blockchain technology for data provenance and proof of integrity

Jack Tallis
Cyber Security and forensic computing

Abstract

This report aims to evaluate blockchain technology's transformative idea in providing a decentralised, secure, and immutable tech for data provenance and proof of integrity. This is particularly crucial for healthcare organisations and cloud services, as it addresses data provenance and proof of integrity issues.

The aim is to implement a blockchain-based provenance architecture (ProvChain) and evaluate its interchangability. Objectives include identifying an architecture, examining various combinations of cloud services/blockchain networks, and understanding the improvements

Contents

Methodological approach

Methodology	Description
Waterfall	Sequential timeline, with phases: requirements, design, etc
Agile	Incremental and iterative, emphasising collaboration and flexibility
Scrum	Scrum is an agile project management methodology emphasising iterative progress, collaboration, and continuous improvement.
Lean development	Focuses on delivering value, by minimising waste
DevOps	Integration of development and operations to automate the software delivery process
Hybrid approach	Combination of multiple approches

Table 1: Methodology Description

Methodology	Drawbacks in the context of the Blockchain project
waterfall	Lack of flexibility for accommodating changes
Agile	Requires active involvement from third parties/stockholders
Scrum	Need help integrating complex technical requirements
Lean Development	Limited emphasis on documentation is crucial for ensuring transparency and accountability in blockchain projects.
DevOps	priorotises speed over security, potentially compromising the integrity.
Hybrid Approach	Careful planning should be taken when combinine two techniques together

Table 2: Drawbacks

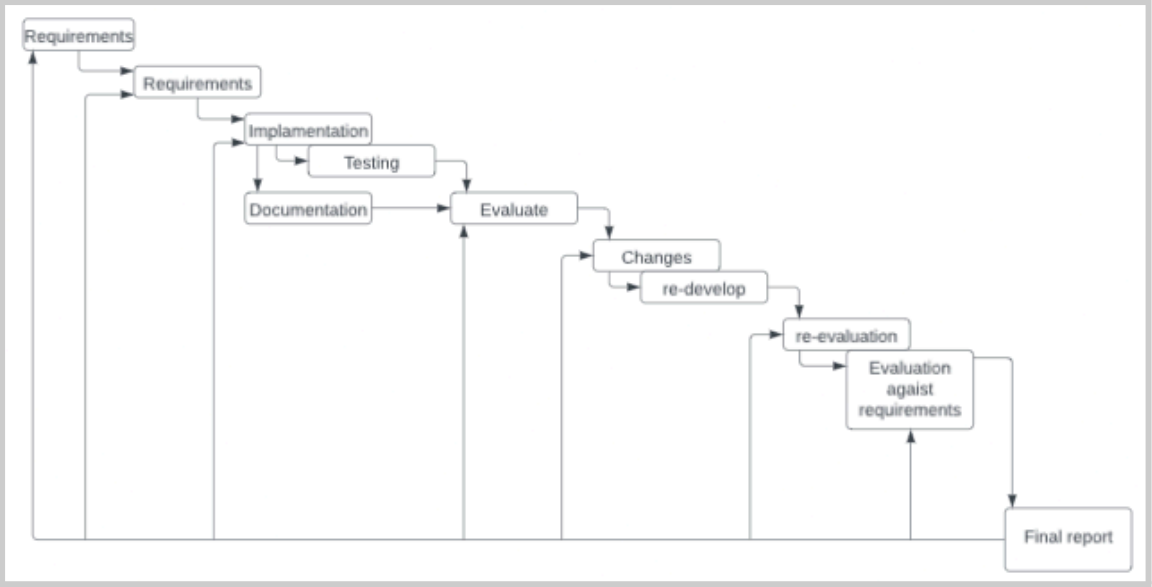


Figure 1: Waterfall, iterative method

Figure 1 illustrates the confluence of iterative waterfall approaches. The main focus is on the advantages of the structured waterfall approach for administration and documentation, although periodic switches to iterative methods will account for advancements in software development.

Every alteration will be recorded in the Project Initiation Document for traceability and transparency. This strategy aims for efficiency, adaptability while balancing rigidity and flexibility, enforcing effectiveness, flexibility.

1 Introduction

Blockchain offers decentralised, secure, and unchangeable data management solutions. In cloud environments, ensuring this data security via data provenance and proof of integrity is vital. The blockchain enhances data authenticity and transparency over time.

This report will evaluate blockchain technology implementations for data provenance and integrity solutions in cloud service environments. Objectives include:

- Understanding architecture.
- Seeing how it could be improved.
- Evaluating and compare the benefits of interchangeable features.

This is done by exploring different cloud service providers and blockchain networks to see how the solution could be optimised for performance and integration. By understanding these combinations, the project aims to identify the most effective solution, but this depends on the use case.

2 Blockchain technology for data provenance and proof of integrity

This project aims to understand the issues with data provenance and evidence of integrity. It will then utilise blockchain technology solutions to manage the logs for data provenance, simplifying the process of proving data integrity and placing it all in a secure location available to the admins.

Data provenance

Data provenance is the lineage and processing history of data/any product or anything and the record of the processes of who had and it what happened with it (Wang et al., 2015, October). However, the application of data provenance for IoT can be an issue, as stated by Alkhalil and Ramadan. This is due to the unique characteristics of IoT, particularly the vast data and the connection of devices with minimal computation power (Alkhalil et al., R. A. (2017)).

The significance of data provenance in data management is defined as the lineage, and processing history must be balanced. Understanding the lineage of data is crucial for ensuring data integrity and reliability, particularly in scientific databases where provenance plays a significant role in the validation.

Issues of data provenance

The application of viewing the lineage of a piece of data. Lineage means the beginning of where that data set began and where it is now. It can be defined as detecting the lineage and the derivation of data and data objects (Alkhalil et al., R. A. (2017)). This is the main idea of data provenance. (Buneman et al., 2000) In scientific databases, where provenance is essential to data validation, it is a severe problem.

(Bonneman & Davidson, 2010) Data copying and transformation are two of the main challenges with data provenance. While copying and transforming data are essential to processes, managing these actions effectively in the context of data provenance is crucial to maintaining data's reliability, integrity, and security. Using data provenance systems helps to address the challenges by providing tracking and documentation abilities.

In this sense, copying is when someone creates a similar or identical version of something (Bruneman & Davidson, 2010). When a copy of an artefact is made, copies are often used; copying said artefact destroys its provenance. This idea represents the issues that come when copying anything; the loss of lineage information happens when copying if it is made without proper tracking and documentation. These are the main components of provenance; with a clear understanding, it can become easier to trace provenance.

With this idea, copying itself is manageable for provenance. Multiple copies of data have to be made, especially if you want to try to preserve the integrity of the data; however, versions of that artefact can be lost since the copy could become the primary source of that artefact.

Proof of data integrity

Data integrity refers to data's accuracy, completeness, and quality as maintained/transferred over time. Integrity means that a document has not been altered. If something is authentic, its source may be determined. We can choose a time reference for the document's existence by obtaining proof (Vigil et al., 2015). However (Tan et al. (2018) suggest preserving the availability and integrity of data, overcoming obstacles related to communication, processing, and storage effectiveness; compatibility with devices with limited resources, vulnerability to malicious attacks, and inherent conflicts with data deduplication are some issues that come with proof of data integrity.

Developing effective and safe plans that complement cloud storage's dynamic features requires addressing these problems, mainly when dealing with various devices, possible cyber threats, and changing requirements for data manipulation.

Issues with proof of integrity

since data integrity refers to the ability to demonstrate that data has not been altered or tampered with during its lifecycle

Tan et al. (2018) emphasise the critical role of processing and storage effectiveness in maintaining data integrity. According to their findings, efficient processing ensures accurate data capture, validation, and storage, thereby safeguarding the overall integrity of the data. Wei et al. (2020) further underscore the importance of correctness and consistency in data across its lifecycle for ensuring data integrity. By synthesising these sources, it becomes evident that effective processing and storage are essential components in the broader discourse on data integrity.

In data integrity, the concept of a digital chain of custody, as explored by Prayudi and Sn (2015), is pivotal. The chain of custody, as outlined by these authors, documents crucial information regarding data handling, including where, when, why, who, and how. Establishing a transparent chain of custody is integral to proof of integrity, providing an accurate record of every interaction with the data from its inception to its final destination.

Compatibility with devices with limited resources (Tan et al., (2018); for proof of integrity, processing effectiveness is crucible. More device resources may lead to inefficiencies in data processing and delays.

Vulnerability to malicious attacks (Tan et al., (2018); aspects of malicious attacks like collision, immediately breaking the integrity of that file, since you break the integrity by creating an identical copy that is no longer the original file.

Inherent conflicts with data deduplication (Tan et al., (2018): Data duplication can lead to potentially incorrect calculations and decisions based on wrong data. They are causing data integrity issues when the company checks overall data integrity.

Utilising blockchain technology for data provenance and proof of integrity

Data provenance and integrity are critical issues in IoT-based environments such as smart cities, smart grids, vehicular networks and more (Javaid et al., 2018, November). However, provenance remains a concern for cloud storage applicaitions, since provenance data may contain sentitive information about data ownders (Liang et al., 2017, May). Data in the blockchain is immutable and permanent; it cannot be modified or deleted, preserving its integrity and allowing for traceability.

Blockchain is a database of transaction records distributed, validated, and maintained by a network of computers worldwide (Sarmah et al. (2018)). This allows multiple computers to make what is known as “blocks”. Each new block links with other blocks, adding cryptographic chains that make it nearly impossible to tamper with. With this idea, Golosova and Romanovs suggest that several issues in the industrial sector, including those involving trust, transparency, security, and data processing dependability, can be resolved using this technology (Golosova & Romanovs (2018)).

By leveraging blockchain, data can be logged to ensure its traceability, authenticity and reliability. One notable application of blockchain technology is Provchain (Liang et al., 2017, May). The architecture brings several benefits:

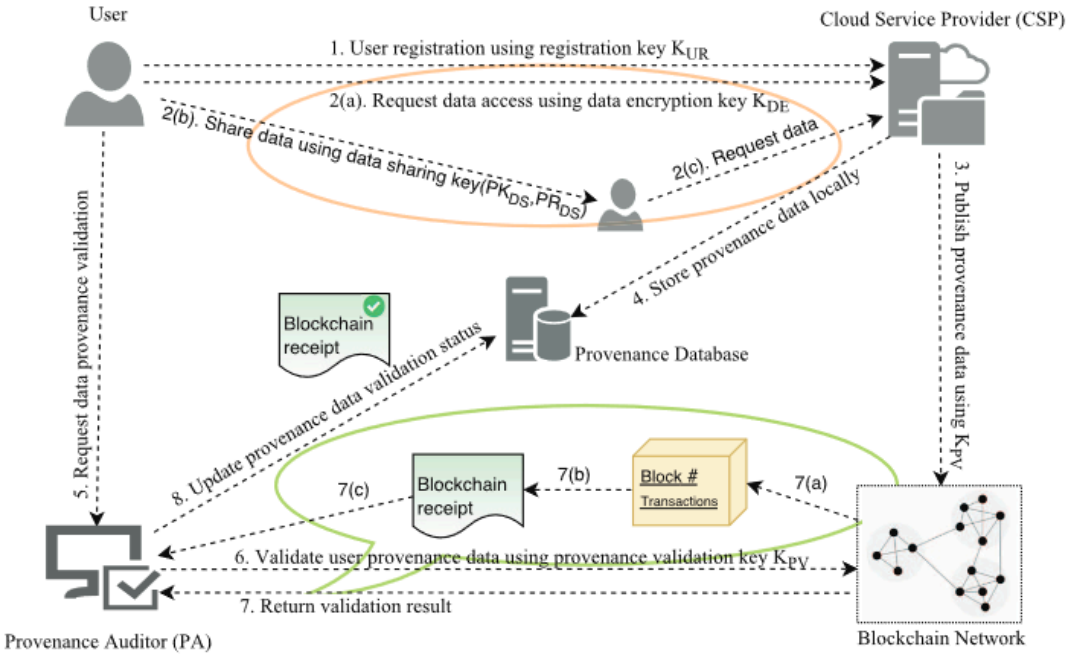


Figure 2: ProvChains architecture

1. **Real-time Monitoring:** ProChain continuously tracks and records all operations performed on data, such as editing, deleting, and creating files. Ensuring that admins can detect unauthorised access or alteration.
2. **Tamper-proof records:** All operations complete are stored on the blockchain, creating a tamper-proof ID of that data history. Since the blockchain is a distributed ledger, admins can independently verify the integrity of the data on that network.

3. **Privacy protection**: ProvChain uses a hashing mechanism to anonymise user identifiers while publishing provenance data to the blockchain.
4. **Verification:** The blockchain's distributed nature enables anyone to verify its provenance

The architecture includes essential components such as a provenance database, provenance auditor, and blockchain network.

1. **Provenance database:** This database stores all proofs on the blockchain.
2. **Provenance auditor:** The auditor retrieves and validates the provenance data from the blockchain.
3. **Blockchain Network:** The net consists of globally distributed nodes that verify and store provenance data.

ProvChain provides a comprehensive approach to data provenance and integrity in cloud environments by utilising all these components.

Implementing blockchain technology in data provenance and integrity verification exhibits significant potential to improve data management procedures in diverse fields. Subsequent investigations may broaden and modify the functionalities of this kind of system, tackling current issues and opening doors for data handling solutions.

Gaps in knowledge

However, a critical analysis of the existing literature reveals specific gaps in our understanding of how these components interact and the challenges that may arise in real-world scenarios. This literature review aims to address these gaps and contribute to a comprehensive understanding of the intricacies surrounding data integrity and its proof in processing, storage, and establishing a digital chain of custody.

3 Requirements - Software requirement specification

https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8029796

3.1 Introduction

This project's software requirement definition will follow the ISO/IEC/IEEE standard template. Given that the programme or application being produced is merely an idea. Certain sections of this template will no longer be required because they may not apply to this report.

3.2 Scope

The project integrates blockchain technology, a cloud service provider, a provenance auditor, anchoring software/services, and a database. These components manage the proofs' uploading, validating, and uploading, with data transfers managed via APIs.

User activities will be logged on the cloud service, hashed, and published on the blockchain. A webpage provides a blockchain transaction with proof confirmation; users can verify evidence via a command-line application.

One of the main advantages of blockchain technology is enhancing data provenance and integrity evidence; these features are essential for applications like supply chain management, healthcare, finance, and law that demand trust, transparency, and tamper-proof records. This system saves time by simplifying intricate procedures, reducing manual reconciliation work, and eliminating intermediaries. Blockchain's traceability and openness make it possible to quickly identify and resolve differences, which improves the effectiveness of audits and data validation.

3.3 Purpose

The Software Requirements Specification (SRS) document is necessary for project stakeholders to collaborate and communicate clearly. Outlining functional and non-functional needs offers a thorough picture of the software system. The SRS helps guarantee that the development team, project managers, and clients have a common understanding by outlining the programme's features, limitations, and expectations. This document allows developers to avoid misunderstandings, sets expectations, and lays the groundwork for the software development lifecycle by providing clear directions.

To manage modifications and avoid scope creep, the SRS is a dynamic reference that changes as the project does. It is a formal contract for validation and verification, guaranteeing that the finished product satisfies client requirements and established criteria. Transparency, accountability, and effective project outcomes are encouraged by the SRS, which adjusts to project developments.

3.4 Acronyms, abbreviations and definitions

API: An application programming interface, is a set of protocols that allow one software applications to connect with access to features or data of another applications

Applications: Chainpoints different programs, core, bitcoin header node, js, etc.

Blockchain technology: A decentralized and distrabutive network system that records and validates transactions acorss multiple different computers, ensuring transparaency, immutability, security and tamper-proof .

Bitcoin: Bitcoin is a decentralised digital currency operating on a peer-to-peer network, utilising blockchain technology to enable secure, transparent, and verifiable transactions, with a limited supply capped at 21 million coins.

Bitcoin header node: A Bitcoin header node is a component in the Bitcoin blockchain structure that contains essential information about a block, including its cryptographic hash, timestamp, Merkle root, and other metadata. This information is crucial for maintaining the integrity and chronological order of the blockchain.

Proofs: Blockchain proofs refer to cryptographic validations that confirm the authenticity and integrity of data stored in a blockchain, such as Proof of Work (PoW) or Proof of Stake (PoS). These mechanisms provide consensus mechanisms ensuring the legitimacy of transactions and maintaining the overall security of the decentralised ledger.

System: The final overall application of everything involved

User: Any person who uses the system.

4 Requirement elicitation

This requirement elicitation document is integral to a software project that evaluates the capabilities of a system. The system is designed to enhance data provenance and integrity by utilising blockchain technology, providing a decentralised and tamper-proof ledger.

The detailed specifications outlined in this document serve as explicit criteria for the task at hand, establishing guidelines that must be met to ensure the successful development and implementation. From functional and non-functional requirements to user expectations, testing criteria, compliance considerations, and beyond, this document provides a comprehensive overview of the desired outcomes for the project.

4.1 Specified requirements

Specified requirements refer to detailed and explicit criteria or conditions clearly defined and outlined for a particular task, or process. These requirements are instructions or guidelines that must be met to ensure that the desired outcome achieved. The term can be used in various contexts, such as software development, product design, etc.

4.1.1 Functional requirements:

Functional Requirements				
Requirement #	Name	Description	Justification	Priority
1	Data Provenance Tracking on blockchain	Tracks the origin and data throughout its lifecycle: creation, modification, deletion, access.	Tracking data's origin and lifecycle ensures data integrity and accountability. Recording events on the blockchain establishes a transparent and immutable history of data activities.	10
2	Data Provenance Collection	It automatically collects provenance data about stored data in the cloud environment, including the time and date of creation, creator, and modifications made.	Automatic collection ensures the accuracy and completeness of provenance records. Facilitates efficient data lineage tracking, forensic analysis, and compliance verification.	10
3	Data provenance storage	Provenance data is securely stored	Storing on the blockchain ensures tamper-proof	10

		on the blockchain, providing tamper-proof and permanent records of data history.	records. Blockchain's distributed ledger technology prevents retroactive alterations, establishing a reliable source of truth	
4	Data provenance validation	Users can validate data authenticity and integrity using the blockchain.	Enhances trust and confidence in data integrity, which is critical for decision-making and regulatory compliance scenarios.	10
5	Cryptographic technology	Utilises cryptographic techniques to protect user privacy and ensure data security.	Ensures confidentiality, integrity, and authenticity of data transactions, safeguarding sensitive information from unauthorised access or tampering	5
6	Data hashing	Data is hashed before transmission to the blockchain.	This is done to anonymise the user when submitting to the blockchain.	7
7	Access controls	Users can control who accesses their data and what actions can be performed.	Allows data owners to manage permissions, enhancing privacy and restricting unauthorised usage.	5
8	Tamper-proof	Blockchain's tamper-proof nature ensures data cannot be modified.	immutability feature provides strong assurance of data integrity, preventing unauthorised modifications, But the Blockchain provides these	3
9	User access to the cloud service provider	Users can access the cloud service provider's platform to interact with the	Providing user access to the cloud service provider's platform ensures seamless interaction with the	7

		blockchain-based data provenance system.	blockchain-based data provenance system, facilitating data management and analysis tasks. This functionality is essential for users to use the system effectively, so priority is high.	

Table 3: Functional requirements

The priority score is on completing the most crucial functionalities addressed first and aligning the most critical systems at the most effective level they need to be.

4.1.2 Nonfunctional requirements

Nonfunctional requirements				
Section	Requirement #	Requirement	Description	Justification
Scalability	1	Data volume	Handle data volumes at leachate point with minimal performance degradation.	Ensures the system can accommodate growing data needs without significant performance impact.
	2	User Concurrency	Support “x” users querying and accessing provenance data. (Justification: Ensures smooth operation even under heavy loads.	Ensures smooth operation even under heavy loads.
Reliability	1	Data integrity	Cryptography: Use algorithms to verify tamper-proof data.	Ensures data remains intact and trustworthy.

	2	Verifiable storage	Data and provenance are stored in a reliable and verifiable storage system, e.g., Blockchain.	Ensures data remains accessible and immutable
	3	Data Validation	Have a mechanism to validate data provenance records	Ensures accuracy and consistency of provenance data.
	4	Transaction assurance	Ensure data provenance records are updated automatically.	Ensures data integrity and consistency in case of errors
	5	Audit Logs	Firm audit logs for all data access, modification, and system events	Ensures accountability, transparency, and compliance auditing
Availability	1	Uptime	Try to achieve maximum uptime.	Ensures continuous accessibility to users.
	2	Fault tolerance	try to have redundant systems to ensure a fault mechanism if there are issues.	Minimises downtime and ensures continuity of service.
	1	Robust User Verification	Users must authenticate themselves using multi-factor authentication (MFA) before allowing access.	Enhances security by adding additional verification layers

Security Measures and Constraints

	2	Data Integrity	Cryptographic hashing: Cryptographic hash functions generate unique fingerprints of data and provenance records to identify tampering.	Ensure anonymity
	3		Immutability of blockchain: Take advantage of the blockchain's inherent tamper-proof nature to ensure the integrity of recorded provenance records.	Ensures the integrity of recorded provenance records.
		Constraints:	Overhead in performance: Encryption and cryptography procedures might cause some overhead, affecting system performance.	Balancing security with performance considerations

	4		The complexity of key management: Secure key management procedures can be complex and require careful execution.	Requires careful execution for effective security.
	5		User experience: Security measures should not significantly impair user experience or usability.	Balancing security with usability.

Tables 4: Non-functional requirements

4.1.3 User requirements

User Requirement			
Section	Requirement	Description	Justification
Data tracking and provenance	Strong comprehensive provenance	The system collects and stores details of the data provenance.	Through provenance tracks the data history (Origin, changes, access, and transfers), fostering transparency, compliance and trust
	Granular provenance	The user should be able to view the provenance at various levels of	Granular provenance enables a detailed ability to inspect data lineage,

		detail, including individual data elements.	offering insights into the evolution auditing anomaly detection
Data integrity and verification	Secure access	The user should have fast and authorised access to the system	Secure, fast access allows users with authorisation to efficiently interact with the system while keeping the data safe
	Access control	The system shall provide access controls to allow users to set boundaries for the viewability of their data	Access control allows users to have specific permission for the safety of the system by restricting access to specific sections of the system
	Data Sharing	The users should be able to share data with other authorised users	Allowing data sharing promotes collaboration; by implementing secure mechanisms for data sharing, users can facilitate collaboration without compromising data integrity or risking unauthorised access.
integration and usability	User interface	A user-friendly and understandable user interface should exist.	User-friendly interface enhances usability by offering a seamless experience for interacting with data and accessing provenance information. Simplifying complex processes, reducing learning curves and

			Improving user experience.
	Reports	The ability to generate reports and analyse the provenance of data	Report generation and provenance analysis support data auditing, monitoring compliance, and benefit performance analysis.

Table 5: User requirements

4.1.4 Testing requirements

Functionality Testing

- Provenance tracking
- Blockchain storage
- User authentication and access control
- Data integrity validation
- Data lineage tracking

Performance testing

- Scalability
- Latency

4.1.5 Compliance and Regulatory Requirements

These requirements depend on the geographic location of where this software is being used. Each country has its requirements for data regulation.

Data Privacy

- General data protection regulation (GDPR), since data can be stored, European countries must follow GDPR: these reference data minimisation, transparency, user content, and the rights to be removed.

Data security

- The General Data Protection Regulation (GDPR) also includes requirements for data security. This means a company must implement technology to protect data from unauthorised access.
- Network information security (NIS): Since ProvChain will be running in the European Union, it can be subject to NIS directives, which are required for managing cyber security.

Other regulations

- Information Organisations for Standards (ISO) 27001: This is an international standard for managing risk to information systems. It allows for the management and monitoring of potential information risks, allowing the ability to patch/remove potential threats.

4.1.6 Users and characteristics:

This idea caters to a diverse range of users with specific needs and interactions with the platform. Here's a breakdown of key user classes:

1. Data Owners and Providers:
    - Characteristics: Individuals or organisations generating and owning data
    - Needs: Secure data storage, manage provenance, track lineage, demonstrate authenticity and ownership, and control access.
    - Priorities: Security, privacy, tamper-proof records, granular access control, audit trails.

2. Data Consumers and Analysts:
    - Characteristics: Researchers, data analysts, and business users working with diverse data sources.
    - Needs: Seamless access to verifiable data, lineage analysis, understanding data history, trust in data integrity.
    - Priorities: Transparency, data visualisation tools, efficient querying and analysis, data reliability.

3. IT Administrators and Security Experts:
    - Characteristics: Technical personnel managing system security and deployment.
    - Needs: Integration with existing infrastructure, robust security, regulation compliance, and monitoring tools.
    - Priorities: Scalability, performance, compatibility, access control, advanced security features.

4. Compliance Officers and Auditors:
    - Characteristics: Regulatory bodies and internal auditors ensure data privacy and regulation compliance.
    - Needs: Access to audit trails, proof of data integrity, control over data access and modification.
    - Priorities: Traceability, auditing tools, tamper-proof records, comprehensive reporting.

5. Cloud Service Providers:
    - Characteristics: Companies offering cloud storage and data management services.
    - Needs: Integration with existing platforms, value-added services, user access management, and resource allocation.
    - Priorities: Scalability, customisation, integration tools, customer support, competitive pricing.

Importance:
- Primary Users: Data Owners, Providers, and Consumers/Analysts drive core functionality and value.
- Important Users: IT Administrators and Security Experts are crucial for smooth deployment and security.
- Supporting Users: Compliance Officers, Auditors, and Cloud Service Providers have specific needs but are less central to core functions.

4.1.7 Software requirements

The software needed to run and apply the blockchain data provenance.

- An operation capable of running a web server to host its cloud
    - Windows 7+
    - Ubuntu 20.0.4+
    - Debian
    - Apache2
- A web browser
    - Edge
    - IE11 (or newer)
    - Firefox 60
    - Chrome
    - Safari

4.1.8 Hardware requirements

refer to the hardware needed to run Owncloud and blockchain applications.

- Operating system
    - Windows @ 7 +
    - Ubuntu@latest+
    - Debian@latest+
- Memory requirements
    - 8GB Minimum
    - 16GB Recommended

4.1.9 Usability requirements

Usability refers to what the user should be able to do when the system is fully implemented.

- The user should be able to create, remove, and modify files on the system
- The user should be able to check the provenance of their data
- The user should be able to prove the integrity of this data
- The user should be able to see who did say “action” to the data

4.1.10 Development environment

There are different programming languages associated with the development of this ideology:

- Javascript
- PHP
- HTML
- CSS

Also, there will be external cloud service providers for this idea:

- Owncloud
- Chainpoint Gateway
- Chainpoint BHN
- Chainpoint Cli

- Chainpoint js
- Nextcloud

4.1.11 Verification

All these requirements will be tested to determine whether the system's security has been met. These results will be presented in this report.

4.1.12 Overall project description

This project is being developed to use blockchain to benefit data provenance and proof of integrity. Blockchain integration enhances these factors, ensuring a decentralised and tamper-proof ledger for reading data provenance.

5 Design

5.1 Main Design

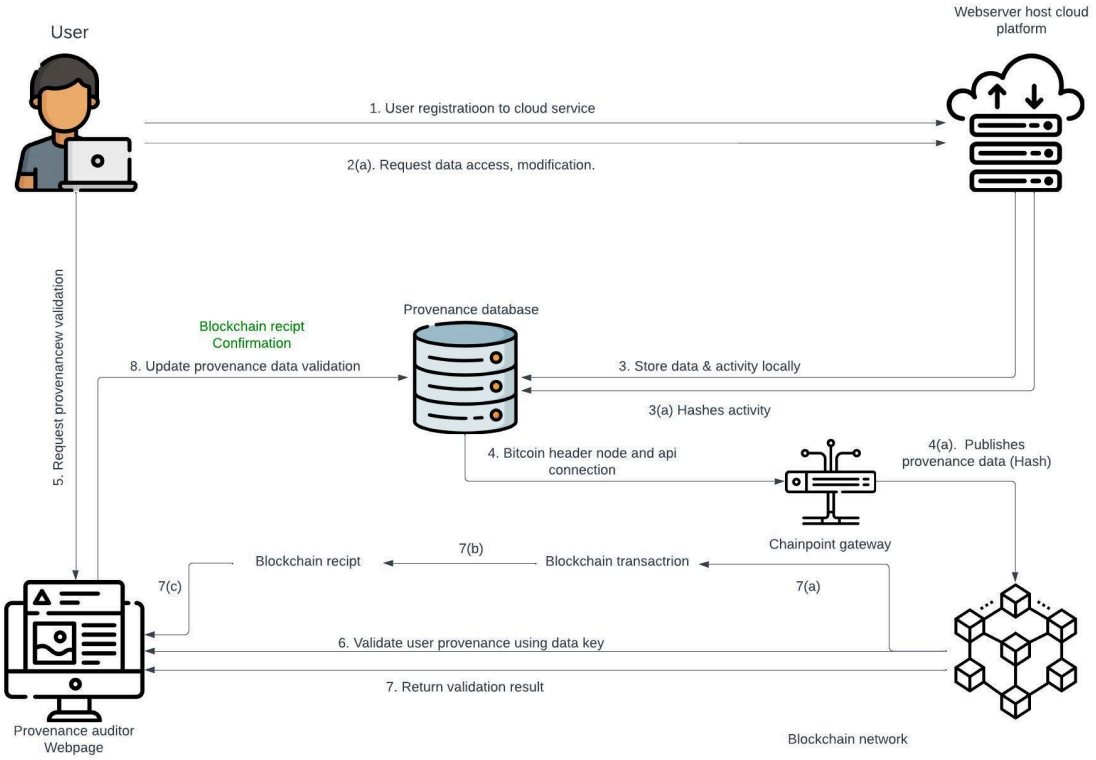


Figure 3: My design

The design has several parts. The design idea relies heavily on Provchaijn; since some parts no longer work, I have swapped them out for newer options provided by Tierion.

- Apache2
- Provenance database (MariaDB: Since Provchain did not explain which one they used)
- Chainpoint gateway
- Chainpoint core
- Chainpoint js
- Chainpoint cli
- Bitcoin header-node
- A provenance auditor script

Owncloud was initially incorporated into the project architecture for data management and cloud storage. Nonetheless, the idea of incorporating alternative/alternating cloud service providers like NextCloud was considered as part of the project's development. This can bring many advantages, such as scalability, performance, and provider interdependence that will be tested later on.

5.2.1 Chainpoint

Due to Tierion no longer providing services related to Provchain, I will be using Tierios' new service, Chainpoint, to develop the blockchain aspect. This service also allows for the submission and proof of blockchain submissions.

Provided by chainpoint, there is a diagram that briefly explains the network of the Chainpoint.

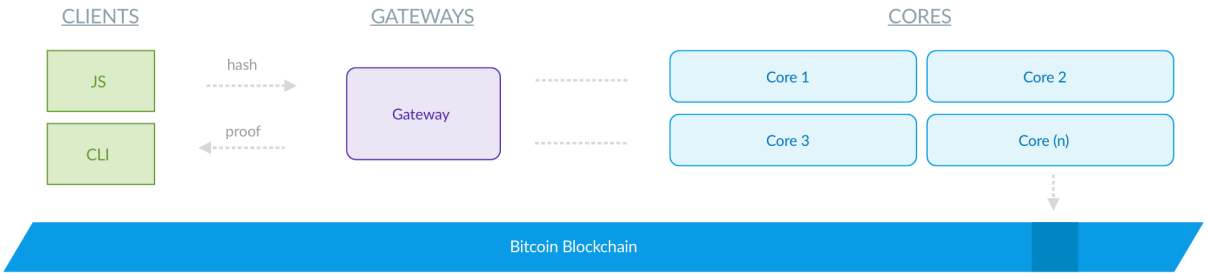


Figure 4: Chainpoint network

Figure 4 shows various components of the chainpoint network. This section of the report will help describe each design aspect with the network idea.

The main components of the network are described using Chainpoint Git Hub for a more in-depth look at the Chainpoint documentation.

5.2.2 Chainpoint-js

A JavaScript client called Chainpoint-js generates and validates Chainpoint proofs. Every cryptographic proof establishes the existence and integrity of data at a particular moment in time.

5.2.3 Chainpoint gateway

A dedicated server called a Chainpoint Gateway generates multiple Chainpoint proofs by sending requests to the Chainpoint Network.

An integrated Liofode running LND is present in every Gateway. When Gateways submit a Merkle root, they employ Lightning Service Authentication Tokens (LSATs) to pay Cores an anchor fee. Two satoshis are the anchor cost by default.

5.2.4 Chainpoint cores

The Chainpoint core node submits hashes from Gateway maintains the chainpoint calendar, and anchors data to the Bitcoin blockchain

Every Core incorporates a Lightning Node and obtains anchor fee payments via Lightning from Gateways, with a two-satoshi default charge. Core operators can modify their fees to remain competitive.

5.2.5 Chainpoint CLI

The Chainpoint CLI permits hash submission to a Chainpoint Gateway. The Gateway accumulates hashes regularly and forwards them to Chainpoint Core for public blockchain anchoring.

The CLI makes it easier to retrieve and validate Chainpoint proofs, which individually cryptographically attest to the presence and integrity of data at a particular point in time.

It maintains a local database tracking submitted hashes and managing Chainpoint proofs locally for effortless retrieval, export, and verification

5.2.6 Overview

The Chainpoint network provides a framework for developing a blockchain network. The development section of this design document will discuss how the utilisation of this network will achieve the project scope.

5.3 Development

Chainpoint has many repositories for applications and the steps to get them running. Once finished, the application will be a background process of the architecture.

5.3.1 Web Applications and Web Auditor

According to the article on the development of Provchain, there is a need for two main web applications and web pages. Due to the existing paper, it makes sense to use the same one they have to get the same result, with the possibility of future additions made to the application.

The first web application will be a cloud service provider known as ‘Owncloud’, shown in Figure 3 as the ‘Webserver hosted cloud platform’. The web server will be run off using apache2. This will all be linked to a ‘MariaDB’ since it will create input in the database for activity, fulfil some of the functional requirements, and be one of the languages used for development.

Also, the web application is completed with PHP, which allows it to run on Apache2 and make it accessible by a web browser.

The overarching user interface will be Owncloud, around 90%, but approximately 10%, depending on privileges, will be the auditor. This means that all user requirements must be met with Owncloud.

OwnCloud will allow for most system factors, such as different user accounts. The MariaDB will then go and hash all activity done on the system using Sha256, fulfilling a function requirement of using cryptographic algorithms to mask the data.

The reason for Sha256 is that Chainpoint needs the string to be around 40-256 characters long. This information will be stored locally on a server since it could lead to potential tampering issues before being uploaded to the blockchain.

This hash will hold a lot of data related to the activity done by the user, such as Time, date, user, activity_type, etc. This relates to another requirement about the overarching subject of data provenance and keeping the integrity of the data.

Then we have the auditor web page; this web page will be constantly updated to provide a Bitcoin address relating to the Merkel trees for the blockchain receipt about that piece of data submission from the blockchain. Allowing the users to verify their proof receipt and get confirmations on that blockchain submission

5.3.2 API

In the system's original design, an API was used to contact the blockchain. The API was given by Tierion; due to the extent of how long it has been since the last iteration of Provchain, there is no API anymore. I must adapt the design to Tierion's new program, Chainpoint.

5.3.3 Chainpoint

Chainpoint is one of the main features of this system; it plays a vital role in passing the data and information, such as hashes, date, time, user, etc., to the blockchain, where we can get a proof ID which will allow us to verify where our data is in the blockchain.

As mentioned in the web application and auditor section, we will use Chainpoint to develop applications; these applications will be used to gain access to the blockchain network to allow for submission and verification of data proof.

5.3.4 Database

There are two different databases needed for this system; the first is a Maria database, which is the backend of the Owncloud web service provider. The database will store the hashes of the activity on Owncloud, which will later be submitted to the blockchain. Using MariaDB also allows for a more easy way to manage the database under a main administrator.

The system will need to track the proof information. Chainpoint will run this database through its CLI. This is so that when the auditor tries to receive the information, it can take that proof and receive it on the blockchain. Every new proof generated after this will also be stored in this database. This database will have security, which is all done by Chainpoint.

5.3.5 Bitcoin Header Node / Blockchain

The BHN (Bitcoin Header Node) allows running a part of a node on the system that connects to multiple nodes and creates a chain of additional headers. This allows them to be

separated from the leading overarching network. The network will be set up as a single node. This is because that is all needed to allow for different peer connections and a connection to start data submission. This connects to the chainpoint-cli and chainpoint-js and allows commands to be executed in the system terminal.

5.3.6 Sawtooth hyperledger

The Sawtooth Hyperledger has an architecture that separates the core system's code from the application. This allows smart contracts to describe application rules without understanding the underlying design.

6 Behaviour diagrams

This is intended to provide some idea of how the system will work.

6.1 Sequence diagram

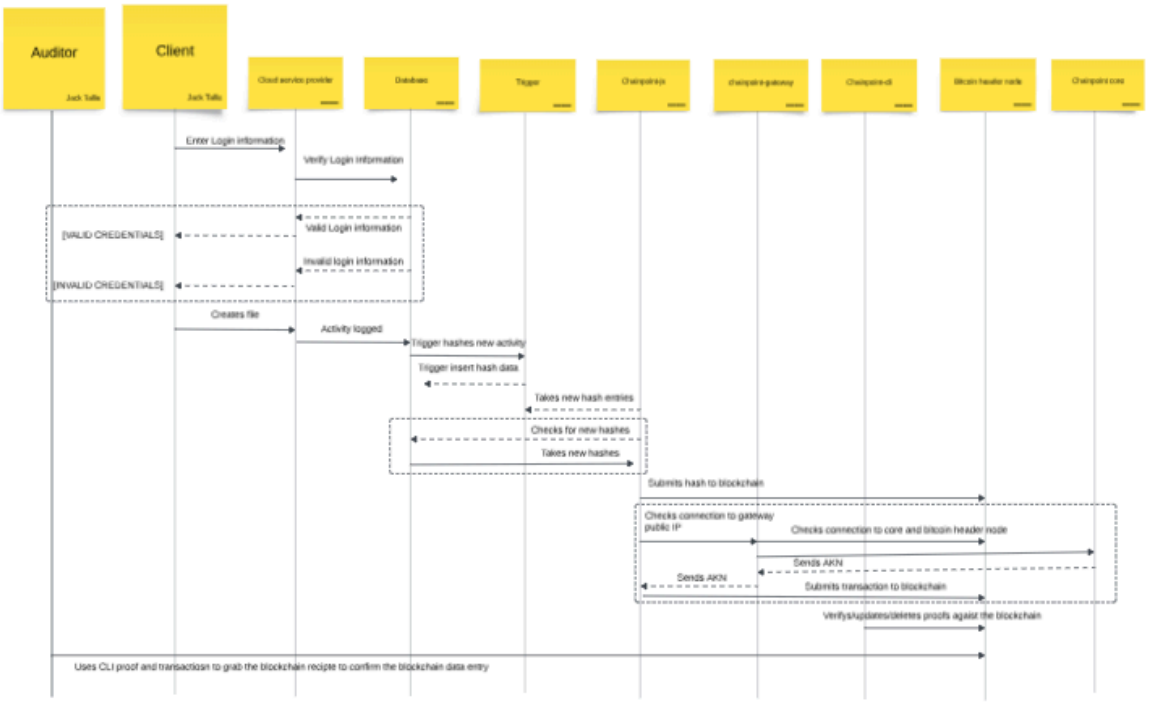


Figure 5: sequence diagram

The sequence diagram in Figure 5 provides an overview of the general appearance of a typical system operation, illustrating the steps a user takes to conduct transactions on the blockchain. For simplicity, the diagram merges several operations—submitting a hash, validating proof, updating evidence, and removing proof—into a single "submission" concept.

The diagram represents how the user interacts with the system. In development, things may change, causing the system's flow to differ from that of the representation. If this occurs, it will be highlighted during the implantation.

Use case diagrams: Use case Diagrams

7 Implementation

The full VM, plus source code, will be provided on GitHub https://github.com/JackTallis.
The repository could be private; please contact me at UP2054361@myport.ac.uk for access.

Overview

This section presents an overview of the system, dealing with development decisions and methodologies and offering context and explanations. Issues encountered during development could include some troubleshooting steps and solutions.

System components

The design document details that the system is built using Apache2, PHP, JavaScript, and MariaDB. This section explains how each component fits together. I will describe each main section's functionality, specific development choices or methodologies observed, and what went well or did not.

Key applications:

The system comprises multiple programs, as documented.

- OwnCloud
- MariaDB
- Chainpoint-js
- Chainpoint-cli
- Chainpoint-core
- Node
- Node.js
- JavaScript
- NextCloud

Each program will be described individually to provide an understanding of the system’s inner workings, followed by an overall description to show how they integrate.

7.1 OwnCloud

OwnCloud is a self-hosted platform for file synchronization and sharing. It allows users to securely store, read, and share data across multiple devices. OwnCloud offers a robust and flexible solution for individuals and organisations seeking a customisable file synchronisation and sharing platform. Its self-hosting capability, data encryption, collaboration tools, and cross-platform compatibility make it a popular choice for those prioritising data privacy and control.

OwnCloud has several key functions that contribute to the overall system, which I will describe at a high level.

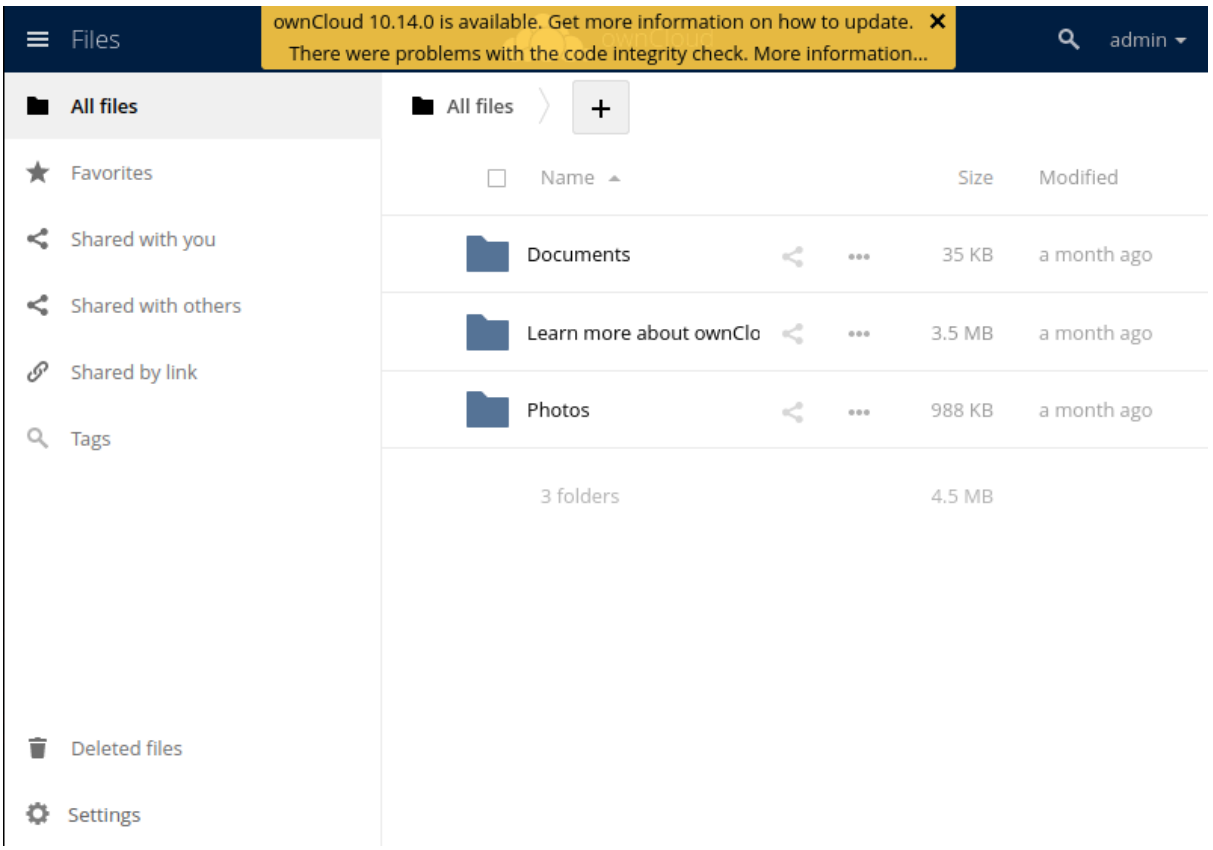


Figure 6: Owncloud cloud platform

7.1.2 Users

Users play a vital role in leveraging blockchain technology. Multiple users will be available on the system and can be allocated privileges. They can also be assigned to multiple admins on the system, letting them do things other than having a main administrator.

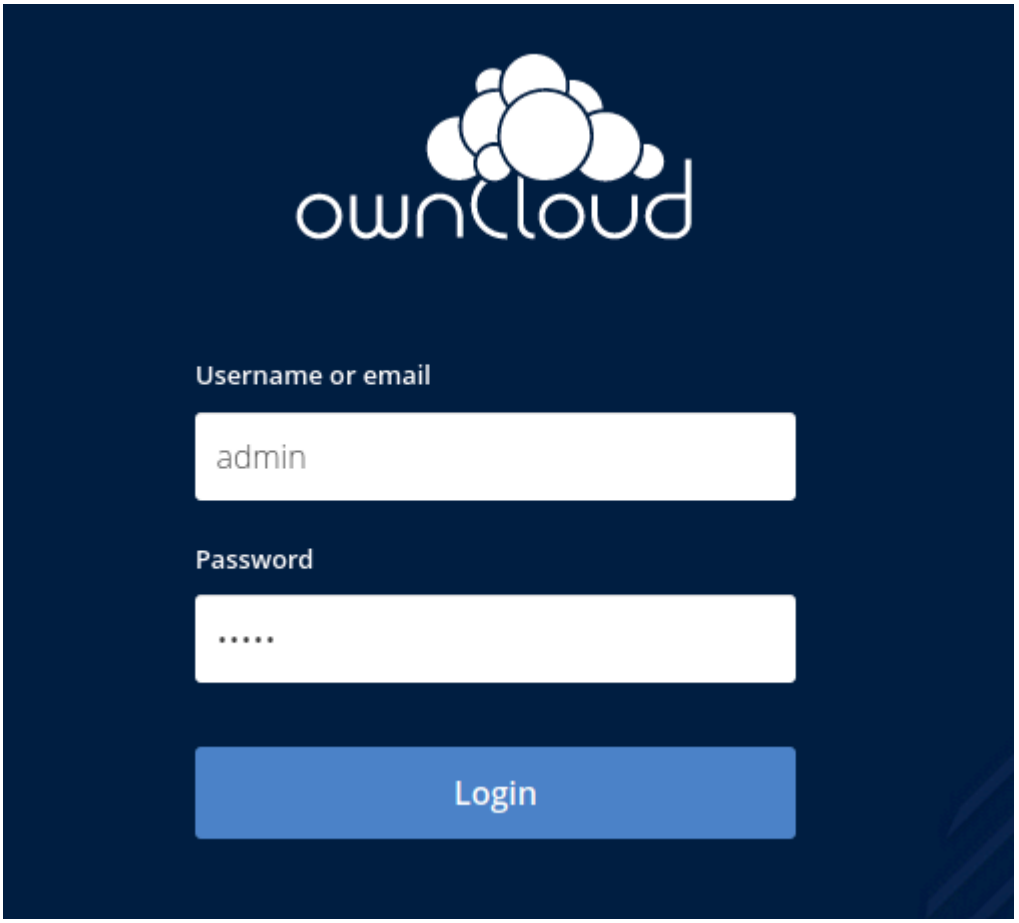


Figure 7: Login page

7.1.3 Create/modify/read function

Creating, modifying, and reading functions are fundamental to our project's implementation, which leverages blockchain technology for data provenance. These functions enable users to interact securely and transparently with data. Managed by smart contracts distributed on the blockchain network, they guarantee immutability, integrity, and transparency throughout the data lifecycle.

7.1.4 Activity plugin

The activity plugin is crucial to our system as it facilitates the storage of information related to system activity. This plugin enhances the system's provenance by providing an activity section within OwnCloud, offering users a basic overview of actions taken on the system.

7.1.5 Hash trigger

In the database, a trigger function automatically executes in response to a specified event or condition. This ensures that everything is hashed after an activity is completed.

7.2 apache2

The cloud service will be locally hosted, allowing users to access the webpage from the same network. Port forwarding has been considered but wasn't possible due to some restrictions.

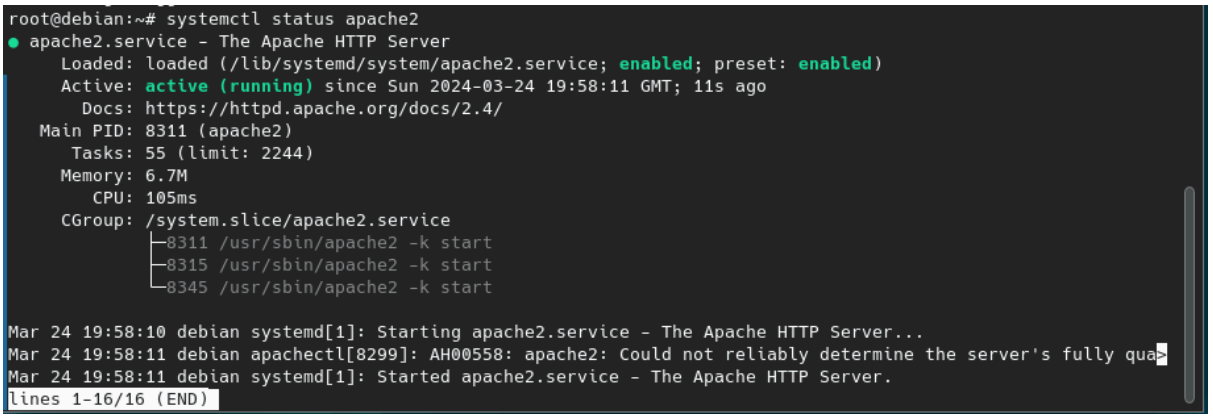


Figure 8: apache2 service running

7.3 Home page

When you first visit the website, they reach the OwnCloud login screen (Figure 9). The login page only allows authorised users to log in, and there is no other information besides a password change section. If the user is not authenticated, they must contact the system's admin to gain an authorised account.

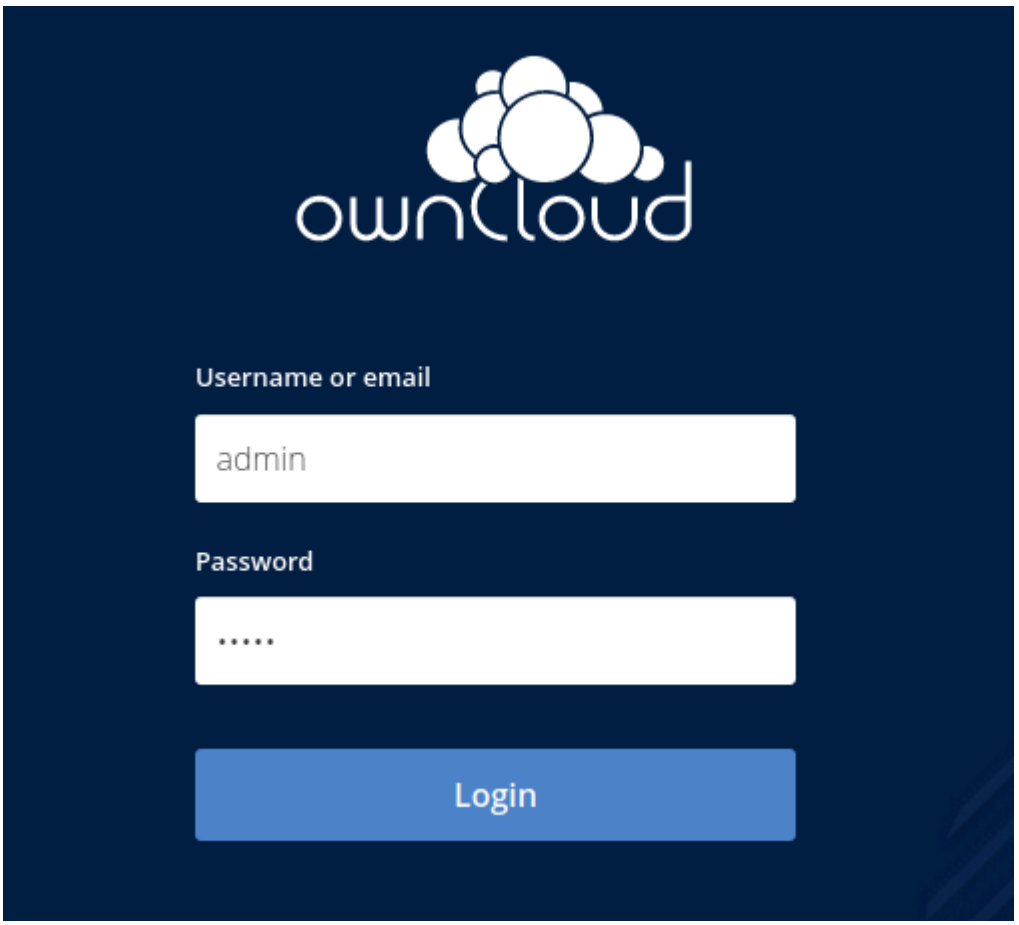


Figure 9: Login page

7.4 Applications needed

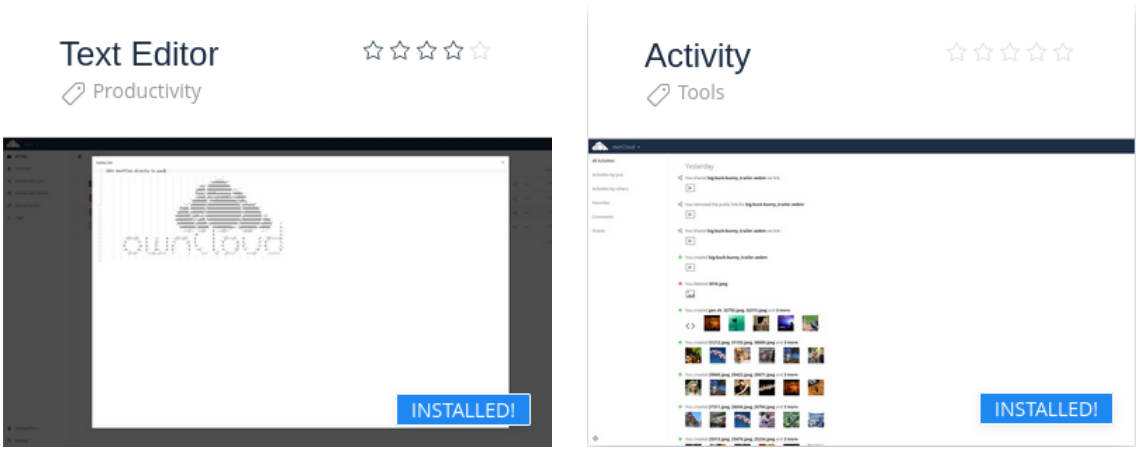


Figure 10: Activity application and text editor application

These applications will allow two major scenarios for the system; the text editor application allows the users on the system to create text files on the system where they can write and share them with other people (Figure 11)

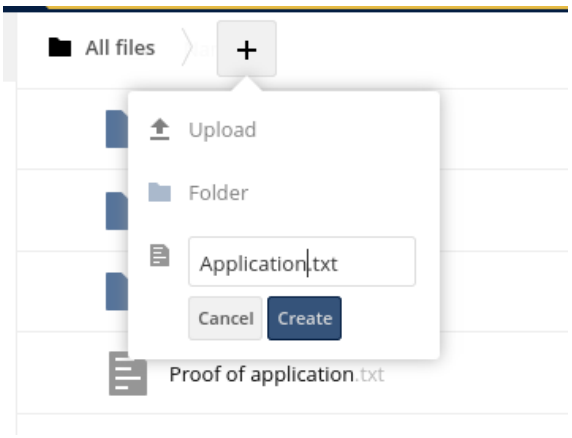


Figure 11: Text editor application

The other application was the activity one; this application allows the admin to see what has been done to the system in a basic view; it will display simple information shown in (Figure 12)

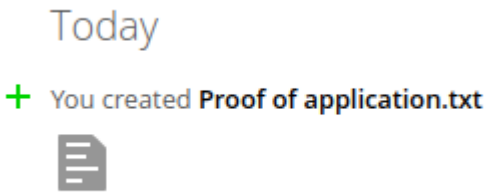


Figure 12: Activity tab

7.5 New Cloud Service

Introduce interchangeable parts to enhance product adaptability. In this project, a new cloud service provider called Nextcloud has been implemented to explore differences in the overall product implementation.

7.5.1 Backend database

The backend database serves as the central repository for OwnCloud's data, facilitating efficient storage, retrieval, and manipulation of information. It includes user data configurations essential for file synchronisation and sharing features, managing files, user accounts, permissions, and system settings.

OwnCloud employs ORM (Object-Relational Mapping) to generate optimised SQL queries, enhancing data access efficiency.

Indexing optimises database performance, particularly for frequently queried columns like user_id, files_id, and timestamps.

The database implements security measures such as encryption of sensitive data, secure storage of user credentials through hashed passwords, and stringent access controls.

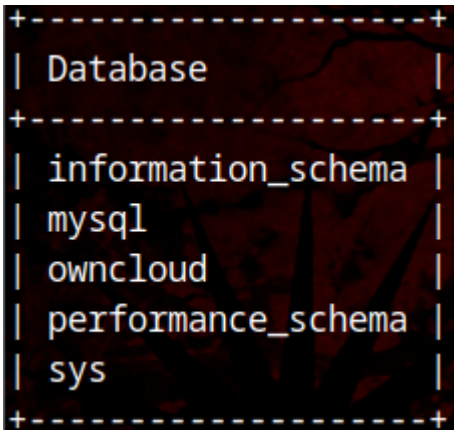


Figure 13: Owncloud database

Figure 21 shows the database table for OwnCloud. This database will store all information relating to the OwnCloud service.

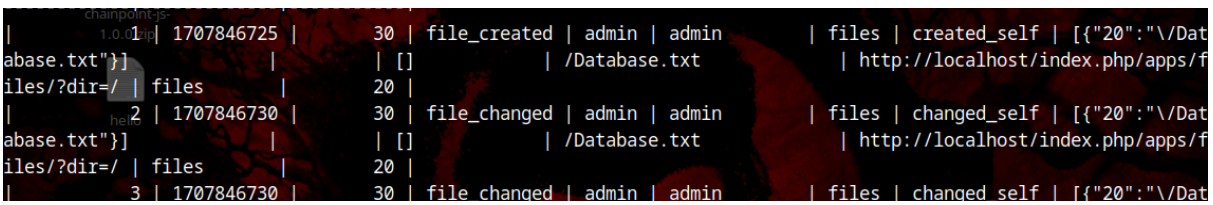


Figure 14: Activity application storage

There is a trigger in the database. When data is inserted into the activity section, this trigger takes all the information related to that insert, such as the user, time stamp, action, etc., and hash it all to have all its authentic details relating to this one hash.

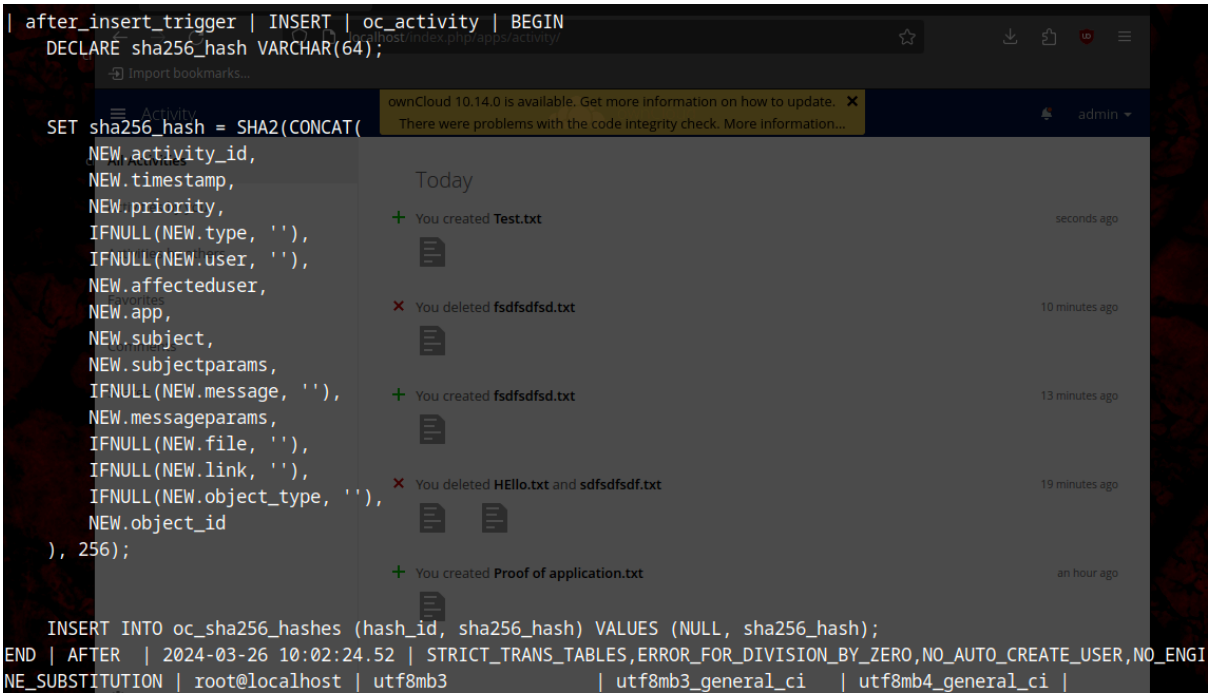


Figure 15: sha256 trigger

During the testing of the trigger, there was an issue where the trigger would be added to the database, but it would scrub the data from the activity tab. I figured out it was an issue with my table for the hashes, so I had to change how the table worked with a rewritten trigger, and it started working (Figure 16).

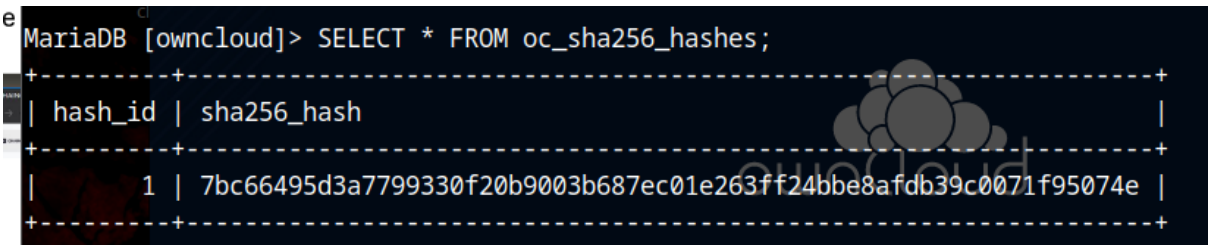


Figure 16: The result of the trigger

This serves as the data storage for these hashes. Later, they will be pulled out using a script, which will then upload them to the blockchain.

7.6 Chainpoint Bitcoin-header node

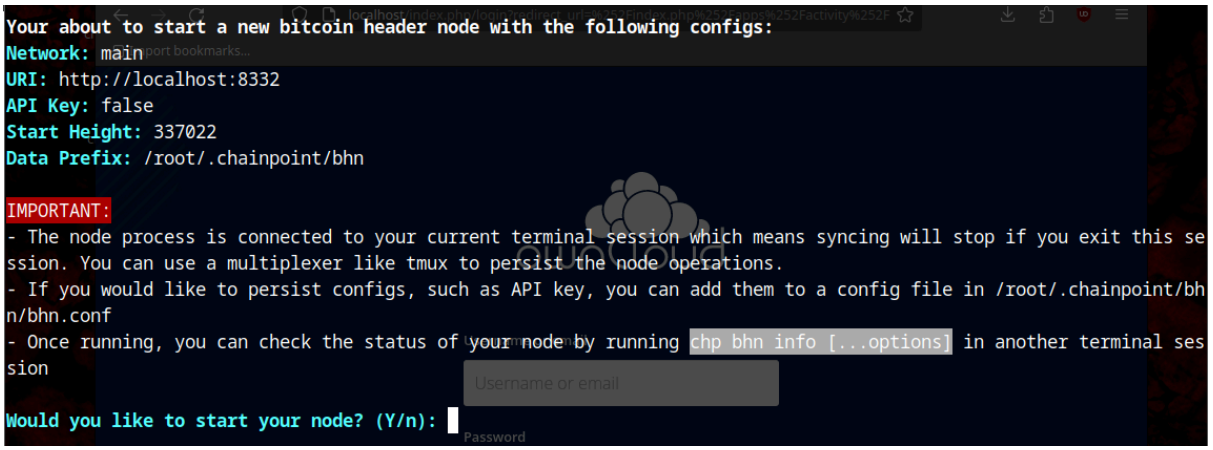


Figure 17: BHN config

The Bitcoin header node has a basic configuration that allows users to allocate specifics to how the node works.

“Network: main” the "main" network refers to Chainpoint's “mainnet”, where most nodes are connected for actual transactions. The "testnet" is used for testing and development purposes.

Mainnet:

- The mainnet refers to the main blockchain network where network participants record and validate actual transactions.

Testnet:

- The testnet is a separate blockchain network that mirrors the mainnet but is used for testing and development.

“URI: http://localhost:8332” is where the blockchain will be viewable (Figure 25). Here, we can see some specific information.

“API Key: false” The documentation does not mention the API but concerns the Bitcoin node from a different company.

The “Start Height: 337022” chainpoint allows for the ability to start at different start heights; in this example, we start at 337022; this also allows the node to sync faster.
“Data Prefix: /root/.chainpoint/bhn” is where the node will be stored.

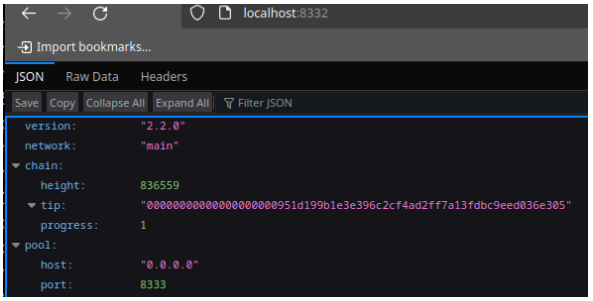


Figure 18: localhost:8332

7.7 Chainpoint gateway

Figure 19: Chainpoint-gateway

The gateway program is accessible only to the server administrator. Upon connection, the admin needs to reconnect it if disconnected. They specify whether to use the mainnet or testnet for the Bitcoin header node and enter the public IP of their node, establishing a gateway between Chainpoint Core and the gateway.

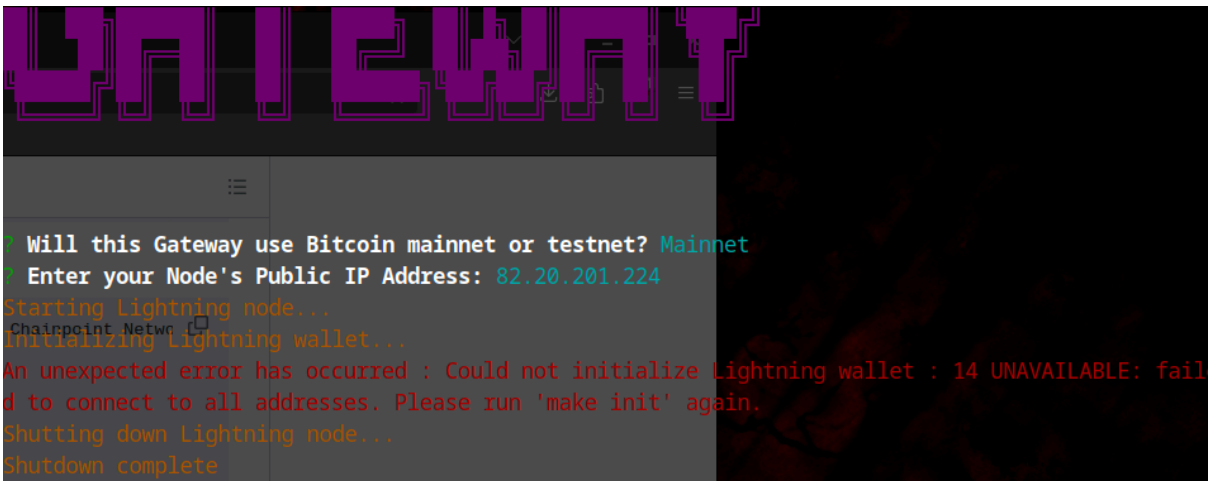


Figure 20: Chainpoint network program

The chainpoint network does not want to connect (Figure 20). I have tried troubleshooting this but cannot find anything to fix it. In Figure 29 below, we can see what is meant to happen.

```
Initializing Lightning wallet...
Create new address for wallet...
Creating Docker secrets...
*****************************************************
Lightning initialization has completed successfully.
*****************************************************
Lightning Wallet Password: kPlIshurrduurSQoXa
LND Wallet Seed: absorb behind drop safe like herp derp celery galaxy wait orie
Lightning Wallet Address:tb1qglvlrlg0velrserjuuy7s4uhrsrhuzwgl8hvgm
*****************************************************
You should back up this information in a secure place.
*****************************************************

TODO: REMOVE Please fund the Lightning Wallet Address above with Bitcoin and wa

How many Cores would you like to connect to? (max 4) 2
Would you like to specify any Core IPs manually? No

You have chosen to connect to 2 Core(s).
You will now need to fund you wallet with a minimum amount of BTC to cover cost

? How many Satoshi to commit to each channel/Core? (min 120000) 500000
? 500000 per channel will require 1000000 Satoshi total funding. Is this OK? (Y
```

Figure 21: Response

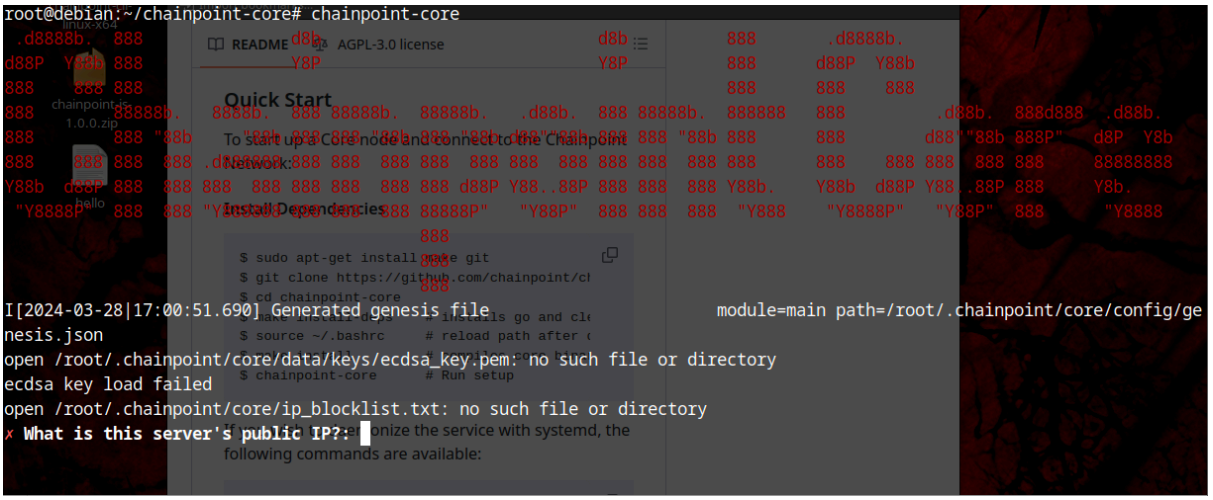


Figure 22: Chainpoint-core

Again, the chainpoint core will be run by the administrator; they will need to add the public IP and the network they will be on. Once it has been developed, the program will start your lightning wallet. The admin must deposit 0.3 BTC to join the BTC network and allow full access. Below in Figure 23 is our lightning wallet information

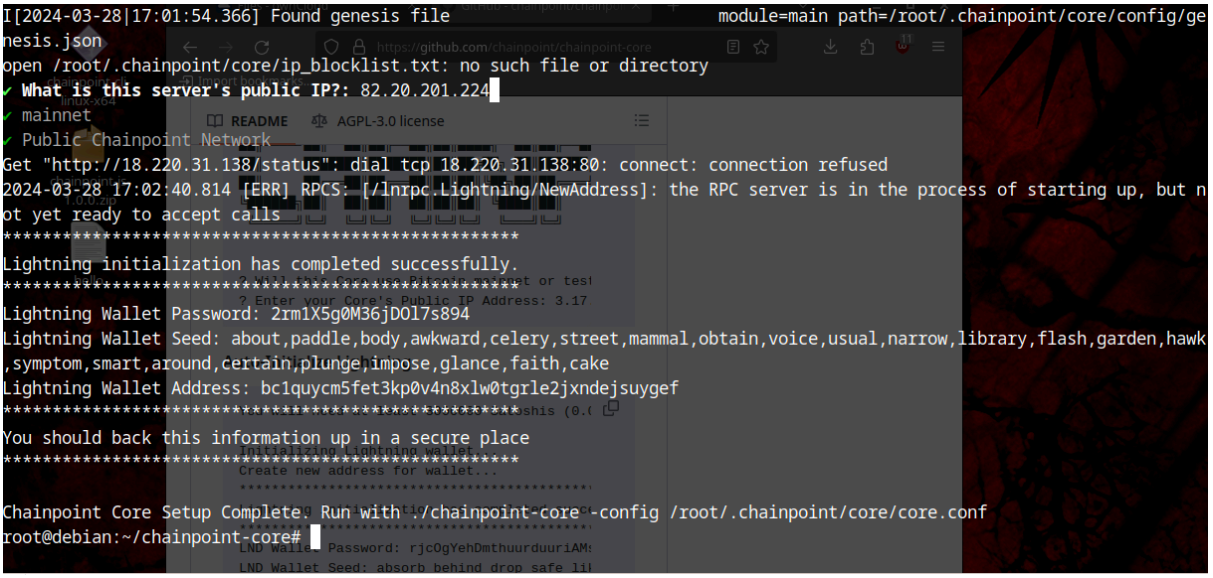


Figure 23: Chainpoint-core lightning wallet

This is the user's wallet. They must finish the setup and send the wallet around 0.3 BTC.

7.8 Chainpoint-cli

Chainpoint-cli is the interface for submitting and receiving transactions from the blockchain; Figure 24 shows the application's commands.

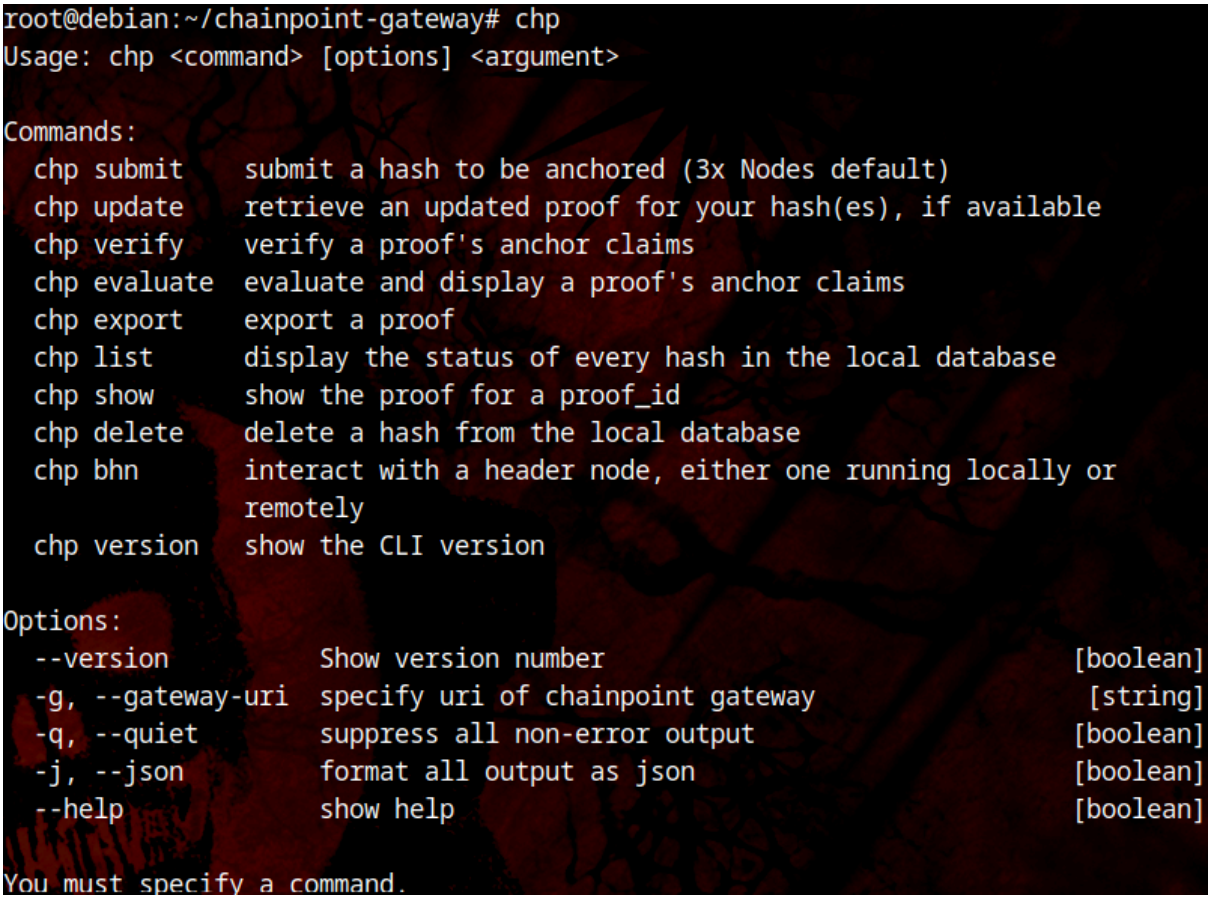


Figure 24: Chainpoint cli commands

Chainpoint-cli refers to the connection to the chainpoint network via a command line interface.

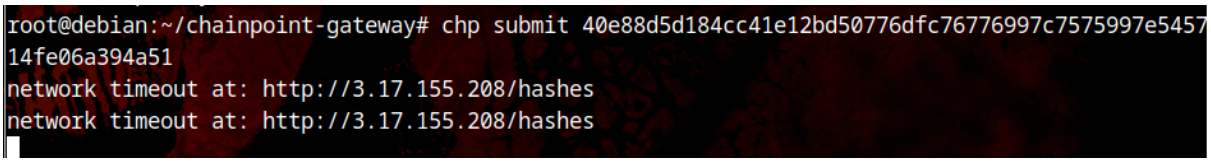


Figure 25: Chainpoint submission

After submitting the hash, it will generate a proof ID for that hash and store it in the proofs database. We can then verify it using the proof using the command “chp verify “proof” “

7.9 Chainpoint-js

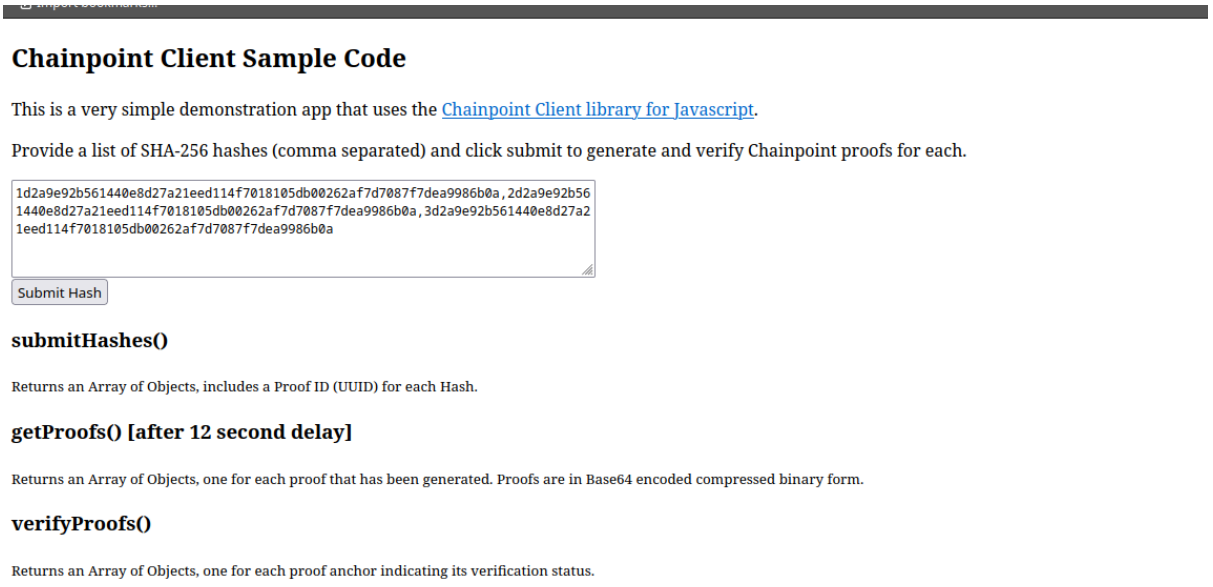


Figure 26: Chainpoint-js

The Chainpoint JS website makes hash administration easier, featuring an intuitive interface that lets users enter multiple hashes. For faster processing, users can enter several hashes separated by commas. After submission, the portal quickly examines the hashes through sophisticated cryptographic techniques, collecting pertinent information from each hash, like line numbers and distinct roof IDs.

The website has an easy-to-use interface and provides a flawless user experience. It has a hash submission input table and dynamically displays the results. While validation checks to guard against input errors, error-handling algorithms guarantee efficient operations.

Chainpoint JS is incredibly performant and scalable. It efficiently handles many hashes to satisfy a wide range of application requirements. It guarantees optimal performance even under high load by prioritising hash-generating speed and accuracy.

Chainpoint JS was built with security in mind and uses robust security mechanisms to preserve the integrity and privacy of supplied hashes and generated data. These precautions include strict access controls, safe data transit, and encryption technology.

7.10 Auditor

For the auditor, there will be 3 different scripts needed: Angular.min.js, jquery.min.js, and my script.js

Figure 27: Angular.min.js

A streamlined version of the Google-maintained AngularJS framework library is called "angular.min.js." AngularJS is a free JavaScript framework for creating dynamic websites. The file has been minified, which reduces its size and improves production efficiency by eliminating extraneous whitespace and characters, indicated by the '.min.js' extension.

AngularJS can create single-page applications (SPAs) by adding more properties to HTML and including advanced features like dependency injection, dynamic data binding, and more. To take advantage of the AngularJS framework's features and capabilities, developers include its fundamental functionality in their web applications through the 'angular.min.js' file.

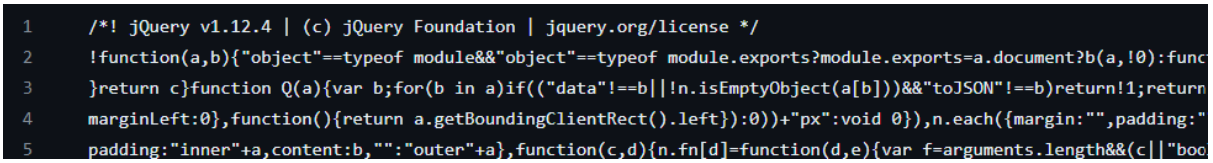
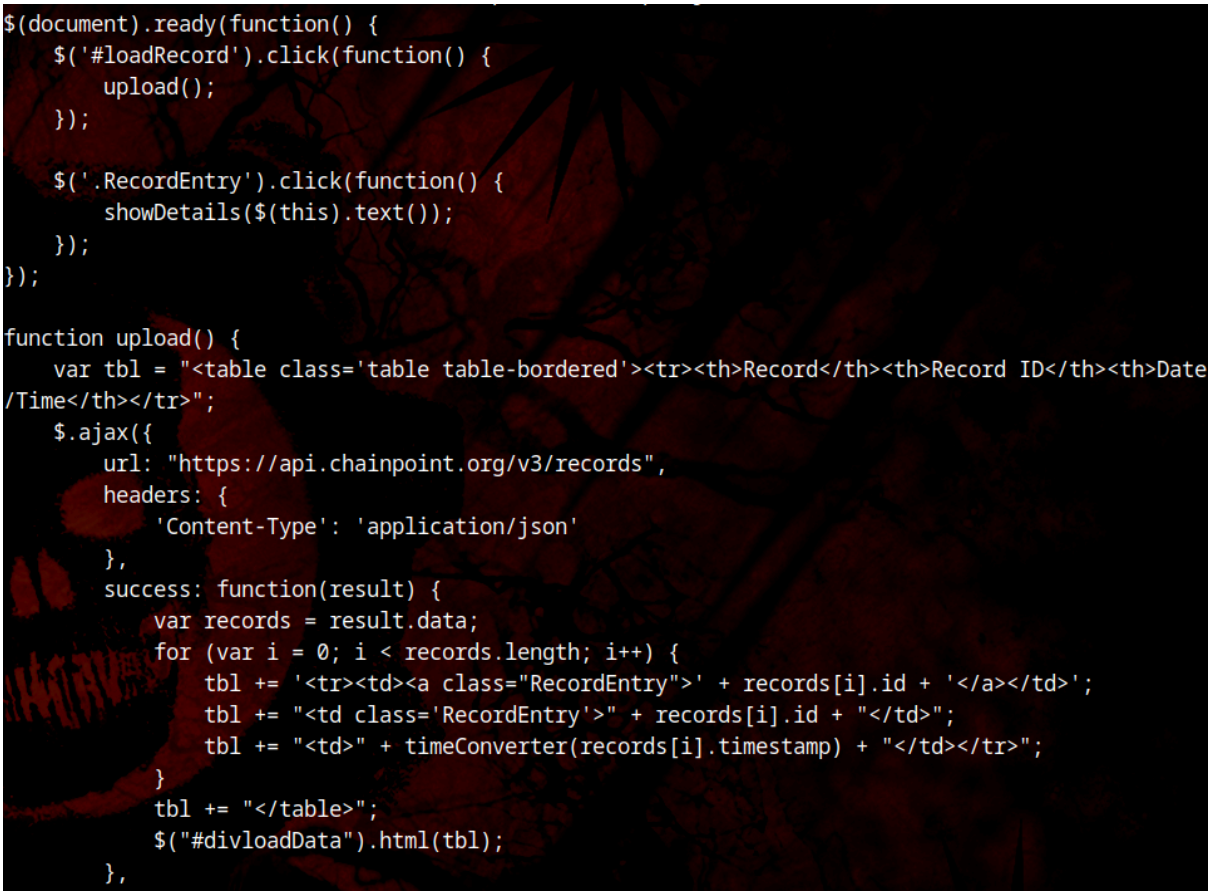


Figure 28: jQuery.min.js

A compressed version of the popular JavaScript library jQuery, which makes client-side HTML programming easier, is called "jQuery.min.js." By simplifying event handling, animation, Ajax interactions, and HTML document navigation and manipulation, jQuery speeds up web development.

The extension '.min.js' signifies minification, which compresses a file by eliminating unnecessary whitespace, comments, and other unnecessary characters to improve production efficiency and minimise file size.

'jQuery.min.js' is integrated by developers into online applications to take advantage of cross-browser compatibility and necessary utility functions that simplify routine JavaScript programming activities. JQuery has been a mainstay of online development for many years, bolstered by a sizable developer community.



```
$(document).ready(function() {
    $('#loadRecord').click(function() {
        upload();
    });

    $('.RecordEntry').click(function() {
        showDetails($(this).text());
    });
});

function upload() {
    var tbl = "<table class='table table-bordered'><tr><th>Record</th><th>Record ID</th><th>Date
/Time</th></tr>";
    $.ajax({
        url: "https://api.chainpoint.org/v3/records",
        headers: {
            'Content-Type': 'application/json'
        },
        success: function(result) {
            var records = result.data;
            for (var i = 0; i < records.length; i++) {
                tbl += '<tr><td><a class="RecordEntry">' + records[i].id + '</a></td>';
                tbl += "<td class='RecordEntry'>" + records[i].id + "</td>";
                tbl += "<td>" + timeConverter(records[i].timestamp) + "</td></tr>";
            }
            tbl += "</table>";
            $("#divloadData").html(tbl);
        },
```

Figure 29: scripts.js

8 Verification and validation

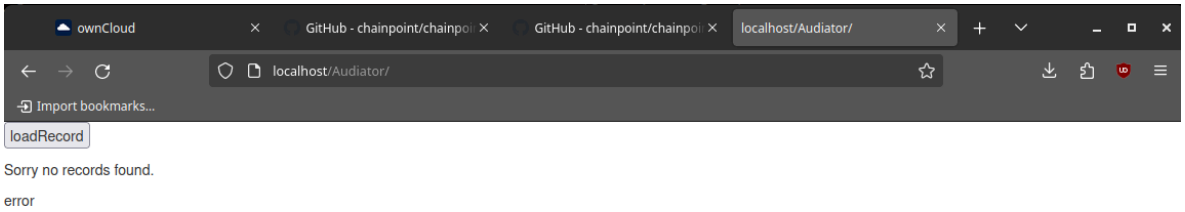
This section of the document describes all of the results from the architecture's testing; it will follow all of the requirements outlined in the SRS document.

8.1 Testing against Requirements

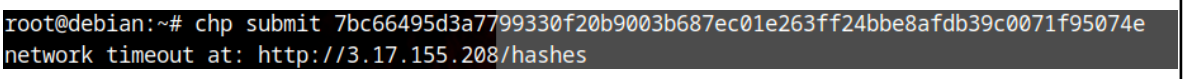
Testing will be done on each criteria separately to ensure they are all met. While some requirements only need one test, others might need several. There will be a PASS or FAIL result for each test. A test will be corrected and documented with it if it fails but can be quickly repaired. These tests will be used as supporting documentation in this evaluation section.

8.1.1 Testing Against Functional Requirements

Requirement 1 - Data provenance tracking on blockchain

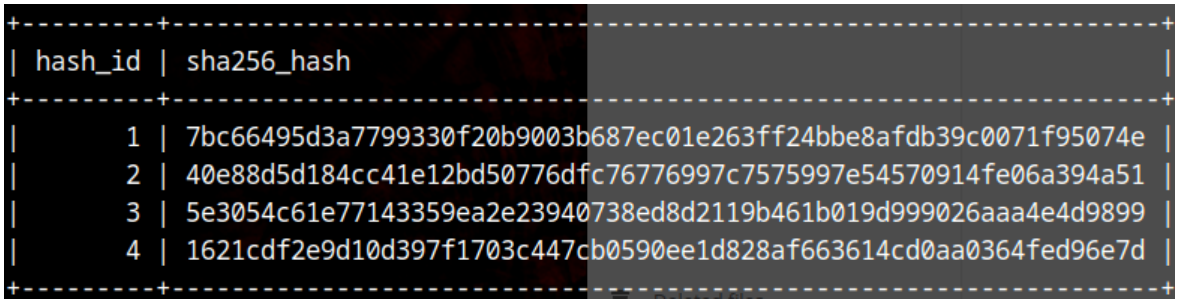
<table>
<tr><td>Test: 1.1</td><td colspan="2">Testing against Requirement 1</td></tr>
<tr><td colspan="3">Description: Tracks the origin and data throughout this lifecycle: “Creation, modification, deletion, access”.</td></tr>
<tr><td colspan="2">Pass condition:
The user should be able to access the data that has been uploaded to the blockchain via the javascript webpage or through chainpoint cli</td><td>Fail condition:
Users cannot access their provenance data on the blockchain or chainpoint cli.</td></tr>
<tr><td colspan="3">Evidence:
 
</td></tr>
</table>

Due to issues with the overarching network, the ability to submit to the network was a failure; the system could not audit the records stored on the blockchain.	
Test type: Tracking	**Result:** Fail

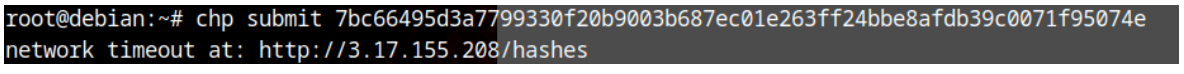
Test: 1.2	**Testing against Requirement 1**
Description: This system tracks the origin and data throughout this lifecycle: “Creation, modification, deletion, access.”	
Pass condition: The user should be able to verify their proof of ID	**Fail condition:** The user cannot verify their proof ID
Evidence:  Test 2 shows the verification failure since the user cannot submit to the network, meaning they will never get a proof ID to verify. This is because chainpoint is no longer operational.	
Test type: Tracking	**Result:** Fail

The test findings show that the system has problems connecting to the blockchain network, preventing data submission. These mistakes make it difficult to accurately track the provenance of data at every point of its lifecycle, including creation, modification, deletion, and access. The lack of Chainpoint services and problems with network connectivity prevent users from accessing or verifying their data. The second iteration will investigate a possible fix that uses the Sawtooth Hyperledger blockchain.

Requirement 2: Data Provenance Collection

Test 2.1	**Testing against requirement 2**
Description: Automatically collects provenance data about data stored in this cloud environment: “Time and date of creation, who created it, and modification made to it”.	
Pass condition: The system should automatically hash data stored in the activity section	**Fail condition:** The system does not automatically hash that data.
Evidence:  Using a trigger, the system automatically hashes all the data stored in the oc_activity table.	
Test type: Collection	**Test result:** Pass

```
+---------+------------------------------------------------------------------+
| hash_id | sha256_hash                                                      |
+---------+------------------------------------------------------------------+
|       1 | 7bc66495d3a7799330f20b9003b687ec01e263ff24bbe8afdb39c0071f95074e |
|       2 | 40e88d5d184cc41e12bd50776dfc76776997c7575997e54570914fe06a394a51 |
|       3 | 5e3054c61e77143359ea2e23940738ed8d2119b461b019d999026aaa4e4d9899 |
|       4 | 1621cdf2e9d10d397f1703c447cb0590ee1d828af663614cd0aa0364fed96e7d |
+---------+------------------------------------------------------------------+
```

Test 2.2	**Testing against requirement 2**
Description: Automatically collects provenance data about data stored in this cloud environment: “Time and date of creation, who created it, and modification made to it”.	
Pass criteria: The system should allow submission using the chp command line.	**Fail criteria:** The user cannot verify that their data is being collected
Evidence:  The system cannot connect to the blockchain, so submitting it via the command line is impossible.	
Test type: Collection	**Test result:** Fail

```
root@debian:~# chp submit 7bc66495d3a7799330f20b9003b687ec01e263ff24bbe8afdb39c0071f95074e
network timeout at: http://3.17.155.208/hashes
```

Test 2.2	**Testing against requirement 2**

Description: Automatically collects provenance data about data stored in this cloud environment: “Time and date of creation, who created it, and modification made to it”.	
Pass condition: The user should be able to connect to the chainpoint gateway	**Fail condition:** The user cannot verify that their data is being collected
Evidence:  The gateway is one of the key communicators between the BHN and the submission; since it no longer functions, I cannot collect the provenance data.	
Test type: Collection	**Test result:** Fail

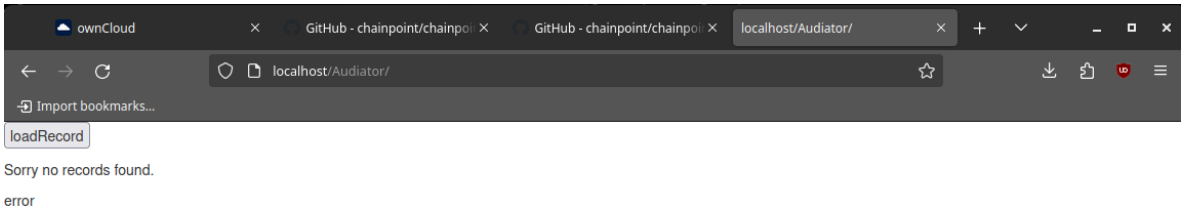
The test results for requirement 2 are mixed. Test 2.1 demonstrates efficient data collecting via automating data hashing. However, Tests 2.2 and 2.3 encounter severe difficulties because the system cannot connect with the Chainpoint gateway or the blockchain. This lack of connectivity hampered users' ability to input data via the command-line interface and receive data verifications.

Requirement 3: Data Provenance Storage

Test 3.1	**Testing against requirement 3**
Description: The provenance data is securely stored on the blockchain; this distribution ledger is tamper-proof and provides a permanent record of the data's history.	
Pass condition: The user should be able to store their data on the blockchain	**Fail condition:** The user cannot store their data on the blockchain
Evidence: In a previous test, I discussed the issues with the gateway not allowing communication between the network, which caused issues with the submission process and prevented storage. However, there is provenance storage in the Maria database and provenance storage in the activity tab on OwnCloud.	
Test type: Storage	**Test result:** Fail

Due to gateway connectivity issues that prevent the network and submission process from communicating, the system needs help securely storing provenance data on the blockchain. As a result, users cannot preserve data on the blockchain, and the system does not satisfy the requirements for a safe and permanent record of data history. Still, the system provides other alternatives for provenance storage on the OwnCloud and MariaDB activity tabs. These fallback options maintain the data's provenance but also need the blockchain's extra security and durability.

Requirement 4: Data Provenance Validation

Test 4.1	**Testing against requirement 4**
Description: The user can check the authenticity of the data by using the blockchain to check its provenance.	
Pass condition: The user should receive a confirmation on their blockchain receipt	**Fail condition:** The user does not receive a recipe for their blockchain submission
Evidence:  The user can access the webpage where the data will be stored, but no records can be loaded because the user is unable to submit.	
Test type: Validation	**Test result:** Fail

Due to data submission limits, the system cannot use blockchain to confirm data's provenance. Users can still access the audits homepage, but records cannot be loaded and confirmed until successful input is received. Because of this, users cannot obtain a blockchain receipt, which restricts their ability to use blockchain provenance checks to verify data validity.

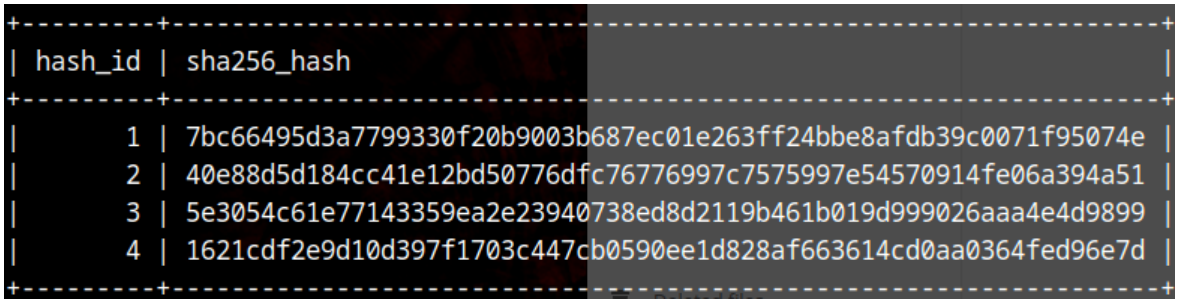
Requirement 5: Cryptography technology

Test 5.1	**Testing against requirement 5**
Description: Cryptography technology protects the user's privacy and ensures data security.	
Pass condition: There should be cryptographic techniques to protect the data submitted to the blockchain	**Fail condition:** There are no cryptographic techniques
Reason: The blockchain is fully encrypted when data is submitted, meaning that all uploaded data will be fully encrypted.	
Test type: Encryption	**Test result:** fails

Encryption technology standards must be met to protect user privacy and data security. The failure of cryptographic approaches reveals potential problems in the accompanying security measures or the data submission process itself, even if blockchain automatically encrypts data upon submission, creating a secure and encrypted ledger. All data uploaded to the blockchain must be protected by encryption to guarantee its integrity and tamper-proofness.

To improve compliance with these criteria, all stages of data handling must consistently employ cryptographic techniques. Encryption rules must be implemented and upheld to ensure user data security and privacy for the duration of the blockchain.

Requirement 6: Data hashing

Test 6.1	**Testing against requirement 6**
Description: The data is hashed before being transmitted to the blockchain.	
Pass condition: The data in the activity log should be hashed	**Fail condition:** Data can not be hashed
Evidence: Data can be hashed using a trigger during the overall process of creating, editing, and removing it. 	
Test type: Hashing	**Test result:** Pass

```
+---------+------------------------------------------------------------------+
| hash_id | sha256_hash                                                      |
+---------+------------------------------------------------------------------+
|       1 | 7bc66495d3a7799330f20b9003b687ec01e263ff24bbe8afdb39c0071f95074e |
|       2 | 40e88d5d184cc41e12bd50776dfc76776997c7575997e54570914fe06a394a51 |
|       3 | 5e3054c61e77143359ea2e23940738ed8d2119b461b019d999026aaa4e4d9899 |
|       4 | 1621cdf2e9d10d397f1703c447cb0590ee1d828af663614cd0aa0364fed96e7d |
+---------+------------------------------------------------------------------+
```

The system does not comply with the encryption technology standards to protect user privacy and secure data. The failure of cryptographic approaches reveals potential problems in the accompanying security measures or the data submission process itself, even if blockchain automatically encrypts data upon submission, creating a secure and encrypted ledger. All data uploaded to the blockchain must be encrypted to guarantee its security and immutability.

Data handling must employ cryptographic techniques consistently to improve compliance with these criteria. Encryption rules must be implemented and upheld to ensure user data security and privacy for the duration of the blockchain.

Requirement 7: Access controls

Test 7.1	**Testing against requirement 7**
Description: Users can control who accesses their data and what actions can be performed.	
Pass condition: Access controls will be in place for access to the main program and the cloud service provider	**Fail condition:** There are no access controls
Reason: No user other than the admin can access the main host server when it is fully developed and the port forwarded. The cloud service provider has different user accounts with different access levels. For example, the Admin can manage what is accessible on the cloud service provider, such as plugins, while regular accounts can just do the basics.	
Test type: Access controls	**Test result:** Pass

Users may ensure access control guidelines are followed by controlling access to their data. Test 7 illustrates the access controls on the main program and the cloud service provider. The administrator is the only person with access to the primary host server, guaranteeing tight control over crucial system operations. Furthermore, the cloud service provider provides several user accounts with different access levels: administrators can manage and administer advanced cloud capabilities like plugins, while average users can access basic features. This tiered access management system protects users' data, and only authorised users can carry out certain tasks depending on their level of access.

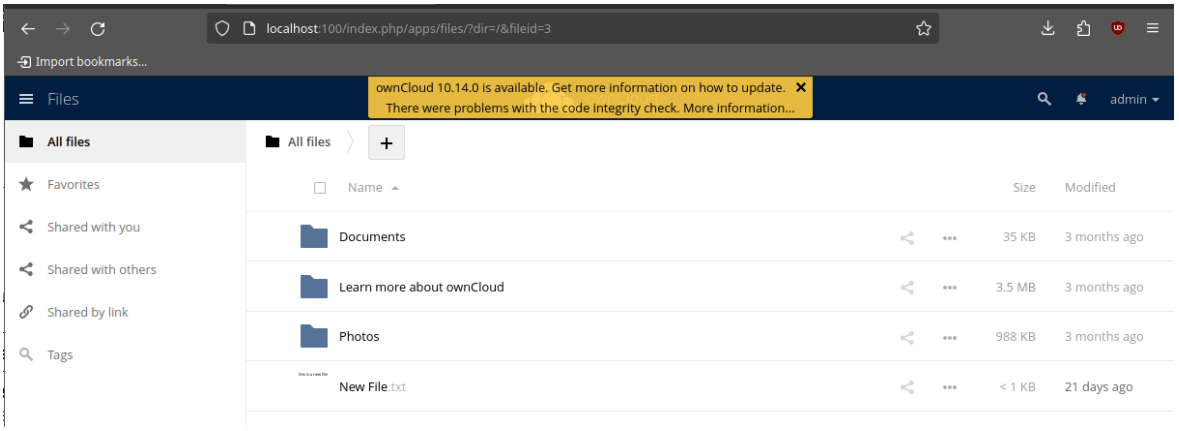
Requirement 8: Tamper-proof

Test 8.1	**Testing against requirement 8**
Description: Having an idea of tamper-proof data to make sure that the data can not be modified when stored.	
Pass condition: The data is tamper-proof	**Fail condition:** The data is not taper-proof
Reason: Fundamentally, the blockchain is tamper-proof. A blockchain-based system for distributed and tamper-proof media transactions (Bhowmik, D., & Feng, T. (2017, August)).	
Test type: Integrity	**Test result:** Pass

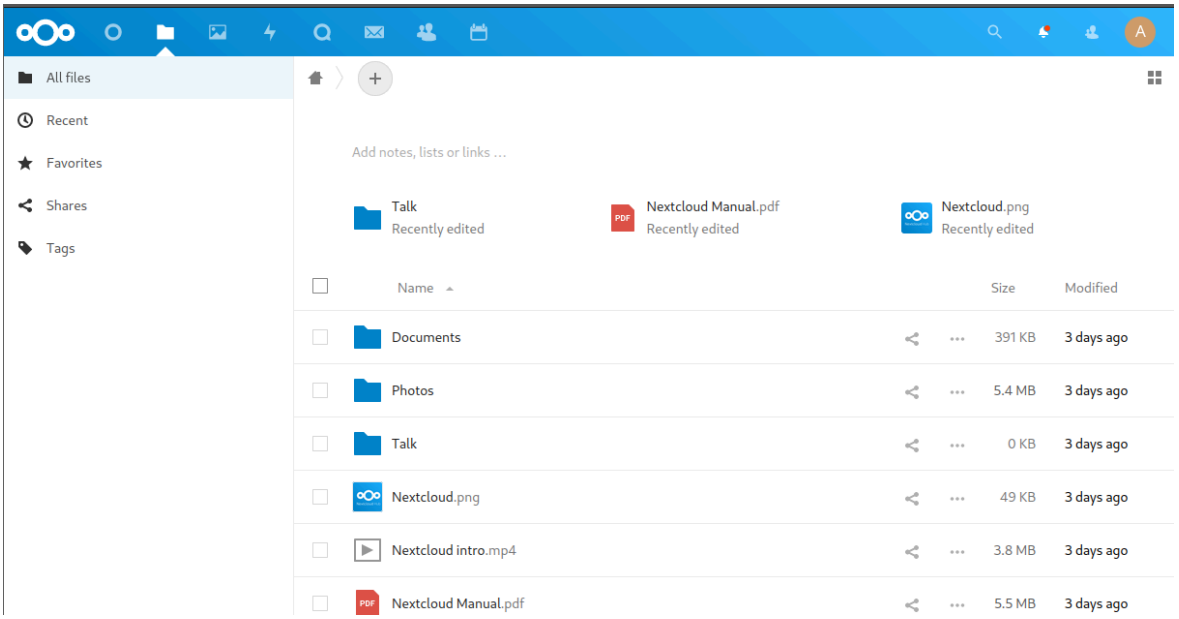
Blockchain's decentralised design and cryptographic hashing techniques provide tamper-proof functioning by default. These features shield data from being altered without anyone noticing once it has been saved. Blockchain is a reliable solution for guaranteeing data integrity because the consensus procedures within the network further strengthen this immutability. Bhowmik and Feng (2017, August) highlight the potential of blockchain technology in media transactions by proposing a blockchain-based system for distributed and tamper-proof transactions.

Additionally, the blockchain's transparency makes it easier to conduct efficient audits and verifications, which boosts trust in the system's ability to keep records that cannot be tampered with.

Requirements 9: User access to cloud service provider

Test 9.1	**Testing against requirement 9**
Description: Users can access the cloud service provider's platform to interact with the blockchain-based data provenance system.	
Pass condition: Users can access the cloud service provider	**Fail condition:** The user cannot access the cloud service provider
Evidence: The OwnCloud service provider is accessible.  	
Test type: Access	**Test result:** Pass

Test 9.2	**Testing against requirement 9**
Description: The users can access the new cloud service provider.	
Pass condition: Users can access the cloud service provider	**Fail condition:** The user cannot access the cloud service provider
Evidence: The NextCloud service provider is accessible	



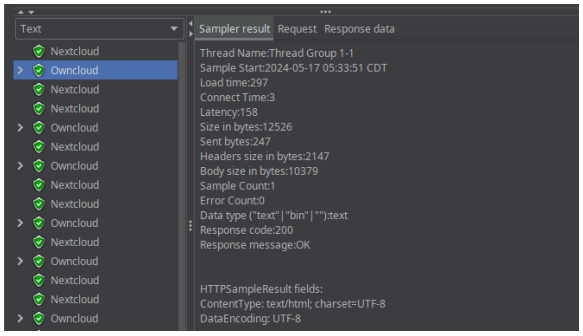
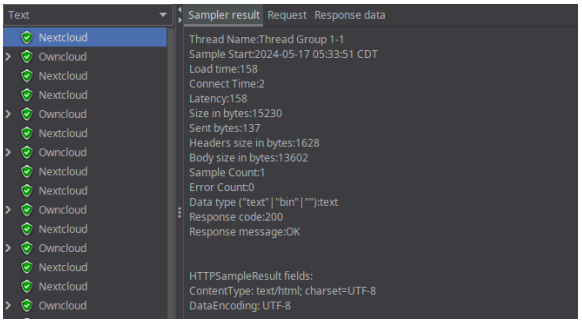
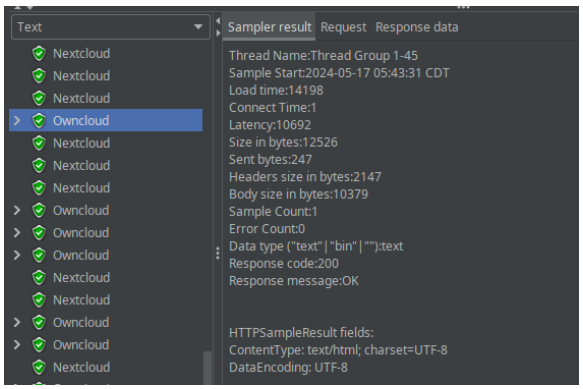
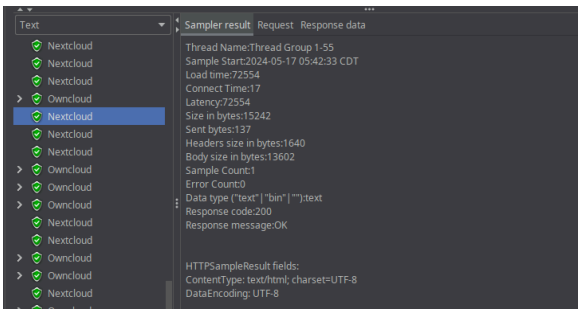
Test type: Access	**Test result:** Pass

Tests 9.1 and 9.2, which validate user access to the cloud service provider, meet the criterion. Users can connect with the blockchain-based data provenance system through the provider's platform. Test 9.2 shows communication with the new NextCloud service provider, while Test 9.1 confirms that the existing OwnCloud service provider is reachable. These tests demonstrate that users can easily interact with the system through new and established cloud service providers.

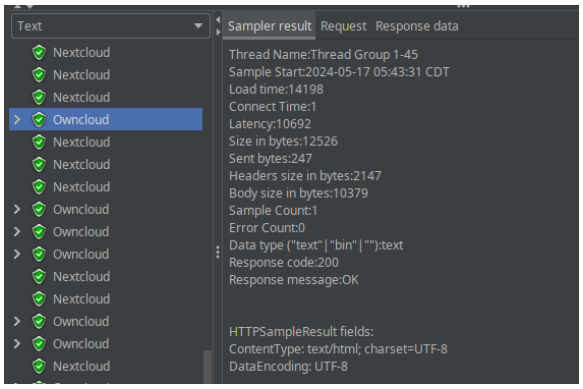
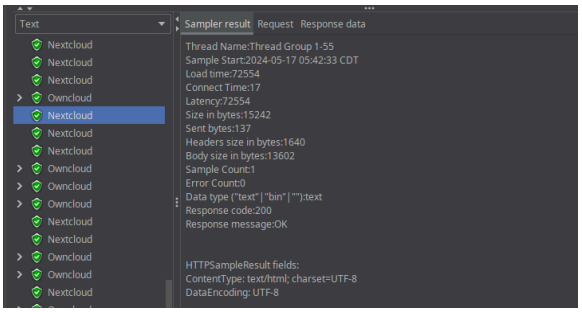
8.1.2 Testing against non-functional requirements

Scalability:

Data Volumes:

<table>
<tr><th>Test 1</th><th>Loading testing</th></tr>
<tr><td colspan="2">Description: Jmetere is used to stimulate a large number of users.</td></tr>
<tr><td>Owncloud:
 
</td><td>Nextcloud:
 
</td></tr>
<tr><td colspan="2">Results: Nextcloud has a better load time with more users.</td></tr>
<tr><th>Test 2</th><th>Performance metrics</th></tr>
<tr><td colspan="2">Description: Using Jmeter to test performance for 100 queries.</td></tr>
<tr><td>Owncloud:
 
</td><td>Nextcloud:
 
</td></tr>
<tr><td colspan="2">Result: Owncloud has better load time under higher concurrent queries.</td></tr>
</table>

User Concurrency:

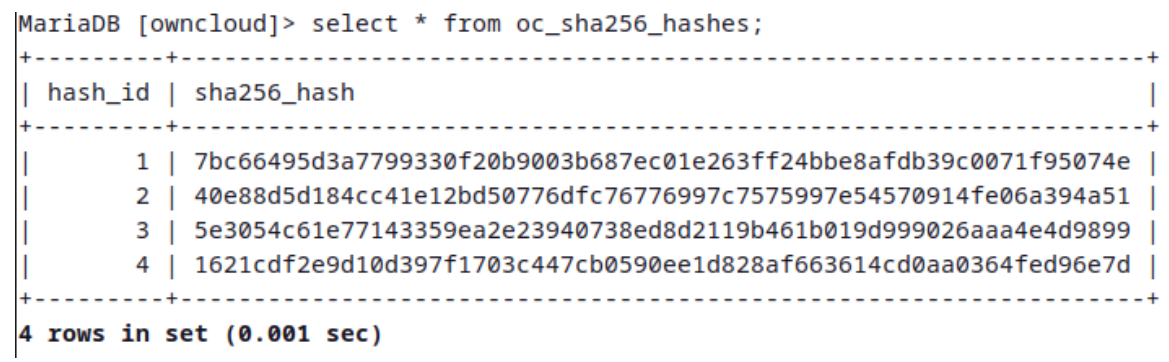
Test 3	**Concurrency testing**
Description: Testing performance of “x” users	
Owncloud: 	Nextcloud: 
Results: Having multiple users use the system shows how NextCloud gradually gets slower	
Test 4	**Scalable Metrics**
Description: Checking performance while under query	

```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
79741 root      20   0 1468072 487852  12860 R  79.5  12.3 102:52.53 node
81809 root      20   0 2918048 630768  21532 S   9.9  15.9   2:51.78 java
48097 mysql     20   0 1550656  89300   8364 S   5.6   2.2   2:34.54 mariadbd
83033 www-data  20   0  286272  44752  34064 R   3.1   1.1   0:00.12 apache2
83034 www-data  20   0  286272  44752  34064 R   3.1   1.1   0:00.12 apache2
83035 www-data  20   0  286272  44752  34064 R   3.1   1.1   0:00.13 apache2
83016 www-data  20   0  306604  67964  52500 R   2.8   1.7   0:00.40 apache2
83018 www-data  20   0  296040  61568  47328 R   2.8   1.5   0:00.32 apache2
83019 www-data  20   0  296320  61804  47456 R   2.8   1.6   0:00.31 apache2
83020 www-data  20   0  306348  62248  47040 R   2.8   1.6   0:00.30 apache2
83021 www-data  20   0  286612  50468  39360 R   2.8   1.3   0:00.27 apache2
83022 www-data  20   0  296816  56548  42424 R   2.8   1.4   0:00.21 apache2
83024 www-data  20   0  284432  46320  37312 R   2.8   1.2   0:00.22 apache2
83026 www-data  20   0  295956  56032  42028 R   2.8   1.4   0:00.21 apache2
83027 www-data  20   0  295808  55928  41940 R   2.8   1.4   0:00.21 apache2
83028 www-data  20   0  295516  54252  40468 R   2.8   1.4   0:00.20 apache2
83029 www-data  20   0  295532  54276  40468 R   2.8   1.4   0:00.20 apache2
```

Result: When under high load, the system starts to spike in usage from the Apache server. If more users are added, the system will eventually crash.
Test 5 | **Stress testing**
Description: Checking system performance when under load
Result: Constantly increasing the concurrent users to around 200 causes the system to crash.

Reliability:

Data integrity:

Test 1	**Crypto verification**
Description: Use a hashing algorithm to make sure that any tampered data has a hash for the system to monitor	
	
Result: The database does this since everything must be hashed before submission. The system always has a hash value for tampered and non-tampered data.	

Verifiable storage:

Test 2	**Blockchain testing**
Description: Test to see if data is stored on the blockchain	
Result: Show in functional requirements that this cannot be completed since there is no way to send it to the blockchain.	

Transaction assurance:

Test 3	**Validation**
Description: Testing blockchain transaction assurance	
Result: The transaction assurance cannot be completed since submission cannot be completed.	

Aduit logs:

Test 4	**Log monitoring**
Description: Have audit logs to monitor the system.	
Result: Show functional requirements; there are audit logs to monitor the systems.	

Availability:

Test 1	**Uptime**
Description: Testing the system uptime	
Result: When testing uptime, I monitored the system for a good amount of time, noticing no difference between when it was used and when it was left unattended.	
Test 2	**Fault tolerance**
Description: Bring the system down under use	
Result: While the system was being used, I shut it down to test the provider's redundancy. This showed me that when the system went down, it tried to save whatever was being done.	

9 Future work / Iteration Two

The primary idea of this iteration is to help fix issues with chainpoint connectivity by suggesting Sawtooth Hyperledger. This blockchain technology offers far greater control over the blockchain code, which can improve overall system performance and reliability.

The main benefits of using this blockchain are:

- Customisability: The hyperledger allows for a high degree of customizability since you are writing the code for the blockchain. This will allow the user to write the Blockchain to whatever they want (Figure 38)
- Localhost: By hosting the blockchain locally, we can avoid network issues at the chainpoint, allowing for better system control.
- Security and privacy, using the sawtooth architecture, allows the user to define the security parameters and access controls, providing a more secure and private network.

9.1 Implementation

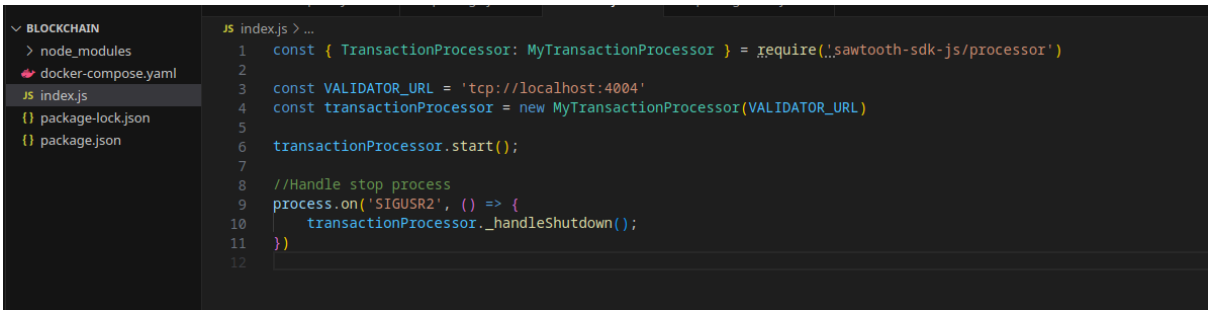


Figure 38: Beginning of the code

Figure 38 shows the beginning of the code and what is possible. This basic code allows the blockchain to be run alone. Still, if added to the overarching architecture, we can see how it will develop and benefit the architecture, and we can create it to make it better since we cannot use Chainpoint.

10 Evaluation

The project builds upon the ProvChain model, which ensures data provenance and integrity using blockchain technology. However, I have introduced interchangeability into various aspects of the model. This flexibility optimises architecture by swapping different components, like the blockchain or cloud service provider. The project seeks to tackle potential performance and scalability concerns by integrating adaptability into the established model. This strategy allows for adjustments based on evolving technology and data needs, ensuring the system's resilience and longevity.

10.1 Evaluation against functional requirements

Requirements will be explained to determine whether the requirement has been met. Any extra pertinent data and the corresponding test results will support this evaluation. If there are situations when the standards are not fulfilled, an examination will be conducted to determine the root causes, including development, planning, or technology issues. In these situations, options or workarounds are used in the current project or might be considered for upcoming system iterations.

Test results and additional data are used in this evaluation to evaluate the project's compliance with each functional requirement. Explanations will be given for needs that were not met, identifying problems resulting from planning or development difficulties as well as any constraints that prevented the criteria from being realised. In these situations, prospective solutions for future iterations, workarounds, or alternative approaches during the current project.

Requirement 1: Data provenance tracking on blockchain

The project encountered numerous challenges when trying to connect to the blockchain network. This hindered the ability to accurately track data provenance throughout the stages of creation, modification, deletion, and access. As a result, users were unable to access or verify their data provenance through the command-line interface. The connectivity issues with the blockchain network, particularly with chainpoint services, were the main contributors to this failure. In the second iteration, considering a different blockchain, such as the Sawtooth Hyperledger blockchain, could potentially offer a better solution.

Requirement 2: Data provenance collection

The data provenance collection was partially successful because the system automatically hashes data stored in the activity section, fulfilling the data collection aspect. However, the connectivity issues with the gateway prevent proper data submission and collection of provenance data. The connectivity issues need to be addressed for successful data submission.

Requirement 3: Data provenance storage

Issues where securely storing provenance data on the blockchain due to gateway network problems. However, alternative storage solutions such as MariaDB, Owncloud activity, and Nextcloud activity have succeeded. While alternative storage options offer a temporary solution, the project focuses on improving blockchain connectivity for data storage security and permanence.

Requirement 4: Data provenance validation

The project was unable to validate data provenance using blockchain due to submission minitations. As a result, users were unable to recieve blockcgain receipes or verify data authenticity through blockchain. An idea of an iteration two section in this report has been suggests for improvements.

Requirement 5: Cryptographic technology

Although encryption was a;ready present, there is room for improvement in cryptographic concepts and measures. Future iterations should try and enhance cryptographic technology through the data process to ensure all encryption standards are met

Requirement 6: Data hashing

The data was successfully hashed before being transmitted to the blockchain, meeting hashing requirements

Requirement 7: Access controls

Access controls were implemented through the cloud service provider and the main programs. Users have different levels of access based on their account. No significant iossue were observed with the access controls; the system effectivly provided them.

Requirement 8: Tamper-proof data

Tamper-proof ideas have been established in the project; the blockchain is naturally tamper-proof. This ensures that immutable data is achieved, but due to connectivity issues, this cannot be 100% successful.

Requirement 9: User access to cloud service provider

Users could access the cloud service provider platform to interact with the blockchain-based data provenance system. Both Owncloud (Original) and Nextcloud (New) services were accessible. No issues were observed in providing user access to these cloud services.

10.2 Evaluation against non-functional requirements

Requirement 1: Scalability

Under current conditions, the project can efficiently manage moderate data volumes and user concurrency. NextCloud handles loads better with many users, whereas OwnCloud performs well under high query loads. However, connectivity faults can cause the system to fail under heavy load. Future network infrastructure enhancements are expected to increase system performance during peak loads, perhaps reducing crashes and boosting overall scalability.

Requirement 2: Reliability

The system meets reliability standards by consistently supplying hashing techniques to protect data integrity. However, there are challenges with the blockchain-based verified storage, which has limited the transaction assurance audit capabilities. Prominently, the issues of sending data to the blockchain have hindered the system from utilising cryptographic technology to ensure data integrity and have system monitoring. The development will focus on supporting the blockchain submission to improve the system's reliability and dependability.

Requirement 3: Availability

The system experienced challenges in achieving uptime due to connectivity issues with the blockchain network. Network disruptions affected continuous accessibility and fault tolerance. However, the system generally maintained an adequate level of availability. Improving communication with the blockchain network is a critical project that will be completed in the future.

Requirement 4: Security measures and constraints

The system encrypted both in-transit and at-rest data. The project used digital signatures and cryptographic hashing for data integrity. However, one possible problem was the complexity of essential management requirements for frequent security assessments. Enhancing key management procedures, conducting frequent security audits, and adding more secure communication protocols are possible improvements.

Requirement 5: Usability

Usability requirements were met. The cloud service provider offered a user-friendly interface for file management and provenance tracking. Users could interact with the cloud service platform efficiently, but challenges with data provenance tracking on the blockchain impacted overall usability. While the interface was excellent, future work could improve the usability of the blockchain interactions for smoother data provenance.

10.3 Development evaluation

The methodology and practices is critically evaluated to determine if the techniques employed in creating the system was acceptable and practical.

10.3.1 Development methodology

A hybrid strategy was used for the project, combining elements of interactive and waterfall methodologies. The goal of this strategy was to combine the flexibility of iterative development with the extensive documentation and organisation of water. This hybrid technique did, however, provide specific difficulties in keeping a distinct project phase because it introduced unforeseen complexity when the planned process was deviated from.

10.3.2 Virtual Machines

The project employed virtual machines (VMs) as its primary development environment. The benefits of using virtual machines (VMs) include the following:

- Separating the development environment.
- Making testing and troubleshooting simpler.
- Providing more flexibility in handling various setups and dependencies.

10.3.3 Version control

practices have been impacted by the following method of using personal machines. The lack of structure branching and strategies merging from something like git flow could have made tracking code changes and reverting to previous versions easier, but since there was a lot of applications used this would not make sense. Still, since I was using virtual machines, I could set restore points if there were issues with the system integration at that moment.

10.3.4 Testing and Quality Assurance

The project conducted numerous tests related to both functional and non-functional requirements. Although the testing strategy was excellent, it is imperative to consider utilising automated testing solutions to improve the effectiveness and dependability of testing. Pair programming or regular code reviews can raise code quality.

10.3.5 adaptability and learning

The project had to adjust and troubleshoot to overcome obstacles, including network problems and technological constraints. To proactively address these difficulties, the development environment would have benefitted from more research and training.

Using virtual machines gave my project a strong base by allowing for isolation and flexibility during the development phase. Nonetheless, there exist avenues for refinement that could augment my methodology even further. Using more organised version control techniques

could increase productivity and simplify code maintenance. My testing procedures would be more dependable and quick if I used automated testing technologies. Peer review and mentorship could also provide insightful criticism and encouragement that would help me improve my code and development processes. I can improve my project's effectiveness and quality by implementing these strategies.

10.4 Project management evaluation

From the outset, it would be simple to assume that the project's completion time estimates were inflated by examining a preliminary Gantt chart utilised for planning.

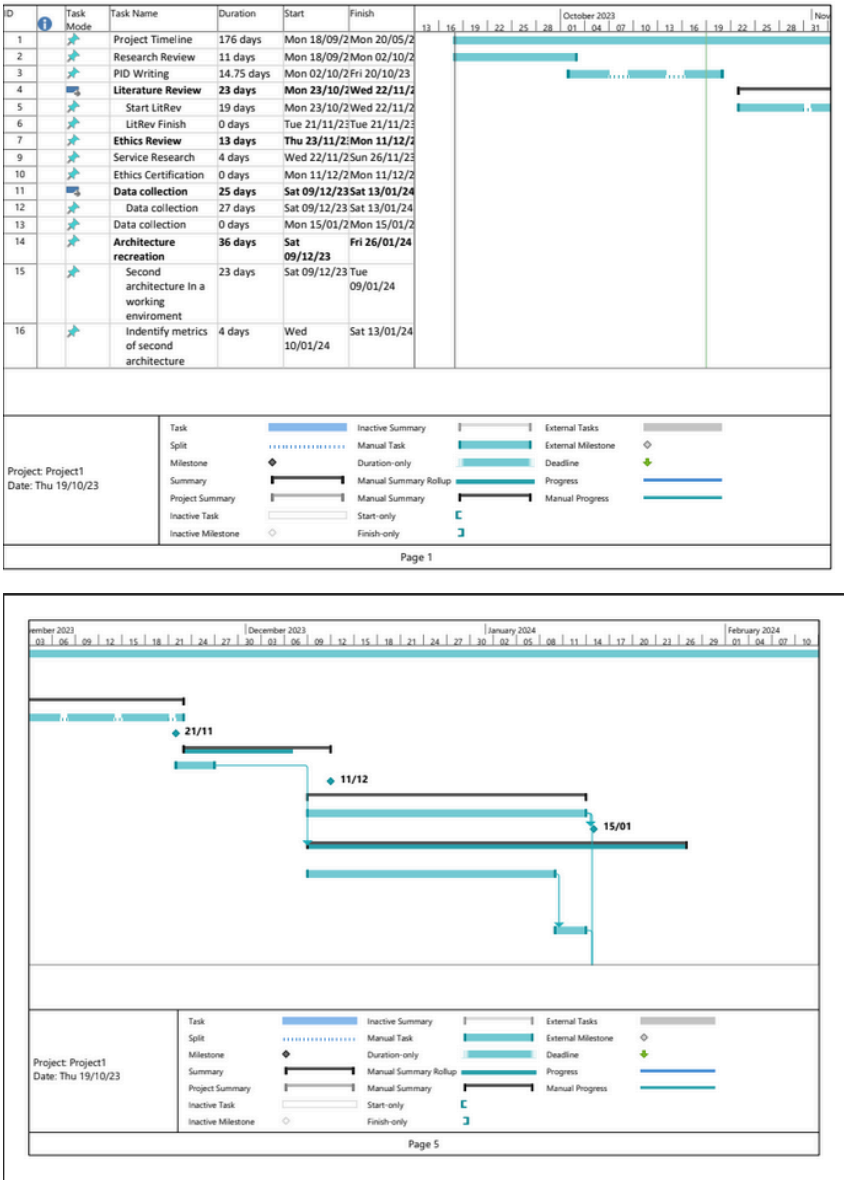


Figure 30: Gannt chart 1

The project was supposed to last from October to April, but progress could have been more steady, given the numerous initial challenges and revisions. The basic notion of testing multiple solutions proved unworkable. An earlier completion would have clarified objectives, allowing for better focus on critical elements like the early construction of Provchain. The

difficulties originated from the need for more accurate knowledge and planning for a project of this magnitude rather than a failure in project management.

Much of the project entailed studying applications like Chainpoint and Bitcoin header nodes, emphasising the importance of a thorough starting understanding. Many of the techniques I had to learn on the fly were distinct from the original Provchain concept, demonstrating the rapid growth of technology. My initial Gantt chart needed to be more ambitious; by February, I had amended it to reflect the updated project scope.

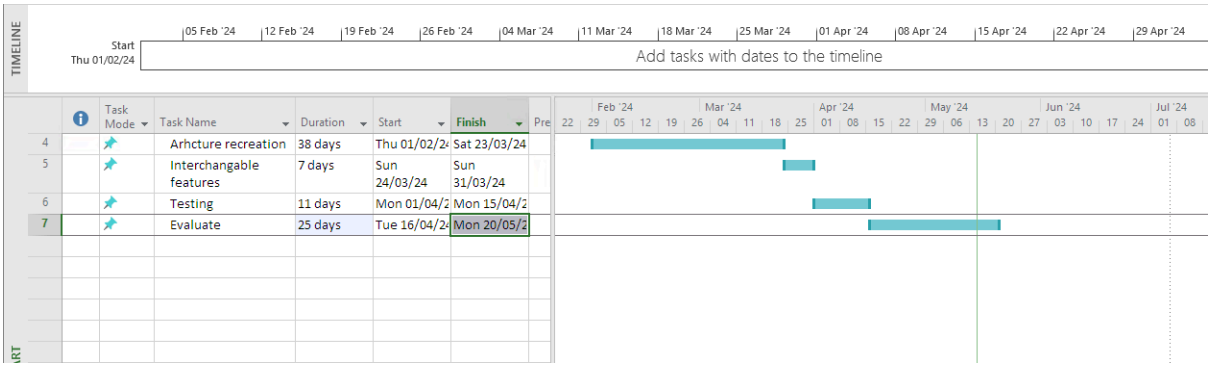


Figure 31: New chart

Although this is a more straightforward chart, it depicts the new idea and how it should be finished within the project's time range; this enabled me to succeed based on the ambition reached within the project's period.

11.5 Methodology

The chosen methodology for this project was the iterative waterfall. It was significantly followed, but I should have considered some aspects. This methodology style allowed me to do a lot with the project, especially when I had to go back and change things. However, I should have spent more time swapping stuff out of the original architecture. Since a lot of the development time was spent on implementing the original design, even though many aspects of it would not work.

10.6 Testing

In the testing phase, the project reached a 50/50 success rate in meeting the functional and non-functional requirements. Critical successes are the implantation of the system and the ability to perform some of the basic ideas when using it. However, significant challenges arise due to the central connectivity issue to the gateway. This made using the blockchain for provenance unobtainable since there was no way to submit to the blockchain.

These issues impacted user verification, data storage, and tracking on the blockchain. Iteration two, of the system discusses implementing a new blockchain system to allow all distribution to be stored locally on the host system.

10.7 Scale of project

When analysing the entire project, it is critical to recognise its total scope and whether it was possible to accomplish it within the timeframe specified. The project has been simplified from its original concept. Knowing this issue beforehand allowed for exploring other ideas, such as those offered in the second iteration, which could result in a more user-friendly system and a better project outcome.

Despite the simplifications, the project yielded a primarily functional architecture. Some realised concepts show the project's promise, although more time would have helped. Given the project's scope, it is critical to recognise that some material may need to be included, affecting the report's conclusions. It is essential to recognise that this initiative is still in its early phases, and there may be more effective solutions than those now proposed.

10.8 Interchangeable providers

The project aimed to build upon the ProvChain model, ensuring data provenance and integrity using blockchain technology while introducing interchangeable parts into the architecture. This allowed for the integration of cloud service providers, such as Owncloud and NextCloud, with a blockchain-based provenance system. OwnCloud performed better under high query loads, but NextCloud handled more concurrent users more effectively. This interchangeability enhances the system's flexibility and resilience, ensuring it can adapt to different environments and demands.

However, many challenges with blockchain connectivity, mainly with chainpoint, hindered the full implementation of data provenance tracking with validation. Highlighting the need for a more reliable and maintainable blockchain solution. The proposed second iteration suggests adopting a Sawtooth hyperledger blockchain to address these issues. These changes aim to achieve better system performance and reliability, benefiting local hosting and greater control over the blockchain code. The project allows for a more robust idea with capabilities of evolving with technological advancements and user needs by enabling the system to have interchangeable areas

Conclusion

The project explored enhancing data provenance and integrity with blockchain technology, focusing on introducing interchangeable components. The main objective was to build upon the ProvChain architecture and have interchange components, such as blockchain or cloud service. This addresses performance and scalability concerns, ensuring the system has adaptability, resilience and longevity. Significant progress was made when implementing it through the development and testing phases, demonstrating the idea's potential despite some issues with chainpoint.

The project partially successfully integrated blockchain technology for data provenance with interchangeable features with cloud service providers, namely OwnCloud and NextCloud. These providers show some distinct advantages, with OwnCloud performing well under high loads and NextCloud handling a more significant number of concurrent users. Implementing essential features like data hashing, access controls, and encryption met some functional requirements, showcasing the project's potential to enhance data security. However, the project met a significant challenge with blockchain connection through the chainpoint gateway; these services could have improved the ability to track and validate data provenance.

These connectivity issues showed the need for a more reliable blockchain solution, showing the consideration of alternative frameworks like the sawtooth hyperledger. The hybrid methodology of the iterative waterfall approach introduced project management complexities. Despite these issues, virtual machines provided a great development environment, although future planning could benefit from more structured version control.

The first project management timeline proved ambitious, necessitating adjustments and revisions. The updated Gannt chart reflected a more realistic scope, allowing the project to be completed within the university time frame. This demonstrated the importance of initial planning, continuous learning, and dealing with evolving technology and unknown circumstances. Future work should explore integrating alternative blockchain frameworks to enhance connectivity and further research into the interchange features.

Overall, this project demonstrates the potential of blockchain technology for ensuring data provenance and integrity across various applications. Further research and refinement are needed to realise the potential fully; the project demonstrated the foundation for more developments.

References

Apache (2015). Welcome! - The Apache HTTP Server Project. [online] Apache.org. Available at: https://httpd.apache.org/.

Bhowmik, D., & Feng, T. (2017, August). The multimedia blockchain: A distributed and tamper-proof media transaction framework. In *2017 22nd International conference on digital signal processing (DSP)* (pp. 1-5). IEEE. https://doi.org/10.1109/ICDSP.2017.8096051

Frequently asked questions about the alternative blockchain. (2023, October 10). CoinCash. https://coincash.eu/faq/frequently-asked-questions-about-the-alternative-blockchain

Github. Chainpoint (n.d.). Chainpoint. [online] GitHub. Available at: https://github.com/chainpoint

GitHub. (2023a). chainpoint/chainpoint-cli. [online] Available at: https://github.com/chainpoint/chainpoint-cli

GitHub. (2023b). chainpoint/chainpoint-gateway. [online] Available at: https://github.com/chainpoint/chainpoint-gateway

GitHub. (2024a). chainpoint/chainpoint-core. [online] Available at: https://github.com/chainpoint/chainpoint-core

GitHub. (2024b). chainpoint/chainpoint-js. [online] Available at: https://github.com/chainpoint/chainpoint-js

Hyperledger Sawtooth - Hyperledger Sawtooth (Archived) - Hyperledger Foundation. (n.d.). Wiki.hyperledger.org. from https://wiki.hyperledger.org/display/sawtooth

Lin, P.Y. (2023). *What is Cryptography in Blockchain? How Does it Work?* [online] CFTE. Available at: https://blog.cfte.education/what-is-cryptography-in-blockchain/.

MariaDB.org. (2015). MariaDB Foundation - MariaDB.org. [online] Available at: https://mariadb.org/.

Owncloud.com. (2020). Older Versions - ownCloud. [online] Available at: https://owncloud.com/older-versions

Prayudi, Y., & Sn, A. (2015).
Digital chain of custody: State of the art. *International Journal of Computer Applications*, *114*(5).
https://www.researchgate.net/profile/Yudi-Prayudi/publication/273694917_Digital_Chain_of_Custody_State_of_The_Art/links/5508eb510cf2d7a2812b6945/Digital-Chain-of-Custody-State-of-The-Art.pdf

The PHP Group (2019). PHP: Hypertext Preprocessor. [online] Php.net.
Available at: https://www.php.net/.

Tierion (2018). Tierion: Blockchain Proof Engine. [online] Tierion.com.
Available at: https://tierion.com/.

Tan, C. B., Hijazi, M. H. A., Lim, Y., & Gani, A. (2018).
A survey on proof of retrievability for cloud data integrity and availability: Cloud storage state-of-the-art, issues, solutions and future trends. *Journal of Network and Computer Applications*, *110*, 75-86.
https://doi.org/10.1016/j.jnca.2018.03.017

Vigil, M., Buchmann, J., Cabarcas, D., Weinert, C., & Wiesmaier, A. (2015).
Integrity, authenticity, non-repudiation, and proof of existence for long-term archiving: A survey. *Computers & Security*, *50*, 16-32
https://doi.org/10.1016/j.cose.2014.12.004

Wei, P., Wang, D., Zhao, Y., Tyagi, S. K. S., & Kumar, N. (2020).
Blockchain data-based cloud data integrity protection mechanism. *Future Generation Computer Systems*, *102*, 902-911.
https://doi.org/10.1016/j.future.2019.09.028

Wikipedia Contributors (2019). SHA-2. [online] Wikipedia.
Available at: https://en.wikipedia.org/wiki/SHA-2.

Appendix

Certification of ethics

UNIVERSITY OF PORTSMOUTH

Certificate of Ethics Review

Project title: blockchain technology for data provenance and proof of integrity

Name:	jack tallis	**User ID:**	up2054361	**Application date:**	06/11/2023 22:20:20	**ER Number:**	TETHIC-2023-106713

You must download your referral certificate, print a copy and keep it as a record of this review.

The FEC representative(s) for the **School of Computing** is/are **Elisavet Andrikopoulou, Kirsten Smith**

It is your responsibility to follow the University Code of Practice on Ethical Standards and any Department/School or professional guidelines in the conduct of your study including relevant guidelines regarding health and safety of researchers including the following:

- University Policy
- Safety on Geological Fieldwork

It is also your responsibility to follow University guidance on Data Protection Policy:

- General guidance for all data protection issues
- University Data Protection Policy

Which school/department do you belong to?: **School of Computing**
What is your primary role at the University?: **Undergraduate Student**
What is the name of the member of staff who is responsible for supervising your project?: **David williams**
Will financial inducements (other than reasonable expenses and compensation for time) be offered to participants?: No
Are there risks of significant damage to physical and/or ecological environmental features?: No
Are there risks of significant damage to features of historical or cultural heritage (e.g. impacts of study techniques, taking of samples)?: No
Does the project involve animals in any way?: No
Could the research outputs potentially be harmful to third parties?: No
Could your research/artefact be adapted and be misused?: No
Will your project or project deliverables be relevant to defence, the military, police or other security organisations and/or in addition, could it be used by others to threaten UK security?: No
Please read and confirm that you agree with the following statements: I confirm that I have considered the implications for data collection and use, taking into consideration legal requirements (UK GDPR, Data Protection Act 2018 etc.), I confirm that I have considered the impact of this work and and taken any reasonable action to mitigate potential misuse of the project outputs, I confirm that I will act ethically and honestly throughout this project

Supervisor Review

As supervisor, I will ensure that this work will be conducted in an ethical manner in line with the University Ethics Policy.

Supervisor comments: **The engineering project involves two principle tasks: 1. recreate an existing blockchain for provenenace solution setup, 2. extend/develop/enhance this. As it remains unclear at this early stage how to extend/develop/enhance the existing solution, please seek sign off of the novel design ahead of implementation.**

Supervisor's Digital Signature: **david.williams2@port.ac.uk** Date: **07/11/2023**

School of Computing
Final Year Engineering Project

Project Initiation Document

Jack Tallis

Cyber Security and Forensic computing

Blockchain technology for data provenance and proof of integrity

1. Basic details

Student name:	Jack Tallis
Draft project title:	Blockchain technology for data provenance and proof of integrity
Course and year:	Cyber security and forensic computing
Project supervisor:	David Will]iams
Client organisation:	
Client contact name:	

2. Degree suitability

Cyber security protects systems, networks, programs, applications, and data from potential attackers. The main objective of these attackers is to damage, delete, change, and potentially steal data. Having ways of protecting data and ensuring it is out of harm's way of other people when a company or person is trying to keep confidentiality, integrity, and availability for their assets. This can also be very important in digital forensics; if there is a case and you want the most authentic way of proving something, making sure that what you're looking for or looking at is correct is very important.

My course is Cyber Security and Forensic Computing. My topic is using blockchain technology for data provenance and proof of integrity. We can adapt blockchain technology to ensure data security. This meets the criteria for my course because cyber security is all about data protection and protecting integrity, making sure the data is immutable. Having these two aspects as the main priority is excellent for forensic computing when collecting evidence proving the data is essential to a case and bringing an idea of a chain of custody if the data can be proven. Also, blockchain technology is a new and evolving technology that investigates its appliance in the context of cyber security, and forensics provides an opportunity for potential research. With the potential for data immutable, it does come with some risk to people if they do not want that; since immutability means you can nor delete or modify the data, it could potentially cause an integrity issue.

3. The project environment and problem to be solved

With the use of blockchain technology, I want to provide a way to prove data provenance and proof of integrity using this technology. This ensures me set up two ways to architect data and how I can present it and allow ways for this data to be validated. This is meaningful because it will help when trying to be transparent about data. The ledgers that the blockchain provides allow us to specifically see where the data goes, whose data is it, and its origin. This is important because it allows us to change the potential ways that data can be authenticated.

Benefits of this project can include:

- **Proof of integrity**
    - With the immutability of the blockchain, I may investigate how blockchain technology can successfully address potential issues with data tampering.
    - The idea of immutable data can also be an issue to see the problems data immutability can cause. Since the idea makes it so it can not be changed\deleted

- **Provenance**
    - With blockchain technology it allows for ease of traceability; this is great for data provenance it allows us to follow that data from its origin, seeing the steps it takes and where it goes, if there are holders of the data we can see those individuals as well.

- **Blockchain for digital forensics**
    - Security changes are evolving with the digital forensics process, and the need for potential new tech for the potential changes in the future and now.
    - My report could evaluate how blockchain technology can be used practically for forensic investigation when dealing with evidence, seeing how the technology can improve readability and reliability and work for more authentic presentation of data that can be vital for a specific case.

4. Project aim and objectives

Aim: Evaluate the implementation of blockchain technology solutions for ensuring data provenance and proof of integrity

Objectives:

- Identify architectures
    - Set up the architectures
    - Identify any potential improvements
- Improved atchitecture
    - Make improvements to architecture
    - Compare/Evaluate my solution against the other architectures
- Evaluate the improvements and analyse its importance
- Evaluate how this implantation is important

5. Project constraints

- We have a time limit of "May 20th, 2024" to complete our project. **Time** is a constraint due to a variety of variables, including:
    - I will have particular time frames for research and development. This deadline means I must manage my time to the best of my ability to meet it.
- **Resource management/limitation:** Limiting the use of specific resources and determining when resources are available will slow down the project's production.
- **Data availability**: I must ensure that the data is available and accessible for relevant testing; if I do not have this data, it can slow down the testing/usage of the potential architectures.
- **Technology constraints: With the potential use of "block-chain technology" and Cloud-service providers "GSP," having access to this technology and knowledge of its usage will be needed. If not, it will constrain** how fast the project takes off.
- **Skillset gaps:** I do not understand specific aspects of my project. I need to make sure I can do the task asked and understand the whole project and the potential use of technologies.
- **Testing and validation:** understanding potential challenges to validating the processes and achieving the required outcome
- **Budget for equipment:** some cloud service providers have a cost-of-usage ratio, making sure I can afford to use these services when needed
- **Word/Page Limit:** 10k words

6. Facilities and resources

All projects will need resources (equipment and people) to be completed. List:

- Any computing/IT you will use/require?
    - E.g. personal laptop, Personal desktop, word, PowerPoint, survey generator, potential APIs for blockchain usage, Project, Programming software, cloud-providing software
- Any other facilities/resources you will use/require?
    - E.g. library, Academic Skills Unit (only open from 9:00am - 5:00),

7. Log of risks

Asset	No	Description	Likelihood (high, medium, low)	Impact	Mitigation/Avoidance
Personal					
	1	**Laptop failure**	Medium	Cause delays to the project timeline and data loss	Back up work to a USB and potentially some online sources like google drive. This gives me many more options to protect my work
	2	**Backup lost** - a loss of a potential backup such as google drive, usb, file.	low	Can cause a delay to a project timeline	Making multiple backups on different platforms online **and** physical; Google drive, One drive, USB, HDD
	3	**Device Charger breaks\power cable** - Not being able to use device that has access to my work/software	Low/Medium	Can cause a time delay for if I have to wait for a new one	Having a backup device that I can use, eg: Desktop, Phone
	4	**Network problems** - Such as not being able to use the network for potential programs or document readings	Medium/High	Can cause some time constants depending on how long it is	Use services where network potential is provided: library
	5	**Scope creep** - Addition features requirements getting added as the project is in “development”	Medium/High	Can cause issues with the development time if have to go back and implement something	Make sure to communicate with superior about the objectives and make a clear plan
	6	**Deadline** - Maybe what happens is	Medium	Causes panic and issues with	make sure to manage time to make sure there is always

		that something is taking longer than thought, having a deadline can cause issues with that		the development time	development/research time for potential new objectives
	7	**Lack of clarity**	Medium/High	Time issues to try and make sense of something especially if it is wrong	Make sure to understand the question and deconstruct it with supervisor
Cloud service providers					
	1	Downtime	Medium	If the service is down I will not be able to carry out some aspects of my project	See if any other parts of the project need to be completed during this downtime.
	2	Data security	Medium	Make sure to provider I potentially use is secure for holding and protecting my data	Do a potential risk assessment of the service
	3	Cost	Low/Medium	If the cost of a cloud service provider is too much overtime I may not be able to spend to keep going	View many different cloud provider solutions, and see the different prices if there is any, and see which is most beneficial
	4	Potential migration challenges	Low/Medium	If in the event that I need to change CSP, being able to migrate quite easily would be useful since it will help with the time that it takes me to start working on the project again	Make sure to understand any potential ways of switching CSP, if there is any issues make sure to know the ways of migrating

8. Project deliverables

All projects produce ‘things’ which need to be listed here

For a study or research project, what documents and artefacts will be produced, e.g., primary research tools, datasets, data analysis, and your report?

- Literature review
- Ethics review
- Architecture recreation
    - Setup of ProvChain in a working environment
- My architecture
    - Add potential new features to Provchain
    - Run tests to test if new features are improvements
    - Identify the metrics for my architecture
- Evaluate the differences
- Proof of testing
- Evaluate the performance of the new architecture against the others
- Report

9. Project Approach

I will adapt to the “waterfall methodology”, which allows me to work in more linear, sequential development steps: “requirements, Design, implantation, Verification, deployment”. Working with this methodology in mind allows me to understand clearly how I can develop. It is all clearly laid out for me to go by to ensure that I get all the information I need for perfect development. It will also allow me to ensure I have all the primary data I need for the other parts of my project.

With this in mind, my project will follow Scrum, a more agile methodology. With this idea, I can break other project parts into phases. This will let me micromanage project details that do not need “development” and keep track of where I am with those parts.

literature Research: FYP Paper Search

10. Project tasks and timescales

No	Stage	Dates	Main Tasks
1	Project start-up	18/09/23 - 02/10/23	Choose a topic and supervisor
2	Project ignition document	2/10/23 - 20/10/23	Complete the Project Ignition document
3	Literature review	23/10/23 - 22/11/23	Conduct a literature review
4	Ethics review	27/11/23 - 9/12/23	Get ethics certification from the University
5	Data collection	9/12/23 - 13/01/23	Start to collect data
6	Architecture recreation	09/12/23 - 13/01/23	Start the re-creation of ProvChain and the second architecture
7	Create own architecture	24/01/24 - 16/02/23	Start adapting to creating my own
8	Evaluate the differences and how the new one is potentially better	23/03/24 - 01/05/24	Start to take all the data collected from the architectures and evaluate the differences
9	Project submission	20/05/2024	Submit the project on moodle

11. Supervisor meetings

Over the course of three weeks there will be three different meetings, room dependency is available to change to need to check timetable every week:

Tuesday week 1: 11AM group meeting with all supervisees, for 1 hour. In person meeting.
Tuesday week 2: 9AM 1-1 meeting, for 30 mins. Online meeting.
Tuesday week 3: 11AM drop in meeting, for 1 hour. In person meeting. (not mandatory)

12. Legal, ethical, professional, social issues

Legal:

- Intellectual property: make sure to understand the IP rights of anything I use relating to my report
- Data privacy laws: Ensure to compile with local data protection laws, "General Data Protection Regulation"
- Recordkeeping compliance

Ethical:

- Informed consent: if I get data from individuals, I ensure people fill out a proper consent form with the information of what they ill be consenting to—possible approval from the University with ethics review.
- Academic integrity: Make sure to cite appropriately any potential sources used.
- Transparency: Make sure to be transparent about the project and inform of what is being done and what it does

Professional:

- Documentation/report: document the process of the project, making sure to see everything just in case something changes in the future that can cause some issues
- Risk Management: understand the potential risks of any of the services that I will be using, if needed to perform a potential risk assessment
- Fulfilling CIA: meeting the standards of confidentiality, integrity, and availability
- distributed ledger technology regulation

Social (if any):

-

13. Permission

Please tick

❑ I give permission for my PID to be made available to other students as examples of previous work.

❑ I do not give permission for my PID to be made available to other students as examples of previous work.

Date: 19/10/2023__

Appendix A: Gantt chart

Place your detailed project plan here.

Below I have provided the ".mpp" file just in case the PDF does not work, I will also upload photos just in case again the PDF does not work.

Project1.mpp

Project1.pdf