PROJECT DELIVERABLE REPORT

Grant Agreement Number: 101058732

Joint Industrial Data Exchange Pipeline

Type: Delivery Report

D3.5 JIDEP APIs and SDK documentation – Final

Issuing partner	Technovative Solutions (TVS)
Participating partners	University of Cambridge (UCAM) Universita di Trento (UNITN)
Document name and revision	D3.5 JIDEP APIs and SDK documentation - Final
Author(s)	Rasel Ahmed (TVS)
Reviewers(s)	Miah Raihan Mahmud Arman (TVS) Tanvir Islam (TVS) Feroz Farazi (UCAM) Simone Bocca (UNITN)
Deliverable due date	30/11/2024
Actual submission date	30/11/2024

Project Coordinator	Vorarlberg University of Applied Sciences
Tel	+43 (0) 5572 792 7128
E-mail	florian.maurer@fhv.at
Project website address	www.jidep.eu

Dissemination Level		
PU	Public	✓
PP	Restricted to other programme participants (including the Commission services)	
CO	Confidential, only for members of the consortium (including the Commission services)	
SE	Sensitive, limited under the conditions of the Grant Agreement	

Table of Contents

List of Tables

Executive summary

This report details the APIs developed for the services in WP2 and WP3. These APIs are crucial to JIDEP's functionality, offering strong security through encrypted data storage accessible only to authorised users.

RESTful APIs were also implemented to enable seamless interactions with the services developed in WP2. These APIs facilitate services, improve platform accessibility, and reduce potential conflicts for data contributors. Blockchain Anchoring (Proof of Existence) APIs were developed to enhance security further, ensuring greater traceability and integrity by linking off-chain data with the JIDEP blockchain.

JIDEP's technological infrastructure is designed to provide a secure, efficient, and user-friendly environment. This deliverable supports the platform's current functionalities and is essential for future growth, potentially widening its range of use cases.

1. Introduction

The primary focus of this deliverable is to describe the APIs that form the foundation of the services and drive the implementation of JIDEP's platform. RESTful APIs have been utilised to connect the services developed in WP2 and WP3. These APIs are essential for allowing smooth data transformation functionalities, reducing any obstacles for data contributors, and making the platform easy to navigate. Additionally, Blockchain Anchoring (Proof of Existence) APIs enhance security by ensuring data transactions are secure and traceable by linking off-chain data with the JIDEP blockchain.

JIDEP's innovative approach reflects its adaptability, allowing data providers to manage data usage efficiently. Across WP2 and WP3, a wide range of services has been developed to ensure the smooth operation and fulfilment of requirements within the JIDEP project. The following is a list of services already developed. These APIs are also used in Work Package 4 (WP4) to support tool development within the JIDEP platform.

1. Ontology and Query
2. Environmental Analytics
3. Material Passport
4. Catalogue
5. Circularity Calculator
6. Collaborative Space
7. KG Builder
8. Proof of Existence

2. Background

The terms API (Application Programming Interface) and SDK (Software Development Kit) are fundamental in software development. They provide mechanisms and tools that allow different software programs to interact and enable developers to create applications efficiently.

2.1 APIs and SDK

An API is a set of rules and specifications that software programs can follow to communicate with each other [1]. It serves as an interface between different software programmes and facilitates their interaction, similar to how the user interface facilitates interaction between humans and computers. APIs are designed to expose specific functionality and data of an application or service while protecting other parts of the application, which ensures programmatic access in a controlled manner [2].

An SDK or Software Development Kit is a collection of software tools, guidelines, and programs used to develop applications for specific platforms or frameworks [3]. In the scope of the JIDEP project, APIs and SDKs are treated as functionally equivalent, serving as comprehensive sets of tools for software development. Each SDK in the project includes APIs and a suite of

other development tools designed to facilitate the creation of applications specific to the JIDEP platform. These SDKs provide libraries, documentation, code samples, and guidelines, essentially encompassing everything developers need to efficiently build and integrate software within the JIDEP ecosystem. This approach ensures a streamlined development process, emphasising the seamless integration of APIs and additional tools under the umbrella of each SDK.

3. Material Passport APIs

3.1 Overview

The Material Passport API offers comprehensive access to essential product data. It empowers developers with easy integration into existing systems while also providing users with transparent and accurate product information.

3.2 Request Format

The request body should be formatted as JSON for POST and PUT requests and include the Content-Type header set to application/json. Authentication is managed through HTTP headers. In this instance, a Bearer token is used for authentication and should be included in all API requests to the server.

Authorisation: Bearer eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJi

3.3 Response Format

The response format for all requests within this system is structured as a JSON object. This standardised approach ensures consistency across various API endpoints, facilitating easier data parsing and integration for developers and systems interacting with the API. Being lightweight and widely supported, JSON is ideal for web-based communication, allowing for quick transmission and straightforward interpretation of data, regardless of the client's software or hardware environment.

Status Codes:

a. **200 OK:** Successfully retrieved the passport details.
b. **201 Created**: The passport was created successfully.
c. **400 Bad Request**: The request was malformed or missing the required parameters.
d. **404 Not Found:** No passport found with the provided ID.
e. **500 Internal Server Error:** An error occurred on the server while processing the request.

3.4 List of APIs

API access can be provided over HTTPS and HTTP. We recommend utilising HTTPS for anything other than local development. Table 1 provides information regarding APIs that have been used for managing material passports.

Table 1 List of Material Passport APIs

Method	Endpoint	Description	Returns
GET	/passports/{id}	Retrieve detailed information about a specific passport, id (String): The unique identifier of the passport.	The passport data includes circular economy potentials, environmental performance, identification details, documents, physical properties, and manufacturing information.
POST	/passports	Create a new passport with provided attributes.	A JSON object returns success or failure status and message, plus the ID of the newly created product passport.
GET	/passports	Retrieve all detailed information of passports created by the user	List of passport data includes circular economy potentials, environmental performance, identification details, documents, physical properties, and manufacturing information.

PUT	/passports/{id}	Update a new passport entry with updated information.	A JSON object returns success or failure status and message, plus the ID of the updated product passport.
DELETE	/passports/{id}	Archive a specific passport, id (String): The unique identifier of the passport.	A JSON object returns success or failure status and message.

4. Ontology and Query APIs

4.1 Overview

UCAM has developed an ontology, a formal representation of knowledge in a specific domain related to materials, products and their properties. The ontology defines concepts, relationships, and properties associated with material passports. These APIs assist in generating ontology schemas, which are essential for creating material passports.

4.2 Request Format

The request body should be formatted as JSON for POST and PUT requests and include the Content-Type header set to application/json. Authentication is managed through HTTP headers. In this instance, a Bearer token is used for authentication and should be included in all API requests to the server.

4.3 Response Format

The response format for all requests within this system is structured as a JSON object. This standardised approach ensures consistency across various API endpoints, facilitating easier data parsing and integration for developers and systems interacting with the API. Being lightweight and widely supported, JSON is ideal for web-based communication, allowing for quick transmission and straightforward interpretation of data, regardless of the client's software or hardware environment.

Status Codes:

a. **200 OK:** Successfully retrieved the ontology or schema details.

b. **400 Bad Request**: The request was malformed or missing the required parameters.
c. **404 Not Found:** No schema found.
d. **500 Internal Server Error:** An error occurred on the server while processing the request.

4.4 List of APIs

The API exposes endpoint(s) to retrieve schema information for different categories of properties of the Material Passport creation process. The specific endpoint(s) similar to /schema/step/{name}, where the name represents a step of the Material Passport. Table 2 provides information regarding APIs that have been used for managing ontology and query.

Table 2 List of Ontology and Query APIs

Method	Endpoint	Description	Returns
GET	/schema/category/identifier	Get schema for the "Identifiers" category.	Schema as JSON format from a file or Ontology.
GET	/schema/category/physical_properties	Get schema for the "Physical Properties" category.	Schema as JSON format from a file or Ontology.
GET	/schema/category/composition_properties	Get schema for the "Composition Properties" category.	Schema as JSON format from a file or Ontology.
GET	/schema/category/circular_economy	Get schema for the "Circular Economy" category.	Schema as JSON format from a file or Ontology.
GET	/schema/category/environmental_performance	Get schema for the "Environmental Performance" category.	Schema as JSON format from a file or Ontology.

5. Environment Analytic APIs

5.1 Overview

Environment Analytic Tool is a sophisticated life cycle assessment (LCA) tool designed for professionals to analyse the environmental impact of products and services. It facilitates comprehensive LCA studies, offering a robust database of materials and processes, which supports transparent, science-based decision-making. This API provides dynamic modelling capabilities, allowing users to simulate various scenarios and compare the environmental footprints of alternatives. Its intuitive interface and extensive reporting features make it accessible to both experts and newcomers in sustainability practices. This tool is essential for organisations committed to reducing their ecological footprint and achieving sustainability goals.

5.2 Request Format

The request body should be formatted as JSON for POST and PUT requests and include the Content-Type header set to application/json. Authentication is managed through HTTP headers. In this instance, a Bearer token is used for authentication and should be included in all API requests to the server.

5.3 Response Format

The response format for all requests within this system is structured as a JSON object. This standardised approach ensures consistency across various API endpoints, facilitating easier data parsing and integration for developers and systems interacting with the API. Being lightweight and widely supported, JSON is ideal for web-based communication, allowing for quick transmission and straightforward interpretation of data, regardless of the client's software or hardware environment.

Status Codes:

a. **200 OK:** Successfully retrieved the details.
b. **400 Bad Request**: The request was malformed or missing the required parameters.
c. **404 Not Found:** No subcategories were found with the provided category.
d. **500 Internal Server Error:** An error occurred on the server while processing

5.4 List of APIs

This guide is designed to provide developers with the necessary information to integrate with the Example API, which facilitates data transactions in various domains. Table 3 provides information regarding APIs that have been used for calculating environmental analytics.

Table 3 List of Environment Analytic APIs

Method	Endpoint	Description	Returns
GET	/api/countries	Get list of countries available for calculating LCA	Return a JSON object containing an array of country objects.
GET	/api/categories?type=material	Get list of categories available for	Return a JSON object containing an arr

		calculating LCA	ay of category objects.
GET	/api/sub-categories?category=chemicals	Get list of subcategories available for calculating LCA	Return a JSON object containing an array of subcategory of objects.
GET	/api/product-by-sub-category?sub_category=Organic	Get list of products by subcategories	Return a JSON object containing an array of produ

			cts .
GET	/api/product-by-category ?category=chemicals&type=material	Get list of products by categories	Return a JSON object containing an array of products of objects.
GET	/api/units	Get list of units available for calculating LCA	Return a JSON object containing an array of subcategory of obj

			ect s.
GE T	/api/lca-methods	Ge t list of LC A me tho ds av ail abl e for cal cul ati ng LC A	Re tur n a JS ON obj ect co nta ini ng an arr ay of LC A me tho d obj ect s.
PO ST	/api/perform-assembly	Cal cul ati ng LC A res ult	Re tur n a JS ON obj ect for LC A res ult.

6. Marketplace APIs

6.1 Overview

Marketplace APIs include product management for listing and inventory and order management for processing and tracking orders. They also secure

payment processing (Which will be introduced in the final version of this report) and support advanced search and filtering functionalities. Shipping APIs integrate with logistics providers for tracking and delivery management. Admin APIs facilitate administrative tasks for maintaining and moderating the marketplace that includes product approval.

6.2 Request Format

The request body should be formatted as JSON for POST and PUT requests and include the Content-Type header set to application/json. Authentication is managed through HTTP headers. In this instance, a Bearer token is used for authentication and should be included in all API requests to the server.

6.3 Response Format

The response format for all requests within this system is structured as a JSON object. This standardised approach ensures consistency across various API endpoints, facilitating easier data parsing and integration for developers and systems interacting with the API. Being lightweight and widely supported, JSON is ideal for web-based communication, allowing for quick transmission and straightforward interpretation of data, regardless of the client's software or hardware environment.

Status Codes:

a. **201 Created**: The product was published successfully.
b. **400 Bad Request**: The request was malformed or missing the required parameters.
c. **500 Internal Server Error:** An error occurred on the server while processing

6.4 List of APIs

The Marketplace API offers comprehensive access to essential product data. It empowers developers with easy integration into existing systems while also providing users with transparent and accurate product information. Table 4 provides information regarding APIs that have been used for managing products in marketplace.

Table 4 List of Marketplace APIs

Method	Endpoint	Description	Returns

POST	/publish/product	Create a new product including product image, title, description and price.	The product id including product information.
PUT	/product/{id}/update	Update product including product image, title, description and price.	A JSON object returns success or failure status and message, plus details of the updated product.

7. Catalogue APIs

7.1 Overview

The Catalogue API facilitates the seamless publication of catalogue information. Users submit a passport containing essential authentication details. A dictionary response confirms the operation's success, signalling the completion of catalogue publishing.

7.2 Request Format

The request body should be formatted as JSON for POST and PUT requests and include the Content-Type header set to application/json. Authentication is managed through HTTP headers. In this instance, a Bearer token is used for authentication and should be included in all API requests to the server.

7.3 Response Format

The response format for all requests within this system is structured as a JSON object. This standardised approach ensures consistency across various API endpoints, facilitating easier data parsing and integration for developers and systems interacting with the API. Being lightweight and widely supported, JSON is ideal for web-based communication, allowing for quick transmission and straightforward interpretation of data, regardless of the client's software or hardware environment.

Status Codes:

a. **201 Created**: The passport was published to the catalogue successfully.
b. **400 Bad Request**: The request was malformed or missing the required parameters.

c. **500 Internal Server Error:** An error occurred on the server while processing

7.4 List of APIs

This streamlined process ensures the efficient distribution of catalogue data, empowering users to promptly access and utilise updated information. With robust authentication measures, the API guarantees the secure handling of sensitive data, providing a reliable platform for catalogue management and distribution within diverse organisational settings. Table 5 provides information regarding APIs that have been used for publishing material passport to catalogue service developed by UNITN.

Table 5 List of Catalogue APIs

Method	Endpoint	Description	Returns
POST	/publish/catalog/v2	Publish a new catalogue metadata including material passport image, title and descrip	Upon successful execution of the publishing process, this API returns a dictionary containing a message con

		tion.	firming the completion of the publishing operation.

8. Circularity Calculator APIs

8.1 Overview

The Circularity Calculator is an advanced tool designed to help businesses and industries improve sustainability and reduce waste. It uses the MacArthur Methodology, a well-known framework developed by the Ellen MacArthur Foundation, to assess circular economy practices [4]. This methodology allows for detailed, data-driven analysis, ensuring precise results align with industry standards. Using the Circularity Calculator, companies can make informed decisions to support the circular economy and enhance their sustainability efforts.

8.2 Request Format

The request body should be formatted as JSON for POST and PUT requests and include the Content-Type header set to application/json. Authentication is managed through HTTP headers. In this instance, a Bearer token is used for authentication and should be included in all API requests to the server.

8.3 Response Format

The response format for all requests within this system is structured as a JSON object. This standardised approach ensures consistency across various API endpoints, facilitating easier data parsing and integration for developers and systems interacting with the API. Being lightweight and widely supported, JSON is ideal for web-based communication, allowing for quick transmission and straightforward interpretation of data, regardless of the client's software or hardware environment.

Status Codes:

a. **201 Created**: The circularity index of the passport has been calculated successfully.

b. **400 Bad Request**: The request was malformed or missing the required parameters.
c. **500 Internal Server Error:** An error occurred on the server while processing

8.4 List of APIs

The Circularity Calculator returns a JSON object indicating success or failure, providing a calculated circularity result for a product on a scale from 0 to 1. Table 6 provides information regarding APIs that have been used for the calculation circularity of a product.

Table 6 List of Circularity Calculator APIs

Method	Endpoint	Description	Returns
POST	/circularity-calculator/macarthur/v2	Calculate Circularity of a passport or product the result will be between	A JSON object returns success or failure status and message, it provide

		0 and 1.	s the cal cul at ed re sul t of cir cul ari ty of a pr od uc t wh ich is be tw ee n 0 an d 1.

9. Collaborative Space APIs

9.1 Overview

A Collaborative Space is a service that helps stakeholders manage their teams and control access to material passports and users. It provides a mechanism for adding and organizing users, setting roles, and assigning permissions, ensuring everyone has the right level of access. The Collaborative Space includes features like shared passports, making it easier for teams to work together.

9.2 Request Format

The request body should be formatted as JSON for POST and PUT requests and include the Content-Type header set to application/json.
Authentication is managed through HTTP headers. In this instance, a Bearer token is used for authentication and should be included in all API requests to the server.

9.3 Response Format

The response format for all requests within this system is structured as a JSON object. This standardised approach ensures consistency across various API endpoints, facilitating easier data parsing and integration for developers and systems interacting with the API. Being lightweight and widely supported, JSON is ideal for web-based communication, allowing for quick transmission and straightforward interpretation of data, regardless of the client's software or hardware environment.

Status Codes:

a. **200 OK:** Successfully retrieved the user details.
b. **201 Created**: The user was created successfully.
c. **400 Bad Request**: The request was malformed or missing the required parameters.
d. **404 Not Found:** No passport found with the provided ID.
e. **500 Internal Server Error:** An error occurred on the server while processing the request

9.4 List of APIs

Collaborative Space APIs help teams work together efficiently by managing users, setting roles, and controlling access. Table 7 provides information regarding APIs that have been used for managing users and user access within the organisation.

Table 7 List of Collaborative Space APIs

Method	Endpoint	Description	Returns
GET	/api/users/{id}	Retrieve detailed information about a user.	The detailed information of the newly created user.
POST	/collaborative-space/users	Create a new user under the company.	A JSON object returns success or failure status and message, plus the ID of the newly created user.

GET	/collaborative-space/passports	Retrieve all detailed information of passports shared to the user	List of passports data includes circular economy potentials, environmental performance, identification details, doc-unmints, physical properties, and manufacturing information.
PUT	/api/users/{id}/update	Update a user with updated information.	A JSON object returns success or failure status and message, plus the ID of the updated user.
DELETE	/api/users/{id}/delete	Delete a user, id (String): The unique identifier of the user.	A JSON object returns success or failure status and message.

10. Knowledge Graph (KG) Builder APIs

10.1 Overview

Knowledge Graph Builder is a comprehensive methodology designed to facilitate the efficient design, development, and implementation of domain-specific Knowledge Graphs (KGs). The methodology is particularly suited for organisations seeking to leverage structured data in specialised domains, ensuring their KGs are scalable and relevant to their unique needs. More details about the Knowledge Graph are available on Deliverable **D3.2**.

10.2 Request Format

The request body should be formatted as JSON for POST and PUT requests and include the Content-Type header set to application/json.

10.3 Response Format

The response format for all requests within this system is structured as a JSON object. This standardised approach ensures consistency across various API endpoints, facilitating easier data parsing and integration for developers and systems interacting with the API. Being lightweight and widely supported, JSON is ideal for web-based communication, allowing for quick transmission and straightforward interpretation of data, regardless of the client's software or hardware environment.

Status Codes:

1) **201 Created**: The knowledge graph has been created successfully.
2) **200 OK**: The knowledge graph has been retrieved successfully.
3) **400 Bad Request**: The request was malformed or missing the required parameters.
4) **500 Internal Server Error:** An error occurred on the server while processing

10.4 List of APIs

The KG Builders API returns a JSON object that will be used to create a .ttl file to represent the knowledge graph for a specific domain. Table 8 provides information regarding APIs that have been used for building a knowledge graph based on material passport data.

Table 8 List of KG Builder APIs

Method	Endpoint	Description	Returns
POST	/dataset	Create a knowledge graph based on	A JSON object returns a success

		the domain of the material passport.	or failure status and message.
GET	/build	Build a knowledge graph based on the domain of the material passport.	A JSON object returns raw data to create a .ttl file.

11. Proof of Existence APIs

11.1 Overview

The Proof of Existence API enables users to securely store encrypted data on the Polygon blockchain. Once the data is stored, it becomes tamper-proof,

allowing for easy verification of its existence at any future point. This API is well-suited for creating permanent, reliable records that require secure tracking and verification without dependence on third parties. More details about Proof of existence are available on Deliverable **D3.4**.

11.2 Request Format

The request body should be formatted as JSON for POST and PUT requests and include the Content-Type header set to application/json.

11.3 Response Format

The response format for all requests is consistently structured as a JSON object. This standardised approach ensures uniformity across various API endpoints, simplifying data parsing and integration for developers and systems. Given its lightweight nature and broad support, JSON is well-suited for web-based communication, allowing for efficient data transmission and straightforward interpretation.

Status Codes:

- **200 OK:** The request was successful, and the data verification was completed.
- **201 Created:** The data has been successfully stored on the blockchain.
- **400 Bad Request:** The request was invalid, likely due to missing or incorrect data.
- **404 Not Found:** The data could not be found on the blockchain.
- **500 Internal Server Error:** An error occurred on the server while processing the request

11.4 List of APIs

The Proof of Existence API returns a JSON object that is used to store encrypted data on the blockchain. This JSON object can later be referenced to check the existence and integrity of the data in the future. Table 9 provides information regarding APIs that have been used to store encrypted data on blockchain.

Table 9 List of Proof of Existence APIs

Method	Endpoint	Description	Returns

POST	/store	Store encrypted data for future validation	A JSON object returns success or failure status and message.
POST	/check	To check the data using the Proof of Existence API, submit the data to the API, and it will return a Boolean value (True or False). True indicates that the provided data matches the data stored on the Polygon blockchain, confirming its integrity and existence.	A Boolean value returns either True or False, where True indicates that the payload data matches the blockchain data, False means otherwise.

12. Conclusions

The final version of this deliverable provides a detailed account of the services developed in Work Packages 2 and 3, along with their associated APIs and SDKs. It includes thorough descriptions of the Ontology and Query APIs, which assist in generating schemas for the material passport. The document also covers the Collaborative Space service, which manages users within the organisation and controls their access levels. Additionally, it describes the Environmental Analytics APIs that support the generation of sustainability reports, as well as the Circularity Calculator, which offers insights into product circularity. The final version includes complete documentation for the KG Builder and Proof of Existence APIs, along with updates to all relevant APIs and their features.

References

[1] R. T. Fielding, “Architectural Styles and the Design of Network-based Software Architectures,” University of California, Irvine, 2000.

[2] A. Rodriguez, “RESTful Web services: The basics.,” in *IBM DeveloperWorks*, 2008.

[3] O. Zimmermann, C. Pautasso and L. Frank, “Restful Web Services vs. 'Big' Web Services: Making the Right Architectural Decision,” in *17th International Conference on World Wide Web*, ACM, 2008.

[4] E. MacArthur, Towards the Circular Economy: Economic and Business Rationale for an Accelerated Transition, Cowes, UK: Ellen MacArthur Foundation, 2012.