



Blockchain in PHP

IIS EINAUDI SCARPA

Lorenzo Mancini | 5 B Informatica | a. s. 2019/2020

Introduzione

Il mio progetto è la creazione di un sito web realizzato mediante il linguaggio PHP che ha lo scopo di connettere un utente con la tecnologia blockchain.

Questo sito permette agli utenti di accedere alla piattaforma tramite registrazione, al fine di poter effettuare delle transazioni nella cryptovaluta chiamata “Lor€nso”, offrendo le principali funzionalità della tecnologia.

Una blockchain è un registro digitale aperto e distribuito, in grado di memorizzare record di dati (denominati transazioni) in modo sicuro, verificabile e permanente. Una volta scritti, i dati in un blocco non possono essere retroattivamente alterati senza che vengano modificati tutti i blocchi successivi ad esso.

Ogni nodo o miner del sistema decentralizzato ha una copia della blockchain: la qualità dei dati è mantenuta grazie a una massiva replicazione del database. Non esiste nessuna copia ufficiale centralizzata e nessun utente è più credibile di altri, tutti sono allo stesso livello di credenziali. I miner sono gli utenti che validano le nuove transazioni e le aggiungono al blocco che stanno costruendo dopo aver verificato l'intera blockchain. Una volta completato il blocco, lo trasmettono agli altri nodi della rete.

Caratteristiche di una blockchain

- **Decentralizzazione:** l'intero registro degli eventi organizzato in blocchi è distribuito sui diversi nodi della rete, non esiste un unico repository contenente la totalità delle informazioni.
- **Incontestabilità:** la decisione riguardante la validità di un'informazione non viene presa unilateralmente ma attraverso un meccanismo di raccolta del consenso all'interno della rete, rendendo così particolarmente difficile metterne in discussione l'esito.
- **Sicurezza:** tutti gli eventi possono essere fatti risalire con certezza alle identità digitali che li hanno generati, attraverso l'utilizzo di meccanismi di crittografia asimmetrici.
- **Inalterabilità:** i dati che sono stati scritti all'interno della Blockchain non possono essere modificati, se non attraverso specifiche regole del protocollo che definiscono rigorosamente le modalità con cui si possono effettuare cambiamenti.
- **Tracciabilità:** a tutti gli eventi registrati vengono assegnati un identificativo e una marca temporale che li rende facilmente tracciabili e verificabili.
- **Programmabilità:** all'interno dei blocchi possono essere incluse istruzioni che facciano scatenare specifiche azioni al verificarsi certe condizioni.

Obiettivi del progetto

All'interno del sito un utente può svolgere diverse operazioni:

- Eseguire nuove transazioni;
- Monitorare le transazioni che lo coinvolgono;

- Validare i blocchi da inserire nella catena (mining);
- Visualizzare l'intera struttura, blocchi e transazioni.

La base di dati gestisce i seguenti dati:

- I dati dell'utente: chiave pubblica (email), chiave privata (password) e il conto dell'utente;
- I dati delle transazioni: ogni transazione contiene la chiave pubblica del mittente e del destinatario, l'ammontare della transazione, l'orario e lo stato (in attesa/avviata/confermata);
- I dati dei blocchi: ogni blocco contiene 2 transazioni, le informazioni di identificazione e quelle dell'intera struttura. Inoltre contiene anche l'orario di creazione del blocco e la chiave pubblica della persona che ha validato il blocco stesso.

L'infrastruttura di rete:

- Non ci sono transazioni dirette tra mittente e destinatario, tutte le transazioni sono condivise con l'intera rete in broadcast per la verifica;
- Le transazioni sono verificate dai nodi e memorizzate in blocchi;
- Per aggiungere un blocco alla catena i nodi devono utilizzare le loro risorse per iniziare il processo di validazione e risolvere un puzzle crittografico (POW);
- Se più miner trovano una soluzione valida nello stesso momento, solo la catena più lunga viene considerata valida.

Priorità di realizzazione

Gli aspetti su cui ho lavorato di più sono l'esecuzione di una nuova transazione e il processo di mining.

Nuova transazione

Nella sezione in cui si effettua la transazione, all'utente viene chiesta la chiave pubblica del destinatario, la quantità di Lorenzo da trasferire, e la chiave privata del proprio account, senza la quale non può essere effettuata la transazione. Perché la transazione sia valida, devono essere eseguiti dei controlli:

1. La chiave privata inserita dall'utente deve essere corretta;
2. La somma da trasferire deve esser presente nel conto dell'utente;
3. Garantire il "conservation of value", la parte più importante della validità di una transazione.

La transazione da eseguire deve contenere i seguenti dati:

- *Transazioni in input*: l'indice della transazione è un hash composto da tutte le transazioni che l'utente ha ricevuto, in modo da riconoscere da quale fonte un utente ha guadagnato quella somma di Lorenzo che si intende trasferire.

- *Ammontare da trasferire*: la somma di Lorenzo.
- *Chiave pubblica del destinatario della transazione*: l'indirizzo email di chi riceverà i Lorenzo.

Una transazione è identificata univocamente dall'indice della transazione (ID), che rappresenta il valore hash delle transazioni input del mittente, dalla chiave pubblica del destinatario e dal timestamp della nuova transazione.

Per garantire il "conservation of value" vengono aggiornati i conti di mittente e destinatario. La somma in uscita da un conto deve essere uguale alla somma in entrata di un altro.

Una transazione deve essere pubblicata in un blocco valido. L'esistenza della transazione nel blocco conferma la sua validazione.

Mining

Nella sezione di mining viene rappresentato il *Proof of Work (POW)*, cioè la prova di quanto lavoro è stato effettuato per validare un blocco. Per accettare un blocco e la sua aggiunta alla blockchain è necessario risolvere un puzzle crittografico.

I miner sono i nodi che accumulano le transazioni verificate in un blocco e impiegano le loro risorse (come la potenza di calcolo o l'elettricità) per trovare un valore che renda il valore hash sha256 di questo blocco inferiore ad un valore target che varia dinamicamente.

Il blocco deve contenere il nonce, l'hash del blocco precedente, il Merkle root hash, il timestamp, e l'indice del blocco (versione).

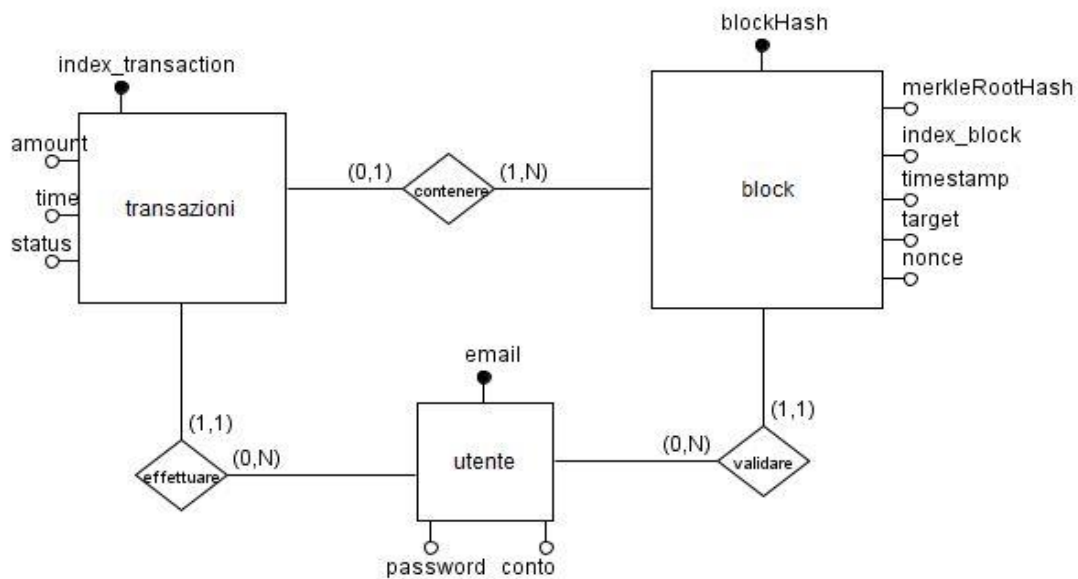
Progettazione del database

Schema E/R

Un utente non deve necessariamente effettuare transazioni o validare blocchi, ma se vuole può farlo più volte (0, N)

Una transazione deve essere effettuata per forza da un utente (1, 1) e se non è valida può non essere presente in nessun blocco (0, 1).

Un blocco deve contenere almeno una transazione (1, N) e deve essere validato solo da un utente (1, 1).



Modello relazionale

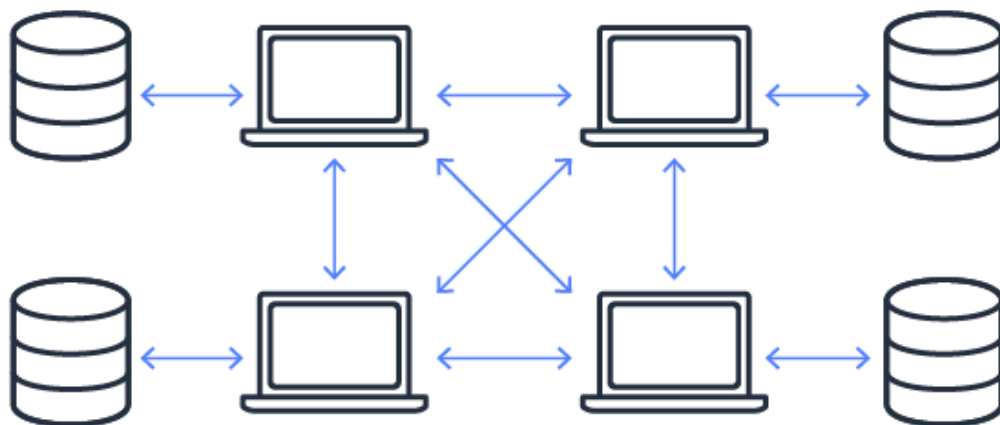
UTENTE (email PK, password, conto)

TRANSACTIONS (index_transaction PK, from_email FK, to_email FK, amount, time, status, index_block FK)

BLOCK (blockHash PK, merkleRootHash, previousBlockHash FK, index_block, timestamp, transaction1 FK, transaction2 FK, target, nonce, miner FK)

Nella **progettazione fisica** del database sono state utilizzate le tecniche di gestione delle transazioni per garantire la consistenza e l'affidabilità, nel momento in cui vengono eseguite operazioni di manipolazione dei dati. In questo modo, se devono essere effettuati più aggiornamenti correlati fra di loro, e uno di questi non dovesse avere esito positivo, tutti gli aggiornamenti vengono annullati.

Progettazione dell'architettura di rete



La blockchain decentralizzata sfrutta il passaggio di messaggi ad-hoc e un networking distribuito per fare in modo di memorizzare i dati su tutta la sua rete ed evitare di avere un single point of failure in modo che non esista una centralizzazione che i cracker potrebbero sfruttare per abbattere l'intero sistema.

Risorse hardware

Gli utenti della blockchain hanno usato vari tipi di hardware nel corso del tempo per creare e validare i blocchi:

- Mining attraverso CPU: utilizzata dalle prime versioni dei client
- Mining attraverso GPU: molto più veloce ed efficiente della CPU
- FPGA mining: consumano meno energia, più redditizi delle GPU
- ASIC mining: *Application Specific Integrated Circuits*, microchip specifici per il mining

Risorse software

In una rete decentralizzata e distribuita in cui gli utenti non sono fidati, come questa, il consenso è uno degli aspetti determinanti per certificare lo stato successivo della catena.

PBFT – Practical Byzantine Fault Tolerance Algorithm

I nodi mantengono uno stato attuale, che alla ricezione di un nuovo messaggio viene alimentato insieme al messaggio ricevuto per i calcoli, per aiutare il nodo a raggiungere una decisione. Questa decisione viene poi trasmessa alla rete. La maggioranza delle decisioni determina il consenso della rete.

POW – Proof of Work

È stato il primo protocollo di consenso decentralizzato, in cui i nodi della rete competono per calcolare il valore hash del blocco successivo, che dovrebbe essere inferiore ad un valore target che varia dinamicamente, determinato dalla regola del consenso.

POS – Proof of Stake

Questo algoritmo propone di acquistare la criptovaluta e di utilizzarla come partecipazione alla rete. La partecipazione è direttamente proporzionale alla possibilità di diventare il validatore del blocco. Per raggiungere il consenso, il validatore del blocco è selezionato in modo casuale e non è predeterminato. I nodi che producono blocchi validi ricevono incentivi, ma se il loro blocco non è incluso nella catena esistente, perdono anche una parte della loro quota di partecipazione.

Realizzazione degli archivi

-- Struttura della tabella 'block'

```

CREATE TABLE `block` (
  `blockHash` varchar(128) NOT NULL,
  `merkleRootHash` varchar(128) NOT NULL,
  `previousBlockHash` varchar(64) NOT NULL,
  `index_block` int(11) DEFAULT NULL,
  `timestamp` datetime NOT NULL,
  `transaction1` varchar(64) NOT NULL,
  `transaction2` varchar(64) NOT NULL,
  `target` int(11) NOT NULL,
  `nonce` int(11) NOT NULL,
  `miner` varchar(64) NOT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- Dump dei dati per la tabella `block`

INSERT INTO `block` (`blockHash`, `merkleRootHash`, `previousBlockHash`,
`index_block`, `timestamp`, `transaction1`, `transaction2`, `target`, `nonce`, `miner`)
VALUES

('27b469d72f2052f0000e35079c66be0af56f6b2a03e38d58481ofica3defc9c9',
'ab79400405b2471f14f927e86c68d49231de620ca30bb48c808768bb28bf742c',
'da1f6b70f235e2e074fcb9525bc21c7ac8c825b3a14c717633cb0701ea22901', 1, '2020-06-11
19:02:00', '84b75494dfdf2294ef2301e79c0756a4d2817c4b23b4bc482a31fbbeba9ff8f',
'da179706696fc252dde15428a0355313ee18ca05c0107c1eb9f056e39e3241a7', 10, 8,
'0d4b6071c87aadbd2da19cd66e1aa1f3a55b32940b61f5b227d071523fa8b53bf'),

('7d3cb051d98e878a5ad7c5daaaf3481d51e052ba9ae6baea310bd7e4c47f30a',
'7e6b02f81b6da181ee1d3508616036c40db0f1849bff4bcd6631194844ec3cf4',
'27b469d72f2052f0000e35079c66be0af56f6b2a03e38d58481ofica3defc9c9', 2, '2020-06-11
19:13:10', '7d55cd4b07af80c1db58b91a75a925983321f86a6a1cd69d892c432a014685bf',
'9668051ab8dcadaac2c36a0e98313bb81b835de6a70f72ce8f895d9d77fc4263', 2, 2,
'f72ea745f0995cfa10f236bd95f08713f39a0eb2942bbbd72f53cc089ffa430e'),

('da1f6b70f235e2e074fcb9525bc21c7ac8c825b3a14c717633cb0701ea22901',
'1c7f165648a9b89491a765772981611e44d12708bd178f05e0f30f8f5a27bee9',
'da1f6b70f235e2e074fcb9525bc21c7ac8c825b3a14c717633cb0701ea22901', 0, '2020-06-11
18:59:48', '7dc33eb2d976b050b0c14f4516d4977bf7229f46d83c9b58eef1b6764e57317b',
'f54e1b052a2a3121310fb79cce32429ee613fc35cdcfdb1f4027579cdd4a2feo', 10, 3,
'561b9ca3fe688437860c0oc22a1a764b6d5439c7cacb2ab378c88fe18c474ao8');

```

```

-- Struttura della tabella `transactions`

CREATE TABLE `transactions` (

  `index_transaction` varchar(64) NOT NULL,

  `from_email` varchar(64) NOT NULL,

  `to_email` varchar(64) NOT NULL,

  `amount` int(11) NOT NULL,

  `time` timestamp NOT NULL DEFAULT current_timestamp(),

  `status` varchar(10) NOT NULL DEFAULT 'pending',

  `index_block` int(11) DEFAULT NULL

) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;

-- Dump dei dati per la tabella `transactions`

INSERT INTO `transactions` (`index_transaction`, `from_email`, `to_email`,
`amount`, `time`, `status`, `index_block`) VALUES

('7d55cd4b07af80c1db58b91a75a925983321f86a6a1cd69d892c432a014685bf',
'3009be769fb8f956e8413ee9f3e0836e34968bc40457doaioc549d2edcfoocci',
'od4b6071c87aadbd2da19cd66e1aa1f3a55b32940b61f5b227do71523fa8b53bf', 100, '2020-06-
11 17:01:44', 'confirmed', 2),

('7dc33eb2d976b050b0c14f4516d4977bf7229f46d83c9b58eefib6764e57317b',
'3009be769fb8f956e8413ee9f3e0836e34968bc40457doaioc549d2edcfoocci',
'561b9ca3fe688437860c00c22a1a764b6d5439c7cacb2ab378c88fe18c474a08', 100, '2020-
06-10 15:00:19', 'confirmed', 0),

('84b75494dfdf2294ef2301e79c0756a4d2817c4b23b4bc482a31fbbefaf9ff8f',
'561b9ca3fe688437860c00c22a1a764b6d5439c7cacb2ab378c88fe18c474a08',
'f72ea745f0995cfa10f236bd95f08713f39a0eb2942bbbd72f53cco89ffa430e', 10, '2020-06-10
15:11:46', 'confirmed', 1),

('9668051ab8dcadaac2c36a0e98313bb81b835de6a70f72ce8f895d9d77fc4263',
'od4b6071c87aadbd2da19cd66e1aa1f3a55b32940b61f5b227do71523fa8b53bf',
'561b9ca3fe688437860c00c22a1a764b6d5439c7cacb2ab378c88fe18c474a08', 10, '2020-06-
11 17:12:25', 'confirmed', 2),

('da179706696fc252dde15428a0355313ee18ca05c0107c1eb9f056e39e3241a7',
'561b9ca3fe688437860c00c22a1a764b6d5439c7cacb2ab378c88fe18c474a08',
'f72ea745f0995cfa10f236bd95f08713f39a0eb2942bbbd72f53cco89ffa430e', 3, '2020-06-10
15:50:24', 'confirmed', 1),

```



```
('f54e1b052a2a3121310fb79cce32429ee613fc35cdcfdb1f4027579cdd4a2feo',  
'3009be769fb8f956e8413ee9f3e0836e34968bc40457doaioc549d2edcfoocci',  
'f72ea745f0995cfa10f236bd95f08713f39a0eb2942bbbd72f53cco89ffa430e', 100, '2020-06-  
10 15:11:21', 'confirmed', o);
```

-- Struttura della tabella `utente`

```
CREATE TABLE `utente` (  
  
    `email` varchar(64) NOT NULL,  
  
    `password` varchar(64) NOT NULL,  
  
    `conto` decimal(10,2) NOT NULL DEFAULT 0.00,  
  
    `nome` varchar(20) NOT NULL  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

-- Dump dei dati per la tabella `utente`

```
INSERT INTO `utente` (`email`, `password`, `conto`, `nome`) VALUES  
  
(  
'od4b6071c87aadb2da19cd66e1aaf3a55b32940b61f5b227d071523fa8b53bf',  
'64b4dof47c93ce23d157e68a58767356283dc9b63c459d45doeoe39b3a64b9b9', '90.00',  
'mike'),  
  
(  
'3009be769fb8f956e8413ee9f3e0836e34968bc40457doaioc549d2edcfoocci',  
'3009be769fb8f956e8413ee9f3e0836e34968bc40457doaioc549d2edcfoocci', '0.00',  
'network'),  
  
(  
'561b9ca3fe688437860c00c22a1a764b6d5439c7cacb2ab378c88fe18c474a08',  
'03fd72f81572805dd59f829b94fd8a6f82077fb435ca2b406d9595718e521afa', '97.00',  
'lorenzo'),  
  
(  
'f72ea745f0995cfa10f236bd95f08713f39a0eb2942bbbd72f53cco89ffa430e',  
'4ea7ea4917057a1fcbb3bffd673602d9b961ffi4b239cc7a8d96933b8a18b51', '133.00',  
'natalia');
```

-- Indici per le tabelle `block`

```
ALTER TABLE `block`  
  
    ADD PRIMARY KEY (`blockHash`),  
  
    ADD UNIQUE KEY `id_transaction1_FK` (`transaction1`) USING BTREE,  
  
    ADD UNIQUE KEY `id_transaction2_FK` (`transaction2`) USING BTREE,  
  
    ADD UNIQUE KEY `index_block` (`index_block`) USING BTREE,  
  
    ADD KEY `id_miner_FK` (`miner`),
```

```

ADD KEY `previousBlockHash_FK` (`previousBlockHash`) USING BTREE;

-- Indici per le tabelle `transactions`

ALTER TABLE `transactions`

ADD PRIMARY KEY (`index_transaction`),

ADD KEY `from_email_FK` (`from_email`),

ADD KEY `id_to_email_FK` (`to_email`),

ADD KEY `index_block` (`index_block`) USING BTREE;

-- Indici per le tabelle `utente`

ALTER TABLE `utente`

ADD PRIMARY KEY (`email`);

-- Limiti per la tabella `block`

ALTER TABLE `block`

ADD CONSTRAINT `id_miner_FK` FOREIGN KEY (`miner`) REFERENCES `utente`
(`email`) ON DELETE NO ACTION ON UPDATE NO ACTION,

ADD CONSTRAINT `id_transaction1_FK` FOREIGN KEY (`transaction1`)
REFERENCES `transactions` (`index_transaction`) ON DELETE NO ACTION ON
UPDATE NO ACTION,

ADD CONSTRAINT `id_transaction2_FK` FOREIGN KEY (`transaction2`)
REFERENCES `transactions` (`index_transaction`) ON DELETE NO ACTION ON
UPDATE NO ACTION,

ADD CONSTRAINT `previousBlockHash_FK` FOREIGN KEY (`previousBlockHash`)
REFERENCES `block` (`blockHash`) ON DELETE NO ACTION ON UPDATE NO
ACTION;

-- Limiti per la tabella `transactions`

ALTER TABLE `transactions`

ADD CONSTRAINT `from_email_FK` FOREIGN KEY (`from_email`) REFERENCES
`utente` (`email`) ON DELETE NO ACTION ON UPDATE NO ACTION,

ADD CONSTRAINT `id_to_email_FK` FOREIGN KEY (`to_email`) REFERENCES
`utente` (`email`) ON DELETE NO ACTION ON UPDATE NO ACTION,

ADD CONSTRAINT `index_block_FK` FOREIGN KEY (`index_block`) REFERENCES
`block` (`index_block`) ON DELETE NO ACTION ON UPDATE NO ACTION;

```

Realizzazione delle funzionalità

Nuova transazione:

Destinatario:

HASH destinatario

Lor€n\$0 .00

Password:

Password

Avvia transazione

```
//indice transazione
```

```
$$sql_UTXO = "SELECT * FROM transactions WHERE to_email = " . $email . " AND  
status='confirmed'";
```

```
//inserimento transazione nel database
```

```
$$sql1 = "INSERT INTO transactions (index_transaction, from_email, to_email, amount)  
VALUES (" . $index_transaction . ", " . $email . ", " . $to_email . ", " . $amount . ")";
```

```
//update conti mittente e destinatario
```

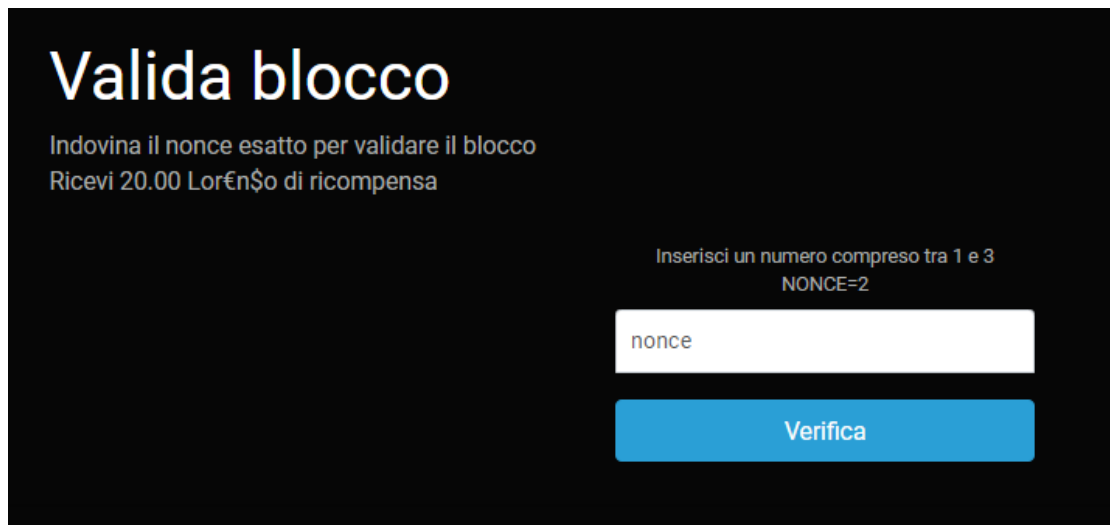
```
$$sql2 = "UPDATE utente SET conto = " . $conto_mittente . " WHERE email = " .  
$email . " ";
```

```
$$sql4 = "UPDATE utente SET conto = " . $conto_destinatario . " WHERE email = " .  
$to_email . " ";
```

```
//conferma transazione
```

```
$$sql5 = "UPDATE transactions SET status = 'started' WHERE index_transaction=" .  
$index_transaction . " ";
```

Mining blocco:



```
$blockHash = hash("sha256", $merkleRootHash . $previousBlockHash . $index_block .  
$timestamp . $transaction1 . $transaction2 . $target . $nonce . $miner);
```

```
$final_sql = "INSERT INTO block (blockHash, merkleRootHash, previousBlockHash,  
index_block, timestamp, transaction1, transaction2, target, nonce, miner) ";
```

```
$final_sql .= "VALUES ('$blockHash', '$merkleRootHash', '$previousBlockHash',  
$index_block, '$timestamp', '$transaction1', '$transaction2', $target, $nonce, '$miner')";
```

```
//inserimento indice blocco nella tabella transazioni
```

```
$sql_confirm1 = "UPDATE transactions SET status='confirmed',  
transactions.index_block = $index_block WHERE index_transaction=" . $transaction1 .  
""";
```

```
$sql_confirm2 = "UPDATE transactions SET status='confirmed',  
transactions.index_block = $index_block WHERE index_transaction=" . $transaction2 .  
""";
```

Cloud computing

Il cloud computing (in italiano “nuvola informatica”) indica un paradigma di erogazione di servizi offerti su richiesta da un fornitore a un cliente finale attraverso la rete internet (come l'archiviazione, l'elaborazione o la trasmissione dati), a partire da un insieme di risorse preesistenti, configurabili e disponibili in remoto sotto forma di architettura distribuita.

I servizi che possono essere erogati sfruttando infrastrutture cloud, sono di tre tipologie distinte.

IAAS – Infrastructure-as-a-Service

Lo IaaS è l'infrastruttura hardware che sta alla base di ogni servizio cloud. Il provider offre un hardware virtuale (CPU, RAM, spazio e schede di rete) e quindi la flessibilità di un'infrastruttura fisica, senza l'onere per l'utente, della gestione fisica dell'hardware. Questa tipologia è dedicata agli amministratori di sistema o sistemisti, i quali non gestiscono fisicamente la struttura ma le istanze da attivare, le caratteristiche network ed infine le risorse da utilizzare.

PAAS – Platform-as-a-Service

è un modello nel quale vengono situati i servizi di piattaforme online, grazie al quale un utente, di solito uno sviluppatore, può effettuare il deployment di applicazioni e servizi web che intende fornire. In questo caso, l'utilizzatore può sviluppare ed eseguire le proprie applicazioni attraverso gli strumenti forniti dal provider, il quale garantisce il corretto funzionamento dell'infrastruttura sottostante.

Esempi: Amazon Relational Database Service (RDS), Amazon DynamoDB, Amazon API Gateway, Google Cloud App Engine, Google Cloud SQL, Google Cloud Datastore, ecc.

SAAS – Software-as-a-Service

È la tipologia di servizio più completa. Il modello racchiude applicativi e sistemi software, accessibili da un qualsiasi tipo di dispositivo (computer, smartphone, tablet, ecc.), attraverso il semplice utilizzo di un'interfaccia client. In questo modo, l'utilizzatore non deve preoccuparsi di gestire le risorse e l'infrastruttura, in quanto controllati dal provider che li fornisce.

Esempi: G Suite

BAAS – Blockchain as a service

Si tratta di un interessante sviluppo nell'ecosistema della blockchain che viene visto come un incentivo all'adozione della blockchain in tutte le aziende.

Blockchain-as-a-Service consente alle aziende di utilizzare soluzioni basate sul cloud per costruire, ospitare e utilizzare le proprie applicazioni blockchain, smart contracts e funzioni sull'infrastruttura blockchain sviluppata da un fornitore. Proprio come la tendenza ad utilizzare SaaS in cui l'accesso al software è fornito su abbonamento, BaaS fornisce ad un'azienda l'accesso ad una rete blockchain della configurazione desiderata senza che l'azienda debba sviluppare una propria blockchain e costruire competenze interne in materia.

Molti dei principali fornitori di servizi cloud forniscono ora Blockchain-as-a-Service, tra cui IBM, Microsoft, Amazon, Alibaba, e Oracle.

Sviluppi futuri

Nell'informatica, l'endianness è l'ordine o la sequenza di byte di una parola nella memoria del computer o durante la trasmissione. L'endianness si esprime principalmente come big-endian o little-endian. I sistemi big-endian memorizzano il byte più significativo di una parola all'indirizzo di memoria più piccolo e il byte meno significativo al più grande. Un sistema little-endian, invece, memorizza il byte meno significativo all'indirizzo più piccolo.

Nello svolgimento di questo progetto, mi sono imbattuto nella Merkle root (o hash tree), un albero in cui ogni nodo foglia è etichettato con l'hash crittografico di un blocco di dati e ogni nodo non foglia è etichettato con l'hash crittografico delle etichette dei suoi nodi figlio. Nel calcolo di questo albero di hash andrebbero però utilizzati i due algoritmi di memorizzazione dei dati nella memoria, che vorrei applicare in una rivisitazione del progetto in un futuro.

Le transazioni iniziali sono nella rappresentazione big endian. Bisogna cambiare l'ordine dei bytes in little endian, e infine cambiare nuovamente l'ordine dei bytes della root per ottenere la rappresentazione in big endian, così da confrontarla con quella del blocco originale, e verificando l'integrità dei dati delle transazioni che sono state inserite nel blocco. Lavoro che attualmente esula da questo elaborato.

Conclusioni

Il progetto che ho realizzato permette di effettuare transazioni tra persone. In alcuni settori può certificare lo scambio di titoli e azioni, operare come fosse un notaio e "vidimare" un contratto o rendere sicuri e non alterabili i voti espressi tramite votazione online (democrazia partecipativa).

I punti di forza, come già detto, sono rappresentati da immutabilità, irreversibilità, decentralizzazione, persistenza e anonimato, caratteristiche essenziali per queste transazioni. Grazie a questi aspetti la blockchain ha trovato applicazione in diversi campi che richiedono la condivisione di dati ma in sicurezza. Tuttavia gli svantaggi sono rappresentati dal raggiungere rapidamente il consenso in una vasta rete (più difficile), consumo di energia in calcoli computazionali e dalla richiesta di memorizzare l'intera blockchain per la verifica (occupazione della memoria).

Il tempo impiegato per questo progetto è stato all'incirca di 14 giorni, per un totale di 70 ore (circa 5 ore al giorno), che è il tempo che avevo già preventivato come tempo necessario per lo svolgimento dell'elaborato.

Durante lo svolgimento di questo progetto ho incontrato diverse difficoltà, dalla comprensione della tecnologia blockchain, alla scrittura del codice per applicare le funzionalità di questa tecnologia al sito che ho creato. Le ho risolte documentandomi su internet e approfondendo la mia conoscenza in materia.

È la prima volta che realizzo un progetto di questo tipo e sono rimasto molto soddisfatto, nonostante la mia inesperienza. Man mano che procedevo con lo

svolgimento di questo compito mi sono sentito sempre più motivato perché mi sono reso conto dell'interesse che suscitava in me questo tipo di lavoro.

Spero in futuro di potermi ancora cimentare nella realizzazione di progetti riguardanti questa tecnologia.