frontiers | Frontiers in Blockchain

TYPE Technology and Code
PUBLISHED 12 March 2024
DOI 10.3389/fbloc.2024.1374655

Challenges of user data privacy in self-sovereign identity verifiable credentials for autonomous building access during the COVID-19 pandemic

Denver Naicker* and Mackaylan Moodley

Council for Scientific and Industrial Research, Next Generation Enterprise and Institutions Department, Distributed Ledger Technology Research Group, Pretoria, South Africa

OPEN ACCESS

EDITED BY
Shitharth Selvarajan,
Leeds Beckett University, United Kingdom

REVIEWED BY
Hariprasath Manoharan,
Panimalar Institute of Technology, India
Praveen Kumar Balachandran,
Vardhaman College of Engineering, India
Anupama Mishra,
Swami Rama Himalayan University, India

*CORRESPONDENCE
Denver Naicker,
✉ DNaicker2@csir.co.za

RECEIVED 22 January 2024
ACCEPTED 20 February 2024
PUBLISHED 12 March 2024

CITATION
Naicker D and Moodley M (2024), Challenges of user data privacy in self-sovereign identity verifiable credentials for autonomous building access during the COVID-19 pandemic.
Front. Blockchain 7:1374655.
doi: 10.3389/fbloc.2024.1374655

Self-sovereign identity is an emerging blockchain technology field. Its use cases primarily surround identity and credential management and advocate the privacy of user details during the verification process. Our endeavor was to test and implement the features promoted for self-sovereign identity through open- and closed-source frameworks utilizing a scenario of building access management to adhere to health risk and safety questionnaires during the COVID-19 pandemic. Our investigation identifies whether user data privacy could be ensured through verifiable credentials and whether business practices would need to evolve to mitigate storing personal data centrally.

KEYWORDS

self-sovereign identity, verifiable credentials, building access, autonomous, privacy, health and safety, risk assessment

1 Introduction

The management and protection of personal identity information have become increasingly complex and fraught with challenges. Traditional identity systems, characterized by centralized databases and intermediaries, often fail to adequately safeguard individuals' privacy and security while also hindering seamless interactions across digital platforms. In response to these shortcomings, a paradigm shift toward self-sovereign identity (SSI) has emerged as a promising alternative. At its core, SSI represents a groundbreaking approach to identity management that prioritizes individual control, autonomy, and privacy over personal identity information. Unlike conventional identity systems, which rely on centralized authorities to verify and authenticate user identities, SSI empowers individuals with the ability to assert and manage their own digital identities autonomously. Key to the concept of SSI is the principle of user-centricity, which places individuals at the center of the identity ecosystem, granting them sole ownership and control over their identity attributes. Through the use of decentralized technologies such as blockchain, individuals can create, manage, and selectively disclose their digital identities without reliance on intermediaries or third-party service providers. At its most fundamental level, an SSI comprises a set of verifiable credentials or claims that attest to various aspects of an individual's identity, such as their name, age, address, or qualifications. These credentials are cryptographically signed by issuers, such as government agencies, educational institutions, or employers, to ensure their authenticity and integrity. Crucially, individuals retain full control over the sharing and disclosure of their identity

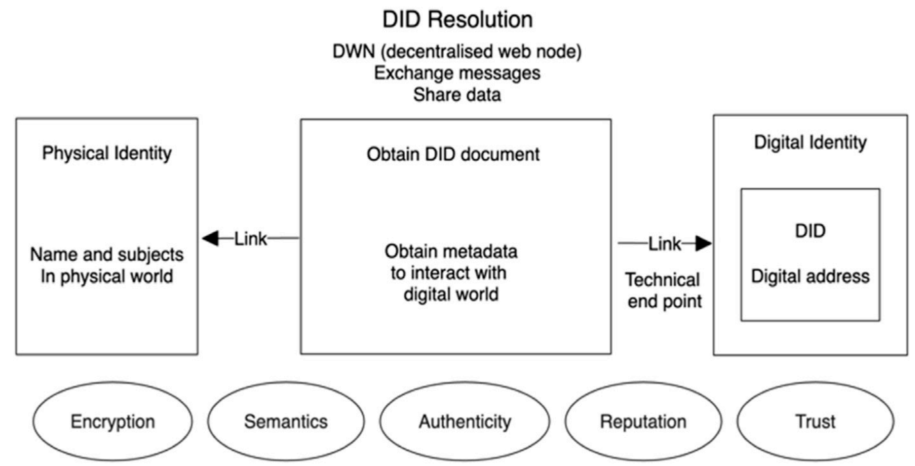


FIGURE 1
High-level representation of DID resolution linkage.

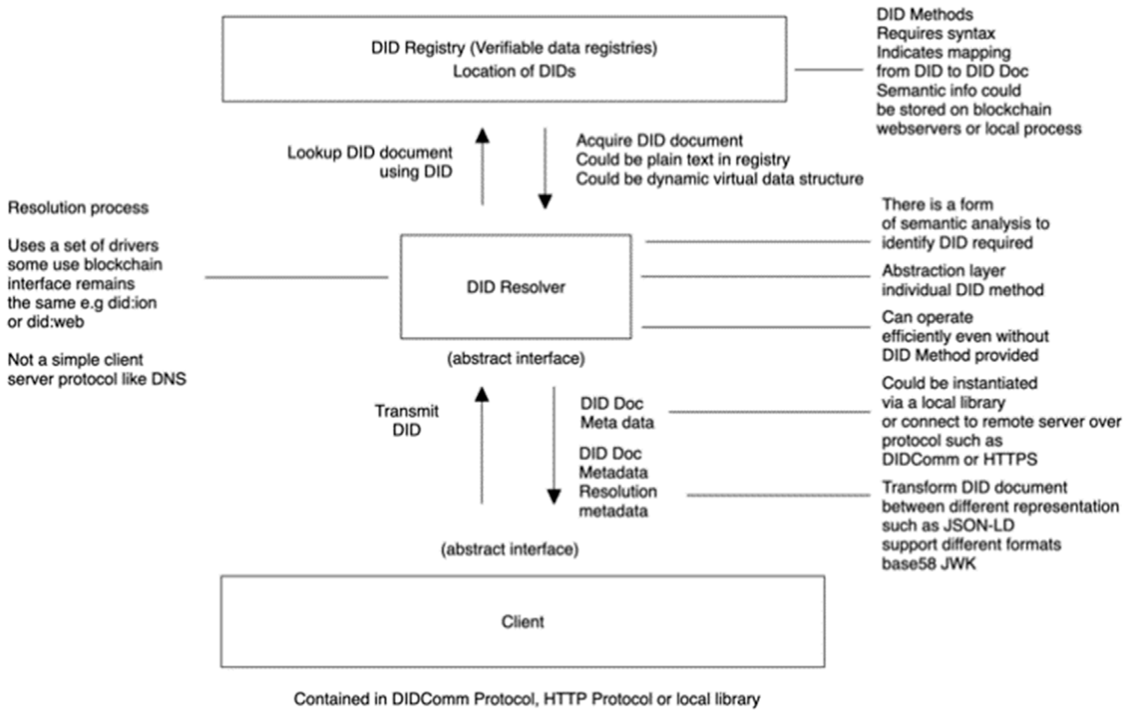
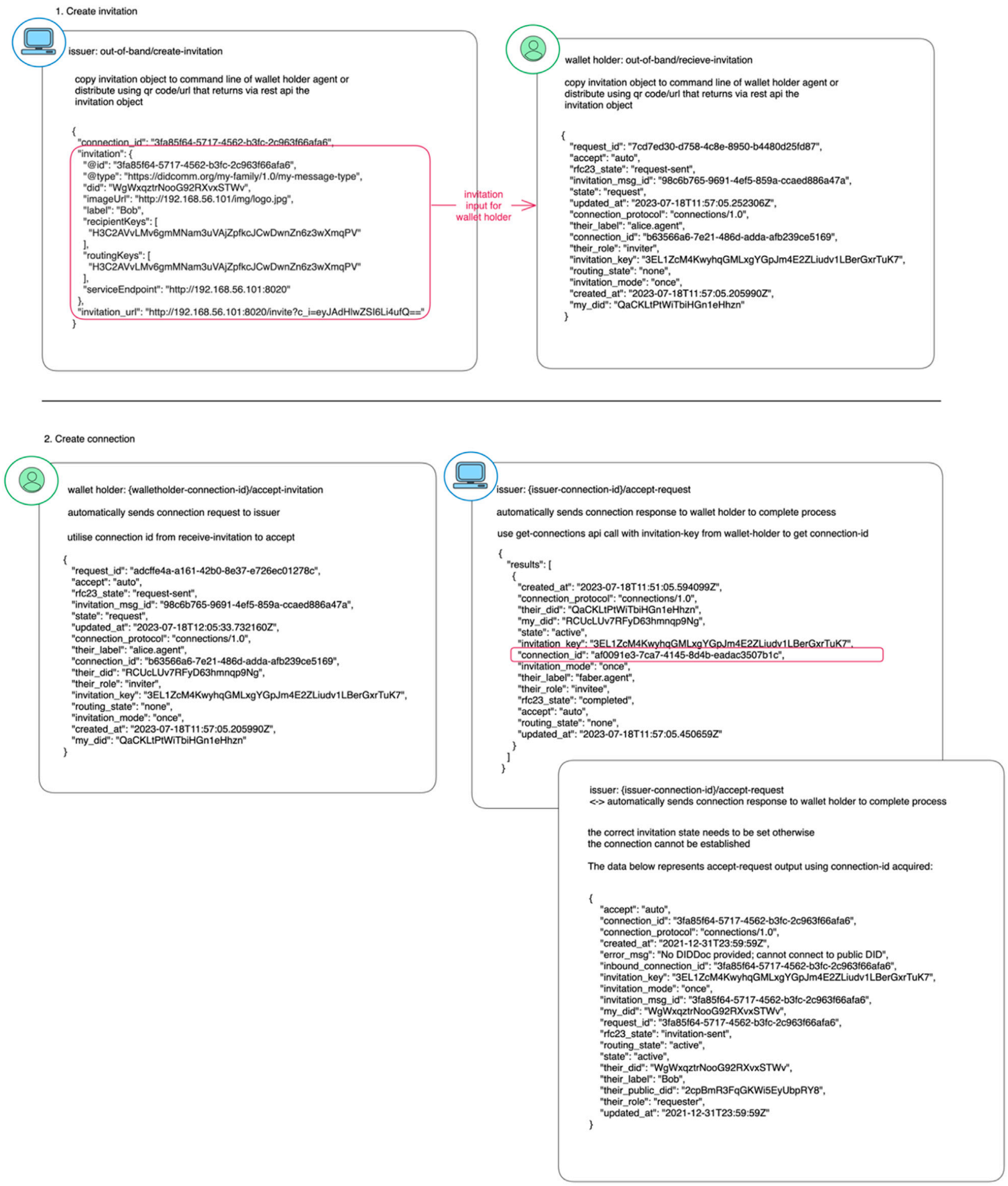


FIGURE 2
Process flow of DID lookup from DID resolver to trust registry.

attributes, deciding when, where, and to whom they wish to present their credentials. This not only enhances privacy and data protection but also streamlines identity verification processes, reducing the need for redundant identity checks and minimizing the risk of identity theft or fraud. SSI holds the potential to revolutionize digital interactions by enabling seamless and secure identity authentication across diverse applications and services. From accessing online banking services and e-commerce platforms to participating in healthcare systems and voting processes, individuals can leverage their self-sovereign identities to assert their identity and establish trust in a wide range of contexts.

The SSI system developed integrates a credential management application for issuers as well as a mobile application designed for secure storage and exchange of verifiable credential presentation proofs. This system enables interaction with business operations using quick response (QR) codes. The solution leverages third-party frameworks to enhance efficiency and explore the applicability of SSI in automating access verification for public buildings. This is achieved without



1. Create invitation

issuer: out-of-band/create-invitation

copy invitation object to command line of wallet holder agent or
distribute using qr code/url that returns via rest api the
invitation object

```
{
  "connection_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "invitation": {
    "@id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
    "@type": "https://didcomm.org/my-family/1.0/my-message-type",
    "did": "WgWxqztrNooG92RXvxSTWv",
    "imageUrl": "http://192.168.56.101/img/logo.jpg",
    "label": "Bob",
    "recipientKeys": [
      "H3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
    ],
    "routingKeys": [
      "H3C2AVvLMv6gmMNam3uVAjZpfkcJCwDwnZn6z3wXmqPV"
    ],
    "serviceEndpoint": "http://192.168.56.101:8020"
  },
  "invitation_url": "http://192.168.56.101:8020/invite?c_i=eyJAdHlwZSI6Li4ufQ=="
}
```

wallet holder: out-of-band/recieve-invitation

copy invitation object to command line of wallet holder agent or
distribute using qr code/url that returns via rest api the
invitation object

```
{
  "request_id": "7cd7ed30-d758-4c8e-8950-b4480d25fd87",
  "accept": "auto",
  "rfc23_state": "request-sent",
  "invitation_msg_id": "98c6b765-9691-4ef5-859a-ccaed886a47a",
  "state": "request",
  "updated_at": "2023-07-18T11:57:05.252306Z",
  "connection_protocol": "connections/1.0",
  "their_label": "alice.agent",
  "connection_id": "b63566a6-7e21-486d-adda-afb239ce5169",
  "their_role": "inviter",
  "invitation_key": "3EL1ZcM4KwyhqGMLxgYGpJm4E2ZLiudv1LBerGxrTuK7",
  "routing_state": "none",
  "invitation_mode": "once",
  "created_at": "2023-07-18T11:57:05.205990Z",
  "my_did": "QaCKLtPtWiTbiHGn1eHhzn"
}
```

2. Create connection

wallet holder: {walletholder-connection-id}/accept-invitation

automatically sends connection request to issuer

utilise connection id from receive-invitation to accept

```
{
  "request_id": "adcffe4a-a161-42b0-8e37-e726ec01278c",
  "accept": "auto",
  "rfc23_state": "request-sent",
  "invitation_msg_id": "98c6b765-9691-4ef5-859a-ccaed886a47a",
  "state": "request",
  "updated_at": "2023-07-18T12:05:33.732160Z",
  "connection_protocol": "connections/1.0",
  "their_label": "alice.agent",
  "connection_id": "b63566a6-7e21-486d-adda-afb239ce5169",
  "their_did": "RCUcLUv7RFyD63hmnqp9Ng",
  "their_role": "inviter",
  "invitation_key": "3EL1ZcM4KwyhqGMLxgYGpJm4E2ZLiudv1LBerGxrTuK7",
  "routing_state": "none",
  "invitation_mode": "once",
  "created_at": "2023-07-18T11:57:05.205990Z",
  "my_did": "QaCKLtPtWiTbiHGn1eHhzn"
}
```

issuer: {issuer-connection-id}/accept-request

automatically sends connection response to wallet holder to complete process

use get-connections api call with invitation-key from wallet-holder to get connection-id

```
{
  "results": [
    {
      "created_at": "2023-07-18T11:51:05.594099Z",
      "connection_protocol": "connections/1.0",
      "their_did": "QaCKLtPtWiTbiHGn1eHhzn",
      "my_did": "RCUcLUv7RFyD63hmnqp9Ng",
      "state": "active",
      "invitation_key": "3EL1ZcM4KwyhqGMLxgYGpJm4E2ZLiudv1LBerGxrTuK7",
      "connection_id": "af0091e3-7ca7-4145-8d4b-eadac3507b1c",
      "invitation_mode": "once",
      "their_label": "faber.agent",
      "their_role": "invitee",
      "rfc23_state": "completed",
      "accept": "auto",
      "routing_state": "none",
      "updated_at": "2023-07-18T11:57:05.450659Z"
    }
  ]
}
```

issuer: {issuer-connection-id}/accept-request
<-> automatically sends connection response to wallet holder to complete process

the correct invitation state needs to be set otherwise
the connection cannot be established

The data below represents accept-request output using connection-id acquired:

```
{
  "accept": "auto",
  "connection_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "connection_protocol": "connections/1.0",
  "created_at": "2021-12-31T23:59:59Z",
  "error_msg": "No DIDDoc provided; cannot connect to public DID",
  "inbound_connection_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "invitation_key": "3EL1ZcM4KwyhqGMLxgYGpJm4E2ZLiudv1LBerGxrTuK7",
  "invitation_mode": "once",
  "invitation_msg_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "my_did": "WgWxqztrNooG92RXvxSTWv",
  "request_id": "3fa85f64-5717-4562-b3fc-2c963f66afa6",
  "rfc23_state": "invitation-sent",
  "routing_state": "active",
  "state": "active",
  "their_did": "WgWxqztrNooG92RXvxSTWv",
  "their_label": "Bob",
  "their_public_did": "2cpBmR3FqGKWi5EyUbpRY8",
  "their_role": "requester",
  "updated_at": "2021-12-31T23:59:59Z"
}
```

FIGURE 3
Identifying patterns of common data between SSI agent communication in Aries protocol.

compromising user confidentiality, utilizing zero-knowledge proof (ZKP) attribute verification methods to query the validity of the credential schema and trustworthy checks to verify the credential source. Central to this system is the incorporation of business rule validation into the credential acquisition and verification process. Credential proofs are exchanged via the user's mobile wallet device and verified autonomously via the business rule set on the verifier's database. This verification is facilitated by deriving rules from a maintenance table within a verifier database, replacing the need for further source code development with configuration of the maintenance

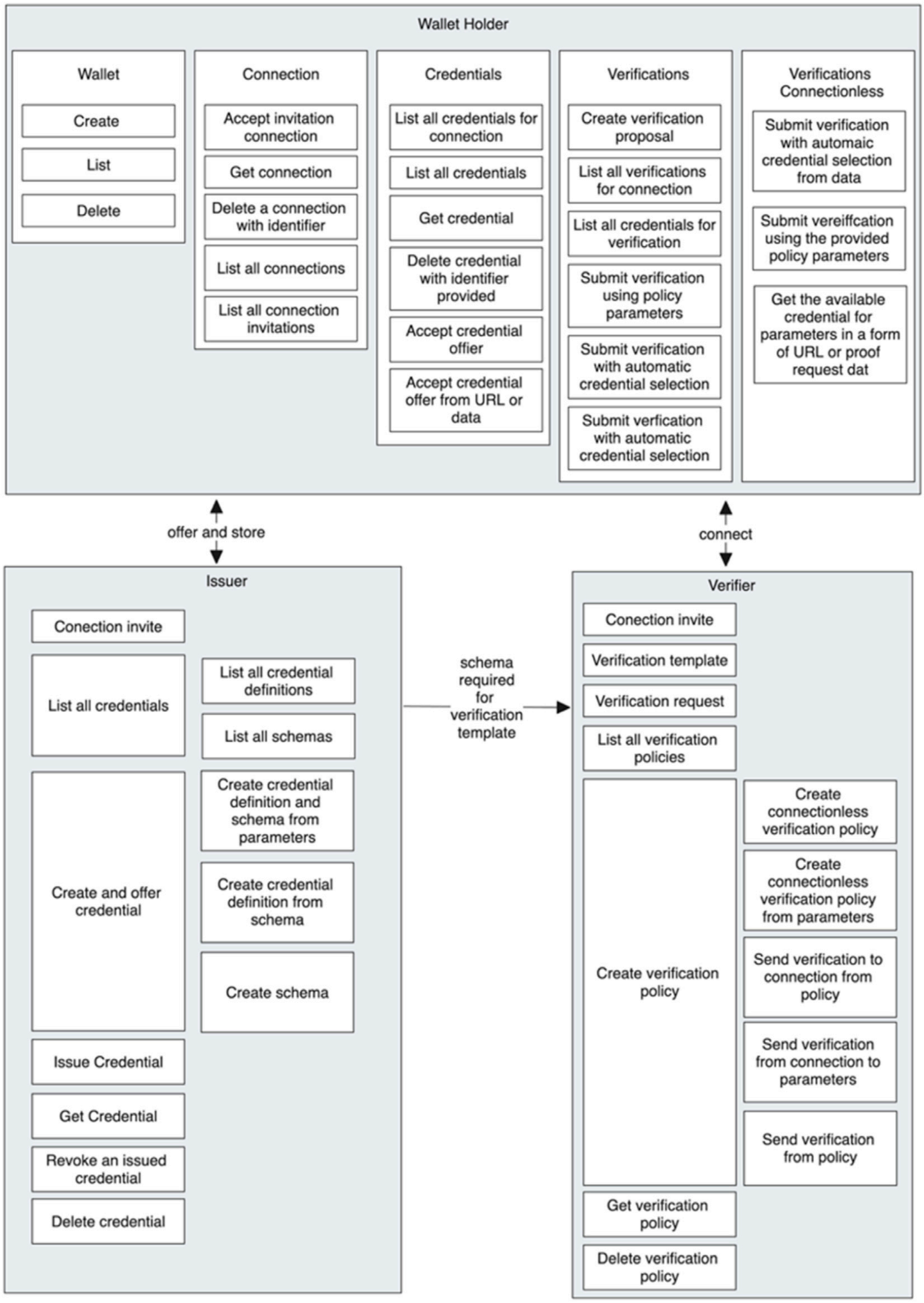
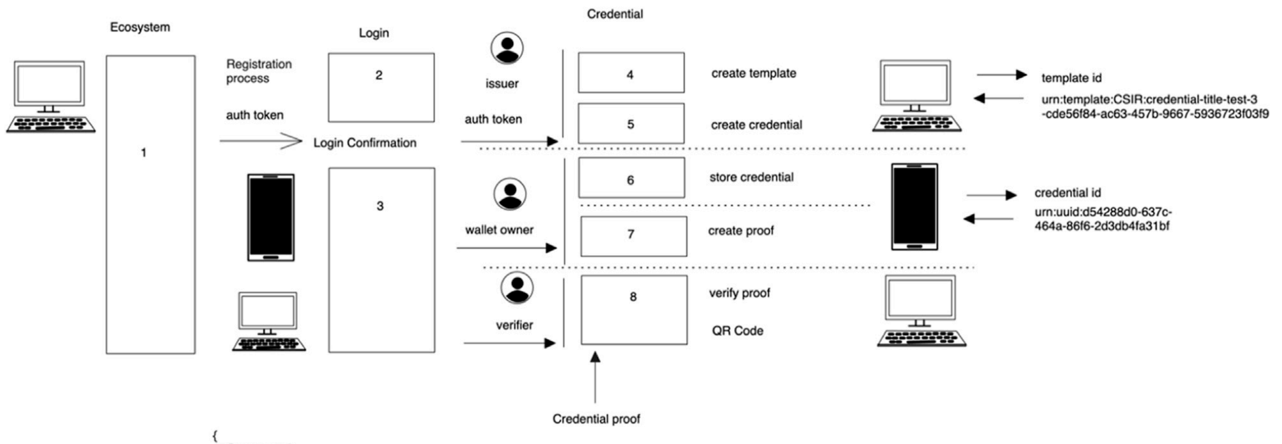

Wallet Holder
Wallet
Create
List
Delete
Connection
Accept invitation connection
Get connection
Delete a connection with identifier
List all connections
List all connection invitations
Credentials
List all credentials for connection
List all credentials
Get credential
Delete credential with identifier provided
Accept credential offier
Accept credential offer from URL or data
Verifications
Create verification proposal
List all verifications for connection
List all credentials for verification
Submit verification using policy parameters
Submit verification with automatic credential selection
Submit verfication with automatic credential selection
Verifications Connectionless
Submit verification with automaic credential selection from data
Submit vereiffcation using the provided policy parameters
Get the available credential for parameters in a form of URL or proof request dat
offer and store
connect
Issuer
Conection invite
List all credentials
List all credential definitions
List all schemas
Create and offer credential
Create credential definition and schema from parameters
Create credential definition from schema
Create schema
Issue Credential
Get Credential
Revoke an issued credential
Delete credential
schema required for verification template
Verifier
Conection invite
Verification template
Verification request
List all verification policies
Create verification policy
Create connectionless verification policy
Create connectionless verification policy from parameters
Send verification to connection from policy
Send verification from connection to parameters
Send verification from policy
Get verification policy
Delete verification policy

FIGURE 4
Trinsic v1 architecture.

table via administration interfaces to implement business rules. A significant advantage of this approach is its compliance with data protection standards, as it reduces the need to store user data. This is possible due to the use of a verifiable credential store in the user's mobile wallet, localizing user data solely to their own device. This demonstrates business services can be executed without monitoring user behavior post-transaction. It enables autonomous validation of specific rule checks at entry points, potentially allowing users to repurpose verifiable credentials for accessing other systems through QR-code-enabled Internet of Things (IoT) devices.



```
{
  "@context": [
    "https://www.w3.org/2018/credentials/v1",
    "https://w3id.org/bbs/v1",
    "https://schema.trinsic.cloud/CSIR/credential-title-test-3-cde56f84-ac63-457b-9667-5936723f03f9/context"
  ],
  "id": "urn:uuid:64b72f1c86f34ad2a65d7b20a92adbe1",
  "type": [
    "CredentialTitleTest3-cde56f84-ac63-457b-9667-5936723f03f9",
    "VerifiableCredential"
  ],
  "credentialSchema": {
    "id": "https://schema.trinsic.cloud/CSIR/credential-title-test-3-cde56f84-ac63-457b-9667-5936723f03f9",
    "type": "JsonSchemaValidator2018"
  },
  "credentialStatus": {
    "id": "urn:revocation-registry:intelligent-goodall-8QdAp6zVFeT:E95rX3mLwqYVPtt3DLcrC7#1",
    "type": "RevocationList2020Status",
    "revocationListCredential": "urn:revocation-registry:intelligent-goodall-8QdAp6zVFeT:E95rX3mLwqYVPtt3DLcrC7",
    "revocationListIndex": "1"
  },
  "credentialSubject": {
    "id": "urn:uuid:f4ba57366bbb428ebcea16dd37550a25",
    "test_name_3": "test value 3"
  },
  "issuanceDate": "2022-11-15T07:35:07.2344257Z",
  "issuer": "did:key:z5TcDaC3hWxTACAbQNXTKpumJUCGPjyoRiJjYBKZj6jRFGrGUR4d58BYTg4Btqi5LA2udUE4pkCqyHrNdq
Y5A3MoRCWX5dhiZvhrHhvdqYujfRthubmmGwFjNFBoHPjiZSNkq2p2Cu5ZSzNzHjPzxuFEYFJijNELmXybBspJBnDkVktQMBo9v
WysKRk5w3c72A3C2Hvah",
  "proof": {
    "type": "BbsBlsSignatureProof2020",
    "created": "2022-11-15T07:35:07Z",
    "nonce": "LtJOevCb/C8idEBQ8y1jvlNf0V27odDdAuAEW0cx/8oHp+YspQCRkGxval6d74R0j34=",
    "proofPurpose": "assertionMethod",
    "proofValue": "ABAA//+jrFt7Cxf523IkonGm4E6wThl/3+UBLuc9v9cvTypbNNprn69bF9IgfLBMQENjWG+u/+FcSzBPDnrlGpERoD9VQnBwxw
wxjk6D157h8lFda4PjZjgtDTJc+YfyjZUC9raJEN41XCOIM/secxRfmesC2iqEDDUdMS1SDehIX2rOnuxoVDWW7rRSRdzNhOi75GcAAAB0sr
Bc1PJ3NOk2ust62A/LubvaVY0TOufsTiorDImhWA04jSJQbtdrlXphXemn/OEcAAAAAke5S0CqN5EwhE8HjNxJcMpCVZQMBQAzanjiU357Z+
xCWhF5fPV173FaTi+mEOlbKHJGb+HUf8UTNHHDG8YZsWCuPhmjZO0nlBWAbLK56F7yBUm7QjU8THhXGGvRmxSumlMgpZrAY1e93RM
SOgbkd5kAAAACb+m518v7qsJCE9AD7lgW64tmlYDTqwAiv+zHC4pSYEYVcX+D5k7pZ7o3rw5O1RT+TBtldoQw9DCinY99OrFQ+A==",
   "verificationMethod": "did:key:z5TcDaC3hWxTACAbQNXTKpumJUCGPjyoRiJjYBKZj6jRFGrGUR4d58BYTg4Btqi5LA2udUE4pkCqyHrNdqY5A3MoRCWX5dhi
ZvhrHhvdqYujfRthubmmGwFjNFBoHPjiZSNkq2p2Cu5ZSzNzHjPzxuFEYFJijNELmXybBspJBnDkVktQMBo9vWysKRk5w3c72A3C2Hvah#zUC711nEkxR5RikV
trNWwrmvFpuWvbmcePXajGgraXp9772W8pE8YWYcPUkcjCb6gpUfzmPVfATtoir86kLxDGRWfhitGw9AdsL2E8LAiVybKAHSU5pnpVz53wZG9pM8ShntG9K"
  }
}
```

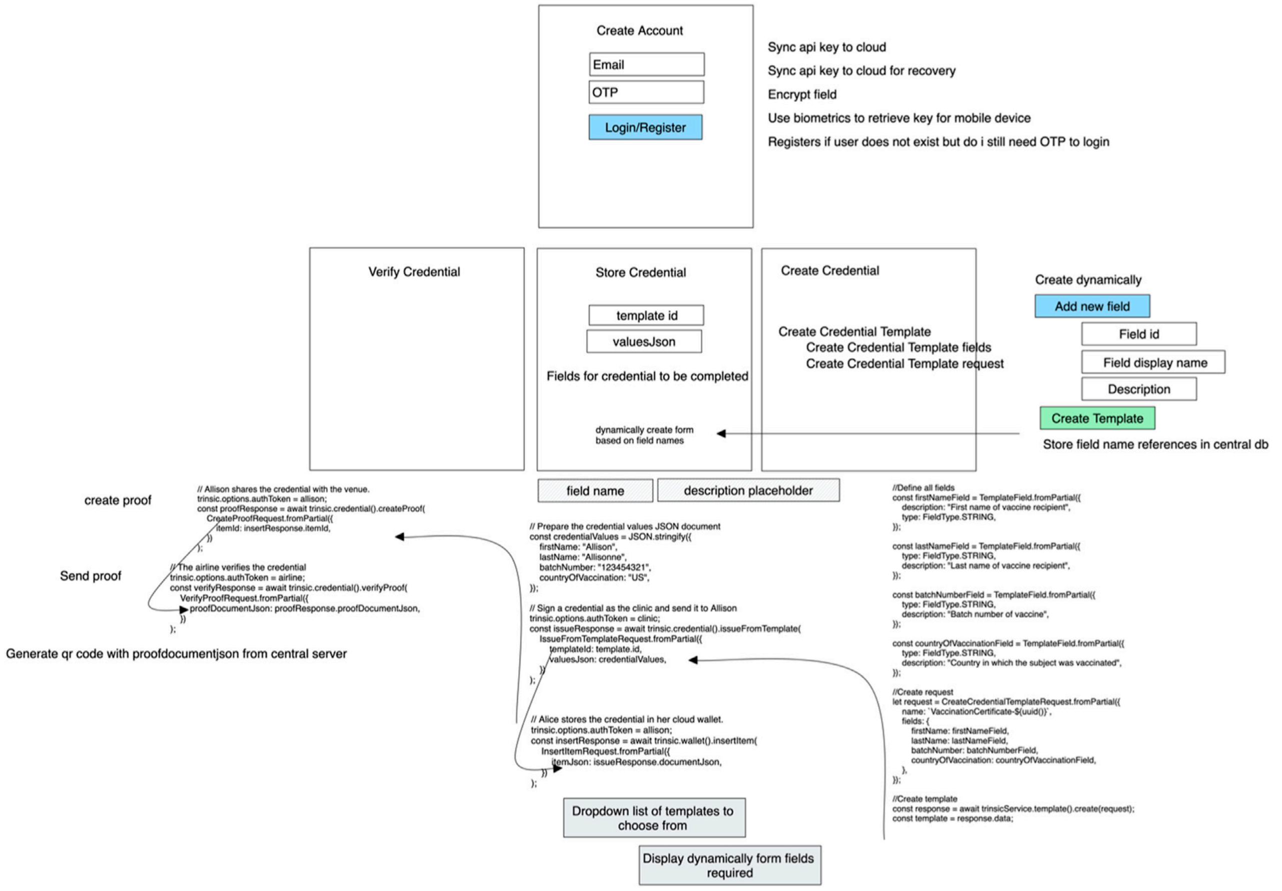
FIGURE 5
Trinsic v2 overview of core system processes.

The trust in this system is derived from the credibility of the credential presentation proof issued by a reputable institution. This approach not only simplifies but also secures the process of access verification, showcasing the potential of SSI to enhance user privacy and operational efficiency in public building facility management. The contribution of this paper establishes a facility of providing a credential management system that integrates business rule validations that can automate the verification process for a specific business use case, in this case, public building access rights, by attaching a ruleset validation alongside the credential for entry into a building. This could address the privacy and health and safety concerns that arose during the COVID-19 pandemic to demonstrate a way to streamline capturing visitor details whilst preserving the privacy of the data through verifiable credential presentation proofs. Localizing that data onto a user's mobile device without storing it on a centralized server mitigates data policy storage regulation, behavior tracking, and exploiting user data concerns that are encapsulated with cybersecurity risks and threats.

This research and development effort was undertaken by the Council for Scientific and Industrial Research (CSIR). The SSI system demonstrates the potential of emerging technologies to address complex challenges and revolutionize established practices in identity management. Moving forward, the CSIR remains committed to advancing the capabilities of SSI systems, driving innovation in digital identity solutions, and empowering individuals with greater control over their personal data. Through ongoing research, development, and collaboration, CSIR seeks to further enhance the security, usability, and scalability of SSI systems, ensuring their widespread adoption and positive impact on society.

2 Background

This paper explores the SSI landscape with a view to developing an application that showcases SSI capabilities. The problem statement for this research encompasses multiple facets. These include understanding where SSI can be leveraged in industrial


Create Account
Email
OTP
Login/Register
Sync api key to cloud
Sync api key to cloud for recovery
Encrypt field
Use biometrics to retrieve key for mobile device
Registers if user does not exist but do i still need OTP to login
Verify Credential
Store Credential
template id
valuesJson
Fields for credential to be completed
dynamically create form based on field names
Create Credential
Create Credential Template
Create Credential Template fields
Create Credential Template request
Create dynamically
Add new field
Field id
Field display name
Description
Create Template
Store field name references in central db
field name
description placeholder
create proof
Send proof
Generate qr code with proofdocumentjson from central server
Dropdown list of templates to choose from
Display dynamically form fields required

```
// Allison shares the credential with the venue.
trinsic.options.authToken = allison;
const proofResponse = await trinsic.credential().createProof(
   CreateProofRequest.fromPartial({
      itemId: insertResponse.itemId,
   })
);

// The airline verifies the credential
trinsic.options.authToken = airline;
const verifyResponse = await trinsic.credential().verifyProof(
   VerifyProofRequest.fromPartial({
      proofDocumentJson: proofResponse.proofDocumentJson,
   })
);
```

```
// Prepare the credential values JSON document
const credentialValues = JSON.stringify({
   firstName: "Allison",
   lastName: "Allisonne",
   batchNumber: "123454321",
   countryOfVaccination: "US",
});

// Sign a credential as the clinic and send it to Allison
trinsic.options.authToken = clinic;
const issueResponse = await trinsic.credential().issueFromTemplate(
   IssueFromTemplateRequest.fromPartial({
      templateId: template.id,
      valuesJson: credentialValues,
   })
);

// Alice stores the credential in her cloud wallet.
trinsic.options.authToken = allison;
const insertResponse = await trinsic.wallet().insertItem(
   InsertItemRequest.fromPartial({
      itemJson: issueResponse.documentJson,
   })
);
```

```
//Define all fields
const firstNameField = TemplateField.fromPartial({
   description: "First name of vaccine recipient",
   type: FieldType.STRING,
});

const lastNameField = TemplateField.fromPartial({
   type: FieldType.STRING,
   description: "Last name of vaccine recipient",
});

const batchNumberField = TemplateField.fromPartial({
   type: FieldType.STRING,
   description: "Batch number of vaccine",
});

const countryOfVaccinationField = TemplateField.fromPartial({
   type: FieldType.STRING,
   description: "Country in which the subject was vaccinated",
});

//Create request
let request = CreateCredentialTemplateRequest.fromPartial({
   name: `VaccinationCertificate-${uuid()}`,
   fields: {
      firstName: firstNameField,
      lastName: lastNameField,
      batchNumber: batchNumberField,
      countryOfVaccination: countryOfVaccinationField,
   },
});

//Create template
const response = await trinsicService.template().create(request);
const template = response.data;
```

FIGURE 6
Trinsic v2 vaccine demonstration code analysis.

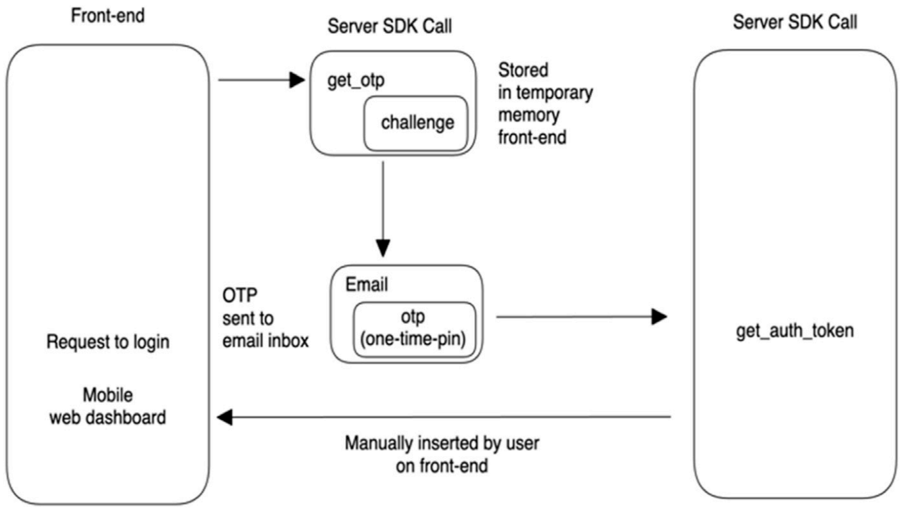


FIGURE 7
Login OTP process flow of return auth token for a user session.

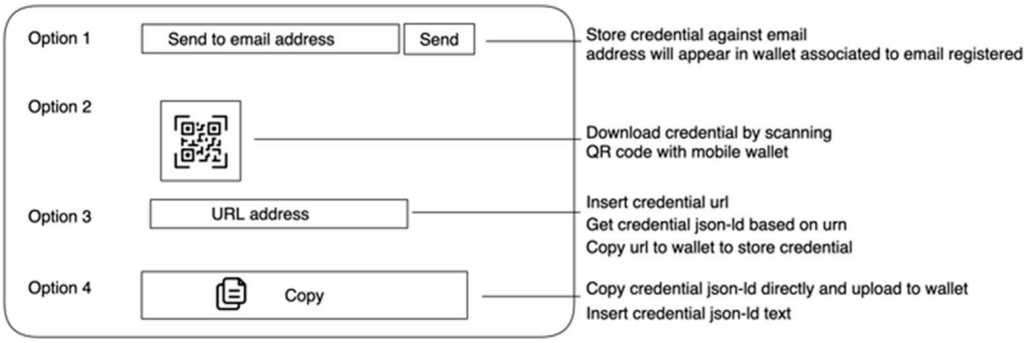

Option 1
Send to email address
Send
Store credential against email
address will appear in wallet associated to email registered
Option 2
Download credential by scanning
QR code with mobile wallet
Option 3
URL address
Insert credential url
Get credential json-ld based on urn
Copy url to wallet to store credential
Option 4
Copy
Copy credential json-ld directly and upload to wallet
Insert credential json-ld text

FIGURE 8
Providing a variety of options for downloading and storing a credential.

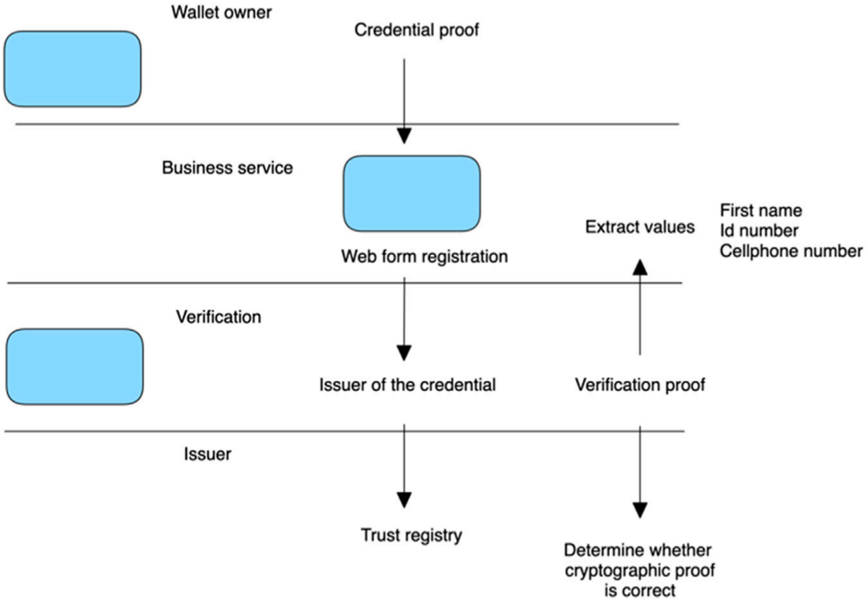


FIGURE 9
High-level process flow of the verification procedure.

sectors for profit, user safety, security, and privacy. The article is also an exploration of the role of privacy in traditional identity model use spaces, specifically in government and institutional workplaces, and the preservation of core values. This research utilizes third-party frameworks like Hyperledger Aries and Trinsic v1 and v2 to demonstrate the viability of SSI in adhering to health and safety guidelines, particularly in the context of post-COVID access to public buildings. The aim is to provide a solution that upholds the privacy of individuals, complies with business operations requirements, and addresses the limitations of current technologies, such as the lack of QR code functionality in Trinsic v2. The exploration extends to open-source and third-party solutions to gauge their ease of use, ease of implementation, and potential for customization, all with a view to packaging and marketing these as business solutions.

Digital identity management has emerged as a significant issue in the modern information era. SSI offers an approach where individuals or businesses have sole ownership over their digital identities and control how these personal data are shared and used (Tech, 2023). SSI solutions come with their own set of challenges. Exploration of these complexities begins with the frameworks and tools used to develop an SSI solution, dissected previous SSI solutions and possible sample demo applications, and explored multiple frameworks and platforms such as Hyperledger Aries, Hyperledger Indy, and Trinsic. Hyperledger Aries introduced the principles of SSI but posed difficulties with a steep learning curve and a plethora of requests for comment (RFC) protocols (Tech, 2023). This paper adopted Trinsic primarily to work around these challenges to focus on a hands-on coding approach with integrated business functionality. This paper further explored business-specific

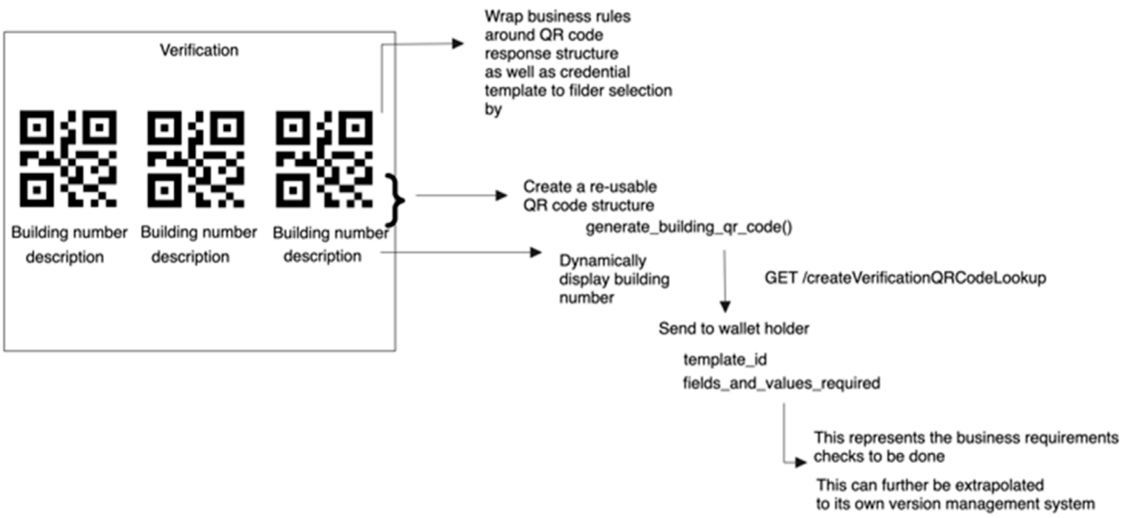

Verification
Building number description
Building number description
Building number description
Wrap business rules around QR code response structure as well as credential template to filder selection by
Create a re-usable QR code structure
generate_building_qr_code()
Dynamically display building number
GET /createVerificationQRCodeLookup
Send to wallet holder
template_id
fields_and_values_required
This represents the business requirements checks to be done
This can further be extrapolated to its own version management system

FIGURE 10
Verification of building number through QR code scanning process flow.

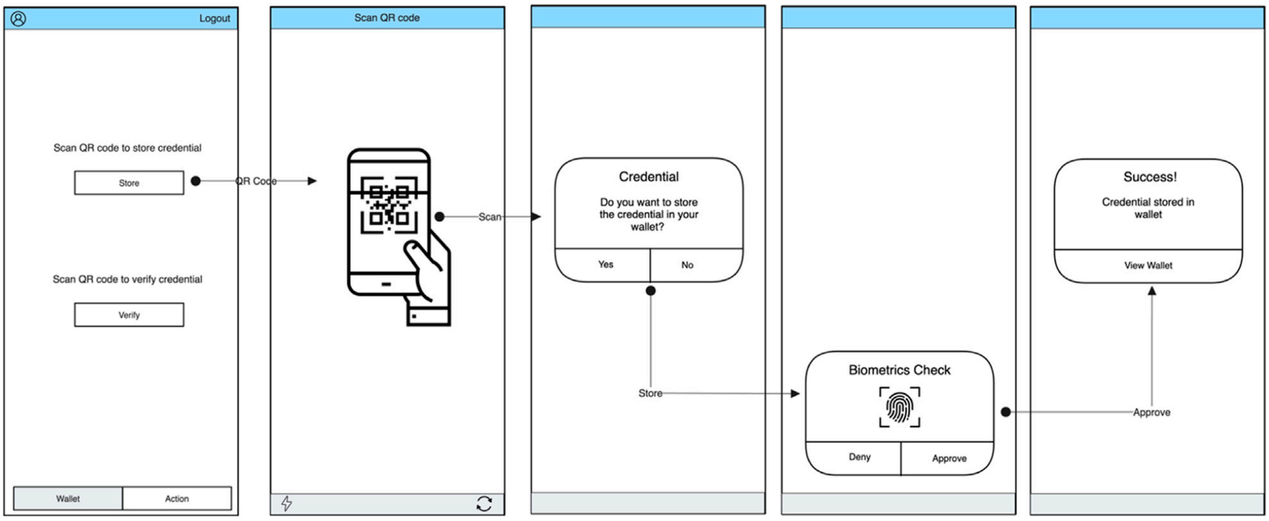

Logout
Scan QR code to store credential
Store
QR Code
Scan QR code to verify credential
Verify
Wallet
Action
Scan QR code
Scan
Credential
Do you want to store the credential in your wallet?
Yes
No
Store
Biometrics Check
Deny
Approve
Approve
Success!
Credential stored in wallet
View Wallet

FIGURE 11
Mobile phone interaction process flow for capturing a credential through a QR code.

use cases, identifying gaps in the existing information database, and compared the process flows of Hyperledger Aries with Trinsic's implementations.

As the digital world evolves, SSI has emerged as a revolutionary concept in identity management that empowers individuals or organizations to control their digital identities. The development of SSI applications offers a unique blend of challenges and opportunities in both open-source and proprietary environments. This paper undertakes an in-depth exploration of these environments, juxtaposing the ease of use of open-source tools against third-party solutions in the SSI industry. The article develops a novel application for public building access and describes the procedures for setting up and interacting with actors within an SSI system.

A critical component of SSI application development is the choice between open-source tools and proprietary or subscription solutions. While open-source tools offer accessibility, flexibility, and community collaboration, they also present obstacles, such as minimal support when faced with technical setup implementation that requires managing the complexity of different versions of prerequisite libraries and dependencies. In contrast, tools and frameworks that provide "software-as-a-service" behind a paywall infrastructure offer a guaranteed response to a support ticket logged, a technical grouping of complex workflows, and easier setup procedures to get started. These benefits can come at a significant financial cost and the potential for vendor lock-in.

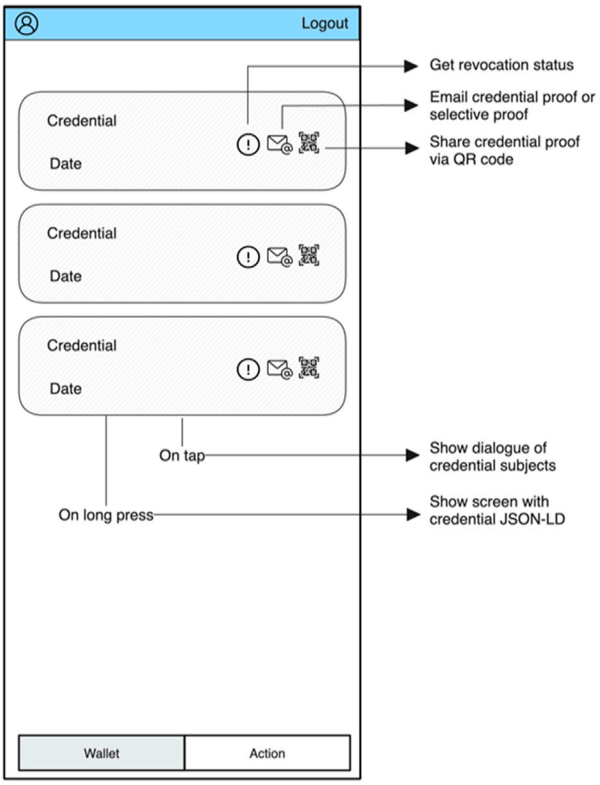

Logout
Credential
Date
Credential
Date
Credential
Date
Get revocation status
Email credential proof or selective proof
Share credential proof via QR code
On tap
Show dialogue of credential subjects
On long press
Show screen with credential JSON-LD
Wallet
Action

FIGURE 12
Mobile application viewing a list of credentials.

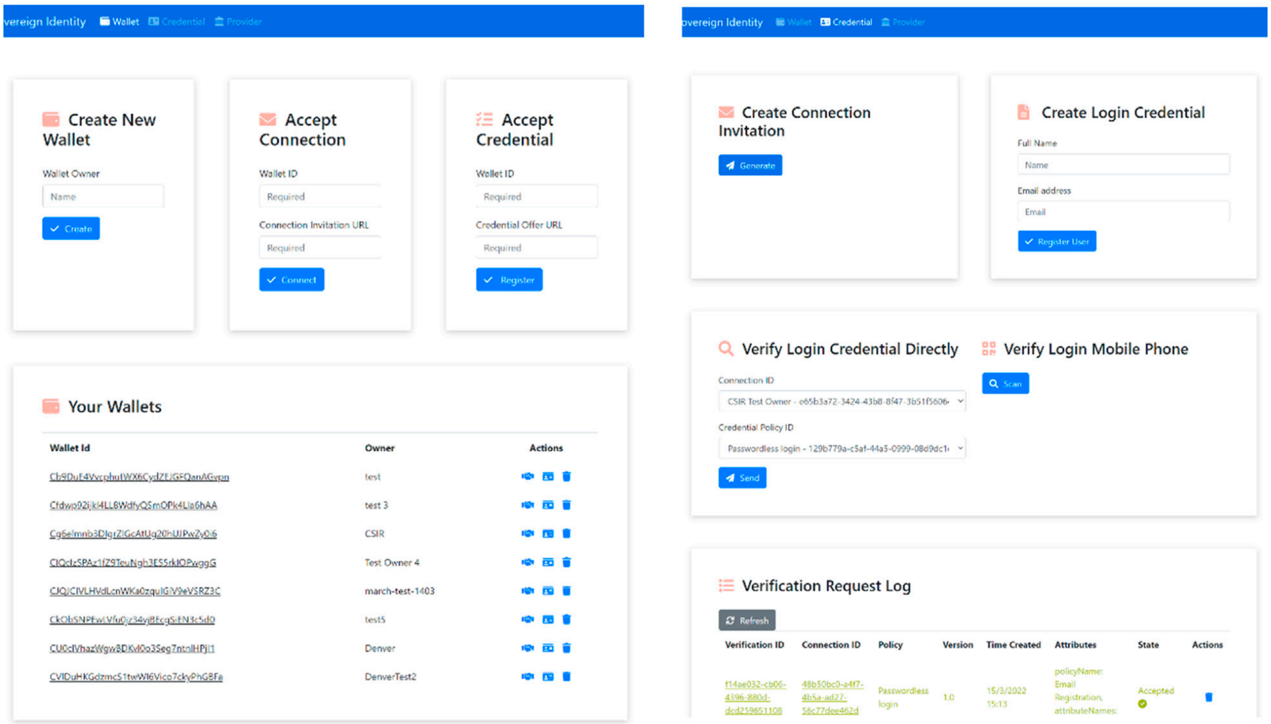
Open-source solutions, while rich in community-generated content, often suffer from information overload, inconsistency, and a lack of clarity in troubleshooting procedures. Developers are often asked to reverse-engineer docker scripts to find libraries not included in setup instructions, further complicating the development process.

The inconsistent locations and repositories for libraries and versions introduce additional complexity in open-source environments. Compounding these challenges is the scenario where, upon installation, libraries are not found either because environment variable names have changed or because they have become obsolete. Changes in newer versions often break the functionality documented for older versions, posing a significant challenge in maintaining and updating SSI applications developed with open-source tools.

Frequently, business terminology is overlaid onto novel technical concepts, which sometimes convolutes the pragmatic implementation. This paper aims to provide context around the complex decision-making process involved in choosing between open-source and proprietary solutions for developing SSI applications, emphasizing the trade-off deficits between ease of use, support, cost, and flexibility. Findings contribute to the understanding of SSI application development and offer insights to developers and organizations in the SSI industry.

2.1 Problem statement

In the context of an ever-evolving technological landscape, the ability to efficiently assess public health risks for building access



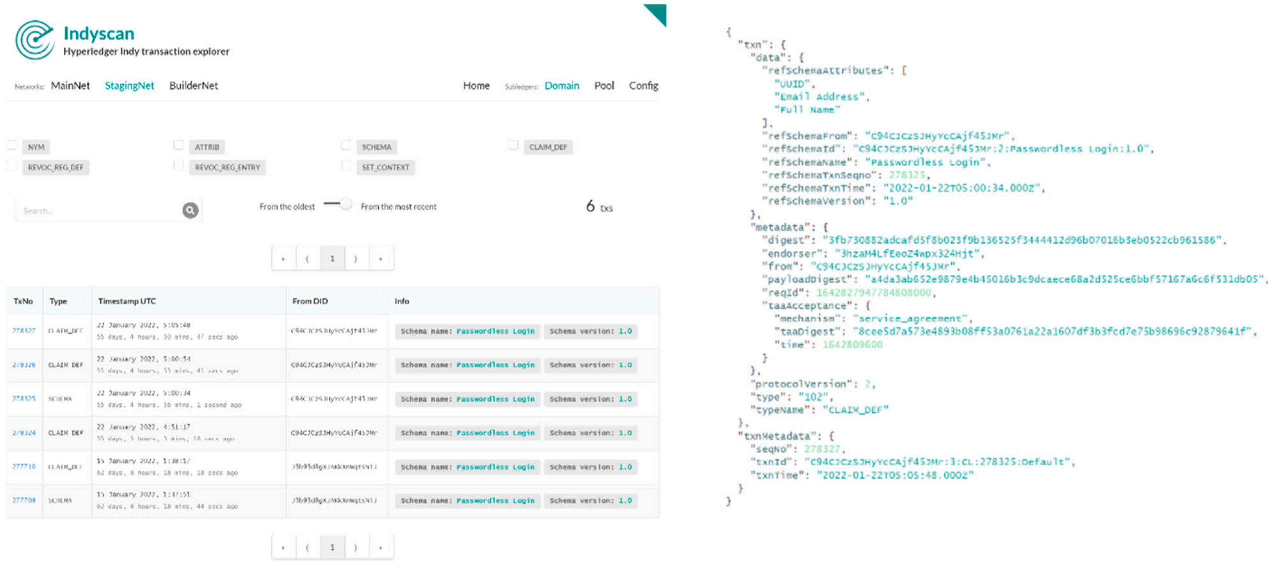
Wallet Id	Owner	Actions
Cb9DuF4VvcphuIWX6CydZEJGFQanAGvpn	test	
Cfdwp92ijkl4LL8WdfyQSmOPk4Lla6hAA	test 3	
Cg6elmnb3DlgrZlGcAtUg20hUJPwZy0i6	CSIR	
ClQclz5PAz1fZ9TeuNgh3ES5rklOPwggG	Test Owner 4	
CJQJCIVLHVdLcnWKa0zqulGiV9eVSRZ3C	march-test-1403	
CkObSNPEwLVfu0jz34yjBEcgSiEN3c5d0	test5	
CU0clVhazWgwBDKvl0o3Seg7ntnlHPjl1	Denver	
CVlDuHKGdzmcS1twWl6Vico7ckyPhGBFa	DenverTest2	

Connection ID: CSIR Test Owner - e65b3a72-3424-43b8-8f47-3b51f5606

Credential Policy ID: Passwordless login - 129b779a-c5af-44a5-0999-08d9dc1

Verification ID	Connection ID	Policy	Version	Time Created	Attributes	State	Actions
f14ae032-cb06-4396-880d-dcd259651108	48b50bc0-a4f7-4b5a-ad27-56c77dee462d	Passwordless login	1.0	15/3/2022 15:13	policyName: Email Registration, attributeNames:	Accepted ✓	

FIGURE 13
Wallet and credential management dashboard.



TxNo	Type	Timestamp UTC	From DID	Info
278327	CLAIM_DEF	22 January 2022, 5:05:48 55 days, 4 hours, 50 mins, 47 secs ago	C94CJCzSJHyYcCAjf45JMr	Schema name: Passwordless Login Schema version: 1.0
278326	CLAIM DEF	22 January 2022, 5:00:54 55 days, 4 hours, 55 mins, 41 secs ago	C94CJCzSJHyYcCAjf45JMr	Schema name: Passwordless Login Schema version: 1.0
278325	SCHEMA	22 January 2022, 5:00:34 55 days, 4 hours, 56 mins, 1 second ago	C94CJCzSJHyYcCAjf45JMr	Schema name: Passwordless Login Schema version: 1.0
278324	CLAIM DEF	22 January 2022, 4:51:17 55 days, 5 hours, 5 mins, 18 secs ago	C94CJCzSJHyYcCAjf45JMr	Schema name: Passwordless Login Schema version: 1.0
277710	CLAIM_DEF	15 January 2022, 1:38:17 62 days, 8 hours, 18 mins, 18 secs ago	J3b93d8gXJAKkXnWqtSNtJ	Schema name: Passwordless Login Schema version: 1.0
277709	SCHEMA	15 January 2022, 1:37:51 62 days, 8 hours, 18 mins, 44 secs ago	J3b93d8gXJAKkXnWqtSNtJ	Schema name: Passwordless Login Schema version: 1.0

```
{
  "txn": {
    "data": {
      "refSchemaAttributes": [
        "UUID",
        "Email Address",
        "Full Name"
      ],
      "refSchemaFrom": "C94CJCzSJHyYcCAjf45JMr",
      "refSchemaId": "C94CJCzSJHyYcCAjf45JMr:2:Passwordless Login:1.0",
      "refSchemaName": "Passwordless Login",
      "refSchemaTxnSeqno": 278325,
      "refSchemaTxnTime": "2022-01-22T05:00:34.000Z",
      "refSchemaVersion": "1.0"
    },
    "metadata": {
      "digest": "3fb730882adcafd5f8b023f9b136525f3444412d96b07016b3eb0522cb961586",
      "endorser": "3hzaM4LfEeoZ4wpx324Hjt",
      "from": "C94CJCzSJHyYcCAjf45JMr",
      "payloadDigest": "a4da3ab652e9879e4b45016b3c9dcaece68a2d525ce6bbf57167a6c6f531db05",
      "reqId": 1642827947784808000,
      "taaAcceptance": {
        "mechanism": "service_agreement",
        "taaDigest": "8cee5d7a573e4893b08ff53a0761a22a1607df3b3fcd7e75b98696c92879641f",
        "time": 1642809600
      }
    },
    "protocolVersion": 2,
    "type": "102",
    "typeName": "CLAIM_DEF"
  },
  "txnMetadata": {
    "seqNo": 278327,
    "txnId": "C94CJCzSJHyYcCAjf45JMr:3:CL:278325:Default",
    "txnTime": "2022-01-22T05:05:48.000Z"
  }
}
```

FIGURE 14
Hyperledger Indy Sovrin staging environment.

becomes imperative. The paper seeks to comprehend the technological advancements in identity systems by juxtaposing conventional centralized federated identity systems against emerging decentralized identity systems. Leveraging both open and closed-source SSI frameworks, the primary focus is to ascertain whether the attribute of privacy is inherently preserved within the decentralized identity model, especially when applied within governmental and private institutional workspaces. The investigation delves into the prerequisites of data to operate such a decentralized identity system while questioning its security robustness. Through the integration of third-party frameworks, the paper aims to highlight the feasibility of employing self-sovereign identity to adhere as much as possible to health and safety regulations through consent mechanisms when granting access to public buildings.

2.2 Research objectives

The primary objective of this research is to design and develop a solution that facilitates adherence to health and safety guidelines in the post-COVID era, specifically for granting access to public buildings. This solution aims to ensure that while individuals' confidential information remains protected, it concurrently supports business service operations that necessitate specific data access.

To achieve this, the following activities will be conducted:

1. Evaluate and compare the capabilities of Hyperledger Aries, Trinsic v1, and, subsequently, Trinsic v2.
2. Examine the migration process from Trinsic v1 to Trinsic v2 to circumvent the challenges associated with running Hyperledger Aries.
3. Address the existing limitation in Trinsic v2 concerning the absence of QR code functionality. This involves the deployment of a temporary location and protocol to delete stored credentials post-usage.
4. Extensively explore open-source and third-party solutions to gauge their ease of use, simplicity in implementation, and adaptability.

2.3 Research methods

This paper adopted the design science research (DSR) paradigm, which serves as a structured approach encompassing literature reviews, prototype development, and evaluation techniques. This methodology was chosen to address the challenges posed during the COVID-19 pandemic, specifically focusing on optimizing the performance of business operations related to health and safety risk assessments to access private building facilities.

A key consideration during the research approach was ensuring user privacy and confidentiality. The aim was to design a system that minimizes the need for extensive personal data while still effectively verifying an individual's identity. The DSR methodology is particularly apt for this paper, as it intertwines theoretical foundations with practical applications, ensuring the derived solutions are both robust in concept and actionable in real-world scenarios.

2.4 Related work

The following section looks at related work that has impacted the SSI field and influenced the current paper. An SSI solution promoted within the healthcare industry focused on safeguarding user data when sharing data between patients and doctors by utilizing a hybrid model of defining access policies with associated role attributes to provide owner consent (De Salve et al., 2023). The claims extracted from different credentials using predicate-based ZKPs enabled preserving the confidentiality

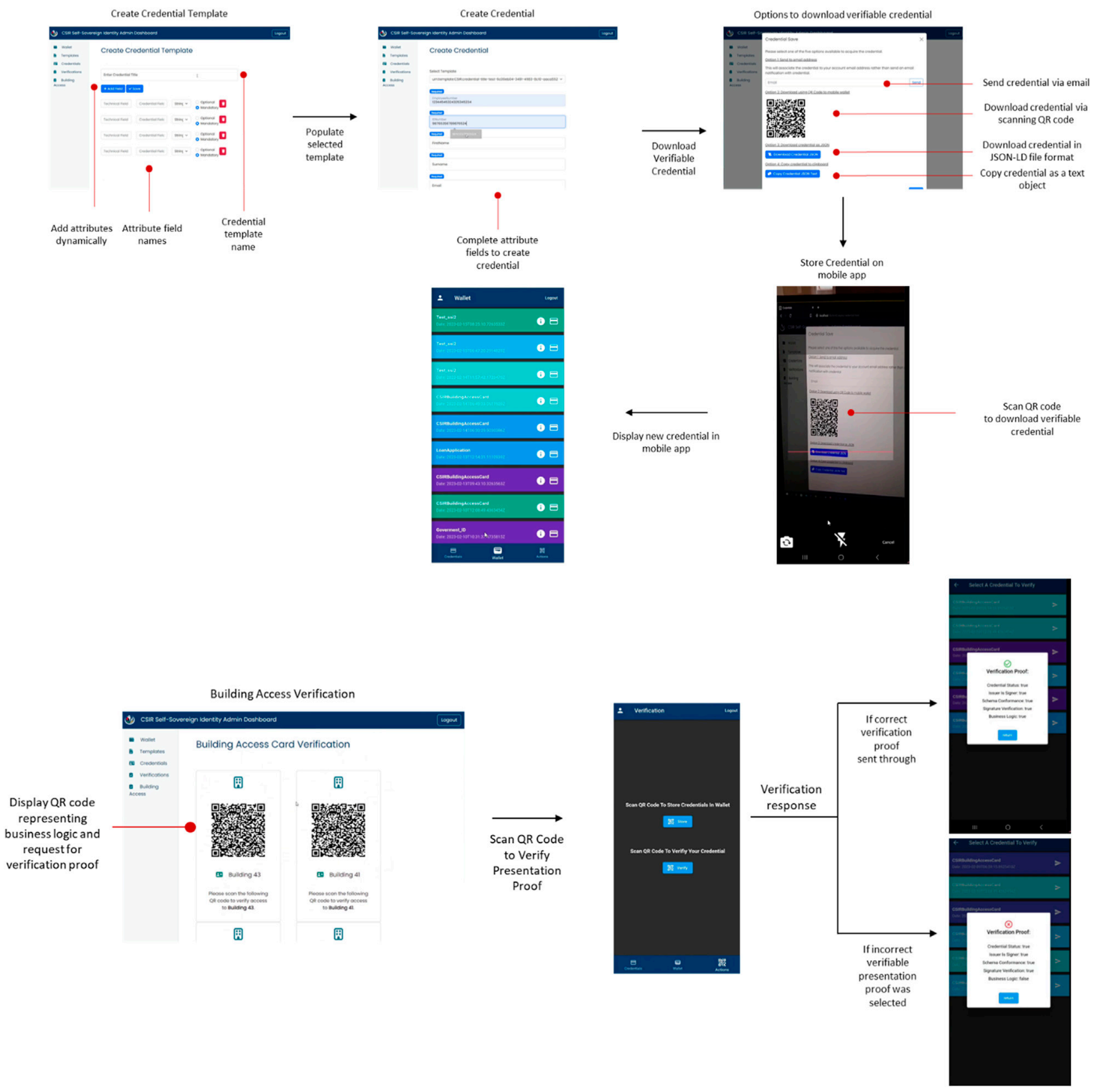

Create Credential Template
Add attributes dynamically
Attribute field names
Credential template name
Populate selected template
Create Credential
Complete attribute fields to create credential
Download Verifiable Credential
Options to download verifiable credential
Send credential via email
Download credential via scanning QR code
Download credential in JSON-LD file format
Copy credential as a text object
Store Credential on mobile app
Scan QR code to download verifiable credential
Display new credential in mobile app
Building Access Verification
Display QR code representing business logic and request for verification proof
Scan QR Code to Verify Presentation Proof
Verification response
If correct verification proof sent through
If incorrect verifiable presentation proof was selected

FIGURE 15
Workflow for credential issuance and building access.

of user data while extracting only the health sensor data required for that service. A different study that looked at forms of trust in the news industry and tackled forms of fake news using a multi-contextual layer approach, scoring the trustworthiness of an issuer using a parameter associated with a public key that enabled wallet holders and verifiers to score an issuer and accept only authenticated credentials through this mechanism (Pujari et al., 2023). A further study looked at centralized identity management providers like Google and Facebook, which store user data in centralized servers and have the risk of being hacked and having that data exploited. By utilizing verifiable credentials enhanced by a layered approach of access control rules (ACRs), web access control (WAC) ontology, decentralized identifier communication (DIDComm) messaging protocol, and Hyperledger Aries Present-Proof protocol, this method attempts to replace the internet identity layer with SSI verifiable credentials using access tokens with validation aspects of trustworthiness rankings (Tan et al., 2023). A study that used SSI to verify student identity documents and transcripts across country borders streamlined the administrative process, thereby reducing the cost of verification methods and creating a standardization utilizing the Europass Digital Credential Infrastructure (EDCI) data model and the ELMO/EMREX XML schema standard when creating a verifiable credential for interoperability (Tan et al., 2023).

The current study approaches SSI in a similar manner to the above-related works by focusing on the privacy of user data as well as the interoperability of verifiable credentials across services. In addition to the above studies, appending the business validation rule set upon the acquisition of a verifiable credential is an issue. This would be sent through during the verification phase via scanning a QR code at the building entry point and mapping the business rule set against the verifiable credential extracted using predicate ZKPs. This would be done automatically without storing any additional data on a centralized server or relying on an administrator to facilitate the validation. The end-to-end logic is encapsulated in the verifiable credential securely stored in a user's mobile wallet, preserving the privacy of the user data.

2.4.1 Building risk assessment

Compliance with regulations while maintaining privacy, especially during the COVID-19 pandemic, requires a multifaceted approach that combines legal adherence, privacy protection, and safety measures (Brostoff et al., 2013). For identification purposes, organizations can provide ID document verification and health credential confirmation from the granting institute without exposing personal details. This process involves advanced identity technology such as LifeID and Bio-ID, which can consolidate multiple SSI accounts in a secure vault (Simpeh and Amoah, 2021).

The introduction of system security personnel to control the site, along with a good record-keeping system, assisted in monitoring visitors and parking (Stockburger et al., 2021). Utilizing digital video conferencing platforms like Skype and Zoom to hold meetings reduced physical interactions and lowered the risk of transmission. Construction companies added measures such as accessible door handles and webcam captures to their existing protocols (Mahula et al., 2021).

Creating awareness among employees and visitors is key and includes educating them about the risks of non-compliance. Non-compliance with regulations can lead to different risk profiles, varying degrees of risk, and even legal implications (Song et al., 2021). Conducting track and trace protocols due to COVID-19 case confirmation became part of a comprehensive safety plan that also required access approval processes that included documenting security date, time, visitor name, ID number, and other details regarding the visit (Gans et al., 2021).

A few construction companies implemented extra measures and developed new protocols to mitigate the spread of COVID-19, which include comprehensive screening, site access rules, careful handling of material, and collaboration with policymakers to inform them about COVID-19 site health and safety measures. Such practices demonstrated the balance required to meet regulatory compliance while protecting privacy and safety and serve as examples for various industries navigating the complexities of operating during a global pandemic (Gans et al., 2021).

2.4.2 South African data privacy regulations

The Protection of Personal Information Act (POPIA) is a South African digital regulation that ensures user data are kept encrypted and private so that they are unreadable without a decryption key and protected from individuals and organizations that might steal data from central organizations. Redistributing the location of where the verifiable credential data are held toward a user's local mobile device displaces ownership of how these data can be exploited according to the consent a user provides. It also removes some of the burden of managing the encryption and storage of these data on a business if they choose to store user data in their central server, thereby having to adhere to POPIA regulations (Brown and Attorneys, 2023). The relevance of where a decentralized identifier document (DID) is stored from a trust registry blockchain location perspective is a topic of further research. This topic was not covered during the exploration and prototype phase as the reliance on a test net infrastructure such as BCovrin, Sovrin, and Trinsic was locked into that specific vendor infrastructure for testing and implementation purposes. Local self-attested DIDs were utilized as entry points for connecting SSI agent wallets to issuers via blockchain test networks during Trinsic v1 testing and a central server for v2, which was blockchain agnostic provided by Trinsic.

2.4.3 Decentralized identifiers

A decentralized identifier document (DID) is associated with a public key that is stored on a public ledger. The public ledger is a decentralized network that provides a way to register and authenticate DIDs. The entity represented by a DID is associated with a private key that is used to decrypt messages and sign digital documents (Naik and Jenkins, 2020). The DID provides a consistent and unique way to refer to an entity or object on a decentralized network. The public–private key pair provides a way to secure and authenticate communications with that entity. Together, they form an essential part of the infrastructure for decentralized systems, such as blockchain.

DIDs are similar to uniform resource locators (URLs) in that they offer a consistent way to refer to a particular entity or object. However, there are some crucial differences between DIDs and URLs. DIDs are designed for use in decentralized systems, like blockchain, while URLs are intended for use in centralized systems, such as the web. Additionally, DIDs are typically associated with a public–private key pair, while URLs are not. DIDs provide a way to uniquely identify an entity or object, while URLs provide a way to access a resource on the web (Tadjik et al., 2022). Figure 1 indicates the benefical attributes associated to DIDs when linking a physical identity to a digital identity.

In federated identity management (FIM), users often sacrifice some privacy as their identity information is shared with the identity provider and potentially other services (Di Francesco Maesa et al., 2023). A DID is an address, whereas a public key is used to read messages signed with a private key.

A DID is essentially a unique identifier that can be registered on a decentralized network such as a blockchain. It provides a way to uniquely refer to a DID that can identify a particular entity or object. This can be used in various contexts, such as identity verification and supply chain tracking.

DIDs enable verifiable credential proofs, allowing users to present only the necessary information for a transaction without oversharing further confidential details. This reduces excessive personal data exposure and enhances user privacy (Di Francesco Maesa et al., 2023). The DID and its associated public–private key pair form a fundamental infrastructure within decentralized systems, contributing to their robustness and trustworthiness. Figure 2 presentst the layers of interaction during the retrieval process of a DID document when a lookup is conducted from a DID registry using a DID method.

2.4.4 Invitation connection protocol

The invitation connection protocol is the first step in connecting two separate SSI agents, establishing a mechanism of trust by using a form of digital handshake protocol. During the following procedures, patterns were identified pertaining to communication of data during transmission to see whether a form of traceability could occur and to identify whether any pattern exposes connective identifiers. This would ensure transparency of the inner workings of the protocol and understanding of the communication flow in establishing trust in an SSI ecosystem between two agents. This is the underlying fundamental protocol in which SSI separates itself from a previous centralized federated identity system.

Each issuer and wallet holder agent needs a separate configuration based on starting parameters that indicate the type of wallet (key, name, type, and local-DID), endpoint, admin, and genesis URL to connect to. An issuer is instantiated with a global DID to be recognized by various authorities and verifiers. This costs a fee to register on a blockchain and is not a process that a wallet holder would require. A wallet holder instead relies upon a local-DID to identify themselves (Jong, 2021).

A separation of practices occurs that depends on the initiation of the request and received messages. An issuer could possibly generate a public or non-public invitation request, enabling a wallet holder to connect to an issuer. A public invitation has no specific endpoint parameters to send the recipient invitation keys to; it instead provides its own recognized global DID to establish trust between the wallet holder and itself (Jong, 2021).

Once the wallet holder receives the invitation, the wallet holder then sends a connection request to the issuer. It would seem this is the reverse direction of the flow of instantiation of the procedure of communication from issuer to wallet holder (Belchior et al., 2020). The wallet holder now becomes the initiator of the request. The issuer then accepts this request from the wallet holder and navigates the various statuses from “creating connection response,” “created connection response,” “connection prompted to active,” and “received connection complete.” An issuer can now send credentials to the wallet holder after creating a credential schema and definition. In Figure 3 a trace flow is conducted to identify whether there is a common pattern of identifiers during the connection handshake to tie each SSI agent (Issuer, Wallet holder) together. This would impact privacy concerns regarding un-traceability or unlinkability if this information were to be stored on the blockchain. The outcome presented shows that each entity has its own identifers and unique connection identifiers during the process and only the invitation key could be used to connect the two entities together although this invitation key attribute is not returned in isolated separate calls when using the SSI framework in Trinsic V1 which is based on Hyperledger Aries to get connection details and status. It is only displayed once during the establishment of the connection.

This outlines the fundamental flow of connecting two SSI agents. The common thread identifier that connects the two agents is the invitation key represented in the diagram below. The other attribute identifiers represent the configuration for each separate channel of communication between the updating of statuses when communicating between each agent. These are held separately from each agent, which is why it is important to highlight the invitation key as a common traceability thread for the short period during which it is used.

A layer 2 communication protocol that is readily available through closed-source software development toolkits (SDK) like Trinsic version 1 and version 2 was used to abstract the process of connecting to the issuer and wallet holder. One can connect initially through public or non-public invitation access points that either specify a direct connection ID or a URL to request an invite from the issuer.

2.5 SSI ecosystems

The following sections will investigate the intricacies in terms of processes, actors, and interactions that represent the fundamental building blocks of an SSI ecosystem. The discussions will briefly stipulate the background for why the versions of the ecosystem were updated and outline the new features where applicable.

2.5.1 Trinsic v1 ecosystem

The setup infrastructure challenges confronted during the implementation setup of Hyperledger Aries Python and Indy tools and libraries were mostly alleviated by using Trinsic v1 to consolidate these tools. The perspective taken was to delve deeper into the intricacies of process flows and sequencing of data necessary for establishing agent connections, securely storing credential data, issuing credentials, and verifying them to solidify the concepts of SSI rather than disrupt the process of creating a potential SSI proof of concept prototype challenged by infrastructure setup.

It was crucial to recognize the architecture of this solution even though it leaned toward being a more custodial solution in nature. This approach, while effective, slightly deviates from the authentic essence of SSI, which fundamentally aspires toward a decentralized wallet application that upholds the principles of non-custodial control. The trade-off in this aspect was strategic, aimed at bridging the gap and creating a functional model for interaction. This diverged from the core ethos of SSI; however, this exploration was driven by a pragmatic understanding of the intricate workings at play in an SSI ecosystem, particularly in contrast to Hyperledger Aries’ standalone setup beyond the confines of its Docker-based demonstrations. Through this careful balance of innovation and pragmatism, Trinsic v1 emerges as an instrumental stepping stone toward the realization of more comprehensive and decentralized SSI solutions in the future. In Figure 4 we provide an overview of the components and theirs interaction in Trinsic V1. This assisted in reducing the overlapping features that would provide the same result although through a different communication channel.

2.5.1.1 Primary system components and connections

1. A wallet represents a secure digital location where credentials and verifications can be stored and managed.
2. A connection is a linkage or network bridge between two agents or entities that could be established during agent setup procedures or instantiated through public or non-public invitations.
3. Credentials and verifications are digital attestations or proofs concerning a specific claim or set of claims about an entity.

2.5.1.2 Connection types

To emulate a primary use case end-to-end, the processes of wallet creation, credential issuance, and verification operate through varied communication channels. The most direct method entails the use of connectionless properties for verification, credential issuance, and invitation to connect (Docs.trinsic.id, 2023).

Discrepancies between connection types:

1. A direct connection between two agents requires an initial agent configuration setup to generate a connection ID that is used as a parameter to initiate a subsequent new connection with a separate SSI agent.
2. A connectionless invitation is independent of any prior established connections. This can be publicly available and has unlimited utility. It provides the option to connect with an issuer or store a credential without any direct linkage to a connection ID.

Connectionless invitations streamline the process of forming a bond between two agents. They negate the need to undergo detailed confirmation steps, as described in the Hyperledger Aries protocol.

2.5.1.3 Basic interaction procedure

1. Wallet creation is initiated by specifying an identifier and a holder name. The metadata aliases captured here are decorative in nature for display purposes, separate from true unique identifiers that are autonomously generated in the background.
2. Credential schema and issuance have no distinct issuer roles or wallet holder types associated with them by Trinsic at the time of writing. Any entity with application programming interface (API) access could execute these functions, such as creation, revocation, or viewing different ecosystem components. To establish a credential or its schema, one only needs to identify a credential ID and a schema ID. A credential schema contains an array of attribute field names that are all string data types.
3. A credential definition could be generated afterward that would be populated by values associated with string attribute fields outlined by the credential schema created previously.
4. A credential would be distributed utilizing either a connection ID originating from the initial agent setup or a connectionless method using a URL.
5. The wallet holder may consider accepting the credential with their wallet ID and connection ID or utilize a connectionless method to store the credential data associated with a specific wallet ID.
6. A credential deletion may occur by using the credential ID as a lookup parameter.

In Figure 5 we provide an overview of the entire process flow for interacting with each individual agent in Trinsic V2. We focused on emphasizing the credential verification proof generated as the outcome of the entire procedure.

2.5.1.4 Leaning toward centralization

Despite the abstraction levels provided by Trinsic v1, the ledger visibility is not always directly transparent to developers when interfacing with API calls presented by Trinsic v1. The parameters for input vary based on the use case: providing connection ID or connectionless values. The system's tangibility remains focused on a form of centralized data storage. The inconsistency in data reflection times occurred when using blockchain test net storage, which produced delays in acquiring data response feedback from the ledger when querying credential availability, data, and status. The interaction here with Trinsic indicated that the option to consider alleviating these concerns would be further developing a centralized server solution to ensure customers have fewer production downtime response incidents from blockchain test net dependencies. The variability in data availability at times made this option less appealing for production environment applications.

2.5.2 Trinsic v2 ecosystem

Part of our journey was to see how much a verified credential is relied upon to work on its own without interference from a centralized authority. To understand this issue, we migrated from Trinsic v1, which contained ZKP and examples of QR codes, to Trinsic v2, which had an adaptation of selective disclosure to compensate for the lack of ZKP at the time of writing this paper (Docs.trinsic.id, 2023). The migration required handling the different process flows and considering how to automate testing these calls whilst validating the TypeScript examples in the docs. To balance the notion of usability and security, we considered whether certain features hindered the user from returning to use a service even though they add a layer of security and what would be the extra steps required that make a process cumbersome to utilize for a user.

2.5.2.1 Novel system business use case

The vaccine demo from Trinsic v2 was used to understand how components fit together and how core information was exchanged by de-structuring the demo into reusable components and providing clarity, whereby the documentation at the stage only provided a rough outline of APIs without sequencing how to utilize the services in an entire ecosystem. The vaccine demo contained the components to implement a one-time PIN account registration and issue an access token that provides a session key to manage user interaction for creating credential templates and issuing and receiving verifiable credentials. The separation of steps when verifying and creating verifiable credentials provided a form of compartmentalization to enable an exchange of transactions without exposing a credential to potential theft. In Figure 6 we provide an overview of a use case to issue a vaccination credential using Trinsic V2 to provide a cohesive technical overview of the interaction between data and methods required to parse real world data to generate a credential verfication proof using Trinsic V2.

2.5.2.2 New user creation

This is a distinct process from the normal procedure of online registration, whereby a formal account registration process would collect user data to validate who they are. Instead, the email address and one-time PIN (OTP) become the means to log in and create a session using an auth token as both a login and registration procedure. The auth token is used in subsequent user calls when verifying and storing a credential against a wallet. The issuing of a credential is done using the ecosystem auth token wherever possible, which is provided by Trinsic upon organization registration. As of the time of writing this paper, Trinsic had yet to define roles related to wallet holder, issuer, and verifier. Therefore, every user can technically utilize all the calls provided by Trinsic as an issuer, verifier, and wallet holder. An organization implementing the SSI tool then takes on the responsibility to ring-fence and provide certain parameter restrictions and prerequisites when conducting REST API calls for certain operations using Trinsic v2.

The following list identifies the benefits of the authentication scheme (Docs.trinsic.id, 2023):

1. It uses Trinsic's ZKP based on the "*Oberon Auth Scheme*" to issue multi-factor capable tokens and prove their validity without disclosing the tokens.
2. Endpoints need only store a single public key and not any tokens.
3. An attacker who breaks into the server does not have any password/token files to steal and would only see a public key.
4. The proof of token validity is only 256 bytes, while the token itself is only 48 bytes.

A challenge value that is in the form of an array of numeric values provided during the SDK call "*account login*" is then passed through with a "*one-time PIN*" that is received via a user's email inbox. The OTP is inserted manually on the front-end side and sent through to the server in an API request called "*register account*," which forms the main point of contact to register and log into an account automatically without the need to capture the private details of a user during registration. This process aligns with the principles of SSI to some degree for privacy. Figure 7 provides a simple overview indicating key steps involved to acquiring a one-time-pin to login through the Trinsic V2 framework.

2.5.2.3 Create credential

The following options are provided after the issuer creates a credential. The following describes how each method operates for the user to capture the credential.

1. Store the credential against an email address: storing the credential in a mobile wallet using the user's email account as a transfer medium. This operates using the email address registered when logging in to the system. The issuer can insert an email address manually, and the user will automatically receive the credential. If a user does not exist on Trinsic, they will not be able to receive the credential. A user simply needs to log into either the mobile application or the desktop application to automatically create a wallet using the OTP.
2. Download the credential through a QR code: this is represented as a URL in an image source tag referring to an API lookup called "*get credential with lookup ID*," which returns a JavaScript Object Notation for Linked Data (JSON-LD) credential when queried. Once this credential is stored in a user's mobile wallet, it is then deleted based on the procedure implemented in the wallet app. This credential is temporarily stored in a lookup table for the QR code to function, as there are size limitations when displaying a QR code with huge amounts of data.
3. Copy the URL of where the credential is hosted to acquire the credential: simply provide a URL instead of a QR code that contains the same URL. This is used in circumstances where a web wallet is used, and it is easier to copy a URL than to use a QR code.
4. Copy the specific section in JSON-LD into wallet storage: instead of using either a URL or a QR code, one could just copy the JSON-LD text and store that wherever they like; this could be a security concern as if stored in a non-secure location that could be opened by someone else fraudulently posing as the person using this credential.

Using QR codes is a potential security risk if they are displayed too long on the screen because someone else might capture the credential with their mobile phone. The QR code is intended to be used momentarily with supported applications that can remove the credential ID from a lookup table once the credential is captured.

A verifiable credential (VC) document can be stored on any device. Its traceability is not linked or attached to a user or device at this point in time of the SSI development ecosystem landscape, although future work is being conducted to implement this. On 3 April 2023, Trinsic confirmed they are looking at binding the credential to a device so that it may not be transferrable. This might operate similarly to how hashing works in a blockchain, whereby the details of the device are linked to a verifiable credential (Docs.trinsic.id, 2023).

List of steps for the QR code lookup procedure for credential issuance:

1. "*Create credential*" is a core Trinsic SDK call provided to consume data values to generate a JSON-LD credential.
2. "*QR code lookup ID*" is sent back. This is an expanded database proxy table created outside of Trinsic.
3. "*Create QR code image*" is displayed on the front-end dashboard for the mobile wallet application to scan and download credentials.
4. "*Get credential with lookup ID*" is a database proxy table that facilitates the retrieval of a credential from a scanned QR code.
5. Download the credential by scanning a QR code from a mobile wallet application.
6. "*Insert into wallet*" is a fundamental Trinsic SDK call that is used to store data.
7. "*Delete credential with lookup ID*" utilizes a database proxy table as a form of data cleanup that ensures there is no leakage of credentials to external entities.

Figure 8 provides a wireframe mockup of the front-end user interface, depicting the different methods to download a credential.

2.5.2.4 Create credential template

This credential template screen is the starting point before creating an actual credential. A template is essentially a skeleton or a scaffolding that provides placeholder fields for when an administrator or an issuer decides to fill in these fields at a later stage to create a credential. The aim of this procedure is to be dynamic in creating a set of multiple user interface (UI) elements, such as field name, description, data type, and optional fields, and assigning event handlers dynamically during runtime to either delete or update a value associated with a UI element. The end result is to group all of these data and present them in a manner suitable for the backend SDK call to Trinsic to process. There is a separation between what the UI captures from the user and how Trinsic prefers the data to be presented in their TypeScript (SDK) for Trinsic V2.

2.5.2.5 Verify credential

This process describes how to transfer an SSI credential utilizing a QR code by implementing out-of-band business checks that pass through via QR code whilst also filtering the lookup of the credential template.

When a QR code is created, a request is made to the backend database to create a lookup ID for the QR code, storing the template ID, date and time, fields and values required for business rules validation, as well as presenting a status of pending or complete to notify the user when the verification is done. A trail log is displayed to notify a user that a verification has been done if a read request has been conducted to confirm whether the status has been updated to "completed." The business rules provided from the front end will be extrapolated to the back end, whereby business rules could be expanded upon and maintained by various institutions. Figure 9 provides an overview of the end-to-end business procedure to verify a credential indicating the position at which credential subject values are extracted and verified a long side credential proof.

The procedure of how a verification is executed is described below:

1. The request received by the user is in the form of a template object alongside a "lookup ID" and fields that will be required to conduct business verifications at the end of the procedure.
2. The "template field" in the request queries allows the user to specify this structure in their credential proof with or without attribute values.
3. If the user decides to provide a credential proof without values, this will produce an invalid status response when conducting the business verification checks toward the last stage of the procedure.
4. Once the user has processed this request, they will send through their "credential proof" with a "QR code lookup ID" to be validated alongside a business rule check.
5. After the verification proof procedure has passed the validation steps, the status of the record in the verification table is updated to complete.
6. Additional verification related to trust registry governance checks is executed.
7. The entire verification procedure is grouped together into one API call, and a list of status checks is returned to the user upon completion.

Figure 10 depicts the sequence of steps to that occur during the verification process once a QR code has been scanned.

2.5.2.6 Wallet

A mobile app was developed to easily store and interact with servers to verify credential proofs. This process will be conducted utilizing QR codes as a medium of interaction. A secure encrypted *hive* database was used on Android to store credentials. Mobile phone biometric authentication was used to simplify login and consent authentication permissions, thereby increasing usability, which would previously have required a *one-time PIN* to log in for session management to instantiate. The app was built using *Flutter*, which communicated to a server written in *TypeScript*, which facilitated the *Trinsic v2 SDK* interactions, thus providing a form of reusability across multiple front-end frameworks for web and mobile development. Figure 11 presents a process flow of the front-end mobile application during the storing a credential into a smart wallet.

2.5.2.6.1 Login authentication. New users of the mobile application will be greeted with an onboarding guideline to describe the functionality of the application. There are two primary options for accessing the mobile application. The first is an *OTP* that is emailed to the user. The second operates after the OTP has been first utilized, streamlining the process to use local biometric identification thereafter, which has cached the session management key.

2.5.2.6.2 Hive localized mobile storage. The authentication token for session management is stored in a *hive* table. User fingerprint biometrics were utilized to retrieve this token from the *hive* database. Flutter secure storage was utilized to contain the encryption key. AES was utilized to encrypt and decrypt data from hive tables.

2.5.2.6.3 View credentials. The wallet tab provides the user with a holistic view of all the credentials stored in their wallet. The card detail widget provides three primary options to confirm revocation status, email a credential proof, and share the full credential proof via QR code. Further details of a credential can be displayed by using the gesture "on tap," displaying all credential subject's fields and values, whereas "on long press" provided the credential in plain text JSON-LD format for distributing manually as an alternative method of transferring credential proof. Figure 12 provides an overview of the mobile front-end view credential screen and the functionality one can conduct against each credential such as sharing and acquiring status.

2.5.2.6.4 Store credentials. When the user scans a QR code using the QR code scanner, the result obtained includes the API call and the "*lookup ID*." The "*lookup ID*" is used to obtain the corresponding JSON-LD credential from the proxy database. A credential stored will go through biometric verification. Any previous data used during the steps to temporarily capture credential data are removed from the central server, represented on the proxy database table, to ensure no leakage of private data occurs.

2.6 Results

2.6.1 Use case

This article demonstrates two approaches to evaluate the solution. The first uses Trinsic v1, and the second uses Trinsic v2. This will demonstrate the fundamentals and principles of SSI technology to execute an end-to-end interaction between the wallet holder, issuer, and verifier. The connection establishment between two agents, the exchange of these credentials, and the verification proofs thereof represent the core interaction in an SSI system.

In this example, a public access system utilizing SSI infrastructure, dependent on third-party tools and frameworks to provide access to local facilities, displaces the need to share private information upon each request that requires disclosure of personal identity information with a risk assessment health questionnaire form. The following diagram depicts the legacy use case compared to the suggested SSI replacement.

2.6.2 Trinsic v1 solution

An SSI web application dashboard was created to demonstrate capabilities through the interaction use case of an issuer and wallet owner connecting and verifying the sent credentials. Figure 13 represents an overview of the Trinsic V1 dashboard to issue, store and verify credentials as a proof of concept to demonstrate the fundamentals of SSI functionality in practice. The framework used to create this dashboard is Trinsic SDK, which is a tool that provides a layer of access to Hyperledger Aries and Hyperledger Indy to the Sovrin Staging network. Hyperledger tools are developed by IBM and are open-source tools to demonstrate SSI capabilities. Support was provided by Trinsic for accessing these tools, ensuring the infrastructure setup functions correctly. Initial endeavors were to set up Hyperledger Indy and Aries as separate instances, but due to the nature and volatility of open-source tools, these attempts failed. No resolution could be found to circumnavigate these errors through a lack of skills and support from the issues posted on the GitHub page and direct communication to the open-source team (GitHub, 2021). Trinsic provided this support, and a prototype was developed through its API. Some of the operations included in the dashboard are

1. Create a wallet by instantiating a local authenticated object profile to be able to store verifiable credentials.
2. Create an invitation utilizing “create connection” for the wallet from the issuer.
3. Accept an invitation by inserting a wallet ID and a connection ID from the wallet owner side.
4. Create a login credential from the issuer side to insert user details for the credential.
5. Accept the credential on the wallet owner side—insert credential via “offer URL” and “wallet ID.”
6. View the credential on the wallet owner side to display details captured to the user.
7. Verify the login credential from the issuer side to decide whether to select connection ID and policy ID to display the request.
8. View verifications from the wallet owner side and decide whether to accept the verification conducted.

Two main users form part of the system: an issuer and a wallet owner:

1. An issuer can create credentials to send to wallet holders and verify whether those credentials being held are accurate via ZKP.
2. A wallet holder can store credentials sent through from an issuer as well as respond to verification requests from an issuer.

A third system entity that is not involved in day-to-day interactions is a provider that symbolically represents an organization. Wallet holders and issuers conduct actions against each other using the API key that the provider provides them.

The functionality presented from the issuer side:

1. Create a connection invitation in the form of a QR code or a URL.
2. Create a credential; in this use case, create a login credential.
3. Verify a credential that a wallet holder might utilize a connection ID and a wallet ID or connect through a QR code that is represented as a connectionless verification check.

Figure 14 represents the stored credential schema on Sovrin Staging Net blockchain from Trinsic V1 through the online IndyScan blockchain explorer.

2.6.3 Trinsic v2 solution

The objective was to develop a comprehensive credential management system, encompassing both desktop and mobile platforms, designed to enhance user awareness during the consent process. This system aims to make users more discerning about the nature of the data they are sharing, their intended utilization, and the ultimate objectives of such sharing. It remains the prerogative of individual enterprises to adhere to legislative guidelines pertaining to user data management, whether such adherence involves retaining such data for routine operations or merely ascertaining the authenticity of a credential to facilitate a service. This initiative seeks to address a prevailing challenge in today’s digital landscape: the ubiquitous and often redundant collection of user data has led to a diminished sense of its potential misuse. Such indiscriminate accumulation can facilitate malicious actors’ fraudulent endeavors or enable extensive user behavior profiling during large-scale data analyses. This unchecked data acquisition and its potential misuse have, regrettably, fostered a pervasive sentiment of mistrust among users. Many users now approach online platforms and services with caution, wary of the possibility that their personal information might be commodified and traded to marketing agencies or major corporations for monetary gain.

The subsequent process flow in Figure 15 describe the interactions between the web and mobile application, serving as a representation of the technical demonstrator. The administrative workflow is segmented into four distinct phases: the OTP authentication process, template establishment, credential formulation, and the verification protocol. Prior to the issuance of any credential, it is imperative that a template be established, providing a foundational framework upon which specific values can be assigned when bestowing a credential. The verification segment of the demonstration emphasizes the deployment of a QR code, serving as an access point for the SSI mobile wallet’s scanning functionality.

The mobile application was structured according to three distinct sections of user experience: the authentication process, the credential storage mechanism, and the verification protocol. The mobile application’s login procedure mirrors that of its desktop counterpart, augmented with the provision for utilizing local biometrics and facial recognition for authentication post-initial registration. Upon gaining access, users can seamlessly browse all their stored credentials within their digital wallet and have the liberty to execute both storage and verification operations. Subsequent steps delineate the method by which the mobile wallet captures credentials displayed on the desktop via QR code integration and secures them within the user’s digital repository. Pertaining to the verification segment of the demonstration, when users employ the QR code to gain entry into a facility, their mobile wallet intuitively filters the credentials to match the pertinent verification type, facilitating a streamlined selection of the appropriate access card. The server side was implemented with a four-tiered verification procedure comprising out-of-band business validations, schema verifications, revocation assessments, and trust registry governance evaluations.

3 Discussion

This section raises several crucial questions about the proposed solution. These include issues related to version management of business rulesets, reducing dependence on third-party architectures, and QR code usage. It also questions the interoperability of credentials and challenges the prevailing model of storing user data to access services. Furthermore, the viability of a business trusting an issuer credential over conducting further out-of-band business verification checks is examined. Questions surrounding the efficiency and impact of the Identity Overlay Network (ION) for decentralized identifier management, business change, consent and credential management, and General Data Protection Regulation (GDPR) compliance in the context of credentials are also posed. Security risks associated with handling verifiable credentials, including storage, transfer, and protection against theft and spoofing, are addressed. The discussion also delves into the operation of the DID communication protocol, its utility, and how it interacts with other platforms like Trinsic. Finally, there is a reflection on the recommendation of SSI for future products, acknowledging the current solution's imperfections, such as architectural and protocol ambiguities, a lack of set standards, and skepticism about the security of these systems. The paper concludes with concerns about the reluctance of businesses to adopt SSI principles due to the perceived absence of direct capital market gains and the need for a business model change instead of merely creating a security adapter.

When focusing on the adoption of Self-Sovereign Identity (SSI) and related technological innovations, the reluctance of businesses to embrace SSI principles, to the perceived lack of direct financial incentives, requires the necessity for a fundamental shift in business models. The concern is not just about adopting a new security technology but about the broader implications for how businesses operate and generate value in a digital-first economy.

One of the core considerations explored is the practicality and trustworthiness of businesses relying on issuer credentials. This examination brings to light the challenges associated with foregoing additional out-of-band verification processes, which raises questions about the efficacy and security.

The following components are integral to enhancing the interoperability of credentials and addressing the complexities of securely storing user data for accessing services. The nuanced management of business rulesets through version control, the imperative to minimize reliance on third-party architectures, and the innovative use of QR codes to streamline operations.

The paper reflects on the recommendation of adopting SSI principles for future digital products and infrastructures. It acknowledges the current limitations of SSI solutions, including architectural and protocol ambiguities, the absence of universally accepted standards, and prevailing concerns regarding the security of these systems. These issues underscore the nascent state of SSI technology and the hurdles that must be overcome to achieve widespread adoption and trust.

4 Conclusion

SSI frameworks provide the ability to authenticate the source of a credential using cryptographic proofs such as BBS+ (Dan Boneh, Xavier Boyen, and Hovav Shacham) proof of signature. One of the main benefits of SSI is the relocation of where user data are stored, separating it from a centralized third-party identity management provider and storing VC data directly on a user's mobile phone. This alleviates the legality issues associated with centralized database hacks and exploitation of user data through marketing and behavior profile tracking confronting third-party identity management providers. Further exploration into offline verification utilizing only cryptographic signature proofs as a form of trust will provide further insights into privacy-enhancing capabilities for users.

5 Project

https://github.com/dnaicker/admin_dashboard, https://github.com/dnaicker/hapi_typescript, no restrictions for non-academic use, all platforms (Windows, Mac, and Linux).

Data availability statement

The original contributions presented in the study are included in the article/supplementary material; further inquiries can be directed to the corresponding author.

Author contributions

DN: writing–original draft and writing–review and editing. MM: software and writing–original draft.

Funding

The authors declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors, and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

Belchior, R., Putz, B., Pernul, G., Correia, M., Vasconcelos, A., and Guerreiro, S. (2020). "SSIBAC: self-sovereign identity based access control," in 2020 IEEE 19th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Guangzhou, China, 29 December 2020 - 01 January 2021. doi:10.1109/TrustCom50675.2020.00264

Brostoff, S., Jennett, C., Malheiros, M., and Angela Sasse, M. (2013). Federated identity to access e-government services. Available at: http://dx.doi.org/10.1145/2517881.2517893.

Brown, T., and Attorneys, V. (2023). Unpacking personal information, lexis nexus. Available at: https://www.lexisnexis.co.za/__data/assets/pdf_file/0007/790540/POPIA_Unpacking_personal_information_pdf_8SCGqr0q.pdf.

De Salve, A., Di Francesco Maesa, D., Mori, P., Ricci, L., and Puccia, A. (2023). A multi-layer trust framework for self-sovereign identity on Blockchain. *Online Soc. Netw. Media* 37–38, 100265. doi:10.1016/j.osnem.2023.100265

Di Francesco Maesa, D., Lisi, A., Mori, P., Ricci, L., and Boschi, G. (2023). Self sovereign and blockchain based access control: supporting attributes privacy with zero knowledge. *J. Netw. Comput. Appl.* 212, 103577. doi:10.1016/j.jnca.2022.103577

Docs.trinsic.id (2023). Overview - documentation. Available from https://docs.trinsic.id/reference/ (Accessed December 13, 2023).

Gans, R. B., Ubacht, J., and Janssen, M. (2021). Self-sovereign identities for fighting the impact of COVID-19 pandemic. *Digital Gov. Res. Pract.* 2 (2), 1–4. doi:10.1145/3429629

GitHub (2021). No ledger available when not using "--wallet-type indy" for agent startup Issue #1398 hyperledger/aries-cloudagent-python. Available at: https://github.com/hyperledger/aries-cloudagent-python/issues/1398.

Jong, L. (2021). Becoming a hyperledger aries developer - getting started. Laurence de Jong. Available at: https://ldej.nl/post/becoming-a-hyperledger-aries-developer-getting-started/.

Mahula, S., Tan, E., and Crompvoets, J. (2021). "With blockchain or not? Opportunities and challenges of self-sovereign identity implementation in public administration," in DG.O2021: The 22nd Annual International Conference on Digital Government Research. doi:10.1145/3463677.3463705

Naik, N., and Jenkins, P. (2020). "Governing principles of SSI applied to blockchain enabled privacy preserving identity management systems," in 2020 IEEE International Symposium on Systems Engineering (ISSE), Vienna, Austria, 12 October 2020 - 12 November 2020. doi:10.1109/ISSE49799.2020.9272212

Pujari, C., Muniyal, B., B, C. C., Rao, A., Sadiname, V., and Rajarajan, M. (2023). Identity resilience in the Digital Health Ecosystem: a key recovery-enabled framework. *Comput. Biol. Med.* 167, 107702. doi:10.1016/j.compbiomed.2023.107702

Simpeh, F., and Amoah, C. (2021). Assessment of measures instituted to curb the spread of COVID-19 on construction site. *Int. J. Constr. Manag.* 23, 383–391. doi:10.1080/15623599.2021.1874678

Song, W., Zaeem, R. N., Liau, D., Chang, K. C., Lamison, M., Khalil, M., et al. (2021). "Self-sovereign identity and user control for privacy-preserving contact tracing," in IEEE/WIC/ACM International Conference on Web Intelligence. doi:10.1145/3486622.3493914

Stockburger, L., Kokosioulis, G., Mukkamala, A., Mukkamala, R. R., and Avital, M. (2021). Blockchain-enabled decentralized identity management: the case of self-sovereign identity in public transportation. *Blockchain Res. Appl.* 2, 100014. doi:10.1016/j.bcra.2021.100014

Tadjik, H., Geng, J., Jaatun, M. G., and Rong, C. (2022). "Blockchain empowered and self-sovereign access control system," in 2022 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Bangkok, Thailand, 13-16 December 2022. doi:10.1109/CloudCom55334.2022.00021

Tan, E., Lerouge, E., Du Caju, J., and Du Seuil, D. (2023). Verification of education credentials on European Blockchain Services Infrastructure (EBSI): action research in a cross-border use case between Belgium and Italy. *Big Data Cogn. Comput.* 7 (2), 79. doi:10.3390/bdcc7020079

Tech, D. (2023). The power of DIDs #1: DID resolution. Danube Tech. Available at: https://medium.com/danube-tech/the-power-of-dids-1-did-resolution-d1d994130319.