POLITECNICO DI BARI

DIPARTIMENTO DI INGEGNERIA ELETTRICA E DELL'INFORMAZIONE

CORSO DI LAUREA MAGISTRALE IN INGEGNERIA INFORMATICA CORSO CYBERSECURITY AND CLOUD

Business Report

Project: AHEREUM

Software Architecture and Pattern Design

Business Report AHEREUM

Docente:
Prof.ssa Marina Mongiello
Ing. Marco Fiore

Codice Gruppo:
TISM

Componenti gruppo:
Detomaso Riccardo,
Iannone Alessandro,
Silvestri Giovanni,
Tangari Mauro

Anno Accademico 2021/2022

Business Report Ahereum

EXECUTIVE SUMMARY

Scopo & Obiettivi:

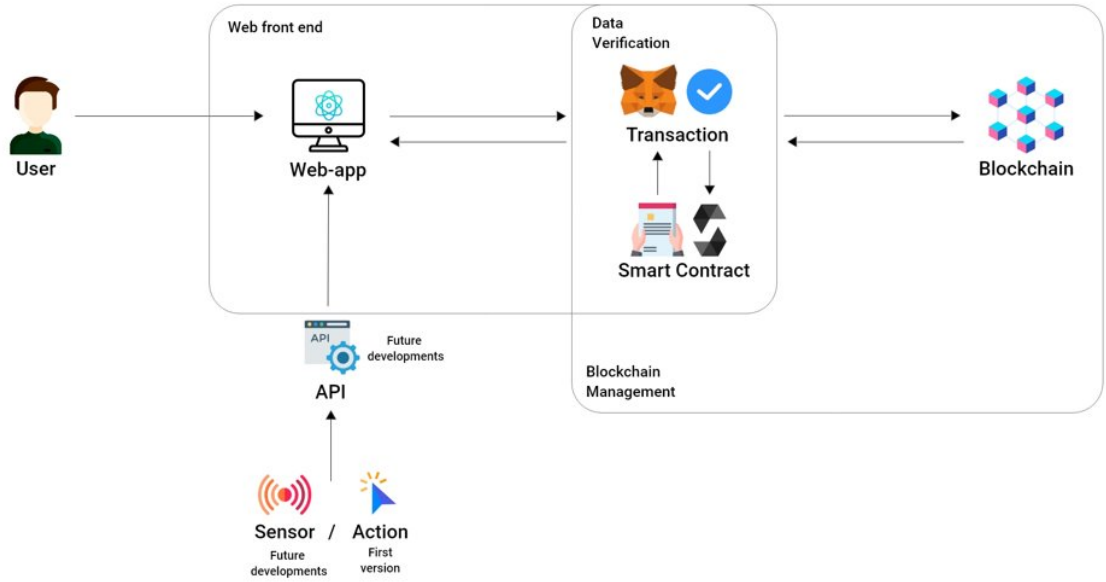
Il progetto “AHEREUM” mira a fornire una piattaforma, basata su una blockchain autorizzata, per salvare, in modo immutabile, il percorso di un drone o di uno stormo di droni. L'obiettivo finale sarebbe quello di integrare una simulazione, basata sul Digital Twin, per testare la corretta integrazione del drone con i dati salvati nella blockchain.
Partendo da un’idea della Prof. Mongiello e guidati dall’Ing Fiore, “AHEREUM” è stato ideato e realizzato da un gruppo di 4 studenti dell’università “Politecnico di Bari”, costituito in particolare da:

1) Iannone Alessandro: Full Stack Developer
2) Silvestri Giovanni: Frontend Developer
3) Detomaso Riccardo: Frontend Developer
4) Tangari Mauro: Frontend Developer.

La nostra Web App si divide in 3 obiettivi principali:

1) **Frontend:** permettere di inserire le coordinate del drone, attraverso una mappa o semplicemente scrivendole in una casella di testo.
2) **Web3.js**: raccogliere librerie JavaScript che permettono di comunicare con la blockchain per salvare i dati nella seconda.
3) **Blockchain**: con l’implementazione della Blockchain possiamo rendere immutabili ed assoluti i dati scritti, in maniera tale da ottenere il massimo livello di sicurezza nella gestione dei percorsi dei velivoli.

Architettura del Progetto



L'utente comunica con la nostra WebApp, inserendo le coordinate del percorso virtuale da Frontend. Per iniziare la transazione l'utente deve prima accedere con le proprie credenziali al proprio wallet Metamask. Subito dopo c'è la fase di Data verification nella quale i dati inseriti da Frontend vengono confrontati con la struttura dei dati definita nello Smart Contract.

A seguito di questa verifica i dati vengono salvati in un blocco che viene caricato in BlockChain.

Una volta salvati i dati in BlockChain, attraverso una funzione get, quando il drone raggiunge una centralina si riesce a prelevare le informazione legate alla centralina dello step successivo, situata in coordinate prestabilite.

Una implementazione effettuata in script Python, e che successivamente verrà riportata nello SmartContract, riguarda l'Encryption e Decryption dei dati salvati nel blocco, in modo da rendere non leggibili le coordinate del percorso a un utente esterno e quindi renderlo sicuro.

Per controllare le transazioni effettuate in BlockChain noi di “AHEREUM” ci serviamo di Etherscan che è un blockchain explorer per la rete Ethereum, il quale consente di cercare transazioni, blocchi, indirizzi di wallet, smart contract e altri dati.

Per quanto riguarda l’implementazione di cifratura e decifratura delle informazioni del blocco ci siamo serviti di Python, in particolar modo abbiamo sfruttato la classe Fernet della libreria Cryptography.

Per un maggior approfondimento sulle tecnologie e sulle piattaforme usate, è consigliabile leggere la sezione 1) PIATTAFORME E TECNOLOGIE di cui si parla in seguito.

Come capitale di partenza, nessun membro del team ha dovuto contribuire ad una spesa iniziale per l’avvio del progetto.

CONTENUTI

1) PIATTAFORME E TECNOLOGIE

Frontend

Per il Frontend della Web App abbiamo utilizzato React, una libreria JavaScript open source gestita da Meta per la creazione di interfacce utente basate su componenti.

È stato ideato per consentire, in maniera semplice, lo sviluppo di applicazioni con metodologia «cross-platform», quindi in grado di girare su PC, smartphone e tablet.

Nello specifico, per lo sviluppo della nostra web-app, abbiamo sfruttato la libreria «Material UI», che fornisce una serie di componenti in Material Design.

Perché React?

-è un linguaggio di programmazione JavaScript, semplice, intuitivo e di rapido apprendimento;

-la rappresentazione del componente si traduce in chiamate API a React che intervengono nel modo più rapido ed efficiente possibile sul DOM per creare gli elementi necessari;
-creazione di web app «responsive», ossia il cui design può adattarsi dinamicamente alle dimensioni del display su cui l'applicazione è visualizzata.

Backend

Web3: è un recente sviluppo del concetto di Internet. Una nuova idea di web totalmente decentralizzato, basato su protocolli inediti e, soprattutto, sulla tecnologia blockchain. Grazie a Web3.js abbiamo ottenuto: una implementazione specifica per l'interazione con blockchain e degli smart-contracts in maniera facile e di rapido apprendimento;

Ethereum: è una piattaforma decentralizzata per la creazione e pubblicazione peer-to-peer di contratti intelligenti, tra le più famose e soprattutto aperta a tutti e ovunque nel mondo.

Solidity: è il linguaggio principale utilizzato per la programmazione di Smart Contract su Ethereum. Esso è abbastanza semplice e consente di creare contratti facili da sviluppare e sicuri.

Python

Python è un linguaggio di programmazione orientato a oggetti. Le caratteristiche più immediatamente riconoscibili di Python sono le variabili non tipizzate e l'uso dell'indentazione per la sintassi delle specifiche, al posto delle più comuni parentesi.
Sebbene Python venga in genere considerato un linguaggio interpretato, in realtà il codice sorgente non viene convertito direttamente in linguaggio macchina. Infatti passa prima da una fase di pre-compilazione in bytecode, che viene quasi sempre riutilizzato dopo la prima

esecuzione del programma, evitando così di reinterpretare ogni volta il sorgente e migliorando le prestazioni.

Visual Studio Code

Visual Studio Code, è un editor di codice sorgente sviluppato da Microsoft che include il supporto per debugging, un controllo per Git integrato e può essere usato con vari linguaggi di programmazione tra cui Python, Solidity e framework come React.

Git Hub

GitHub è un servizio di hosting per progetti software, è una implementazione dello strumento di controllo versione distribuito Git, che permette di tenere traccia delle modifiche e delle versioni apportate al codice sorgente del software. L'abbiamo usato come sistema per la gestione dello sviluppo di software di tipo collaborativo.

Ethereum

Ethereum è una piattaforma decentralizzata del Web 3.0 per la creazione e pubblicazione peer-to-peer di contratti intelligenti (Smart Contracts) creati in un linguaggio di programmazione Turing-completo. La criptovaluta a esso legata, Ether, è seconda in capitalizzazione dietro ai Bitcoin.

A differenza di Bitcoin, Ethereum opera utilizzando conti e saldi secondo le cosiddette transizioni di stato, che non si basano su output di transazione non spesi (unspent transaction outputs, UTXOs), ma sui saldi correnti (chiamati stati) di tutti i conti, oltre ad alcuni dati aggiuntivi. L'informazione relativa allo stato non è memorizzata nella blockchain, bensì è archiviata in un proprio albero di Merkle, vale a dire un albero binario nel quale ogni nodo è padre di due figli e il suo hash è dato ricorsivamente dalla concatenazione degli hash dei due blocchi a esso associati, secondo il seguente schema:

$$hash_0 = hash(hash_{0-0} + hash_{0-1})$$[6]

Metamask

MetaMask è un wallet di criptovalute che consente agli utenti di memorizzare ETH e altri token ERC-20. Può essere utilizzato anche per interagire con applicazioni decentralizzate e la sua popolarità è dovuta alla semplicità d'uso.

❓ Cos'è:	Wallet criptovalute
👥 A chi si rivolge:	A chiunque
📜 Sicurezza:	Bassa
💶 Costi:	Gratuito
🏆 Miglior wallet alternativo	eToro

2) PROBLEMI E RISOLUZIONI

Cifratura e Decifratura

Un point of failure da noi riscontrato nel progetto è stato quello della Cifratura e Decifratura collegata alla BlockChain.

Nello specifico, avendo utilizzato un crittosistema simmetrico, la difficoltà è stata quella di assegnare e poi prelevare, oltre alle informazioni cifrate da decifrare, anche la chiave simmetrica del crittosistema. Infatti per ogni coordinata cifrata c'era una diversa chiave che, in fase di decifratura, doveva essere testata con ogni singola coordinata per trovare la corrispondente corretta.

L'implementazione in blockchain di questo meccanismo ha rappresentato un point of failure. Pertanto, abbiamo provveduto a creare una simulazione in Python per il nostro sistema.

OSSERVAZIONI CONCLUSIVE E FUTURI SVILUPPI

Nonostante il nostro gruppo abbia fatto il massimo si può sempre migliorare, e per alcuni sviluppi futuri legati a questo progetto ci potrebbero essere:

1) Integrazione di una mappa grazie alla quale l'utente può selezionare una cabina da quelle disponibili, visualizzandone la posizione geografica;

2) Integrazione di API grazie alla quale la Web App è in grado di interagire con un drone reale per lo scambio di informazioni;

3) Integrazione di una blockchain privata e autorizzata per garantire riservatezza dei dati