Identity Cloud Agent APIs

This document describes the Identity Cloud Agent APIs to be developed. Those APIs will simplify the use of Identus, so that external programs can plug into the Identity Cloud Agent, and benefit from DIDs and Verifiable Credentials via a simple integration.

General Notes:

- Most endpoints require API key authentication.
- Response times expected to be under 400ms for all endpoints.
- All responses are expected to be in JSON format.

Wallet APIs

This API suite covers wallet management (for cryptocurrency or blockchain applications) and a PRISM agent system that handles verification policies, schema registries, and event webhooks. The PRISM agent part relates to decentralized identity and credential management.

1. **POST /api/v1/wallet/mnemonic**
   - Purpose: Generates a new wallet mnemonic
   - Authentication: None required
   - Response: Returns a JSON object containing a random mnemonic and seed
   - Expected Status: 201 (Created)

2. **GET /api/v1/wallet/{{mnemonic}}**
   - Purpose: Retrieves wallet information based on a provided mnemonic
   - Authentication: None
   - Parameters: mnemonic - The wallet's mnemonic phrase
   - Response: Returns a JSON object with wallet details including public address, private key, and seed
   - Expected Status: 200 (OK)

Communication APIs

This API suite focuses on managing connections within a PRISM agent system, with different authentication keys for different subsystems or access levels.

3. **GET /api/v1/p2p**

- Purpose: Retrieves a list of connections between Alice and Bob (from Alice)
- Authentication: Alice's API key
- Response: Returns a JSON object
- Expected Status: 200 (OK)
- Performance: Expected to respond within 500ms

4. **GET /api/v1/p2p/{{id}}**

- Purpose: Retrieves a specific connection maintained for Alice and a third party
- Authentication: Alice's API key
- Response: Returns a JSON object
- Expected Status: 200 (OK)
- Performance: Expected to respond within 500ms

5. **POST /api/v1/p2p/invite**

- Purpose: Create a connection invitation, handling out-of-band invitations.
- Authentication: Alice's API key
- Parameters:
    - from: to generate a nicely formed message
- Response: Returns a JSON response.
- Expected Status: 200 (OK)
- Performance: Expected response within 500 milliseconds.

6. **POST /api/v1/p2p/accept**

- Purpose: For accepting a specific connection invitation.
- Authentication: Alice's API key
- Parameters:
    - invitation: the encoded invite
- Response: Returns a JSON response.
- Expected Status: 200 (OK)
- Performance: Expected to respond within 500ms

Entity APIs

This API suite is focused on managing identity entities within a system, allowing for their creation, retrieval, listing, and deletion. The entities are associated with cryptocurrency wallets or blockchain identities. All those APIs require an Admin authentication.

7. GET /api/v1/identity/entities

- Purpose: Retrieves a list of identity entities
- Authentication: Requires an admin API key
- Response: Returns a JSON object containing information about identity entities

8. GET /api/v1/identity/entity/{{entity}}

- Purpose: Retrieves information about a specific identity entity
- Authentication: Requires an API key
- Parameters: entity - An identifier for the specific entity
- Response: Returns a JSON object with details about the specified entity

9. POST /api/v1/identity/entity

- Purpose: Creates a new identity entity
- Authentication: Requires an API key
- Content-Type: application/x-www-form-urlencoded
- Parameters:

    - name: A string identifier for the entity (e.g., "test-1")
    - role: the role of this entity (caller, worker, verifier, admin)
    - mnemonic: A mnemonic phrase (likely for wallet generation)
    - id_wallet: optional, id of the existing wallet (will not use mnemonic in this case)
- Response: Returns a JSON object with the created entity information
- Expected Status: 201 (Created)

10. DELETE /api/v1/identity/entity/{{...}}

- Purpose: Deletes a specific identity entity
- Authentication: Requires an admin API key
- Parameters: The entity identifier is included in the URL path
- Response: Likely returns a confirmation of deletion
- Expected Status: 201

Identity APIs

This API suite is focused on managing Decentralized Identifiers (DIDs), allowing for their creation, retrieval, and management within an identity system.

11. GET /api/v1/identity/dids

- Purpose: Retrieves a list of Decentralized Identifiers (DIDs)
- Authentication: Requires a user API key
- Response: Returns a JSON object containing an array of DIDs

12. GET /api/v1/identity/dids/{{didRef/didLong}}

- Purpose: Retrieves information about a specific DID using a short-form or a long-form reference
- Authentication: Requires a user API key
- Parameters: didRef - A short-form DID reference; didLong - A long-form DID identifier

- Response: Returns a JSON object with details about the specified DID

13. POST /api/v1/identity/did

- Purpose: Creates a new Decentralized Identifier (DID) for a specific purpose
- Authentication: Requires a user API key
- Parameters:
    - id: A string identifier for the DID (example provided is a hash-like string)
    - purpose: The purpose of the DID (eg "authentication")
- Response: Returns a JSON object with the created DID information, including a "longFormDid"

14. PATCH /api/v1/identity/did/{{did}}

- Purpose: Updates a Decentralized Identifier (DID), possibly to add a new purpose
- Authentication: Requires a user API key
- Content-Type: application/x-www-form-urlencoded
- Parameters:
    - id: A string identifier for the DID (example provided is a hash-like string)
    - purpose: The purpose of the DID (eg "issue" for issuance)
- Response: Returns a JSON object with the created DID information, including a "longFormDid"

Verifiable Credentials - Definition APIs

This API suite focuses on defining the types of verifiable credentials.

15. GET /api/v1/vc/definitions

- Purpose: Retrieves all credential definitions registered within the system.
- Authentication: Requires a user API key
- Parameters: None.
- Response: Returns a JSON object containing the list of credential definitions.

16. GET /api/v1/vc/definitions/{{guid}}

- Purpose: Retrieves a specific credential definition registered within the system.
- Authentication: Requires a user API key
- Parameters:
    - guid: The uid of the definition (e.g., “12345”).

Response: Returns a JSON object containing the matching credential definition.

17. POST /api/v1/vc/definition

- Purpose: Creates a new credential definition in the system.
- Authentication: Requires a user API key
- Parameters:
    - name: The name of the credential (e.g., “origin”).
    - description: A description of the credential (e.g., “Identity of an AI”).
    - version: The version of the credential (e.g., “1.0.1”).
    - tag: A tag to identify the credential (e.g., “AI”).
    - author: The DID of the author (e.g., "did:prism:...").
    - location: the location of the schame (e.g., https://..../<name>.json)
- Response: Returns a JSON object with the newly created credential definition’s details.

Verifiable Credentials - Issuance APIs

This API suite focuses on issuing the verifiable credentials.

18. GET /api/v1/vc/offers?thid={{thid}}

- Purpose: Retrieves the list of issued credential offers (no thid) or a specific offer (thid specified), either pending, accepted, or issued (point of view of Received or Issuer).
- Authentication: Requires a user API key (receiver or issuer)
- Parameters:
    - thid: The thid of the offer (e.g., “12345”).

- Response: Returns a JSON object containing the list of issued credential offers

19. POST /api/v1/vc/offer-noschema

- Purpose: Creates a credential offer without using a referenced schema.
- Authentication: Requires a user API key
- Request Body:
    - connectionId: Connection ID for the recipient of the offer.
    - author: The published short DID of the author of this offer.
    - validityPeriod: The time in seconds the offer remains valid (e.g., 3600).
    - claims: Object containing the claims to be included in the credential (e.g., uid, name, etc.).
- Response: Returns a JSON object with the recordId of the created credential offer.

20. GET /api/v1/vc/offer/{{id}}

- Purpose: Retrieves details about a specific offer record by its ID.
- Authentication: Requires a user API key
- Parameters:
    - {{id}}: The unique identifier of the credential offer record.
- Response: Returns a JSON object with details of the specified offer record, including DIDs and related metadata.

21. POST /api/v1/vc/accept

- Purpose: Accepts a credential offer and issues the credential to the specified subject.
- Authentication: Requires a user API key (holder)
- Request Body:
    - recordId: ID of the pending offer to accept.
    - did: DID of the VC offer to accept.

- Response: Returns a JSON object with details about the accepted offer.

22. POST /api/v1/vc/issue

- Purpose: Finalizes the credential issuance for the offer and officially issues the credential.
- Authentication: Requires a user API key (issuer)
- Request Body:
    - recordId: ID of the pending offer to issue.

Response: Returns a JSON object with the details of the issued credential.

Verifiable Credentials - Proof APIs

This API suite focuses on verifying proof of credentials.

23. GET /api/v1/proof/presentations

- Purpose: Retrieves the list of presentations for proof requests. This can be used to check the status or details of proofs.
- Authentication: Requires a user API key
- Response: Returns a JSON object containing the details of the proof presentations, including the presentation ID.

24. GET /api/v1/proof/presentations?thid={{thid}}

- Purpose: Retrieves a specific presentation for proof requests. This can be used to check the status or details of proof.
- Authentication: Requires a user API key
- Parameters:
    - thid: the thid from the caller viewpoint
- Response: Returns a JSON object containing the details of the proof presentation, including the presentation ID.

25. POST /api/v1/proof/presentation

- Purpose: Creates a credential offer.
- Authentication: Requires a user API key
- Request Body:
    - connection: Connection ID between verifier and prover.
    - challenge: A text string for the prover.
    - domain: Domain where this VC applies.
- Response: Returns a JSON object with the recordId of the created credential offer.

26. POST /api/v1/proof/presentation/accept

- Purpose: For the holder to accept a presentation request.
- Authentication: Requires a user API key (holder)
- Request Body:
    - PresentationId: ID of the presentation from holder point of view.
    - recordId: RecordID of the VC that will be provided as proof.
- Response: Returns a JSON object with confirmation of acceptance.

27. GET / api/v1/proof/presentation/{{id}}

- Purpose: Retrieves a specific presentation proof for a user (verifier point of view).
- Authentication: Requires a user API key (verifier)
- Parameters:
    - id: the unique id of the presentation from the verifier point of view
- Response: Returns a JSON object containing the details of the proof presentation