Artifact

Eduardo Oliveira

March 8, 2025

1 Open Science Platform

1.1 Overview

Traditional centralized systems often exhibit data silos, limited verifiability, and susceptibility to manipulation, impeding the openness and reliability of scientific practices. The decentralized model introduced in this work is designed to mitigate these challenges by enabling efficient data sharing, fostering collaboration, and enhancing the validation of research outputs, thereby strengthening reproducibility and transparency.

This chapter details the design and implementation of the Open Science Platform, a decentralized system that integrates blockchain, IPFS, and smart contracts to improve research reproducibility. By leveraging immutable records and decentralized storage, the platform ensures transparent and verifiable research artifact management. Additionally, extended services are incorporated to facilitate file indexing, metadata extraction, and search functionality. The proposed platform aligns with Open Science principles by providing verifiable and persistently traceable access to research artifacts.

1.2 Technology Stack

The Open Science Platform is developed using a hybrid architecture that combines decentralized and off-chain technologies to ensure secure, traceable, and efficient data management.

1.3 Core Services

The core services of the Open Science Platform provide the fundamental infrastructure for secure and verifiable research artifact management.

- **Hyperledger Iroha v1 Blockchain:** Acts as the core infrastructure for managing user and project accounts, recording transactions, and enforcing business rules via smart contracts to ensure secure and transparent data exchange.

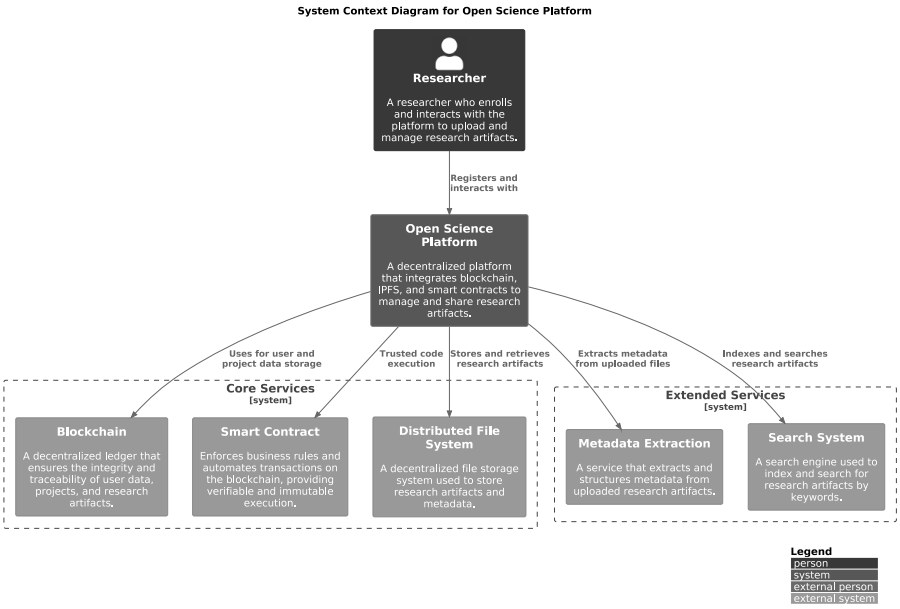


Figure 1: System context diagram for the Open Science Platform

- **InterPlanetary File System (IPFS):** Provides decentralized, tamper-proof storage for research artifacts and metadata, ensuring persistent and verifiable access to shared data.

1.4 Extended Services

The extended services enhance the platform's features by improving file and metadata processing.

- **Apache Tika:** Extracts metadata from uploaded files, enhancing artifact organization and searchability.
- **Whoosh:** Facilitates efficient indexing and keyword-based search for stored artifacts.

1.5 User Interface, integration and execution

- **Jupyter Notebooks (Python):** Powers the front-end interface, facilitating the automation and display of the execution steps. Blockchain interactions are managed via the Iroha v1 Python library, while communication with the IPFS network is handled through the HTTPS client library.

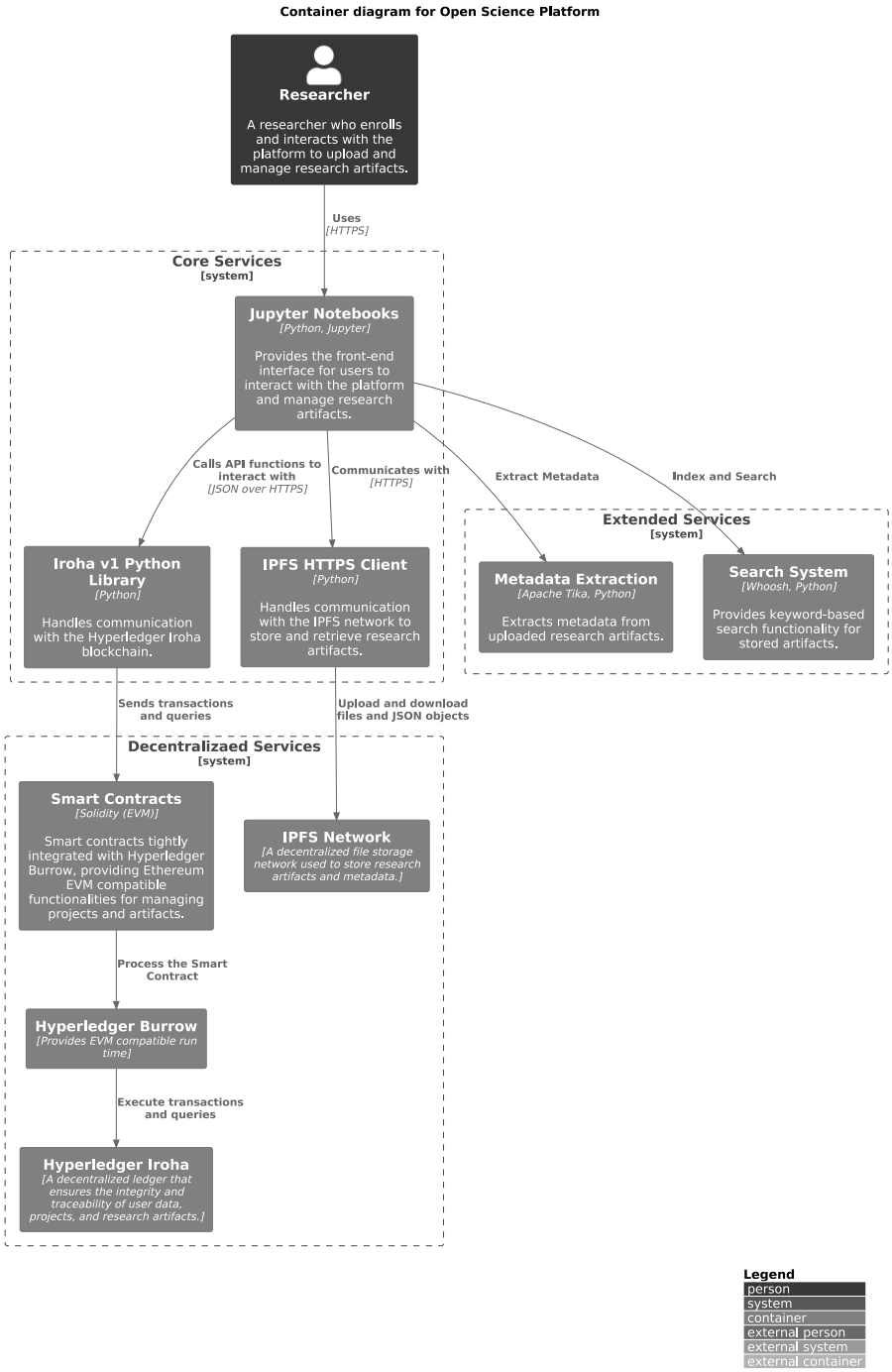


Figure 2: Container diagram for the Open Science Platform

1.6 System Components and Interactions in the Open Science Platform

The Open Science Platform consists of multiple interconnected components, each serving a distinct role in ensuring secure, verifiable, and reproducible research data management. The primary components include Jupyter Server, the blockchain Hyperledger Iroha v1 and the InterPlanetary File System (IPFS). Each of these elements are encapsulated within a Docker container to provide modularity, ease of deployment and reproducibility.

1.6.1 Jupyter Server

The Jupyter Server acts as the primary interface for users interacting with the platform. This component provides a Python kernel for the execution environment that integrates the Iroha v1 Library, the IPFS HTTPS client, Apache Tika for metadata handling, and the Woosh Indexer and Search system. It enables users to:

- Execute Python scripts to submit transactions and queries to the blockchain via smart contracts.
- Upload and retrieve files and metadata (JSON objects) stored in IPFS.
- Process and index research data using Apache Tika and Woosh for enhanced searchability.
- Access and visualize blockchain-stored metadata for Open Science applications.

1.6.2 Blockchain

The blockchain runs based on a Hyperledger Iroha v1 network and acts as a distributed ledger for recording transactions. It ensures immutability, transparency, and verifiability of stored research metadata. This component:

- Receives transactions from the Jupyter Server via a gRPC API.
- Stores metadata references, ensuring that uploaded research artifacts can be authenticated.
- Interacts with PostgreSQL for structured storage of blockchain metadata.
- Supports smart contracts through the integration of Hyperledger Burrow, which provides a modular blockchain client with a permissioned smart contract interpreter partially developed to the specification of the Ethereum Virtual Machine (EVM).

1.6.3 Storage

The InterPlanetary File System (IPFS) is a decentralized storage solution that manages the research outputs. This component:

- Stores digital research artifacts in a content-addressed manner.
- Allows the Jupyter Server to upload and retrieve files via an HTTPS API.
- Ensures long-term availability of scientific data through distributed storage principles.

1.6.4 Relational Database (PostgreSQL)

The PostgreSQL database provides structured storage for blockchain-related data. It is used exclusively and managed by Iroha v1 to:

- Maintain an efficient and queryable record of transactions.
- Ensure that research metadata stored on the blockchain can be retrieved and verified.
- Support blockchain operations requiring fast access to structured data.

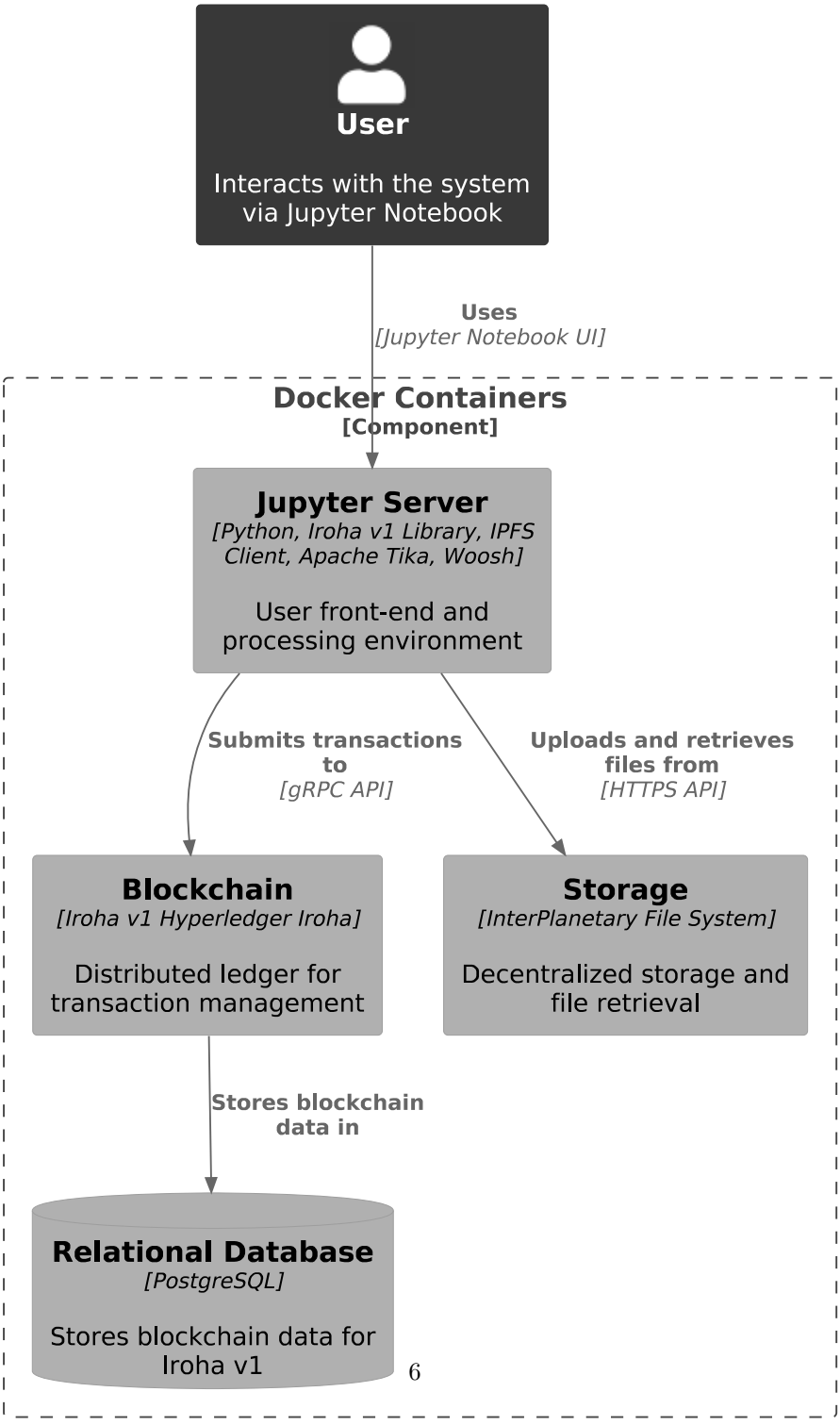
1.6.5 Component Interactions

The components interact in aseamless and decentralized manner:

1. **User Interaction**: The user submits transactions, uploads files, and queries research data through the Jupyter Server.
2. **Blockchain Transactions**: Jupyter Server sends and retrieves research metadata to the Iroha blockchain via gRPC API.
3. **Metadata Storage**: Iroha stores data in the PostgreSQL database for efficient retrieval.
4. **Decentralized Storage**: Research artifacts are stored in IPFS, with their unique file identifiers recorded on the blockchain.
5. **File Retrieval**: Users can retrieve files from IPFS using their content identifiers (CID), ensuring authenticity and reproducibility.

This architecture guarantees trustworthy and reproducible scientific research by leveraging blockchain for integrity, IPFS for decentralized storage, and Jupyter as an accessible research environment.

Component Diagram for Open Science Platform



1.7 Platform Operations

The platform supports a set of core operations that regulate user interactions with projects and data management.

- **User Self-Enrollment** – A user self-enrolls on the platform by providing a private key that complies with the ED25519 or SHA-3 standards and identity information, including full name, institution, email, ORCID, and role. An account is created for the user in the blockchain. All data provided in the enrollment is structured in key/value pairs into a JSON object and uploaded to IPFS, with the corresponding Content Identifier (CID) recorded on the blockchain attributes of the user account.
- **Project Registration** – Users can register a project by specifying a descriptive name, an abstract, relevant keywords, start and end dates, funding agency, and location. Upon registration, a blockchain account is created. This data is structured in key/value pairs into a JSON object and uploaded to IPFS, with the corresponding Content Identifier (CID) recorded on the blockchain attributes of the project account.
- **User and Project Accounts Linkage** – Once both user and project accounts are created, the system updates their attributes to establish a bidirectional association. This ensures that querying a user account reveals linked project accounts, and vice versa, facilitating traceability and efficient project management.

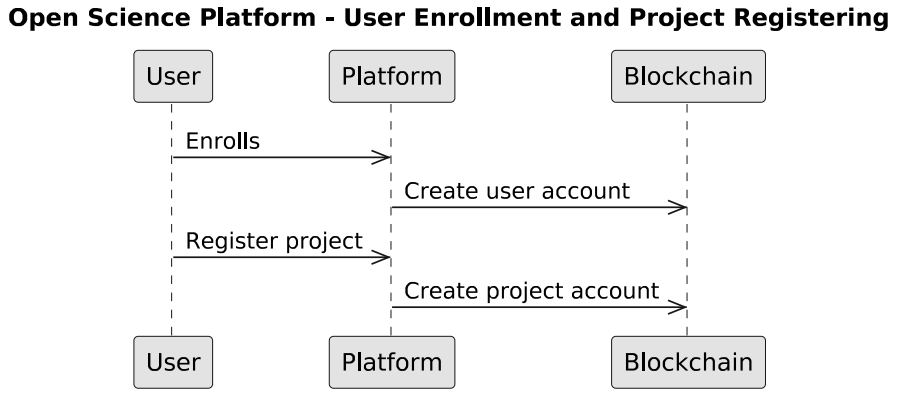


Figure 4: User enrollment and project registering for the Open Science Platform

1.8 Artifact Management

- **File Upload** – A user may upload research artifacts, including papers, datasets, and images. Each file is stored on IPFS, generating a unique

Content Identifier (CID) that ensures traceability and integrity. The CID is then recorded on the blockchain attributes of the project, establishing a verifiable reference to the artifact.

- **Metadata Extraction and Storage** – After the upload,the file available metadata is extracted. The extracted metadata is structured in key/value pairs into a JSON object and uploaded to IPFS, with the corresponding Content Identifier (CID) recorded on the attributes of the project account in the blockchain, ensuring metadata provenance and verification.
- **Indexing** – To facilitate efficient retrieval, the system indexes the metadata of every uploaded file, including full text indexing for text based files.
- **Searching** – Users can perform keyword-based searches to locate relevant research artifacts, with search results displaying metadata details, including descriptions, subject and authorship.

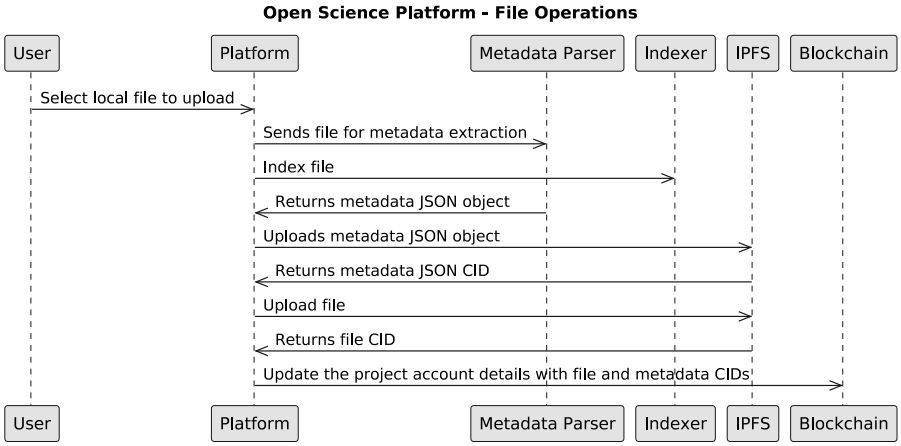


Figure 5: File operations diagram for the Open Science Platform

1.9 Validation and Download

- **File Validation** – To ensure data integrity and authenticity, the platform verifies whether the CID of a file stored on IPFS matches the CID recorded on the blockchain. A discrepancy between these identifiers signals potential tampering or corruption.
- **File Download** – The system retrieves and downloads validated files from IPFS to the user local file system for later usage.

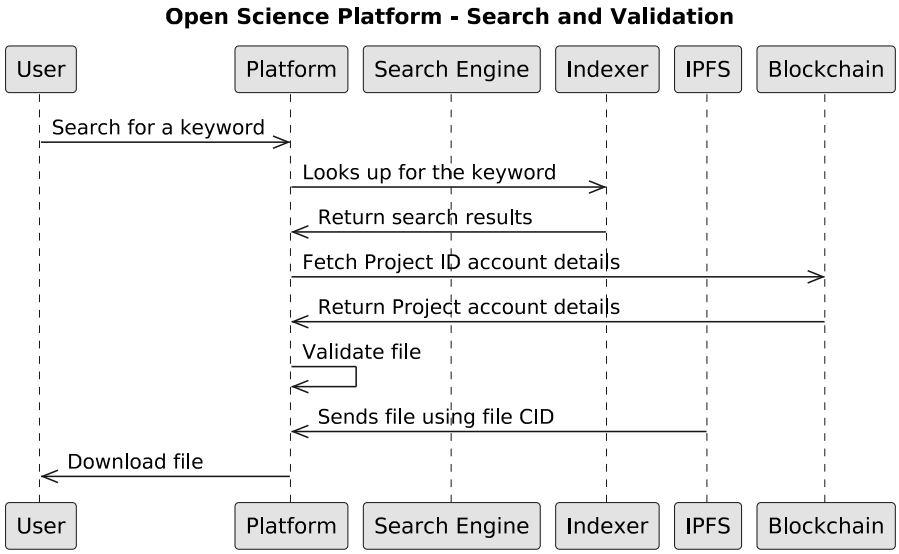


Figure 6: keyword search, file validation and download

1.10 Entity-Relationship Model for the Open Science Platform

The entity-relationship model for the Open Science Platform defines the logical structure of users and research projects, capturing the associations between these entities. The primary entities in this model are `User` and `Project`, which are connected through an ownership relationship.

1.11 User Entity

The `User` entity represents an individual interacting with the platform. Each user is uniquely identified by an account ID and has attributes that describe personal and institutional information. The attributes of the `User` entity are listed in Table 1.

Table 1: Attributes of the User entity

Attribute	**Description**
`account_id`	A unique identifier assigned to the user
`full_name`	The complete name of the user
`email`	The email address used for communication
`institution`	The organization to which the user is affiliated
`orcid`	The Open Researcher and Contributor ID
`role`	The role of the user within the platform

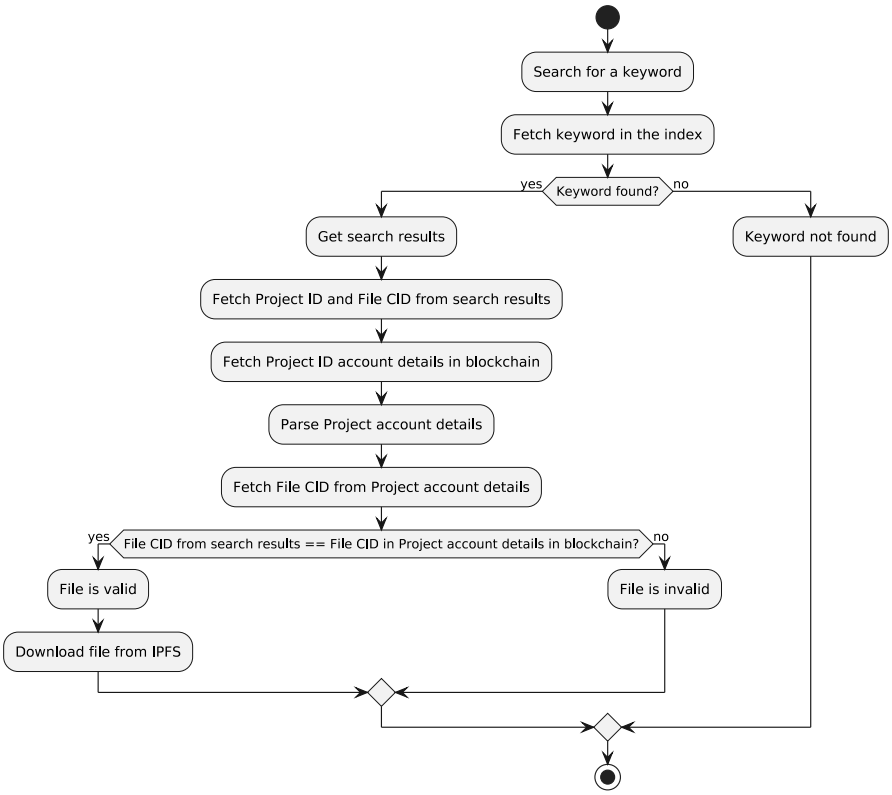


Figure 7: File validation and download

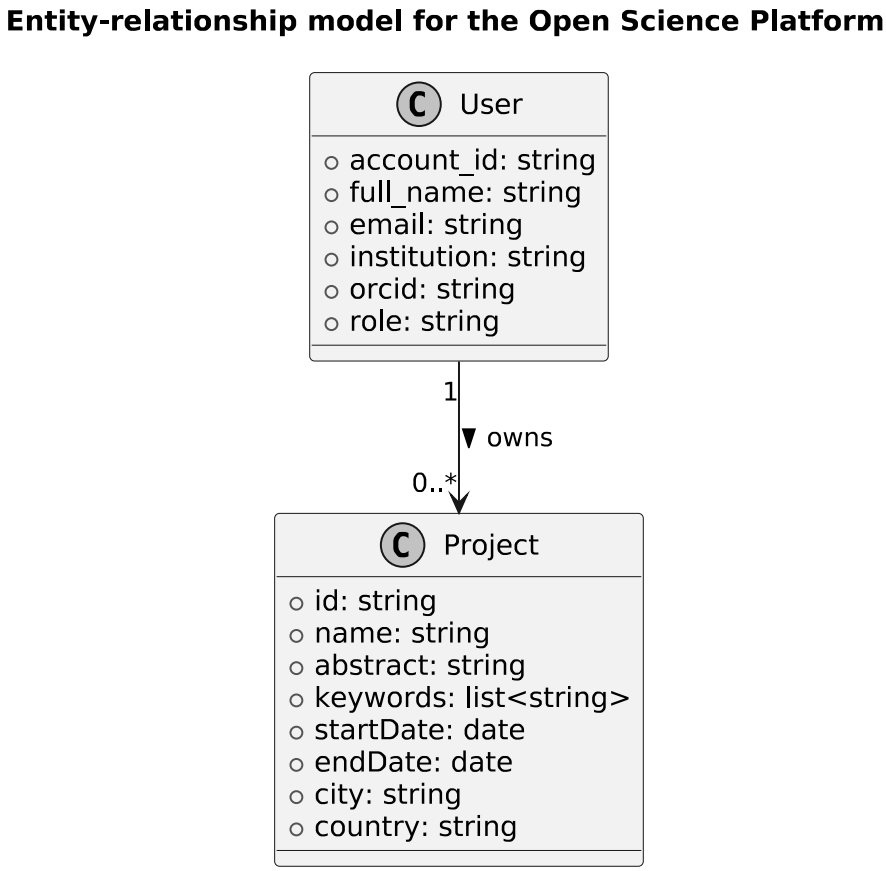


Figure 8: Entity-relationship model for the Open Science Platform

1.12 Project Entity

The `Project` entity represents a research project registered in the platform. It contains essential metadata to describe the project and facilitate discovery and collaboration. The attributes of the `Project` entity are listed in Table 2.

Table 2: Attributes of the Project entity

Attribute	Description
`id`	A unique identifier assigned to the project
`name`	The official name of the project
`abstract`	A brief summary outlining the research objectives
`keywords`	A list of relevant keywords associated with the project
`startDate`	The date when the project officially begins
`endDate`	The date when the project is expected to conclude
`city`	The city where the project is primarily conducted
`country`	The country associated with the research project

1.13 Ownership Relationship

A `User` owns one or more `Project` entities, establishing a one-to-many relationship. This means that a single user can be responsible for multiple projects, but each project is owned by exactly one user. This relationship is crucial for managing project access, ensuring accountability, and maintaining provenance of research activities. This model ensures a structured representation of research projects and their associated users, supporting an organized approach to data management in the Open Science Platform.

1.14 Hyperledger Iroha v1 Data Model

The entity-relationship (ER) model represents a permissioned blockchain system, likely based on Hyperledger Iroha. This model defines the core entities, their attributes, and relationships to enable role-based access control, asset management, and multi-signature security.

2 Core Entities and Their Attributes

2.1 Account Entity

The `account` entity represents a user or system account registered on the blockchain. Table 3 lists its attributes.

The account entity is associated with multiple other entities:

- `account_has_signatory` links accounts to public keys.
- `account_has_roles` assigns roles to accounts.

Table 3: Attributes of the Account Entity

Attribute	Description
account_id	Unique identifier of the account
domain_id	Links the account to a specific domain
quorum	Required number of signatories for multi-signature transactions
data	Stores additional metadata in JSON format

- account_has_asset tracks asset ownership.
- account_has_grantable_permissions defines access control for delegation.

2.2 Signatory Entity

The signatory entity manages cryptographic authentication.

Table 4: Attributes of the Signatory Entity

Attribute	Description
public_key	Cryptographic key associated with an account

Each account can have multiple public keys, enabling multi-signature security.

2.3 Asset Entity

The asset entity represents digital assets on the blockchain.

Table 5: Attributes of the Asset Entity

Attribute	Description
asset_id	Unique identifier for the asset
domain_id	Links the asset to a domain
precision	Defines decimal precision for asset amounts

The entity is associated with account_has_asset, which links assets to accounts.

3 Role and Permission Management

3.1 Role Entity

The role entity defines permission-based access control.

role_has_permissions links roles to specific permissions, and account_has_roles assigns roles to accounts.

Table 6: Attributes of the Role Entity

Attribute	Description
role_id	Unique identifier for the role

3.2 Domain Entity

The domain entity organizes accounts and assets within logical boundaries.

Table 7: Attributes of the Domain Entity

Attribute	Description
domain_id	Unique identifier for the domain
default_role	Default role assigned to accounts created in the domain

4 Permission and Transaction Management

Permissions are enforced using:

- role_has_permissions: Maps permissions to roles.
- account_has_grantable_permissions: Defines permissions that an account can delegate.
- account_has_roles: Assigns roles to accounts.

5 Relationship Cardinalities

5.1 One-to-Many Relationships

- A domain can have multiple accounts and assets.
- A role can have multiple permissions.
- An account can be linked to multiple roles, signatories, and assets.

5.2 Many-to-Many Relationships

- account_has_roles: An account can have multiple roles; a role can be assigned to multiple accounts.
- role_has_permissions: A role can have multiple permissions, and each permission can belong to multiple roles.
- account_has_grantable_permissions: An account can delegate specific permissions to multiple others.

6 Conclusion

This ER model follows Hyperledger Iroha's permissioned blockchain structure. It ensures fine-grained access control, multi-signature security, and domain-based account management. The model enables role-based permissions and asset tracking, making it well-suited for financial applications, identity management, and blockchain-based ecosystems.

6.1 Comparing Models

6.2 Metadata Representation

To standardize metadata representation and enhance interoperability, three key ontologies were incorporated: **FOAF (Friend of a Friend)**, **Dublin Core**, and **Schema.org**. These vocabularies provide a common semantic framework that facilitates integration with other systems based on W3C standards, such as knowledge graphs and linked data environments. FOAF is particularly useful for modeling user identity and social connections, Dublin Core provides metadata descriptors for digital assets, and Schema.org enhances discoverability by aligning with web-based search engines and indexing mechanisms.

6.3 Metadata

Metadata in the context of the platform has a two fold approach. The first is related to the identity of an user account, holding key value pairs of attributes related to the user such as full name, email, institution, ORCID.

```
{
    "@context": {
        "schema": "http://schema.org/",
        "foaf": "http://xmlns.com/foaf/0.1/"
    },
    "@graph": [
        {
            "@type": "foaf:Person",
            "foaf:name": "Jolly Noether",
            "foaf:mbox": "jolly_noether@email.com",
            "foaf:organization": {
                "@type": "foaf:Organization",
                "foaf:name": "Morris College"
            },
            "schema:identifier": {
                "@type": "PropertyValue",
                "propertyID": "ORCID",
                "value": "9833-6461-2701-X"
            },
            "foaf:holdsAccount": {
```

```
                "schema:identifier": "jolly_noether@test",
                "schema:roleName": "author",
                "schema:publicKey": <publick ey value>},
            "schema:linked_project": "10278@test"
        }
    ]
}
```

6.4 Smart Contract

The platform deploys standard Ethereum EVM contracts in Solidity for account creation and detail setting. These contracts are deployed through the Iroha v1 Python Library.

6.5 Benefits

The Open Science platform offers numerous benefits for researchers and members of the scientific community, including:

- Secure data sharing: By utilizing blockchain technology and IPFS, the platform ensures tamper-proof data exchange.
- Transparent data management: The use of smart contracts and decentralized storage guarantees transparency in data access and modification history.
- Collaborative research environment: The platform enables researchers to collaborate on projects, share artifacts and results, and track progress.

6.6 Challenges

The Open Science platform faces several challenges, including:

- Scalability: As the number of users increases, the platform needs to be able to handle a growing amount of data and transactions efficiently.
- Interoperability: Ensuring seamless integration with existing research platforms and tools is crucial for widespread adoption.
- User Adoption: Educating researchers about the benefits of decentralized technologies and the Open Science platform can be an uphill battle.

6.7 Future Work

The Open Science platform has several areas for future development, including:

- Integration with existing research platforms: Collaborations with established research platforms to expand the platform's reach and user base.

- Enhanced security measures: Implementing additional security protocols to protect against potential threats and maintain the integrity of shared information.
- User interface improvements: Enhancing the web interface to make it more user-friendly and accessible for researchers from diverse backgrounds.

7 Conclusion

The Open Science platform is a comprehensive solution for secure, transparent, traceable, and tamper-proof data sharing and collaboration. By leveraging decentralized technologies, the platform empowers researchers to share project artifacts and data in a reliable and trustworthy manner.