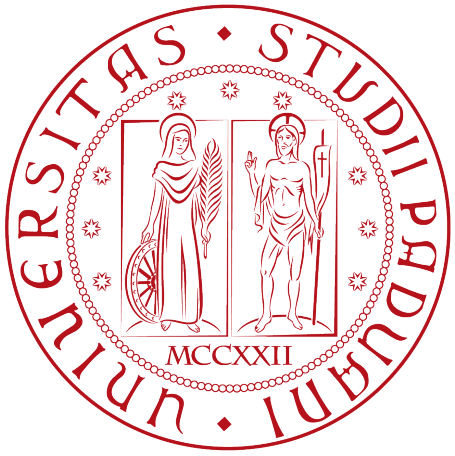
Università degli Studi di Padova

Dipartimento di Matematica "Tullio Levi-Civita"

Corso di Laurea in Informatica



Prospettive e Sfide dell'Identità Decentralizzata nell'era della Blockchain

Tesi di laurea

Relatore

Prof. Alessandro Brighente

Correlatore

Dott. Mattia Zago

Laureando

Nicola Lazzarin

Anno Accademico 2022-2023

Nicola Lazzarin: *Prospettive e Sfide dell'Identità Decentralizzata nell'era della Blockchain,* Tesi di laurea, © Dicembre 2023.

”We didn’t realise we were making memories, we just knew we were having fun”

— A. A. Milne, Winnie the Pooh

Abstract

Il presente documento descrive il lavoro svolto durante il periodo di stage, della durata di circa 276 ore, dal laureando Nicola Lazzarin presso l'azienda *Athesys s.r.l*, con periodo dal 06/07/2023 al 23/07/2023.

Il paradigma delle identità digitali è fondamentale nell'era digitale, ma spesso sottolinea l'inefficienza dei tradizionali sistemi centralizzati. L'emergere delle *Self-sovereign identity (SSI)* offre un nuovo approccio promettente per migliorare la gestione delle identità digitali e delle *Verifiable Credentials (VC)*. Tuttavia, l'attuazione pratica di soluzioni basate su *SSI*, in particolare utilizzando il *framework Hyperledger Aries*, rimane una sfida complessa e impegnativa.
La presente tesi trae ispirazione dalla rilevanza delle identità digitali nella società odierna e dalla crescente necessità di soluzioni *SSI*. Tuttavia, essa si concentra su una prospettiva critica basata sull'esperienza personale dell'autore, che, nonostante gli sforzi, non è riuscito a pieno a completare l'implementazione di un sistema di scambio di *VC* utilizzando il *framework Hyperledger Aries Javascript*.

Il principale contributo di questa tesi è rappresentato dalla realizzazione parziale di un *Proof of Concept (PoC)* per un sistema di scambio di *VC* basato sul *framework Hyperledger Aries Javascript*. Questo *PoC* offre una base incompleta per ulteriori sviluppi nel campo delle *SSI*. Inoltre, questa ricerca fornisce una documentazione dettagliata sul processo di implementazione, comprese le scelte di progettazione e le tecnologie utilizzate, per agevolare ulteriori studi e implementazioni nel settore delle

identità digitali decentralizzate.

In conclusione, questo tirocinio ha fornito un'opportunità di apprendimento e applicazione pratica di concetti avanzati nel campo delle identità digitali e delle credenziali verificabili.

Ringraziamenti

Innanzitutto, vorrei esprimere la mia gratitudine al Dott. Mattia Zago e al Prof. Alessandro Brighente, relatore della mia tesi, per l'aiuto e il sostegno fornitomi durante la stesura del lavoro.

Desidero ringraziare con affetto i miei genitori e mia sorella per essermi stati accanto con amore e sostegno in ogni momento.

Un ringraziamento speciale lo dedico a Caterina, per il suo aiuto e la sua determinazione nel farmi raggiungere questo obiettivo.

Ringrazio infine i miei amici per aver alleggerito il mio percorso con tutti i bellissimi anni passati insieme e le mille avventure vissute.

Padova, Dicembre 2023 Nicola Lazzarin

Indice

Elenco delle figure

Capitolo 1

Introduzione

1.1 Contesto

Viviamo in un'era in cui gran parte delle nostre attività si svolge online, dalla comunicazione ai pagamenti, dall'accesso ai social media, all'acquisto di beni e servizi. Tuttavia, Internet ha sì espanso l'Io in differenti forme, fornendo all'utente la possibilità di collegarsi e collegare il proprio sé con un numero di individui impensabile fino a pochi decenni prima, ma contemporaneamente non è stato costruito per garantire al soggetto il pieno controllo sulla propria identità[1].

Kim Cameron cercò di spiegare il problema nel suo articolo: *The Laws of Identity* mostrando che *"The Internet was built without an identity layer"*[2]. Ma cosa intendeva *Kim Cameron* nel lontano 2005?

Internet è stato creato senza un modo per sapere a chi e a cosa ti stai collegando. Questo limita ciò che possiamo farci e ci espone a pericoli crescenti. Se non facciamo nulla, affronteremo rapidamente episodi di furto e inganno che eroderanno cumulativamente la fiducia del pubblico in Internet[3]. Quello che *Kim Cameron* stava cercando

[1] *Self Sovereign Identity*. URL: https://www.selfsovereignidentity.it/introduzione-alla-self-sovereign-identity/.
[2] Kim Cameron. «The Laws of Identity». In: *Architect of Identity* (2005).
[3] Ibid.

di preannunciare era che Internet, forse, non era ancora pronto, per come era stato progettato agli albori, a risolvere il problema dell'identità digitale. Ed in fondo come non dargli torto.

Quello che mancava per *Cameron* era un modo per identificare la persona sulla rete, un protocollo informatico per l'identità (*identity layer*). Ad oggi ci sono miliardi di persone e miliardi di dispositivi connessi a Internet, i quali quasi tutti estranei l'uno all'altro.
Applicazioni come *Facebook, Gmail, LinkedIn*, riuscirono, almeno inizialmente, a colmare il *gap* dell'identità, portando però la stessa ad essere gestita in maniera "*federata*", piuttosto che centralizzata. L'identità digitale, gestita attraverso organizzazioni centralizzate, ha semplificato l'identificazione degli utenti online. Tuttavia, ha anche sollevato preoccupazioni significative relative alla sicurezza e al controllo dei dati personali. La centralizzazione dei dati delle identità digitali ha reso questi dati un obiettivo primario per la criminalità informatica, creando una minaccia economica considerevole.
Inoltre, con l'incremento della quantità di dati personali online, cresce l'interesse nel loro accesso non autorizzato.
Si cercano quindi nuovi modelli per una maggior sicurezza dei propri dati personali. Da qualche anno si sta parlando molto di *Identità Decentralizzata* e di *blockchain*, due tecnologie che messe assieme formano la *Self-sovereign identity (SSI)*, un modello di Identità Decentralizzata. Questo modello garantisce completa trasparenza e immutabilità dei dati.

1.2 Organizzazione del testo

Il secondo capitolo Descrive in dettaglio lo scopo dello stage con tutti i vari requisiti e obiettivi.

Il terzo capitolo Introduce i preconcetti necessari per la comprensione della tesi.

Il quarto capitolo Spiega i vari esperimenti condotti durante lo sviluppo del *Proof of Concept (PoC)*.

Il quinto capitolo Discute quanto appreso.

Il sesto capitolo Conclusioni.

Riguardo la stesura del testo, relativamente al documento sono state adottate le seguenti convenzioni tipografiche:

- gli acronimi, le abbreviazioni e i termini ambigui o di uso non comune menzionati vengono definiti nel glossario, situato alla fine del presente documento;

Capitolo 2

Descrizione del tirocinio

2.1 L'azienda

Athesys s.r.l è un *system integrator* che si occupa principalmente di soluzioni di *Identity and Access Managemen (IAM)*. Nasce nel 2010 dalla sinergia di affermati professionisti del settore *IT*, con lo scopo di offrire consulenza altamente specializzata in ambito *System Integration*, *Database Management*, Sicurezza applicativa, *Governance Cloud Platform*, *Hyperconvergenza* e Sviluppo Software in modalità *Agile*.

Figura 2.1: Immagine scaricata dal sito ufficiale di Athesys s.r.l[1]

2.2 L'idea

Lo scopo dello stage è quello di sviluppare un *Proof of Concept (PoC)* utilizzando la tecnologia *Hyperledger Aries* per il design e l'implementazione di un sistema di scambio di *Verifiable Credentials (VC)*. Il *PoC* sarà integrato con *Monokee IAM*, un framework

[1] https://athesys.it

di gestione delle identità e degli accessi, al fine di fornire un'interfaccia visuale per la costruzione dei processi di autenticazione.

Figura 2.2: Immagine scaricata dalla pagina GitHub ufficiale[2]

2.3 Requisiti ed obiettivi

Nel corso del tirocinio, il candidato ha avuto una completa formazione su diverse tecnologie, comprese *Self-sovereign identity (SSI)* e *API REST*. Il progetto mirava a realizzare un sistema per lo scambio di *VC*, utilizzando il framework *Hyperledger Aries JavaScript* mediante richieste *HTTP* tramite *API REST*. La gestione delle chiamate era affidata al framework *Monokee IAM* con interfaccia *frontend*.

Si farà riferimento ai requisiti mediante queste notazioni:

- **OB**: Requisito obbligatorio
- **DE**: Requisito desiderabile ma non strettamente necessario, dal valore aggiunto
- **OP**: Requisito opzionale

Le abbreviazioni menzionate in precedenza saranno seguite da una coppia numerica sequenziale che identifica il requisito. In particolare, abbiamo pianificato il conseguimento dei seguenti traguardi:

Obbligatori

- **OB01** - Studio, analisi e comprensione del paradigma *SSI* e delle sue implicazioni per l'identità digitale.
- **OB02** - Implementazione di un *PoC* utilizzando il framework *Hyperledger Aries* per il *server agent* utilizzando *JavaScript* per emissione e verifica di credenziali verificabili, richiesta e verifica di presentazioni verificabili.

[2] https://github.com/hyperledger/aries

Desiderabili

- **DE01** - Implementazione di un *PoC* per il *mobile agent* (applicazione a scelta dallo store) che possa interagire con il *server agent*.
- **DE02** - Aggiunta di un sistema di revoca delle credenziali verificabili al *server agent* di *Hyperledger Aries*.

Opzionali

- **OP01** - Documentazione di integrazione che descriva il processo di integrazione tra la *PoC* sviluppata e il framework *Monokee IAM*.
- **OP02** - Realizzazione di un manuale d'uso che fornisca istruzioni dettagliate sull'utilizzo e la configurazione del sistema sviluppato.

2.4 Pianificazione

Il tirocinio è stato suddiviso in tre macro sezioni. La prima sezione consisteva nella formazione sulle tecnologie, la seconda nella modellazione e implementazione del software e la terza nella scrittura della documentazione. Un resoconto lo si può trovare nella figura 2.3

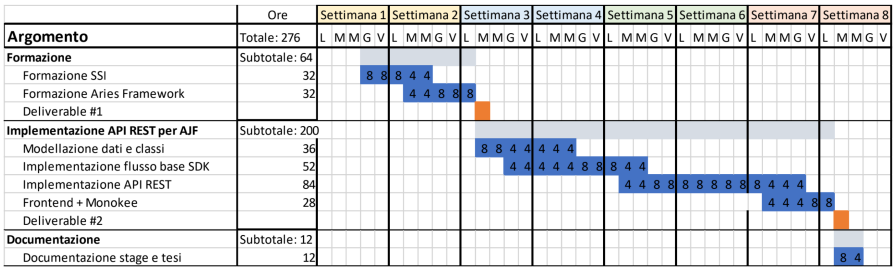


Figura 2.3: Il calendario di tirocinio. Immagine proveniente dal piano di lavoro personale.

2.4.1 Formazione

Durante la formazione sono stati approfonditi i seguenti temi:

- Formazione *SSI* (32 ore): Approfondimento teorico sul paradigma *SSI* e sulle sue applicazioni pratiche.
- Formazione *Hyperledger Aries* (32 ore): Studio approfondito del framework *Hyperledger Aries Javascript* e delle sue funzionalità.

Alla fine del primo periodo è stato sviluppato il seguente prodotto:

- *Deliverable 1*:

  Composto da:

  - Materiale di descrizione del progetto *Hyperledger Aries Javascript* e delle credenziali verificabili per la Knowledge Base interna aziendale.

2.4.2 Modellazione e implementazione del software

Durante la modellazione e implementazione del software sono stati sviluppati i seguenti prodotti:

- Modellazione dati e classi (36 ore): Progettazione del modello dati necessario per la gestione delle Verifiable Credentials (VC).
- Implementazione flusso base con SDK (52 ore): Sviluppo del flusso di scambio delle VC utilizzando l'SDK di *Hyperledger Aries Javascript*.
- Implementazione *API REST* (84 ore): Creazione di *API RESTful* per consentire l'interazione con l'applicazione attraverso richieste *HTTP*.
- Frontend (28 ore): Realizzazione dell'interfaccia utente per consentire agli utenti di interagire con il sistema.

Alla fine del secondo periodo è stato sviluppato il seguente prodotto:

- *Deliverable 2*:

  Composto da:

  - *PoC* funzionante del sistema di scambio di credenziali verificabili con integrazione *Monokee IAM*
  - Report completo delle attività svolte durante l'implementazione, inclusi i dettagli tecnici, le decisioni di progettazione e i risultati ottenuti.

2.4.3 Scrittura della documentazione

Durante la scrittura della documentazione e stato sviluppato il seguente prodotto:

- Documentazione stage e tesi (12 ore): Stesura di una documentazione dettagliata che descriva il lavoro svolto durante il tirocinio, inclusi gli obiettivi raggiunti, le tecnologie utilizzate e decisioni di progettazione prese.

2.5 Modalità di svolgimento

L'attività di stage è stata svolta in regime di *smart working* dalle 09:00 alle 18:00 con 1h di pausa pranzo. Il lavoro è stato svolto sul proprio computer personale senza vincoli da parte di *Athesys s.r.l* sul luogo di lavoro. *Athesys* ha messo a disposizione spazi e risorse presso il suo *HQ* a Padova (Via Longhin 79) qualora lo studente preferisca lavorare fisicamente in presenza.

Una riunione settimanale è stata pianificata con il tutor aziendale ed il management per l'allineamento sullo stato avanzamento lavori (indicativamente di lunedì pomeriggio). Ogni due settimane è stata prevista una riunione generale con i membri del team di sviluppo per *brainstorming* e validazione risultati.

Capitolo 3

Background

Nel seguente capitolo, esploreremo il mondo delle identità digitali e le loro tecnologie. Queste sezioni sono fondamentali per comprendere il concetto di identità digitale e come essa sta rivoluzionando il modo in cui interagiamo online. Inizieremo distinguendo tra l'identità cartacea tradizionale e quella digitale, per poi approfondire il ruolo della tecnologia *blockchain* in questo contesto. Scopriremo i diversi modelli di identità digitale, dai sistemi centralizzati a quelli federati, fino ad arrivare al modello di identità decentralizzata. All'interno di quest'ultimo, esamineremo concetti chiave come la *Self-sovereign identity (SSI)*, i *digital wallet SSI*, le *Verifiable Credentials (VC)* e i *Decentralized Identifiers (DID)*. Inoltre, approfondiremo l'importante ruolo di *Hyperledger Foundation* e del suo progetto *Hyperledger Aries*. Infine, concluderemo con un'analisi delle *7 leggi dell'identità* di *Kim Cameron*, che ci aiuteranno a comprendere meglio l'importanza di gestire le identità digitali per un futuro più sicuro e responsabile.

3.1 Identità digitale

L'identità digitale si definisce come l'insieme di dati e informazioni legati a un individuo o a un'entità sulla rete. Questi dati svolgono un ruolo cruciale nell'identificazione e nell'autenticazione degli utenti durante l'accesso a servizi online, piattaforme web, reti sociali, *e-commerce* e altre risorse digitali.

In dettaglio, è possibile suddividere l'identità digitale in due componenti distintive:

- A chi o cosa è riferita quella entità
- Che credenziali possiede quella entità

Un semplice esempio di identità digitale potrebbe essere un profilo su un social media come *Facebook, LinkedIn o Twitter*. Questi profili rappresentano una parte della nostra identità digitale e spesso includono informazioni personali, foto e connessioni sociali. Poiché l'identità digitale coinvolge questioni legate alla privacy, alla riservatezza, all'autorizzazione e all'integrità dei dati, è essenziale affrontare tali problematiche.
Senza una garanzia del rispetto di questi aspetti, l'identità digitale perderebbe la sua validità formale e potrebbe comportare rischi per la privacy e la sicurezza dell'individuo. Pertanto, la gestione adeguata dell'identità digitale è fondamentale per garantire che le informazioni personali siano trattate in modo sicuro e che l'utente mantenga il controllo su chi ha accesso a tali dati e a quali scopi.

3.1.1 Differenza fra Identità cartacea e digitale

- **Identità cartacea**

  L'identità cartacea è rappresentata da documenti fisici come carta d'identità, patente di guida, passaporto e altri documenti simili emessi dal governo o da enti autorizzati. I documenti possono essere portati fisicamente con sé, ma sono sottoposti a rischi di smarrimento, furto o danneggiamento.

  Nel momento in cui presentiamo un documento per confermare la nostra identità o le nostre qualifiche, chi lo verifica solitamente ha accesso a tutte le informazioni contenute in quel documento. Molte di queste informazioni, tuttavia, potrebbero non essere necessarie per il contesto in cui viene effettuata la verifica. In più in caso di smarrimento o furto il processo di riottenimento del documento è burocraticamente lungo e costoso;

- **Identità digitale**

  L'identità digitale è costituita da dati e informazioni immagazzinati in formato digitale su dispositivi elettronici, spesso protetti da password, autenticazioni

biometriche o altre forme di sicurezza. Sono utilizzate online e in applicazioni digitali per l'accesso a servizi, l'invio di messaggi, la partecipazione a reti sociali e altro. L'accesso alle identità digitali avviene attraverso metodi di autenticazione elettronica, come password, riconoscimento facciale o impronte digitali. Tuttavia, in queste situazioni, non si ha sempre il controllo completo su quali dati vengano effettivamente condivisi e con chi.

Molto spesso le identità digitali sono gestite da un *Identity Provider (IdP)* ovvero da servizi di terze parti per la gestione delle identità. Questi servizi fanno largo uso dei nostri dati, come ad esempio il tracciamento delle nostre ricerche online per una personalizzazione delle pubblicità, o ancora peggio, la vendita dei propri dati personali a servizi terzi[1]. Inoltre, è comune che i nostri dati personali siano archiviati nei *server* dell'ente emittente, e queste banche dati centralizzate spesso diventano obiettivi per potenziali attacchi, comportando così rischi significativi.

3.2 Blockchain

La *blockchain* è una tecnologia di registrazione digitale che consente di memorizzare una serie di record o transazioni in modo sicuro e immutabile, utilizzando una rete distribuita di computer. È spesso associata alle *criptovalute*, come il *Bitcoin*, ma ha molteplici applicazioni al di là delle valute digitali. Le caratteristiche principali della *blockchain* includono:

- **Decentralizzazione**: La *blockchain* non è controllata da una singola entità o autorità centrale. Invece, i dati vengono distribuiti su una rete *peer-to-peer* di nodi (computer) che partecipano alla rete;
- **Immutabilità**: Le transazioni o i record memorizzati in una *blockchain* non possono essere modificati o cancellati una volta confermati e aggiunti al registro. Questa immutabilità è garantita dalla *crittografia* e dal consenso della rete;

[1] *TEMU App is Cleverly Hidden Spyware*. URL: https://grizzlyreports.com/we-believe-pdd-is-a-dying-fraudulent-company-and-its-shopping-app-temu-is-cleverly-hidden-spyware-that-poses-an-urgent-security-threat-to-u-s-national-interests.

- **Sicurezza**: Le informazioni memorizzate in una *blockchain* sono *crittografate* e protette da algoritmi di *crittografia avanzati*. Inoltre, la rete richiede il consenso della maggioranza dei nodi per validare le transazioni, il che la rende altamente sicura;
- **Trasparenza**: La *blockchain* è spesso pubblica e aperta a chiunque voglia esaminare le transazioni. Questo favorisce la trasparenza e la fiducia nell'integrità del sistema;
- **Registro distribuito**: Tutti i nodi nella rete *blockchain* mantengono una copia dello stesso registro. Questo elimina la necessità di una terza parte di fiducia e riduce il rischio di errori o frodi.

3.2.1 Confronto fra Blockchain Permissionless e Permissioned

Un confronto fra le due tipologie di *blockchain* e la gestione degli accessi ad esse.

Blockchain Permissioned

Rappresenta una forma di registro distribuito in cui solo individui autorizzati hanno il permesso di accedere e svolgere azioni specifiche. Tale approccio offre un livello addizionale di sicurezza e riservatezza. Di conseguenza, qualsiasi modifica eseguita può essere facilmente tracciata e gli utenti possono essere nel caso identificati. Nuovi utenti possono essere inclusi o tolti e hanno la capacità di accedere ai dati attraverso la verifica digitale o altri tipi di certificati. La *blockchain* autorizzata si distingue notevolmente dalla sua controparte pubblica poiché l'accesso è rigorosamente controllato. Questo modello è ampiamente adottato dalle grandi imprese e sta guadagnando sempre maggiore popolarità grazie alla sua robusta sicurezza dei dati.

Pro

- Essendo un modello decentralizzato, le imprese possono agevolmente partecipare evitando i rischi associati ai modelli centralizzati;

- Questo sistema può essere facilmente personalizzato tramite diverse configurazioni, adattandosi alle esigenze specifiche. Gli utenti possono sfruttare integrazioni ibride o componenti modulari;
- Poiché l'accesso è ristretto, poche modalità gestiscono le transazioni, contribuendo a migliorarne le prestazioni e la scalabilità.

Contro

- Dato il numero limitato di partecipanti, si accresce il rischio di collusioni e manipolazioni dei dati;
- Ottenere il consenso in questo modello non è agevole poiché le regole di consenso possono essere modificate dall'operatore in qualsiasi momento;
- La partecipazione limitata comporta minore trasparenza nei confronti della supervisione esterna.

Blockchain Permissionless

Rappresenta l'opposto di una *blockchain permissioned*. Nel modello *permissionless*, comunemente noto come *blockchain* pubblica, non esistono restrizioni e la partecipazione non è regolata da un amministratore. Ognuno può contribuire al consenso e verificare i dati. Non vi sono amministratori che autorizzano l'ingresso agli utenti o concedono loro il permesso e i diritti per apportare modifiche. Si tratta di una piattaforma *blockchain* completamente decentralizzata e con partecipanti anonimi.

Pro

- Ha una piattaforma decentralizzata più ampia, che consente un maggior numero di partecipanti attraverso la rete rispetto alla *blockchain permissioned*;
- Ha un alto livello di trasparenza e accelera il processo di riconciliazione. Avendo un gran numero di partecipanti, è molto resistente alla censura;

- È quasi impossibile interrompere il *repository*, poiché è necessario che il 51% della rete sia corrotto e corrompere una così grande quantità di dati è quasi impossibile.

Contro

- Minore privacy rispetto alle *blockchain permissioned*;
- Prestazioni e scalabilità ridotte, poiché un maggior numero di utenti comporta una maggiore pressione sulle risorse informatiche;
- Il modello non è efficiente dal punto di vista energetico, poiché la verifica a livello di rete utilizza molte risorse in modo intensivo.

3.3 Attuali modelli di Identità digitale

I modelli di identità digitale sono le modalità con cui le organizzazioni gestiscono l'identità digitale degli utenti. Esistono due principali modelli: *centralizzato* e *federato*

3.3.1 Centralizzato

Il modello centralizzato di identità digitale, noto anche come *"modello a silos"*, si basa sul concetto che le organizzazioni che creano l'identità dell'utente per i propri servizi fungano da punto centrale nella gestione delle identità. Gli utenti creano un account e l'organizzazione gestisce l'identità centralizzando i dati personali all'interno dei loro database interni. Tuttavia, questo modello comporta alcuni rischi, come la possibile esposizione dei dati sensibili degli utenti in caso di violazione di sicurezza informatica. Inoltre, i sistemi di identità digitale centralizzati possono creare il fenomeno delle identità multiple, in cui gli utenti devono gestire diverse identità digitali per ogni servizio utilizzato. Sebbene sia un modello semplice da utilizzare e implementare per le organizzazioni, presenta alcune debolezze in termini di sicurezza e user experience.

Pro

- **Semplicità di implementazione**: Il modello centralizzato è relativamente semplice da implementare per le organizzazioni. Consente loro di gestire i dati degli utenti in modo efficace e mantenere una relazione costante con gli utenti;
- **Controllo delle informazioni**: Le organizzazioni che utilizzano questo modello mantengono il controllo completo delle informazioni dell'utente, il che può essere importante per scopi di sicurezza e conformità normativa;
- **Facilità di accesso ai servizi**: Gli utenti possono accedere facilmente ai servizi associati a un'organizzazione centralizzata utilizzando le loro credenziali di accesso uniche, come password o PIN.

Contro

- **Vulnerabilità dei dati**: Poiché la maggior parte dei dati personali è conservata in database centralizzati, c'è il rischio di violazioni dei dati che possono portare all'esposizione di informazioni sensibili dell'utente;
- **Fenomeno delle identità multiple**: Il modello centralizzato ha portato alla proliferazione di identità digitali separate per diversi servizi, rendendo l'esperienza online dell'utente più complicata e difficile da gestire;
- **Dipendenza da organizzazioni**: Gli utenti diventano dipendenti dalle organizzazioni che detengono i loro dati, senza un controllo diretto sulla propria identità digitale. Ciò può sollevare preoccupazioni sulla privacy e sulla gestione delle informazioni personali;
- **Mancanza di incentivi per migliorare l'esperienza dell'utente**: Il modello centralizzato spesso si concentra sulla gestione dei dati per le organizzazioni, piuttosto che sull'ottimizzazione dell'esperienza dell'utente. Ciò può portare a problemi come la frammentazione delle identità digitali.

3.3.2 Federato

Il modello di gestione centralizzato delle identità online presenta delle limitazioni nella gestione dell'esperienza utente. Di conseguenza, si è sviluppato un modello federato in

cui *Identity Provider (IdP)* svolge un ruolo cruciale come intermediario tra l'utente e il servizio. In questo modello, *IdP* detiene l'identità digitale dell'utente e i relativi dati, consentendo all'utente di usufruire dei servizi attraverso l'*IdP* stesso. *Sistema Pubblico di Identità Digitale (SPID)* è un esempio di *IdP* italiano, in cui un'identità unica consente l'accesso a diversi servizi. *Google* è un altro esempio di gestione federata dell'identità digitale in cui l'utente può accedere a vari servizi mediante un singolo account *Google*. Questo modello di gestione delle identità evita la presenza di identità multiple per gli utenti e permette un'esperienza di accesso con un solo login. Tuttavia, l'utente non è il proprietario effettivo dei suoi dati digitali, che rimangono in possesso del fornitore di identità, mettendo a rischio sia la privacy dell'utente che l'intera identità digitale stessa.

Pro

- **Single Sign-On (SSO)**: Un importante vantaggio del modello federato è la possibilità di utilizzare un'unica identità per accedere a diversi servizi. Questo semplifica notevolmente l'esperienza dell'utente, eliminando la necessità di ricordare numerose credenziali diverse;
- **Evitare identità multiple**: Il modello federato aiuta a evitare il problema delle identità multiple. Gli utenti possono utilizzare una sola identità per una varietà di servizi, riducendo la complessità e il rischio di dimenticare le credenziali;
- **Facilità di gestione per i fornitori di servizi**: Per i fornitori di servizi, il modello federato semplifica l'integrazione con sistemi di identità. Non devono gestire direttamente l'archiviazione delle credenziali utente, ma possono fare affidamento sui servizi dell'*IdP*.

Contro

- **Dipendenza dagli *identity provider***: Nel modello federato, l'identity provider diventa un attore cruciale. Gli utenti sono dipendenti da tali fornitori per accedere a servizi online. Se un *IdP* dovesse avere problemi tecnici o decidere di interrompere i servizi, gli utenti potrebbero subire inconvenienti[2];

[2] *A dad took photos of his naked toddler for the doctor. Google flagged him as a criminal.* URL:

- **Privacy e sicurezza**: L'*IdP* detiene l'identità digitale dell'utente e può avere accesso ai dati relativi all'uso dei servizi. Ciò solleva preoccupazioni sulla privacy, poiché l'*IdP* potrebbe monitorare l'attività dell'utente. È fondamentale che i fornitori di servizi e gli *IdP* rispettino le normative sulla privacy e implementino robuste misure di sicurezza;

- **Centralizzazione dei dati**: Anche se non è centralizzata come nel modello centralizzato, la federazione comporta ancora una certa centralizzazione dei dati presso gli *IdP*. Questo può essere un problema se un *IdP* subisse una violazione dei dati;

- **Interoperabilità limitata**: Il modello federato richiede che i servizi siano compatibili con il protocollo o standard di federazione utilizzato. Questo potrebbe limitare l'accesso a determinati servizi che non supportano il modello federato.

3.4 Modello *Self-sovereign identity (SSI)*

Un modo nuovo e completamente rivoluzionario di gestire l'identità digitale è rappresentato dal paradigma della *Self-sovereign identity (SSI)*. Il concetto di *SSI* è completamente incentrato sull'utente, che è l'unico e indipendente proprietario della propria identità digitale e di tutti i dati ad essa associati.

La *SSI* è un modello che si differenzia dai precedenti e mira a garantire che l'utente rimanga l'unico proprietario dei propri dati (dal termine *Self-Sovereign*), grazie all'utilizzo di protocolli come la *Blockchain*. Infatti, i modelli sopra descritti utilizzano un "sistema di nomi centralizzato" come database per memorizzare le varie informazioni relative all'identità. SSI lo sostituisce utilizzando la *blockchain*, creando quello che viene definito un "sistema di nomi decentralizzato". Va inoltre sottolineato che grazie alle caratteristiche essenziali di una *blockchain*, come l'immutabilità e la resilienza (nessun tempo di inattività), il modello della *Self-Sovereign Identity* vanta una maggiore

https://www.forbesindia.com/article/news/a-dad-took-photos-of-his-naked-toddler-for-the-doctor-google-flagged-him-as-a-criminal/79159/1.

sicurezza per tutti gli attori coinvolti[3].

Pro

- **Controllo dell'Utente**: Gli utenti mantengono il controllo completo delle proprie informazioni personali, decidendo chi può accedervi e come vengono utilizzate;
- **Privacy Potenziata**: L'utente può condividere solo le informazioni necessarie, riducendo la divulgazione non necessaria di dati personali;
- **Sicurezza Migliorata**: L'archiviazione decentralizzata delle informazioni e le firme digitali aumentano la sicurezza e la resistenza ai furti di identità;
- **Interoperabilità**: L'identità digitale *SSI* è progettata per funzionare su diverse piattaforme e servizi, migliorando l'interoperabilità;
- **Riduzione dell'Uso di Password**: L'*SSI* può ridurre la dipendenza da password, riducendo il rischio di accessi non autorizzati.

Contro

- **Recupero delle Credenziali**: Nel caso in cui l'utente perda l'accesso alle proprie credenziali digitali, il recupero è praticamente impossibile, poiché non esiste un intermediario centrale a cui rivolgersi e non esiste nessun: hai dimenticato la password?

  "Your keys, your coin. Not your keys, not your coin"[4]

- **Sicurezza Personale**: Gli utenti devono essere responsabili della sicurezza delle proprie informazioni, il che potrebbe comportare rischi se non trattato con attenzione;

[3] *Digital Identity Models.* URL: https://www.selfsovereignidentity.it/identity-management-models/.
[4] *Not your keys, not your coins.* URL: https://decrypt.co/resources/coins-keys-wallet.

- **Regolamentazione e Standard**: La mancanza di standard e regolamentazioni uniformi può complicare l'adozione su larga scala e la sicurezza dell'identità digitale.

3.4.1 *SSI* digital *Wallet*

Il wallet, di solito, è una applicazione *cloud*, o per dispositivi mobili, che svolge un ruolo fondamentale nel consentire agli utenti di gestire le proprie identità digitali in modo sicuro e conveniente.
Pertanto, essendo il *Wallet* lo strumento per autoregolare la nostra identità nel mondo digitale, è anche lo strumento che ci permette di interagire digitalmente sia con le aziende e i servizi privati, sia con le pubbliche amministrazioni. Ciò significa che *issuer*, *holder* e *verifier* dell'ecosistema *SSI* utilizzano il *wallet* digitale per interagire.
Le funzionalità essenziali sono:

- **Conservazione e gestione delle credenziali verificabili**
  Il *Wallet* Digitale *SSI* è il luogo in cui archivi tutte le tue *Verifiable Credentials (VC)*. Queste *VC* rappresentano le tue prove di identità, competenze e attributi
- **Condivisione Sicura**;
  Quando è necessario condividere informazioni sulle tue identità digitali, il *wallet SSI* ti permette di selezionare quali informazioni condividere e con chi, garantendo la privacy e la protezione dei tuoi dati;
- **Autenticazione**
  Il *wallet* Digitale *SSI* offre metodi di autenticazione, tra cui la possibilità di utilizzare il riconoscimento facciale, le impronte digitali o altre misure biometriche.

3.4.2 *Verifiable Credentials (VC)*

Nel mondo fisico, una credenziale potrebbe essere costituita da:

- Informazioni relative all'identificazione del soggetto della credenziale (ad esempio, una foto, un nome o un numero di identificazione);

- Informazioni relative all'autorità che l'ha rilasciata (ad esempio, un'amministrazione comunale, un'agenzia nazionale o un ente di certificazione);
- Informazioni relative al tipo di credenziale (ad esempio, un passaporto olandese, una patente di guida americana o una tessera di assicurazione sanitaria);
- Informazioni relative a specifici attributi o proprietà che l'autorità emittente fa valere sul soggetto (ad esempio, la nazionalità, le classi di veicoli che si possono guidare o la data di nascita);
- Prove relative al modo in cui la credenziale è stata ottenuta;
- Informazioni relative ai vincoli sulla credenziale (ad esempio, la data di scadenza o i termini di utilizzo).

Una *Verifiable Credentials (VC)* può rappresentare tutte le stesse informazioni che rappresenta una credenziale fisica.

Ecosistema

Questa sezione descrive i ruoli degli attori principali e le relazioni tra loro. Un ruolo è un'astrazione che può essere implementata in molti modi diversi e può variare in base al caso d'uso. Un esempio di ecosistema lo si trova in figura 3.1

- **Holder**:

  è un'entità che possiede una o più *Verifiable Credentials (VC)* e può generare una o più *Verifiable Presentation (VP)* da esse. Esempi di *holder* sono studenti, dipendenti e clienti;

- **Issuer**:

  è un'entità che crea affermazioni (*claims*) su uno o più soggetti (*subject*), creando una *VC* da tali affermazioni e trasmettendo la credenziale ad un *holder*. Esempi di *issuer* sono le aziende, le organizzazioni non profit, le associazioni di categoria, i governi e gli individui;

- **Subject**:

  Un'entità sulla quale vengono fatte affermazioni. Tra i soggetti esemplificativi

vi sono esseri umani, animali e cose. In molti casi il titolare di una credenziale verificabile è il soggetto, ma in alcuni casi non è così. Ad esempio, un genitore (*l'holder*) potrebbe detenere le *VC* di un figlio (il *subject*);

- **Verifier**:
  è un'entità che riceve una o più credenziali verificabili, facoltativamente all'interno di una presentazione verificabile, per l'elaborazione. Esempi di verificatori sono i datori di lavoro, il personale di sicurezza e i siti web;

- **Verifiable data registry**: Un ruolo che un sistema potrebbe svolgere mediando la creazione e la verifica di identificatori, chiavi e altri dati rilevanti, come schemi di *VC*, registri di revoca, chiavi pubbliche dell'*issuer* e così via, che potrebbero essere richiesti per utilizzare le *VC*.

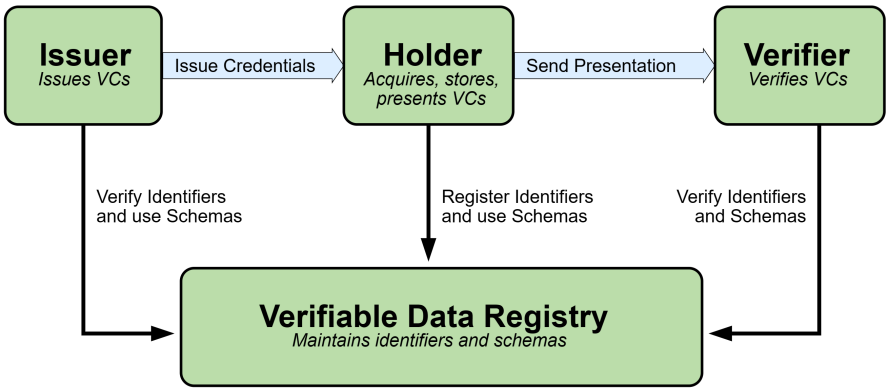


Figura 3.1: I ruoli e il flow di informazioni che costituiscono l'intero ecosistema[5]

3.4.3 DID

Questa specifica sulle identità decentralizzate introduce un nuovo tipo di identificatore unico a livello globale. Molti dei nostri identificatori attuali, come numeri di telefono o codici fiscali, sono emessi da autorità esterne e non sono sotto il nostro controllo. Al contrario, i *DID* permettono a individui e organizzazioni di generare i propri identificatori utilizzando sistemi di cui si fidano, consentendo loro di dimostrare il

[5]https://www.w3.org/TR/vc-data-model

controllo su di essi attraverso la crittografia. Gli identificatori decentralizzati possono essere personalizzati per diversi contesti e supportano le interazioni con altre entità senza dipendere da un'autorità centrale. Questa specifica non presuppone alcuna tecnologia specifica, ma consente agli implementatori di utilizzare diversi sistemi di identificazione, creando un ponte di interoperabilità tra identificatori centralizzati, federati e decentralizzati.

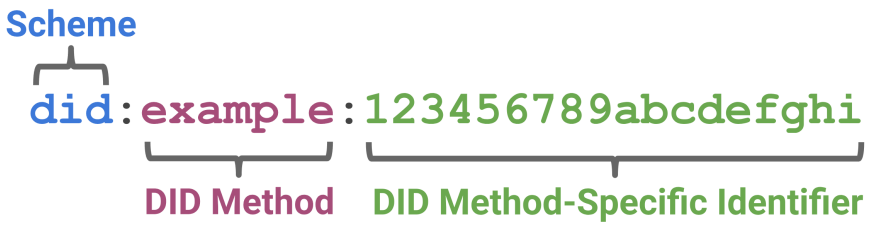


Figura 3.2: Un semplice esempio di identificatore decentralizzato (*DID*)[6]

Una *DID* è uno schema ben preciso ed è formato da 3 parti, quali:

- **l'identificatore dello schema *Uniform Resource Identifier (URI)* *did***: si chiama *Scheme* ed non è nient'altro che il prefisso "did:";
- **l'identificatore del metodo *DID***:
  viene definito come *DID Method*, è molto importante perché identifica su quale registro decentralizzato o *blockchain* tale *DID* è stato risolto. Per esempio, se viene risolto sulla *blockchain* di Bitcoin sarà così espresso "did:btcr";
- **l'identificatore specifico del metodo *DID***:
  viene definita come *Method Specific-Identifier* ed è l'estensione che caratterizza il *DID* per la specifica risorsa (il mio *DID*, il tuo *DID*).

Un esempio per un *DID* per Bitcoin potrebbe essere il seguente:

did:btcr:13n29cn30cn3ncr84dc.

Ogni *DID* viene registrato in modo autonomo, direttamente dal proprietario all'interno di *blockchain permission less e/o permissioned*. Non ci sono intermediari, ed è

[6]https://www.w3.org/TR/did-core

per questo che viene definita *Self-Sovereign Identity*. Ogni utente è completamente sovrano di se stesso e della propria identità.

L'esempio di un *DID* come in figura 3.2 si risolve in un *Decentralized Identifier Document (DID Document)*. Questo *DID Document* comprende informazioni relative al *DID* stesso, inclusi i dettagli sulle modalità di autenticazione crittografica del proprietario del *DID*.

Architettura *DID*

Nella figura 3.3 viene fornita una panoramica di base dei principali componenti dell'architettura *DID*

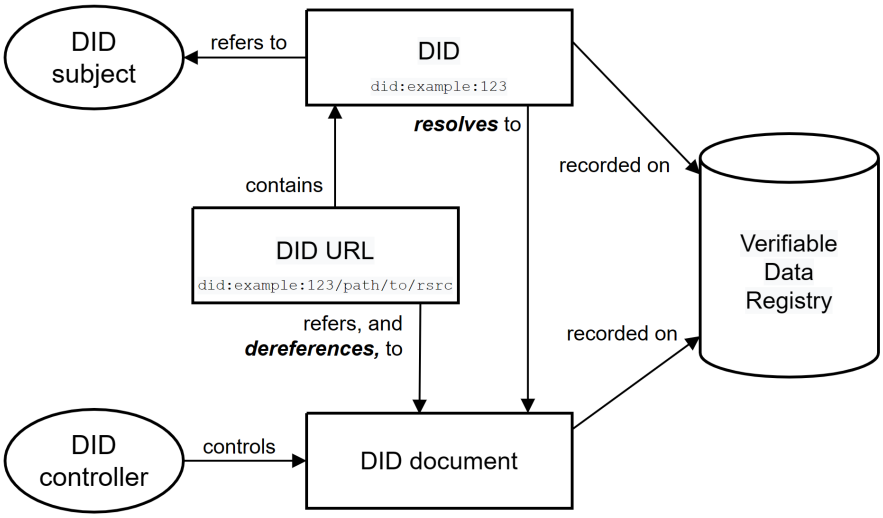


Figura 3.3: Panoramica dell'architettura DID e delle relazioni tra i componenti di base[7]

- **DID Subject:**

  Per definizione, è l'entità identificata dal *DID*. Il *DID Subject* può anche essere il *DID Controller*. Qualsiasi cosa può essere il *subject* di un *DID*: persona, gruppo, organizzazione, cosa o concetto.

- **DID URL:**

  estende la sintassi di un *DID* di base per incorporare altri componenti *URI* standard, come path, query e fragment, al fine di individuare una particolare

risorsa, ad esempio una chiave pubblica crittografica all'interno di un documento *DID* o una risorsa esterna al documento *DID*;

- **DID Document:**

  Si tratta di un documento accessibile attraverso un *Verifiable Data Registry (VDR)* e che contiene le informazioni associate a un determinato *DID*, come ad esempio il *wallet* (in questo contesto denominato *"repository"*) a cui è collegato e la chiave pubblica;

- **DID Controllers:**

  è l'entità (persona, organizzazione o software autonomo) che ha la capacità, definita da un *DID method*, di apportare modifiche a un *DID Document*. Questa capacità è tipicamente affermata dal controllo di un insieme di chiavi crittografiche utilizzate da un software che agisce per conto del controllore, sebbene possa essere affermata anche attraverso altri meccanismi. Si noti che un *DID* può avere più di un controllore e il *DID subject* può essere il *DID controller* o uno di essi;

- **Verifiable Data Registry (VDR):**

  È un sistema generico in grado di gestire la registrazione dei *DID* e di fornire le informazioni necessarie per creare il *DID Document*, indipendentemente dalla tecnologia utilizzata (ad esempio, *database* o *blockchain*);

- **DID Methods:**

  rappresentano il processo tramite il quale un tipo specifico di *DID* e il suo relativo *DID Document* vengono generati, recuperati, aggiornati e disabilitati. Questi metodi *DID* sono descritti mediante specifiche distinte dei metodi *DID*;

- **DID Resolvers and DID Resolution**

  è un componente del sistema che riceve in ingresso un *DID* e produce in uscita un *DID Document* conforme. Questo processo è chiamato risoluzione *DID*. Le fasi di risoluzione di un tipo specifico di *DID* sono definite dalla specifica del *DID method*.

[7] https://www.w3.org/TR/did-core

3.5 Hyperledger Foundation

Nel 2015, la *Linux Foundation* ha riconosciuto l'importanza della *blockchain* e ha deciso di promuoverne lo sviluppo e la diffusione in molti settori. Ha lanciato *Hyperledger* come un progetto avviato da 21 membri fondatori.
Sei anni dopo, *Hyperledger* è diventato *Hyperledger Foundation* ed è sede di 18 progetti con più di 75 tecnologie in fase di sviluppo negli *Hyperledger Labs*. Oggi, più di 350 aziende hanno contribuito al codice dei progetti di *Hyperledger*. La comunità di sviluppatori di *Hyperledger* concepisce e costruisce questi progetti come software di livello enterprise, resi liberamente disponibili per venditori, fornitori di servizi, start-up, enti accademici e altri, che possono sfruttarli per costruire e implementare reti *blockchain* anche per soluzioni commerciali[8].

Figura 3.4: Logo di Hyperledger Foundation. Immagine scaricata dalla pagina GitHub ufficiale[9]

Hyperledger Foundation ha i seguenti goal

- La creazione di *framework ledger open source* distribuiti di livello aziendale a supporto delle transazioni commerciali;
- Fornire un'infrastruttura neutrale, aperta e guidata dalla comunità, supportata da una governance tecnica e aziendale;
- Costruire comunità tecniche per sviluppare PoC, casi d'uso, percorsi sul campo e implementazioni di *blockchain* e *shared ledger*;
- Educare il pubblico sulle opportunità di mercato della tecnologia *blockchain*.

Hyperledger promuove la Self-sovereign identity (SSI), attraverso tre strutture principali basate su blockchain:

[8] *Hyperledger Foundation*. URL: https://www.linuxfoundation.org/resources/case-studies/hyperledger.
[9] https://github.com/hyperledger

- **Hyperledger Indy** Hyperledger Indy è stato il primo framework blockchain incentrato sull'identità all'interno della fondazione. Offre strumenti, librerie e componenti riutilizzabili per la creazione di identità digitali decentralizzate. Indy è progettato per essere interoperabile con altre blockchain oppure può essere utilizzato autonomamente;

Figura 3.5: Logo di Hyperledger Indy. Immagine scaricata dalla pagina GitHub ufficiale[10]

- **Hyperledger Ursa** Hyperledger Ursa rappresenta una libreria crittografica condivisa, destinata a essere impiegata in diversi progetti all'interno dell'ecosistema Hyperledger, nonché in qualsiasi altro software che richieda una robusta base crittografica. Questa libreria costituisce un repository opzionale dove trovare e utilizzare gli strumenti crittografici necessari;

Figura 3.6: Logo di Hyperledger Ursa. Immagine scaricata dalla pagina GitHub ufficiale[11]

- **Hyperledger Aries** Hyperledger Aries costituisce un set di strumenti concepito per iniziative e soluzioni focalizzate sulla creazione, trasmissione, archiviazione

[10] https://github.com/hyperledger/indy-node
[11] https://github.com/hyperledger-archives/ursa

e utilizzo di credenziali verificabili. Si tratta di un'infrastruttura che facilita interazioni peer-to-peer basate sulla tecnologia blockchain.

Figura 3.7: Logo di Hyperledger Aries. Immagine scaricata dalla pagina GitHub ufficiale[12]

3.6 Hyperledger Aries Javascript

Hyperledger Aries è un software sviluppato dalla *Hyperledger Foundation* che si concentra sulla gestione delle Verifiable Credentials (VC), occupandosi della loro creazione, trasmissione, archiviazione e utilizzo sicuro in un contesto decentralizzato.
Hyperledger Aries JavaScript rappresenta un framework *open source* che mette a disposizione le funzionalità offerte da *Hyperledger Aries* attraverso *TypeScript*. Questo strumento permette l'integrazione delle capacità di *Aries* in progetti sviluppati in *TypeScript*, offrendo agli sviluppatori un'accessibilità agevole e flessibile per la gestione delle identità e delle transazioni sicure basate sulla *blockchain*.
Il progetto nasce principalmente per rendere accessibile lo sviluppo della Self-sovereign identity (SSI) rendendolo adattabile alle esigenze web based.
Aries JS si concentra sulla creazione di un canale crittografato tra *Issuer* ed *Holder* per scambiare Verifiable Credentials (VC). Questo avviene grazie a tre componenti principali:

- **Agent**: Agisce come intermediario e possiede chiavi crittografiche per l'autorizzazione delegata, interagendo attraverso protocolli *DIDComm*;
- **DIDComm**: Un protocollo che garantisce comunicazioni sicure e private tra *Issuer* ed *Holder*, utilizzando un codice alfanumerico (DID) risolto per ottenere informazioni aggiuntive;

[12] https://github.com/hyperledger/aries

- **Wallet**: Strumento installato sui dispositivi di *Issuer* ed *Holder* per la gestione delle identità digitali e delle interazioni con altre entità. Nel mio caso ho avuto a che fare con due wallet differenti: *Indy* (deprecato) e *Aries Askar*.

Per stabilire un canale crittografato, *Issuer* e *Holder* devono configurare *wallet* e *agent*, inviare e accettare un invito di collegamento, e infine confermare il collegamento. Le credenziali vengono emesse dall'*Issuer* e raccolte dall'*Holder*. L'*Issuer* definisce la credenziale e l'emette, mentre l'*Holder* la riceve e conferma l'emittente specificato nella definizione della credenziale. Questo processo garantisce un triangolo di fiducia e consente lo scambio sicuro di documenti e credenziali.

3.7 API RESTful

Un'*Application Programming Interface Representational State Transfer (API REST)*, comunemente chiamata anche *API REST*, è un'interfaccia di programmazione delle applicazioni (*API* o *API web*) che segue i principi dell'architettura *REST* e permette l'interazione con servizi web che rispettano questi principi.

REST è un insieme di vincoli architetturali, non un protocollo né uno standard e può essere implementato in diversi modi. Quando una richiesta *client* viene inviata tramite un'*API RESTful*, questa trasferisce al richiedente o all'*endpoint* uno stato rappresentativo della risorsa. L'informazione, o rappresentazione, viene consegnata in uno dei diversi formati tramite *HTTP*: *Javascript Object Notation (JSON)*, *HTML, XLT, Python, PHP* o testo semplice. Il formato *JSON* è uno dei linguaggi di programmazione più diffusi, perché, a dispetto del nome, è indipendente dal linguaggio e facilmente leggibile da persone e macchine.

Affinché un'*API* sia considerata *RESTful*, deve rispettare i criteri indicati di seguito:

- Un'architettura *client-server* composta da *client*, *server* e risorse, con richieste gestite tramite *HTTP*;

- Una comunicazione *client-server stateless*, che quindi non prevede la memorizzazione delle informazioni del *client* tra le richieste *Get*; ogni richiesta è distinta e non connessa;
- Dati memorizzabili nella *cache* che ottimizzano le interazioni *client-server*;
- Un'interfaccia uniforme per i componenti, in modo che le informazioni vengano trasferite in una forma standard. Ciò impone che:
  - le risorse richieste siano identificabili e separate dalle rappresentazioni inviate al *client*;
  - le risorse possano essere manipolate dal *client* tramite la rappresentazione che ricevono poiché questa contiene le informazioni sufficienti alla manipolazione;
  - i messaggi autodescrittivi restituiti a un *client* contengano le informazioni necessarie per descrivere come il *client* deve elaborare l'informazione;
  - le informazioni siano ipermediali e ipertestuali, ovvero accedendo alla risorsa il *client* deve poter individuare, attraverso *hyperlink*, tutte le altre azioni disponibili al momento.
- Un sistema su più livelli che organizza ogni tipo di *server* (ad esempio quelli responsabili della sicurezza, del bilanciamento del carico, ecc.) che si occupa di recuperare le informazioni richieste in gerarchie, invisibile al *client*;
- Codice *on demand* (facoltativo): la capacità di inviare codice eseguibile dal *server* al *client* quando richiesto, estendendo la funzionalità del *client*.

Sebbene l'*API REST* debba essere conforme a questi criteri solitamente vengono applicate quando necessario, il che rende le *API REST* più rapide, leggere e con una maggiore scalabilità, ottimali quindi per *l'Internet of Things (IoT)* e lo sviluppo di app mobili.

3.8 Le 7 leggi dell'identità

Kim Cameron, noto per il suo lavoro nel campo della sicurezza informatica e delle identità digitali, ha formulato le *"7 Leggi dell'Identità"* (*The Laws of Identity*)[13] per affrontare questioni legate alla privacy, alla sicurezza e alla gestione delle identità digitali. Ecco un riassunto delle Sette Leggi dell'Identità di *Kim Cameron*:

- **User Control and Consent**: Il successo del sistema di identità digitale dipende dalla fiducia dell'utente, che può essere ottenuta attraverso la convenienza, la semplicità e la possibilità di controllare l'uso delle proprie informazioni. Il sistema deve proteggere l'utente da frodi e verificare l'identità delle parti coinvolte. È importante informare l'utente sulla raccolta di dati e consentire di scegliere quali informazioni rilasciare. Quando l'utente sceglie un *Identity Provider (IdP)*, deve essere informato sul tracciamento online. Il controllo dell'utente deve essere mantenuto, sia nel contesto aziendale che in quello dei consumatori, anche se ciò potrebbe influire sulle condizioni di lavoro delle aziende. La legge sul controllo e sul consenso dell'utente permette l'utilizzo di meccanismi che memorizzano le decisioni dell'utente e possono essere applicate automaticamente in futuro;

- **Minimal Disclosure for a Constrained Use**: Dobbiamo costruire sistemi che utilizzano le informazioni identificative considerando sempre la possibilità di una violazione, che rappresenta un rischio. Per ridurre questo rischio, è meglio condividere e conservare solo le informazioni necessarie. Seguendo queste pratiche, possiamo garantire un danno minimo in caso di violazione. Allo stesso tempo, il valore delle informazioni identificative diminuisce all'aumentare della loro quantità. Un sistema costruito con il principio del minimalismo delle informazioni è quindi un target meno attraente per i furti di identità, riducendo così il rischio. Limitando l'utilizzo delle informazioni ad uno scenario specifico e aderendo ad una politica di controllo, l'efficacia del principio del "bisogno di sapere" nel ridurre il rischio viene ulteriormente amplificata. Inoltre, il concetto di "minimo di informazioni identificative" significa non solo il minor numero di

[13] Cameron, «The Laws of Identity».

richieste, ma anche evitare informazioni che potrebbero identificare un individuo in più contesti. Ridurre al minimo l'aggregazione delle informazioni identificative è anche importante per minimizzare il rischio;

- **Justifiable Parties**: Il sistema d'identità deve mettere a conoscenza l'utente del soggetto o dei soggetti con cui interagisce e che condivide le informazioni. È quindi importante che il sistema di identità sia prevedibile, trasparente e comprenda meccanismi per stabilire politiche di utilizzo delle informazioni;

- **Directed Identity**: Le entità pubbliche hanno identificatori noti e invarianti, come gli *URL* dei siti web, che possono essere considerati fari che emettono identità a chiunque si presenti. Al contrario, un visitatore di un sito web può stabilire una relazione di identità unidirezionale con quel sito, selezionando un identificatore da utilizzare solo con quel sito. Questa relazione unidirezionale non può essere condivisa con altri siti;

- **Pluralism of Operators and Technologies**: Non esiste un unico modo per esprimere l'identità, poiché i contesti in cui l'identità viene richiesta possono variare. Ad esempio, un'identità digitale rilasciata dal governo può essere utilizzata per i servizi governativi, ma non sarebbe adatta per i datori di lavoro o per la navigazione su internet. Ciò implica che abbiamo bisogno di diversi sistemi di identità, che offrano caratteristiche diverse. Tuttavia, questi sistemi devono esistere all'interno di un metasistema che permetta l'interazione tra di loro. Un metasistema di identità universale non deve essere un altro monolite, ma deve essere policentrico e polimorfo, consentendo all'identità di emergere, evolversi e auto-organizzarsi. È necessario un protocollo incapsulante per il trasporto delle informazioni e un'esperienza utente unificata per selezionare i fornitori di identità e le caratteristiche appropriate. In questo modo, l'identità può essere espressa in un metasistema, consentendo la diversità dei contenuti ma una rappresentazione comune;

- **Human Integration**: Il test degli utenti è un modo per valutare se i sistemi di identità digitale interagiscono in modo efficace con gli utenti. Attualmente,

l'identità digitale viene trasmessa sotto forma di certificati, ma questi non sono significativi per gli utenti. È necessario sviluppare un sistema di identità universale che definisca l'utente umano come parte integrante del sistema distribuito e offra meccanismi di comunicazione uomo-macchina non ambigui. Inoltre, è necessario garantire la sicurezza del canale tra il display del browser e il cervello dell'utente, al fine di proteggerlo da attacchi di *phishing* e frodi. L'obiettivo è creare un sistema di identità che sia sicuro su tutte le piattaforme e che offra un'esperienza utente prevedibile e non ambigua;

- **Consistent Experience Across Contexts**: un futuro in cui ci saranno diverse identità contestuali tra cui scegliere fra un'identità di navigazione, personale, comunitaria, professionale, basata sulla carta di credito e cittadinanza. Per rendere possibile questo, le identità digitali devono essere trasformate in "cose" che l'utente può gestire e condividere. Le identità digitali vengono specificate dai servizi Web e visualizzate all'utente, che può scegliere quale utilizzare in base alle informazioni richieste. Alcune parti possono richiedere più di un tipo di identità e gli utenti devono poter selezionare l'identità più adatta per il contesto in cui si trovano. L'interazione tra le parti deve essere sicura e l'esperienza utente deve essere chiara e coerente, prevenendo l'ambiguità nel consenso e nella comprensione delle informazioni sull'identità. Gli utenti vogliono vedere le diverse identità come parte di un mondo integrato, ma che rispetti la separazione dei contesti.

Le Sette Leggi dell'Identità di *Kim Cameron* forniscono un quadro concettuale per la progettazione di sistemi di gestione delle identità che mettano al centro il controllo, la privacy e la sicurezza degli utenti. Sono state un importante contributo al campo dell'identità digitale e delle politiche di gestione delle identità.

Capitolo 4

Esperimenti

In questa sezione vengono spiegati i vari esperimenti e tecnologie utilizzate per il tentativo di sviluppo del Proof of Concept (PoC)

4.1 Tecnologie e strumenti

- **Hyperledger Aries Javascript**: framework per lo sviluppo di una infrastruttura per identità digitali decentralizzate, spesso chiamate SSI;
- **TypeScript**: linguaggio di programmazione *open source* sviluppato da *Microsoft*. Si tratta di un'estensione di *JavaScript*;
- **Node.JS**: è un *runtime system open source* multipiattaforma orientato agli eventi per l'esecuzione di codice *JavaScript*;
- **Visual Studio Code**: Editor di codice;
- **Macchina Virtuale di Athesys s.r.l** (*Debian*): da cui si accede in *Secure SHell (SSH)* tramite *Virtual Private Network (VPN)*;
- **Express**: Per la creazione dell'API REST;
- **Nodemon**: tool che permette il riavvio automatico di un *server Express*;
- **Postman**: per il test delle chiamate *HTTP*.

4.2 Analisi Architetturale

La *Decentralized Applications (DApp)* da sviluppare aveva l'obiettivo di sperimentare e illustrare l'impiego potenziale del modello *SSI* e delle sue funzionalità mediante l'uso del framework *Hyperledger Aries JavaScript*.
La piattaforma doveva consistere in un'interfaccia utente (*frontend*) sviluppata con il framework *Monokee IAM*, che, attraverso chiamate *HTTP* all'*API REST*, gestiva le seguenti funzionalità:

- Connessione tra l'*holder* e l'*issuer*;
- Creazione di *schema* personalizzati per nuove *Verifiable Credentials (VC)*;
- Assegnazione di tali *VC* basate su vari *schema* creati agli *holder*;
- Verifica delle *VC* assegnate.

In breve, l'*API REST* agiva come un agent *issuer* nell'architettura *SSI*.
Inoltre, si è lavorato per una possibile interoperabilità con un *wallet SSI* esterno, come richiesto da un requisito opzionale.

4.3 Backend

Per lo sviluppo del lato *server*, ho optato per *Node.js* come base fondamentale, che offre l'infrastruttura necessaria per eseguire codice *server-side*, e ho scelto *TypeScript* come linguaggio di programmazione. La decisione di utilizzare queste tecnologie è stata influenzata sia dalle necessità aziendali sia dal fatto che sono recenti e ben documentate.
Come punto di riferimento iniziale, mi sono ispirato alla demo disponibile sulla pagina *GitHub* ufficiale del framework *Aries Javascript*. Questa dimostrazione implementa il nuovo *wallet Aries Askar*, che ha preso il posto dell'ormai superato *Indy SDK*. *Indy SDK* è stato dichiarato obsoleto il 24 ottobre 2023[1], principalmente a causa di problemi legati alle sue dipendenze, poiché si basava su una versione datata di *Ubuntu*, esponendolo a gravi vulnerabilità.

[1] https://github.com/hyperledger/indy-sdk

La decisione di utilizzare il *wallet Aries Askar* per il nuovo progetto è stata influenzata dalle raccomandazioni fornite dalla documentazione del framework, che sottolineava la prossima obsolescenza del *wallet Indy*, rendendo quindi *Aries Askar* la scelta più adeguata e aggiornata.

Per la realizzazione del progetto, mi sono informato sulla demo disponibile nella pagina *GitHub* ufficiale degli sviluppatori. Inizialmente, la demo presentava alcune problematiche relative alle dipendenze di *Indy* su *Windows*. Per superare questo ostacolo, *Athesys s.r.l* mi ha fornito generosamente una macchina virtuale *Debian*, creando un ambiente adeguato per lo sviluppo e l'esecuzione del progetto.

Tuttavia, anche su *Debian* sono emerse difficoltà legate a *Indy*. Dopo aver consultato uno degli sviluppatori della libreria *Aries* tramite una *issue* su *GitHub*, è emerso che il problema non era attribuibile all'ambiente operativo, bensì a un difetto della demo stessa e ai file di configurazione di *Node.js*.

Nello specifico, il problema era collegato alla libreria "*ref-napi*", che, come indicato da uno degli sviluppatori, aveva un processo di elaborazione troppo lento, portando a timeout nel programma[2].

Inserendo nel file *package.json* questo codice si è andato a risolvere il problema di computazione lenta.

```
{
  "overrides": {
    "ref-napi": "npm:@2060.io/ref-napi"
  }
}
```

Oltre alla problematica legata a "*ref-napi*", il *wallet Aries Askar* si basa sulla versione *18.04* di *Ubuntu*, ormai un po' datata, e richiede la versione *18* o superiore di *NodeJS*. Questo aspetto riduce la compatibilità del framework e aumenta il rischio di problemi futuri.

[2]https://github.com/hyperledger/aries-framework-javascript/issues/1520#issuecomment-1652010793

Il *backend* sviluppato prevede due tipologie di servizi: *Servizio Holder* e un servizio combinato di *Issuer* e *Verifier*. Per agevolare e velocizzare il testing, ho deciso di unificare *Issuer* e *Verifier* in un unico servizio.

4.3.1 Servizio Holder

Il servizio *Holder* è molto simile a quello nella demo del framework *Aries*.
Quello che può fare è:

- Richiedere un link di invito (che viene generato dal servizio *Issuer*) per instaurare la connessione fra i due *agent*;
- Accettare/Rifiutare una *VC*;
- Accettare/Rifiutare un *proof*

4.3.2 Servizio *Issuer-Verifier*

Il servizio *Issuer-Verifier* serve per:

- Generare il link di invito (da consegnare all'*holder*) per instaurare una connessione fra i due *agent*;
- Registrare uno *schema* per l'emissione di VC;
- Emettere una *VC* ad un *holder*;
- Richiedere un *proof* ad un *holder* (*verifier*).

Riguardo all'ultimo aspetto, quando un *holder* non è in grado di fornire il *proof* richiesto (cioè non possiede le *VC* necessarie per sostenere il *proof*), il sistema dell'*holder* va in *crash* e non invia alcuna risposta all'*issuer*, che rimane in attesa indefinitamente. Per questa problematica specifica, non ho potuto trovare una soluzione, principalmente perché il problema è emerso solo verso la fine del mio tirocinio, lasciandomi senza tempo sufficiente per condurre un'indagine approfondita.

Problemi Riscontrati e accorgimenti

- Durante lo studio e la programmazione del *backend* ci sono stati molti problemi e rallentamenti, accentuati sopratutto per la quasi mancanza e poco aggiornata documentazione del framework *Aries Javascript*;
- Il costante problema legato alle dipendenze e una demo poco funzionante hanno notevolmente rallentato il progresso nello sviluppo del *backend*;
- La transizione da *Indy SDK* a *Aries Askar* ha generato numerosi problemi e difficoltà, aiutati dalla quasi assenza di casi d'uso di *Askar*;
- Le molte problematiche legate alle rigide versioni dei sistemi operativi e delle tecnologie potrebbero portare *Askar* verso un futuro simile a quello di *Indy*, cioè rischiare di essere abbandonato e considerato obsoleto. Di conseguenza, per uno sviluppatore ciò significherebbe più lavoro e un significativo spreco di tempo;
- Il *crash* improvviso dell'*holder* quando viene richiesto un *proof* che non soddisfa.

4.4 API RESTful con integrazione backend

Per creare l'*API REST*, ho utilizzato il framework *Express* per la gestione delle varie chiamate *Hypertext Transfer Protocol (HTTP)* in connessione con il framework *Aries JavaScript*. Su suggerimento dell'azienda, ho scelto di adottare direttamente *Express*, in quanto è già utilizzato internamente e inoltre è un framework ampiamente documentato e rinomato per la sua facilità d'uso.
Questa *API REST* opera come *issuer*, ovvero l'entità che rilascia credenziali verificabili agli *holder*.
Le principali chiamate gestite dall'*API REST* includono:

- `"baseURL/invitationLink"` (*GET*): Utilizzato per generare il link di invito, necessario all'*issuer* per instaurare una comunicazione con l'*holder*;
- `"baseURL/registerSchema"` (*POST*): Impiegato per registrare uno *schema* di una nuova *VC*, richiede come input una stringa *JSON* come il seguente esempio:

```
{
    "name": "Identity card",
    "version": "1.0.0",
    "attrNames": ["name", "surname", "date"]
}
```

- "**name**": indica il nome dello *schema*. In questo contesto, stiamo creando uno *schema* chiamato "*Identity Card*", il quale rappresenta una forma di carta d'identità di base;
- "**version**": rappresenta la versione dello *schema*. Nel corso del tempo, potrebbe presentarsi la necessità di apportare modifiche o aggiornamenti a una *VC*. Piuttosto che modificare il "*name*" dello schema, è sufficiente cambiare la versione e apportare le varie modifiche necessarie per mantenere invariato il nome della credenziale;
- "**attrNames**": elenca gli attributi inclusi nello *schema*, che possono essere utilizzati come metodo di verifica. In questo caso, abbiamo inclusi il nome, il cognome e la data di nascita; Dopo aver consegnato la credenziale verificabile all'*holder*, questi attributi possono essere richiesti da un *verifier* per effettuare una verifica tramite Verifiable Presentation (VP). Gli attributi dello *schema* possono essere aggiunti, aggiornati o modificati. In ogni caso, sarà necessario creare un nuovo *schema* aggiornando la versione.

Gli attributi "*name*" e "*version*" devono essere univoci, il che implica che se si tenta di creare un nuovo *schema* utilizzando la funzione "*/registerSchema*", la chiamata *HTTP* restituirà un errore indicando che esiste già uno *schema* con lo stesso nome e versione dichiarati. Questa scelta è stata fatta a livello progettuale per comodità d'uso, nonostante il sistema differenzi gli *schema* non per nome ma tramite un identificatore univoco (id).

- `"baseURL/credential"` (*POST*): usato per assegnare le credenziali ad un *holder*. Come input bisogna passare una stringa *JSON* come questa:

```
{
    "name": "Identity card",
    "version": "1.0.0",
    "attributes":{
        "name": "Federico",
        "surname": "Franchini",
        "date": "10/11/1990",
    }
}
```

Gli attributi "*name*" e "*version*" sono essenziali per la ricerca e l'assegnazione della credenziale verificabile all'*holder*. In "*attribute*" si va ad assegnare ai vari attributi i parametri corrispondenti alla credenziale come nell'esempio precedente. Per poter assegnare una credenziale verificabile ad un *holder* prima di tutto bisognerà avviare una connessione con esso passandogli il link di invito usando l'*endpoint* "*/invitationLink*", poi si potrà assegnarli la credenziale chiamando questo *endpoint*.

- `"baseURL/proof"` (*POST*): usato per far richiesta di *proof* su una credenziale. Come input bisogna passare una stringa *JSON* come questa:

```
{
    "proof-1":{
        "name": "Identity card",
        "version": "1.0.0",
                "attrNames": ["name", "surname", "date"]
    },
    "proof-2":{
        "name": "Driver's licence",
        "version": "2.0.0",
                "attrNames": ["name", "surname", "date", "type"]
    }
}
```

In questo modo andremo a richiedere all'*holder* 2 tipi di VC da verificare;

4.5 Test sui vari *Wallet SSI*

Durante lo sviluppo del *backend*, ho condotto delle ricerche sui *wallet SSI* disponibili come applicazioni Android e il loro funzionamento. Sono emersi due potenziali candidati:

- Lissi wallet[3];
- Orbit Edge[4].

Questi *wallet* operano come "*holder*", gestendo tutte le Verifiable Credentials dell'utente attraverso un'interfaccia intuitiva e semplice, basta scannerizzare il *QRCode* che mette in contatto il *wallet* (l'*holder*) con l'*issuer* (il creatore del *QRCode*).
Prima di adattare il *backend* per la comunicazione con i *wallet*, ho ricevuto da un membro di Athesys s.r.l una demo trovata su internet da un'azienda chiamata *Animo*[5].
Questa demo è implementata utilizzando il framework *Aries JavaScript* e sembra

[3] *Lissi wallet.* URL: https://www.lissi.id.
[4] *Orbit Edge.* URL: https://northernblock.io/orbit-edge-wallet/.
[5] *Demo credenziali verificabili Animo.* URL: https://demo.animo.id.

funzioni con i vari *wallet Lissi* e *Orbit*. Tuttavia, nonostante il funzionamento della demo fosse sporadico e spesso instabile, ho iniziato i test di comunicazione tra il *backend* e i diversi *wallet SSI*. Durante i test, ho notato che c'erano delle incompatibilità. Leggendo dentro al *QRCode* della demo di *Animo* ho notato che l'invito al collegamnto fra *issuer* e *holder* era diverso dal link del mio *backend*. In poche parole i due *wallet SSI* richiedevano il protocollo dell'invito al collegamento cosi:

```
https://didcomm.org/connections/1.0/invitation...
```

Mentre Il link di invito del *backend* era così:

```
https://didcomm.org/out-of-band/1.1/invitation...
```

Oltre a questo analizzando l'implementazione della demo da parte di *Animo*, ho notato che utilizzava il vecchio *wallet Indy* anziché il nuovo *Askar*. Quindi la demo di *Animo* usava il protocollo di invito "*connections/1.0/invitation*", mentre il *backend* quello "*out-of-band/1.1/invitation*". A questo punto, ho comunicato tutto al mio tutor esterno e abbiamo convenuto di non procedere in questa direzione.

Problemi Riscontrati e accorgimenti

- Pochi *wallet SSI* disponibili, e quelli presenti non sono aggiornati e mantenuti;
- Per nulla chiara e senza documentazione come deve essere creato il *QRCode* che il *wallet SSI* deve leggere per accettare la connessione con l'*issuer*.

4.6 API RESTful di Aries

All'interno del *framework Aries* esiste un'*API REST* simile a quella che avrei dovuto sviluppare. Tuttavia, questa *API REST* non è attualmente aggiornata e continua a utilizzare il vecchio *wallet Indy*. Inoltre nella pagina della documentazione avvisano che l'*API REST* non è compatibile con l'ultima versione di "*@aries-framework/core*". Di conseguenza, è stata prontamente esclusa come opzione decisionale.

4.7 Monokee IAM

Per testare al meglio l'*API REST* ho creato un semplice *frontend* usando il framework *monokee IAM*. Questo *frontend* agisce come *agent issuer* e coordina le diverse interazioni con gli *holder*.

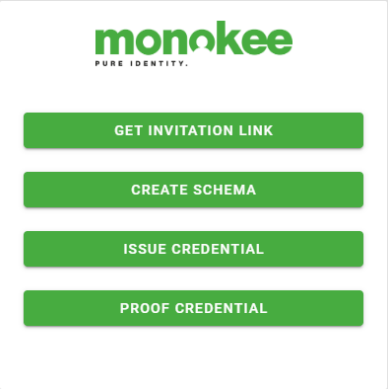


Figura 4.1: Interfaccia principale del *frontend*

Nella figura 4.1 viene rappresentato il workflow dall'alto verso il basso del frontend.

- **Get invitation link**

  Questa sezione genera il link di invito tramite una chiamata *HTTP* all'API REST, il quale sarà utilizzato dall'*holder* per instaurare una connessione con l'*issuer*. Tale link può essere richiesto sia in formato testuale che sotto forma di *QRCode*;

- **Create schema**

  Qui è possibile generare lo *schema* per una nuova VC tramite una chiamata *HTTP* all'*API REST*. Si richiede un input in formato *JSON*. Un esempio di *frontend* già precompilato lo si trova in in figura 4.2;

- **Issue credential** In questa sezione è possibile effettuare l'emissione della VC all'*holder* attraverso una chiamata *HTTP* all'*API REST*. Si richiede un input in formato *JSON*. Un esempio di *frontend* già precompilato lo si trova in in figura 4.3

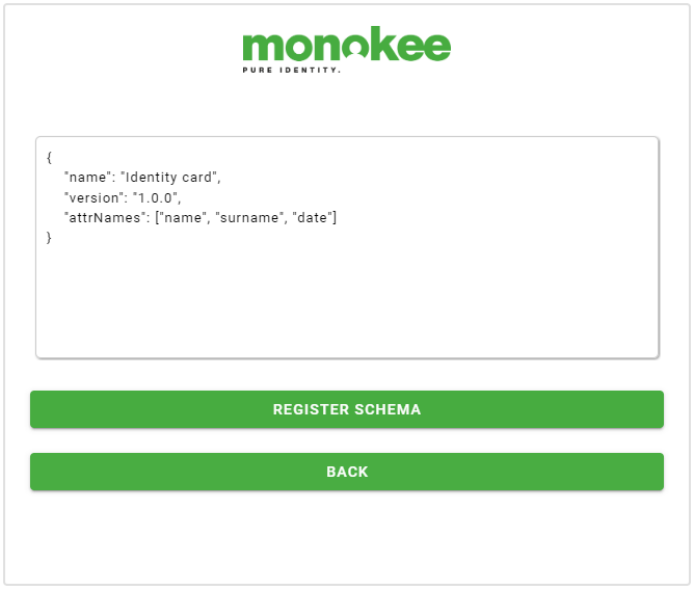


Figura 4.2: Interfaccia per la creazione dello *schema*

- **Proof credential** In questa sezione è possibile richiedere il *proof* all'*holder* di una VC attraverso una chiamata *HTTP* all'*API REST*. Si richiede un input in formato *JSON*. Un esempio di *frontend* già precompilato lo si trova in in figura 4.4

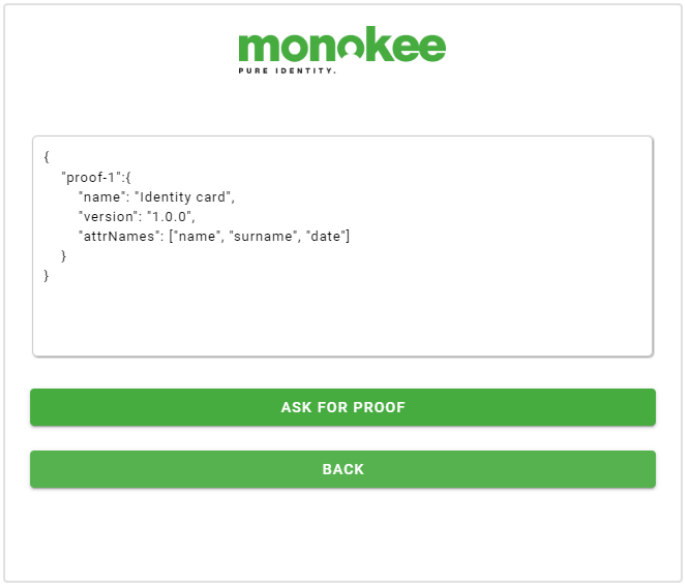


Figura 4.4: Interfaccia per la richiesta di un *proof*

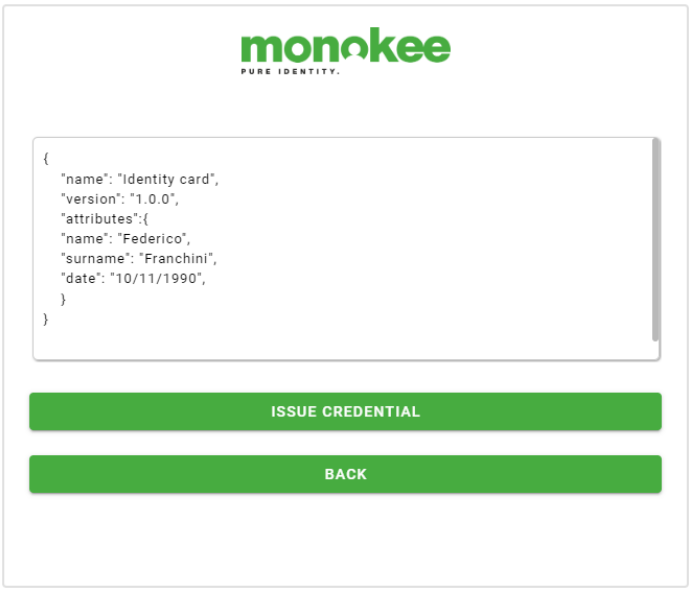


Figura 4.3: Interfaccia per l'issuing della credenziale

Nel *frontend*, ho implementato il supporto per i *callback* lanciati dal *backend*. Quando viene effettuato il rilascio di una VC, il *backend* invia un *callback* di accettazione che indica se la credenziale è stata accettata o meno. Tuttavia, c'è un problema nel *callback* di verifica: quando la verifica delle credenziali non ha esito positivo (quindi l'*holder* non possiede tali credenziali), anziché gestire il rifiuto, il *backend* termina inaspettatamente, provocando il *crash* e non lanciando il *callback*.

Capitolo 5

Verifica e validazione

Durante l'intero ciclo di sviluppo del progetto è stata posta particolare attenzione alla verifica delle attività al fine di individuare e correggere eventuali errori. Questo processo di verifica è stato integrato in tutte le fasi del progetto, interagendo principalmente con Athesys s.r.l e il team di sviluppo.

5.1 Interazione costante con Athesys s.r.l

Nell'ambito dell'analisi e della progettazione del *Proof of Concept (PoC)*, si è mantenuta un'interazione continua con *Athesys*. Questa collaborazione è stata fondamentale per raccogliere i requisiti specifici e guidare la definizione della specifica tecnica. L'obiettivo era apprendere al meglio le tecnologie e le architetture dietro al framework *Hyperledger Aries* e la condivisione dei risultati ottenuti per una *knowledge base* comune.

5.2 Sviluppo del PoC e Test

Il *PoC* sviluppato non aggiunge nuove funzioni al framework *Hyperledger Aries JavaScript*, ma piuttosto funge da verifica del framework stesso e della sua efficienza operativa. Tuttavia, sono stati implementati alcuni test, in particolare per la parte delle *API REST*, utilizzando specifici strumenti per questo scopo.
Come ad esempio:

- **Postman**: per le chiamate HTTP all'*API REST*;
- ***QRCode* scanner**: per visualizzare il contenuto di un *QRCode*;
- **Cyberchef**: per la decriptazione di stringe *base64*.

Questi strumenti sono stati impiegati per garantire una verifica accurata delle funzionalità implementate, contribuendo così a consolidare la documentazione necessaria per la tesi e a fornire una base solida per la comprensione e l'analisi del progetto.

5.3 Scrittura documentazione

Durante l'analisi del framework *Aries*, ho redatto della documentazione dettagliata per arricchire la *knowledge base* aziendale, utilizzando il programma *Notion*. Questa documentazione non solo serve come riferimento per i futuri progetti, ma è stata anche una risorsa importante per la scrittura della tesi.

Capitolo 6

Discussione

Si discute quanto appreso

6.1 Rilevanza del Paradigma *SSI*

Il lavoro e lo studio svolto durante il periodo di stage presso *Athesys s.r.l* ha evidenziato la fondamentale importanza delle identità digitali nell'attuale era digitale. L'inefficienza dei tradizionali sistemi centralizzati ha portato all'analisi e all'implementazione di nuovi approcci, in particolare della *Self-sovereign identity (SSI)*, per migliorare la gestione delle identità digitali e della loro sicurezza. Questo approccio soddisfa due esigenze fondamentali degli utenti: la gestione completa delle proprie informazioni (*self-sovereign*) e la sicurezza di una struttura affidabile. *SSI* elimina efficacemente la divulgazione impropria di dati personali e elimina attori superflui nella dinamica tra utente e servizio. Di conseguenza, può influire positivamente sulla società apportando notevoli miglioramenti nella gestione delle identità.

6.2 Implementazione di Soluzioni *SSI* con *Hyperledger Aries Javascript*

L'esperienza diretta ha rivelato le complessità e le sfide nell'implementare soluzioni che utilizzano l'*SSI*, in particolare durante lo sviluppo del *PoC* basato sul *framework Hyperledger Aries Javascript*. Quest'ultimo si posiziona come il *framework* più avanzato nell'ambito degli *SSI*. Tuttavia, un ostacolo significativo è stata la scarsa e obsoleta documentazione, spesso priva di casi d'uso pratici e di conseguenza di scarso aiuto. Un altro problema è stato la recente obsolescenza del *wallet Indy* e l'introduzione del suo sostituto *Askar*, rallentando la curva di apprendimento del framework e influenzando negativamente lo sviluppo del *PoC*. Questi fattori hanno costituito i principali impedimenti per comprendere a fondo le funzionalità e le limitazioni del framework, rendendo arduo completare lo sviluppo di un sistema efficiente per lo scambio di *Verifiable Credentials (VC)*.

6.3 Opportunità di Apprendimento e Applicazione Pratica

Il periodo di tirocinio è stato un'esperienza educativa di grande valore, offrendo l'occasione di applicare in modo pratico concetti avanzati relativi alle identità digitali decentralizzate, alla tecnologia *blockchain* e alle *Verifiable Credentials (VC)*. Le sfide incontrate durante l'implementazione hanno arricchito ulteriormente questa esperienza formativa, stimolando non solo una riflessione critica sulla complessità delle tecnologie *SSI* e dei loro framework, ma anche incentivando lo sviluppo di nuove strategie di *problem-solving* e l'approfondimento di competenze tecniche.

6.4 Future Work

Sebbene *Hyperledger Aries* si distingua come un framework promettente, esistono alternative come *uPort*, *Sovrin* e altri, che potrebbero essere valide scelte per futu-

re implementazioni di *SSI*. La varietà di implementazioni *SSI*, tuttavia, introduce complessità, in particolare per quanto riguarda l'interoperabilità tra diverse piattaforme. Questa sfida richiede lo sviluppo di standard unificati, come sottolineato da *Kim Cameron*, che enfatizza la necessità di un *metasistema identitario* unificatore per semplificare le complessità delle diverse implementazioni e favorire un'accoppiamento flessibile dell'identità digitale.[1]

Allo stato attuale, la *blockchain*, spesso erroneamente associata solo alle *criptovalute*, è una tecnologia emergente che necessita di tempo per essere pienamente riconosciuta e considerata affidabile. Pertanto, è cruciale continuare la ricerca sia nel campo della *blockchain* che delle *SSI* per progredire nella gestione delle identità digitali.

In sintesi, il mio tirocinio ha rappresentato un'opportunità inestimabile di esplorare e mettere in pratica concetti avanzati legati alle identità digitali e alle *Verifiable Credentials (VC)*. Nonostante le sfide incontrate, questo ambito si conferma come uno dei più stimolanti e promettenti per il futuro dell'identità digitale e della gestione dei dati personali.

[1] Cameron, «The Laws of Identity».

Capitolo 7

Conclusioni

7.1 Identificazione del Contesto

L'importanza crescente delle identità digitali nell'era digitale è indiscutibile, specialmente in un contesto dove i sistemi tradizionali centralizzati mostrano crescenti inefficienze. L'ascesa delle *Self-sovereign identity (SSI)* rappresenta un cambiamento significativo, proponendo un modello più efficiente per la gestione delle identità digitali e delle *Verifiable Credentials (VC)*. Tuttavia, l'attuazione pratica di soluzioni basate su *SSI*, in particolare tramite l'uso del framework *Hyperledger Aries*, è emersa come una sfida complessa.

7.2 Contribuzione

La mia tesi ha sottolineato la rilevanza critica delle identità digitali nel contesto moderno, con un focus sulle soluzioni *SSI*. Il contributo principale è stato lo sviluppo parziale di un *PoC* per un sistema di scambio di *VC*, basato su *Hyperledger Aries Javascript*. Anche se incompleto, questo *PoC* costituisce una fondamentale base di partenza per futuri sviluppi nel campo delle *SSI*. Inoltre, la ricerca ha prodotto una documentazione dettagliata sui vari ostacoli trovati durante lo sviluppo e dei vari processi di implementazione, inclusi design decisionali e tecnologie impiegate, che può

servire come guida per ulteriori ricerche e implementazioni nel settore delle identità digitali decentralizzate.

7.3 Sfide e next steps

Affrontare le sfide tecniche e concettuali legate all'implementazione delle soluzioni *SSI* è stata una parte cruciale del lavoro. Le principali sfide riguardavano la comprensione e l'applicazione del framework *Hyperledger Aries*, la gestione di documentazione spesso carente e la risoluzione di problemi legati all'interoperabilità tra sistemi diversi. I prossimi passi dovrebbero concentrarsi sul continuo aggiornamento e standardizzazione nel campo delle *SSI* per facilitare l'adozione su larga scala.

7.4 Valutazione personale

Questo tirocinio è stato un'ottima occasione per immergermi nell'ambito delle identità digitali e del paradigma *SSI*. Nonostante non sia riuscito a completare pienamente l'implementazione del sistema di scambio di *credenziali*, ho acquisito una profonda comprensione delle complessità e delle potenzialità del campo. Questa esperienza ha rafforzato il mio interesse e la mia determinazione nel perseguire ulteriori studi e contribuire allo sviluppo di soluzioni innovative per la gestione delle identità digitali nell'era digitale. La strada da percorrere è ancora lunga, ma le fondamenta gettate in questa tesi costituiscono un punto di partenza promettente per il futuro.

Acronimi e abbreviazioni

API REST Application Programming Interface Representational State Transfer. 29, 34, 35, 42, 55

DApp Decentralized Applications. 35

DID Decentralized Identifiers. vii, 10, 22–24, 28, 56

DID Document Decentralized Identifier Document. 24, 25

HTTP Hypertext Transfer Protocol. 5, 34, 35, 38, 43, 47

IAM Identity and Access Managemen. 4, 35, 57

IdP Identity Provider. 12, 17, 31, 58

JSON Javascript Object Notation. 29, 38, 39, 43

PoC Proof of Concept. iii, 3–6, 8, 26, 34, 46, 49, 51, 58

SPID Sistema Pubblico di Identità Digitale. 17, 59

SSH Secure SHell. 34

SSI Self-Sovereign identity. iii, vi, vii, 2, 5, 7, 10, 18, 20, 26, 28, 34, 35, 41, 48, 51, 52, 59

SSO Single Sign-On. 17, 60

URI Uniform Resource Identifier. 23, 24

VC Verifiable Credentials. iii, vii, 4, 5, 8, 10, 20–22, 28, 35, 37–39, 41, 43–45, 49–51, 58, 60, 61

VDR Verifiable Data Registry. 25

VP Verifiable Presentation. 21, 39, 60, 61

VPN Virtual Private Network. 34

Glossario

Agent Nella *Self-Sovereign Identity (SSI)*, un agent è un software che facilita la gestione e l'uso delle identità digitali. Svolge ruoli chiave come la creazione, la gestione e l'uso di credenziali digitali, assicura comunicazioni sicure, supporta l'interoperabilità tra diversi sistemi, e aiuta a mantenere la privacy e il controllo dell'utente sulla propria identità digitale. Gli agenti operano in ambienti decentralizzati, spesso interagendo con tecnologie *blockchain* o *Distributed Ledger Technology (DLT)*. 29, 35, 37, 43

API REST Un'API REST (Application Programming Interface Representational State Transfer) è un tipo di architettura per la progettazione di servizi web che si basa su principi di rappresentazione delle risorse tramite *URL*, supporta operazioni *CRUD* tramite metodi *HTTP* standard, è *stateless* e utilizza risposte generalmente formattate in *JSON* o *XML*. Le API REST sono ampiamente utilizzate per creare servizi web scalabili e facili da utilizzare. 5, 30, 38, 43, 46, 53

Athesys s.r.l Azienda ospitante dello stage. iii, vii, ix, 4, 9, 34, 36, 41, 46, 48, 58

Blockchain Una *blockchain* è un registro digitale decentralizzato e sicuro che tiene traccia delle transazioni o dei dati in modo che siano immutabili e accessibili pubblicamente. Utilizza la crittografia per garantire l'integrità e la sicurezza dei dati, ed è spesso associata alle *criptovalute* come il *Bitcoin*, ma può essere utilizzata per una varietà di scopi, tra cui la registrazione di *"smart contract"*. 2, 12, 18, 23, 26, 28, 49, 50, 57

Callback In questa tesi callback si intende nel caso della programmazione basata sugli eventi. I callback vengono spesso usati per definire come reagire a determinati eventi, come clic del mouse o pressioni di tasti. Il callback non viene eseguito immediatamente; è eseguito solo quando l'operazione alla quale è associato è completata, permettendo così di avere un controllo maggiore sull'ordine di esecuzione del codice. 45

Claims Un'affermazione (*claim*) su un attributo di un soggetto. Esempi di asserzione sono la data di nascita, l'altezza, il numero di identificazione del governo o l'indirizzo postale, tutti possibili attributi di un individuo. Una credenziale è costituita da un insieme di asserzioni. 21, 60

Criptovalute Monete digitali o virtuali che utilizzano la crittografia per la sicurezza delle transazioni. Le criptovalute operano su una tecnologia decentralizzata, come la *blockchain*, e permettono scambi finanziari al di fuori dei sistemi bancari tradizionali. 50, 55

Cyberchef Strumento online open-source sviluppato principalmente per l'analisi e la decodifica di dati. CyberChef offre una serie di operazioni di codifica, decodifica, cifratura e hashing, utili in ambito forense digitale e nella cybersecurity. 47

Debian Sistema operativo *open-source* basato su Linux. Debian è noto per la sua stabilità e affidabilità e viene utilizzato sia su server che su desktop. È la base per molti altri sistemi operativi, inclusi Ubuntu e Raspberry Pi OS. 34

DID Nella *Self-Sovereign Identity (SSI)*, un *DID (Decentralized Identifier)* è un identificativo unico e persistente creato e gestito in modo decentralizzato. Un DID consente ad un individuo o un'entità di dimostrare il controllo sulla propria identità digitale senza dover fare affidamento su un'autorità centralizzata. Questi identificativi sono spesso registrati su una *blockchain* o un'altra forma di *Distributed Ledger Technology (DLT)*, garantendo sicurezza, trasparenza e resistenza alla censura.. 23, 53

Endpoint Un endpoint in un'API RESTful è un punto di accesso tramite URL che consente di interagire con una specifica funzionalità o risorsa. Ogni endpoint corrisponde a un'azione definita, come recuperare informazioni, aggiornare dati, eliminarli o eseguire altre operazioni sulle risorse offerte dall'API. Ad esempio, un endpoint potrebbe essere /books per ottenere l'elenco dei libri o /users/123 per accedere ai dettagli dell'utente con ID 123. 29

Express Framework per applicazioni web Node.js. Express semplifica lo sviluppo di applicazioni web e API fornendo funzionalità come il routing, la gestione degli errori e l'integrazione con altri moduli e middleware. 38

GitHub Piattaforma di hosting per il controllo versione e la collaborazione nello sviluppo software. GitHub utilizza Git, un sistema di controllo versione distribuito, per permettere a sviluppatori di collaborare su progetti da qualsiasi parte del mondo. 35

Holder Un ruolo che un'entità può svolgere possedendo una o più credenziali verificabili e generando presentazioni verificabili da esse. Esempi di holder sono gli studenti, i dipendenti e i clienti. 20, 28, 35, 37–39, 41, 43

Hyperledger Aries Fornisce un *toolkit* progettato per iniziative e soluzioni incentrate sulla creazione, trasmissione, archiviazione e utilizzo di credenziali verificabili. *Aries* è un'infrastruttura per interazioni *peer-to-peer* radicate su *blockchain*. iii, 4, 5, 10, 35, 46, 49, 51

IAM Gestione delle identità e degli accessi è un insieme di processi, politiche, strumenti e tecnologie utilizzati per gestire l'identificazione, l'autenticazione, l'autorizzazione e la gestione delle autorizzazioni degli utenti all'interno di un sistema informativo o di un'applicazione. L'obiettivo principale dell'*Identity and Access Management* è garantire che le risorse digitali siano accessibili solo agli utenti autorizzati, al tempo stesso facilitando la gestione delle identità e dei diritti di accesso. Questo approccio è fondamentale per garantire la sicurezza, la conformità normativa e la gestione efficiente delle risorse digitali in un'organizzazione. vii, 4, 5, 43, 53

IdP Un *Identity Provider* (IdP) è un sistema che crea, archivia e gestisce le identità digitali. L'*IdP* può autenticare direttamente l'utente o fornire servizi di autenticazione a provider di servizi di terze parti (app, siti web o altri servizi digitali). 53

Issue In ambito di sviluppo software, una "*issue*" è un termine utilizzato per descrivere un problema, un bug, un compito o una richiesta di miglioramento in un progetto software. Gli issue sono spesso tracciati e gestiti attraverso sistemi come GitHub o Jira. 36

Issuer Un ruolo che un'entità svolge asserendo affermazioni su uno o più soggetti, creando una Verifiable Credentials (VC) da tali affermazioni e trasmettendo la credenziale verificabile a un titolare. Esempi di emittenti sono le aziende, le organizzazioni non profit, le associazioni di categoria, i governi e gli individui. 20, 28, 35, 38, 41, 43

Monokee Gestisci le tue identità digitali attraverso i principali standard del mercato e abbraccia il nuovo paradigma in cui le persone riprendono possesso della loro identità digitale. Collabora con Athesys s.r.l. 4, 5, 35, 43

Notion Applicazione multifunzionale che funge da spazio di lavoro tutto-in-uno. Consente la gestione di note, task, database e documenti collaborativi, integrando diverse funzionalità di organizzazione e pianificazione in un'unica piattaforma. 47

Peer-to-Peer Modello di rete in cui ogni partecipante (*peer*) ha capacità e responsabilità simili. Utilizzato in vari contesti, dai file *sharing* alle *criptovalute*, il P2P elimina la necessità di server centralizzati. 12, 57

PoC La locuzione inglese *Proof of Concept*, abbreviata come *PoC* (che letteralmente significa, "prova di concetto", e può essere tradotta in italiano come "prova di fattibilità", o "dimostrazione di fattibilità"), si riferisce a una realizzazione incompleta o abbozzata (sinopsi) di un determinato progetto o metodo. Il suo

scopo è quello di dimostrare la fattibilità o confermare la validità di alcuni principi o concetti fondamentali. Un esempio tipico è quello di un prototipo. 53

Postman Strumento popolare per lo sviluppo e il test di API. Postman permette agli sviluppatori di creare, condividere, testare e documentare API in modo semplice e intuitivo. 47

Proof In ambito informatico e matematico, si riferisce a una dimostrazione o a una prova che conferma la veridicità di un'affermazione o di un algoritmo. In criptografia, termini come "*proof of work*" o "*proof of stake*" descrivono i meccanismi alla base del funzionamento di diverse *criptovalute*. 37

Repository In informatica, un repository è un luogo centralizzato in cui vengono conservati e gestiti i dati, spesso codice sorgente o documentazione. In contesti come GitHub, un repository è il luogo in cui vengono conservate tutte le informazioni relative a un progetto software. 15

Schema Si tratta di una definizione comprensibile per le macchine, che specifica la semantica di una struttura di dati. Questa definizione è impiegata per determinare gli attributi usati in una o più *Credential Definition*. 37, 39

SPID Lo SPID, acronimo di "Sistema Pubblico di Identità Digitale," è un sistema italiano che consente ai cittadini di autenticarsi in modo sicuro su servizi online forniti da enti pubblici e privati. In pratica, lo SPID funge da identità digitale federata 3.3.2, consentendo alle persone di accedere a vari servizi online senza dover creare un account separato per ciascun servizio. È stato introdotto per semplificare l'accesso e migliorare la sicurezza online in Italia, consentendo l'identificazione e l'autenticazione attraverso un'unica credenziale digitale. Gli utenti possono ottenere un SPID attraverso specifici fornitori di servizi di identità autorizzati dal governo italiano. 53

SSI Nell'ambito dell'identità digitale è un modello che offre agli individui il controllo sui propri dati personali e sulle modalità di condivisione. Con l'*SSI*, gli individui

possono memorizzare le informazioni sulla propria identità sui propri dispositivi o in una rete decentralizzata, invece di affidarsi ad autorità centralizzate per gestire la propria identità. L'*SSI* è spesso utilizzato nel contesto delle credenziali verificabili, che consentono agli individui di condividere informazioni specifiche su di sé senza rivelare informazioni non necessarie. Ciò può essere utile per l'autenticazione, il controllo degli accessi e la verifica dell'identità in diversi contesti. 5, 53

SSO Il Single Sign-On (SSO) è un sistema di autenticazione che consente a un utente di accedere a diverse applicazioni o servizi con una sola identificazione e password. In pratica, una volta che un utente si è autenticato con successo su una piattaforma, come ad esempio un'applicazione aziendale o un social media, può accedere ad altre applicazioni o servizi correlati senza dover reinserire le credenziali. 53

Ubuntu Sistema operativo basato su Linux, sviluppato dalla società Canonical Ltd. Ubuntu è noto per la sua facilità d'uso e la sua ampia diffusione sia in ambito desktop che server. Il suo nome deriva dal concetto africano di "Ubuntu", che significa "umanità verso gli altri". 35

VC In italiano *Credenziali Verificabili*, sono un elenco portabile di affermazioni (*Claims*) su un argomento firmate da una o più autorità. Una credenziale verificabile è una credenziale a prova di manomissione la cui paternità può essere verificata *crittograficamente*. Le credenziali verificabili possono essere utilizzate per creare *Verifiable Presentation (VP)*, che possono anch'esse essere verificate *crittograficamente*. 20, 54, 60

Verifier Un ruolo che un'entità svolge ricevendo una o più VC, facoltativamente all'interno di una VP, per l'elaborazione e la verifica. Per la verifica viene fatta una valutazione del fatto che una credenziale verificabile o una presentazione verificabile sia una dichiarazione autentica e tempestiva dell'emittente o del presentatore, rispettivamente. Ciò include la verifica che: la VC (o la VP) sia conforme alla specifica; il metodo di prova sia soddisfatto; e, se presente, la

verifica di stato abbia successo. Esempi di verificatori sono i datori di lavoro, il personale di sicurezza e i siti web. 20, 37

VP In italiano *Presentazioni Verificabili* sono dati derivati da una o più *Verifiable Credentials (VC)*, emesse da uno o più emittenti, condivisi con un verificatore specifico. Una presentazione verificabile è una presentazione a prova di manomissione codificata in modo tale che la paternità dei dati possa essere considerata attendibile dopo un processo di verifica crittografica. Alcuni tipi di presentazioni verificabili potrebbero contenere dati sintetizzati ma non contenenti le credenziali verificabili originali (ad esempio le *zero-knowledge proofs*). 54, 60

Wallet Strumento installato sui dispositivi di *Issuer* ed *Holder* per la gestione delle identità digitali e delle interazioni con altre entità. Nel mio caso ho avuto a che fare con due wallet differenti: *Indy* (deprecato) e *Aries Askar*. vii, 10, 20, 25, 29, 35, 36, 41, 49

Bibliografia

Articoli consultati

Cameron, Kim. «The Laws of Identity». In: *Architect of Identity* (2005) (cit. alle pp. 1, 31, 50).

Siti web consultati

A dad took photos of his naked toddler for the doctor. Google flagged him as a criminal. URL: https://www.forbesindia.com/article/news/a-dad-took-photos-of-his-naked-toddler-for-the-doctor-google-flagged-him-as-a-criminal/79159/1 (cit. a p. 17).

Cos'è un'API REST? URL: https://www.redhat.com/it/topics/api/what-is-a-rest-api.

Demo credenziali verificabili Animo. URL: https://demo.animo.id (cit. a p. 41).

Digital Identity Models. URL: https://www.selfsovereignidentity.it/identity-management-models/ (cit. a p. 19).

Hyperledger Aries Documentation. URL: https://aries.js.org/guides/0.4/getting-started/set-up/aries-askar.

Hyperledger Aries GitHub page. URL: https://github.com/hyperledger/aries.

Hyperledger Foundation. URL: https://www.linuxfoundation.org/resources/case-studies/hyperledger (cit. a p. 26).

Identity Provider, cos'è? URL: https://www.entrust.com/it/resources/faq/what-is-an-identity-provider.

Lissi wallet. URL: https://www.lissi.id (cit. a p. 41).

Not your keys, not your coins. URL: https://decrypt.co/resources/coins-keys-wallet (cit. a p. 19).

Notion. URL: https://www.notion.so.

Orbit Edge. URL: https://northernblock.io/orbit-edge-wallet/ (cit. a p. 41).

Postman. URL: https://www.postman.com.

Self Sovereign Identity. URL: https://www.selfsovereignidentity.it/introduzione-alla-self-sovereign-identity/ (cit. a p. 1).

TEMU App is Cleverly Hidden Spyware. URL: https://grizzlyreports.com/we-believe-pdd-is-a-dying-fraudulent-company-and-its-shopping-app-temu-is-cleverly-hidden-spyware-that-poses-an-urgent-security-threat-to-u-s-national-interests (cit. a p. 12).