi inseriti siano stati inseriti come lo si desiderava e dove, bisogna utilizzare le funzioni di reperimento dati.

4.4 Comparazione

Per comparare due vini è necessario inserirne il nome negli appositi spazi indicati nel fondo della pagina. È possibile nella sezione soprastante, tramite checkbox, scegliere il tipo di comparazione fare. Le checkbox possono essere tutte flaggate o meno. Nel secondo caso la comparazione avverrà solo per i dati base quali Nome del vino, Nome della cantina, Indirizzo della cantina, Indicazione geografica, Particella di coltivazione, Uve utilizzate e Lotto di produzione.
I dati da comparare, tra il primo e il secondo vino, sono in colonna per facilitarne la comparazione. Nelle figure 4.7, 4.8, 4.9, 4.10 alcuni esempi di comparazione effettuati. L'operazione di confronto non richiede nessun tipo di conferma da parte di Metamask, richiede però che chi effettua l'operazione sia un attore riconosciuto, quindi un indirizzo tra quelli riconosciuti dalla blockchain.
Facendo un paragone tra i dati che sono stati inseriti nella fase precedente ed i dati che vengono visualizzati durante il paragone si può facilmente constatare come questi corrispondano. Inoltre, la visualizzazione così impostata rende chiaro all'utente quale dato sta visualizzando e quali dati sta paragonando.
Questo tipo di visualizzazione permette anche di poter verificare con quanta precisione sono stati registrati e poi inseriti i dati. Le informazioni visualizzate corrispondono infatti esattamente a quelle inserite dall'utente, senza alcun tipo di controllo sul formato. Una comparazione tra date può quindi avvenire anche se il formato delle date è diverso, come per le indicazioni geografiche che possono corrispondere a indirizzi o a coordinate. Se un produttore registra le proprie date con un formato timestamp e le indicazioni geografiche con le coordinate mentre un altro fa tali registrazioni prendendo le indicazioni geografiche come indirizzi e le date in formato diverso (e.g. Sabato 9 Settembre 2023) sarà comunque possibile effettuare la comparazione e si potrà aggiungere al proprio metodo di giudizio la precisione con cui è stata registrata una data informazione.

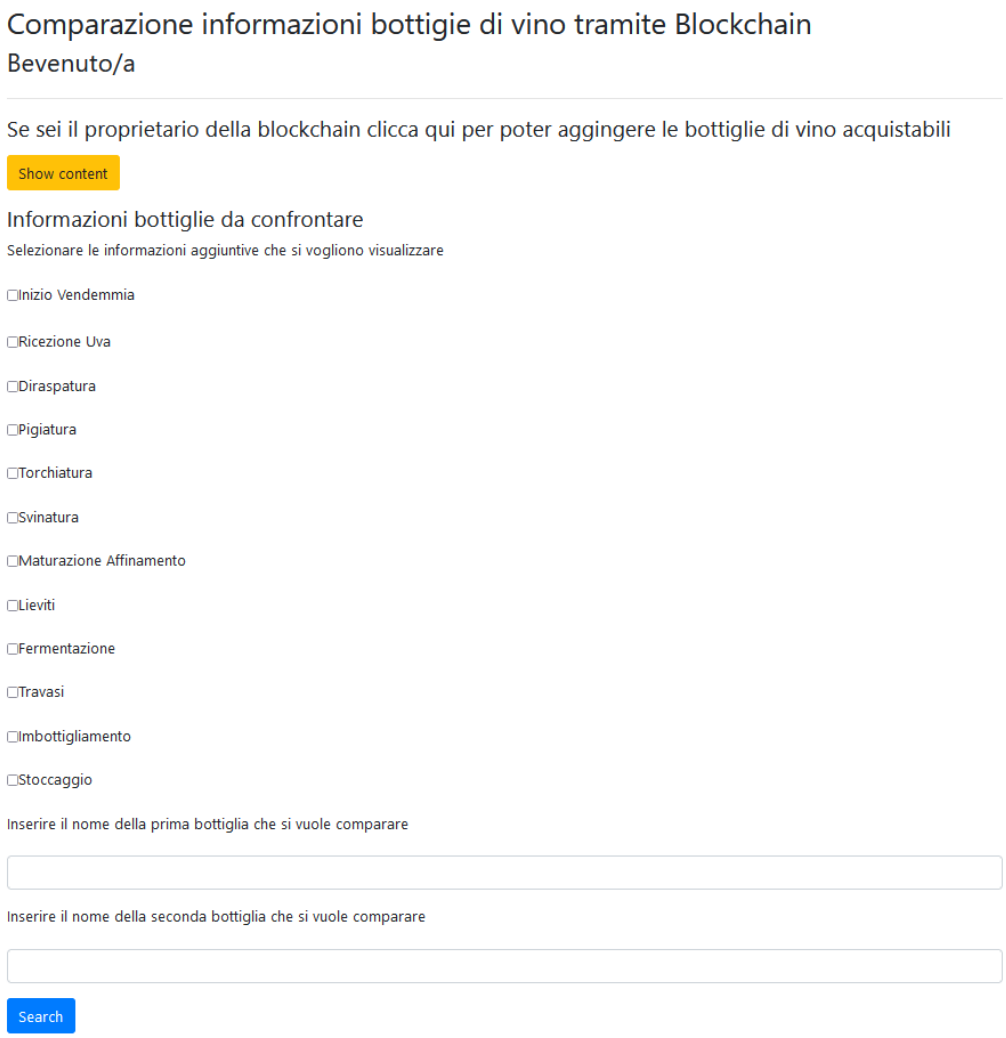

Figure 4.1: Prima Pagina

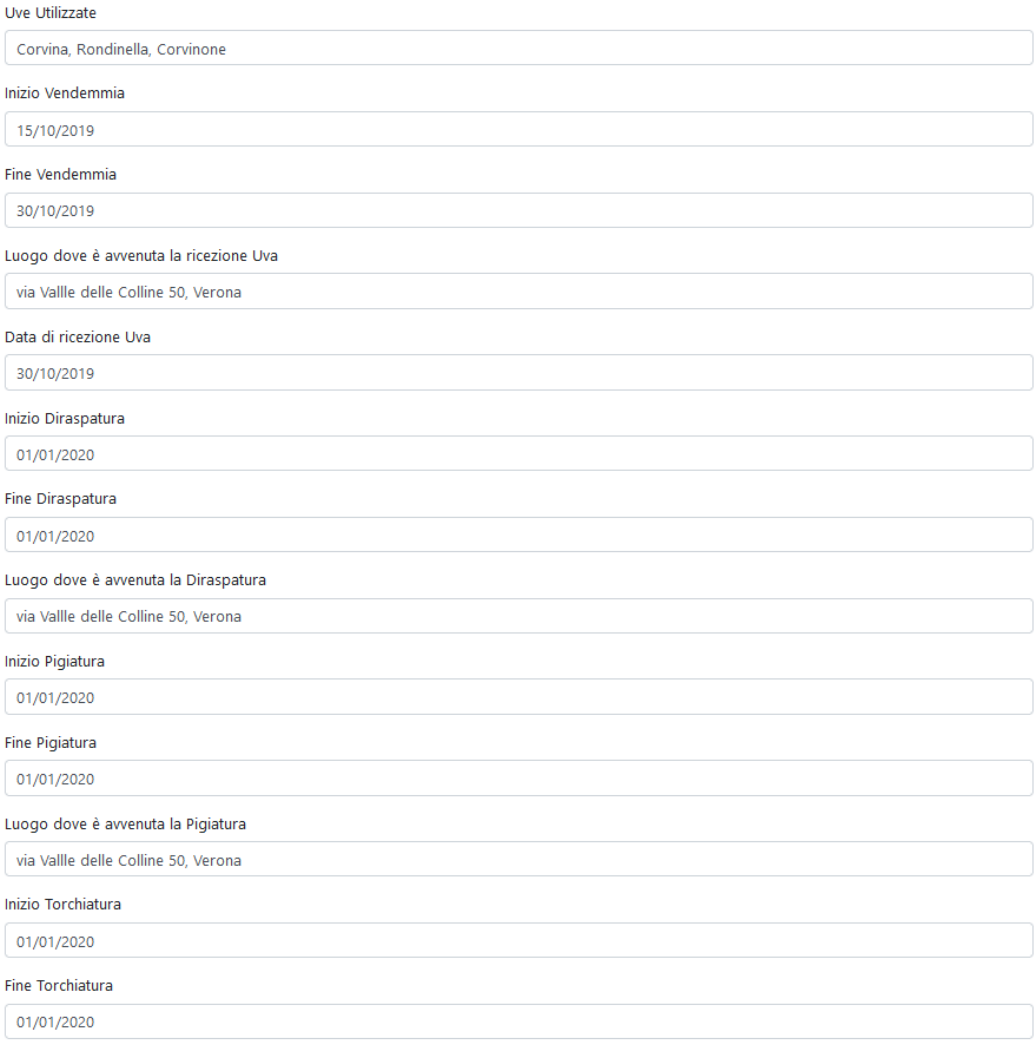

Figure 4.2: Esemipo di inserimento dati

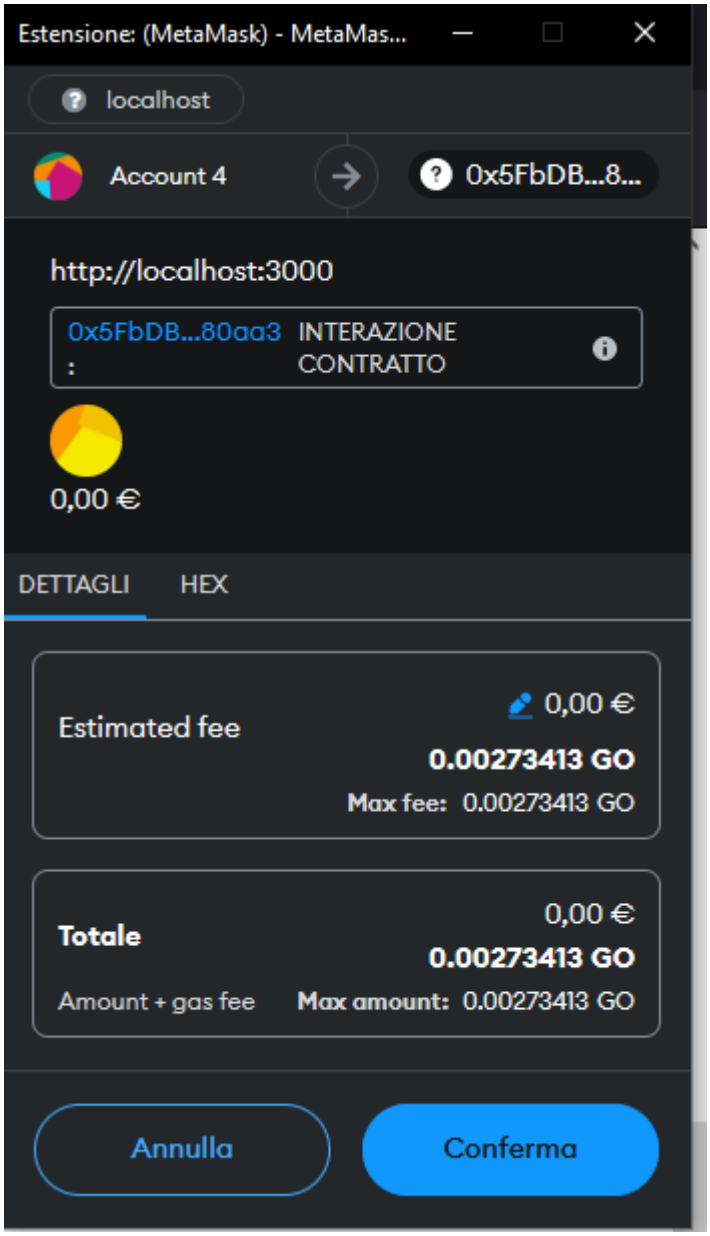

Figure 4.3: Risultato inserimento

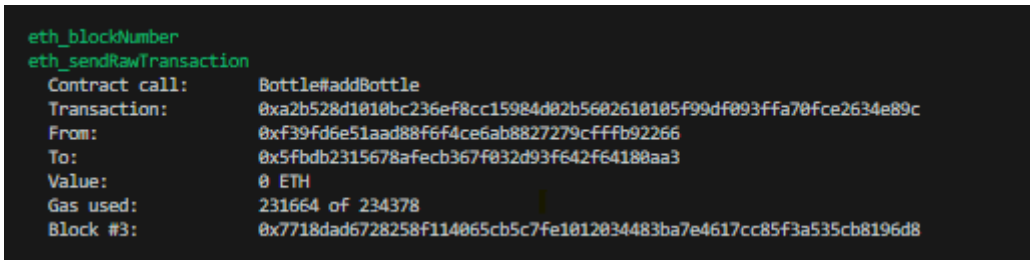

Figure 4.4: Blocco

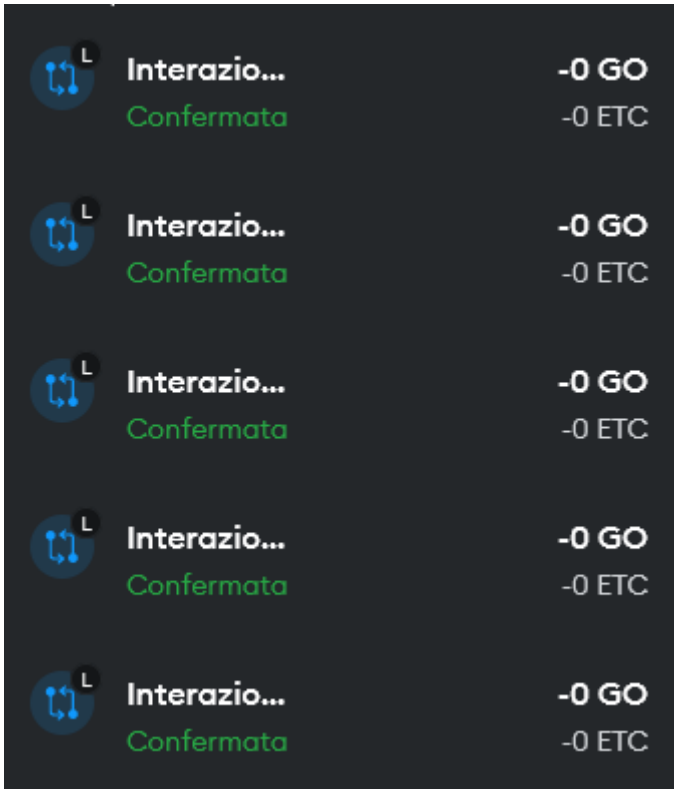

Figure 4.5: Elenco transazioni effettuate correttamente

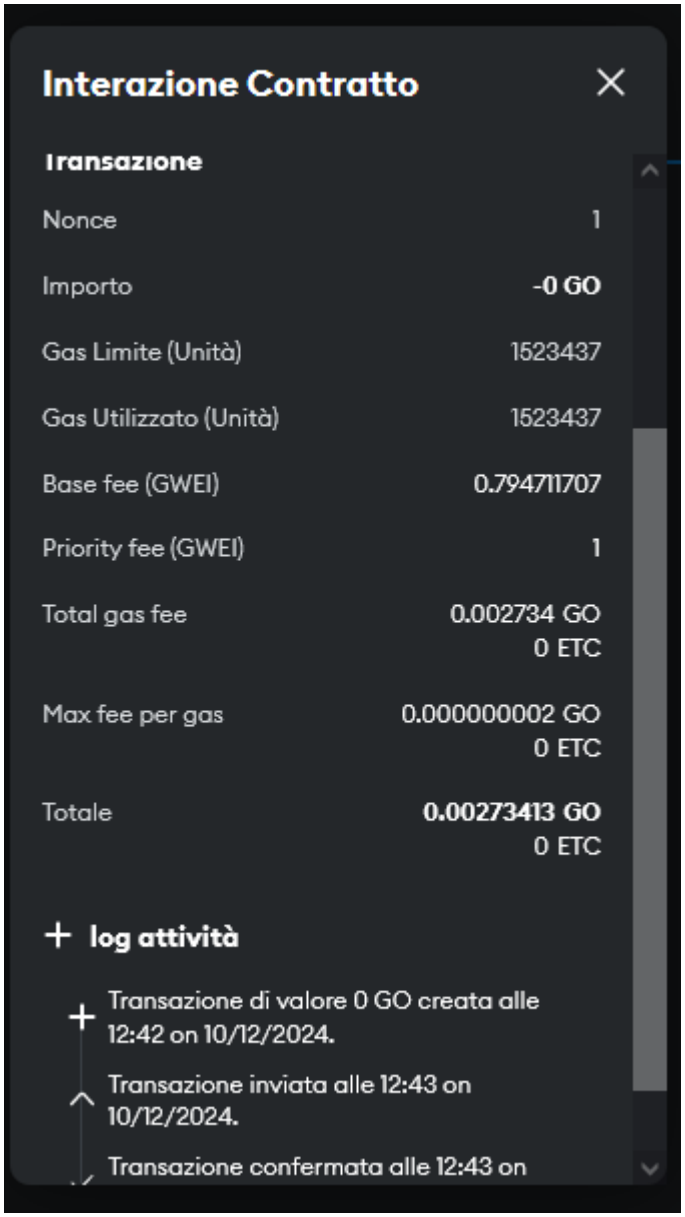

Figure 4.6: Dettaglio transazione effettuata correttamente

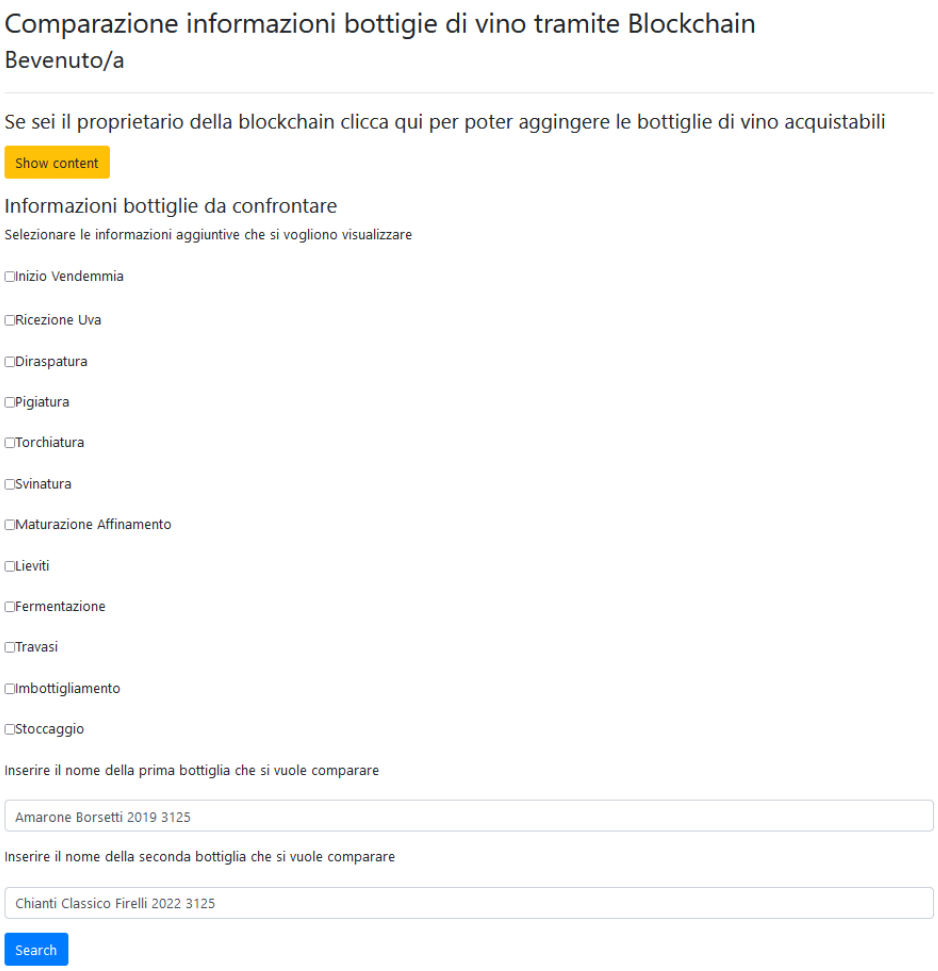

Figure 4.7: Esempio di confronto 1

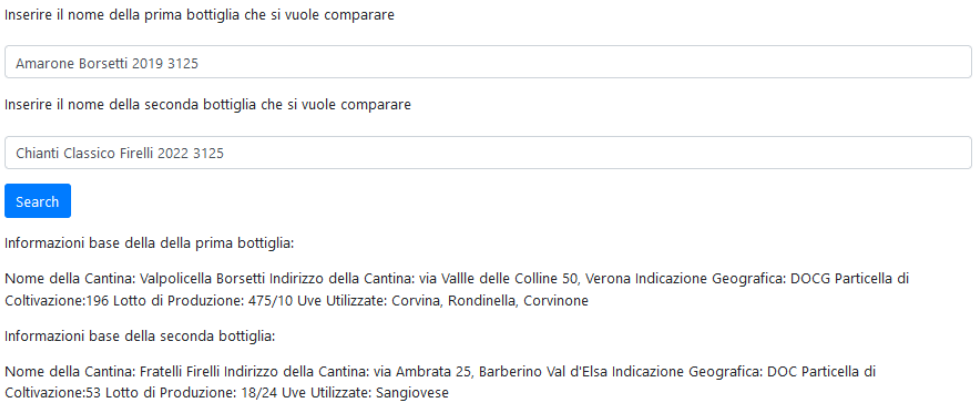

Figure 4.8: Esempio di confronto 2

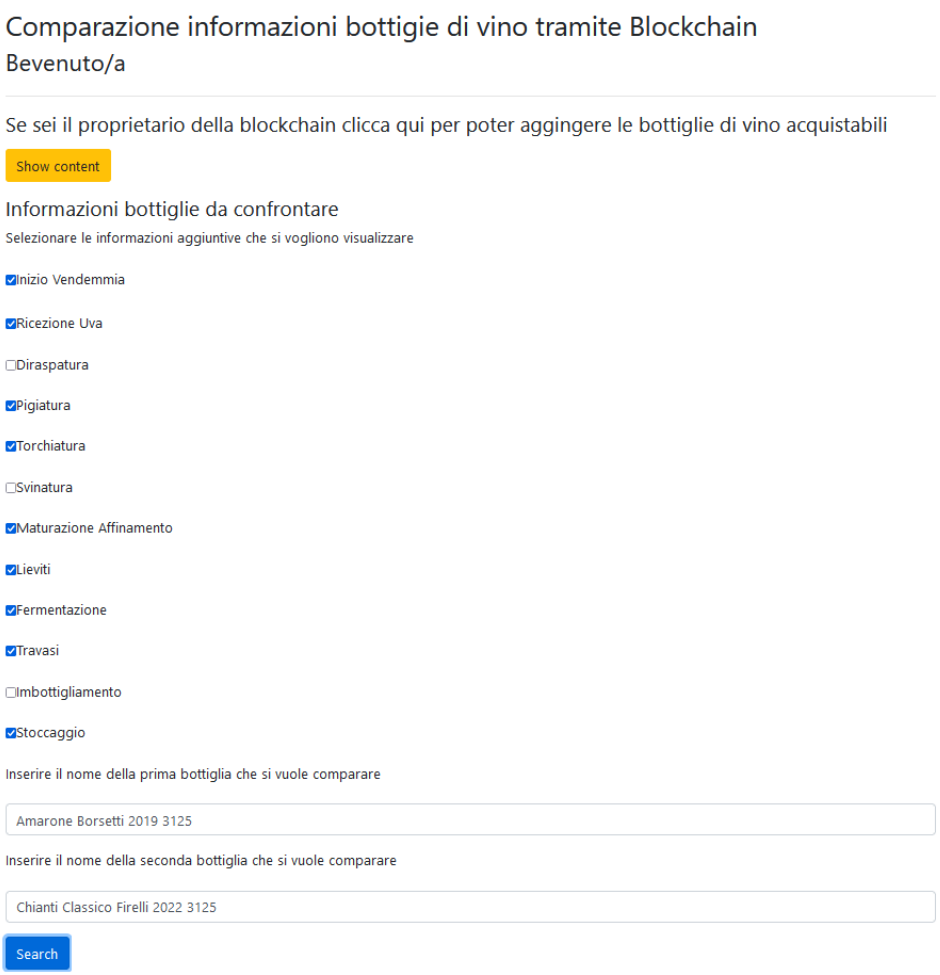

Figure 4.9: Esempio di confronto 3

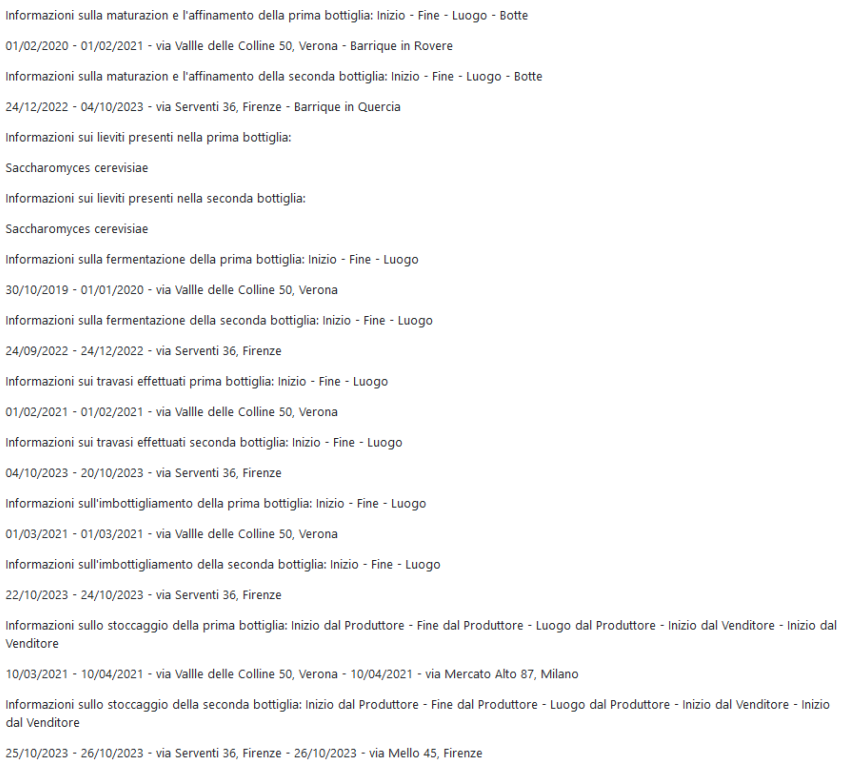

Figure 4.10: Esempio di confronto 4

Chapter 5

Conclusioni

5.1 Introduzione

Una volta concluso il progetto è possibile verificare se gli obiettivi posti inizialmente in questa tesi sono stati raggiunti. Questo passaggio può avvenire grazie al confronto tra ciò che è stato possibile sviluppare all'interno del progetto e paragonare i risultati ottenuti rispetto alle aspettative iniziali.

5.2 Obiettivi

L'obiettivo di questa tesi è quello di sviluppare una Dapp per poter tracciare e poi confrontare in un'unica soluzione i dati di più bottiglie di vino. Nello specifico ci si proponeva di creare una blockchain che implementasse uno smartcontract, utilizzato per preservare i dati, e creare un applicativo che si possa connettere a tale struttura per paragonare tra di loro le informazioni memorizzate.
L'obiettivo di questa tesi può essere misurato verificando che la blockchain implementi lo smartcontract, il quale deve poter comunicare con l'applicazione in maniera corretta. Lo smartcontract deve essere in grado di gestire tutti i dati necessari per una corretta comparazione e l'applicazione non solo deve interagire con esso senza errori ma deve anche poter visualizzare le informazioni per poterle comparare.

5.3 Risultati

5.3.1 Blockchain e Smartcontract

Per la verifica del corretto funzionamento della blockchain sviluppata e dello smartcontract implementato sono stati effettuati vari test, sia a livello frontend che a livello backend. Tra questi test abbiamo avuto l'inserimento di dati, nella forma attesa e il reperimento di essi. Successivamente si è tentato di effettuare inserimenti in maniera errata o con utente non valido. Nel caso di inserimento dato diverso da quello atteso o fatto tramite utente non owner non è stato possibile arrivare al compimento dell'operazione.
Queste verifiche hanno dimostrato come la blockchain sia effettivamente stata implementata e di come questa contenesse lo smartcontract sviluppato. Inoltre, è stato verificato che i meccanismi implementati per il funzionamento dell'applicativo, rispetto alle attese, risultano essere efficaci nel garantire un corretto funzionamento del sistema.

5.3.2 Transazioni

I test eseguiti sulla Dapp sviluppata mostrano come il costo delle transazioni effettuate, ovvero le operazioni effettuate dall'utente tramite wallet su blockchain, sia irrisorio. Solo l'inserimento dei dati prevede un costo mentre la comparazione non ne richiede alcuno. Si sono ovviamente considerate solo le operazioni portate a buon fine, non le operazioni errate che non sono quindi salite sulla blockchain.

5.3.3 Comparazione

La comparazione tra più bottiglie di vino, come sopra descritta, può avvenire nonostante le bottiglie di vino di riferiscano a tipologie di vini totalmente diversi e con provenienze totalmente diverse a patto che i dati siano compilati correttamente. Una compilazione errata da parte dell'owner della blockchain rende errato il tracciamento della bottiglia e quindi rende la comparazione impossibile da effettuare. Si parla di compilazione errata quando si parla di inserimento dati nel formato corretto ma con l'informazione sbagliata, pertanto non è possibile in alcun modo per le tecnologie implementate ovviare a questo problema.
Il tipo di comparazione effettuata presuppone la previa conoscenza o la non necessità di conoscere parametri che caratterizzano la bottiglia di vino e la tipologia di vino in se, che esulano dal percorso di filiera. Se fosse necessario

anche questo tipo di conoscenza per poter fare il paragone questa applicazione risulterebbe inadeguata poichè non fornisce tale tipo di informazioni.

5.4 Upgrade possibili

La selezione dei parametri in se risulta essere completa se si guarda alle fasi della produzione di vini, non se si guarda alle caratteristiche dei vini. Una possibile espansione della Dapp può riguardare l'aggiunta di parametri che riguardino le caratteristiche del vino come la gradazione alcolica, il gusto e gli abbinamenti che è possibile fare con esso. Si potrebbe, tramite IPFS, ovvero tramite protocollo di comunicazione e una rete peer-to-peer per l'archiviazione e la condivisione di dati in un file system distribuito, aggiungere l'etichetta come dato. Inoltre, per poter migliorare l'affidabilità, è possibile implementrare la Proof of Location, una "prova" di geolocalizzazione decentralizzata che può essere utilizzata per convalidare la localizzazione geografica di un dispositivo, per tutte le informazioni che riguardano il luogo dove è avvenuta una particolare fase.

Chapter 6

Ringraziamenti

Desidero in primo luogo ringraziare il Prof. Ferretti per avermi guidato e supportato nella fase finale del mio percorso accademico.
Un ringraziamento speciale ai miei genitori e a mia sorella, che con il loro sostegno mi hanno aiutata ad arrivare fin qui.
Grazie ai miei amici, a quelli nerd con cui giocare, che sia D&D o un videogioco poco importa, a quelli bolognesi che mi hanno fatto scoprire nuovi luoghi in questa meravigliosa città che mi ha accolta, a quelli con cui semplicemente è bello uscire per far quattro chiacchiere, perché con tanta pazienza e un pizzico di pazzia, mi hanno sopportata e accompagnata in questo viaggio.
Ai miei colleghi: finalmente posso passare al full-time e potrete godere della mia meravigliosa presenza (in smartworking) dal lunedì al venerdì per otto ore al giorno.
Grazie al Karate e ovviamente al mio maestro e ai miei compagni di allenamento, perché senza quello sfogo settimanale la mia vita avrebbe avuto un altro sapore (e un po' meno lividi).
Grazie a chi non c'è più e grazie a tutti quelli che, in un modo o nell'altro, mi hanno aiutata in questo percorso.

Bibliography

[1] Nino Adamashvili, Radu State, Caterina Tricase, and Mariantonietta Fiore. Blockchain-based wine supply chain for the industry advancement. *Sustainability*, 13(23), 2021.

[2] Mike Brookbanks and Glenn Parry. The impact of a blockchain platform on trust in established relationships: a case study of wine supply chains. *Supply Chain Management: An International Journal*, 27(7):128–146, Jan 2022.

[3] Vitalik Buterin. On public and private blockchains, 2015.

[4] Bahar Farahani, Farshad Firouzi, and Markus Luecking. The convergence of iot and distributed ledger technologies (dlt): Opportunities, challenges, and solutions. *Journal of Network and Computer Applications*, 177:102936, 2021.

[5] Gloria Luzzani, Erica Grandis, Marco Frey, and Ettore Capri. Blockchain technology in wine chain for collecting and addressing sustainable performance: An exploratory study. *Sustainability*, 13(22), 2021.

[6] Bojana Malisic, Nemanja Misic, Srdjan Krco, Aleksandra Martinovic, Sandra Tinaj, and Tomo Popovic. Blockchain adoption in the wine supply chain: A systematic literature review. *Sustainability*, 15(19), 2023.

[7] Satoshi Nakamoto. Bitcoin: A peer-to-peer electronic cash system. May 2009.

[8] Simon Pearson, David May, Georgios Leontidis, Mark Swainson, Steve Brewer, Luc Bidaut, Jeremy G. Frey, Gerard Parr, Roger Maull, and Andrea Zisman. Are distributed ledger technologies the panacea for food traceability? *Global Food Security*, 20:145–149, 2019.

[9] Christian Straubert and Eric Sucky. How useful is a distributed ledger for tracking and tracing in supply chains? a systems thinking approach. *Logistics*, 5(4), 2021.

[10] Zibin Zheng, Shaoan Xie, Hongning Dai, Xiangping Chen, and Huaimin Wang. An overview of blockchain technology: Architecture, consensus, and future trends. In *2017 IEEE International Congress on Big Data (BigData Congress)*, pages 557–564, 2017.