TALON

Autonomous and Self-organized Artificial Intelligent Orchestrator for a Greener Industry 4.0

Deliverable

D4.2 Lightweight Decentralized Blockchain Toolbox

Actual submission date: 29/11/2024

Funded by
the European Union

Project Number:	101070181		
Project Acronym:	TALON		
Project Title:	Autonomous and Self-organized Artificial Intelligent Orchestrator for a Greener Industry 4.0		
Start date:	October 1st, 2022	**Duration:**	36 months
D4.2 Lightweight Decentralized Blockchain Toolbox			
Work Package:	WP4		
Lead partner:	SIDROCO Holdings Ltd (SID)		
Author(s):	Konstantinos Kyranou (SID)		
Reviewers:	UBITECH, 8BELLS		
Due date:	29/11/2024		
Deliverable Type:	DEM	**Dissemination Level:**	PU
Version number:	1.0		

Revision History

Version	Date	Author	Description
0.1	30/09/2024	SID	First release of the Table of Contents
0.2	08/11/2024	SID	Shared version for internal review deadline
0.3	15/11/2024	UBITECH, 8BELLS	Internal review
0.4	22/11/2024	MINDS	Quality review
1.0	29/11/2024	ENG	Final coordinator review before submission

List of figures

List of Tables

Definitions and acronyms

AI	*Artificial Intelligence*
API	*Application Programming Interface*
AR	*Augmented Reality*
CA	*Certificate Authority*
CAV	*Connected and Autonomous Vehicle*
CFT	*Crash Fault Tolerance*
CID	*Content Identifier*
CPU	*Central Processing Unit*
DB	*Database*
dApp	*Decentralised Application*
DeFi	*Decentralised Finance*
DDoS	*Distributed Denial of Service*
DLT	*Distributed Ledger Technology*
DoA	*Description of Action*
DPoS	*Delegated Proof of Stake*
E2C	*Edge to Cloud*
EC	*European Commission*
EU	*European Union*
FL	*Federated Learning*
GA	*Grant Agreement*
GHz	*Gigahertz*
IBM	*International Business Machines*
HTTPS	*Hypertext Transfer Protocol Secure*
I5.0	*Industry 5.0*
ID	*Identification*
IoT	*Internet of Things*
I/O	*Input/Output*
IPFS	*InterPlanetary File System*
ISP	*Internet Service Provider*
JSON	*JavaScript Object Notation*
KPI	*Key Performance Indicator*
kWh	*Kilowatt-hour*
MB	*Megabytes*
ML	*Machine Learning*
MSP	*Membership Service Provider*
PaaS	*Platform as a Service*
PBFT	*Practical Byzantine Fault Tolerance*
PC	*Project Coordinator*
PoS	*Proof of Stake*
PoW	*Proof of Work*
PPE	*Personal Protective Equipment*
SDK	*Software Development Kit*
SQL	*Structured Query Language*
SSL	*Secure Sockets Layer*
TC	*Technical Coordinator*
TPS	*Transactions per Second*
TWh	*Terawatt hours*
UC	*Use Cases*
UI	*User Interface*
URL	*Uniform Resource Locator*
UX	*User Experience*
VR	*Virtual Reality*
W	*Watt*
ZT	*Zero Trust*

Disclaimer

Executive Summary

The lightweight decentralised blockchain toolbox is a scalable solution designed to fortify the integrity and robustness of TALON's platform. This module functions as a core technology for developing the methods and components designed to boost the privacy, traceability and immutability of the Edge to Cloud (E2C) network of TALON's Security layer. Along with AI, TALON harnesses blockchain to advance beyond Industry 4.0, developing intelligent platforms and introducing innovative service models and applications, all while maintaining strong privacy and security protections. For the scope of this project, blockchain technology is employed to offer everlasting and tailored protection through sophisticated security, privacy, and trust mechanisms.

This document gives a holistic overview of the developments and research efforts undertaken towards the realisation of TALON's blockchain. It showcases the progress and achievements made in this domain and it presents detailed elaborations of the methodologies employed, the innovations achieved, and the initial outcomes, offering a complete review of the work conducted. This detailed exposition aims to highlight the advancements and contributions made, ensuring a clear understanding of the significant steps undertaken to address the pilot requirements and to further contribute to the research and development community.

This work outlines the initial, yet very important phase of TALON's blockchain development and validation process. With the maturity of the implementation being high in terms of efficiency and functionality, the initial implementation is in the process of being deployed to the pilots. Of course, this exercise along with the results originating from the pilot demonstrations will be documented at D4.3 "Final Reusability, Explainability, Trustworthiness, Security & Privacy mechanisms", which is due at M26.

I Introduction

1.1 Purpose of the Deliverable

This document, titled “Lightweight Decentralized Blockchain Toolbox” has been prepared to outline and detail the:

1. Research performed to check the existing cutting-edge approaches,
2. Architectural schema designed to meet TALON’s specific requirements,
3. Development methodology followed to keep up and further boost the technological capacity of this tool,
4. Tangible results in terms of metrics and visualisation outputs,
5. Validation strategy,
6. Expected next steps with regard to the remaining lifecycle of the project.

This deliverable has been produced as part of the Work Package 4 (WP4) technical activities and highlights the blockchain’s merits through the prism of TALON’s core system concepts, namely security, privacy, and trustworthiness. This work is strongly correlated with TALON’s vision, as it aligns with the main pillar that states the decentralized blockchain approach should ensure robust security, privacy, and trust across diverse application domains, to drive the evolution of Industry 4.0 into smart platforms and to pave the way for innovative service models.

In this work, a brief synopsis of the current state-of-the-art methods and trends related to blockchain is presented to demonstrate TALON’s scientific soundness. The various methods and technologies employed are also elaborated upon to substantiate the security, privacy, and trustworthiness specifications of the TALON platform. Finally, the validation actions are outlined to verify the platform’s compliance with key qualitative and quantitative indicators, along with the planned and anticipated actions.

1.2 Relation with other Work Packages, Tasks and Deliverables

Blockchain toolkit is developed under the Task 4.4 “Security & Privacy Blockchain Mechanisms”, which takes into consideration two key input documents: the “Use Case, KPIs, Requirements, Specification, Slices & Technology Enablers Definition Report" from Deliverable 2.1 (D2.1), and the “Installation & Demonstration Planning, Evaluation Methodology & KPIs Definition Report” from Deliverable 5.1 (D5.1). These documents serve as the cornerstone of references for the task's execution, ensuring alignment with the project’s objectives and technical requirements. D2.1 is the major document for the precise and thorough development of this deliverable, as it outlines the key performance indicators (KPIs) and specifies the technical criteria for the Use Cases (UCs), while also presenting an initial assessment of the challenges which the demonstrators confront. These aspects help shaping a clear framework for the subsequent development and implementation phases. D5.1 functions as the main point of reference regarding the installation and evaluation of the different TALON modules.

Deliverable 4.2, "Lightweight Decentralized Blockchain Toolbox" (D4.2) is not explicitly referenced as a contributing input to any other task or deliverable in the Description of Action (DoA). Nonetheless, it will serve as the foundational and reference work for the final pilot outcomes that will be highlighted on deliverable 4.3 “Final Reusability, Explainability, Trustworthiness, Security & Privacy mechanisms” (D4.3).

1.3 Structure of the Document

Following the introductory remarks, the document is organized as follows to facilitate a clear and structured presentation of all relevant aspects:

- Section 2: This section provides an overview of the existing blockchain landscape and a short iteration over the most hyped blockchain technologies, categorising them into several types and exploring their practical applications in the real world. It also examines the landscape of decentralised applications, highlighting their significance and impact within TALON.
- Section 3: This section outlines the relevance and coherence of the developed distributed ledger technology with TALON's requirements, highlighting the added values of the implementation.
- Section 4: This section details the architectural design and methodology followed to realise TALON's blockchain toolkit, especially focusing on its internal architecture and design principles. It also explores the interconnection of various submodules, outlining how they interact to form a cohesive and functional system.
- Section 5: This section covers the implementation of TALON's blockchain toolkit, elaborating the development process and mentioning the technical specifications involved. It also demonstrates the communication interfaces as well as the deployment strategies, illustrating how the toolkit is operationalised.
- Section 6: This section outlines the validation process for TALON's blockchain toolkit, detailing the systematic approach followed to ensure a thorough evaluation. It presents both scientific and technical results, demonstrating the effectiveness and reliability of the toolkit towards the key security, privacy and trust objectives of this task.
- Section 7: This section summarizes the key findings and outcomes related to TALON's blockchain toolkit, reflecting on the achievements and insights gained throughout the exercise. Moreover, it illustrates the pending actions required to complete the refinement of TALON's blockchain toolkit, as well as the outlook for its implementation.

2 Existing Knowledge & Background Technologies

2.1 Blockchain Technologies Categorisation

Blockchain technology can be broadly categorised based on different criteria: consensus mechanisms, level of decentralisation, performance, privacy features, and all the specific use cases they serve [1], [2]. Notably, understanding these big categories significantly supports learning how blockchain technology can be adapted for multiple applications, especially in innovative and demanding environments such as the TALON.

Public blockchains are open and decentralised; hence, anyone can participate. Their consensus mechanism is usually Proof of Work (PoW), Proof of Stake (PoS), or Delegated PoS (DPoS). The PoW consensus model requires nodes to solve complex mathematical puzzles, proving they have expended significant computational resources before adding a block to the blockchain [3]. PoS consensus mechanism allows validators, who hold significant stakes in the network, to create new blocks [4] and in DPoS, stakeholders vote to elect a small group of trusted delegates, known as "witnesses," who are responsible for validating and generating new blocks [5].

Public blockchains are also widely known for their security and transparency, hosting Decentralised Applications (dApps). Indicative examples of public blockchain networks are Bitcoin and Ethereum, which have so far evidenced that significant security can be provided with public blockchains across borders on a global scale. As shown in the Figure 1, all the nodes have access to the blockchain and its information. The downside with public blockchains is the scalability problem, in which transactions raise concerns regarding speed. Since the number of users goes up with transactions, the time used in a consensus process increases; hence, it leads to increased latency, higher transaction costs and higher energy consumption. TALON requires a scalable and lightweight solution; thus, the adoption of a public decentralised ledger technology is not an appropriate option. Despite that, public blockchains remain extremely popular due to the high levels of transparency required in applications such as Decentralized Finance (DeFi) and public voting systems [6].

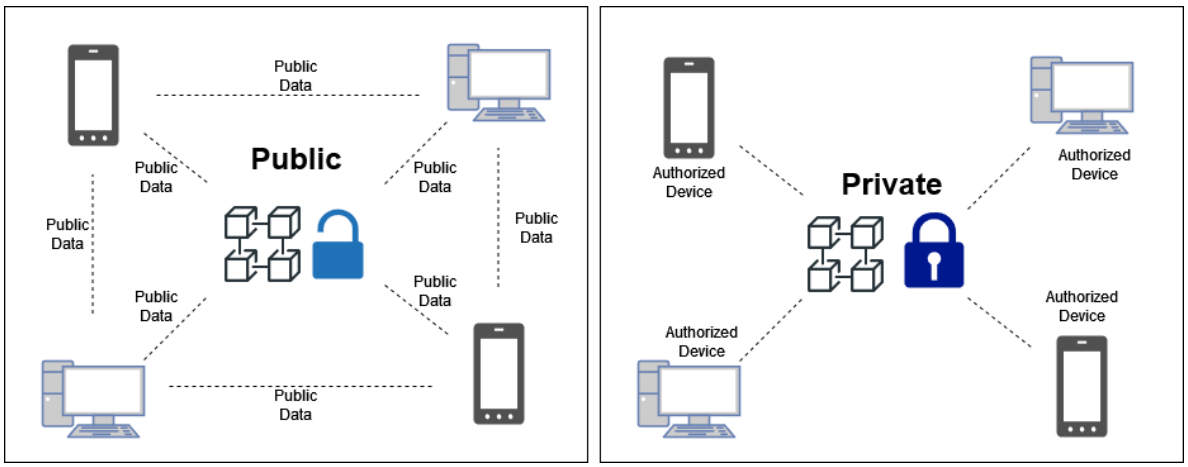

Figure 1: Comparison of Public and Private Blockchain

A private blockchain is a restricted network in which the participants are authorised selectively, as shown in the Figure 1. Single-organisation or consortium-controlled environments operate on private blockchains. Advantages include faster transaction speed, better privacy, and more control of the network governance. However, complete decentralisation may not be delivered at the same level as in public blockchains. An example is Hyperledger Fabric, which is for enterprise solutions where

confidentiality of data and controlled access are crucial [7]. TALON utilises Hyperledger Fabric to enable secure and scalable operations within its industrial applications, providing a controlled environment that meets specific regulatory and security requirements.

A consortium blockchain, or federated blockchain, concerns a group of organisations that maintain the blockchain network together, as shown in Figure 2. In the case of the consortium blockchains, access is restricted to specific entities; hence, these offer more efficiency and privacy compared to public blockchains. Each one of these blockchains lends itself to applications that involve multiple parties-suppliers, partners, or regulators-who would want to access and validate the data in a very secure manner: cross-border payments, shared KYC compliance, and trade finance [8].

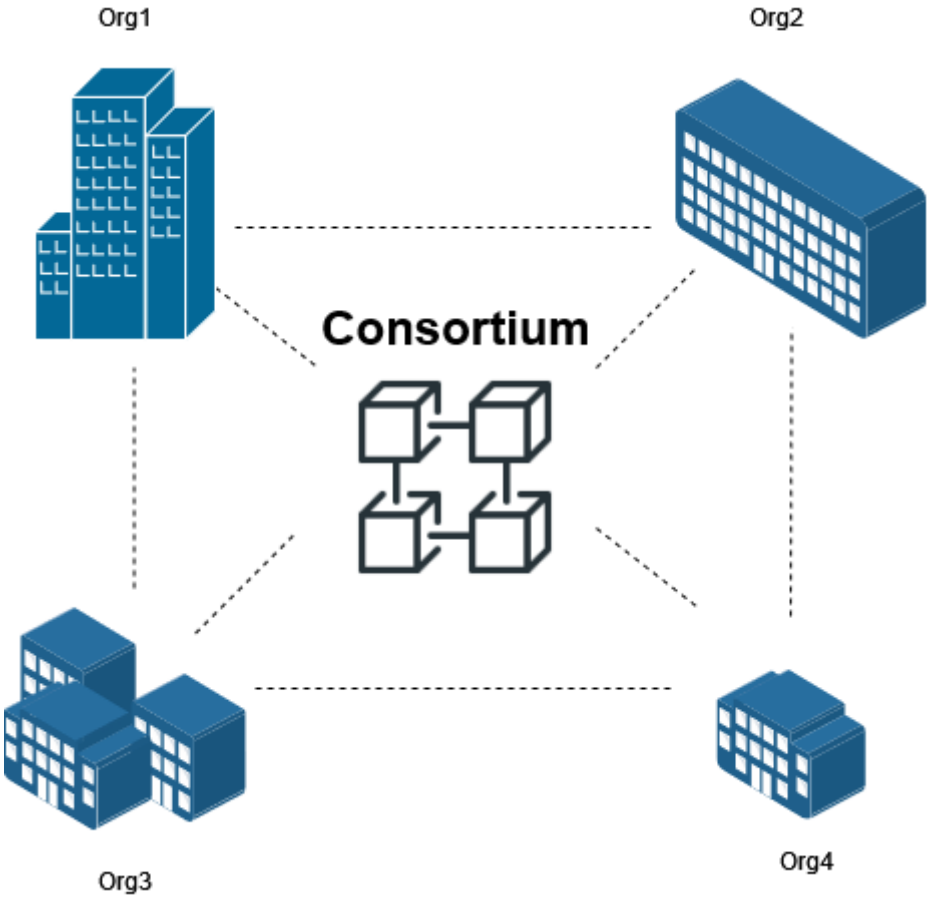

Figure 2: Consortium Blockchain

Hybrid blockchains bring in features from both the public and private blockchains, thus offering flexibility in the network to tap into advantages arising out of both models. They make some data or transactions public, while others remain private, known only to selected participants with, as shown in the Figure 3. This balance of transparency and confidentiality is useful in applications like public auctions, government records, or healthcare data management, wherein strict data privacy is needed but where public verification could be required [9].

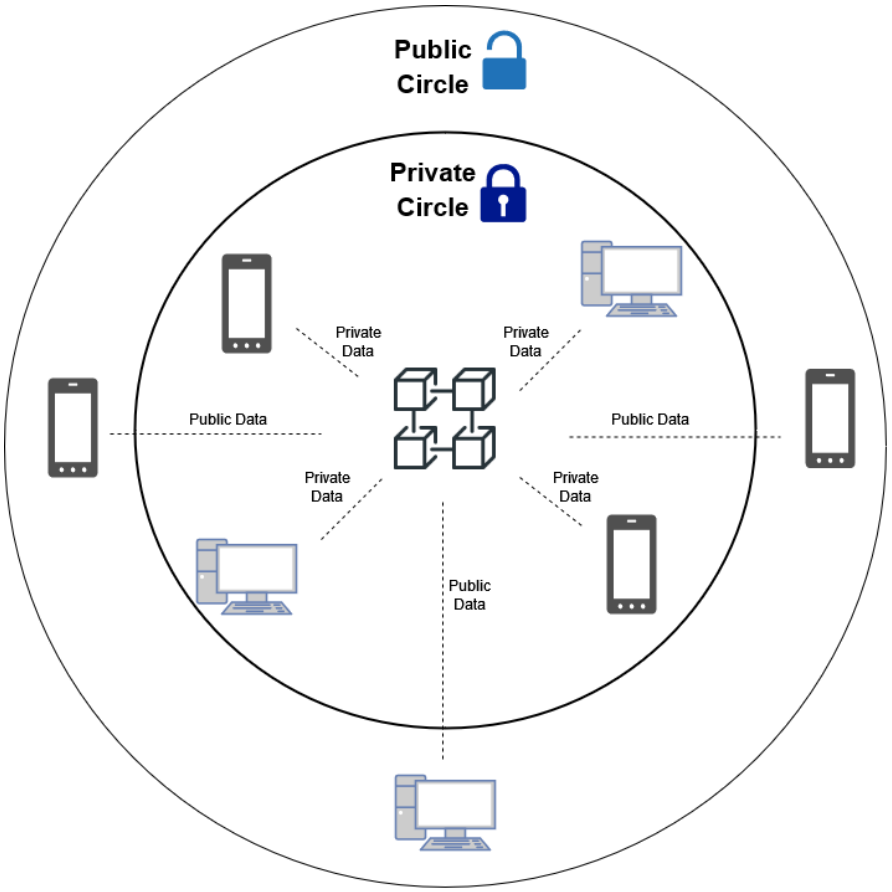

Figure 3: Hybrid Blockchain

In Figure 4, the diagram categorizes blockchain networks based on their permission models. It depicts all the differences of Permissionless and Permissioned blockchains, having the presence of a Hybrid model overlapping in both. Permissionless blockchain: the public blockchain does not have any central authority, and hence open participation is allowed with full decentralization. On the permissioned side, Private blockchains are limited to selected participants, one central authority is in control, which makes it appropriate for organizational control, while Consortium blockchains further distribute such control among a preselected group that often finds its application in collaborative industry networks. The Hybrid blockchain model combines features of both, enabling partial openness with selective access controls, balancing transparency and privacy for versatile applications.

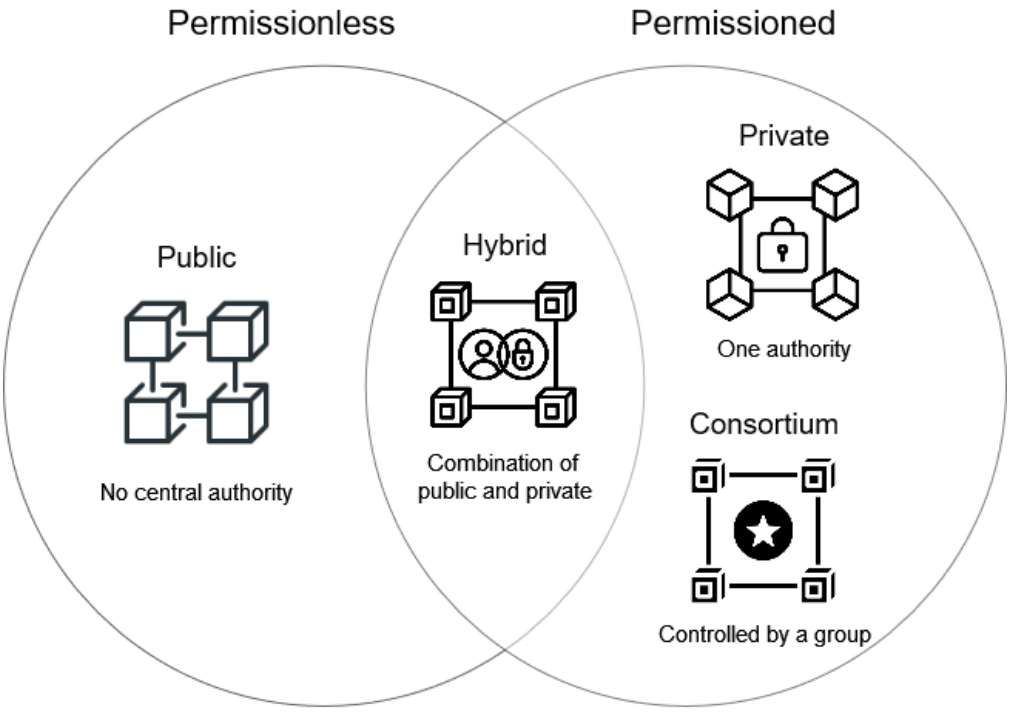

Figure 4: Permissionless vs Permissioned Blockchains

Layer 2 solutions are protocols built on top of existing blockchains that increase scalability and efficiency. Most famous examples include the Lightning Network for Bitcoin [10] or Optimistic Rollups for Ethereum [11]. They are all proposals designed to conduct transactions off the blockchain chain while using the security and decentralization of the main network. In contrast, sidechains represent independent blockchains that are interoperable with the main chain; thus, applications can run on them independently, anchoring data from time to time on the mainchain for security.

Specialised blockchains focus on very specific use cases or industries. Examples are IOTA [12], that scopes on the integrity of data in Internet of Things (IoT) applications, and Quorum [13], which is used in finance when privacy to high levels may be required, and the demand for high speed, low latency transactions is important.

Table 1 summarises these details, classifying the blockchain technologies by type, consensus mechanisms, levels of privacy, scalability, and suitability for industry use. It also shows how each blockchain instance, serves very specific application requests. This classification provides an overview to show blockchain's adaptability to various use case applications, with determined requirements, within dynamic evolving environments (industry, financial services IoT applications, etc).

Table 1: Blockchain Technologies Categorisation

Blockchain	Consensus Mechanism	Smart Contracts	Privacy Features	Scalability	Throughput (TPS)	Modularity	Industry Suitability

Bitcoin (Public)	Proof of Work (PoW)	No	Low	Low	Low (<10)	No	Cryptocurrency
Ethereum (Public)	Proof of Stake (PoS) / Proof of Work (PoW)	Yes	Medium	Medium	Medium (<30)	No	DeFi and dApps
Hyperledger Fabric (Private/Permissioned)	Raft / Practical Byzantine Fault Tolerance (PBFT)	Yes	High	High	High (up to 1000+)	Yes	Industry, Supply Chain, IoT
Quorum (Permissioned)	Raft	Yes	High	High	High	Partial	Finance
IOTA (Specialized)	Tangle	No	Low	Medium	Medium	No	IoT
Corda (Permissioned)	Notary	Yes	High	Medium	Medium	Partial	Finance, Industry

2.2 Real-world applications and TALON relevance

From transportation and manufacturing to virtual reality and IoT, the wave of blockchain technology is transforming sectors by offering secure, decentralized solutions to long-standing challenges. Most importantly, this includes the need for tamper-resistance and transparent data communication.

During the last few years, connected and autonomous vehicles (CAVs) have emerged as a transformative technology that is going to reshape urban mobility and public transports. A key challenge is ensuring that hundreds of IoT-enabled sensors and devices on these vehicles can communicate securely, transparently, and without any risk of tampering. Blockchain, due to its decentralised framework for logging, verifying, and maintaining data integrity provides solutions. Embedding blockchain into the CAV framework can securely handle sensitive data, such as vehicle location and sensor readings, as shown in Figure 5. Incorporating secure and decentralised mechanisms of data exchange keeps CAVs more reliable in trusted communications for safety and efficient transportation networks [14]. While TALON's demonstrators do not explicitly include automobiles, efficient data broadcasting and trustworthy data networks are at the heart of the project, and to TALON's blockchain implementation by extension.

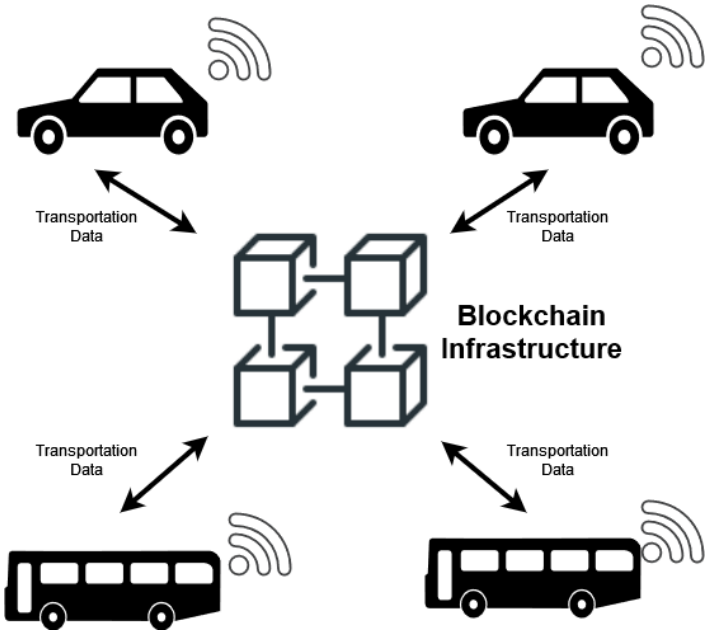

Figure 5: CAVs integration with Blockchain

Industry 4.0 aims at an entirely integrated, more effective, even flexible environment of production. One of the most persistent challenges towards shifting to smart manufacturing is secure, reliable, and traceable communication from all diversified components within the same infrastructure. These are the kind of issues that blockchain addresses, offering in return an immutable decentralised ledger to help improve transparency, security, and manufacturer stakeholder confidence. For example, smart contracts make the process of contracting between the manufacturers and their suppliers more efficient. This has a cost effect due to the automatic execution of contracts concerning terms of service delivery. Middleware frameworks use blockchain to securely handle and verify transactions, allowing different manufacturing systems to work together safely and efficiently—an important feature for automation in Industry 4.0 [15].

Integration of blockchain with Augmented Reality (AR) and Virtual Reality (VR) provides significant added value with data security, collaboration, and operational transparency, in industrial fields. Blockchain's decentralised nature ensures that sensitive information—such as 3D designs, operational data, and real-world visualisations—are secure and tamper-proof when being shared across AR/VR platforms. This becomes even more critical in cases such as remote maintenance, where AR overlays critical system information for technicians, and blockchain secures this information so that only authenticated users would access and manipulate the data. In VR environments, on other occasions, blockchain enables the creation of decentralised, trusted collaboration spaces where users can interact with each other in real-time simulations as well as implement designs [16].

The fusion of federated learning (FL) with blockchain enables IoT applications securely and in a decentralised manner train the models, even across devices in a distributed fashion, without compromising data privacy, security and trust. This is a particularly interesting concept especially within TALON. As shown in Figure 6, each IoT device (or cluster) autonomously trains a local model on their data and shares model updates, not raw data, on a decentralised in this setting. Blockchain provides security to this federated system through an immutable, tamper-resistant record of the model updates, thereby paving the way for the discovery of potential malicious activities or faulty contributions by any peer participant. This hybrid approach, which can be implemented with multiple variations, balances the trade-offs between privacy and performance [17]. This concept can be generalised with any smart IoT-enabled device that could part of a network of entities. The entities,

such as smartphones, local ML models from pc's, analytics data and even cloud computing logs can be part of private or public DLTs, allowing access either to specific stakeholders or to the public. Such information is secure due to the inherent immutability of the network and trustable because it originates from validated peers.

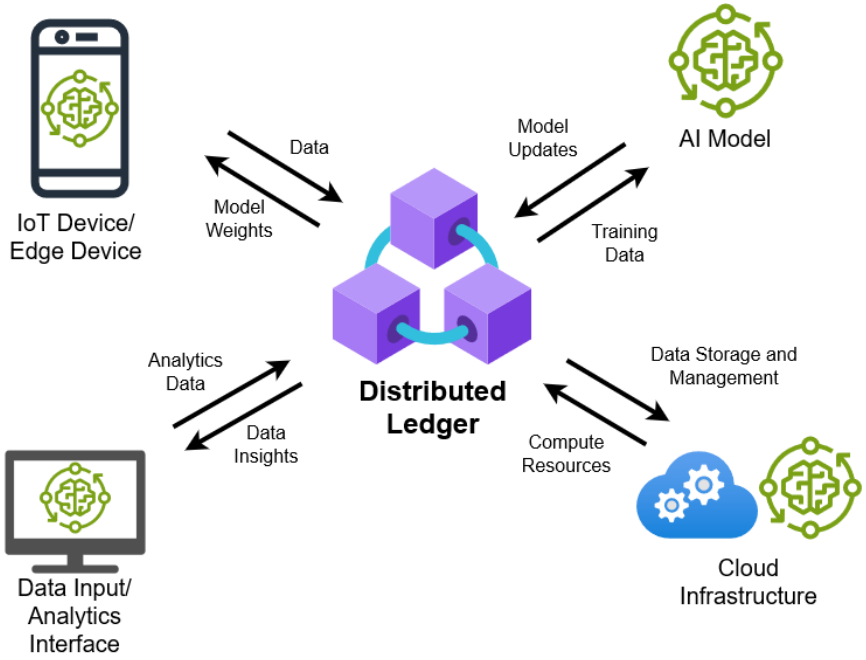

Figure 6: IoT integration with Blockchain

From connected and autonomous vehicle applications, Industry 4.0 automation, AR/VR, to Federated Learning and IoT devices, the presence of blockchain foretells a transformation across diversified domains. It is, in fact, setting new benchmarks for operational transparency, efficiency, and reliability by availing decentralised tamper-proof solutions for secure communication and data integrity.

2.3 Decentralised applications

Decentralised applications, or dApps, are a class of software systems that operate on decentralised networks. Unlike most typical applications, which work on centralised servers, the inherent nature of blockchain technology makes dApps significantly more secure and transparent, with users having full control over their actions. DApps expand the capabilities of blockchain systems, from simple financial transactions to complex and versatile use cases, marking a great technological shift in designing and deploying software applications [18].

To understand dApps better, it is important to examine their unique characteristics. A dApp is an application based on software, but it has some distinct functionalities. Some basic features of dApps include being open source, featuring a decentralised consensus mechanism and functioning in such a way that failure cannot emerge from one central point. As regards the open source, it means their code is open to anyone; thus their functionality is transparent. For the decentralised nature of dApps to be kept, these depend on specific blockchain consensus mechanisms: PoW, PoS or other. These models help users in the system agree upon the state of the application without having a central authority [19]. By distributing data over a decentralised network, as depicted in Figure 7 dApps reduce the risk associated with a single point of failure, enhancing the system's robustness against attacks or outages. Furthermore, Internet Service Providers (ISPs) play a crucial role in connecting users to the decentralised network, by facilitating communication between devices. TALON implements these

decentralised applications using blockchain consensus to ensure transparency, security, and trust within every application that it powers.

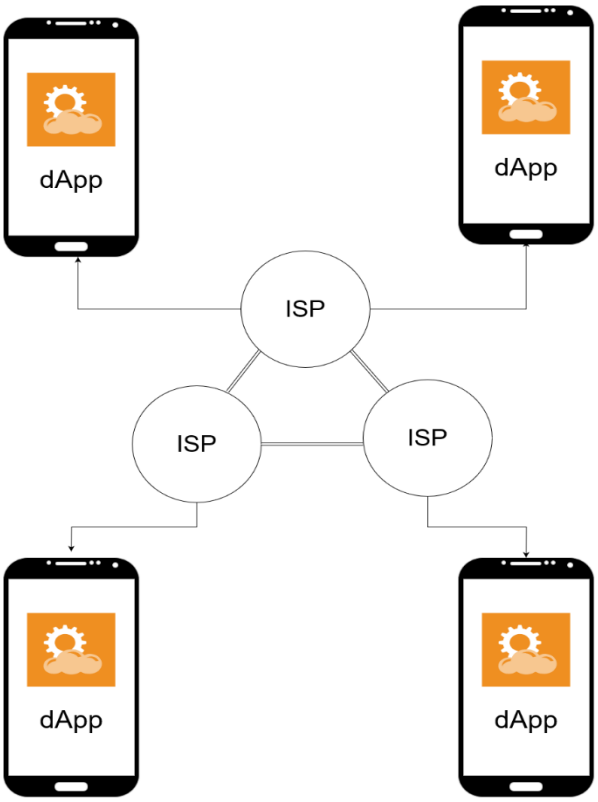

Figure 7: Decentralised Application (dApp)

There are several advantages that decentralised applications offer over their centralised predecessors. Figure 8 shows the potential added values of utilising dApps. More specifically:

- Transparency: dApps rely on open-source code. Hence, everyone is allowed to view and verify the inner mechanisms. Moreover, the transactions made are visible to all members of the network.
- Security and Privacy: using blockchain technology, data can be spread across a decentralised network securely, hence making it highly tamper-proof and resistant to unauthorised access and such, sensitive information is not passed on to a central server, which can be hacked or leaked.
- Lower Cost: dApps work in decentralised networks, and since there are no intermediaries involved, third-party service costs are reduced. The cost efficiency of dApps makes them more accessible and affordable to a greater group of both users and businesses.
- User Control: users are allowed to have ownership of their assets and data in a way that prevents any centralized entity from unilaterally changing or censoring information.
- Autonomy: once a dApp has been deployed onto a blockchain, it will function autonomously without any further intervention or oversight by any central party

DApps offer enhanced security and privacy, due to their decentralised nature and reliance on cryptographic principles. This architecture reduces reliance on a central authority, making them more resilient to attacks and ensuring user data remains private and secure. For example, the use of consensus mechanisms and cryptographic security makes it difficult to attack with Distributed Denial

of Service (DDoS) attacks and attempts of data breach. Moreover, because they use blockchain technology, there is transparency over the network so there is trust among all the participants [10]. DApps also empower users with more control over data and assets, without intermediaries and at a lower cost, while increasing their autonomy.

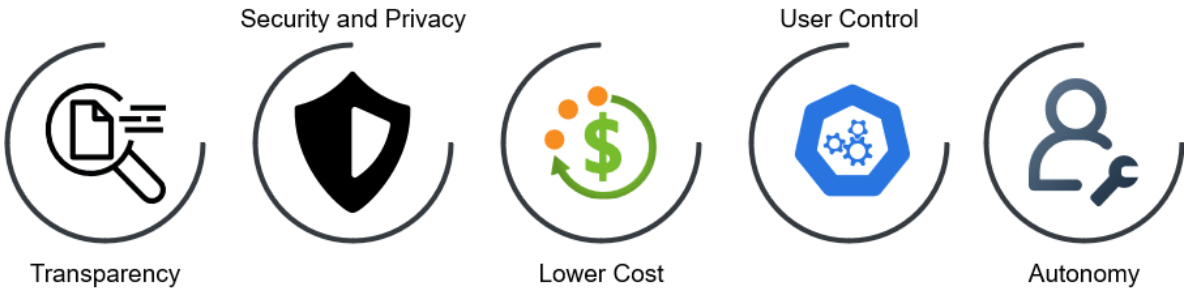

Figure 8: Conceptual benefits of dApps

In recent years, the architectures of dApps have been highly developed to meet the requirements of diversified application scenarios which employ the blockchain technology. Previous research identified at least four main architectures that existed: Native Client as a dApp, Smart Contract as a dApp, Web & Contract as a dApp, Fully-Decentralized dApp [20]. The "Web & Contract as dApp" model—utilising a web-based front end along with blockchain-backed smart contracts—has been one of the most widely used due to user interfaces and ease of deployment. This means that these models bring back some aspects of centralisation, particularly on hosting the front end on some centralized servers, often accompanied by the problems of security and transparency. In comparison, the "Fully-Decentralized dApp" architecture is one step forward in complexity and security. Such a design eliminates all centralsed servers, and so increases the privacy of users while reducing potential vulnerabilities. However, such a purely decentralised model faces many practical problems: insufficient scalability and even increased complexity during development. Future research opportunities are about optimising these architectures for performance, security, and usability; creating novel hybrid models; and tackling specific economic and regulatory challenges associated with each architectural approach [21].

Despite their advantages, decentralised applications also face significant challenges. The most critical challenges faced by decentralised applications (dApps) currently are performance-related constraints like poor throughput and high latency. Major networks like Ethereum have been revealed as unable to support a high frequency of transactions, thus resulting in getting clogged and slower. Another major issue is security vulnerabilities in smart contracts. These contracts, once deployed, are unchangeable or cannot be fixed easily, which has the potential to cause massive loss if exploited. Furthermore, high transaction costs create economic constraints that become further burdensome to access especially in applications that are based on frequent interaction or microtransactions [22]. These are the problems that need to be addressed to release decentralised applications' full potential in industry. TALON also faces similar inherent challenges in ensuring low latency and high throughput while simultaneously maintaining security and privacy, especially when handling high volume and frequency of transactions or data exchanges across its decentralised network.

Various examples illustrate how different blockchain frameworks are used in dApps to meet specific needs. Different blockchain frameworks provide various characteristics serving to several use cases; hence, the flexibility and technical potential of dApps. For instance, Ethereum represents a public blockchain platform with widespread adoption and support for smart contracts written in Solidity. This

obviously makes it great to develop DeFi applications — things like Uniswap [1]or Compound[2] that use transparent, automated, and secure transactions. However, this Proof of Work (PoW) consensus could lead to reduced throughput and hence higher costs and less scalability for applications requiring frequent transactions.

Hyperledger Fabric [23], on the other hand, is developed for enterprises and is a permissioned blockchain—more flexible in that it supports multiple consensus mechanisms, among them Practical Byzantine Fault Tolerance (PBFT) and Raft. Such flexibility in Hyperledger Fabric would allow for higher transaction throughput, of importance on use cases such as supply chain management, where secure, fast, and private transactions are crucial. It supports smart contracts that are written in general-purpose languages such as Java, Go, and others, making it less necessary for developers to learn new languages. TALON leverages Hyperledger Fabric to handle diverse requirements in its industrial applications, utilising its consensus mechanism to ensure secure, efficient, and scalable data management across different edge and cloud environments.

Quorum [24], for instance, is a permissioned blockchain built on Ethereum that comes with features for privacy and high speeds of transactions, all designed to fit the needs of financial applications that require confidentiality, such as private banking networks. Corda, made by R3, is a platform that specifically focuses on financial agreements and supports the development of decentralised applications and protocols, which have complex workflows and also ensure the entities are regulatory-compliant; it also has a unique form of consensus where only relevant parties take part in the process of validating. However, these implementations align closely with financial use cases, rather than industrial requirements, making them less suitable for the requirements of TALON.

Role of Decentralized Applications in TALON

The realisation of dApps in the vision of TALON drives and overcomes these advantages and challenges. Decentralised blockchain technology is used to build high security, privacy, and trust in the heterogeneous application requirements of TALON. It promotes the transformation of industrial environments into efficient next-generation platforms by employing new service models. The use of the blockchain ensures a secure and transparent environment for the independent, scalable AI operations being designed by TALON, and handles all information/data stream across the network to realise greener and more efficient AI operations. This information may originate from federated resources or standalone applications.

In this regard, TALON employs decentralised applications to ensure full, end-to-end, customised, perpetual security and privacy in the E2C AI architecture. Using the blockchain mechanisms enables TALON to operate high-throughput and low-latency dApps with strong security via ZT (zero trust) deployment for edge AI. That will empower TALON to scale, increase flexibility, harden, and boost other critical features available during deploying AI solutions across the Edge AI networks in support of the conceptual notion of distributed/federated intelligence through decentralised networks.

[1] *https://app.uniswap.org/*
[2] *https://compound.finance/*

3 TALON's Blockchain toolkit value proposition

The TALON blockchain toolkit contributes significantly to the overall objectives of TALON, by ensuring security trustworthiness and privacy are embedded into the core architecture. Decentralised blockchain platforms, with immutable records, are highly suited for applications that are demanded in the context of real-world pilots of TALON. This section demonstrates how the blockchain toolkit is applied in the context of TALON, highlighting real-world scenarios from the project that illustrate the benefits of this technology. By integrating blockchain, TALON enhances transparency, efficiency, and security in its processes.

The blockchain toolkit of TALON is designed in such a way that communication and data handling within a system are decentralised. TALON's framework shaped on blockchain technology ensured that all the transactions of data between different devices and/or cloud systems are verified and secure in diverse environments. In realistic state-of-affair applications, the blockchain toolkit provides decentralised authentication, data privacy, and secure communications across diverse components in the TALON ecosystem. These qualities come without sacrificing significant computational resources, thus maintaining an optimal balance between security, efficiency, and performance. The technical details of how this exercise has been realised will be thoroughly elaborated in the sections to follow.

The core underlying concepts behind TALON's blockchain network implementation are (i) security, (ii) privacy and (iii) trustworthiness. Those concepts are technically addressed by integrating TALON's permissioned blockchain with the FL module. Blockchain effectively addresses FL challenges by providing a decentralised and transparent framework for private data sharing and immutable model updates.

One major challenge in FL is ensuring the integrity of model updates, as malicious participants could tamper with data or introduce poisoned models. Blockchain tackles this by recording all transactions—such as model updates in TALON case—on a tamper-proof, decentralised ledger, ensuring transparency and traceability. Information is stored on the IPFS system, while the hash is stored on the ledger itself. This helps with the scalability and lightweight nature of the proposed solution. Another challenge is the lack of trust among participants, particularly when FL involves multiple organisations. Blockchain boosts trust through its consensus mechanisms, which validate and agree on the ML algorithm updates without relying on a central authority. By decentralising the coordination process, blockchain removes the risk of a single point of failure, enhancing the overall robustness and reliability of FL systems.

All the generic concepts are realised through the integration of blockchain toolkit to the FL processes which is directly connected to some of the TALON's demonstrators. More specifically, blockchain could work as a secure medium of dealing with sensitive production data. This is exactly the case with the I4.0 pilot which employs ML for optimising the processes and workplans of industrial manufacturing. Utilising AI on a manufacturing line enables the detection of patterns that could lead to tool breakages, faulty parts and machine issues. Employing a permissioned blockchain that stores the ML models adds significant value on a manufacturing line by enhancing the security and reliability of AI-enabled predictive maintenance solutions. By enabling the detection of patterns that could lead to tool breakages, faulty parts, and machine issues, this system ensures that ML models are securely stored in a decentralised manner, reducing the risk of data loss or tampering. This integrated approach is expected to be finalised in the upcoming months of the project. In addition to the above, blockchain is integrated with the FL techniques that are used in the project. From a methodological standpoint, TALON provides significant advantages of distributed ledgers to the FL architecture, as previously discussed. This is achieved while utilising minimal computational resources, making TALON an efficient and valuable solution for enhancing the performance and security of FL systems.

4 Architectural Design & Methodology of TALON's Blockchain toolkit

This section provides a structured analysis of the architecture and design of the TALON blockchain toolkit, progressing from a high-level demonstration to a more detailed technical depiction. The initial part offers a comprehensive overview of the architecture's primary components and their core functionalities, focusing on the unique capabilities of the blockchain solution. Following this, the document delves into the technical specifics of each architectural layer, including interconnections, methodologies, and the rationale behind design choices. This layered approach enables a holistic understanding of the TALON blockchain toolkit, and how it combines the FL with the blockchain layer, the decentralised storage and the visualisation interface.

4.1 Internal High-Level Architecture and Design

TALON's blockchain architecture, as shown in Figure 9, is composed of four distinct components, with the first being the user visualisation layer. This, in turn, includes both the front-end and back-end logics of the blockchain. With Hyperledger's Node SDKs[3], the blockchain API connects directly to the network's peers to retrieve committed transactions. These transactions are then stored and managed within a PostgreSQL[4] database and the data are visualised through the front-end interface.

[3] *https://hyperledger.github.io/fabric-sdk-node*
[4] *https://www.postgresql.org*

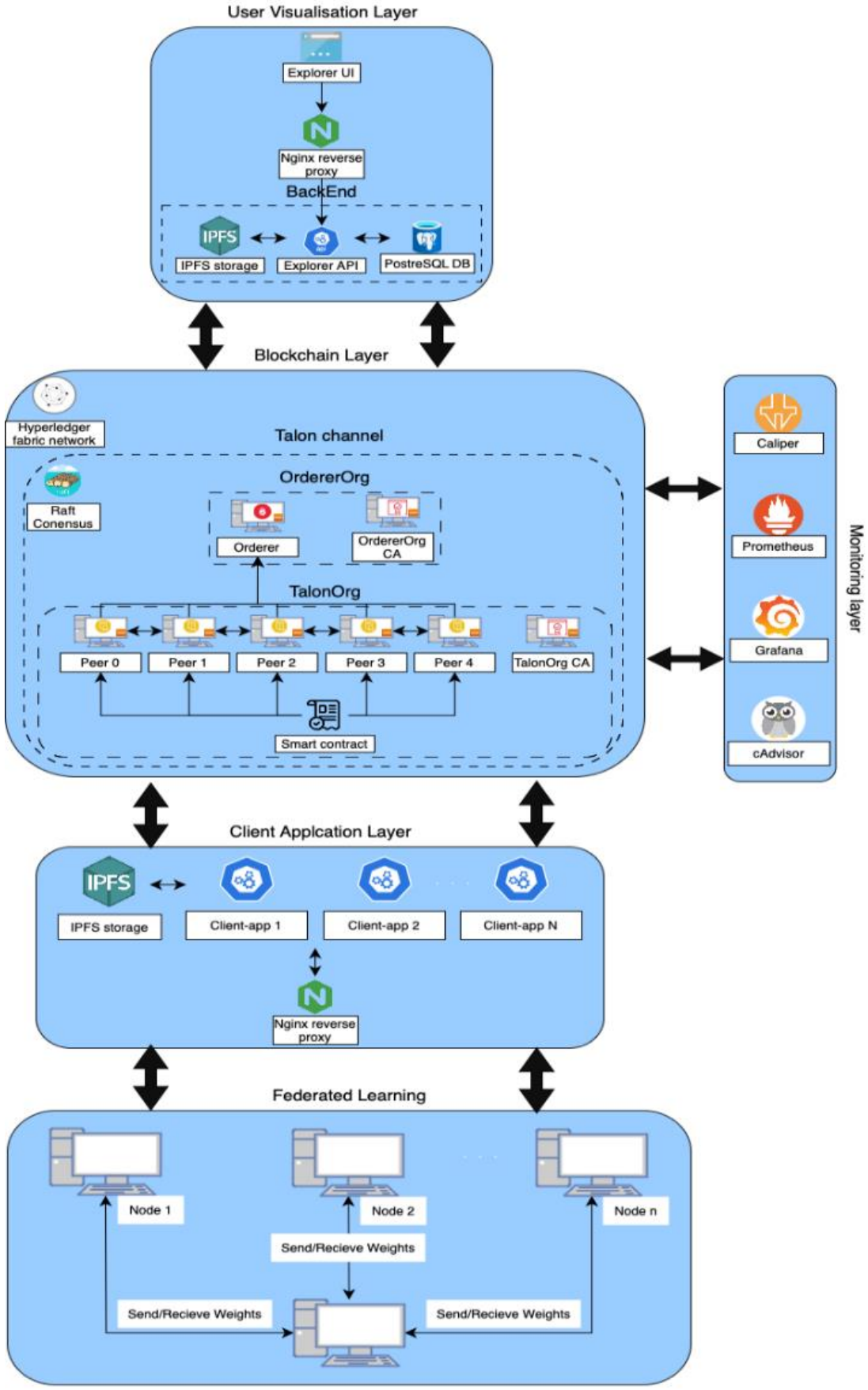

Figure 9: TALON's Blockchain Architecture

The front-end provides a holistic and user-friendly overview of the most important blockchain elements, including blocks, transactions, and the status of network peers. It displays this information in a clear and visually appealing manner, enhancing user experience. Moreover, the API detects dynamically newly added or removed peers, keeping the system with an exact view and always up to date regarding the status of the network

The second layer of TALON blockchain architecture is the blockchain network itself, built on IBM's open-source Hyperledger Fabric[5] framework. This network has of two distinct organisations: the orderer organisation and the TALON organisation. The orderer organisation comprises a single orderer node and a dedicated Certificate Authority (CA). Meanwhile, the TALON organisation includes five peers and its own CA. The existence of different CA's is mandatory for issuing and managing different certificates on each organisation. Communication between these two organisations is facilitated through a dedicated channel named "talon." Within this channel, the five peers of the Talon organisation have deployed a smart contract, enabling them to interact with the blockchain ledger and execute business logic. The network employs the Raft Crash Fault Tolerance[6] (CFT) consensus algorithm to achieve agreement on the transactions committed to the ledger, ensuring consistency and reliability across the distributed network.

The third layer is the Client Application Layer, which enables communication between the edge nodes of the FL layer and the blockchain layer. This layer plays a crucial role in managing data flow between these components. The edge nodes send data to the application's endpoints, which are responsible for two key tasks: uploading the data to the InterPlanetary File System[7] (IPFS) for decentralised storage and executing a transaction within the blockchain's smart contract. The developed solution currently employs a single client application used by all edge nodes, but it is designed to scale horizontally to support multiple client applications if required. This design provides both flexibility and scalability, ensuring that the system can accommodate increasing demand or additional nodes as the network evolves.

The fourth layer is the Monitoring Layer, which is composed of four open-source tools designed to monitor and ensure the overall performance and stability of the network. These tools include:

- Caliper[8] – Used for benchmarking the network, it measures key performance metrics such as throughput and latency.
- Prometheus[9] – A robust monitoring system that gathers and stores time-series data, providing real-time insights into system performance.
- Grafana[10] – An analytics platform that visualises data collected by Prometheus, offering dashboards and alerts to track network health and performance metrics.
- CAdvisor[11] – A daemon that collects, aggregates, and processes information about running containers, enabling monitoring of container performance and resource usage.

Together, these tools provide a comprehensive monitoring solution, ensuring that the blockchain network remains efficient, stable, and reliable by delivering critical performance metrics in real time.

The TALON blockchain toolkit's architecture is designed to meet project's objectives by ensuring scalability, security, and efficiency in decentralised data management for federated learning. The

[5] *https://hyperledger-fabric.readthedocs.io/en/release-2.5*
[6] *https://raft.github.io*
[7] *https://ipfs.tech*
[8] *https://hyperledger-caliper.github.io/caliper*
[9] *https://prometheus.io*
[10] *https://grafana.com*
[11] *https://prometheus.io/docs/guides/cadvisor*

setup with five peers in the “talon” organization provides an optimal balance of resilience and performance, supporting high throughput while maintaining consistency under the Raft CFT consensus model.

By using Hyperledger Fabric for secure, permissioned transactions and IPFS for decentralised storage of big data volume, TALON achieves a lightweight, high-performance blockchain that can handle substantial data volumes without overloading the network. The client application layer and Explorer API facilitate seamless data flow and monitoring, while the monitoring layer—with tools like Caliper, Prometheus, and Grafana—enables real-time insights, as well as performance optimisation.

4.2 Layer Elaboration & Interconnection

In this sub section, a more detailed and technical overview of the blockchain toolkit architecture is shown. Layer by layer all subcomponents, modules and technologies are shortly elaborated to enable the reader grasp the concept with more technical accuracy.

4.2.1 Visualisation Layer

The first layer of Talon's architecture is composed of four key components: an IPFS client, an API, a PostgreSQL database, and a React-based front-end which is built and served through a nginx[12] reverse proxy.

API: The API connects to the blockchain network through two key SDKs: the Fabric-Gateway Node SDK and the Fabric-Network Node SDK. The Fabric-Gateway SDK is responsible for fetching blocks from the blockchain on a specified channel, while the Fabric-Network SDK enables the use of the discovery service to detect active peers dynamically. The blocks and discovered peers are indexed and stored in the PostgreSQL database.

Front-end: The React-based front-end interacts with the API to retrieve data from the database regarding transactions, blocks, and peer statuses. This data is displayed in various data tables, providing users with a clear and organised visualisation of the network’s state. The front page is statically built and served using nginx, a high-performance web server and reverse proxy that ensures efficient and secure content delivery to users.

IPFS Integration: Since the transaction payloads, including AI models, are stored in IPFS, the API leverages the Pinata[13] SDK—a tool that streamlines data management on IPFS by allowing easy upload, retrieval, and pinning of files on the network. Through Pinata, the required data can be fetched from IPFS and made accessible for visualsation and download, enabling efficient processing of large datasets without overloading the blockchain.

4.2.2 Blockchain Layer

Making up to the second layer of TALON’s blockchain, it comprises a set of different interconnected components which are also crucial for maintaining the integrity and functionality of the blockchain system. These include a smart contract, peer nodes, ordering nodes, and certificate authority nodes- all integrated together with the consensus mechanism.

Smart Contract (Chaincode): At the core of this layer is the smart contract, which defines the business logic for the network and is written using goLang[14]. It is deployed on peer nodes and executed

[12] *https://nginx.org/en*
[13] *https://docs.pinata.cloud/sdk/getting-started*
[14] *https://go.dev*

whenever a transaction is invoked. This ensures that all transactions are confirmed to follow predefined rules.

Peer Nodes: These maintain a copy of the distributed ledger and execute the smart contract. Peer nodes validate transactions before they are committed to the ledger. Each peer communicates with the blockchain network and uses the discovery service to locate other peers and orderers in the network, ensuring dynamic adaptability.

Ordering Nodes: The ordering nodes are responsible for establishing the order of transactions within the network. They collect transactions from peers, organise them into blocks, and deliver these blocks back to the peers for validation. This is where the Raft consensus mechanism[15] comes into play, ensuring that all peers agree on the order of transactions, thus maintaining consistency across the network.

Certificate Authority (CA) Nodes: Identity and security is managed with these nodes within the network. Each participant and node must be authenticated and authorised using certificates issued by the CAs. This guarantees that only trusted parties are allowed to participate in the permissioned network activities.

Consensus Mechanism (Raft CFT): The Raft CFT consensus algorithm is used to achieve agreement on the order and validity of transactions across the network. This ensures that even in the event of node failures, the network remains operational and secure.

The components are tightly integrated, with peer nodes communicating with ordering nodes to receive ordered transaction blocks. The CA node authenticates and authorises participants, while the Raft consensus algorithm ensures that the ordering of transactions is agreed upon by all participants in the network. This interconnected framework creates a secure, scalable, and dependable blockchain infrastructure, with each component working together to uphold the network's integrity and performance.

4.2.3 Client Application Layer

The third layer of TALON's blockchain architecture is responsible for integrating the client application (dApp) with both the blockchain network and IPFS (InterPlanetary File System), ensuring seamless data handling and transaction execution. At its core, the client application uses the Hyperledger Fabric Gateway SDK to submit transactions to the blockchain while uploading data to IPFS. Edge nodes connected to the client app generate and send data to the application's endpoints. Upon receiving the data, the client app processes it through two key operations:

IPFS Storage and Pinata Indexing: The client application uploads the received data to IPFS, a decentralized storage solution. Once stored in IPFS, the data is indexed via Pinata, a service that simplifies interaction with IPFS. Pinata generates a unique hash for each data file stored, providing a decentralised reference.

Blockchain Transaction: The hash generated by IPFS is then recorded as an asset on the blockchain network. This is done by using the Gateway SDK to execute a transaction, where the IPFS hash is stored as part of the blockchain's immutable ledger. This process ensures that while the actual data resides on IPFS, its reference and integrity are secured on the blockchain.

The flow between these components is tightly connected. The edge nodes continuously send data to the client app, which acts as a bridge between the IPFS storage and the blockchain. The dApp ensures that data is securely stored in IPFS while the hash, acting as a unique identifier, is submitted

[15] *https://raft.github.io*

to the blockchain network via the Gateway SDK. This interconnection allows for decentralised data storage while ensuring the integrity and traceability of data through the blockchain.

4.2.4 Monitoring Layer

The fourth layer of Talon's architecture is the Monitoring and Performance Benchmarking Layer, which provides detailed insights into the system's performance and resource utilisation. This layer employs four essential open-source tools:

Hyperledger Caliper: Caliper is used to benchmark the performance of the Hyperledger Fabric network. It simulates real-world workloads and submits transactions to the network in a controlled manner, measuring various performance metrics such as throughput, latency and transaction success rate. Caliper connects to the Fabric network through its clients, submitting transactions either individually or in batches. It monitors how quickly and accurately the network processes these transactions. By configuring different workloads (varying transaction volumes and complexities), Caliper can evaluate the network's performance under diverse conditions, identifying potential bottlenecks or weaknesses in scalability and transaction handling. The results from these benchmarks provide crucial insights into the efficiency of the network's smart contracts, consensus mechanism (Raft), and overall transaction throughput. This helps administrators easily fine-tune the network for optimal performance.

Prometheus: Prometheus monitors real-time performance metrics by connecting to the network components using a pull-based model. It extracts various key metrics, including:

- CPU usage from peer nodes via cAdvisor.
- Memory usage and other system resource metrics from the nodes directly.
- Blockchain-specific metrics such as ledger height (the current block count) and consensus healthdirectly from the peer nodes and ordering services.
- Prometheus aggregates these metrics and exposes them for visualisation and alerting.

Grafana: Integrated with Prometheus, Grafana offers real-time visualisation of the collected data through customisable dashboards. It visualises key metrics such as CPU usage, memory consumption, peer and orderer status, ledger height, and network latency, providing system administrators with a clear overview of the network's health and performance.

cAdvisor: cAdvisor monitors containerised applications and provides detailed insights into resource usage, including CPU, memory, network, and storage. Prometheus collects this container-level data from cAdvisor to ensure efficient resource allocation and monitoring of the system's running containers.

5 Implementation of TALON's Blockchain toolkit

The following sections examine two core elements of the TALON blockchain: the development specifications of its technology stack and the communication interfaces that facilitate user interaction and system operations. Section 5.1 provides a detailed overview of the technologies that constitute the blockchain implementation, highlighting how each technology contributes to the overall functionality of the infrastructure. Section 5.2 then focuses on the communication interfaces specifically detailing the user interface (UI) and the Swagger documentation, both of which are essential for ensuring effective interaction between users and the blockchain toolkit. Together, these subsections present a thorough understanding of TALON's inner functionality, underscoring its relevance and application in federated learning environments.

5.1 Development and Specifications

5.1.1 Technology Stack

This section outlines the technology stack used to develop and implement the TALON blockchain toolkit. Each component has been selected to support the toolkit's goals of secure data management, scalability, and efficient processing in a blockchain-based infrastructure. Table 2 provides an overview of the core technologies, frameworks, and tools that form the foundation of the TALON solution, enabling decentralised storage, network monitoring, and seamless interaction between federated learning nodes and blockchain network.

Table 2: TALON's Blockchain Toolkit Technology Stack

Technology	Library/Tools
Blockchain	Hyperledger fabric v2.5.5 Golang v1.22 for smart contract
Explorer Api	Express.js[16] v4.18.2 Fabric Gateway node SDK v1.4.0 Fabric network node SDK v2.2.0 PostgreSQL DB
Client App	Express.js v4.18.2 Fabric Gateway node SDK v1.4.0 Pinata client SDK v2.1.0
Blockchain UI	React v18.2.0 Material UI[17] v5.15 DigitalOcean[18] PaaS (Platform as a service) for deployment
Reverse proxy	Nginx

[16] *https://expressjs.com*
[17] *https://mui.com*
[18] *https://www.digitalocean.com*

Docker	Docker[19] containers
Certifications	Certbot[20] self-signed certificates

The TALON solution integrates multiple technologies to create a secure and scalable infrastructure for managing blockchain-based transactions and decentralised storage. The blockchain network is built using the Hyperledger Fabric framework, establishing a permissioned peer-to-peer network. Access to this network is facilitated through Node.js SDKs—specifically, the Hyperledger Gateway SDK and Fabric Network SDK—which enable secure interaction with the blockchain's peers.

These SDKs are integrated in an Express.js-based REST API, which functions as the backend for managing, processing and indexing blockchain data, such as blocks, transactions and peer statuses. The API retrieves blockchain information in protobuf[21] format – a binary serialisation format used by Fabric for efficiency. Data is then parsed and converted into a human-readable format before being stored in a PostgreSQL database for efficient indexing and querying. The front-end has been developed using React and designed with Material-UI, which is a react component library, fetches this data from the API, offering users a clear and interactive interface to visualise the blockchain activity and network status.

Additionally, a client application (i.e. the dApp) has been developed to facilitate interactions between FL edge nodes and the blockchain. The client app uses the same backend technologies as the blockchain explorer (i.e. the UI), allowing FL edge nodes to upload AI/ML model weights to IPFS for decentralised storage. The application then generates a Content Identifier (CID) or hash for the uploaded data, which is subsequently stored on the blockchain through a transaction managed by the relevant smart contract. This ensures data integrity and traceability, as the hash becomes an immutable asset on the blockchain.

All components are fully dockerised to ensure consistency, scalability, and ease of deployment. The entire system is hosted in a virtual environment and accessed through an NGINX reverse proxy, which routes API requests securely to the respective services.

5.1.2 Code Repository of TALON's Blockchain

From the start of the TALON project, we have used a GitHub repository to manage version control for all the different components. Specifically, for the TALON-Fabric the code repository is available below:

- Repo URL: https://github.com/Sidroco-Holdings-Ltd/Talon-Fabric
- Explorer API and URL: https://github.com/Sidroco-Holdings-Ltd/Talon-Fabric/tree/main-ipfs/talon_dashboard
- Blockchain URL: https://github.com/Sidroco-Holdings-Ltd/Talon-Fabric/tree/main-ipfs/talon_network
- Client APP URL: https://github.com/Sidroco-Holdings-Ltd/Talon-Fabric/tree/main-ipfs/talon_client_app
- Chaincode URL: https://github.com/Sidroco-Holdings-Ltd/Talon-Fabric/tree/main-ipfs/talon_chaincode_go

[19] *https://www.docker.com*
[20] *https://certbot.eff.org*
[21] *https://protobuf.dev/overview*

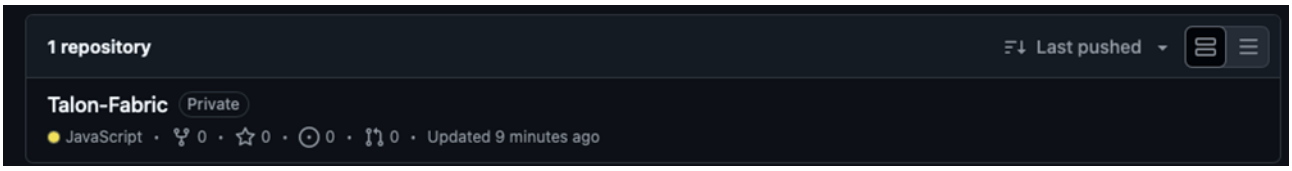

Figure 10: TALON's Repository

The image above shows TALON's repository per se, with access to the source code (Figure 11) being granted upon request.

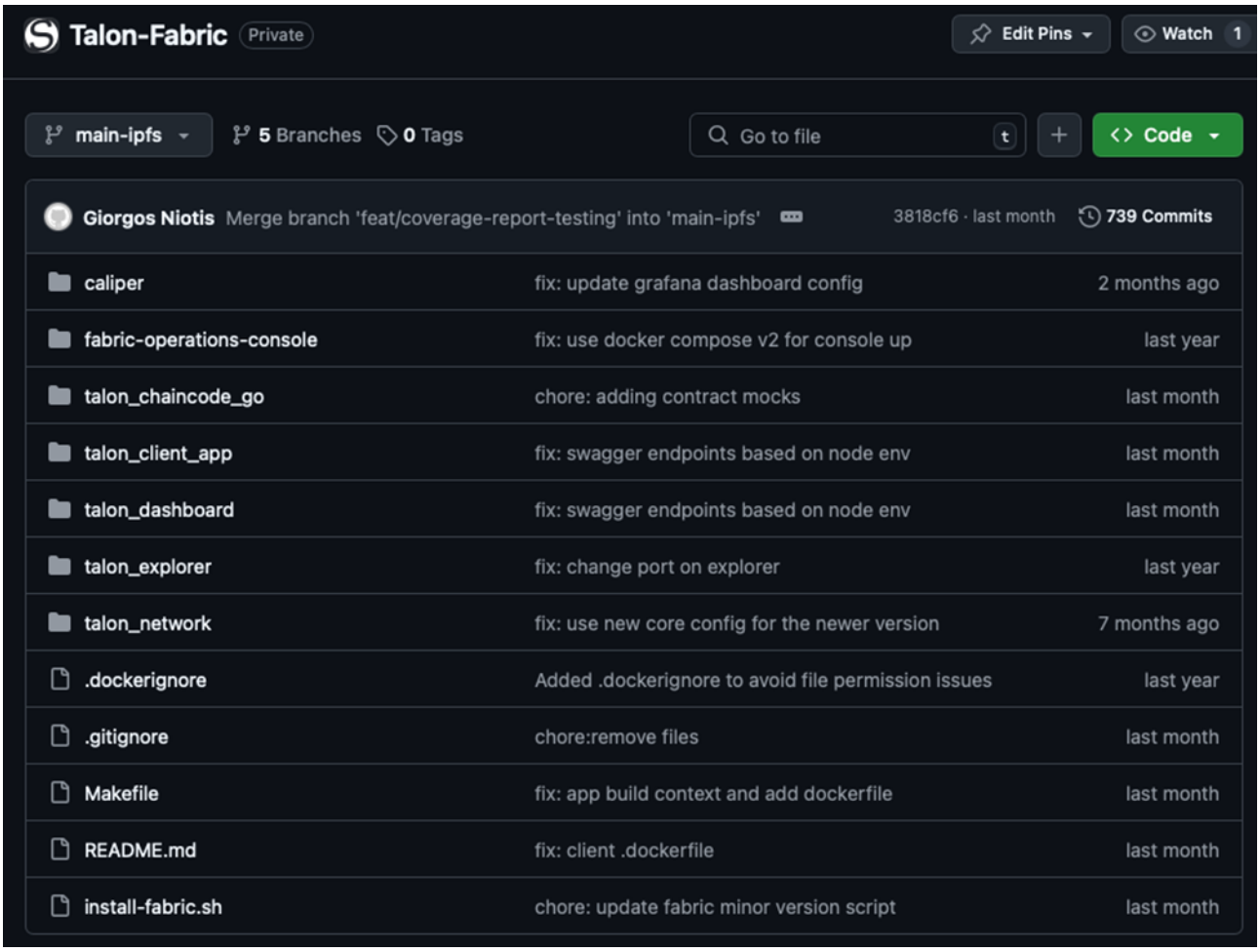

Figure 11: TALON's Source Code Files

Digital Ocean cloud registry has also been employed for storing our explorer images and deploying it using Digital Ocean's Platform as a Service (PaaS) infrastructure.

Figure 12: Explorer Images Registry

Regarding the rest of the components, the blockchain network and the APIs have been deployed on a Digital Ocean's droplet, which is a virtual machine (VM).

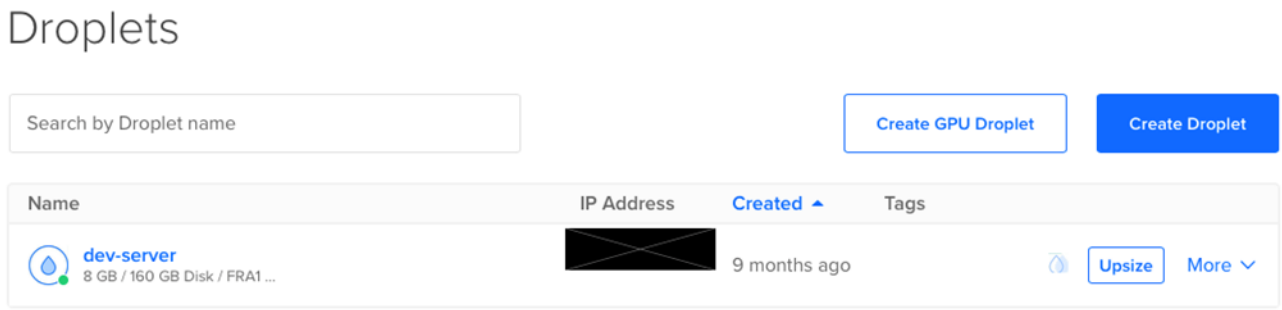

Figure 13: Blockchain and APIs Droplet

5.2 Communication Interfaces

In this section the communication interfaces related to the blockchain toolkit are explored. On the one hand the UI is showcased with its functionalities and utilities thoroughly elaborated, while on the other hand, the swagger is depicted along with its functionalities. These two modules are explored into to two distinct sub-sections.

The first interface regards to the "**User Interface/experience (UI/UX)**"

TALON's blockchain UI is explicitly developed to meet the specific requirements of TALON applications. As previously mentioned, this solution has been developed to accommodate the validation, tracking and storage of AI/ML models in a decentralised way. This is realised through the federated transmission of weights from the FL clients to FL server. To that end, specific data (i.e. a

JSON[22]) volumes are managed with a relevant frequency of transmission. Existing UI solutions such as the Fabric explorer, Etherscan[23] (Etherium explorer), Solscan[24] (Solana explorer) etc, cannot visually accommodate the intricate specifications of TALON's blockchain functionality. This is mainly due to the frequency of the AI operations and the distinct dates and times these operations are performed. As such, scarce ML tasks in non-scheduled timelines are difficult to track by the end user and explore the transactions and information.

The developed TALON Explorer provides a cutting-edge comprehensive platform for monitoring and visualising key aspects of the blockchain network with regard to the information of TALON. The interface is designed to display real-time blockchain data, including blocks, transactions, and peer status. The UI is enhanced with an additional functionality of "Start Date" and "End Date" that allows the end user to query through specific time windows, enabling the display of the only necessary and relevant transactions of the ledger.

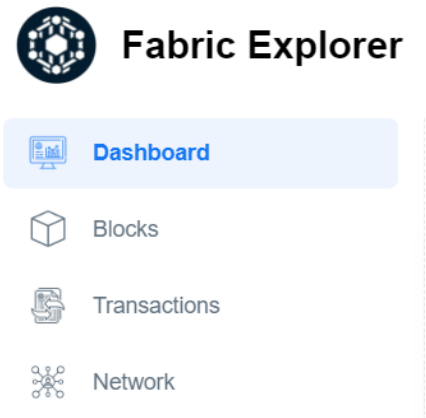

Figure 14: : TALON's Blockchain UI ribbon of panes

As shown in Figure 14, TALON's Fabric-based explorer is comprised of 4 distinct panes, namely the:

- Dashboard
- Blocks
- Transactions
- Network

These panes come at the top left of the window with a ribbon format and serve the end user with specific information live and on demand. From these panes the user can have a holistic overview of the blockchain, check for specific information at block level, access data from the transactions and grasp the perspective of the hole distributed network. More details and in-depth analysis of the functionality will be showcased on the following sub-sections.

5.2.1 Dashboard Pane

The dashboard pane (Figure 15) presents a high-level summary of the blockchain's activity, with key metrics such as:

[22] *https://www.json.org/json-en.html*
[23] *https://etherscan.io*
[24] *https://solscan.io*

- Total Blocks: The current number of blocks in the blockchain (1.32k).
- Total Transactions: The total number of transactions processed on the network (1.32k).
- Nodes: The number of active nodes (5), which includes both peer and orderer nodes.

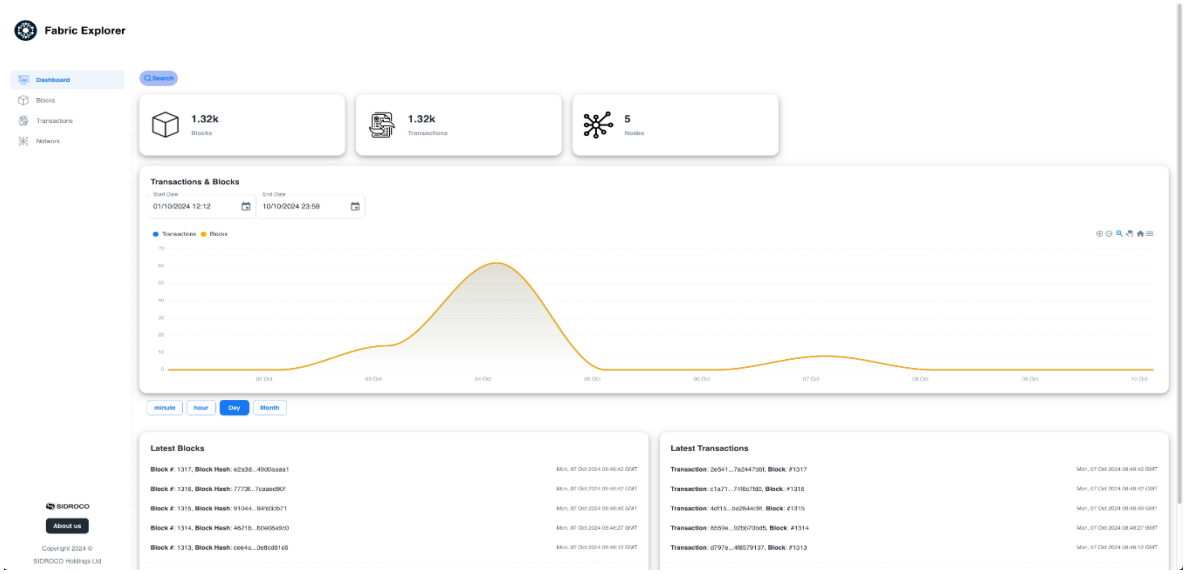

Figure 15: TALON's Dashboard Pane

A time-based graph (Figure 16) shows the transaction and block generation trends for specific time windows, allowing users to visualise network activity over time. This overview helps administrators and users to monitor the network's throughput and operational status briefly. Users also have the functionality to automatically switch between “minute”, “hour”, “day” and “month” coverage to depict either specific transactions over small periods of time or to grasp the greater concept of the network movement over larger time windows. As shown in the figure, the user can hover over the curves of the graph and get information about the date and timestamp of the net activity, as well as the number of blocks and transactions. Additionally, the “Start Date” and “End Date” functionality when combined with the “minute”, “hour”, “day” and “month” functionality allows the users to target specific frames of activity and extract intuitive feedback regarding the federated ML tasks performed and validated over time.

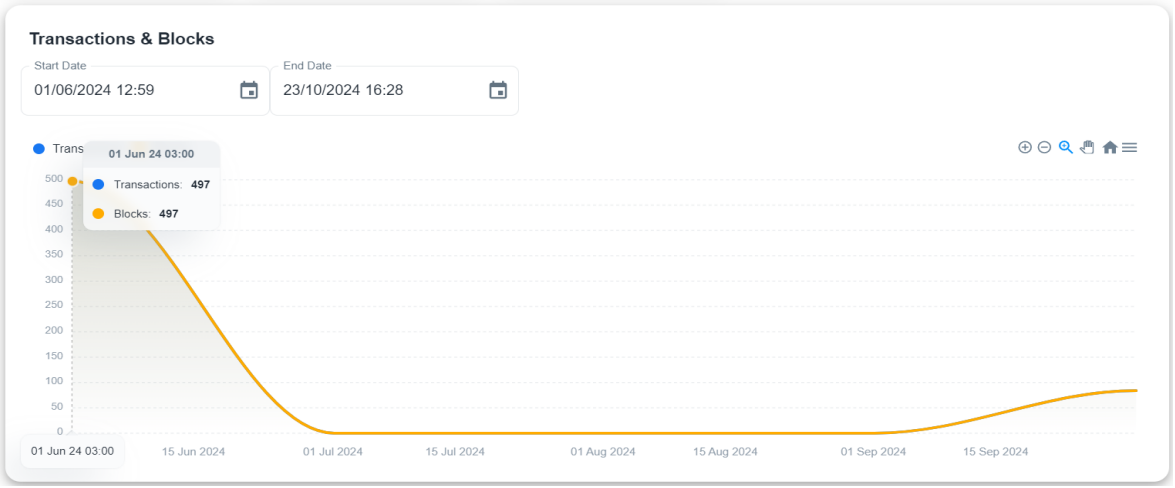

Figure 16: Time-based Graph Exploration of Transactions & Blocks

Moreover, users have the latest transactions and blocks (Figure 17) visualised at the bottom center of the window. In this short view the user can check the enumerated blocks (incremental numbering), the block's or transaction's hash and the corresponding timestamp. The inverse stands with the transaction minitablet, which depicts briefly the transaction hash, the corresponding block number that contains it and the timestamp. The "View all blocks" and "View all transactions" is clickable and moves the user to the corresponding "Blocks" or "Transactions" pane to see the full list and more information. This enables fast and efficient network activity monitoring while simultaneously checking the transaction-blocks relevance.

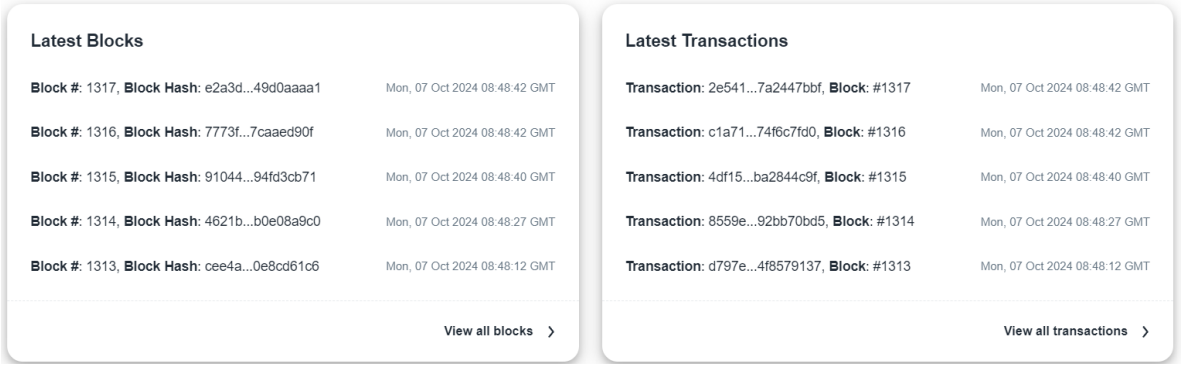

Figure 17: Latest Transactions Blocks

The toolkit is also boosted with a search engine (Figure 18) that enables users to search for specific information querying transaction hashes or block numbers.

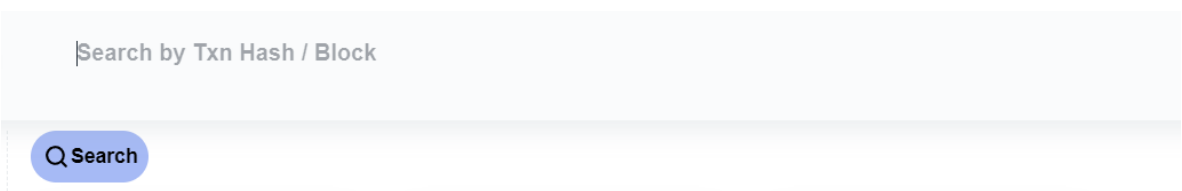

Figure 18: Search Engine functionality

5.2.2 Blocks Pane

The "Blocks" section of the UI provides a detailed list of all blocks added to the network. The interface (Figure 19) offers:

- **Block Number:** Sequential identification of blocks.
- **Transaction Count**: Number of transactions within each block.
- **Block Hash:** A unique hash that identifies each block.
- **Timestamp:** The precise time each block was committed to the blockchain.

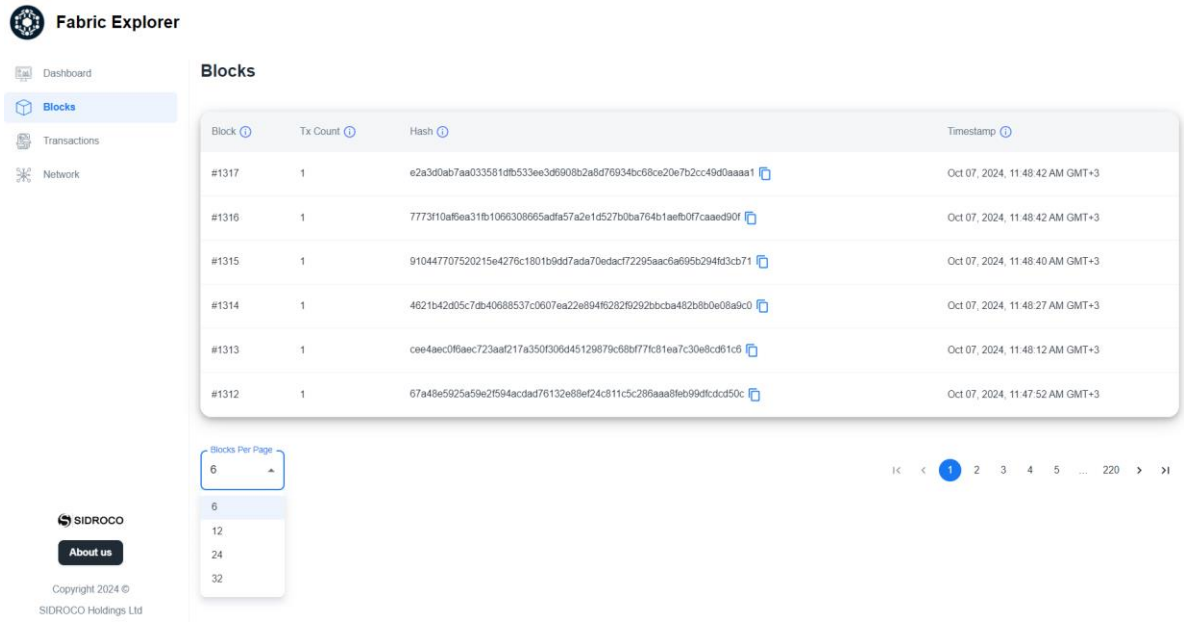

Figure 19: TALON's Blocks Pane

As mentioned previously, the user can access this pane either by clicking the “Blocks” pane or the “View all blocks from the main dashboard. Moreover, the user can hover over the “i” icons right next to each column name (block, tx count, etc) and check info about the information of each column. At the bottom left users can change the number of displayed blocks per page according to their preferences and at the bottom right switch page to access older transactions. The ability to view and search block-level data offers transparency and traceability, which are critical for TALON requirements. This ensures accountability and efficiency, as users can trace every transaction back to its respective block with ease.

5.2.3 Transactions Pane

In the Transactions tab (Figure 20), detailed information about individual transactions is displayed. More specifically:

- Transaction ID: A unique identifier for each transaction.
- Block Number: The block in which the transaction is recorded.
- Transaction Type: Specifies the type of transaction, such as "Endorser Transaction."
- Timestamp: The time when the transaction was committed.

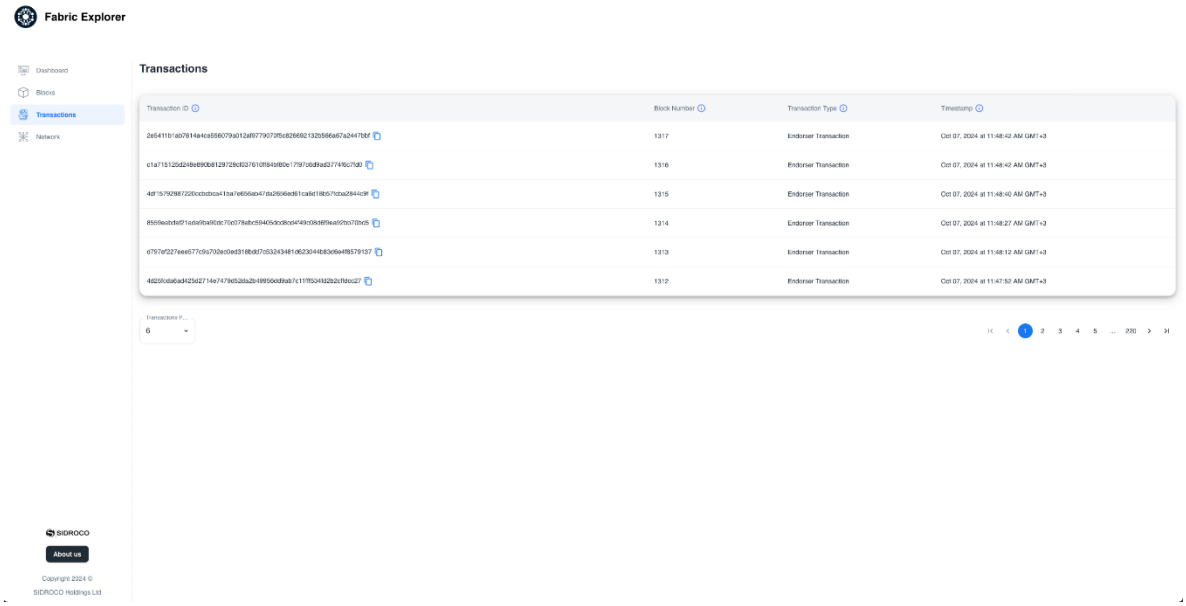

Figure 20: TALON's Transactions Pane

The same principles as the" Blocks" pane apply here regarding the structure of the page. The difference relies on the context which in this case is transactions instead of blocks. The user gets information regarding transaction hash, the block in which this transaction is contained and the transaction type. The transaction type can be an endorsement transaction, which is a transaction type executed for contract interaction , and configuration transaction, which is a transaction that configures the channel (e.g. adding peers, installing chaincodes, etc.).This level of detail is essential for organisations needing to audit specific transactions, track performance, and resolve disputes. The ability to fetch transaction details easily helps maintain transparency and traceability.

5.2.4 Network Peers Pane

The Network peers tab (Figure 21) displays the current state of the network's peer and orderer nodes, including:

- Peer Name: Identifies the peers and orderer nodes in the network.
- Requested URL: The network addresses (URL) of each node.
- Peer Type: Distinguishes between peers and orderer nodes.
- MSP ID: The Membership Service Provider ID that links nodes to their respective organizations.
- Active Status: Visual confirmation of whether each node is active.

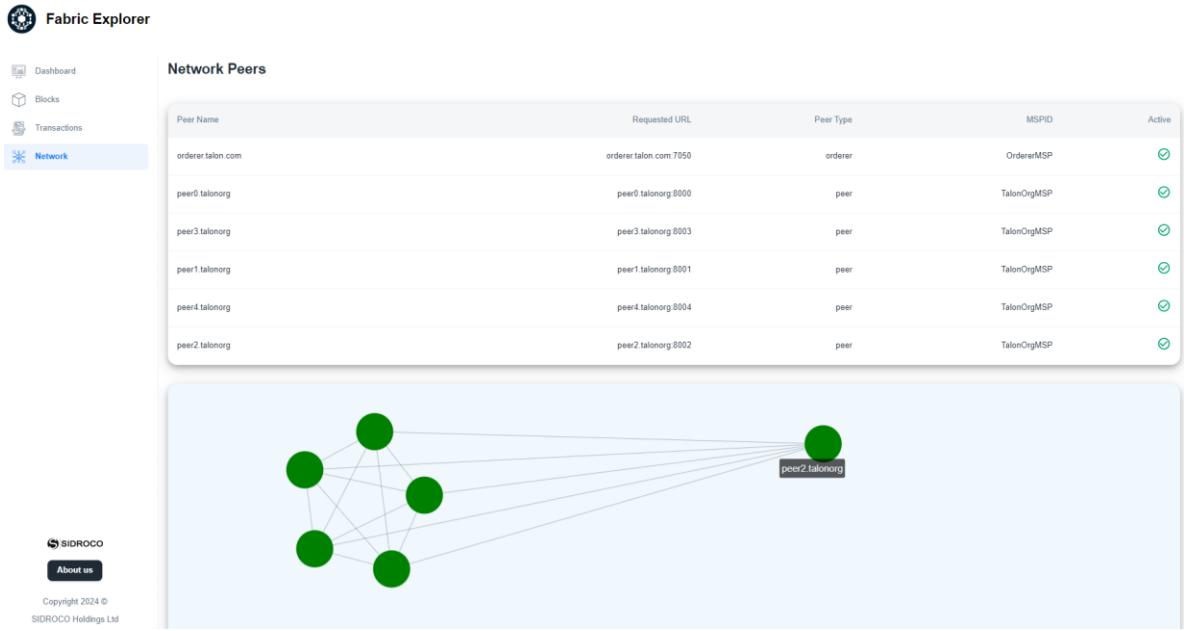

Figure 21: TALON's Network Peers Pane

This view provides real-time visibility into the network's topology and operational status, ensuring that administrators can monitor the health of both peer and orderer nodes, which are critical to maintaining a fault-tolerant network. The user can easily identify which peers are active and able to execute transactions for a specific organisation by providing the names and the endpoints for each peer and make it possible to identify the root cause of possible rejected transactions

The UI enables a more detailed block and transaction information section, when clicking on a specific block or transaction, with detailed information popping up, providing further insight into the blockchain's inner workings:

- Transaction Details: Include the transaction ID, the associated block, transaction metadata (e.g., FL round, node ID), the data's CID from IPFS, and the timestamp (Figure 22).
- Block Details: Include block hash, previous block hash, transaction IDs within the block, and the timestamp (Figure 23).

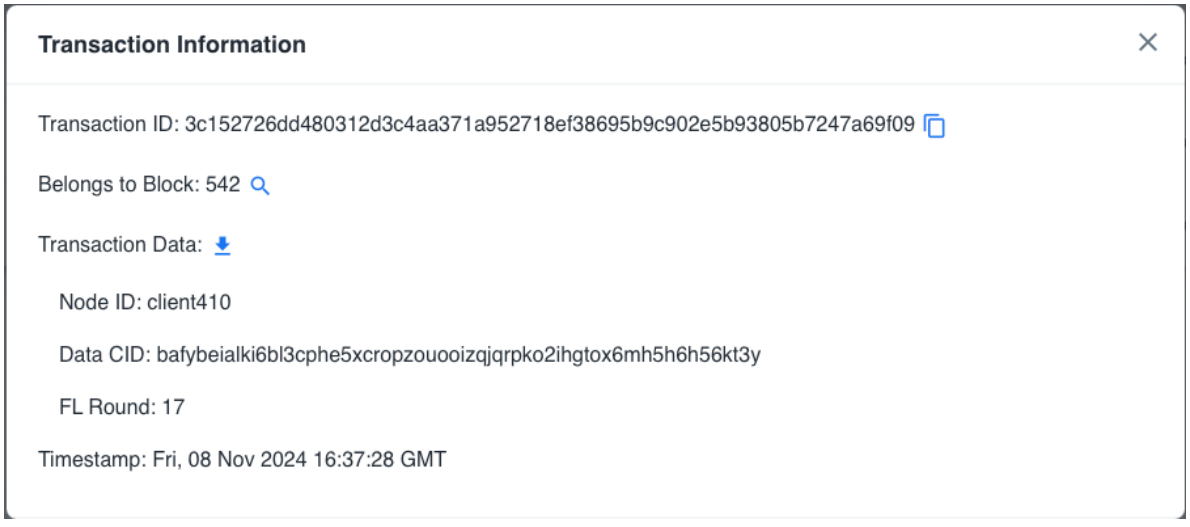

Figure 22: Transactions Information

Figure 23: Block Information

The integration with IPFS highlights the system's capacity to store transaction payloads off-chain, ensuring scalable and decentralised storage. By storing the IPFS CID (Content Identifier) on the blockchain, the Talon Explorer ensures that the data can be retrieved securely while maintaining an immutable record of the file's existence and integrity.

The development of TALON's blockchain visual interface has specific key functionalities and benefits. Those benefits reflect the value proposition of TALON's implementation towards achieving the projects objectives, as well as promoting the robustness of the toolkit among the blockchain community. The functional utilities with the added values are enumerated below:

1. Real-Time Monitoring: The TALON Explorer provides real-time insights regarding the blockchain activities, including block and transaction metrics, allowing administrators to monitor network activity (federated AI/ML operations) and identify any potential issues as they arise.
2. Transparency & Traceability: With detailed views on blocks, transactions, and peer nodes, the Explorer promotes trust in the blockchain network. Users can track any transaction or block back to its source, providing verifiable proof of transactions and network activity.
3. Data Offloading to IPFS: The integration with IPFS allows for efficient storage of large payloads off-chain, while ensuring that the blockchain only stores essential metadata (such as the CID). This approach reduces the load on the blockchain rendering it lightweight, while simultaneously maintaining decentralised, tamper-proof and efficient data storage.
4. Node Management: The Network pane provides a clear picture of the active peers and orderer nodes, allowing administrators to track the health and connectivity status of the nodes in real-time, ensuring the network remains robust and operational.
5. User Interface: The intuitive UI built with React.js and Material-UI ensures that users can easily navigate between different sections, access detailed data, and analyse blockchain performance with minimal effort. The responsive design enhances user experience, making the tool accessible on multiple devices.
6. Scalability & Security: The system is designed to scale as the network grows, with the ability to handle large numbers of blocks and transactions while maintaining performance. By leveraging Docker for containerisation, the platform can be easily deployed and managed across various environments.

5.2.5 API Endpoints

In addition to the insightful UI for monitoring and visualising the blockchain network, the TALON blockchain toolkit also includes an Explorer API and a Client Application, which provide critical

functionality for interacting with the blockchain. These components expose a set of endpoints that facilitate interactions such as querying blockchain data, managing assets, and interfacing with external systems like IPFS for decentralised data storage.

The Explorer API (which is demonstrated at Figure 24), serves as the backbone for fetching real-time blockchain data, such as blocks, transactions, and network status. It allows users and other services to programmatically interact with the blockchain network through the network's peers, enabling efficient monitoring and data extraction.

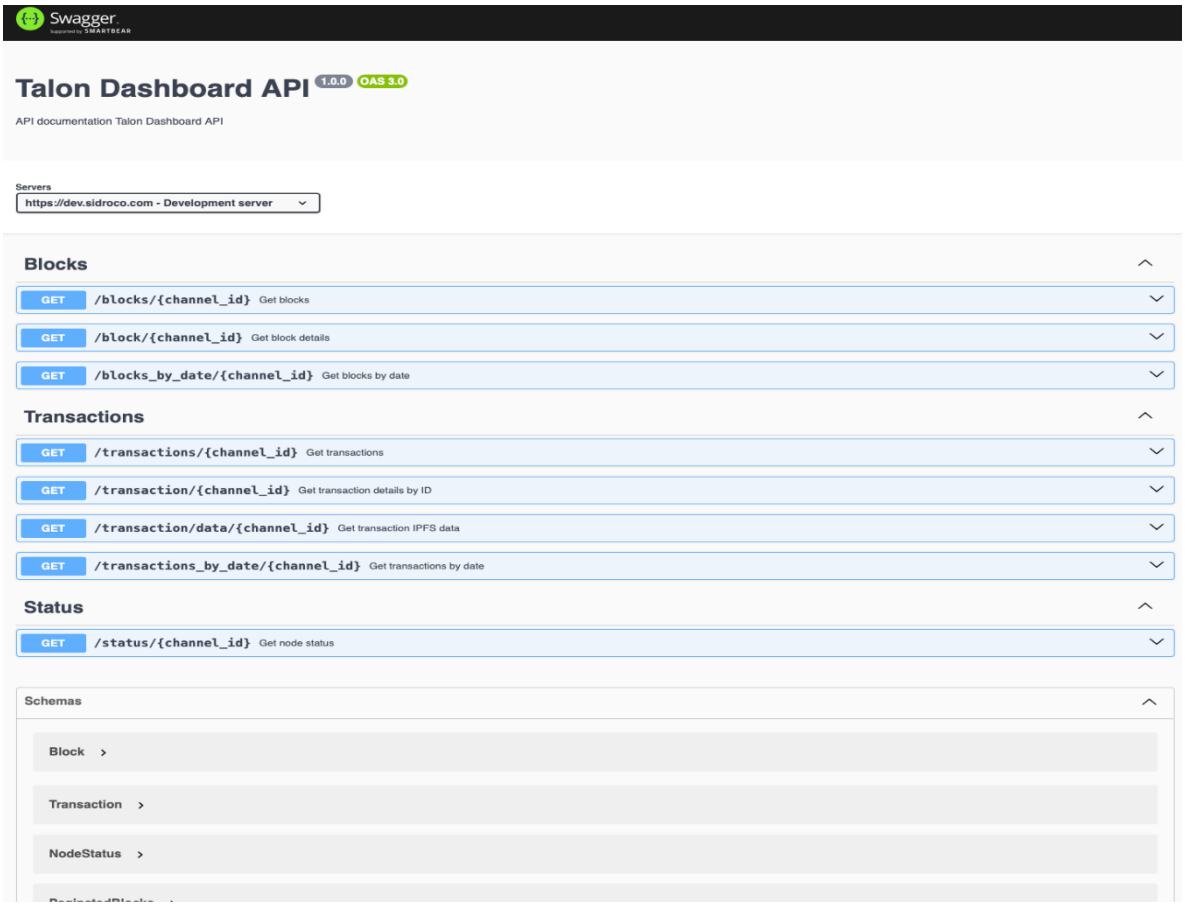

Figure 24: Dashboard API Swagger

Detailed explanations of the 'GET' methods from the UI are provided below:

1. **/blocks/:channel_id** (GET)
    - **Description**: Fetches a paginated list of blocks from the blockchain, including their associated transactions. This endpoint allows users to track block creation and the transactions contained within each block.
    - **Controller**: getBlocks
2. **/block/:channel_id** (GET)
    - **Description**: Retrieves detailed information about a specific block based on its block number or block hash. This endpoint ensures users can inspect specific blocks and validate their content.
    - **Controller**: getBlock
3. **/transactions/:channel_id** (GET)
    - **Description**: Fetches a paginated list of transactions from the blockchain for a given channel. This endpoint is essential for tracking transaction activity in the network.

- **Controller**: getTransactions

4. **/transaction/:channel_id** (GET)
    - **Description**: Retreives detailed information about a specific transaction, allowing users to audit and trace individual transactions.
    - **Controller**: getTransactionById
5. **/transaction/data/:channel_id** (GET)
    - **Description**: Fetches transaction paylod-related data which are stored off-chain in IPFS. This endpoint retrieves the content identified by the CID (Content Identifier) stored on the blockchain, enabling users to access big volumes of data, such as FL model weights.
    - **Controller**: getTransactionDataById
6. **/blocks_by_date/:channel_id** (GET)
    - **Description**: Retrieves the count of blockhs created during spevific time intervals. Useful for analysing network activity over a certain period of time.
    - **Controller**: getBlocksByDate
7. **/transactions_by_date/:channel_id** (GET)
    - **Description**: Similar to the blocks by date endpoint, this retrieves the count of transactions during specified time intervals, helping to assess periods of high or low network activity.
    - **Controller**: getTransactionsByDate
8. **/status/:channel_id** (GET)
    - **Description**: Fetches the status of the network's nodes, including the ledger height and whether the nodes are active. This is critical for network health monitoring and node management.
    - **Controller**: getNodeStatus

5.2.6 Client Application Endpoints

The Client Application allows external services or Federated Learning (FL) edge nodes to interact with the blockchain. The client app exposes endpoints for managing blockchain assets and interacting with IPFS to store and retrieve decentralized data. The developed blockchain implementation has been successfully integrated with MINDS's FL framework seamlessly. More specifically, FL clients connect to that app sending the following data in JSON format:

- Edge node id
- Training round
- Accuracy
- Weights

An actual example of these JSON files is depicted in Figure 25.

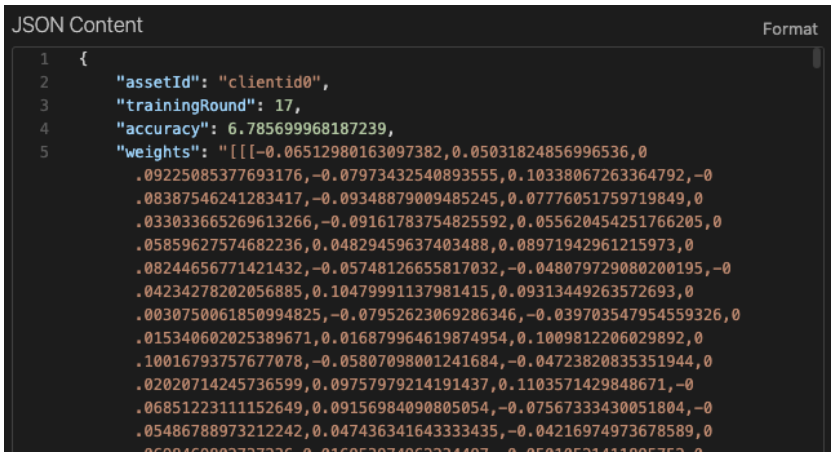

Figure 25 Example JSON output

These data are first uploaded to IPFS, a decentralised storage solution, to ensure secure and distributed file storage. Once uploaded, IPFS generates a unique Content Identifier (CID) or hash, which serves as a permanent reference to the data's location on the network. This CID is then recorded on the blockchain via a transaction, creating an immutable link to the stored data.

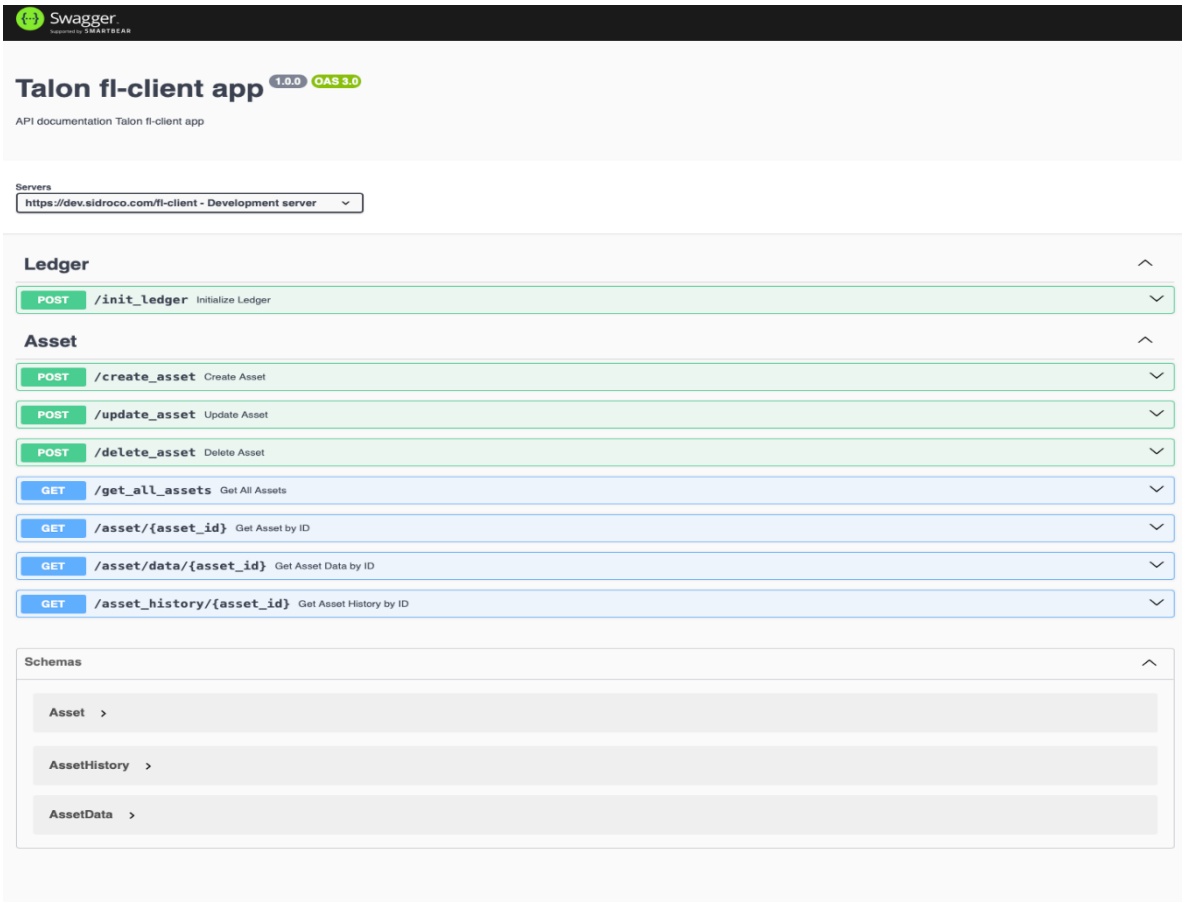

Figure 26: FL Client API Swagger

Detailed explanations of the 'GET' and “POST” methods from the client application swagger (as shown in Figure 26) are provided below:

1. **/init_ledger** (POST)

- **Description**: Initializes the blockchain ledger with a specified number of assets. This is typically used to set up the blockchain's state when a new channel or network is created.
- **Controller**: initLedger

2. **/create_asset** (POST)
   - **Description**: Creates a new asset on the blockchain. This endpoint is used by edge nodes or external systems to register new data assets in the blockchain, such as Federated Learning model weights.
   - **Controller**: createAsset
3. **/update_asset** (POST)
   - **Description**: Updates an existing asset on the blockchain. This involves uploading new data (e.g., updated model weights) to IPFS, storing the resulting CID, and linking it to the existing blockchain asset.
   - **Controller**: updateAsset
4. **/delete_asset** (POST)
   - **Description**: Deletes an asset from the blockchain. This is useful when an asset needs to be removed or invalidated from the network.
   - **Controller**: deleteAsset
5. **/get_all_assets** (GET)
   - **Description**: Retrieves all current assets stored on the blockchain. This is useful for auditing and monitoring the blockchain's state.
   - **Controller**: getAllAssets
6. **/asset/:asset_id** (GET)
   - **Description**: Retrieves detailed information about a specific asset by its asset ID. This allows users to inspect the contents of a particular asset on the blockchain.
   - **Controller**: getAssetByID
7. **/asset/data/:asset_id** (GET)
   - **Description**: Fetches asset data stored in IPFS, identified by the CID stored in the blockchain. This enables users to retrieve large datasets off-chain, improving scalability.
   - **Controller**: getAssetDataByID
8. **/asset_history/:asset_id** (GET)
   - **Description**: Retrieves the full history of changes made to a specific asset, providing a complete audit trail for transparency and accountability.
   - **Controller**: getAssetHistoryByID

5.2.7 Key Features of the API & Client App

The development of TALON's blockchain application interface also comes with specific key functionalities and benefits. Those technical implementations promote scalability, usability and seamless interaction between components. The added value is aggregated and enumerated below:

- Comprehensive Data Access: The Explorer API allows users to access detailed blockchain data such as blocks, transactions, and node status. The Client App provides asset management capabilities, enabling interactions with both the blockchain and IPFS.
- IPFS Integration: Both the Explorer API and Client App employ IPFS for storing large data off-chain, ensuring scalable and decentralised data storage while maintaining blockchain immutability. This renders the solution lightweight without sacrificing storage capacity.
- Auditing and Traceability: Through endpoints like getAssetHistoryByID and getTransactionById, the platform offers complete traceability and auditing of blockchain transactions and assets.

- Efficient Data Retrieval: The paginated responses and time-based queries ensure that users can retrieve large datasets or historical data efficiently, even as the blockchain grows in size.

Smart Contract (Chaincode)

This chaincode is designed for managing simple assets on Hyperledger Fabric, tracking assets by unique identifiers (ID) and customer identifiers (CID). It provides core functionalities to create, retrieve, update, delete, and view the history of each asset. Through these functions, the chaincode enables secure and auditable asset management, allowing participants on the blockchain network to verify an asset's existence, update its details, and review its transaction history. The contract is structured to handle standard ledger interactions, ensuring data consistency and integrity within the distributed ledger, making it suitable for use cases that require transparent and reliable asset tracking. In the following figure, the reader can find the pseudo code of the smart contract and explanations regarding the functions.

```
class SmartContract extends Contract

    # Asset Data Structure
    struct Asset
        ID: String
        CID: String

    # HistoryQueryResult Data Structure
    struct HistoryQueryResult
        Record: Asset
        TxId: String
        Timestamp: Time
        IsDelete: Boolean

    # Initialize Ledger with a Specified Number of Assets
    function InitLedger(numAssets: Integer)
        if numAssets <= 0
            return "Error: Number of assets must be greater than zero"

        for i from 0 to numAssets - 1
            id = "clientid" + i
            asset = Asset(ID: id, CID: "")
            storeAsset(id, asset)

    # Retrieve Asset by ID
    function GetAsset(id: String) -> Asset or Error
        assetData = fetchAsset(id)
        if assetData == null
            return "Error: Asset does not exist"
        return deserialize(assetData) as Asset

    # Delete Asset by ID
    function DeleteAsset(id: String) -> Error
        if !AssetExists(id)
            return "Error: Asset does not exist"
        deleteAsset(id)

    # Check if Asset Exists
    function AssetExists(id: String) -> Boolean
        return fetchAsset(id) != null

    # Retrieve All Assets
    function GetAllAssets() -> List of Asset
        assets = []
        for each assetData in fetchAllAssets()
            asset = deserialize(assetData) as Asset
            append asset to assets
        return assets

    # Create a New Asset with Unique ID
    function CreateAsset(id: String) -> Error
        if AssetExists(id)
            id = generateNewID()
        asset = Asset(ID: id, CID: "")
        storeAsset(id, asset)

    # Update Existing Asset's CID
    function UpdateAsset(id: String, cid: String) -> Error
        if !AssetExists(id)
            return "Error: Asset does not exist"
        asset = Asset(ID: id, CID: cid)
        storeAsset(id, asset)

    # Retrieve Asset History by ID
    function GetAssetHistory(assetID: String) -> List of HistoryQueryResult
        history = []
        for each record in fetchHistory(assetID)
            asset = deserialize(record.value) as Asset if record.value exists else Asset(ID: assetID)
            timestamp = record.timestamp
            historyResult = HistoryQueryResult(TxId: record.txId, Timestamp: timestamp, Record: asset, IsDelete: record.isDelete)
            append historyResult to history
        return history
```

Figure 27: TALON's chaincode pseudo code

- InitLedger
  Initialises the leddger with a specific number of assets. For each asset, it generates a unique ID and creates a default asset with an empty CID. This function is useful for setting up a baseline dataset in the ledger.
- GetAsset
  Retrieves an asset from the ledger by its ID. If the asset doesn't exist, it returns an error. This function allows querying a specific asset's details.
- DeleteAsset
  Deletes an asset from the ledger if it exists. If the asset does not exist, it returns an error. This function supports asset removal from the system.
- AssetExists
  Checks iff an asseit with the specified ID exists in the ledger. It returns true if it is found and false otherwise. This function is often used by other functions to confirm as asset's existence before performing operations on it.
- GetAllAssets
  This function retrives all assets that are currently stored in the ledger. It returns a list of all assets, providing an overview of the current state of all data stored.
- CreateAsset
  Creates a new asset with the specified ID. First it checks if an asset with the given ID already exists. If it does, the function exits with an error, mentioning that he asset already exists. If the asset does not exist, it creates a new asset with an empty CID field and stores it in the ledger.
- UpdateAsset
  Updates the CID of an existing asset with a given ID. If the asset does not exist, it returns an error. This function allows the modification of an asset that is already in the ledger.
- GetAssetHistory
  Retrieves the transaction history of a specific asset, including creation, updates, and deletions. Each historical record includes the transaction ID, timestamp, asset state, and deletion status. This function helps in tracking an asset's lifecycle and changes over time.

5.3 Deployment

TALON' blockchain network (Fabric), a client application (dApp) and the Explorer API (UI), is deployed within a containerised environment using Docker. This architecture is hosted on a DigitalOcean Droplet (a virtual machine running Ubuntu) to ensure scalability, security, and ease of management.

5.3.1 Blockchain Network and Application Deployment

The Hyperledger Fabric network, including the orderer and peer nodes, is deployed within a Docker network on the DigitalOcean Droplet. Each component, such as the peer nodes, certificate authorities, and orderer nodes, operates within individual Docker containers, ensuring isolation and ease of management.

The client application (used by FL edge nodes to upload data to IPFS and submit transactions to the blockchain) and the Explorer API (which interacts with the blockchain network to retrieve data) are also deployed within the same Docker network, ensuring secure internal communication between the containers.

This containerised approach ensures that each component can be managed independently, scaled as needed, and updated without affecting the other services.

5.3.2 NGINX Reverse Proxy for External Access

To make the client application and the Explorer API accessible from outside the DigitalOcean Droplet, an additional Docker container is deployed running NGINX, configured as a reverse proxy. This reverse proxy is responsible for routing incoming traffic from external clients to the appropriate internal services:

- Traffic destined for the Explorer API is directed to its corresponding container.
- Traffic for the client application is similarly routed through the reverse proxy to the client app's container.

By using NGINX, we can effectively manage external requests, ensuring that the client app and Explorer API are securely exposed to the internet while keeping the underlying infrastructure hidden from direct access.

5.3.3 Secure Communication with SSL Certificates

To ensure secure communication between external users and the Talon platform, Certbot is utilised to generate SSL certificates, securing the connection between users and the blockchain dashboard UI. These certificates enable HTTPS encryption, ensuring that data transmitted between clients and the backend services remains confidential and tamper-proof.

Although Certbot can issue certificates signed by a trusted certificate authority, for internal or testing environments, self-signed certificates are also an option. The self-signed certificates ensure encrypted communication even when working in a non-production or private network setup.

5.3.4 Blockchain UI Deployment

TALON's blockchain front-end, developed using React.js, is deployed as a static build application. For hosting, we leverage DigitalOcean's Platform-as-a-Service (PaaS) solution, which is optimised for scalable web applications. By deploying the front-end as a static build, the UI is served efficiently, providing a responsive and seamless user experience. This PaaS deployment ensures that the UI is highly available, automatically scalable, and easy to manage. It decouples the front-end from the blockchain and backend components, allowing for independent updates and maintenance.

5.3.5 Inter-Component Communication

Within the Docker network, the client application, Explorer API, and Hyperledger Fabric network components communicate securely with each other via internal Docker networking. Since all services are isolated within the same Docker network, they can communicate without exposure to external traffic, reducing the risk of unauthorised access.

External users interact with the system through the TALON's blockchain dashboard interface, which communicates with the backend services (client app and Explorer API) via the NGINX reverse proxy. The UI fetches data and provides a real-time view of the blockchain's activity, leveraging secure HTTPS channels.

6 Validation of TALON's Blockchain toolkit

6.1 Overall Validation Approach

For the benchmarking of the implementation, Hyperledger Caliper and Grafana was employed, which are both suitable validation tools for blockchain networks, in order to be sure of conducting performance and robustness testing of the decentralised applications (dApps). The aim is to quantify various performance measures, including transaction throughput, latency, and resource utilisation, to ensure that the dApps are only of appropriate quality.

Hyperledger Caliper is an open-source project to benchmark performance from various blockchain implementations. Benchmarking with Caliper comes with advantages and makes it one of the most used choices for both developers and researchers. First of all, the major benefit using Caliper is the various performance metrics it provides. In particular and the context of our specific needs, transaction throughput, latency, and resource utilisation. Transaction throughput is defined as the number of transactions that can be processed continuously by a blockchain network within a second. This is important to understand the capability of a blockchain network to handle high number of transactions. Latency refers to the time it takes from the point of submitting a transaction up until getting its confirmation. This is an important network metric, especially for applications with strict requirements regarding response time. Caliper also provides flexibility for developers to perform their testing. Modifications can be made in the parameters like block size, transaction rate and network configuration that would simulate different real-world scenarios. For instance, by utilising Caliper, researchers have widely tested permissioned blockchains like Hyperledger Fabric, measuring its performance efficiency under various conditions—like number of nodes and transaction load. The testing exercises are providing lots of insights regarding potential bottlenecks in the scalability of systems and configuration adjustments that could be used to achieve real-world scale performance enhancements [25].

Grafana, when combined with Caliper, raises the benchmarking process to a different level because of its advanced monitoring and visualisation. Grafana is an open-source platform that enables users to create dynamic, interactive dashboards visualizing real-time data on a wide range of system metrics. This is particularly useful in a blockchain network, where performance needs continually to be observed to assure the operation within defined bounds. Grafana shows some key metrics on resource consumption such as CPU and memory, network traffic, and transaction throughput. It allows users to instantly detect performance issues and proactively troubleshoot and optimise. Moreover, the alerting capabilities in Grafana enable notifications on an automated basis once certain thresholds of performances are breached [26].

Coupling and integrating Caliper and Grafana provides enhanced and efficient technological framework for a more open data-driven testing process of blockchain development and maintenance. These tools offer insights to developers regarding how systems perform under various configurations and conditions to make informed decisions on system design and optimisation. This becomes all the more important in projects that seek to deploy blockchain solutions in production environments where performance and scalability are critical.

In addition to the use of Hyperledger Caliper and Grafana, Prometheus has also been integrated within our benchmarking and monitoring environment to make our performance evaluation robust. Prometheus is an open-source monitoring and alerting toolkit that is specially designed for gathering and querying time-series data. Prometheus allows the user to continuously monitor system metrics such as memory usage, CPU utilisation, and Network I/O, therefore, giving deep insight into the real-time performance and resource consumption happening on the deployed blockchain network. Real-time monitoring is key to address performance bottlenecks and system anomalies, especially under

varying network loads and various configurations. Moreover, Prometheus combined with Grafana's visualization, can predict and deliver issues and warning, before allowing them to affect performance or reliability. Taking all the aforementioned parameters into consideration, TALON's blockchain toolkit validation approach employed the combination of the capabilities of Hyperledger Caliper, Grafana, and Prometheus for thorough and reliable testing of the implementation, which met best practices with regard to both performance benchmarking and system reliability [27].

In the table below the key technologies along with their functionalities and added values are showcased:

Table 3: Validation technologies and utilities

Tools	Monitoring	Metrics	Utility
Caliper	Performance	• Latency (sec) • Throughput (TPS)	• Check implementation robustness • Do potential enhancements
Grafana	Energy	• CPU Utilisation (%) • Power Consumption (Watts)	• Check implementation efficiency • Trigger alerts on threshold breach
Prometheus	Visualisation	• Data Trends • System Health Metrics	• Data fetching • Alerting on system anomalies.

6.2 Scientific and Technical Results

This section provides an elaborative and detailed analysis of the scientific and technical results obtained from the performance evaluation of the TALON Blockchain toolkit. Results are based on an extensive exercise of testing and validation techniques using industry-standard tools such as Hyperledger Caliper and Grafana. The benchmarking scenarios have been designed to carefully probe the performance of the toolkit under various configurations and network loads, matching the KPIs identified in previous deliverables. These experiments have been performed with the aim of validating the robustness, scalability and performance of the toolkit in realistic operational conditions.

The test setup involved the following key parameters:

- Maximum transaction count per block: Determines the maximum number of transactions that can be grouped and processed together in a single block, impacting transaction throughput and latency.
- Block size limits: Sets the maximum size allowed for each block, influencing how much data can be stored and processed in a block, affecting network efficiency and scalability.
- Preferred block size: The target block size for optimal performance, balancing transaction throughput with efficient network bandwidth use.
- Number of pending transactions on the network: Reflects the volume of transactions waiting to be processed, simulating different network congestion levels to test the system's performance under load.

We have applied four different configurations as shown in Table 4 and executed an experiment for each one. Configuration 1 was with one transaction per block, and the block size limit was set to 40 MB. Configuration 2 allowed up to 10 transactions per block, having a block size of 20 MB. Configuration 3 supported 20 transactions per block, with its maximum size to be 40 MB. While Configuration 4 was on standard block sizes, it also supported 20 transactions per block and had a preferred size of 10 MB. Each configuration was stress tested under three different network loads— 5, 10, and 20 pending transactions— the purpose being to demonstrate various scenarios of operation and assess how different transaction volumes and block configurations affect the toolkit's performance under realistic network conditions.

Table 4: Network Performance Metrics under Various Configurations

Quantitative Results:

Experiment#	Max Tx in a Block	Max block size (MB)	Preferred block size (MB	Pending Tx on network	TPS	Average Latency (s)	Min Latency (s)	TSR (%)
1	1	40	20	5	17.2	0.40	0.17	100
				10	17.8	0.52	0.19	100
				20	17.9	0.76	0.18	100
2	10	20	10	5	10.3	0.62	0.17	100
				10	12.6	0.69	0.17	100
				20	17.5	0.77	0.20	100
3	20	40	20	5	3.5	1.45	0.18	100
				10	4.1	1.65	0.17	100
				20	13.1	1.58	0.32	100
4 SIDROCO	20	10	2	5	3.0	1.60	0.19	100
				10	4.2	1.01	0.17	100
				20	13.3	1.01	0.29	100

The performance evaluation focused on two primary metrics: Transactions Per Second (TPS) and latency, both average and minimum. In the context of Hyperledger Fabric, TPS measures the rate at which transactions are successfully committed to the ledger, serving as an indicator of throughput under varying network loads. Latency, on the other hand, represents the time delay from when a transaction is submitted to when it is finalised on the ledger, capturing both the responsiveness and efficiency of the network. It can be seen from Table 4 that the highest TPS in combination with the lowest average throughput comes with the setup of experiment #1. Allowing one transaction per block with maximum volume of 40 MB (specific for TALON's FL engine) enables stable TPS ~ 17 and latency less than 0.8 seconds (and minimum latency less than 0.2 in all configurations). Accompanying charts (Figure 28) of Table 4 show several key trends that consolidate this claim. In the following paragraphs the results of Performance Analysis of Blockchain Experiments are further explained with regard their various dimensions specifically to derive the results.

In the top-left chart (TPS vs. Pending Transactions on Network), TPS serves as a proxy for throughput. Experiment 1, which has the largest block size (40 MB) but a single-transaction limit per block, achieves the highest TPS under low network loads, indicating high throughput in these conditions. However, under heavier loads, Experiment 1's throughput declines, as it cannot handle the increased demand. Experiment 3, with a higher transaction limit of 20 and the same large block size (40 MB), sustains better performance under heavier loads, suggesting that while smaller blocks are more efficient with low workloads, larger blocks are better suited for handling a larger volume of transactions. Experiment 2 and Experiment 4 offer intermediate configurations. Experiment 2 has a mid-sized block (20 MB) and a transaction limit of 10, striking a balance between transaction capacity

and block size. Experiment 4, on the other hand, combines a smaller block size (10 MB) with a high transaction capacity of 20, allowing for relatively high throughput in scenarios with lighter network loads but limited block space.

In terms of latency, as shown in the top-right chart (Average Latency vs. Pending Transactions), Experiment 1 consistently exhibits lower average latencies, highlighting its efficiency in processing fewer transactions within smaller blocks. This low-latency characteristic is advantageous for applications that require rapid data integrity verification. However, as the top-left TPS chart shows, Experiment 1's throughput (measured by TPS) suffers as the network load increases, revealing a trade-off between low latency and throughput in high-load scenarios. In contrast, Experiment 3, as illustrated in both the top-left (TPS) and top-right (average latency) charts, maintains higher throughput due to its larger block size and transaction capacity, proving more effective under high network loads.

The bottom-left chart (Min Latency vs. Pending Transactions) shows that minimum latency remains relatively stable across experiment 1 and experiment 2. While experiment 1 is stable and declines as the number of the transactions on queue is increasing, experiment 2 is linearly increasing latency. Moreover, experiment 3 and experiment 4 are increasing latency significantly quicker than the other two setups. Experiment's 1 consistency suggests that the fundamental time required to process individual transactions does not fluctuate significantly, contributing to predictable performance as network conditions vary.

Finally, the bottom-right chart (Max Transactions in a Block vs. TPS) further underscores the relationship between maximum transactions per block and TPS. This chart emphasises that larger blocks with higher transaction capacities can support improved throughput, as reflected by higher TPS under higher transaction volumes. From this graph the reader can see that experiment 1 is achieving the best TPS score with experiment 2 scoring slightly less. Again, here experiment 1 has stable performance fluctuating approximately at 17 TPS with almost no deviations, while experiment 2 is on the scale from 10 to 17 TPS, rendering it more volatile.

rg

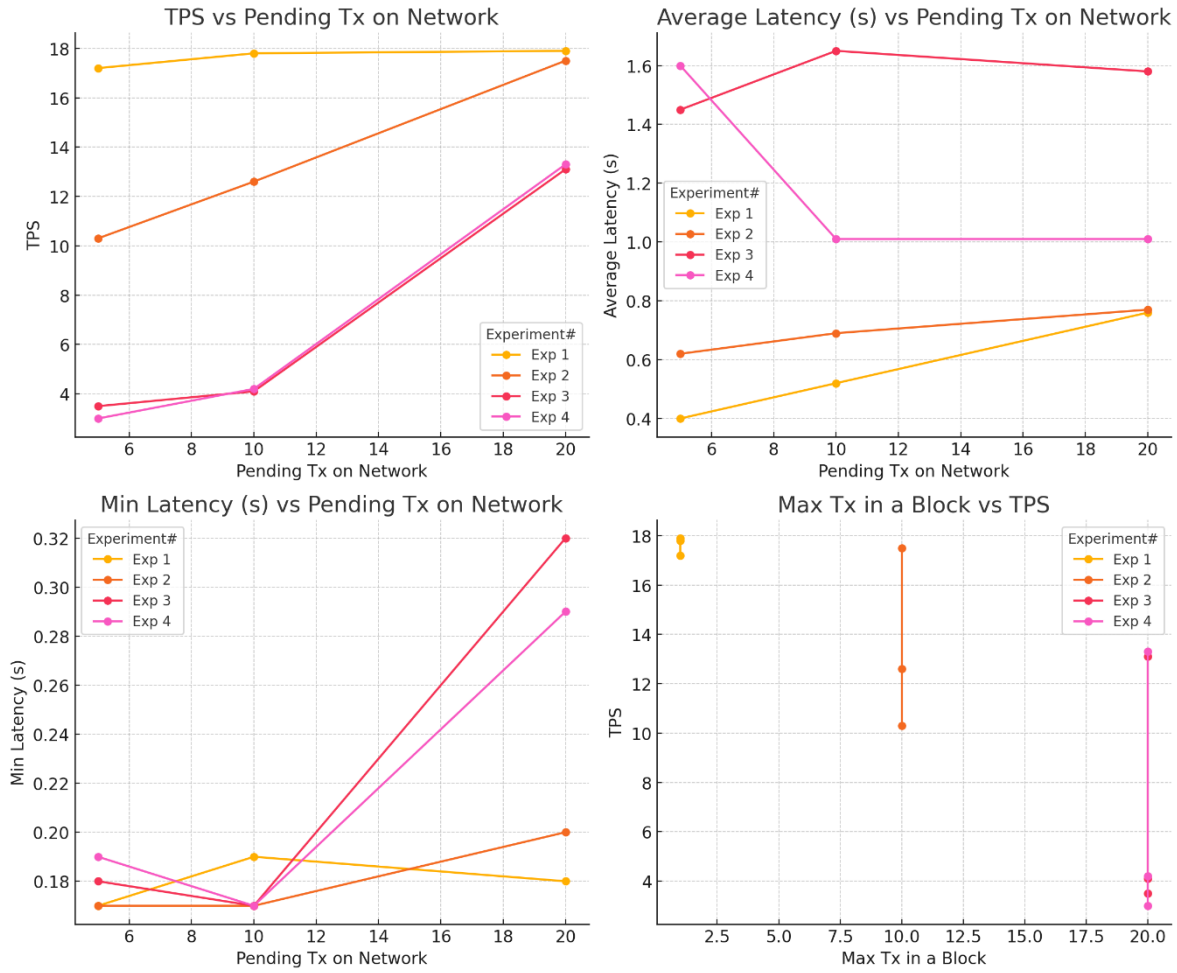

Figure 28: Performance Analysis of Blockchain Experiments

Following the performance testing, an experiment was conducted on a local machine (PC) to measure the energy consumption of TALON's blockchain setup. The test machine, equipped with an i7-14700KF processor, has a maximum Turbo Power of 253W and can boost up to 5.5 GHz. The key objective of this experiment is to capture an intensive overview of system resource utilisation for several operation states. We captured the memory and CPU usage on a per-container basis, measure baseline resource demands and additional loads that were induced by stress testing using Caliper.

All these metrics have been measured in various conditions with the aim of assessing the energy efficiency and resource needs of each container involved in TALON's blockchain network. The results highlight steady-state demands when the network operates on minimum activity level and shed light on extra memory and CPU load created due to intensive testing.

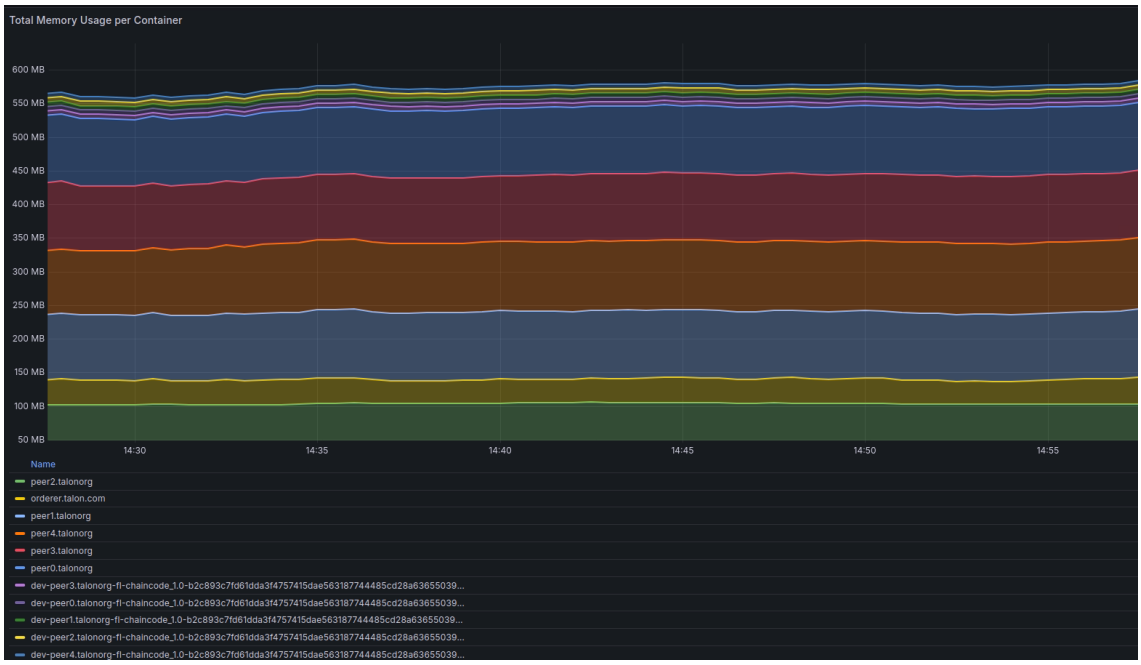

Figure 29: Memory usage per container with Fabric running

Figure 29 shows the total memory usage per container while the TALON's blockchain network is running, which directly allows us to observe the baseline memory footprint of each container participating in the TALON blockchain network. Each container represents a component of the network, such as peers, orderers, and chaincode instances. The relatively stable memory usage across these containers, as seen in Figure 29, suggests that each component maintains a consistent memory demand, which is also beneficial for predictable resource allocation.

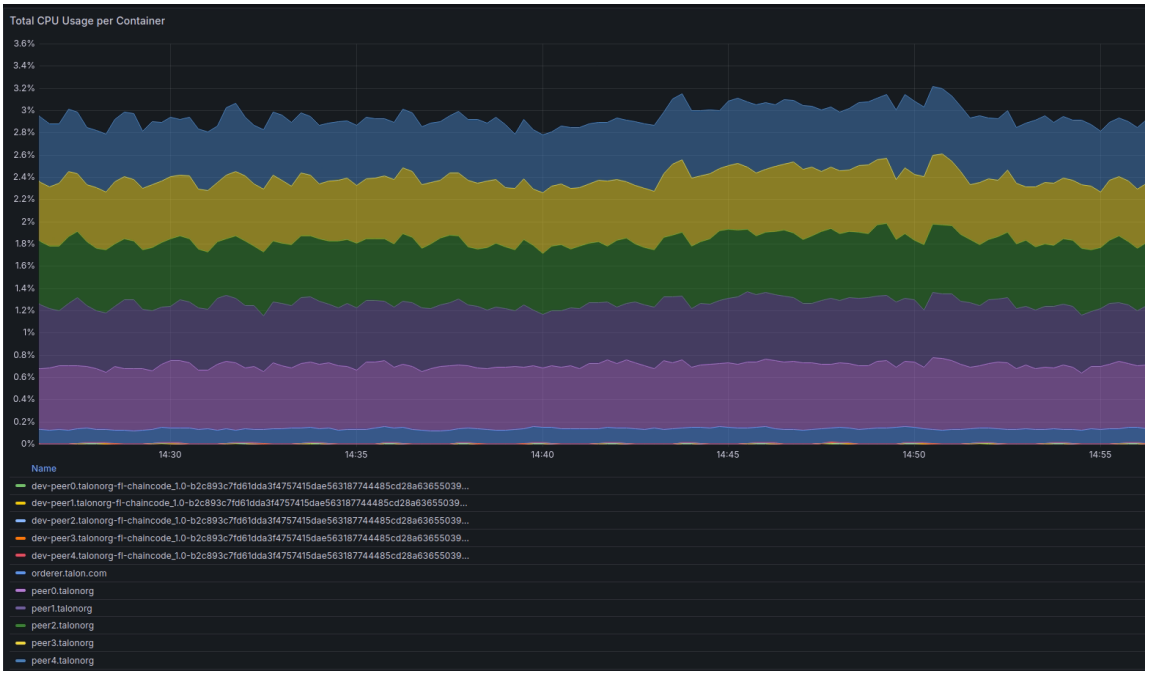

Figure 30: CPU usage per container while Fabric running

Figure 30 summarises the overall CPU utilization per container when Fabric is running. It shows that each container maintains less than 3% CPU utilization, indicating a minimal processing requirement. The relatively stable CPU utilization, with only minor fluctuations, suggests that the system is

operating within a steady state without significant computational demands. This level of stability is advantageous for maintaining predictable performance and reducing the risk of CPU bottlenecks during routine operations.

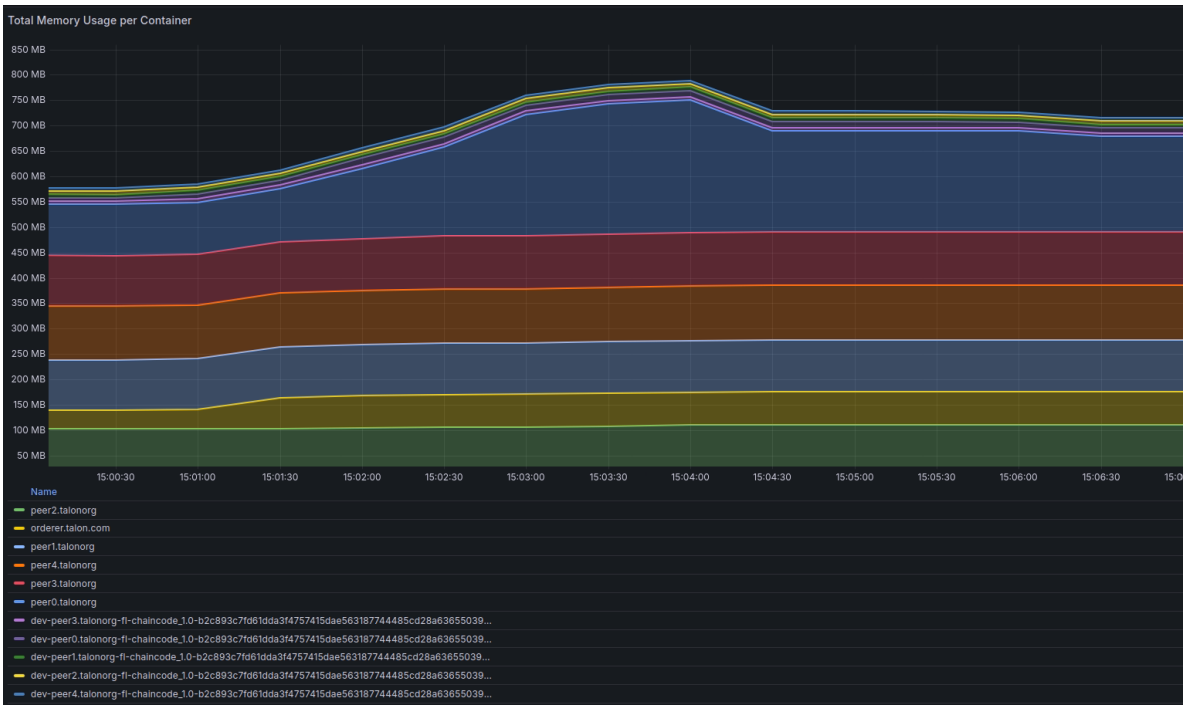

Figure 31: Memory usage per container while caliper running

With Caliper running, the network is actively processing transactions, simulating a realistic workload where various components of the Fabric network engage in executing and validating transactions. Figure 31 visualises the memory usage per container during the Caliper test run. During this test, the memory consumption increased across all containers, indicating the added load from transaction processing. The memory usage peaked in the middle of the test, likely reflecting the highest point of transaction activity, and then stabilized as the workload returned to a steady state. This pattern demonstrates the network's ability to handle increased resource demands during periods of high activity and efficiently return to baseline levels after the peak.

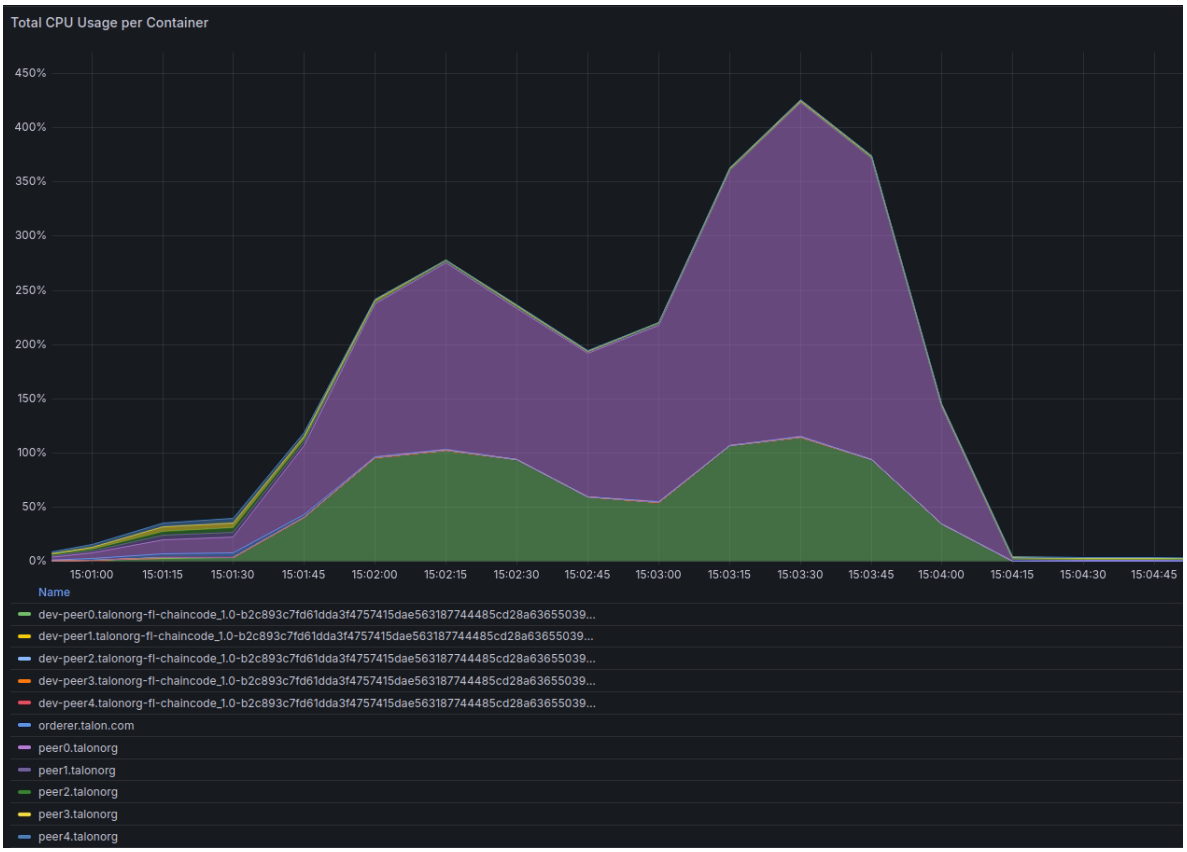

Figure 32: CPU usage per container while caliper running

Figure 32 shows CPU usage per container during Caliper load, with CPU usage spiking significantly as specific containers handled active tasks. The CPU utilization peaks around 15:02, reaching over 250% for some containers, particularly the dev-peer chaincode containers. These chaincode containers experienced the highest CPU demand because they are responsible for executing smart contract logic, validating transactions, and interacting directly with the ledger, which are compute-intensive tasks during high transaction volumes. The peer and orderer containers also showed increased CPU usage, though to a lesser extent, as they manage communication and consensus across the network rather than the direct execution of transaction logic. After this peak, the CPU usage gradually declines as the workload decreases and returns to a stable state. This information is valuable for identifying resource-intensive components and optimizing performance under heavy transactional demands.

Energy consumption is at the forefront of crucial consideration in the operation of blockchain systems, which are usually power-consuming. TALON's implementation scopes to be lightweight and energy-efficient, so we conducted an experiment to measure the watt consumption of the blockchain setup in several operational states. The Powerstat [25] tool, which monitors and reports real-time power usage, allowed us to observe power consumption across different states.

When idle, the machine averaged 40.31W, which increased to 50.75W with Fabric running and jumped significantly to 214.55W when executing Caliper tests. These measurements provided a baseline understanding of how different workloads impact energy usage.

[25] *https://manpages.ubuntu.com/manpages/trusty/man8/powerstat.8.html*

To achieve accurate estimates of energy consumption directly attributable to the blockchain setup, we accounted for the energy impact of the Caliper container itself, which also consumes power as it runs load tests. Since Caliper isn't part of the core blockchain operations but merely a testing tool, its energy usage needs to be subtracted from the total to isolate the power consumed by the blockchain system alone.

We assumed a linear relationship between CPU utilisation and energy consumption, using CPU load as a proxy for power. From Grafana and Prometheus, we observed that the Caliper container used 9.21% of the total CPU during testing, which translates to approximately 19.71W of power out of the system's total 214.55W under load. After isolating the Caliper container's impact, we determined that the blockchain network consumed approximately 194.84W, resulting in a net increase of 144.53W over the baseline with Fabric running. This approach provided a more accurate measure of energy consumption under realistic transaction loads, helping to better assess the implementation's energy efficiency in production-like conditions.

Results are demonstrated in Figure 33 below:

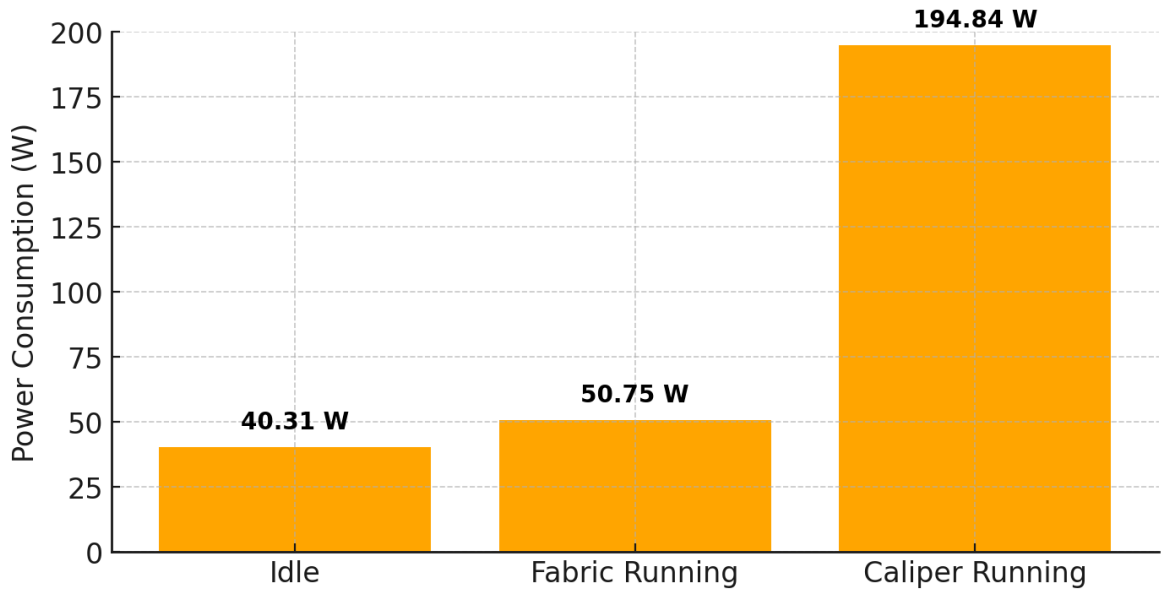

Figure 33: Energy Consumption Analysis of Fabric in Different Operational States

Based on these measurements, we estimated the annual energy consumption of the Hyperledger Fabric network by converting power usage (Watts) into kilowatt-hours (kWh), the standard unit for energy accounting. During high transaction loads, Fabric consumed an average of 194.84W. To calculate the hourly energy consumption, we converted this power usage into kWh using the following calculation:

$$Hourly\ Consumption = 194.84\ W\ \times 0.001\ kWh/W = 0.19484\ kWh$$

This result represents the energy consumed by the network over the course of one hour. We then calculated the daily energy consumption by multiplying the hourly consumption by 24 hours:

$$Daily\ Consumption = 0.19484\ kWh\ \times 24 = 4.67616\ kWh/day$$

This value gives us the energy usage for a full day of continuous operation. Finally, to estimate the yearly energy consumption, we multiplied the daily consumption by 365 days:

$$Annual\ Consumption = 4.67616\ kWh/day\ \times 365\ days = 1{,}707.78\ kWh\ \approx 1{,}708\ KWh$$

This calculation provides an approximate annual energy consumption of 1,708 kWh, assuming that Hyperledger Fabric runs constantly at high transaction load. Figure 34 presents a comparable metric to analyse the energy profile of Fabric in relation to public blockchains like Ethereum[26] and Bitcoin[27]. Ethereum and Bitcoin were selected for comparison due to their prominence as widely adopted public blockchain platforms, representing both Proof-of-Stake (Ethereum) and Proof-of-Work (Bitcoin) consensus mechanisms. By establishing a standardised annual consumption approach, this comparison allows for a clearer understanding of the relative energy requirements of private, permissioned networks versus public, decentralized systems within the blockchain ecosystem.

In contrast, Ethereum's PoS network is estimated to consume between 10,000 and 50,000 kWh per year, according to recent studies [28]. While Ethereum made a great shift from PoW to PoS, which significantly reduced energy output, it is still generating considerably more power compared to Fabric's permissioned environment. This gap ranges from the architectural to the consensus differences between the two networks: Fabric's private model, with controlled nodes, minimises the need for continuous global validation, whereas Ethereum's public, decentralised model requires a distributed network of Validators, each maintaining consistent energy use regardless of transaction load.

The PoW consensus mechanism utilized within Bitcoin is has high power consumption, where powerful computational resources are being used by miners through endless hash calculations to continuously add blocks to the chain. This process has resulted in huge amounts of electricity consumption by the Bitcoin network as a whole—estimated to reach 204.5 TWh per year—equivalent to the annual electricity consumption of an entire nation [29].

Conclusively, our experiment results and estimated yearly energy consumption experiments show that TALON's Fabric-based approach is more environmentally friendly than public, PoW or PoS systems, as shown in Figure 34. More specifically, Fabric's consumption of 1,708 kWh is 83-96% lower than Ethereum's estimated annual consumption of 10,000 to 50,000 kWh, depending on transaction load. When compared to Bitcoin's PoW network, with an estimated 204,500 kWh per year, Fabric's energy usage is over 99% lower, making it a considerably greener choice. These findings highlight Fabric's potential as an energy-efficient solution, particularly suitable for private, permissioned environments where sustainability is a priority.

[26] *https://ethereum.org/en*
[27] *https://bitcoin.org/en*

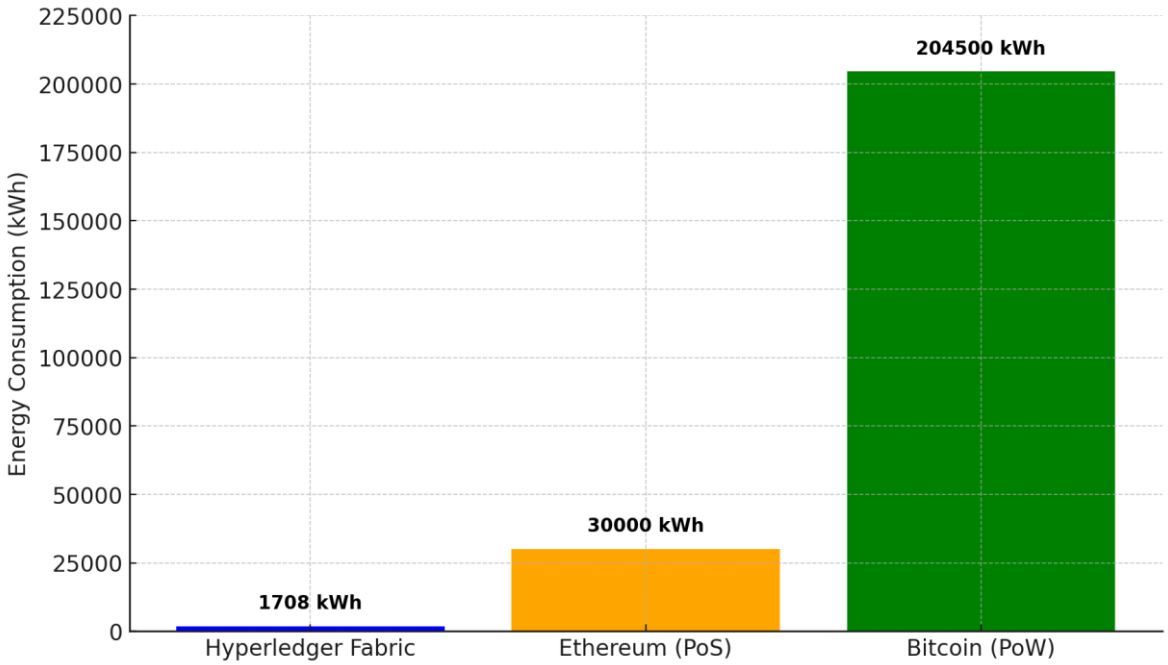

Figure 34: Annual Energy Consumption by Blockchain

Based on insights generated from the above experiments, the system parameters were iteratively fine-tuned to optimize dApp performance in alignment with specific requirements from various components within the TALON ecosystem. Through benchmarking functionalities provided by Hyperledger Caliper, the resilience and capability of these dApps to meet real use case demands were effectively demonstrated. All results and configurations have been documented thoroughly, establishing a clear foundation for continuous optimization and further development. The rigorous testing and systematic validation supported by Hyperledger Caliper enabled effective optimization of dApps for TALON's envisioned complex and dynamic environments.

7 Conclusion & Future Outlook

In the previous sections, the detailed approach of TALON's blockchain has been outlined. Initially, preliminary research has been conducted to classify the various blockchain technologies that are actively utilised by the community along with their merits and drawbacks. This endeavor has been very helpful to select the appropriate technological framework to support our implementation with regards to the specific TALON requirements. Moreover, a generic and brief overview of some blockchain applications for existing use cases has been overviewed in order to perform the necessary correlations with TALON pilots and examine "a priori" that the selected solution is matching the demonstrator demands. In addition to the above, the deliverable explains the significant role of dApps as a technological enabler for decentralised applications and makes a direct reference to TALON's implementation and added value.

After connecting the dots with existing technologies, methods and approaches the document presents TALON's blockchain significance to the project and in conjunction with the FL.

Having concluded with the preliminary yet important prerequisite information, the internal structure of the blockchain is presented. The user can navigate at a higher level on the general architecture and delve into the specifics of why choosing this approach and how the various sub-modules inter-communicate.

After the schema presentation, the implementation phase is showcased. With detailed specifics about software, libraries, APIs, endpoints and UI, the user can grasp the core part of this document. Dockerisation and cloud infrastructure aspects are also demonstrated elaborating how external stakeholders can establish access with the tool.

Validation. This section is separated into to sub sections. One of the most important aspects of the technical validation is the establishment and declaration of the correct and widely used tools by the community, in order to give scientific and pragmatic importance to the generated results. On the other hand, in this section the actual results from the testing and implementation exercises are displayed. The results are performance metrics that attest the robustness and functionality, as well as energy efficiency analytics that demonstrate the lightweight nature of TALON's blockchain as a standalone tool and in comparison with global-scale used tools.

The results provide a comprehensive set of performance metrics that validate the robustness, functionality, and scalability of TALON's blockchain solution. Additionally, detailed energy efficiency analytics underscore its lightweight design, highlighting both its low resource consumption and optimised processing demands. These findings emphasise TALON's effectiveness both as a standalone solution and in comparative benchmarks against widely adopted, large-scale blockchain platforms. Together, the performance and efficiency results illustrate that TALON's blockchain meets the project's standards for reliability and sustainability.

At this stage in the TALON project, the development and initial testing phases of the blockchain framework have been successfully completed, marking a significant milestone and the compilation of this deliverable. Integration cycles with some project partners have been conducted to ensure compatibility and seamless operation within their existing infrastructures (FL). Concurrently, ongoing efforts are focused on integrating blockchain with one or more additional machine learning solutions contributed by other consortium members. This integration aims to leverage the unique capabilities of each partner's ML models, creating a secure and trustable system for optimised data tracking, secure storage and private information sharing.

In conclusion, it is worth mentioning that blockchain stands as an operational and tested module of TALON's platform and is successfully progressing towards the last phase of the project, which is the

pilot implementation. The final blockchain toolkit is expected to be incorporated into the complete TALON framework detailed in the upcoming D4.3 report. This document will act as a thorough resource, consolidating all aspects of TALON's blockchain capabilities and presenting them within the context of the project's fully developed technical modules. It will showcase TALON's blockchain functionalities in their final form, providing a reference for their role and integration within the broader system.

8 References

[1] *X. Xu, I. Weber, M. Staples, L. Zhu, J. Bosch, L. Bass, C. Pautasso and P. Rimba, "A Taxonomy of Blockchain-Based Systems for Architecture Design," in* 2017 IEEE International Conference on Software Architecture (ICSA), *2017.*

[2] *P. Tasca, T. Thanabalasingham and C. J. Tessone, "Taxonomy of Blockchain Technologies. Principles of Identification and Classification," in* Ontology of Blockchain Technologies. Principles of Identification, *2017.*

[3] *J. Berryhill, T. Bourgery and A. Hanson, "Blockchains unchained: Blockchain technology and its use in the public sector," 2018.*

[4] *P. R. NAIR and D. R. DORAI, "Evaluation of performance and security of proof of work and proof of stake using blockchain," in* Third International Conference on Intelligent Communication, *2021.*

[5] *Yang, F. a. Zhou, W. a. Wu, Q. a. Long, R. a. Xiong, N. N. a. Zhou and Meiqi, "Delegated proof of stake with downgrade: A secure and efficient blockchain consensus algorithm with downgrade mechanism,"* IEEE Access, *2019.*

[6] *Belchior, Vasconcelos, Guerreiro and Correia, "A survey on blockchain interoperability: Past, present, and future trends,"* ACM Computing Surveys (CSUR), *2021.*

[7] *R. Yang, R. Wakefield, S. Lyu, S. Jayasuriya, F. Han, X. Yi, X. Yang, G. Amarasinghe and S. Chen, "Public and private blockchain in construction business process and information integration,"* Automation in Construction, *2020.*

[8] *O. Dib, K. Brousmiche, E. T. A. Durand and E. Hamida, "Consortium blockchains: Overview, applications and challenges,"* International Journal of Advanced Telecommunications, *2018.*

[9] *Y. Zhang, Y. Wu, T. Li, H. Zhou and Y. Chen, "Vertical federated learning based on consortium blockchain for data sharing in mobile edge computing,"* Computer Modeling in Engineering Sciences, *2023.*

[10] *A. M. Antonopoulos, O. Osuntokun and Pickhardt, Mastering the Lightning Network, 2021.*

[11] *Armstrong and Matthew, "Ethereum, Smart Contracts and the Optimistic Roll-up," 2021.*

[12] *M. Alshaikhli, T. Elfouly, O. Elharrouss, A. Mohamed and N. Ottakath, "Evolution of Internet of Things from blockchain to IOTA: A survey,"* IEEE Access, *2021.*

[13] *Mazzoni, Marco, A. Corradi and V. D. Nicola., "Performance evaluation of permissioned blockchains for financial applications: The ConsenSys Quorum case study,"* Blockchain: Research and Applications, *2022.*

[14] *Rathee, Sharma, Iqbal, Aloqaily, Jaglan and Kumar, "A blockchain framework for securing connected and autonomous vehicles,"* Sensors, *2019.*

[15] *Mohamed and Al-Jaroodi, "Applying blockchain in Industry 4.0 applications,"* 2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC), *2019.*

[16] *Saranya, Komarasamy, Mohanapriya, Iyapparaja and Prabavathi, "Industry 4.0: The Role of Industrial IoT, Big Data, AR/VR, and Blockchain in the Digital Transformation," 2024.*

[17] *Y. Lu, X. Huang, Y. Dai, S. Maharjan and Y. Zhang, "Blockchain and federated learning for privacy-preserved data sharing in industrial IoT,"* IEEE Transactions on Industrial Informatics, *2019.*

[18] *Cai, W. Wang, Ernst and Zehua, "Decentralized applications: The blockchain-empowered software system,"* IEEE Access, *2018.*

[19] H. Guo and X. Yu, "A survey on blockchain technology and its security," Blockchain: Research and Applications, *2022.*

[20] Sasikala, Sundaram, Biswas, A. Nikhil and Rohith, "Survey of latest technologies on Decentralized applications using Blockchain," 2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS), *2022.*

[21] Zheng, Jiang, Wu and Zheng, "Blockchain-based decentralized application: A survey," IEEE Open Journal of the Computer Society, *2023.*

[22] Harichandan, Kar and Kumar, "A critical review on blockchain frameworks for dapp," International Journal of Technology Management and Information System, *2023.*

[23] C. Cachin, "Architecture of the Hyperledger blockchain fabric," Workshop on Distributed Cryptocurrencies and Consensus Ledgers, *2016.*

[24] A. Baliga, Subhod, I., P. Kamat and S. & Chatterjee, "Performance evaluation of the Quorum blockchain platform," *2018.*

[25] Shuaib, Hassan, Usman, Alam and Bakar, "Performance Evaluation of DLT systems based on Hyperledger Fabric," 2022 4th International Conference on Smart Sensors and Application (ICSSA), *2022.*

[26] S. Tavonatti, D. Battulga, M. Farhadi, C. Caprini and D. Miorandi, "An experimental evaluation of the scalability of permissioned blockchains," 2021 8th International Conference on Future Internet of Things and Cloud (FiCloud), *2021.*

[27] K. Kaushal, "Blockchain Implementation with Hyperledger Fabric and Approach for Performance Evaluation," 2023 IEEE International Conference on Blockchain and Distributed Systems Security (ICBDS), *2023.*

[28] A. Khosravi and F. Säämäki, "Beyond Bitcoin: Evaluating Energy Consumption and Environmental Impact across Cryptocurrency Projects," Energies, *2023.*

[29] Harichandan, S. Kar, Kumar and Abhishek, "A Comprehensive Review of Bitcoin's Energy Consumption and Its Environmental Implications," Machine Learning Approaches in Financial Analytics, *2024.*

[30] J. Berryhill, " Blockchain Technology and its Use in the Public Sector," *2018.*

[31] Becker, Y. a. Naumann and Stefan, "Software based estimation of software induced energy dissipation with powerstat," From Science to Society: The Bridge provided by Environmental Informatics, *2017.*

[32] Dessalgn, Sharma, Chung and Sungheetha, "Generative Adversarial Network-Based Visual-Aware Interactive Fashion Design Framework," *2022.*

Funded by
the European Union

This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No 101070181