MediaVerse

A universe of media assets
and co-creation opportunities

D4.4

Content Discovery, Copyrights Negotiation Services and Blockchain Repository v2

Project Title	MediaVerse
Contract No.	957252
Instrument	Innovation Action
Thematic Priority	ICT-44-2020 Next Generation Media
Start of Project	1 October 2020
Duration	36 months

Deliverable title	Content Discovery, Copyrights Negotiation Services and Blockchain Repository v2
Deliverable number	D4.4
Deliverable version	V1.0
Previous version(s)	D4.2
Contractual Date of delivery	31 March 2023
Actual Date of delivery	5 April 2023
Nature of deliverable	Report
Dissemination level	Public
Partner Responsible	ATOS
Author(s)	Rubén Ramiro, Antonio Calvo (ATOS), Marco Giovanelli, Marco Saltarella, Andrea Cavallaro (FINC)
Reviewer(s)	Tasos Lampropoulos, Spyros Papafragkos (ATC), Rahel Luder (STXT), Manos Schinas (CERTH)
EC Project Officer	Luis Eduardo Martinez Lafuente

Abstract	This deliverable presents a functional and technical description of the content discovery system, copyright compatibilities and blockchain inside MediaVerse.

Keywords	Creative Commons Licence, Federated Search, Payment, Blockchain, Smart Legal Contract, Copyright Management

Copyright

MediaVerse is an H2020 Innovation Project co-financed by the EC under Grant Agreement ID: 957252.

Revision History

VERSION	DATE	MODIFIED BY	COMMENTS
V0.1	07/12/2022	Rubén Ramiro (ATOS)	Table of Content & First Draft.
V0.2	27/12/2022	Marco Giovanelli, Marco Saltarella, Andrea Cavallaro (FINC)	Included Chapter 3.
V0.3	03/01/2023	Rubén Ramiro (ATOS)	Merged v0.1 with v0.2 and added content in sections 2 and 3.
V0.4	04/01/2023	Antonio Calvo (ATOS)	Added section 2.
V0.5	05/01/2023	Rubén Ramiro (ATOS)	Changed content based on FINC feedback. Also changed to a proper style and format.
V0.6	15/03/2023	Marco Giovanelli, Marco Saltarella (FINC)	Added section 4.
V0.7	21/03/2023	Rubén Ramiro (ATOS)	Changed format, cross-references, fixed figures references.
V0.8	27/03/2023	Tasos Lampropoulos, Spyros Papafragkos (ATC), Rahel Luder (STXT), Manos Schinas (CERTH)	Review, comments and suggestions
V0.9	28/03/2023	Rubén Ramiro, Antonio Calvo (ATOS)	Updated content and structure after review.
V1.0	31/03/2023	Evangelia Kartsounidou, Akis Papadopoulos (CERTH)	Final review and quality control

Glossary

ABBREVIATION	MEANING
API	Application Programming Interface
CC	Creative Commons
DAM	Digital Archive Management
DW	Derivative Work
ETH	Ethereum
Fsearch	Federated Search
HTTP	HyperText Transfer Protocol
IP	Internet Protocol
IPFS	InterPlanetary File System
IPR	Intellectual Property Rights
JS	JavaScript
MV	MediaVerse
SC	Smart Contract
SDK	Software Development Kit
SLC	Smart Legal Contract
UI	User Interface
WP	Work Package

Table of Contents

Index of Figures

Index of Tables

Executive Summary

One of the specific objectives of MediaVerse is digital content acquisition using either internal or external assets from creators such as photographers, painters, content creators, 3D artists and other types of producers. Consumers and prosumers can seek and find content inside the MediaVerse network or external content.

Section 2, *Content Discovery*, describes how the users can discover and acquire assets freely or through the MediaVerse Payment System. Section 3, *Copyright Negotiation Services*, provides a description about Derivative Works (i.e., based on already licenced works). This section manages licence compatibility to comply with the current legal framework about copyrights transparently for the user. The users will be advised and guided to choose the best license for their own creations. Finally, section 4, *Blockchain Repository v2*, shows how MediaVerse implements a blockchain-based rights management solution to protect the authenticity of the content and the rights holders.

1 Introduction

The deliverable summarises the work carried out for task T4.3 - Content discovery services and copyrights negotiation within WP4 - Agile Media Rights Management and Content Identification. In addition, it includes relevant information about all the services related to this task.

This deliverable includes the Content Discovery and Copyrights Negotiation Services sections delivered internally in M27 of the project (December 2022). In addition, the current version includes adaptations and updates on the MediaVerse blockchain compared to the first version presented in *D4.2 - Blockchain repository v1*. Hence, this deliverable will gather all the developments regarding blockchain from February 2022 until March 2023.

The implementation period of T4.3 is from M3 (March 2021) to M27 (December 2022) of the project and it covers the following topics:

- Automate the copyright negotiations and the content discovery in the context of copyright conditions and restrictions for content located in the MediaVerse, considering both the local MV node and other nodes participating in the network.
- Extend the MediaVerse capabilities for distributed content identification to integrate rights management.
- Develop software for retrieving the licence and copyrights of each content item according to the standard defined in *D4.1 - Copyright and Procedures for IPR Definition*[1], examining the compatibility of different licences for synthesising different items to create new content.
- Provide copyright suggestions for users uploading new Digital Media assets.

The project addressed these tasks by creating different microservices, as Figure 1 shows (for further information please check *D6.3 - Core Framework, Decentralized Communication and Content Exchanged v2*[2]). The microservices are integrated in each MV node and include the following features:

- **Content discovery:** DAM, ipfs_host, mv_ipfs_api and dashboard-UI.
- **Copyright negotiation:** copyright-negotiation, ipr-service, mv-slc-engine, cicero-server, mv-bcsp, mv-bcspeh, mv-eth.
- **Copyright suggestion:** license_advisor.

[1] https://mediaverse-project.eu/wp-content/uploads/2021/10/MediaVerse_D4.1_Copyright-and-Procedures-for-IPR-Definition.pdf
[2] https://mediaverse-project.eu/wp-content/uploads/2022/10/MediaVerse_D6.3-V1.0.pdf

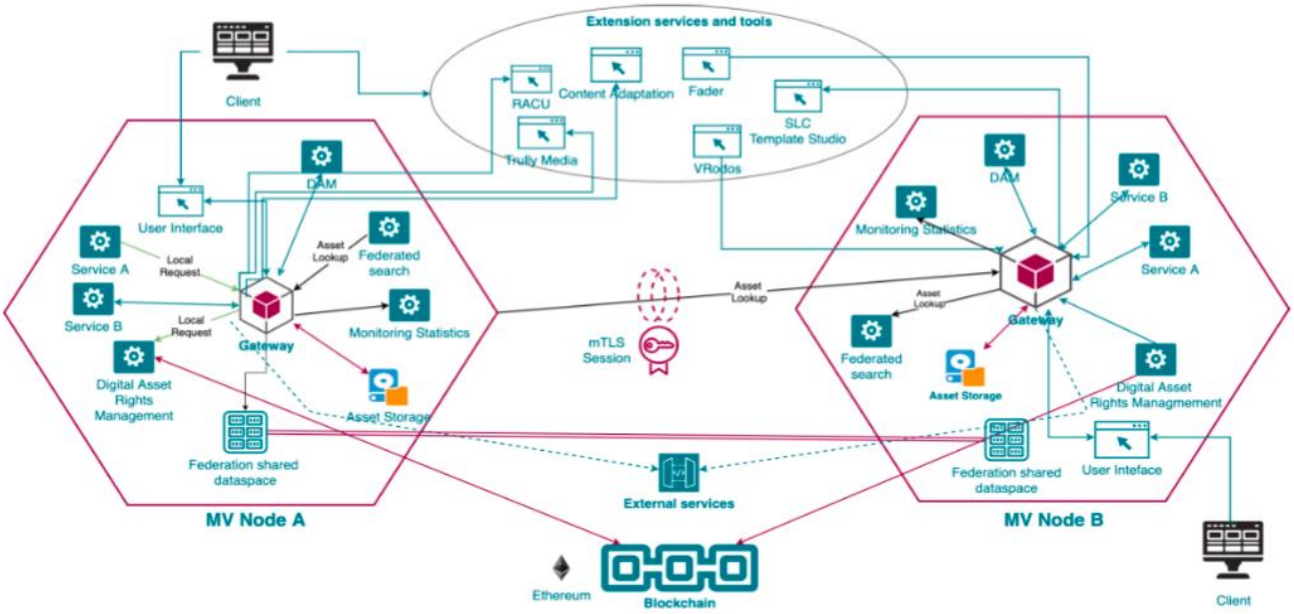

Figure 1: MediaVerse Services

2 Content Discovery

Since its conceptual specification presented in *D2.2 - Conceptual Design of the MediaVerse Framework*[3], the MediaVerse ecosystem is designed as a federation of MV Nodes, where each node is autonomous in managing its resources even if the federation cooperates according to a set of common rules. This decentralised distribution of the platform divides the content discovery into several scopes: the content discovered within the node that the user registers, the content among the network of nodes and the content offered by platforms external to MediaVerse: including authoring tools and free-to-use media repositories. These scopes considered together form a common marketplace, in which the content can be exchanged.

2.1 Content Search

Internal Node Content discovery is based on the direct interaction between the UI and the DAM API. Specifically, the UI consumes the local search DAM calls, as Figure 2 shows.

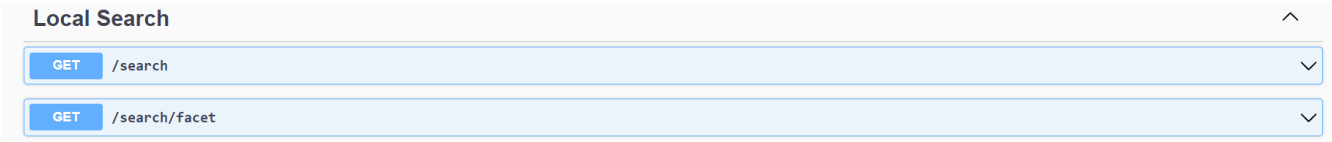

Figure 2: Local search calls in the DAM API documentation

In general, the project infrastructure follows an approach where the computational load is carried out on the server side as much as possible. Therefore, this call offers the interface to directly receive filtered or sorted elements, including pagination. The available parameters are:

- term: search term.
- sort: sort the results by one of the other attributes.
- page / per_page: to get paginated results, page number and results per page.
- media_type: Media type corresponding with the mime types[4] accepted by the platform: image, video, model, audio.
- since / until: filter by asset publication date.
- is_meme: include or exclude assets annotated as meme by the Media Verse automatic annotation service.
- is_disturbing: include or exclude assets annotated as disturbing by the Media Verse automatic annotation service.
- min_faces / max_faces: to filter 3D models according to the number of faces.

In the current version of the UI, some of these options are represented in the local search filtering section as can be seen in Figure 3.

[3] https://mediaverse-project.eu/wp-content/uploads/2021/10/MediaVerse_D2.2_Conceptual-Design-of-the-MediaVerse-Framework.pdf .

[4] *Media types*. Internet Assigned Numbers Authority. Retrieved January 11, 2023, from https://www.iana.org/assignments/media-types/media-types.xhtml

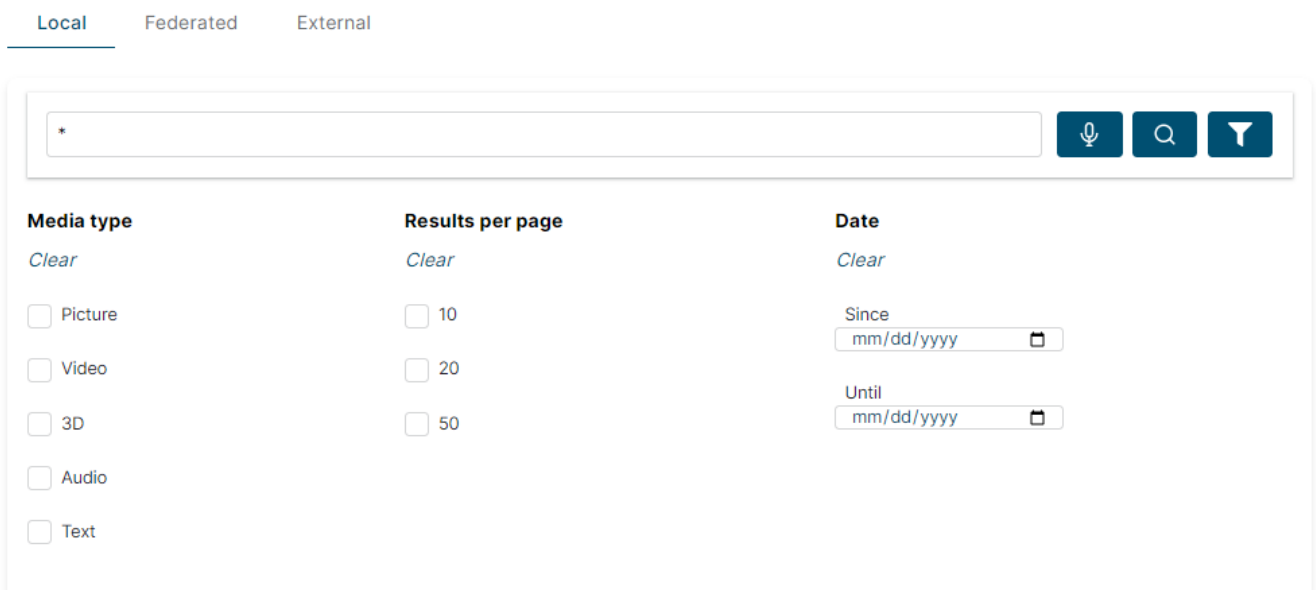

Figure 3: Local search filtering options

D6.3 - Core Framework, Decentralized Communication and Content Exchange v2[5] has already described the details of how the DAM stores digital asset files. In summary, the Storage Systems used for the persistence of the various objects (i.e., metadata, binary files, and indexes) are currently defined as follows:

- MongoDB as NoSQL Database for the metadata of the assets and other entities stored in an MV node
- Apache Solr for the full text search and indexing of assets
- Local Volume, or AWS S3 for the binary storage of assets. Following the adapter design pattern[6], the DAM could switch to support any storage implementation needed.

Apache Solr[7] is deployed in the infrastructure as another microservice included in the general docker-compose file. Solr is a search engine that utilizes an inverted index data structure to enable fast and efficient searches across large collections of documents. Its internal process can be summarized as follows:

- Document Indexing: Solr receives a set of documents in XML or JSON format, which it analyses and indexes using an inverted index data structure. During this process, Solr extracts relevant data and stores it in a way that facilitates quick and accurate searches.
- Query Processing: Users can then query the indexed data using the Solr query language, which enables complex search queries and filtering. When a query is received, Solr quickly identify relevant documents and returns the results.
- Ranking: Solr also ranks the results based on various factors, such as the relevance of the query to the document, the popularity of the document, and any custom ranking criteria defined by the user. This ranking ensures that the most relevant documents appear at the top of the search results.
- Faceting: Solr provides faceting capabilities that enable users to refine their search results based on different categories or facets, such as date ranges, product categories, or geographical regions. These facets help users to navigate and explore the search results more easily.

[5] https://mediaverse-project.eu/wp-content/uploads/2022/10/MediaVerse_D6.3-V1.0.pdf
[6] Adapter Refactoring.Guru. Available at: https://refactoring.guru/design-patterns/adapter (Accessed: January 11, 2023).
[7] Apache Solr Reference Guide. Retrieved January 3, 2023, from https://solr.apache.org/guide/6_6/index.html

As Figure 4 shows, once the answer has been sent to the interface, they are displayed in cards according to the order of relevance as computed by Solr. Once the results are displayed, users can inspect the asset details and purchase the desired assets. We describe this process in more detail in section 2.3, *Marketplace and payments*.

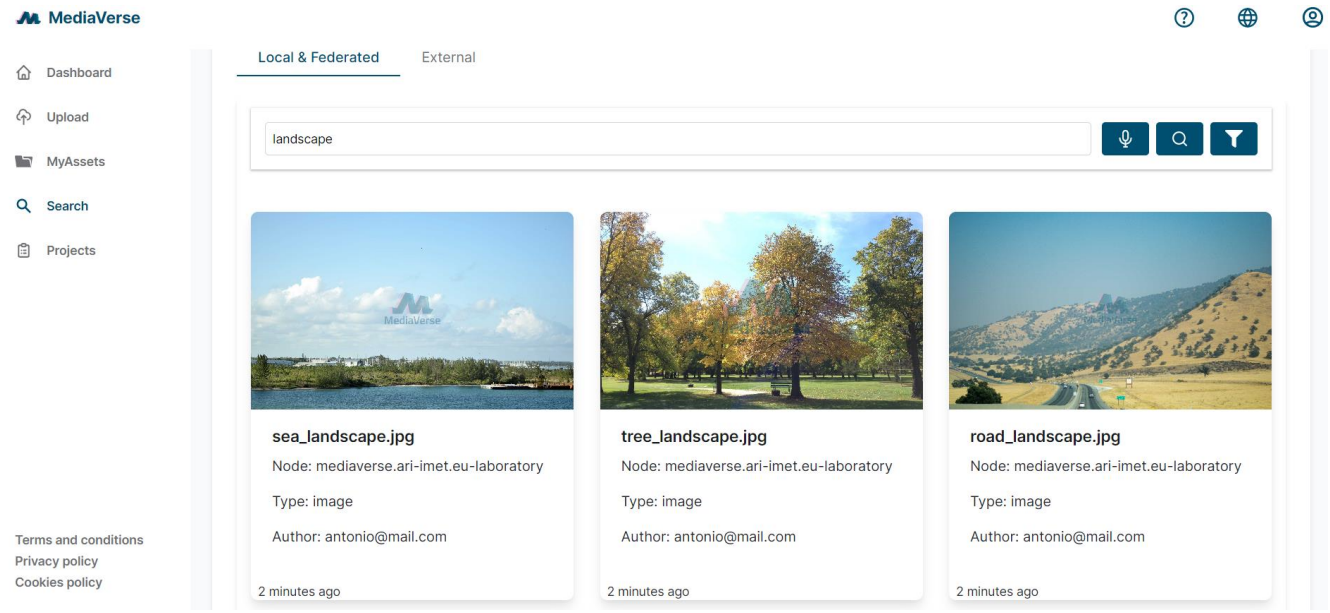

Figure 4: Local search results in MV node

2.2 Inter-node Content Discovery

One of the platform's objectives is not to confine itself to content exchange within a single node, but to be able to discover and acquire content from nodes belonging to external organisations with the least possible friction for the user. The inter-node content discovery process builds on the infrastructure described above. It adds an intermediary service, i.e., the federated search, to manage search queries and responses over the whole MediaVerse network. For this purpose, filtering, sorting, and search capabilities must be aligned across all services. The *D6.3 - Core Framework, Decentralized Communication and Content Exchange v2*[8] describes the aforementioned service's implementation and inner structure. Therefore, in this section, we will provide only a brief description.

Figure 5 shows the implementation in the user interface, where we can observe that the user has triggered the federated search tab and is able to review the details of content belonging to other nodes, in particular the "*mediaverse.ari-imet.eu"* one.

[8] https://mediaverse-project.eu/wp-content/uploads/2022/10/MediaVerse_D6.3-V1.0.pdf

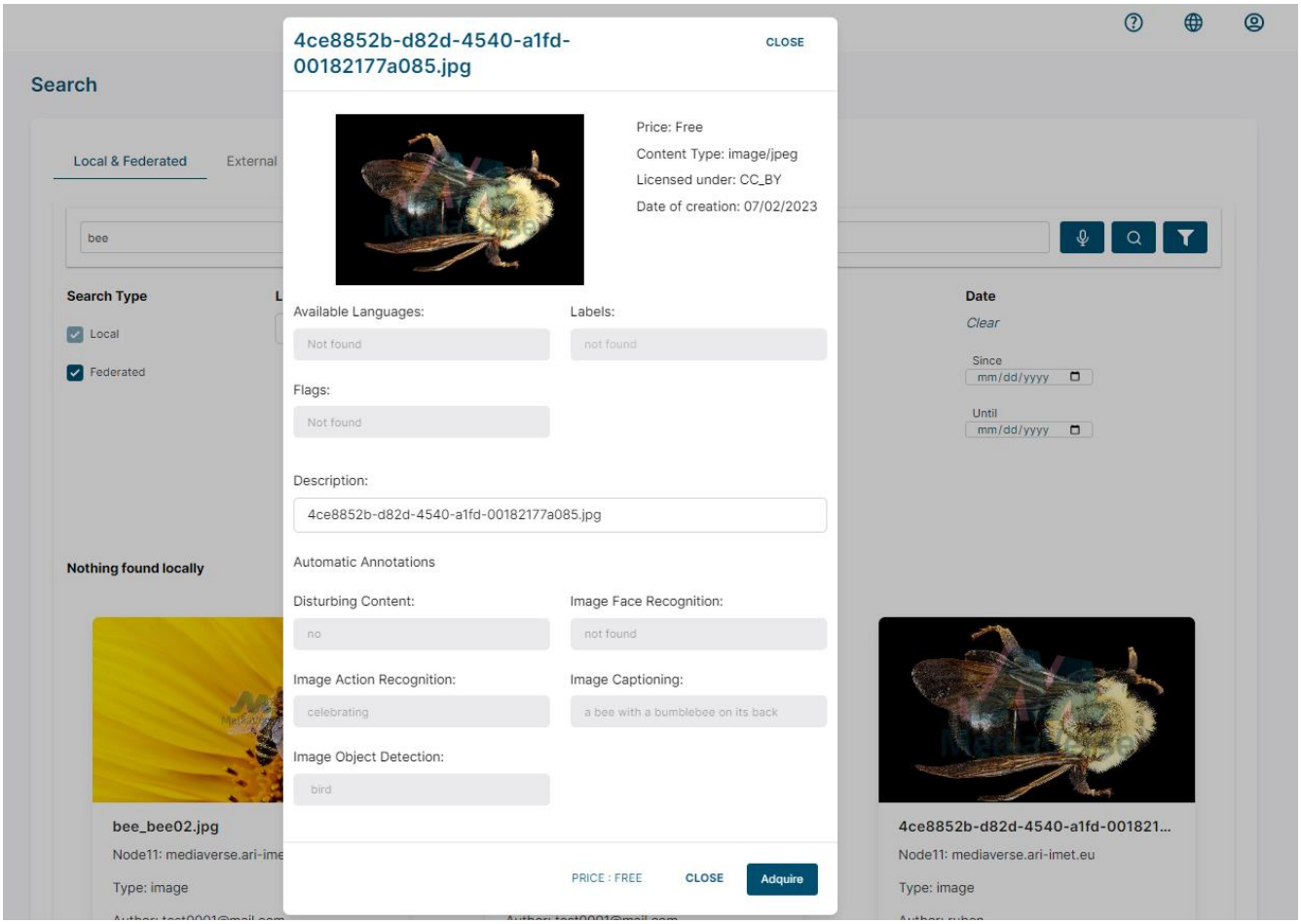

Figure 5: Inter node content discovery in the Media Verse UI

2.2.1 Federated Search in MediaVerse

The federated search service is divided in three main components:

- **Ipfs_api**: The external face of the service; it receives the requests from the interface and manages the status of the responses received by the rest of the nodes.
- **Ipfs_host**: A custom IPFS container, created to generate and manage the IPFS private network to which all nodes are connected.
- **IPFS network**: IPFS technology[9] is the basis of decentralisation in the project. The features used from IPFS's framework are the experimental publish/subscribe model for searches, the generation of a private network of nodes through a swarm key, the automatic discovery of new nodes through bootstraps and the decentralised storage.

Figure 6 shows the communication routes between these three components and the interface. As mentioned above, *D3.2 - Content Discovery and Recommendation, Annotation and Adaptation Framework*[10] describes this process in detail. After receiving a query request, the fsearch uses the same endpoint described above in section 2.1, *Internal Node Content Discovery* to receive local responses to the corresponding node's DAM.

[9] What is IPFS. IPFS Docs. Retrieved March 28, 2023, from https://docs.ipfs.tech/concepts/what-is-ipfs/
[10] https://mediaverse-project.eu/wp-content/uploads/2022/07/D3.2-V1.0.pdf

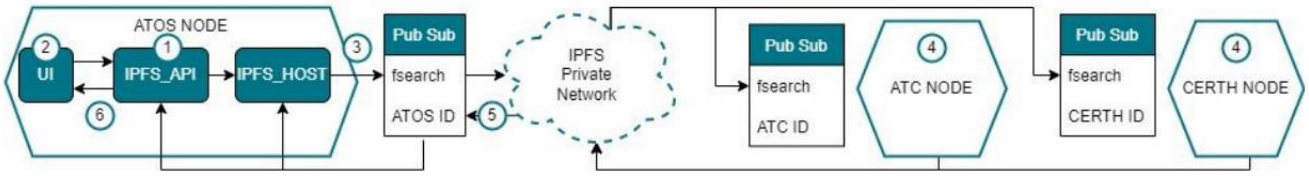

Figure 6: fsearch components communication

2.2.2 Copyright Negotiations in the Federated Data Space

As mentioned in deliverable *D6.3 - Core Framework, Decentralized Communication and Content Exchange v2*[11], the copyright negotiations service needs certain information about each asset to work. This information must always be available, even when the node where the licence was registered is down.

At the application level, this information is stored in two different SLCs for two different purposes:

- **Ownership Deed (OD)** is an SLC where it is stated that the user/s is/are the sole owner/s of the copyrighted work (whether as an author/s or after transfer of such rights).
- **Copyright Licence (CL)** is an SLC where it is stated that the licensor/s (i.e., the owner/s) licence/s to a licensee (i.e., a specific user) the right to use the copyrighted work in a certain way.

D6.3 focused on the interaction between the different microservices of the platform for the process of generating and sending the SLC to the federated dataspace. To avoid repeating the same content, this deliverable focuses on the interaction between the DAM and the federated dataspace, describing, in particular, the process of pinning. When a node DAM uploads a SLC to IPFS, it is broken down into smaller chunks, and each chunk is given a unique hash that serves as the file's address. When other nodes DAMs request the file, they can retrieve it using its hash, which allows to access the SLC from any node on the network that has a copy of it. This decentralized approach to file storage and sharing eliminates the need for a central server, reduces the risk of censorship, and improves file availability and redundancy. Figure 7 shows the sequence diagram for the asset acquisition process concerning SLC federated storage.

The nodes of an IPFS network automatically cache documents, they download and keep them available to other nodes. Since storage is a finite resource, it is needed to clear out previously cached resources to make room for new resources. IPFS uses garbage collection to free disk space on IPFS nodes. Unfortunately, IPFS triggers the garbage collection process on node files by default. To prevent the SLC from being lost after a server shutdown and achieve the aforementioned redundancy and availability, it is necessary to perform the pinning process.

During asset registration, the DAM pins content to original IPFS node using the corresponding option as in the following example:

```
curl -X POST -F file=@slcfile " http://<ipfs_host service
name>:5001/api/v0/add?pin=true&wrap-with-directory=false"
```

[11] https://mediaverse-project.eu/wp-content/uploads/2022/10/MediaVerse_D6.3-V1.0.pdf

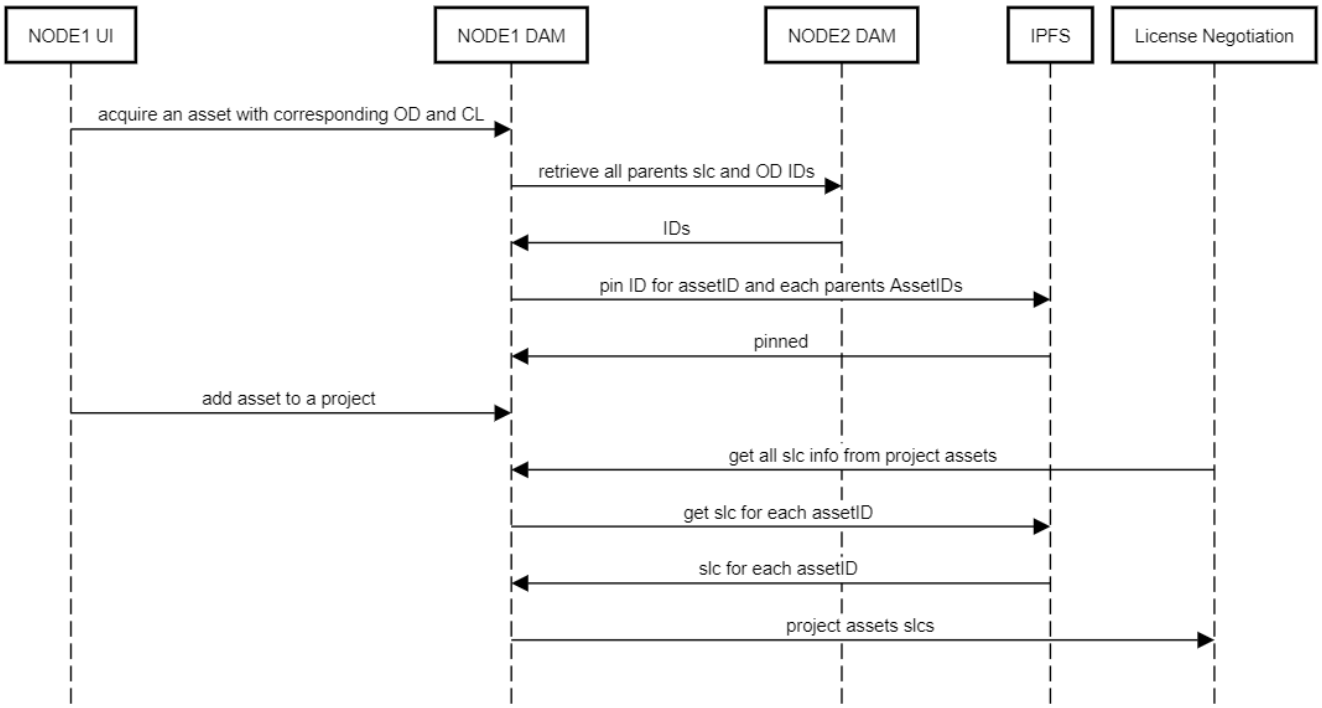

Figure 7: SLC management during asset acquisition

The DAM receives a hash corresponding with the IPFS ID of the SLC:

```
{
"Name": "test_slc.json",
"Hash": "QmaUrqYoahnKCyAsRHLrcz5PZJtBRGWFT7kgaBYvQurfDk",
"Size": "19"
}
```

The hashes of both SLC (OD and CL) are stored in the DAM inside the asset model:

```
slcHashIds: ["QmaUrqYoahnKCyAsRHLrcz5PZJtBRGWFT7kgaBYvQurfDk",
"QmUsaSjaKJYzL4hJCBmkXkGy2PfMGw2J4w3uYGi9WHrRjh"]
```

When the asset metadata is transferred from one node to another:

- If the first node (on which the asset was registered) is available, triggering the “cat” call from any node to its own ipfs_host would retrieve the SLC information:

```
curl -X POST "http://<ipfs_host service
name>:5001/api/v0/cat/QmaUrqYoahnKCyAsRHLrcz5PZJtBRGWFT7kgaBYvQurfDk"
```

- If any second node has previously requested the SLC, it will be available in memory even if the original node (on which the asset was registered) is not available, until IPFS triggers a garbage collection. To ensure that the SLC will be always available, the DAM pins the file using:

```
curl -X POST " http://<ipfs hosts service name>:5001/api/v0/pin/add/
QmaUrqYoahnKCyAsRHLrcz5PZJtBRGWFT7kgaBYvQurfDk"
```

In this case, the DAM will receive the following IPFS response:

```
{
  "Pins": [
    "maUrqYoahnKCyAsRHLrcz5PZJtBRGWFT7kgaBYvQurfDk"
  ]
}
```

Once the SLCs are protected by pinning, the DAM will connect any request for the content of a SLC to a “cat” call to IPFS, being the storage in the federated dataspace completely transparent to the licence negotiation service or any other.

2.3 External Content Discovery

2.3.1 Free-to-use Media Repositories

In order to populate the MediaVerse nodes with content, we proposed that the user can access content from free-to-use media repositories, so that in the future, all these repositories can be accessed, and results can be ranked from the same interface. For this purpose, we have created a first approach where the user can discover content belonging to Wikimedia-commons. To use it, the UI calls the following endpoint created in the DAM for this purpose:

```
curl -X GET
"https://<mediaverse_node_domain>/dam/embedded_search?page=0&per_page=10&media_type
=image&query=cat"
```

The Wikimedia-commons content is accessed and integrated in the UI, and some basic filtering can be performed (see Figure 8).

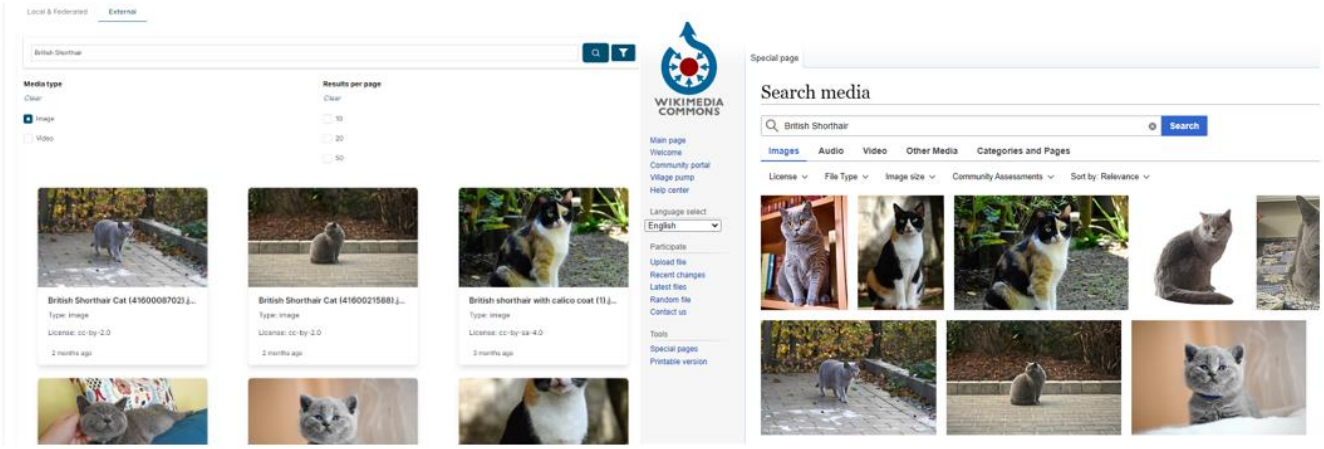

Figure 8: Content search in Wikimedia Commons

2.3.2 Authoring Tools

Another source of external content to MediaVerse would be the platforms that we have designated in the project as Authoring tools (i.e., Fader, VRodos, NerStar).

The authoring tools are connected with the MediaVerse platform through the concept of a project. Users of the MediaVerse platform have the ability to create a project and upload files to it, subsequently gaining access to these projects when logging into the Authoring tools platforms. Within these external platforms, users may peruse content, incorporate it into the project, and produce a new media file. If the resulting media file is compatible with the platform (only files that conform to audio, video, video360, text, images, or 3D model formats) it may be licensed and sold within MediaVerse. If the resulting media file is not compatible, the content discovery process is the same, but the project result will be stored in the original Authoring tool, being registrable in MediaVerse. As an example, Figure 9 depicts the content discovery functionalities in both Fader and VRodos.

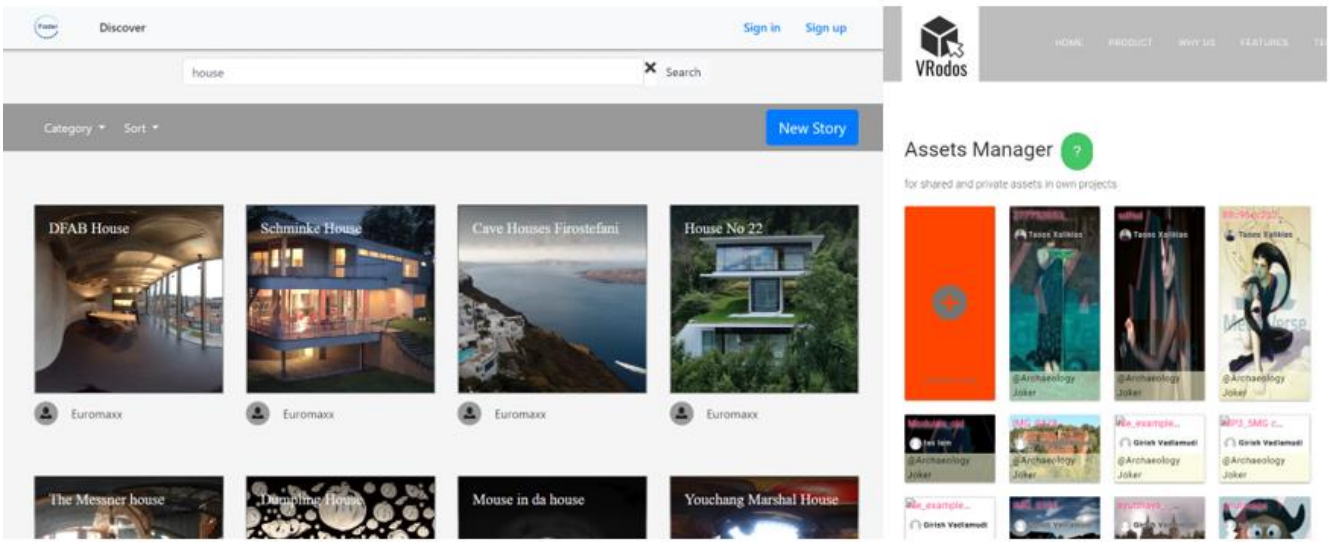

Figure 9: Content discovery in Fader and VRodos

2.4 Marketplace and Payments

2.4.1 Marketplace

As mentioned, MediaVerse allows the user to purchase rights upon an asset. Depending on the assets' source, the user will be able to acquire rights to an asset through the following options:

1. The MV Node, where the user is already registered.
2. Other nodes, i.e., through the MV Network and IPFS.
3. External repositories such as Wikimedia.

2.4.2 Payment Options

There are different options to proceed with the payments, which we describe in the following items:

1. **MVCoins**: use of dedicated tokens to keep track of economic transactions through the Blockchain technology. The tokens are fungible, representing value and held by users' Blockchain Wallets. For instance, in Ethereum, the selected Blockchain platform for the MediaVerse project, the cryptocurrency is the Ether (ETH), but it is possible to define other custom units of account using fungible tokens. In MediaVerse, it has been proposed the MVCoin, a fungible token with an arbitrary value in Euro (e.g., 1 Euro = 1.000 MVCoin). With this approach, the MediaVerse platform leaves out the real money scenario, and the user directly uses the MVCoins.
2. **MV Bank Account**: the user tops up his/her balance by paying the MV Node with Fiat Currency. Hence, each payment appears in Fiat Currency to the user, but the MV Node executes them behind the scenes in MVCoins.
3. **Peer-to-Peer Payments**: when the user purchases a right on an asset, the transaction is performed in Fiat Currency from the user directly to one or multiple owners, using technologies like PayPal Express Checkout[12]. The MV Node just registers the transaction identifier on the Blockchain.
4. **Centralised Payments**: when the user purchases a right on an asset, the transaction takes place in Fiat Currency from the user to the MV Node. Then, the MV Node performs the transactions in Fiat Currency directly to the relative owners (e.g., *PayPal Payouts*[13]) and registers the transaction identifiers on the Blockchain.
5. **One-to-One PayPal**: when the user purchases a right on an asset, the transaction is performed in Fiat Currency directly between the user and the seller (i.e., only one of the owners).

All the aforementioned approaches have not only advantages, but also limitations that discourage us to follow them within MediaVerse. For instance, in "Centralised Payments" and "MV Bank account" methods, MediaVerse will operate the complete procedure, which might trigger the qualification of MediaVerse as an e-money service provider or payment service provider. This would require an official licence as PI/EMI under PSDII regulation. Such a license procedure is heavy, expensive, and lengthy, implying ongoing national bank/financial institution monitoring afterwards. Moreover, "Peer-to-peer payments" is a complex and dedicated solution with development needs that go far beyond the scope of the project. Of the five options, "One-to-One PayPal" fits better in MediaVerse, after a detailed analysis and an in-depth testing, which we describe in more detail below.

[12] PayPal Express Checkout: https://developer.paypal.com/api/nvp-soap/set-express-checkout-nvp/

[13] PayPal Payouts: https://developer.paypal.com/docs/payouts/

One-to-One PayPal Option

We explored the One-to-One PayPal option as a solution to asset payments relying on some widely known platform and reliable service.

The MediaVerse Node acts as a communicator between the buyer, PayPal, and the seller. The buyer clicks on the “buy asset” button and then the MV Node communicates with PayPal, which displays a login and checkout window to the buyer. When the buyer confirms the purchase, PayPal receives the one-time payment sent directly to the seller's account. It is important to highlight that this method consists of a single payment between the buyer and the seller. It is not contemplated for micropayments or profit sharing when the asset has multiple owners, which is a limitation of this approach. In Figure 10, we observe how the communication among the user (buyer), the MediaVerse interface, and the external PayPal service, is established.

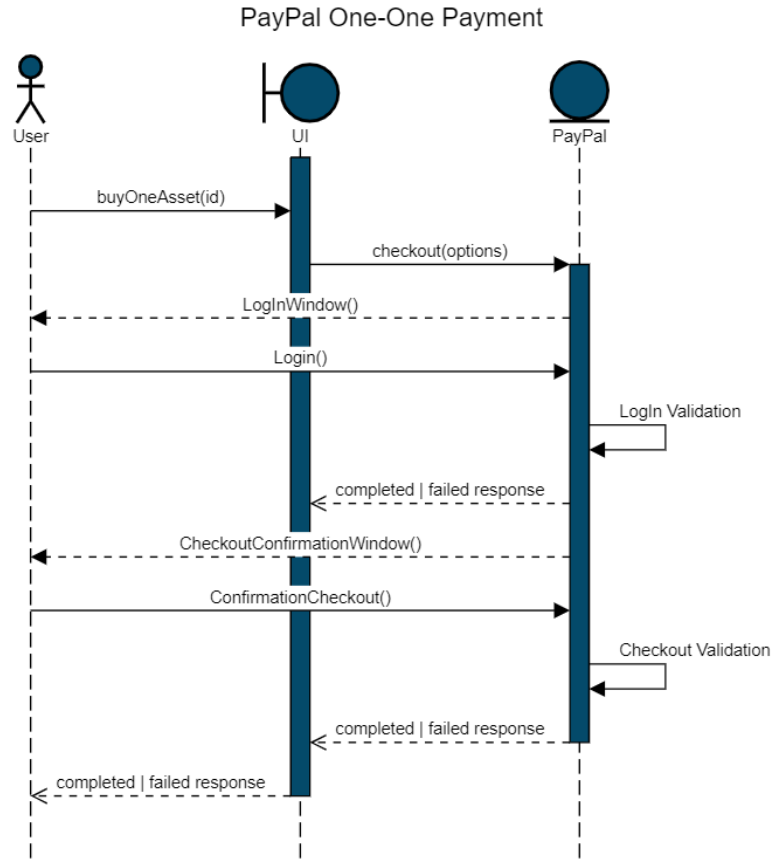

Figure 10: PayPal One-to-One Payment Sequence Diagram

This option offers the user a good experience and it needs only 2-3 clicks to purchase a right on an asset. Therefore, the sequence is:

1. The user clicks on the "Buy with PayPal" button, as shown in Figure 11.
2. PayPal sends the user a login window (the user does not need to perform this step if he/she has previously performed another transaction because he/she has already logged in). The user enters his/her PayPal account credentials.
3. PayPal sends the confirmation window called checkout. The user can check the details of the purchase (see Figure 12).
4. The user clicks on "Complete Purchase", and the purchase finishes with the asset acquisition.

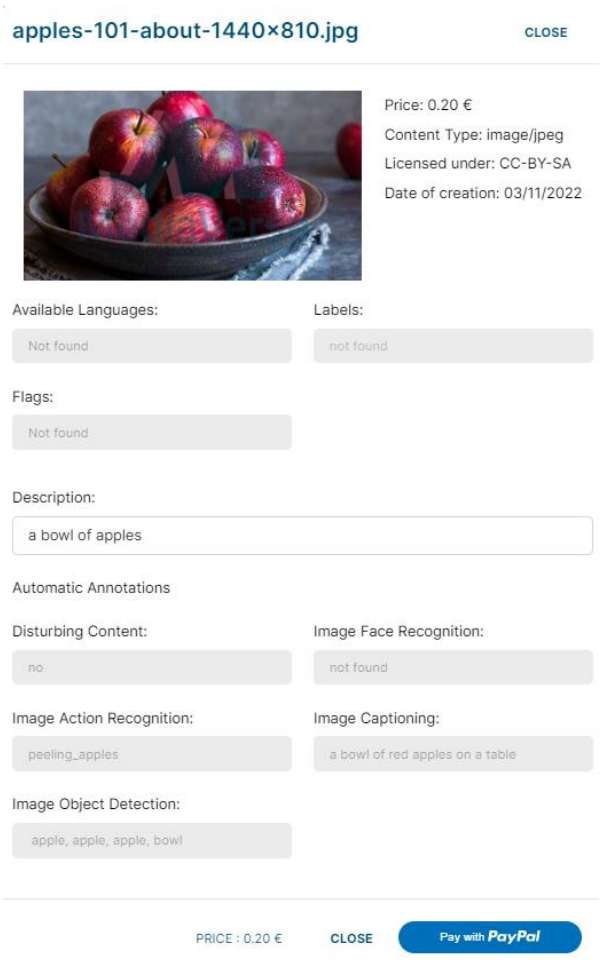

Figure 11: Asset details in a Federated Search

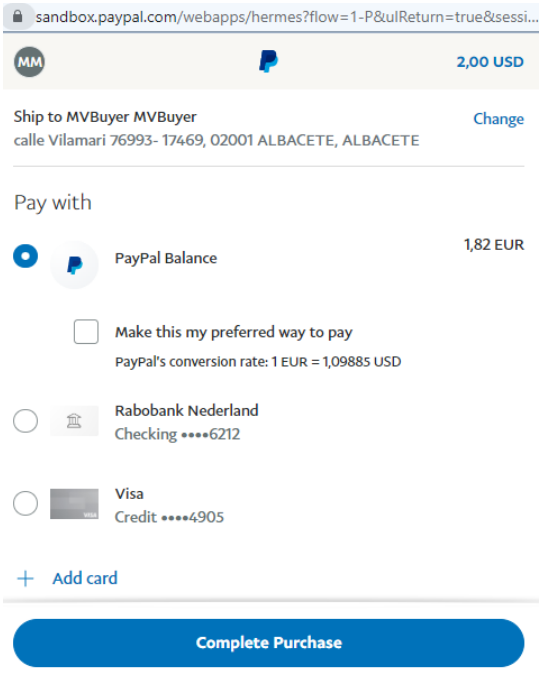

Figure 12: PayPal Checkout Confirmation Window

The requirements for this payment method are:

1. The buyer shall have a regular PayPal account to make purchases.
2. The seller shall have a PayPal business-merchant account to sell.

Merchant Account and PayPal Fees

At this point, it is worth mentioning that using a payment platform within MediaVerse is not free of charge, PayPal applies several fees to the seller[14]. In Europe, this fee corresponds to a percentage per transaction plus a fixed amount:

- 2.9%-3.4% + ≈ 0.35€

However, there are no fees per transaction for the buyer, transaction fees are applied only to the seller. Figure 13 shows an example of transaction fees applied to the seller.

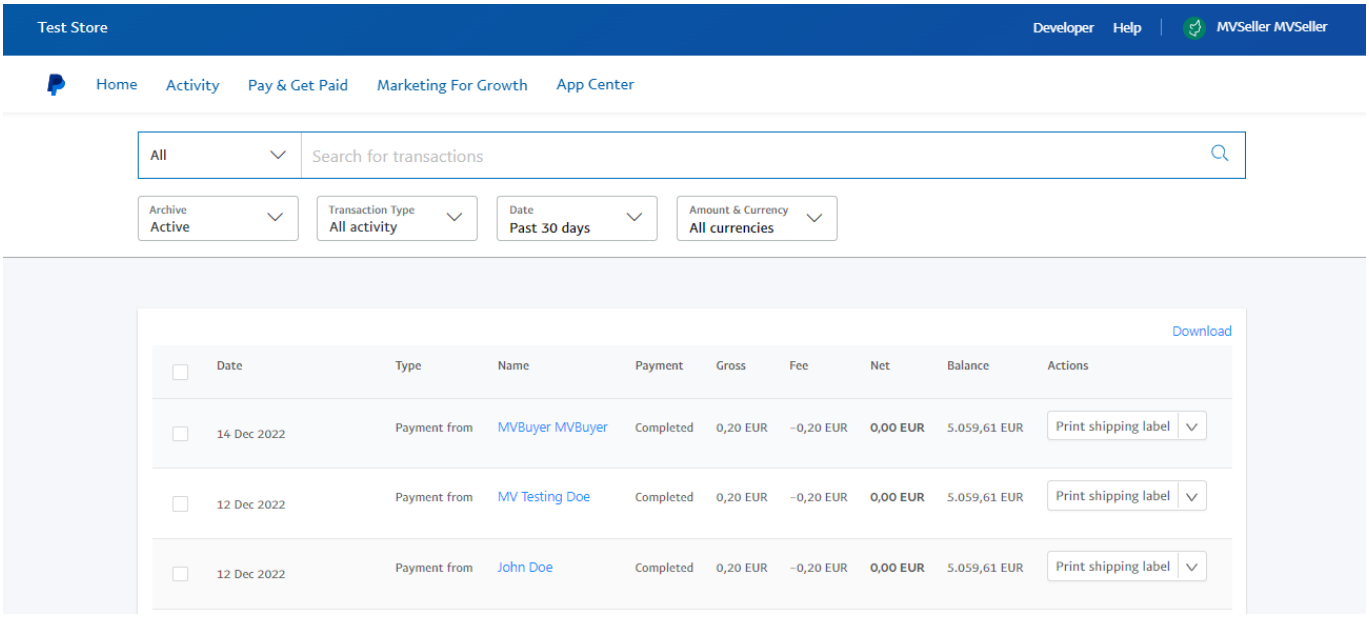

Figure 13: Seller Transaction Activity Tab inside PayPal

Technical Solution and Abstraction for Developers

Taking advantage of the React Framework for the implementation of the MediaVerse UI, the *@paypal/react-paypal-js*[15] library has been used for the integration of the One-to-One payment solution. From a developer's point of view, the One-to-One PayPal would be an ideal payment solution, since it is completely abstracted from the PayPal JS SDK[16] and its API[17], saving considerable time in development, testing and integration. The React library offers to the developers:

- Enforce async loading of the JS SDK upfront so that the buttons render immediately.
- Each buyer can use its currency, and PayPal will automatically exchange it for the seller's currency.

[14] PayPal Merchant Fees: https://www.paypal.com/be/webapps/mpp/merchant-fees?locale.x=en_BE
[15] React-paypal-js npm package: https://www.npmjs.com/package/@paypal/react-paypal-js
[16] PayPal JS SDK: https://developer.paypal.com/sdk/js/configuration/
[17] PayPal API: https://developer.paypal.com/api/rest/

- Support actions to reload the JS SDK and re-render components when global parameters (like currency) change.

Code and Details

The React library has two main components:

1. **PayPalScriptProvider:** this context provider manages the JS SDK script loading and uses the Context API[18] for managing state and communicating to child components. It is essential to note that the auto-reload functionality is triggered when the parameters change.
2. **PayPalButtons:** this SDK is used to render the UI for PayPal products served by the SDK of JS.

The **PayPalScriptProvider** `options` are used to configure the JS SDK to accept an object for passing query parameters and data attributes to the JS SDK script.

- **Client-id**: the seller's token sets the PayPal target account. The seller itself shall assign it beforehand in the user MediaVerse profile.
- **Currency**: the fiat currency the buyer uses to buy an asset. Each user uses the fiat currency he/she previously assigned in the profile.
- **Intent**: determines whether the charges are applied immediately on checkout or if the buyer authorises to apply them later.
- **Locale**: the language and display formats (e.g., time, date, etc.) used for components. By default, PayPal automatically sets the buyer's locale.

The code shown below allows the front-end to set the PayPal account with the buyer's fiat currency, based on his/her profile (see Figure 14), the amount of the transaction, and the country. It sets the PayPal account of the seller (target account) using the PayPal seller token.

```
const initialOptions = {
    "client-id":
      "seller-token",
    currency: "EUR",
    intent: "capture",
    locale: "en_US",
 };
 <PayPalScriptProvider
    options={initialOptions}>
<PayPalButtons
      style={style}
      fundingSource="paypal"
```

[18] React Context API: https://reactjs.org/docs/context.html

```
        createOrder={}
        onApprove={}
         onError={}
               onCancel={}
/>
</PayPalScriptProvider>
```

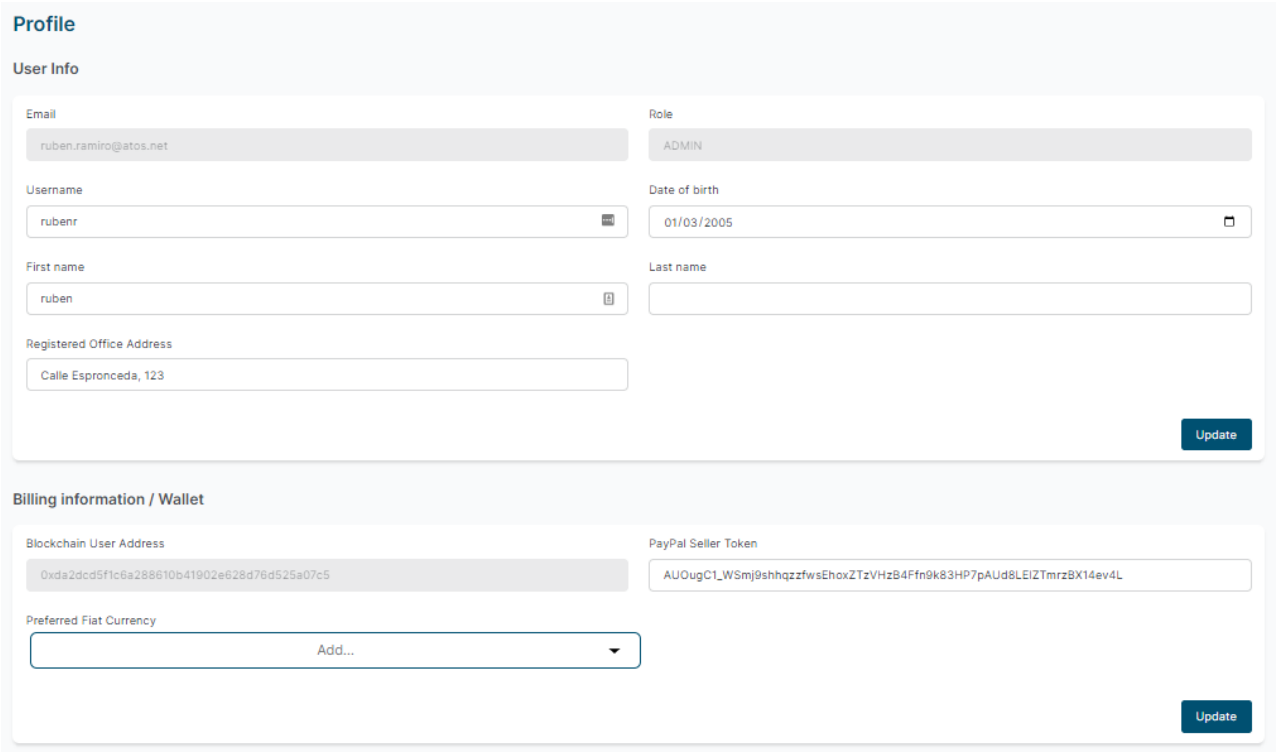

Figure 14: User Profile

3 Copyright Negotiation Services

The Copyright Negotiation in MediaVerse enables the licencing of Derivative Works – i.e., works based on already licensed works (e.g., a video that contains a licensed soundtrack) – under the Creative Commons (CC) licences. This is made possible through the use of different components: the Accord Project Smart Legal Contracts (SLCs) that formalise the licences, the Licence Comparator that compares and combines the licences, the Rights Management that analyses the licences hierarchy and the Licence Advisor, that guides the user in the selection of the proper licencing. The following sections will provide an overview of the implementation of these components in MediaVerse.

3.1 Types of Licences for Digital Assets

The Copyright Negotiation in MediaVerse currently supports the Creative Commons (CC) licences. These licences "give everyone from individual creators to large institutions a standardised way to grant the public permission to use their creative work under copyright law"[19].

There are several CC licence types (for further details, please refer to the CC website[20]):

- **CC 0 4.0** (also known as "CC Zero"): This licence represents the public dedication of the work under no conditions.
- **CC BY 4.0**: This licence requires that the credit shall be given to the creator ("BY").
- **CC BY-SA 4.0**: This licence requires that the credit shall be given to the creator ("BY") and that adapted or derivative works shall be shared under the same terms ("SA").
- **CC BY-NC 4.0**: This licence requires that the credit shall be given to the creator ("BY") and that the work shall be used only for non-commercial scopes ("NC").
- **CC BY-NC-SA 4.0**: This licence requires that the credit shall be given to the creator ("BY") that the work shall be used only for non-commercial scopes ("NC") and that adapted or derivative works shall be shared under the same terms ("SA").
- **CC BY-ND 4.0**: This licence requires that the credit shall be given to the creator ("BY") and that adapted or derivative works shall be avoided ("ND").
- **CC BY-NC-ND 4.0**: This licence requires that the credit shall be given to the creator ("BY") that the work shall be used only for non-commercial scopes ("NC") and that adapted or derivative works shall be avoided ("ND").

To represent these licences, we use the SLCs. As previously explained in *D4.1 - Copyright and Procedures for IPR Definition*[21] and *D4.2 – Blockchain Repository v1*, an SLC consists of the combination of the SLC Template and SLC Data. In order to show the potentiality of the MediaVerse platform, eight SLC Templates are used (see Table 1). Seven SLC Templates are based on the CC original "Plain Text" of the corresponding CC licence type[22]. The eighth one is the "MV Copyright Licence" SLC Template, that can resemble CC licences (we called those "CC Like" licences, see Annex I).

[19] https://creativecommons.org/about/cclicenses/

[20] https://creativecommons.org

[21] https://mediaverse-project.eu/wp-content/uploads/2021/10/MediaVerse_D4.1_Copyright-and-Procedures-for-IPR-Definition.pdf

[22] CC 4.0 licences: https://creativecommons.org/2014/01/07/plaintext-versions-of-creative-commons-4-0-licenses/

Regardless of which version of SLC Templates is used, the associated SLC Data for the same type of CC licence is identical (e.g., the SLC Data for CC BY 4.0 is the same both for CC BY SLC Template and MV Copyright Licence SLC Template), to potentially enable different types of comparison (see Table 1). However, due to the requirement that adapted or derivative works shall be shared under the same terms ("SA"), the "CC Like" versions of the CC BY-SA 4.0 and CC BY-NC-SA 4.0 cannot be created, because they cannot be considered compatible/equivalent with the original CC version of the licences. The SLC information and the relative metadata are stored in the DAM (for further details see Annex III: SLC information and SLC metadata).

Table 1: Comparison of CC and "CC Like" version of the licences

<table>
<tr><th rowspan="2">CC Licence Type</th><th colspan="2">CC</th><th colspan="2">"CC LIKE"</th></tr>
<tr><th>SLC Template</th><th>SLC Data</th><th>SLC Data</th><th>SLC Template</th></tr>
<tr><td>CC 0 4.0</td><td>CC 0
SLC Template</td><td colspan="2">CC 0
SLC Data</td><td rowspan="7">MV Copyright
Licence
SLC Template</td></tr>
<tr><td>CC BY 4.0</td><td>CC BY
SLC Template</td><td colspan="2">CC BY
SLC Data</td></tr>
<tr><td>CC BY-SA 4.0</td><td>CC BY-SA
SLC Template</td><td>CC BY-SA
SLC Data</td><td>N/A</td></tr>
<tr><td>CC BY-NC 4.0</td><td>CC BY-NC
SLC Template</td><td colspan="2">CC BY-NC
SLC Data</td></tr>
<tr><td>CC BY-NC-SA 4.0</td><td>CC BY-NC-SA
SLC Template</td><td>CC BY-NC-SA
SLC Data</td><td>N/A</td></tr>
<tr><td>CC BY-ND 4.0</td><td>CC BY-ND
SLC Template</td><td colspan="2">CC BY-ND
SLC Data</td></tr>
<tr><td>CC BY-NC-ND 4.0</td><td>CC BY-NC-ND
SLC Template</td><td colspan="2">CC BY-ND
SLC Data</td></tr>
</table>

3.2 Licence Comparator

Regarding the objective of abstracting the complexity for the user, task T4.3[23] offers to the user automatic compatibility checking between CC licences when creating a Derivative Work (DW) - i.e., the output of a new project with multiple assets with different sources and authors.

Derivative Work licence compatibility: the licence of a Derivative Work (called "adapter's licence" by CC) shall be compliant with the licences that apply to the embedded works. As an example, CC provides an *adapter's licence chart* (Figure 15), which details the CC licences that can be used as Derivative Work licence. An embedded work under the CC licence in the left-hand column forces one to choose an adapter's licence with a green box. Licences with yellow boxes are not recommended, because they need additional care. The licences under the dark grey boxes cannot be used as adapter's licence.

[23] T4.3 Content Discovery services and Copyrights Negotiation API, Swagger: https://mediaverselab.ari-imet.eu/copyright/api-docs/#/

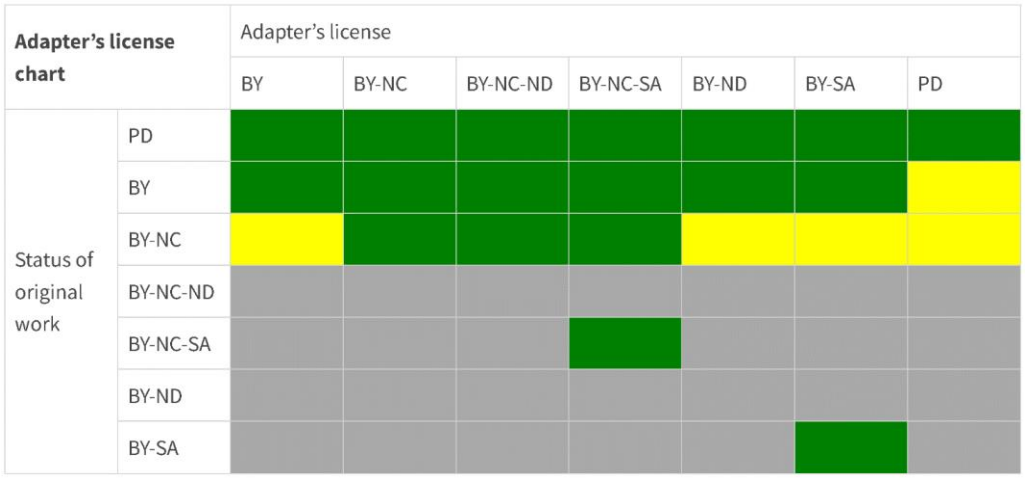

Figure 15: Adapter's licence chart (from Creative Commons FAQ[24])

The following scenario is an example of how the Licence Comparator service works. When a user creates a derivative work with ***"CC_BY"*** and ***"CC_BY_NC"*** licence and he/she needs to know which licences are compatible with each other.

Looking at the matrix, in Figure 16, if ***"BY"*** (blue) is combined with ***"BY_NC"*** (red), the "derivative work" licence shall be chosen among the columns where both ***"BY"*** and ***"BY_NC"*** are on a green background. Therefore, for ***"CC_BY"*** and ***"CC_BY_NC"***, the licences compatible to choose for the derivative work are: ***"BY_NC", "BY_NC_ND",*** *and* ***"BY_NC_SA".*** These results make sense, because the feature ***Non-Commercial (NC)*** excludes commercials licences as ***"BY", "BY-SA",*** and ***"BY-ND".***

Another particular case is the ***Share Alike (SA)*** feature. All the licences with this feature will be compatible only with the same type of licence. Therefore, for example, ***"BY_SA"*** only can be merged or mixed with ***"BY_SA"*** licences. In the case of ***"BY_NC_SA",*** only with ***"BY_NC_SA"*** licences (see Figure 16).

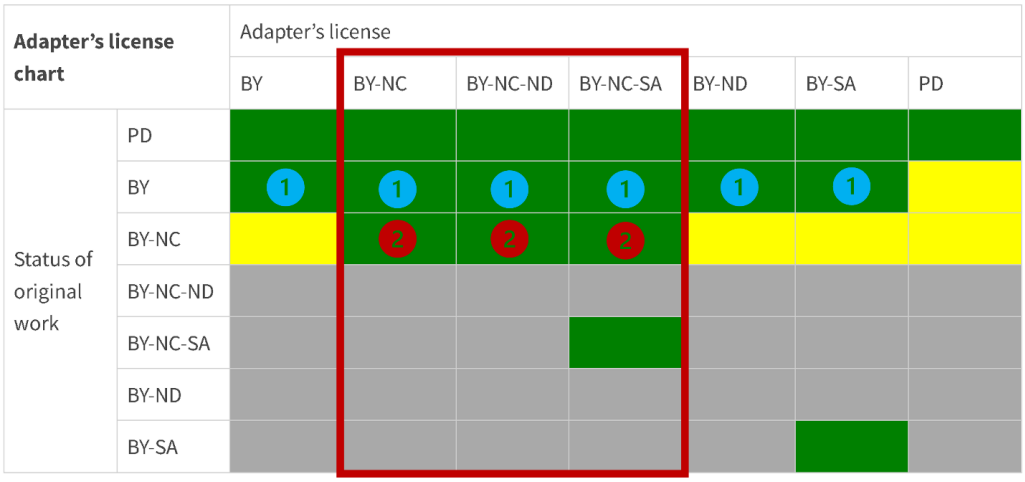

Figure 16: Licence compatibility when a CC_BY and a CC_BY_NC licences are merged

[24]Creative Commons FAQ: https://creativecommons.org/faq/#if-i-derive-or-adapt-material-offered-under-a-creative-commons-license-which-cc-licenses-can-i-use

3.3 Rights Management

Through the support of the Licence Comparator, the Rights Management enables the licensing of Derivative Works. In particular, to address the "Embedded Works licences compatibility" and "Derivative Work licence compatibility" shown in the previous section, the Rights Management implements a solution capable of semi-automatically analysing the licences tree and providing the retrieved licences to the Licences Comparator.

Figure 17 shows an example of licences tree. Each asset could have an Ownership Deed (OD, in blue) and a linked Copyright Licence (CL, in yellow). In this case, the Derivative Work "Asset_09" embeds "Asset_05", "Asset_06", "Asset_07", "Asset_08":

- "Asset_07" specifies a Copyright Licence (CL), so it can be automatically compared with the License Comparator.
- "Asset_06" is licensed, but it is owned by the user (it owns the OD), so its licence can be ignored.
- "Asset_05" is a Derivative Work owned by the user (DW OD), so the licences of its embedded works shall be manually recursively compared, and it is up to the user to choose a compliant licence.
- "Asset_08" is not registered on MediaVerse (NO OD/CL), so it is up to the user to manually choose a compliant licence.

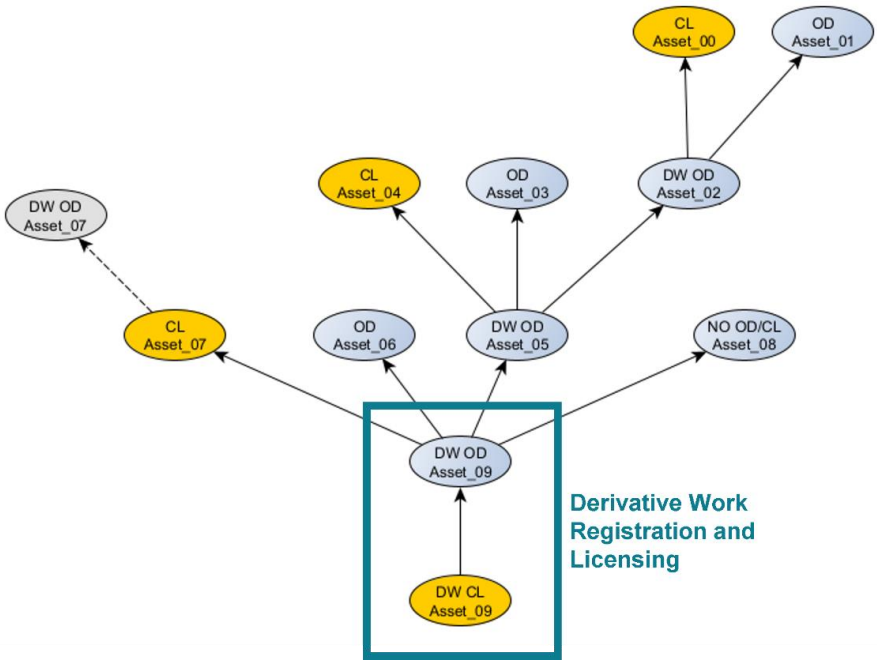

Figure 17: An example of a licences tree

The system can pre-emptively make the user aware in advance about the licensing implications of incorporating a work into a new creation, during the Derivative Work process itself. This helps users determine if it is feasible to use the work, rather than waiting until the end when registering and licensing new work.

3.4 Licence Advisor

The user may find himself in the situation of doubting between the CC licences offered by MediaVerse to register the assets. To clarify these doubts, this functionality offers the user a simple quiz in which, through a series of questions, a CC licence is recommended (Figure 18).

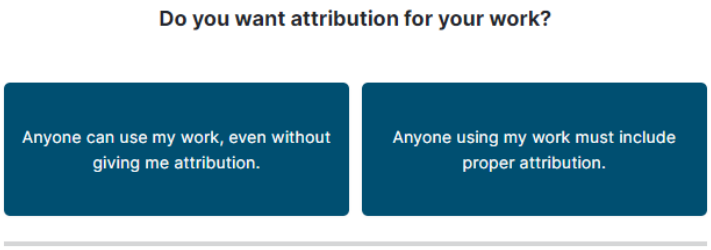

Figure 18: Quiz question example.

The quiz is about four questions, regarding the important features of CC licences. After these four questions, MV offers the user a final recommendation based on his preferences (Figure 19).

1. **Do you want attribution for your work?**

Possible user response (Only one response is possible):

a. Anyone can use my work, even without giving me attribution. (No)
b. Anyone using my work must include proper attribution. (Yes)

In the affirmative case, this fulfils the **Attribution (BY)** feature.
In the negative case, this fulfils the **CC0 (Public domain)** feature.

2. **Do you want to allow others to use your work commercially?**

Possible user response:

a. Others can use my work, even for commercial purposes. (Yes)
b. Others cannot use my work for commercial purposes. (No)

In the negative case, this fulfils the **Non-Commercial Use (NC)** feature.

3. **Do you want to allow others to remix, adapt, or build upon your work?**

Possible user response:

a. Others can remix, adapt, or build upon my work. (Yes)
b. Others may only use my work in unadapted form. (No)

In the negative case, this fulfils the **Non-Derivative Works (ND)** feature.

4. **Do you want to allow others to share adaptations of your work under any terms?**

Possible user response:

a. Others can share adaptations of my work under any terms. (Yes)
b. Others must use the same CC licence if they adapt my work. (No)

In the negative case, this fulfils the **Share Alike (SA)** feature. Therefore, the user needs to set the licence under the same licence as the original.

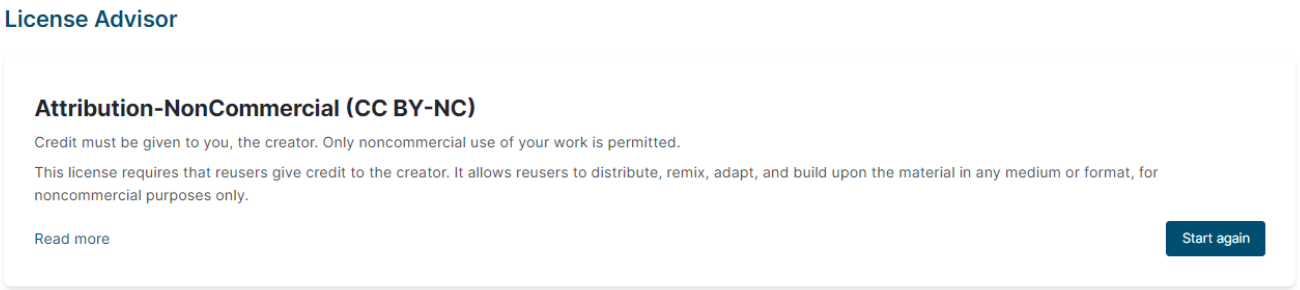

Figure 19: Licence recommended example

4 Blockchain Repository v2

This section describes the advancements in the MediaVerse developments related to the blockchain and rights management concerning deliverable *D4.2 - Blockchain Repository v1*, with a particular focus on the newly implemented features of the Rights Management component and the technical aspects.

4.1 Rights Management

In the following section, we describe the implementation of the Rights Management features that enable the MediaVerse platform to deal with rights and their "notarisation" on the blockchain (for further details, please refer to *D4.1 - Copyright and Procedures for IPR Definition*[25]).

To accomplish this, the Rights Management relies on different components and subcomponents (Figure 20):

- the IPR Service, which provides an API layer to interact with the MV Blockchain Service Provider and the MV SLC Engine sub-components.
- the MV SLC Engine, which handles all the run-time features related to SLCs such as the creation of SLC instances, checking for the trigger of specific SLC-related events.
- the MV Blockchain Service Provider, which provides blockchain capabilities within an MV Node.
- the Licence Comparator, that offers automatic compatibility checking between CC licences.
- the MV DAM, that stores all the information related to SLCs.

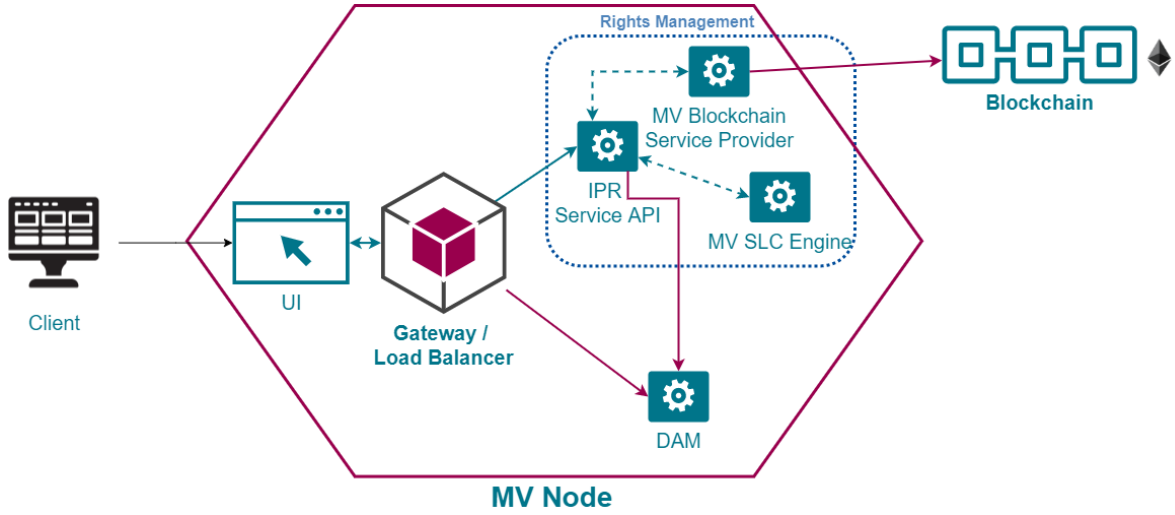

Figure 20: Rights Management Architecture Overview[26]

4.1.1 SLC Creation and Notarisation

As mentioned above, the main focus of the Rights Management component is to create SLCs and "notarise" them on the blockchain. As a brief recap, the following sequence diagram (Figure 21) shows the process of the creation of an Ownership Deed SLC (i.e., a legal contract that claims the user's ownership of a specific asset):

1. The user uploads an asset and requests the Ownership Deed creation.

[25] https://mediaverse-project.eu/wp-content/uploads/2021/10/MediaVerse_D4.1_Copyright-and-Procedures-for-IPR-Definition.pdf

[26] Updated version from MediaVerse D4.1 Copyright and Procedures for IPR Definition

2. The IPR Service obtains the metadata from the MV DAM (i.e., the database).
3. The MV SLC Engine creates and validates the Ownership Deed SLC.
4. The MV Blockchain Service Provider proceeds to “notarise” the Ownership Deed SLC on the blockchain.

The same approach applies to the creation of Creative Commons SLCs or Copyright Licence SLCs, as explained in the following sections.

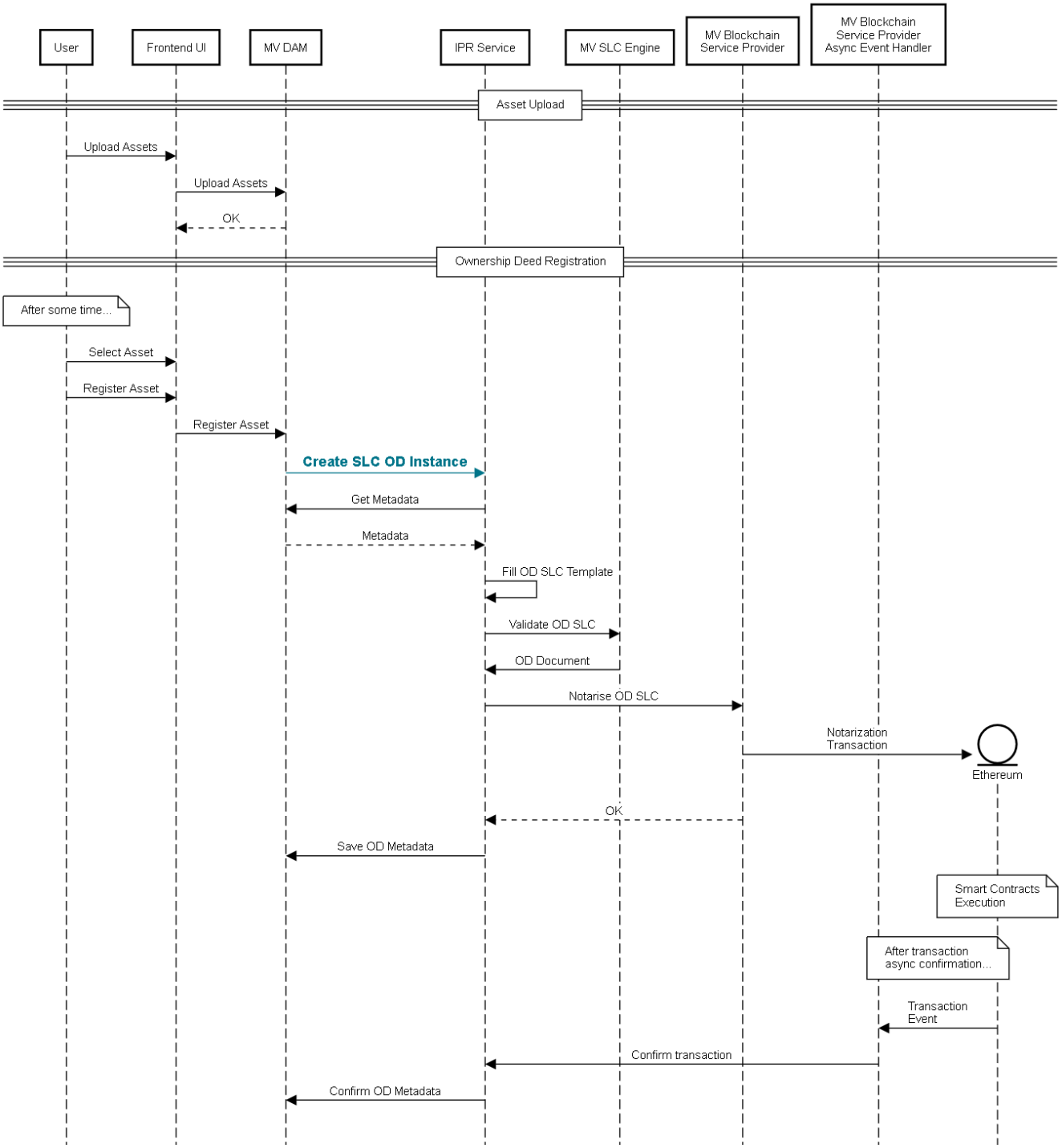

Figure 21: Sequence diagram of Ownership Deed creation and “notarisation” on the blockchain

4.1.2 Derivative Works

As explained in the overview of section 3.3, *Rights Management*, the Rights Management component enables the licensing of a Derivative Work (i.e., a work that embeds other works). From a more technical point of view, Figure 22 shows the sequence diagram of a Derivative Work creation, highlighting the interactions with the Rights Management component:

1. The user can create a project and request to add an asset to it.
2. The system compares the licences of the newly added asset against the licences of other assets (if any).
3. If the process succeeds, the system adds the asset to the project; otherwise, the system returns an error with a brief explanation to the user. If the user wants, he/she can bypass the check under his/her sole responsibility.

(N.B. The aforementioned steps are repeated for each asset the user wants to add to the project.)

4. The user can open the project in the selected Authoring Tool and elaborate on the work.
5. Once done, the user can save the project output as a new asset and possibly proceed to register it.
6. If all the checks succeed, the system proceeds to create the Ownership Deed SLC and notarise it on the blockchain.
7. The user can set a price and request the publication (for further details, please see section “4.1.5 - MV Buyer’s Licence”).
8. In the alternative, the user can proceed to publish the asset under a Creative Commons licence.

Once successfully registered, the user can proceed either setting a price and requesting the publication (for further details, please see section “4.1.5 - MV Buyer’s Licence”) or publishing the asset under a Creative Commons licence. Again, behind the scenes, the Rights Management component takes care of checking that the licences compatibility is always fulfilled.

4.1.3 Moral Rights

The Rights Management component also enables the handling of Moral Rights. In fact, differently from Economic Rights, Moral Rights allow the authors to protect their reputation. Therefore, Moral Rights are always with the author (i.e., the initial creator of the work) and are not transferable.

In general, to fulfil the legal requirements for the Moral Rights, it is sufficient to mention the author’s name in the agreement, but in case of Derivative Works, it could become increasingly difficult to keep track of all of them. In fact, when the initial work “W1” of author “A1” is used in Derivative Work “W2”, both authors “A1” and “A2” shall be mentioned. Therefore, once another author “A3” uses “W2” in Derivative Work “W3”, both authors “A1”, “A2” and “A3” shall be mentioned, and so on.

Following the MediaVerse approach, the Rights Management component uses a special field in the Ownership Deed SLC to keep track of the authors, and automatically add the authors of the parent works to the new authors specified by the user (for further details please see Annex IV: Ownership Deed SLC Template).

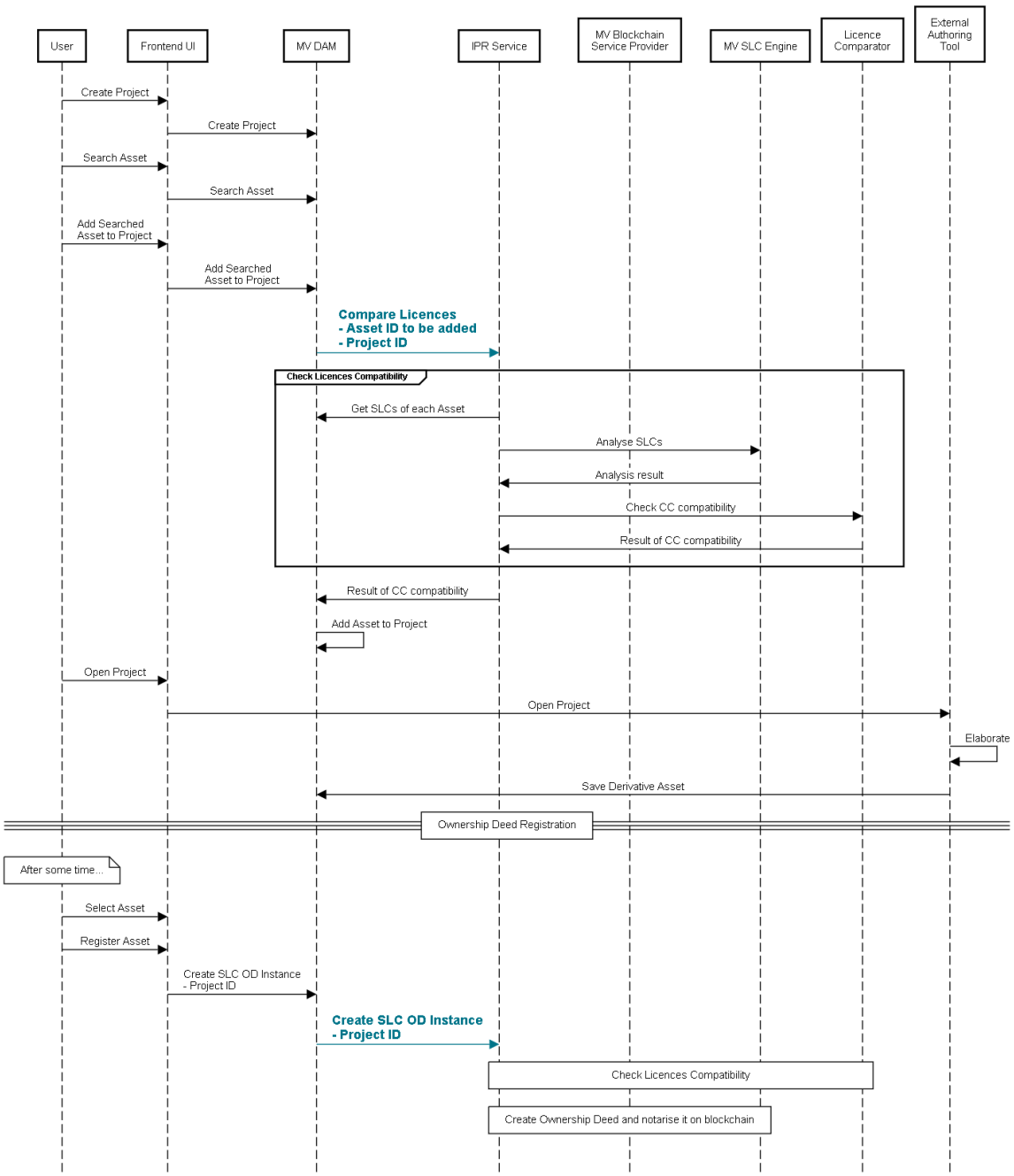

Figure 22: Sequence diagram of Derivative Work creation

4.1.4 Multiple Ownership

In order to support the co-creation paradigm of the MediaVerse platform, the Right Management component enables the multiple ownership feature. In fact, multiple users can claim together the ownership of the rights over an asset, and all of them are involved in the approval process. The following sequence diagrams show the interaction between two users to propose the creation of an Ownership Deed and approve it:

1. In the first sequence diagram (Figure 23):
    a. “User A” uploads the assets and requests the Ownership Deed creation, specifying “User B” as co-owner.

b. The system proceeds to create the SLC without notarising it on the blockchain.
c. The system notifies “User B” about the pending approval of the Ownership Deed.

2. In the second sequence diagram (Figure *24*):
   a. “User B” receives the notification and possibly approves the Ownership Deed.
   b. If all the owners have approved the Ownership Deed, the system creates and “notarises” the Ownership Deed on the blockchain.

The same procedure applies in case of creation of a Creative Commons licence for the co-owned asset.

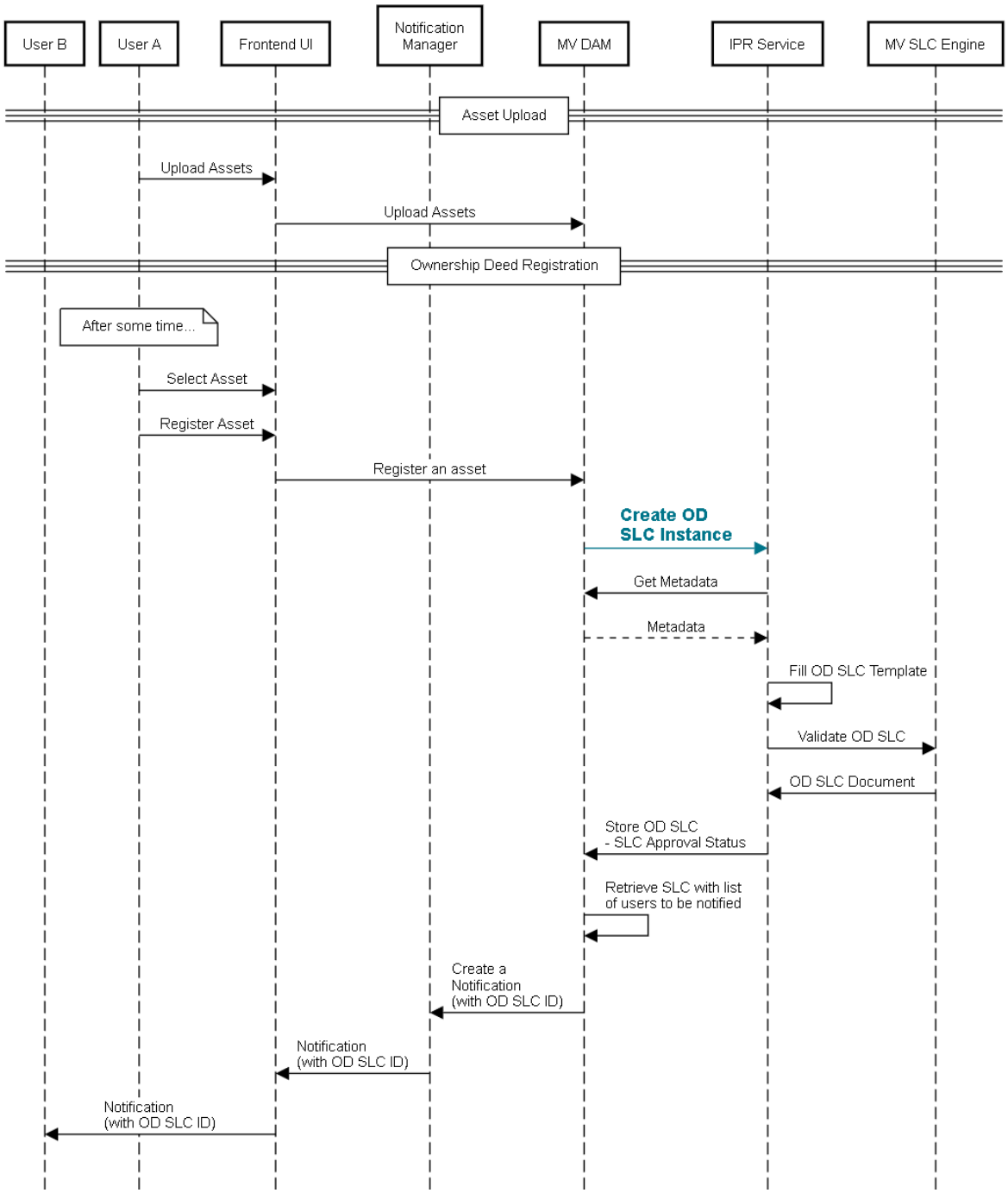

Figure 23: Sequence diagram of Ownership Deed SLC creation for multiple users

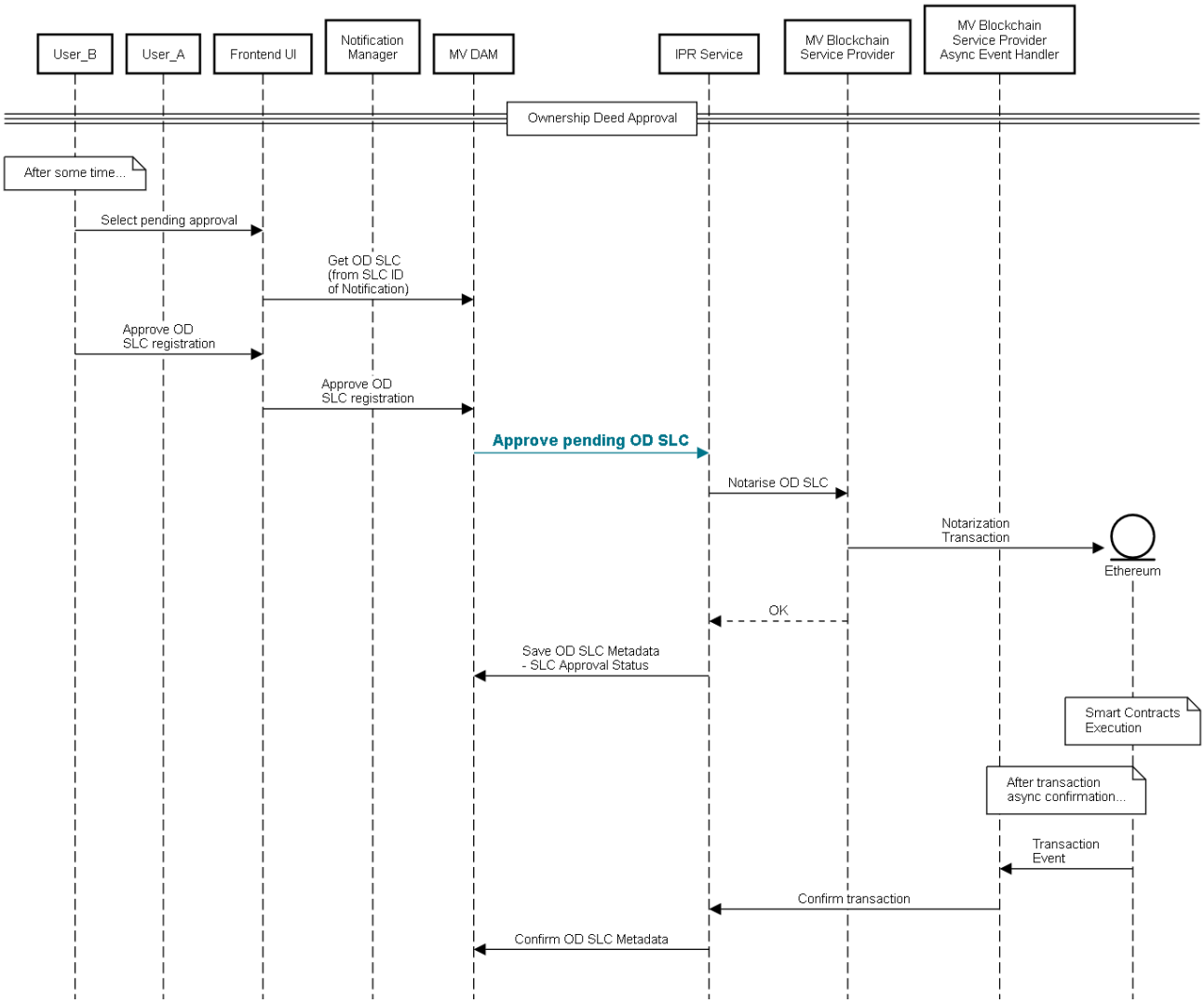

Figure 24: Sequence diagram of Ownership Deed SLC approval and "notarisation" on the blockchain

4.1.5 MV Buyer's Licence

The Rights Management component not only supports the creation of "free" licences like the Creative Commons (see section "3 - Copyright Negotiation Services"), but it also implements a basic support for a "non-free" licence: the MV Buyer's Licence. According to the requests of the Use Case partners in WP2, Table 2 shows the defined requirements for the MV Buyer's Licence.

Thanks to the flexibility of the previously designed Copyright Licence SLC Template, the Rights Management component can obtain the MV Buyer's Licence by setting the proper SLC Data (see Annex II). From a more technical point of view, Figure 25 shows the sequence diagram of the process, highlighting the interactions with the Rights Management component:

1. "User A" (the seller) uploads the owned asset and sets its metadata.
2. "User A" requests the publication on the marketplace.
3. The system creates and "notarises" the Ownership Deed on the blockchain and publishes the asset on the marketplace.
4. "User B" (the buyer) searches the asset on the marketplace and obtains the relative metadata.

5. "User B" purchases the rights of the asset.
6. The system creates and "notarises" the MV Buyer's Licence on the blockchain (similarly to the Ownership Deed).

Table 2: Requirements for the MV Buyer's Licence

TYPE OF ASSET	RIGHTS AND DUTIES OF THE BUYER
Subtitles	• possibility to exploit it • possibility to adapt it (i.e., create Derivative Works) • one-off remunerated licence • PayPal as payment method • possibility to translate it • non-infective licence (i.e., Derivative Works can be licenced differently) • mandatory attribution of the original author • "worldwide" territorial scope • non-sublicensable licence • non-exclusive licence • non-revocable licence • any type of medium • non-transferable licence
Music	
Video	
Image	

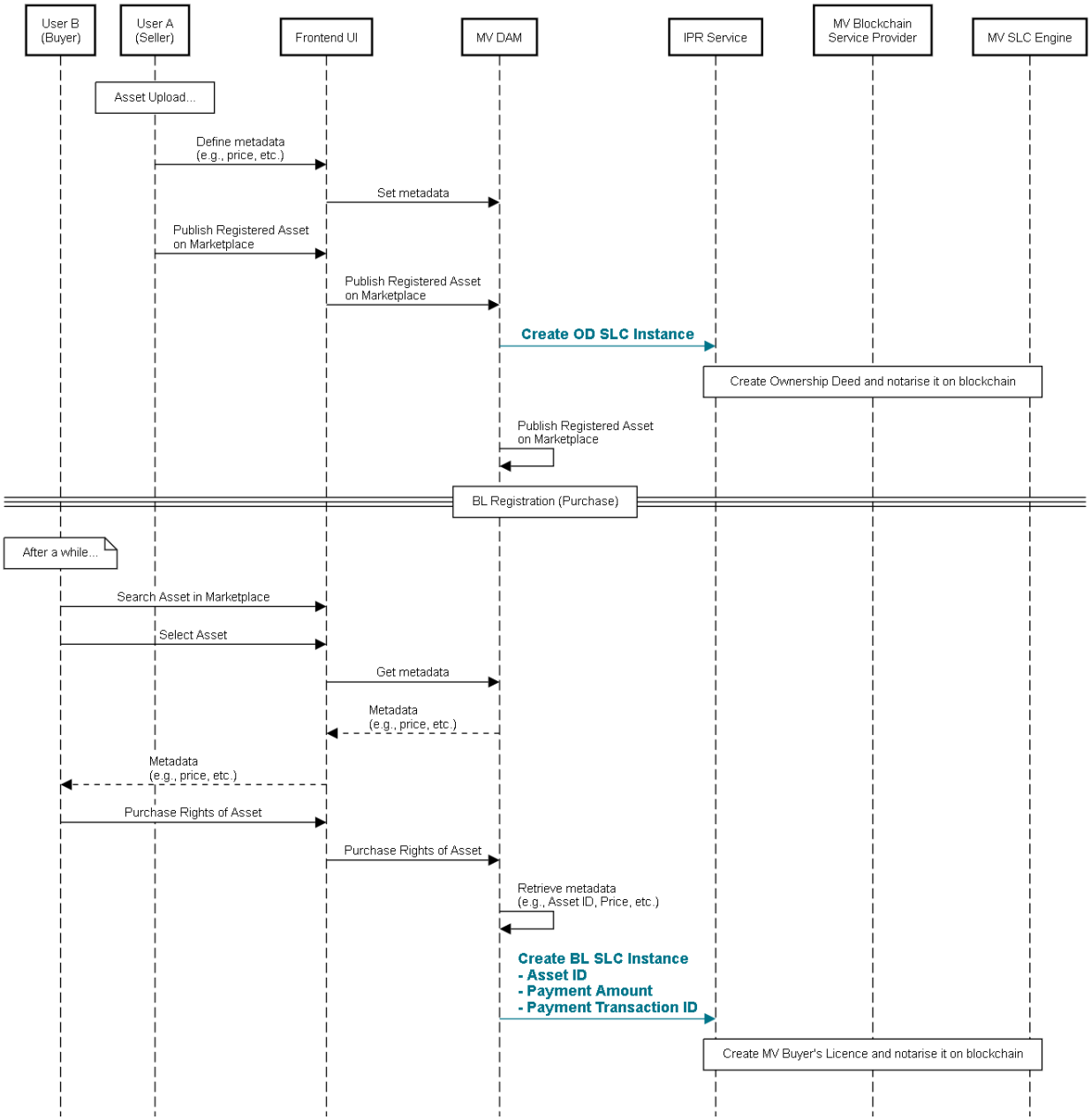

Figure 25: Sequence diagram of MV Buyer's licence creation and "notarisation" on the blockchain

4.2 Blockchain Technical Aspects

This section describes the latest technical updates that have been made to the MediaVerse Smart Contracts and the Blockchain Service Provider. The updates made to Smart Contracts includes the introduction of greater granularity in the Revenues Split and Ownership Shares, support for Gas Station Network, and enabling the MV Node to increase a user's allowance. The updates made to the Blockchain Service Provider adds the support to a new Testnet (called Sepolia[27]), new scripts to create a wallet and prepare deploy, Gas Station Network support, JSON Web Token authentication and the ability of MV Node to increase user's allowance.

[27] https://sepolia.dev/

4.2.1 Ownership Shares and Revenues Split

To allow greater granularity in revenue splits and ownership sharing, the total value was increased from 100 10.000. This allows a more precise split of revenues and ownerships between users. In addition to the changes made to the contracts, additional tests were added to verify the correct behaviour of the code.

4.2.2 Gas Station Network Support

As introduced in *D4.1 - Copyright and Procedures for IPR Definition*[28], the Gas Station Network allows developers to build applications that pay for user transactions, so that they do not need to hold ETH to pay for gas (Figure 26), the fee required to successfully execute a transaction or interact with a contract on the Ethereum blockchain platform.

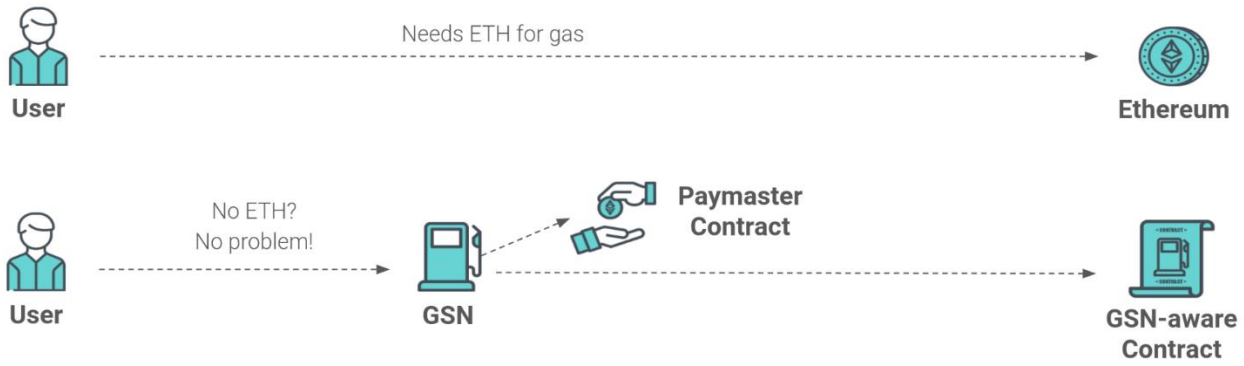

Figure 26: Simplified diagram on how the Gas Station Network works[29]

To support the newer version of OpenGSN[30], some modifications were made to MediaVerse smart contracts:

- We have modified the MV Coin constructor to make it compatible with the ERC2771 Recipient contract[31]
- We added the Whitelist Paymaster[32] contract to the MV Contracts directory.

A GSN Paymaster is the contract that pays for a submitted request. In our case, we opted for a Whitelist Paymaster that only accepts requests from specific known addresses to protect against anonymous attacks. To perform transactions without the constraint of having and managing ETH, a GSN Provider interposes itself between the user and the blockchain node. Once this provider receives the transaction from the user, forwards it to the Relay Server that takes care of making it public on Ethereum by charging the gas to a Paymaster (Figure 27).

A paymaster maintains an ETH balance in the RelayHub and can implement any business logic to decide whether to accept or reject a meta-transaction. In our case, we opted for a Whitelist Paymaster that allows us to define the users authorised to make transactions. This helps us prevent malicious users from carrying out transactions with the aim of emptying the Paymaster's wallet.

[28] https://mediaverse-project.eu/wp-content/uploads/2021/10/MediaVerse_D4.1_Copyright-and-Procedures-for-IPR-Definition.pdf
[29] https://docs.opengsn.org/#ethereum-gas-station-network-gsn
[30] https://opengsn.org/
[31] https://docs.opengsn.org/contracts/#receiving-a-relayed-call
[32] https://github.com/opengsn/gsn/blob/master/packages/paymasters/contracts/WhitelistPaymaster.sol

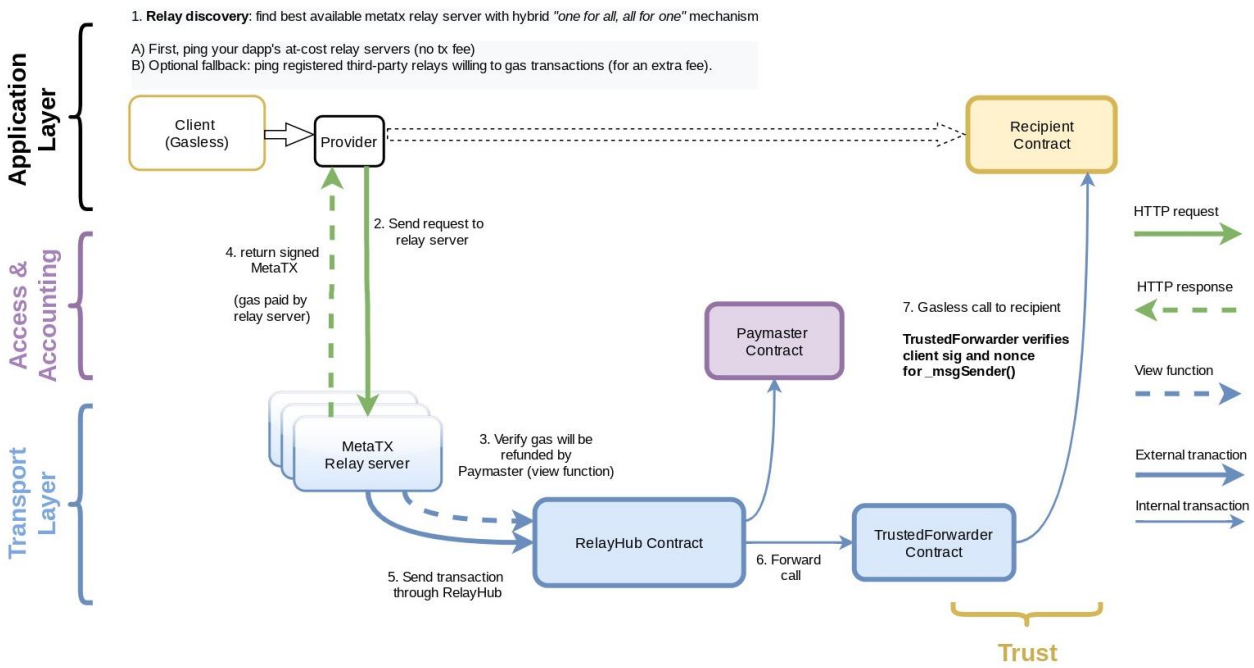

Figure 27: OpenGSN flow

Currently, the supported version of OpenGSN by the Blockchain Service Provider is the 3.0.0-beta.0. The development of OpenGSN is still ongoing and the Relay Servers currently maintained by the community are unstable and still operating on problematic testnets, such as Goerli, making it difficult to test.

To solve this issue and provide a stable and tested system, we decided to fork the MV Coin contract to allow the MV Node to manage a user's allowance. In this way, no operation is required from the user, who may not even hold ETH. By enabling this *GSN-like* behaviour in the MV Node, we can serve a similar purpose while we wait for the developments of the GSN to become more stable. In any case, the MV Blockchain Service Provider remains fully compatible with the GSN, which will only need to be configured accordingly by using either an existing relay server or configuring a proprietary one as explained below:

- **Use an existing relay server:** go to GSN Relay Servers[33] page, select one of the available relays (depending on the testnet where you want to start the BCSP), then put the Forwarder address in the `FORWARDER_ADDRESS` variable located inside the `.env` file.
- **Configure your own GSN relay server:** This choice is a bit trickier but allows for better customization and control. First, you will need to follow the guide provided in the OpenGSN documentation[34] and then you will need to put the address of the relay in the `RELAY_URL` variable and the address of the forwarder in the `FORWARDER_ADDRESS` variable, both are located inside the `.env` file.

4.2.3 Wallet Management

To deploy contracts on blockchain and interact with it, the MediaVerse Node requires a crypto wallet. A blockchain wallet is based on public-key cryptography. Therefore, each wallet has:

[33] https://docs.opengsn.org/networks/addresses.html#deployment-addresses-and-configurations
[34] https://docs.opengsn.org/relay-server/tutorial.html#tutorial-running-a-relay-server-for-fun-and-profit

- A public key: that is used to receive cryptocurrency transactions. It is paired with the wallet private key. Can be shared with anyone.
- A private key: that is a 256 character long binary code (or 64-digit hexadecimal code) and is used to prove the wallet ownership (e.g., made transaction on the blockchain). It must not be shared and is important to store it safely.
- A public address: that is a shortened version of the wallet public key. It is used to receive transactions through the blockchain.

To increase the security of the Blockchain Service Provider, we have added a script to be run by the MediaVerse administrator at deployment time, which generates an Ethereum Wallet (public/private key pairs) and saves it in a JSON file encrypted with a password provided by the MV Node administrator. To generate a new wallet you can run this command:

```
npm run createwallet
```

After the generation, the script returns the address that can be used to transfer ETH (Figure 28).

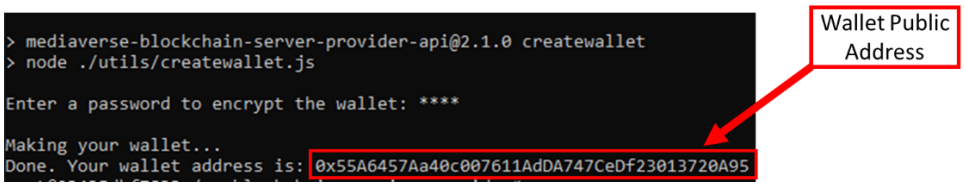

Figure 28: Generating a wallet with the Blockchain Service Provider script

To allow the BCSP to correctly decrypt the newly generated wallet, you must enter the password at the `WALLET_PW` field you can find inside the `.env` file.

4.2.4 Sepolia Testnet Support

A testnet is a blockchain that runs in parallel to the real Ethereum network and is used to test the protocol and the smart contracts. The testing process ensures that the protocol or the smart contracts work as intended before deploying them to the main (non-gas-less) network.

In the previous version of the Blockchain Service Provider, various testnets were supported, including Rinkeby[35], Ropsten[36] and Goerli[37]. However, over time these networks became saturated and were attacked by malicious users with the aim of retrieving as many Test-Eths as possible, making life difficult for developers who want to retrieve Test-Eths for deploying contracts or testing the network.

To facilitate the testing of MediaVerse, we have added support for a new proof-of-stake testnet called Sepolia. In addition, in Sepolia, the total number of tokens is not limited, which means that developers are less likely to face a shortage of Test-Eths. Sepolia is also designed to simulate the real Ethereum Network, which allows developers for faster transaction confirmation and feedbacks. To use the BCSP on the Sepolia testnet, follow these steps:

[35] https://www.rinkeby.io/#stats
[36] https://github.com/ethereum/ropsten
[37] https://goerli.net/

- Create an account on INFURA[38] and generate a new API key (Figure 29)

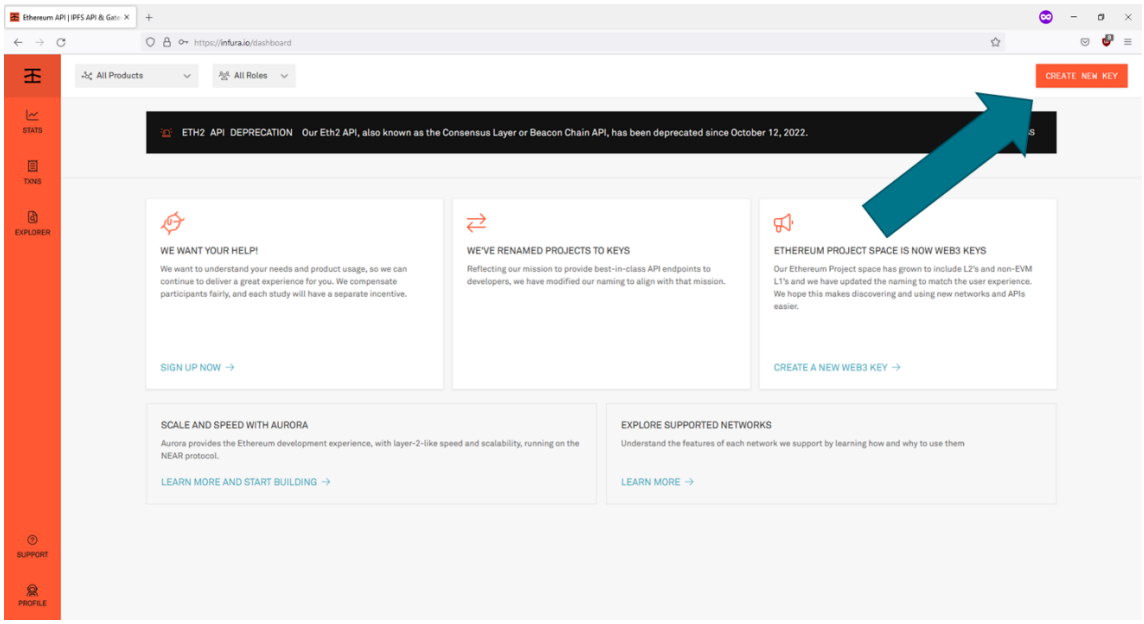

Figure 29: Infura dashboard

- Give a project Name and select **Web3 API** as network (Figure 30)

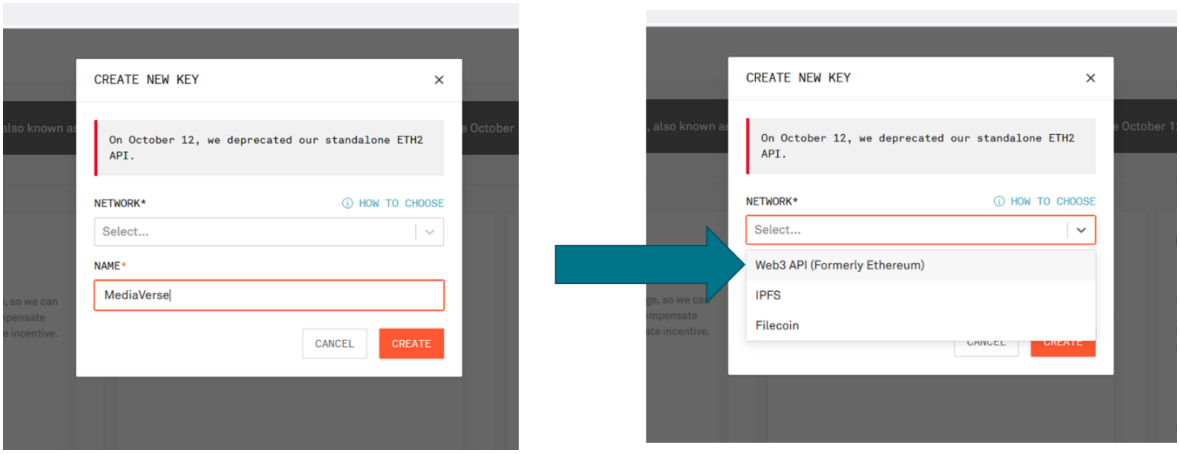

Figure 30: Create a new key on infura dashboard

- Now set your environment variables like the following (Figure 31):

```
services:
  mv-bcsp:
    image: mediaverse/mv-bcsp
    volumes:
      - "$HOME/mv-bcsp-config:/mv-blockchain-service-provider/config"
    environment:
      - NODE_ENV=production
      - DEFAULT_NETWORK=sepolia
      - INFURA_API_KEY="YOUR-INFURA-API-KEY"
      - WALLET_PW="YOUR-WALLET-PASSWORD"
```

Figure 31: A snippet of the docker compose file with BCSP and environment variables

[38] https://infura.io/

Before starting the BCSP, the MV Node wallet must have SepoliaETH to load contracts and execute transactions. There are two methods to obtain SepoliaETH:

1) **Mining:** go to https://sepolia-faucet.pk910.de/, enter MV Node wallet address and click on start mining (Figure 32)

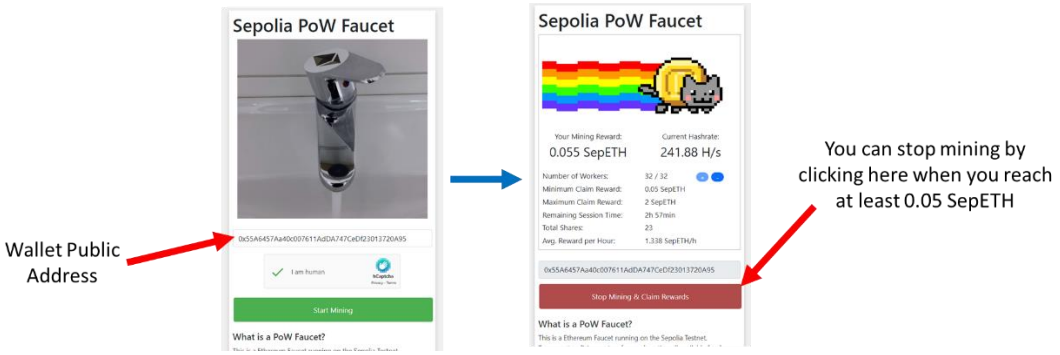

Figure 32: Sepolia mining faucet

2) **With Twitter Post:** create a Tweet with MV Node wallet address, insert the tweet link on https://faucet-sepolia.rockx.com/ and click on "Send Me ETH" to get 0.1 SepoliaETH (Figure 33). You can repeat this process every 45mins.

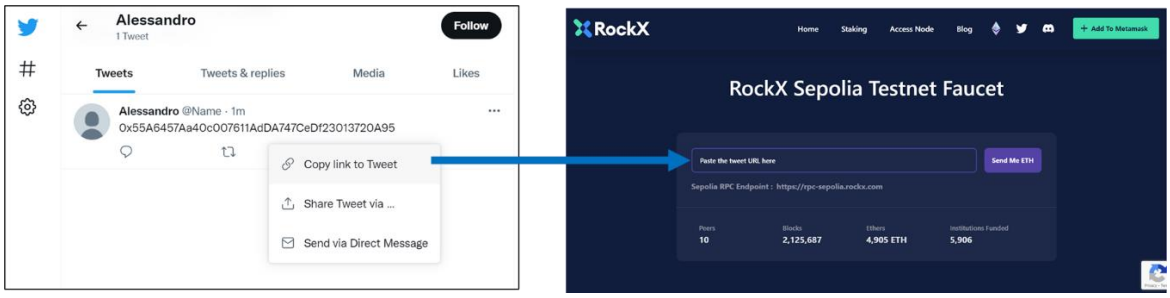

Figure 33: Steps for requesting SepoliaETH via RockX faucet

To check the MV Node Wallet balance, simply go to https://sepolia.etherscan.io/ and search for your wallet address (Figure 34 shows an example).

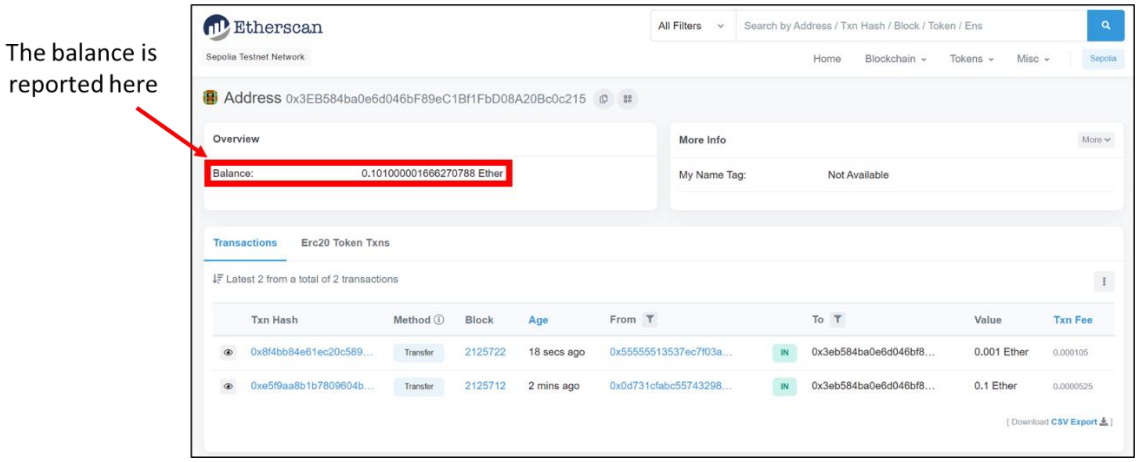

Figure 34: Viewing the details of an example wallet on Ehterscan

Now you can run the MediaVerse Blockchain Service Provider. On the first load, you can see the deployed contract addresses printed on logs (Figure 35).

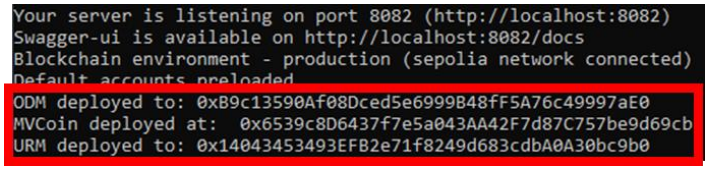

Figure 35: Addresses of contracts deployed by Blockchain Service Provider at first start-up

4.2.5 JSON Web Token Authentication

To strengthen security, we decided to implement an additional layer of security in the MV Blockchain Service Provider (which can currently only be contacted by the docker compose MV Network), to ensure that communication may only originate from authorised components (i.e., IPR Service). This is achieved by means of the JSON Web Token[39] (JWT). This is an open standard (RFC 7519) that defines a compact way to transmit information between parties as a JSON Object in a secure way. This information can be verified and trusted because it is digitally signed. In our case, the JWT is signed with HS256 algorithm with a 256-bits secret.

The structure of a JWT consists of three parts separated by dots:

- Header: contains the type of token (in our case is JWT), and the algorithm used for signing.
- Payload: contains the information to be exchanged by the parties. This in turn is divided, into three stages: registered parameters, public parameters, and private parameters. Our payload has:
    - "iat" (or issued at public parameter) that identifies the time at which the JWT was issued.
    - "username": "iprservice" that is a private parameter which identifies the IPR Service
- Signature: the sign of both Payload and Header. These are first joined and then subjected to the signature operation specified in the JWT header. At the end of the operation, the result is combined with the Header and Payload constituting the token .

A JWT is generated by the BCSP using a random secret. The IPR service has to include its JWT in every request made to the BCSP that can verify its authenticity by re-calculating the signature over the Header and Payload of the provided JWT and verify the match between the calculated signature and the one inside the JWT itself.

To generate a new token, the MediaVerse node administrator can manually configure the related environmental variables or run the following command:

```
npm run generatetoken
```

This script will generate a random secret and then will sign a jwt token that can be used to authenticate API calls (Figure 36).

Then, the MediavVerse node administrator must insert the previously generated secret into the `TOKEN_SECRET` field located inside the `.env` file of the MV Blockchain Service Provider and in the `mv-services-bearer-tokens.mv-blockchain-service-provider` field of the `application.yml` of the IPR Service component to successfully authenticate the requests.

[39] https://jwt.io/introduction

```
> mediaverse-blockchain-server-provider-api@3.0.1 generatetoken
> node ./utils/generatetoken.js

Randomly generated TOKEN_SECRET="9a5165af6a4156826ecd3b9b1da8138a1b72b544dedd0253f6c9d437caced9a
28c9f6002820b56f3ae0ec3ec6a76c41a0a3f14077df5d4249c5510b3c8e55411ad7d1ca668761e246ad37ff164c8c50
3694bf7677ba84f32d588f8a993ce2b2e70e4db4327b6571ea39346adabb0da52c80f049d6cd11d23871ceee58798084
b8791dcd474d5e421109491ea8fcef569cd28c6f566966ae56e17fab8aad77d0fb526d98d6648da07467740330418822
84ddee0cc6e4d9a69a2014cea3660d28f4a5ef483bc46e246af5406c34cc5d81c70e0548c08a6287254a9b4cf263ee60
75b834d84809ed52bacb63e63d10608c398e691da4e176d7a2253bbeb18248a20"
Place it in the .env file!

New IPR Service Authentication Token: eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJ1c2VybmFtZSI6Imlwc
nNlcnZpY2UiLCJpYXQiOjE2Nzg0NjU5Mjl9.iUDoVHuG4SqQjQ4zRKU6vX3RaTL_t6N_wzVmeUG6r2w
```

Figure 36: Generating a new JSON Web Token using the Blockchain Service Provider script

4.3 Further Blockchain Considerations

4.3.1 Proof-of-Work to Proof-of-Stake

As introduced in the previous deliverables, a blockchain is a database of transactions that is updated and shared among many computers in a network. The term blockchain derives from the fact that it stores a set of transactions called blocks. Most blockchains are public and, due to their properties, it is only possible to add data, not remove it. If someone want to alter some of the information or cheat the system, they will have to do that on a large set of computers (the majority) on the network.

Ethereum is a blockchain-based platform that makes it possible to create applications (through smart contracts), send money, hold assets, and communicate without the control of a central authority.

Smart contracts, or dapps, are applications that reside on the blockchain and can be used by users (or other contract) through transactions.

The two most popular consensus algorithms for creating new blocks on the blockchain are proof-of-work and proof-of-stake. The main difference between these two methods is that the first is based on the so-called crypto mining, while the second is based on crypto staking.

Proof-of-work is the mechanism that allows the network to reach consensus, which means the agreement of all nodes, on aspects such as transactions, wallet balances, and so on. This prevents users from spending their coins twice (double spending) and makes the blockchain very difficult to attack and manipulate. This algorithm regulates the work of 'miners', the ones in charge of validating blocks to add them to the chain. However, this mechanism has a very strong criticism related to high energy consumption[40].

Initially, the Ethereum network was using proof-of-work, but in 2022, it switched to proof-of-stake. This mechanism is much less energy-intensive than proof-of-work and is, at the same time, more secure. The consensus mechanism behind Proof-of-Stake involves validators, i.e., users 'staking' capital in the form of ETH on a smart contract. Those ETH act as a guarantee that can be destroyed if the validator behaves dishonestly or lazily. The validator is then responsible for verifying that new blocks propagated on the network are valid and, occasionally, for creating and propagating new blocks.

Another substantial difference between PoW and PoS concerns the timing of blocks. In fact, in PoS the time to insert a block is fixed, whereas in PoW it is determined by the difficulty of mining. In Ethereum, the PoS is divided

[40] https://ethereum.org/en/developers/docs/consensus-mechanisms/pow/

into slots (of 12 seconds) and epochs (of 32 slots). In each slot, a validator is randomly selected as the block proposer. This validator is responsible for creating a new block and sending it to other nodes on the network[41].

By making the switch to the PoS consensus Ethereum ensured a more sustainable and eco-friendlier network eliminating 99.99% of its carbon footprint[42].

4.3.2 Layer-2 Blockchains

The three desirable properties of a blockchain are that it is decentralised, secure and scalable. However, the blockchain trilemma[43] states that a simple blockchain architecture can only achieve two of these three properties. Currently, a blockchain is secure and decentralised, but it is not scalable. Ethereum, in fact, is not capable of processing thousands of transactions per second, preventing long-term scalability.

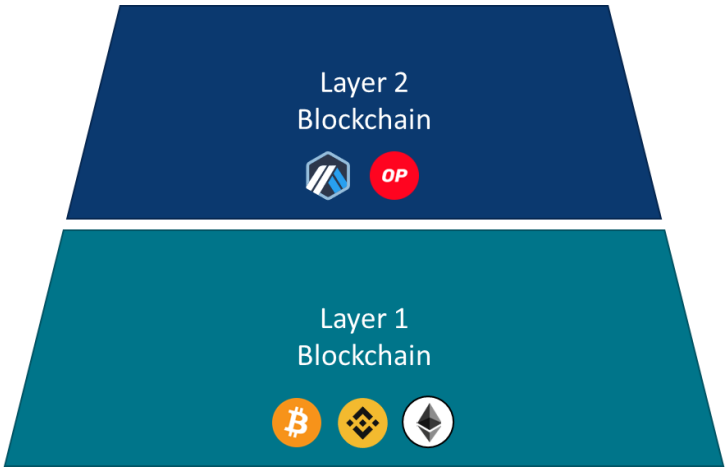

Figure 37: Graphical representation of a blockchain layer structure with some real blockchains

To solve these problems, the level 2 blockchain (Figure 37) comes to help. This is nothing more than a separate blockchain (off-chain) that extends the current traditional blockchain (called level 1). One of the main advantages of using off-chain solutions is that the main chain does not have to change structurally because the second level is added as an additional layer. Consequently, layer 2 solutions have the potential to achieve high throughput without sacrificing network security. This allows layer 1 to handle security, data availability and decentralisation, while layer 2 handles scaling. Layer 2 removes transactional load from layer 1 and then send the finalized proofs to layer 1. By removing this transaction load from layer 1, the basic layer becomes less congested, and everything becomes more scalable[44].

Furthermore, the cost per transaction of level 2 is significantly lower than the current cost per transaction of level 1 (up to 100 times less[45]). To enable MediaVerse to reduce transaction costs even more, in addition to the possible use of a Blockchain Layer 2, future developments planned on Ethereum will be useful like sharding, which will be launched in the course of 2023.

[41] https://ethereum.org/en/developers/docs/consensus-mechanisms/pos/
[42] https://consensys.net/blog/press-release/ethereum-blockchain-eliminates-99-99-of-its-carbon-footprint-overnight-after-a-successful-merge-according-to-new-report/
[43] https://www.ledger.com/academy/what-is-the-blockchain-trilemma
[44] https://ethereum.org/en/layer-2/
[45] https://l2fees.info/

Sharding is the process where a database is split horizontally to distribute the load. In Ethereum, it will operate synergistically with layer 2 rollups, sharing the workload of managing the large amount of data needed for rollups across the entire network[46]. This will continue to reduce network congestion and increase transactions per second, due to the ability for more validator to participate in the network with less restrictive requirements (e.g., the hardware required to start a node). Consequently, by increasing the number of transactions per second, the costs of each individual transaction will tend to decrease further.

4.4 Future Developments for Further Studies

As shown in section 4.1, *Right Management Features*, the Rights Management component is fully functional for the MediaVerse scopes. Nevertheless, there is room to further ease the SLC "notarisation" process for the user and envisage more complex scenarios than the ones foreseen within the MediaVerse project.

This is possible on one side thanks to the MV Triplets approach (i.e., triplets of subject-right-asset). This approach allows to create a "tree" of dependencies and notarise them on the blockchain. On the other side, thanks to the Accord Project SLC Templates system, it can obtain different type of legal agreements by configuring the proper parameters (for further details, please refer to *D4.1 - Copyright and Procedures for IPR Definition*[47]).

In the following sections, we suggest some feasible implementations.

4.4.1 SLC Creation and Notarisation

As shown in section 4.1.1, *SLC Creation and Notarisation*, the Rights Management allows the "notarisation" of a right on the blockchain only afterward a successful "notarisation" of an Ownership Deed. Currently, this is addressed in the MediaVerse frontend with different buttons that are enabled or greyed-out according to the "notarisation" status, which guides the user in the process. However, due to the intrinsic nature of the blockchain, the confirmation of a successful "notarisation" might require some time. In fact, the user has to wait between the first request to "notarise" the Ownership Deed and the following one to "notarise" a right, and this procedure is even longer when multiple owners are involved (for further details please refer to section 4.1.4, Multiple Ownership).

To improve this, a queuing mechanism could be implemented in the future, allowing a single user to do multiple requests at once, leaving up to the system when to perform them, without having to wait between one and the subsequent. The same approach can also be applied to the multiple owners' case, allowing parallelising the requests even if all the parties have not approved the right yet.

4.4.2 Publishing

The Rights Management component could be possibly extended to support the presence of an intermediator (e.g., a publisher) between the owner and the buyer. The following sequence diagrams show a possible future implementation, where the owner ("User A") and a publisher ("User B") propose the creation of a Publisher Licence and approve it:

1. In the first sequence diagram (Figure 38):
    a. "User A" uploads the assets and requests the Ownership Deed creation.

[46] https://ethereum.org/en/developers/docs/scaling/#sharding

[47] https://mediaverse-project.eu/wp-content/uploads/2021/10/MediaVerse_D4.1_Copyright-and-Procedures-for-IPR-Definition.pdf

b. "User A" requests the Publisher Licence creation, specifying "User B" as publisher and the revenue splits.
c. The system proceeds to create the SLC without notarising it on the blockchain.
d. The system notifies "User B" about the pending approval of the Publisher Licence.

2. In the second sequence diagram (Figure 39):
   a. "User B" receives the notification and possibly approves the Publisher Licence.
   b. If the Publisher Licence is approved, the system creates and "notarises" the Publisher Licence on the blockchain.
   c. Once the Publisher Licence has been successfully registered, "User B" can set a price and publish the asset on the marketplace.

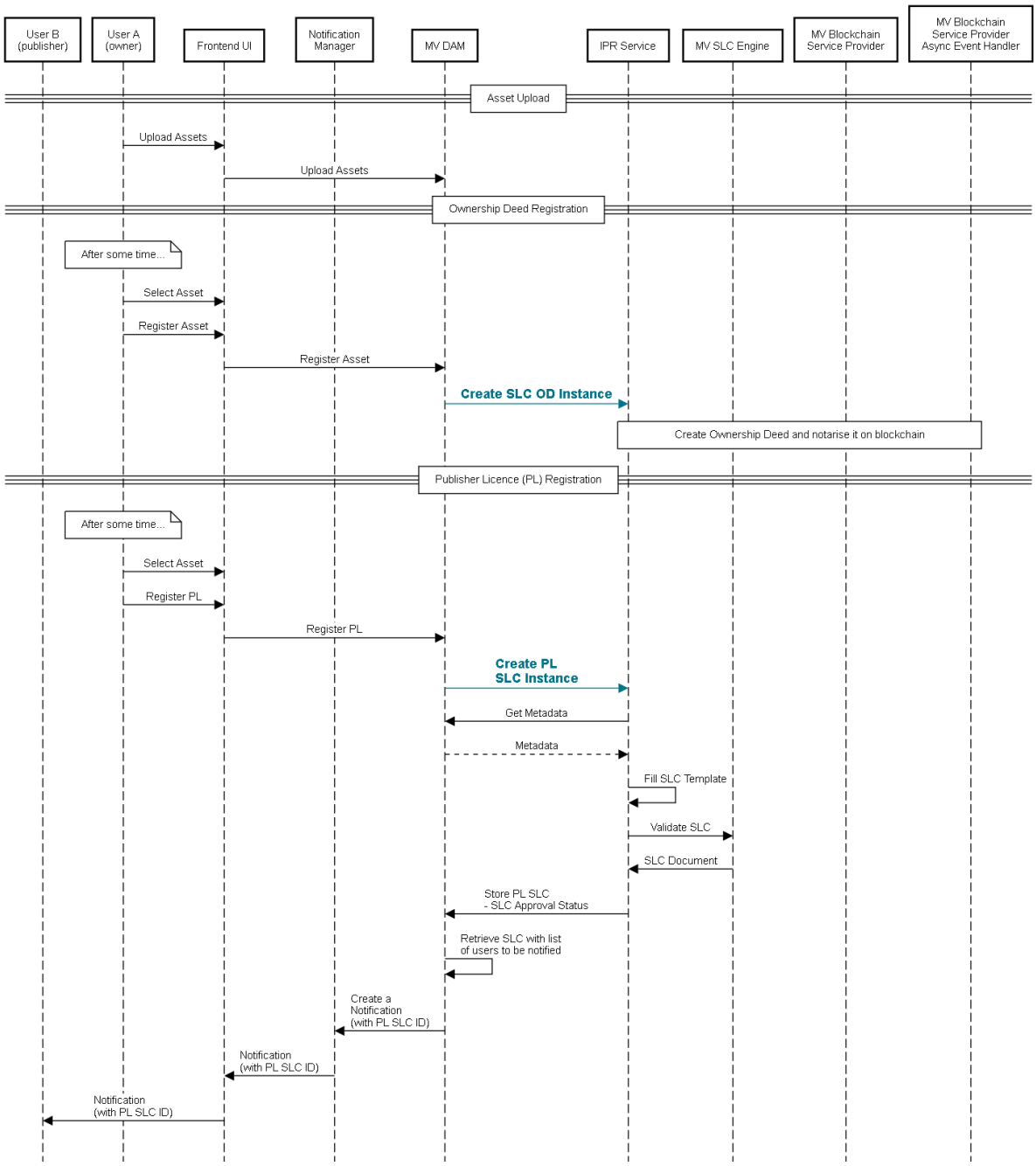

Figure 38: Sequence diagram of the Publisher Licence SLC registration

Once the asset is published on the marketplace, other users will be able to purchase the rights according to the same approach shown in section 4.1.5, MV Buyer's Licence, and the MVCoins will be redistributed according to the agreed revenue splits.

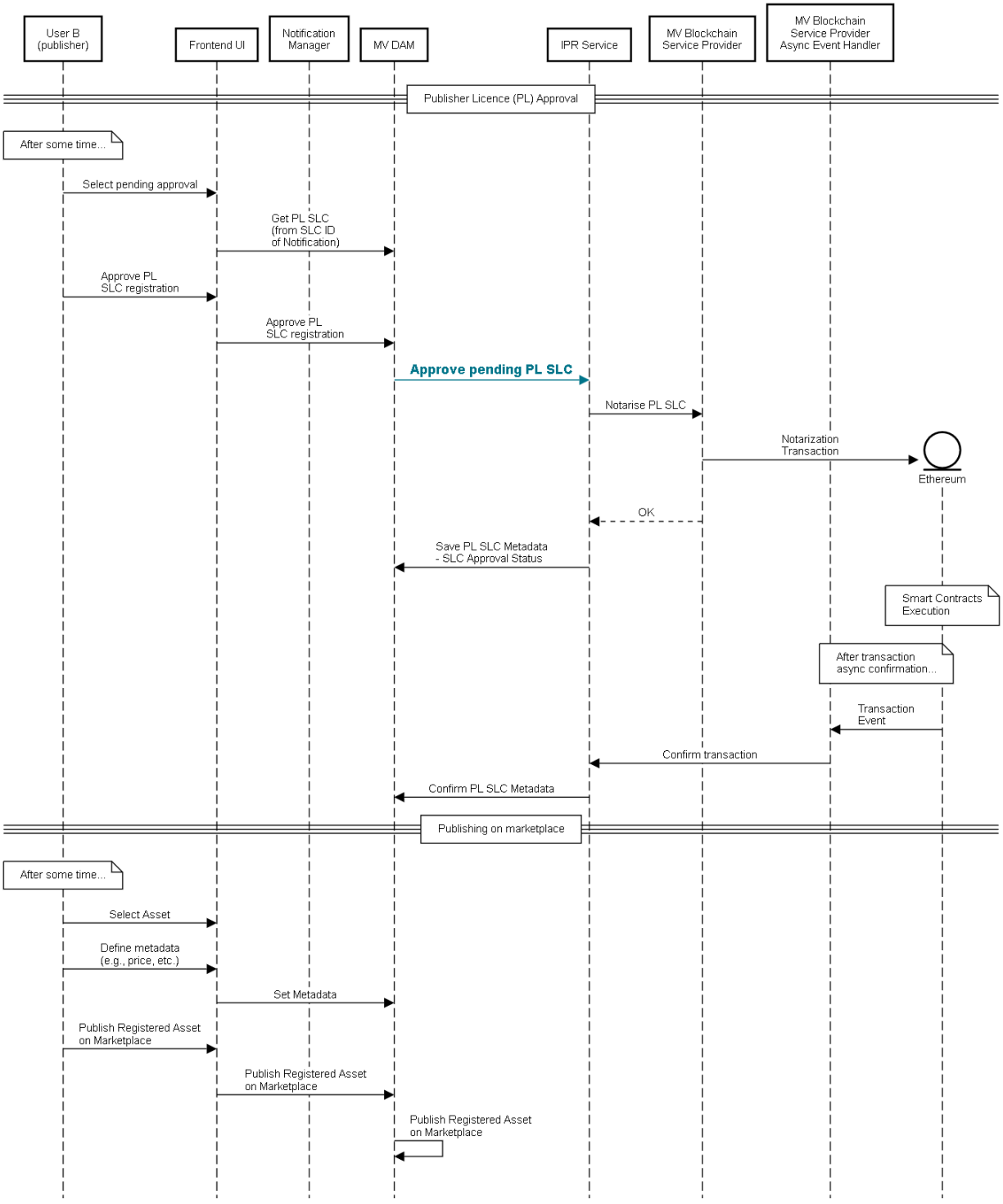

Figure 39: Sequence diagram of the Publisher Licence SLC approval and publishing on marketplace

4.4.3 Ownership Transfer

The Rights Management component is structured to be potentially extended to support the transfer of ownership by the means of creating a new Ownership Deed connected to an existing one. The following sequence diagram show a possible future implementation, where the old owner ("User A") and the new owner ("User B") propose the creation of a new Ownership Deed and approve it:

1. In the first sequence diagram (Figure 40):
    a. "User A" uploads the assets and requests the Ownership Deed creation.
    b. "User A" requests the transfer of the ownership, specifying "User B" as the new owner.
    c. The system creates a new Ownership Deed SLC without notarising it on the blockchain.
    d. The system notifies "User B" about the pending approval of the new Ownership Deed.

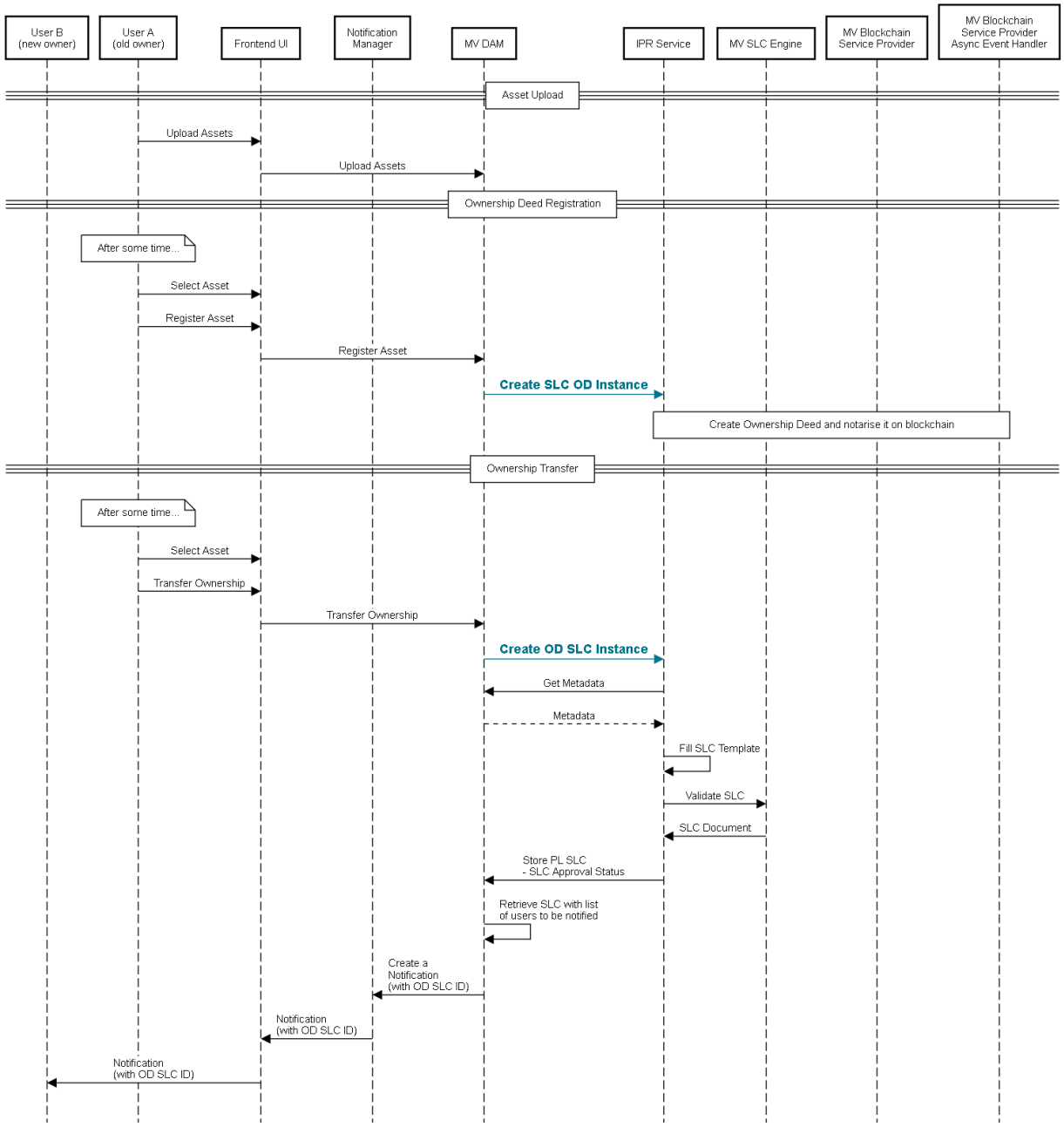

Figure 40: Sequence diagram of the ownership transfer

2. In the second sequence diagram (Figure *41*):
    a. "User B" receives the notification and possibly approves the new Ownership Deed.
    b. If the new Ownership Deed is approved, the system creates and "notarises" the new Ownership Deed on the blockchain.
    c. Once the new Ownership Deed has been successfully registered, "User B" can set a price and publish the asset on the marketplace.

Once the new Ownership Deed has been successfully registered, "User B" is allowed to create new licences of the right of the asset and possibly publish the asset on the marketplace.

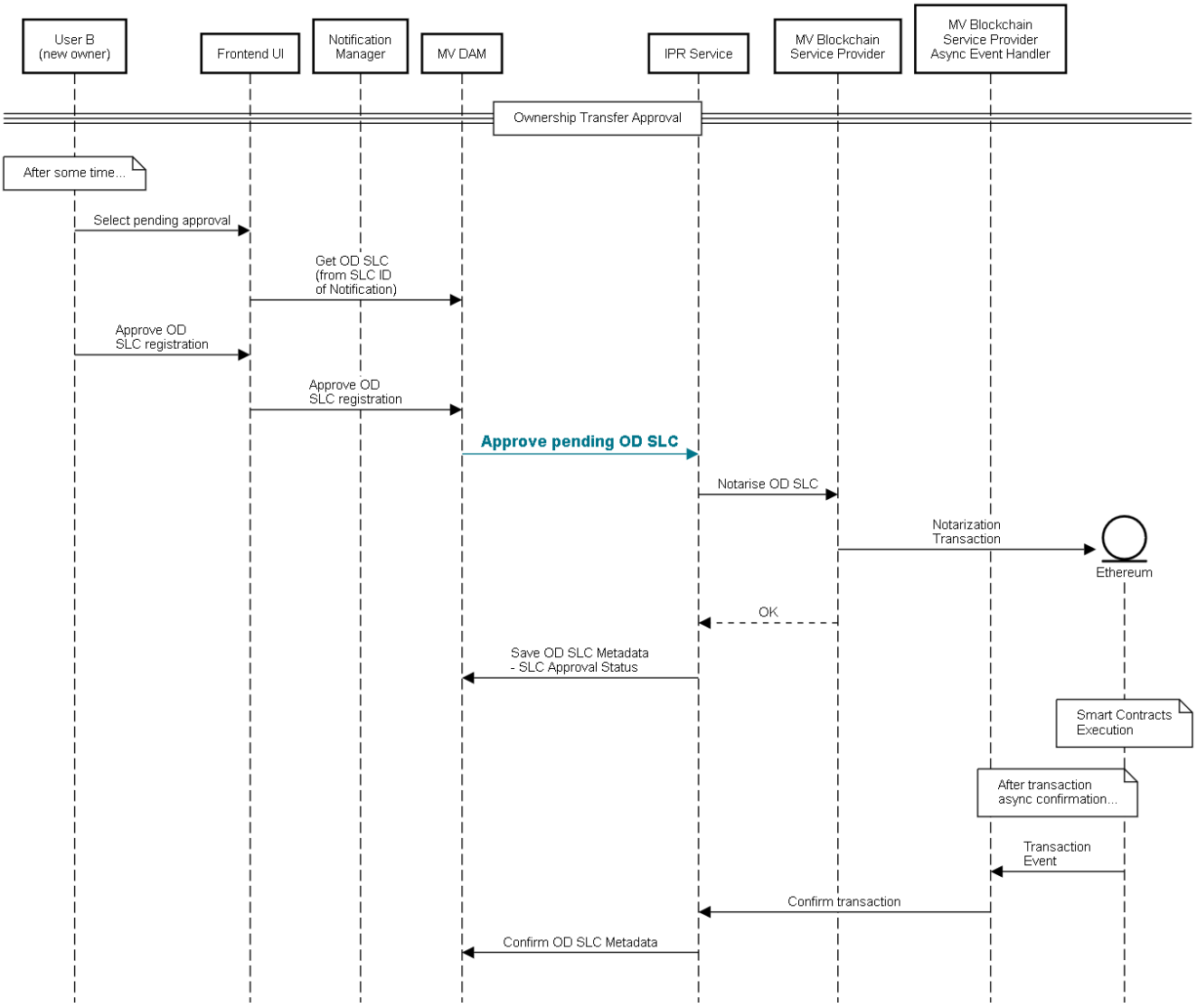

Figure 41: Sequence diagram of the approval of the ownership transfer

5 Conclusions

By M27 (December 2022) MediaVerse had created a federated search system not only to search but also to acquire digital assets. Furthermore, MediaVerse created a system for registering these assets through Creative Commons/Copyright terms and guiding users to choose the best licence based on their preferences while being abstracted from the complexity of negotiations and compatibilities between them.

Blockchain-based smart contracts allow for automated, self-executing contracts that do not require intermediaries inside and outside MediaVerse, which reduce transaction costs and increase efficiency. Overall, MediaVerse and Blockchain exploit the decentralized paradigm in the interests of independence, safety, and content diversity.

Annex I: MV Copyright Licence

MediaVerse uses the Accord Project[48] SLC implementation that defines a SLC as composed by the SLC Template and the relative SLC Data. In turn, the SLC Template is composed of three different parts:

- Text: the natural language of the template.
- Model: the data model that backs the template, acting as a bridge between the text and the logic.
- Logic: the executable business logic for the template.

In the following section is shown the updated version (v3.0.0) of the text part of the MV Copyright Licence SLC Template, and the relative SLC Data to approximate the "Creative Commons Licences" (that we call "CC Like") and the "MV Buyer's Licence".

MV Copyright Licence SLC Template v3.0.0

```
# **Copyright License**

This Copyright License (the "Agreement") is concluded between:

1. Licensor(s) as identified in ownership deed {{ownershipDeedID}} ({{ownershipDeedLink}})

And
2.    {{licenseeFirstname}}{{#optional    licenseeLastname}}    {{this}}{{/optional}}    (the
"Licensee"){{#optional licenseeAddress}}, with registered address at {{this}}{{/optional}}

## 1. **Definitions**

1.1. **Derivative Works**: works which are derived from or based upon the Work, such as
translations, alterations, combinations, adaptations, arrangements, transformations or other
modifications to and/or of the Work, which require permission of the right holder, and which
in itself are also protected under copyright.

1.2. {{#if commercial}}**Commercial Purposes**: intended for or directed towards commercial
advantage or monetary compensation.{{else}} **Non-Commercial Purposes**: not intended for or
directed towards commercial advantage or monetary compensation.{{/if}}

1.3. **Exceptions and Limitations**: the exceptions and limitations to the copyrights on the
Work, as recognised by national law.

1.4. **MediaVerse**: the MediaVerse media platform.

1.5. **Rights**: the copyrights on the Work licensed under this Agreement, excluding any moral
rights on the Work.

1.6.  **Work**: the work on which the Licensor has the lawful copyrights, whether as an author
or after transfer of such rights, in accordance with ownership deed {{ownershipDeedID}}, or
of which the Licensor can sub-license the Rights in accordance with this Agreement.

1.7. **License**: the license as stipulated in article 2 of this Agreement.

## 2. **Scope**

2.1. Subject to this Agreement, the Licensor grants the Licensee a {{territorialScope}}, {{#if
remunerated}}remunerated{{else}}royalty-free{{/if}},                                    {{#if
sublicensable}}sublicensable{{else}}non-sublicensable{{/if}},                          {{#if
exclusive}}exclusive{{else}}non-exclusive{{/if}},   {{#if   revocable}}revocable{{else}}non-
revocable{{/if}} license on the Rights in the Work to:
```

[48] https://accordproject.org/

2.1.1. Reproduce all or part of the Work {{#if anyForm}}in any form{{else}}in the form the Work was provided{{/if}}{{#optional reproductionNumberOfCopies}}, limited to {{this}} copies{{/optional}};

2.1.2. {{#if distribute}}Distribute any physical copies made of the Work in accordance with article 2.1.1.{{else}}-{{/if}} {{#optional typeOfFormat}}, in {{this}},{{/optional}} {{#optional physicalNumberOfCopies}} limited to {{this}} copies{{/optional}};

2.1.3. Make the Work available to the public via {{typeOfMedium}};

2.1.4. {{#if derivativeWorks}}Make and reproduce Derivative Works.{{else}}-{{/if}}{{#if makeAvailableDerivativeWorks}} The License also allows to make them available;{{/if}}{{#if makeNotAvailableDerivativeWorks}} The License however does not allow to make them available;{{/if}}

2.1.5. {{#if translations}}Make translations of the Work.{{else}}-{{/if}}{{#if makeAvailableTranslations}} This License also allows to make them available;{{/if}}{{#if makeNotAvailableTranslations}} This License however does not allow to make them available;{{/if}}

The above is allowed for{{#if commercial}} Commercial and Non-Commercial{{else}} only Non-Commercial{{/if}} Purposes.

2.2. This License shall not apply when Exceptions and Limitations apply to a certain use.

3. **Term**

3.1. The License shall be granted {{#optional numberOfYears}}for {{this}} years{{else}}for the period for which the Work is protected under copyright law{{/optional}}. If any limit applies to the number of copies that can be made and/or distributed, this License shall be granted up until the exhaustion of the amount of reproductions which could be made and distributed.

3.2. {{#if remunerated}}The Agreement shall be terminated when the Licensee has been in default of payment for the License fees in accordance with article 4 of this Agreement, for three months.{{else}}-{{/if}}

3.3. The Licensor shall have the right to terminate the Agreement when the Licensee materially breaches this Agreement, which will include any use with regard to the Work which is not licensed to the Licensee under article 2 of this Agreement or the breach of any obligations relating to Attribution or Assignment contained in articles 5 and 7 of this Agreement.

3.4. In case of termination in accordance with this article 3, all rights licensed will terminate effective immediately and the Licensee will immediately cease using the Work.

4. {{#if remunerated}}**Remuneration**{{else}}**Waiver of right to collect royalties**{{/if}}

4.1. {{#if remunerated}}The License fees are expressed in Euro and are VAT exclusive.{{else}}-{{/if}}

4.2. {{#if remunerated}}Licensee will pay{{else}}-{{/if}}{{#optional paymentAmount}} {{doubleValue}} {{currencyCode}}{{/optional}}{{#optional paymentPercentage}} a percentage fee of {{this}}%, calculated on the basis of the revenue generated by Licensee under this Agreement{{/optional}}{{#if monthlyFrequency}}, on a monthly basis.{{/if}}{{#if yearlyFrequency}}, on a yearly basis.{{/if}}{{#if oneoffFrequency}}, on a one-off basis.{{/if}}

4.3. Parties will each be responsible for their own taxes, other duties, tax penalties and interest, or amounts in lieu thereof however designated, now or hereafter, payable as a result of this Agreement.

4.4. {{#optional paymentMethod}}Payment will be effected through {{this}}.{{else}}-{{/optional}}

4.5. {{#if remunerated}}-{{else}}To the extent possible and to the extent the Licensee uses the Work in accordance with the permitted use stipulated under article 2 of this Agreement, the Licensor will waive its rights to collect royalties for the Licensee’s use of the Work in accordance with this Agreement.{{/if}}

5. **Assignment {{#if sublicensable}}and sub-licenses{{/if}}**

5.1. The Licensee shall {{#if transferable}}be allowed{{else}}not be allowed{{/if}} to transfer its rights under this Agreement.

5.2. {{#if remunerated}}-{{else}}In case the licensed Work is made available to recipients, the Licensee cannot restrict access to and use of the Work by the recipients, as they will have the possibility of concluding an identical license with the Licensor as stipulated in this Agreement.{{/if}}

5.3. {{#if sublicensable}}the Licensee shall in its turn license the Work under the same conditions as stipulated in this Agreement.{{else}}-{{/if}}

6. **Warranty and indemnification**

6.1. {{#if remunerated}}The Licensor warrants that it has all necessary rights and permissions to license the Rights and Works, and that the transferred Rights do not infringe the rights of any third party.{{else}}-{{/if}}

6.2. {{#if remunerated}}The Licensor shall indemnify the Licensee against any claims brought by third parties in relation to the copyrights on the Work.{{else}}-{{/if}}

6.3. {{#if remunerated}}-{{else}}The Licensor provides the Work to the Licensee as-is and, to the extent permitted by law, makes no warranties of any kind as to the title, fitness for a particular purpose, absence of defects or merchantability of the Work.{{/if}}

6.4. To the extent permitted by law {{#if remunerated}}and notwithstanding article 6.2 of this Agreement{{/if}}, The Licensor shall not be liable for any direct or indirect damages in relation to this Agreement, the License included therein, and the Work.

7. **Attribution**

7.1. {{#if attribution}}The Licensee is obliged to mention the identity of the author(s) when exploiting the Work in accordance with article 2 of this Agreement. The Licensee shall mention the identity of the author(s) in a prominent way on or in connection to the Work or copies thereof, taking into account the exploitation methods, context and the format. Where applicable, the Licensee shall comply with the reasonable method of attribution requested by the Licensor (including pseudonymisation where appropriate).{{else}}The Licensee is not obliged to mention the identity of the author(s) when exploiting the Work in accordance with article 2 of this Agreement.{{/if}}

7.2. {{#if attribution}}The Licensee shall not remove any identification or copyright notice added to the Work, as well as any notice referring to this License.{{else}}-{{/if}}

7.3. {{#if indicateModifiedWork}}The Licensee shall indicate that it modified the Work, and retain a record of any previous modifications if applicable.{{else}}-{{/if}}

8. **Modifications**

8.1. The Licensee is {{#if derivativeWorks}}allowed{{else}}not allowed{{/if}} to create any Derivative Works. {{#if derivativeWorks}}The Licensee will make sure however, that any modifications do not cause any damages to the reputation of the Licensor or the author of the Work.{{/if}}

8.2. {{#if infective}}The Licensee will make the Derivative Work available under the same conditions as stipulated in this Agreement.{{else}}-{{/if}}

8.3. The Licensor will however in any case allow modifications which are necessary in order to exercise the rights licensed under article 2 of this Agreement, such as editorial or graphical changes necessary for editing or layout and/or technical modifications. Such modifications will not qualify as a Derivative Work.

9. **Miscellaneous**

9.1. This Agreement shall replace and supersede all previous agreements, negotiations, undertakings and correspondence between parties.

9.2. The Licensor shall not be bound by any terms and conditions of the Licensee.

9.3. Whenever possible, the provisions of this Agreement shall be interpreted in such a manner as to be valid and enforceable under the applicable law. However, if one or more provisions of this Agreement are found to be invalid, illegal or unenforceable, in whole or in part, the remainder of that provision and of these Terms shall remain in full force and effect as if such invalid, illegal or unenforceable provision had never been contained herein. Moreover, in such an event, parties shall amend the invalid, illegal or unenforceable provision(s) or any part thereof and/or agree on a new provision, in such a way as to reflect, insofar as possible, the purpose of the invalid, illegal or unenforceable provision(s).

| Date: {{date}}|

Annex II: SLC Data

The table below (Table 3) lists all the SLC Data used with the MV Copyright Licence SLC Template, in order to create an approximate version of the "Creative Commons Licences" (that we call "CC Like") and the "MV Buyer's Licence" (BL). Due to the intrinsic legal limitations, the "CC Like" versions of the CC BY-SA 4.0 and CC BY-NC-SA 4.0 cannot be created, because they cannot be considered compatible/equivalent with the original CC version of the licences. The same SLC Data are also shared with the seven SLC Templates based on the "Plain Text" version of the CC licences to enable future comparison.

Table 3: Creative Commons SLC Data

PARAMETER	CC 0 4.0	CC BY 4.0	CC BY-SA 4.0	CC BY-NC 4.0	CC BY-NC-SA 4.0	CC BY-ND 4.0	CC BY-NC-ND 4.0	BL
commercial	true			false		true	false	true
infective	false		true	false	true	false		false
makeAvailableDerivativeWorks	true					false		true
makeAvailableTranslations	true					false		true
makeNotAvailableDerivative Works	false					true		false
makeNotAvailableTranslations	false					true		false
attribution	false	true						true
indicateModifiedWork	false	true						false
licenseeFirstname	"you"							<name>
licenseeLastname	/							<surn.>
licenseeAddress	/							<addr.>
ownershipDeedID	<the identifier of the Ownership Deed SLC>							
ownershipDeedLink	<the link to the Ownership Deed SLC>							
territorialScope	"worldwide"							
reproductionNumberOfCopies	/							/
typeOfFormat	/							/
physicalNumberOfCopies	/							/
typeOfMedium	"any type of medium"							
numberOfYears	/							/
paymentAmount	/							<price>
paymentPercentage	/							/
monthlyFrequency	false							false
yearlyFrequency	false							false
oneoffFrequency	false							true
paymentMethod	/							PayPal
remunerated	false							true
sublicensable	false							false
exclusive	false							false
revocable	false							false
translations	true							true
transferable	false							false
derivativeWorks	true							true
anyForm	true							true
distribute	true							true
date	<the creation date of the SLC>							

Annex III: SLC information and SLC metadata

Table 4 lists the SLC information and metadata that are stored in the MV DAM along with a short description.

Table 4: SLC related information stored on the DAM

NAME	FORMAT	DESCRIPTION
SLC ID	ID	Identifier of the SLC
Asset ID	ID	Identifier of the Asset
List of Parent SLC IDs	List of IDs	List of identifiers of SLC
List of Parent Asset IDs	List of IDs	List of identifiers of parent Assets
List of MV User IDs	List of IDs	List of identifiers of the MV Users connected to the SLC
List of Moral Rights Holders	List of Strings	List of Moral Rights Holders' names
SLC Template Type	String	Type of the SLC Template
SLC Template ID	ID	Generic identifier of SLC Template (refers to all the versions/files of an SLC Template)
SLC Template Version	Version	Specific version of the SLC Template (e.g., 0.1.0)
SLC Template File ID	ID	Identifier of the file (e.g., "slc_template.cta") related to a specific SLC Template ID and Version
SLC Data	JSON structure	Values of the parameters of the SLC Template
SLC Document	String	Human readable output (i.e., markdown document) obtained from combining the SLC Template with the SLC Data
SC address	Hex string	ERC 1155 blockchain smart contract address that handles all the SLC of the same type
Right Split	List of couples: - BC User Address - integer	List of the Right Split (in percentage) for each user (Blockchain User Address)
Token ID	Integer	Identifier of the token related to the blockchain smart contract, that is deployed for the SLC
Transaction ID	Hex string	Identifier of the transaction related to the blockchain smart contract, that notarised the SLC
Revenue SC address	Hex string	Address of the blockchain smart contract handling the revenue split for the related SLC instance
Revenue Split	List of couples: - BC User Address - integer	List of the Revenues Split (in percentage) for each user (Blockchain User Address)
SC type	String	Indication of the blockchain smart contract type (e.g., if it manages a standard NFT)
Creation Approval List	MV User ID - Boolean	List of couples MV User ID - Boolean for the approval of the SLC
Payment List	MV User ID - Boolean	List of couples MV User ID - Boolean for the payment of the SLC
Deletion Approval List	MV User ID - Boolean	List of couples MV User ID - Boolean for the deletion of the SLC

Annex IV: Ownership Deed SLC Template

As mentioned in Annex I, MediaVerse uses the Accord Project SLC implementation that defines a SLC as composed by the SLC Template and the relative SLC Data.

In the following section is shown the updated version (v2.0.0) of the text part of the Ownership Deed SLC Template.

Ownership Deed SLC Template v2.0.0

```
# **Ownership Deed**

In this Ownership Deed, the "Owner(s)", as identified below, make the below declaration with regards to Work/Asset: {{assetID}}.

{{#ulist owners}}
{{ownerFirstname}}{{#optional ownerLastname}} {{this}}{{/optional}}, with registered office at {{ownerAddress}}, holding {{ownerShare}} percent of the total Work/Asset;
{{/ulist}}

By making this copyright protected Work available on MediaVerse, you declare that you have the legal rights to do so and to license the Work in accordance with the licensing choices you will make and that you are not in breach of any agreement.

This means that you declare that you are either the sole owner of the copyrighted work (whether as an author or after transfer of such rights), or that you have contractual rights allowing you to make the Work available on MediaVerse and license it. This could be a contract between the different owners in case of joint copyright ownership (which could arise from co-creation), or a license that allows you to sublicense the Work.

MediaVerse allows many different licensing options, but any license will at least include a reproduction right and the right to make the work available through MediaVerse. When allowing a certain type of license through MediaVerse, you must ensure that you have the legal rights to do so.

You acknowledge that the moral rights to the Work/Asset remain exclusively with the author(s) of the Work/Asset at all times, i.e.
{{#ulist moralRightsHolders}}
{{moralRightsHolderFirstname}}{{#optional moralRightsHolderLastname}} {{this}}{{/optional}};
{{/ulist}}

You also declare that you are solely responsible for your use of MediaVerse and that you bear full responsibility in case of granting a license for which you do not have the legal rights to do so. This includes that you will hold harmless, indemnify and defend in law MediaVerse and the entity or entities exploiting MediaVerse in consequence of any issue with regards to the copyright of the Work you made available on MediaVerse.

| Date: {{date}}|
```

MediaVerse is an H2020 Innovation Project co-financed by the EC under Grant Agreement ID: 957252.