UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Smart Contracts Technologies for improved data management towards enhanced data integrity, veracity, interoperability, transparency and traceability in Food Value chain systems

Diploma Thesis

Kanavaris Dimitris

Supervisor: Athanasios Korakis

Volos, September 2023

UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Smart Contracts Technologies for improved data management towards enhanced data integrity, veracity, interoperability, transparency and traceability in Food Value chain systems

Diploma Thesis

Kanavaris Dimitris

Supervisor: Athanasios Korakis

Volos, September 2023

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Τεχνολογίες Έξυπνων Συμβολαίων για Βελτιωμένη Διαχείριση Δεδομένων με Στόχο την Ακεραιότητα, την Φιλαλήθεια, την Διαλειτουργικότητα, την Διαφάνεια και την Ιχνηλασιμότητα στα Συστήματα της Αλυσίδας Τροφίμων

Διπλωματική Εργασία

Καναβάρης Δημήτρης

Επιβλέπων: Αθανάσιος Κοράκης

Βόλος, Σεπτέμβριος 2023

Approved by the Examination Committee:

Supervisor **Athanasios Korakis**
Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Parisis Flegkas**
Assistant Professor, Department of Electrical and Computer Engineering, University of Thessaly

Member **Antonios Argyriou**
Associate Professor, Department of Electrical and Computer Engineering, University of Thessaly

Acknowledgements

First and foremost I would like to thank Professor Athanasios Korakis for his guidance and support towards the completion of this thesis as well as for giving me the opportunity to join NITlab and cooperate with such an inspiring research group. I would also like to thank Assistant Professor Parisis Flegkas and Associate Professor Antonios Argyriou who were members of the examination committee and approved this thesis.

I would also like to thank Senior Researcher and Postdoctoral Apostolos Apostolaras for his valuable contribution to this thesis always helping me and steering me in the right direction. Special thanks to Vasilis Zafeiris and my colleagues at NITLab with whom I had the pleasure of collaborating on multiple projects.

Last but not least, I would like to thank my family and everyone who stood by me during this 5-year journey from the bottom of my heart, for the support they gave me throughout these years.

DISCLAIMER ON ACADEMIC ETHICS AND INTELLECTUAL PROPERTY RIGHTS

«Being fully aware of the implications of copyright laws, I expressly state that this diploma thesis, as well as the electronic files and source codes developed or modified in the course of this thesis, are solely the product of my personal work and do not infringe any rights of intellectual property, personality and personal data of third parties, do not contain work / contributions of third parties for which the permission of the authors / beneficiaries is required and are not a product of partial or complete plagiarism, while the sources used are limited to the bibliographic references only and meet the rules of scientific citing. The points where I have used ideas, text, files and / or sources of other authors are clearly mentioned in the text with the appropriate citation and the relevant complete reference is included in the bibliographic references section. I also declare that the results of the work have not been used to obtain another degree. I fully, individually and personally undertake all legal and administrative consequences that may arise in the event that it is proven, in the course of time, that this thesis or part of it does not belong to me because it is a product of plagiarism».

The declarant

Kanavaris Dimitris

Diploma Thesis

Smart Contracts Technologies for improved data management towards enhanced data integrity, veracity, interoperability, transparency and traceability in Food Value chain systems

Kanavaris Dimitris

Abstract

Food fraud is a major issue that has gained increasing attention in recent years. Food fraud refers to the intentional deception or misrepresentation of food products with the goal of financial gain at the expense of the consumers' health and safety. Mislabeling, ingredient substitution, adulteration, or counterfeiting of food products are some of the ways food fraud can be achieved and go undetected. To address this issue, extensive monitoring of the Food Supply Chain must be established. Furthermore, data integrity must also be ensured in order to create a tamper-proof system that supports veracity, interoperability, transparency, and traceability. In order to achieve that, Blockchain technology is proposed. Through the use of blockchain technology data integrity can be enforced resulting in a fully transparent system where the consumer can view the stages the food product went through thus increasing trust. The proposed solution was tested in real-world scenarios using AGROLOG, a logistics app used to track the stages crops go through from farm to fork.

With the integration of blockchain technology into Food Supply Chain tracking apps, such as AGROLOG, food fraud prevention can be achieved since every single stage the materials go through until they become a finished food product can be monitored by a tamper-proof system guaranteeing data integrity, preventing adulteration and increasing consumer trust.

Keywords:

Food Fraud, Food Supply Chain, Blockchain, Food Fraud Prevention

Διπλωματική Εργασία

Τεχνολογίες Έξυπνων Συμβολαίων για Βελτιωμένη Διαχείριση Δεδομένων με Στόχο την Ακεραιότητα, την Φιλαλήθεια, την Διαλειτουργικότητα, την Διαφάνεια και την Ιχνηλασιμότητα στα Συστήματα της Αλυσίδας Τροφίμων

Καναβάρης Δημήτρης

Περίληψη

Η απάτη στον τομέα τροφίμων είναι ένα μείζον ζήτημα που αποκτά όλο και μεγαλύτερη προσοχή τα τελευταία χρόνια και αναφέρεται στη σκόπιμη εξαπάτηση ή παραποίηση προϊόντων διατροφής με στόχο το οικονομικό κέρδος σε βάρος της υγείας και της ασφάλειας των καταναλωτών. Λανθασμένη επισήμανση, υποκατάσταση συστατικών, νοθεία ή παραποίηση προϊόντων τροφίμων είναι μερικοί από τους τρόπους απάτης που είναι δυνατόν να επιτευχθούν με μεθόδους που είναι ολοένα και πιο δύσκολο να εντοπιστουν. Για την αντιμετώπιση αυτού του ζητήματος, θα πρέπει να υπάρχει εκτεταμένη παρακολούθηση των τροφίμων κατα μήκος της Εφοδιαστικής Αλυσίδας. Επιπλέον, είναι εξίσου σημαντικό να διασφαλίζεται η ακεραιότητα των δεδομένων με στόχο τη δημιουργία ενός συστήματος προστασίας από παραβιάσεις που υποστηρίζει την ακρίβεια, τη διαλειτουργικότητα, τη διαφάνεια και την ιχνηλασιμότητα. Για να επιτευχθεί αυτό, προτείνεται η τεχνολογία Blockchain. Μέσω της χρήσης της τεχνολογίας του Blockchain η ακεραιότητα των δεδομένων μπορεί να επιτευχθεί, με αποτέλεσμα ένα πλήρως διαφανές σύστημα όπου όχι μόνο ο καταναλωτής αλλά και ο κάθε εμπλεκόμενος στην αλυσίδα αξίας μπορεί να δει και να ελέγξει τα στάδια που πέρασε το τρόφιμο, αυξάνοντας έτσι την εμπιστοσύνη ανάμεσα στους καταναλωτές και τους χρήστες μιας τέτοιας αλυσίδας. Η προτεινόμενη λύση δοκιμάστηκε σε πραγματικά σενάρια με τη χρήση του AGROLOG, μιας εφαρμογή λογιστικών η οποία χρησιμοποιείται για την παρακολούθηση των σταδίων από τα οποία περνούν οι τρόφιμα από το αγρόκτημα στο πιρούνι. Με την ενσωμάτωση της τεχνολογίας του Blockchain στις εφαρμογές παρακολούθησης της Εφοδιαστικής Αλυσίδας Τροφίμων, όπως η εφαρμογή AGROLOG, είναι δυνατόν να επιτευχθεί η πρόληψη της απάτης στα τρόφιμα, καθώς κάθε στάδιο που διανύουν τα υλικά μέχρι να γίνουν έτοιμα προϊόντα παρακολουθείται από ένα αδιάβλητο σύστημα, εγγυώμενο την ακεραιότητα των

δεδομένων, προλαμβάνοντας την παραποίηση και αυξάνοντας την εμπιστοσύνη του τελικού καταναλωτή.

Λέξεις-κλειδιά:

Απάτη στον Τομέα Τροφίμων, Εφοδιαστική Αλυσίδα Τροφίμων, Blockchain, Πρόληψη Απάτης στον Τομέα Τροφίμων

Table of contents

List of figures

List of tables

Abbreviations

FSC	Food Supply Chain

Chapter 1

Introduction

Food supply chains refer to the interconnected network of processes involved in producing, processing, distributing, and consuming food. From farmers who cultivate crops to processors, distributors, retailers, and finally consumers, each link in the chain plays a vital and crucial role in ensuring the availability and accessibility of food. However, these supply chains often face plenty of challenges. Environmental factors such as droughts, floods, and pests can disrupt agricultural production. Lack of transparency can lead to food wastage, while unethical practices can exploit vulnerable workers. Another significant concern is food fraud, where products are deliberately altered for economic gain, potentially compromising food safety and consumer trust. Examples of food fraud include mislabeling, adulteration, and selling expired products as fresh. Addressing these issues requires an approach that considers the environmental, social, and economic dimensions of food production and distribution.

1.1 Thesis Goals

This thesis aims to present a solution to challenges in the management of food supply chains, notably the lack of transparency with a primary focus on eliminating food fraud. By harnessing the capabilities of blockchain, an innovative technology built to solve issues related to trust, transparency, and data integrity, businesses can revolutionize various industries, including finance, supply chain management, healthcare, and more. We aim to establish a transparent, traceable, and tamper-proof system for food products, ensuring data authenticity and integrity. We integrate the blockchain solution into a food supply chain tracking app, AGROLOG which is an app built to fully monitor the stages the materials go through in

the food supply chain journey providing valuable insight, in order to test and evaluate it. Through the collaboration of the AGROLOG app and the blockchain network, data integrity and transparency were achieved while maintaining full traceability. The evaluation of the solution proved that food supply chains can greatly benefit and overcome significant issues through the usage of blockchain technology.

1.1.1 Contributions

The contributions of this thesis are:

1. A comprehensive analysis of the pain points Food Supply Chains face
2. To showcase how Blockchain and smart contracts technology can be efficiently utilized to overcome the issues the Food Supply Chains face.
3. An implementation of a Blockchain network supporting a Food Supply Chain system enabling end-to-end data transparency, safe-guarding, integrity, and veracity.
4. An evaluation of the proposed Blockchain system providing valuable insights about the performance and efficiency of the system.

1.2 Outline

The thesis is structured as follows: In section 2, the Food Supply Chain is explained in greater detail while its importance and issues are discussed. In section 3, a preview of AGROLOG, an app that was created to monitor Food Supply Chains, is presented showcasing its usage and functionality. In section 4, Blockchain technology and its benefits are presented while providing an insight into its early days as well as its building blocks and the reasons its integration into Food Supply Chains can revolutionize the field. In section 5, the system architecture used for the Thesis is showcased discussing several tools and services. In section 6, we discuss the implementation of the blockchain network. In section 7, Blockchain integration to AGROLOG is presented discussing the implementation and evaluate its usefulness. In section 8, the thesis is concluded and future work and improvements that can be made are discussed.

Chapter 2

Food Supply Chain

2.1 What is the Food Supply Chain

The Food Supply Chain (FSC) refers to the stages that food products pass through including the initial production or cultivation stage up to the point of consumption. These stages consist of multiple interconnected processes, activities, and entities that work together to maintain the availability of food and meet consumers' demands. [9]
The Food Supply Chain typically includes the following stages (as shown in figure 2.1):

1. **Production**: The production stage consists of the agricultural phase of the Food Supply Chain. This is when the food is cultivated, grown, and finally harvested by the farmers.

2. **Handling and Storage**: After the product has been harvested the next stage is Handling and Safe Storage to ensure the sustainability and longevity of the raw materials.

3. **Processing and Packaging**: The next stage consists of Processing and Packaging the product to get it ready for distribution. Processing and Packaging vary from product to product depending exclusively on the nature of the raw material.

4. **Distribution**: The product is then distributed to retailers (wholesalers or retailers). An interesting metric is the Food Mile. The Food Mile refers to the distance that food items are transported during their journey from the producer to the consumer.

5. **Consumption**: The final product is now available for consumption by the consumers.

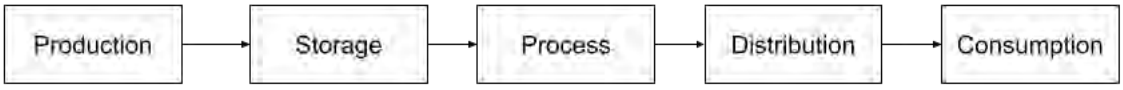

Figure 2.1: Food Supply Chain Flow

2.2 Food Supply Chains Importance

Food Supply Chains play a vital role in today's society, as they are essential to producing safe food that meets the consumer's high demands. Using the Food Supply Chain, food that did not pass a quality check but was, by mistake, packaged or food that failed to pass a quality test after the distribution stage can now be easily revoked by using the chain's tracking capabilities. Furthermore, consumers can also use the chain's tracking features so that they can understand the origin of their food and how it has been produced thus providing important insight that would otherwise be unavailable. Moreover, intentional errors that can occur at any stage can be detected and prevented early thus safeguarding the Food Supply Chains. Finally, transparency and trust among the actors participating in the Food Supply Chain can be gained also resulting in an increase in the confidence level of the final consumers about the product they are purchasing.

2.3 Food Supply Chain Issues

Food supply chains are inherently complex, including multiple actors ranging from local farmers, processors, distributors, and retailers. This complexity is further compounded by the global nature of these supply chains, often spanning multiple countries and continents. In addition to their international reach, these supply chains frequently exhibit fragmentation, with various stakeholders operating independently, making coordination and transparency vital for ensuring the safety, quality, and sustainability of the global food supply. The length and complexity of the supply chain create opportunities for dishonest practices to occur at different stages such as mislabeling, adulteration, and selling expired products as fresh. The main reasons why the Food Supply Chain is vulnerable to fraud are :

1. Lack of Transparency: The lack of transparency in the Food Supply Chain makes it difficult to trace the origin and the authenticity of a product. This lack of transparency

allows individuals or companies to introduce fraudulent ingredients and pass them off as authentic.

2. Fragmentation: The multiple stages included in the Food Supply Chain are heavily fragmented, resulting in challenging monitoring of each stage. Each stage introduces further fragmentation to the chain allowing for fraudulent activities to take place.

3. Globalization: The globalization of the food industry results in even more complex Food Supply Chains where ingredients and products are sourced from various sources worldwide. This complex international network increases the risk of fraud, as it becomes harder to verify the integrity of each ingredient.

4. Economic Incentives: Economic Incentives can be the main driving force behind fraudulent activities. The adulteration of products and ingredients such as the substitution of ingredients for cheaper ones, or misrepresentation of quality can result in higher profits or cost savings.

5. Limited Testing and Verification: Due to the large volume of food products moving along the Food Supply Chain it is often not feasible to test and verify every item thoroughly creating opportunities for fraudulent practices to go undetected.

Chapter 3

AGROLOG: A Food Supply Chain Tracking App

AGROLOG (3.1), developed by NITLab [10], is a logistics application designed to track and oversee the journey of a crop. From the moment of harvesting to its packaging and distribution, the app ensures seamless monitoring. It not only traces the crop's transit to the factory but also its subsequent storage and packaging. This comprehensive process involves the collaboration of users in multiple roles.

Figure 3.1: AGROLOG Logo

3.1 AGROLOG Architecture

The AGROLOG app is built using Nuxt.js [11] and Vue.js [12] for the front end and Express.js [13] for the back end. Vue.js is a popular open-source JavaScript framework for building user interfaces (UIs) while Nuxt.js is a powerful framework for building server-side rendered (SSR) and statically generated (SSG) web applications using Vue.js. It builds upon the Vue.js ecosystem and provides additional features and conventions to streamline the development process. Express.js is a popular web application framework for Node.js. It provides a simple and minimalist approach to building web servers and APIs, allowing developers to create robust and scalable applications. Finally, MongoDB [14] is used as AGROLOG's

database system. MongoDB is a popular open-source NoSQL database management system that stores data in a flexible, JSON-like document format called BSON (Binary JSON). It is designed to handle large volumes of data and provide high scalability and performance for modern applications. This stack is also referred to as MEVN stack, borrowing the first letter of each technology, Mongo, Express, Vue, and Node. Figure 3.2 illustrates the collaborative functioning of the mentioned technologies.

Figure 3.2: MEVN Stack [1]

3.2 Users

AGROLOG users can be distinguished into 3 categories :

- **Farmers**: The farmers are the ones responsible for the cultivation of the raw materials up until the distribution of these materials to the Collection Centers.
- **Collection Center Managers:** The collection center managers are responsible for gathering the raw materials from the farmers as well as shipping these materials to a central factory. The managers have access only to the Collection Center they are responsible for.
- **Factory Manager**: The factory manager is responsible for the reception of the materials and the appropriate storage of the raw food products as well as the processing of these materials in order to produce the final product ready for shipment. Factory managers have access to the factory as well as all of the fields owned by the Farmers and Collection Center to have an overview of the whole shipping process.

Each user has limited access and can only view and manage their side of the Food Supply Chain. The farmers can only view and manage their fields while the collection center managers can view the center they manage. Finally, the Factory Manager can receive, store, and process the raw materials while also having access to all of the collection centers to have an overview of the raw materials of each center available for shipping to the factory.

3.3 Views

As mentioned above each user has access to a different part of the app and thus obtains a different view. Below each view is described in greater detail.

1. **Calendar**

   Every user has access to the calendar view. The calendar is used to display events of the Food Supply Chain such as the receiving or shipment of the crop. Each user can view events that is directly responsible for, for example, a collection center manager can only view the logistics of his center and not the factory's. The calendar view is shown in figure 3.3

Figure 3.3: AGROLOG Calendar View

2. **Field**

   In the Field view, farmers can view their field details such as crop, variety, field id as well, and geographical location. A map is provided that displays the fields. Each displayed card is a different field belonging to a farmer. Through the usage of filters, users can view, sort, and filter the fields they want to view. Also, a way of adding new fields is provided as shown in figure 3.4. Through the form a user can submit a PDF file containing the field information as well as the owner of the fields and the app will extract, create, and add the new fields.

Figure 3.4: AGROLOG Field View

3. **Collection Center**

   The Collection Center view is used to manage the center's logistics. The centers are displayed as cards and the Collection Center Managers can view the center's geographical location on the map, view the fields that deliver crops to the center, view a list of farmers and add new ones when needed, and finally manage the center logistics as depicted in figures 3.5 and 3.6. Admin users can view all of the available centers. Filters are provided to provide a way of sorting and displaying centers that match certain attributes.

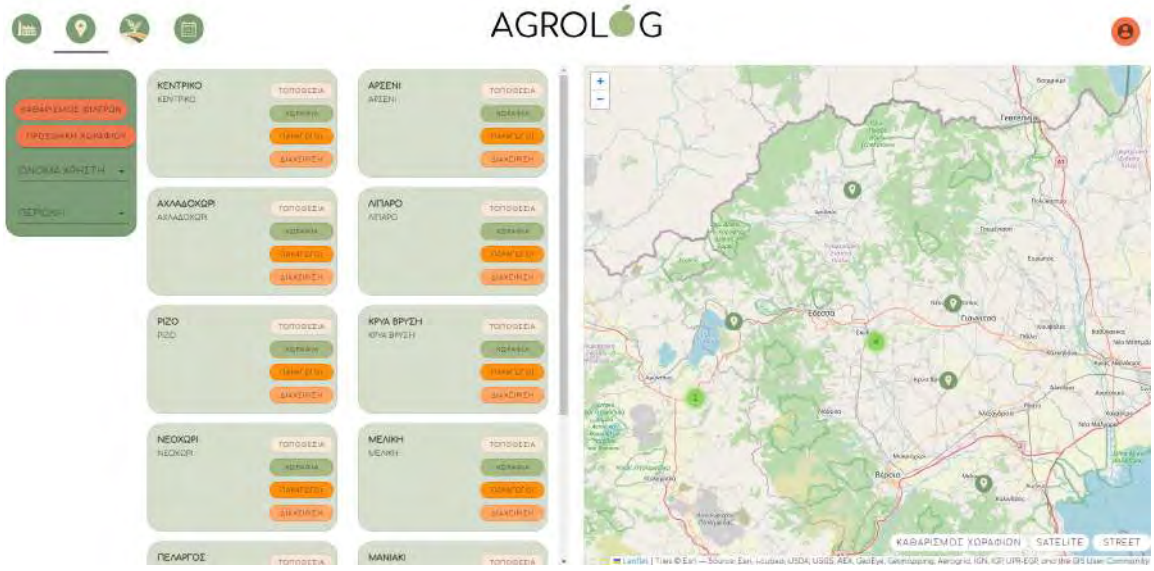

Figure 3.5: AGROLOG Collection Center View

Figure 3.6: AGROLOG Collection Center Logistics View

4. **Factory**

   Finally, in the factory tab, an admin user can manage the factory logistics, storage, and processing steps as shown in figures 3.7, 3.8, 3.9, and 3.10. Information on each stage of the Food Supply Chain is also displayed through graphs.

Figure 3.7: AGROLOG Factory View

Figure 3.8: AGROLOG Factory Logistics View

Figure 3.9: AGROLOG Factory Storage View

Figure 3.10: AGROLOG Factory Process View

3.4 Workflow

The raw materials cultivated by farmers are collected at various collection centers strategically located to accommodate farmers from diverse geographical areas. These centers serve as gathering points where the harvested produce is brought together. Once a sufficient quantity of materials has been accumulated, they are transported via trucks to a central factory. Upon arrival at the factory, the materials are received and stored, ensuring proper handling and preservation. The factory then initiates the processing phase, where the raw materials become finished products ready for distribution. A typical route flow can be seen in figure 3.11. The AGROLOG app provides monitoring capabilities for all the steps of the Food Supply Chain.

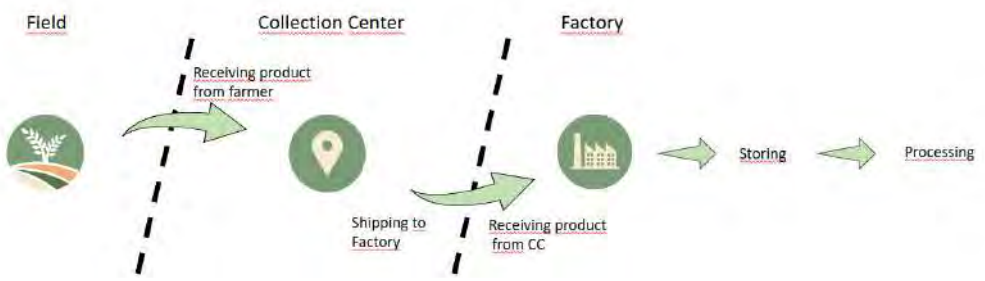

Figure 3.11: AGROLOG Route Flow

In more detail, the stages the crop passes through until it is transformed into the final product are :

1. **Collection Center Receives Crop**:

   The initial step involves the delivery of the crop to the Collection Center. Farmers bring their harvested crops to Collection Centers near them. The user responsible for the collection center then fills out a form stating the farmer's name and field from which the crop came from, the variety, and quality of the crop, the amount received as well and the date and time. Then a new entry is added to the collection center table and the graphs are updated to visualize the crops present at the collection center warehouse as depicted in figure 3.12. The view after the reception of a crop is shown in figure 3.13

Figure 3.12: AGROLOG Collection Center Receive Form

Figure 3.13: AGROLOG Collection Center Logistics after a crop has been received

2. **Shipment to Factory**:

   The next step involves the shipment of the crops to the factory. Once again the user responsible for the Collection Center fills out a form stating the factory to which the crop is shipped, the plate number of the truck that will deliver the shipment, and finally the date and time. Then the user can select from a list of available crops at the Collection Center warehouse and the amount of the shipment as well as the number of bins as shown in figure 3.14. Bins represent containers into which the crops are stored. Once the form has been submitted the crop is shipped to the factory and the graphs are again updated. The view after the shipment is shown in figure 3.15.

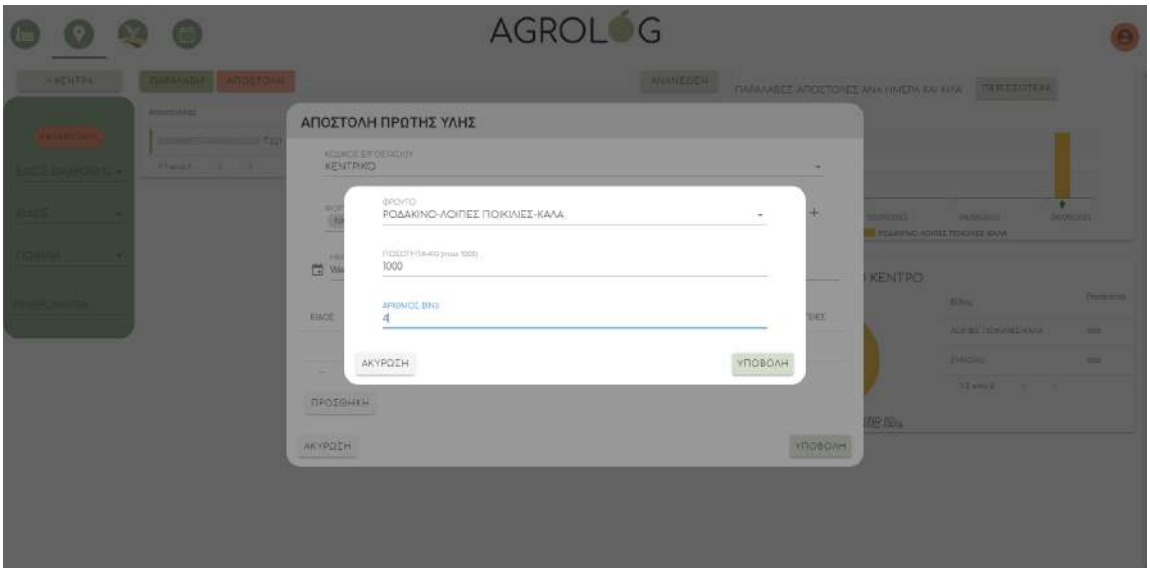

Figure 3.14: AGROLOG Collection Center Shipment form

Figure 3.15: AGROLOG Collection Center Logistics after shipment of a crop

3. **Receiving at the Factory**:

   To receive a shipment an admin user at the factory must submit a form stating the Collection Center name and then select the truck's plate number as well as the date and time as shown in figure 3.16. After that the crop has been received and the factory's data table and graphs have been updated. The factory can also receive crops directly from associations. To do that a form must be submitted stating the association's name, truck plate number, date and time, crop, variety, quality, amount received, and number of bins as shown in figures 3.17 and 3.18. After that, the crop from the association is added

to the data table of the factory, figure 3.19. The factory can view graphs displaying statistics such total amount of crops by quality and variety received per day or per Collection Center or association as depicted in figures 3.20 and 3.21.

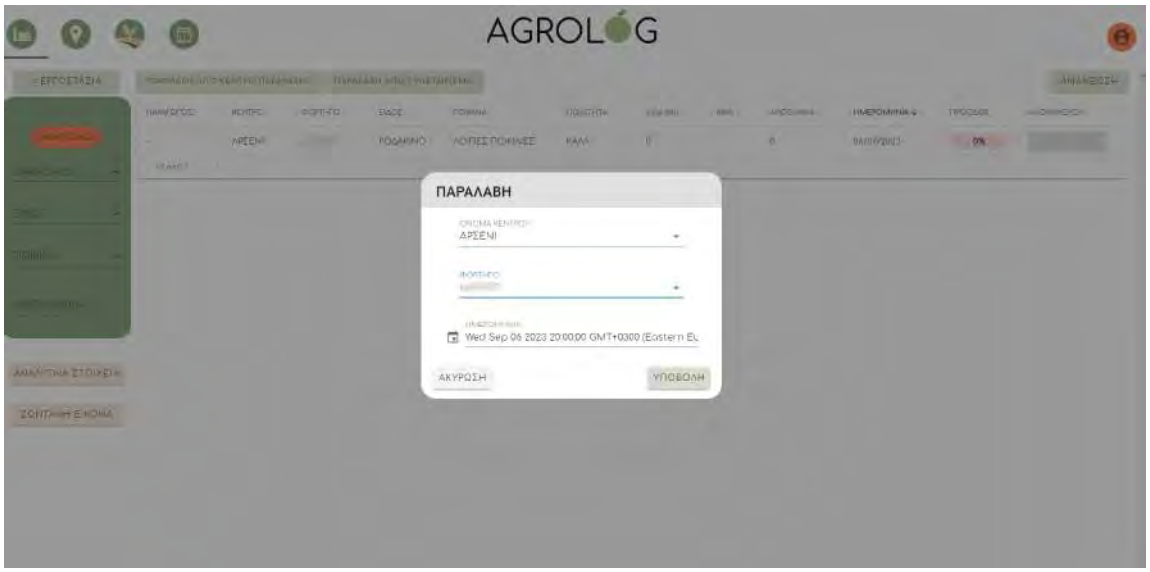

Figure 3.16: AGROLOG Factory Logistics form receiving crops from a Collection Center

Figure 3.17: AGROLOG Factory Logistics form submitting crop data from an Association

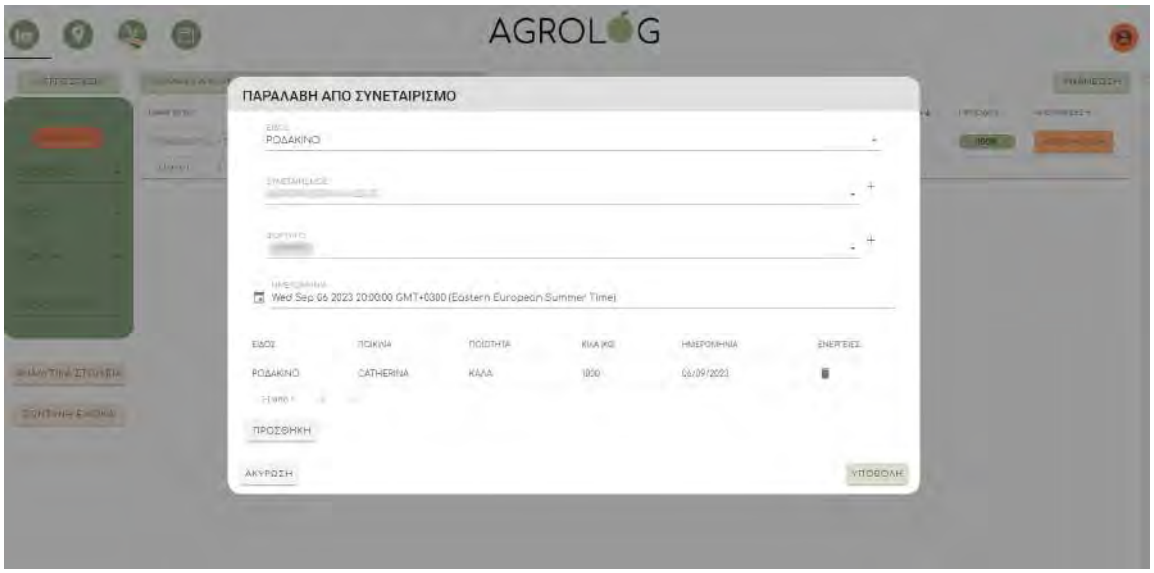

Figure 3.18: AGROLOG Factory Logistics form receiving crop from an Association

Figure 3.19: AGROLOG Factory Logistics after receiving crops from a Collection Center and an Association

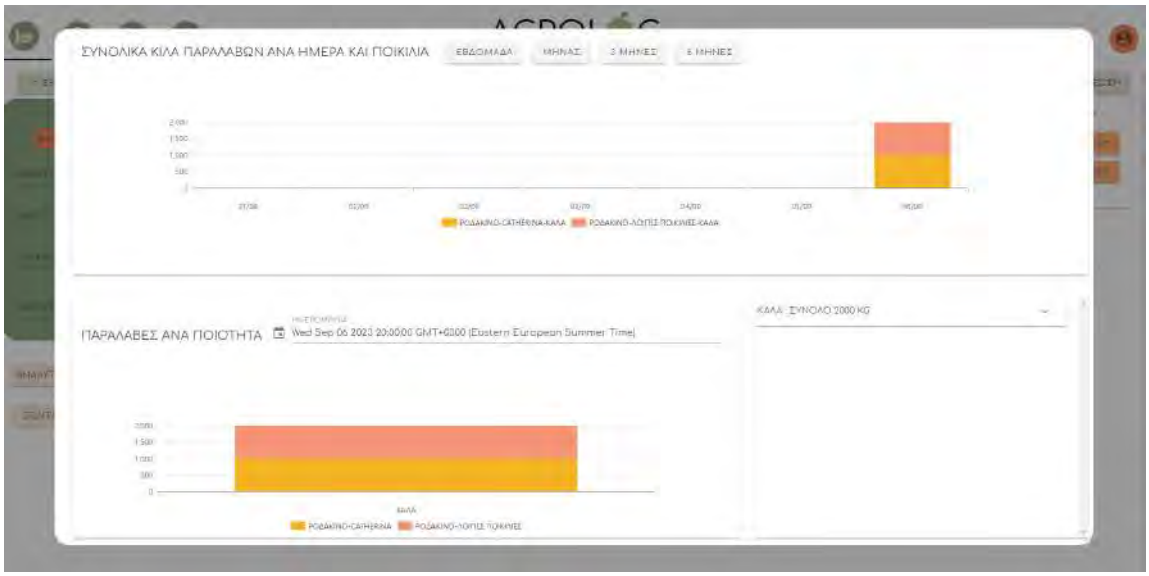

Figure 3.20: AGROLOG Factory Logistics graph informing about the amounts received per day

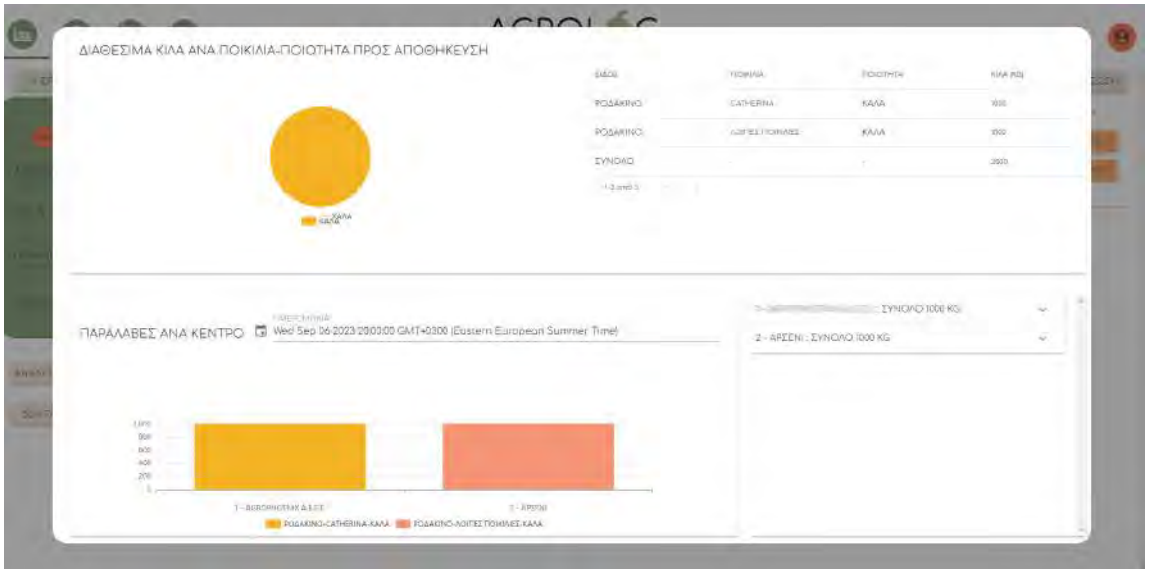

Figure 3.21: AGROLOG Factory Logistics graph informing about the amounts available for storage

4. **Storage of Crop**:

   Once the crop has been received at the Factory the next step is the storage. In order to store the crop the user must fill out a form stating the location of storage, the amount that will be stored, a characteristic of the crop, and the date and time as shown in figure 3.22. After that, the crop has been stored at the desired location and is ready to be

processed immediately or at a later time. The storage view after a crop has been stored is depicted in figure 3.23.

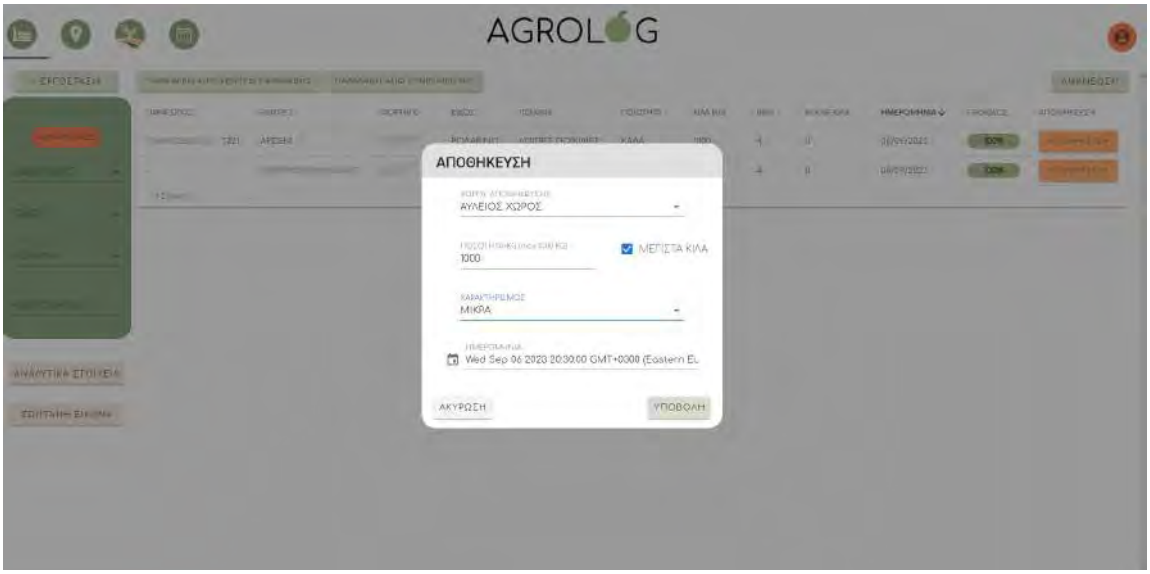

Figure 3.22: AGROLOG Factory Logistics storage form

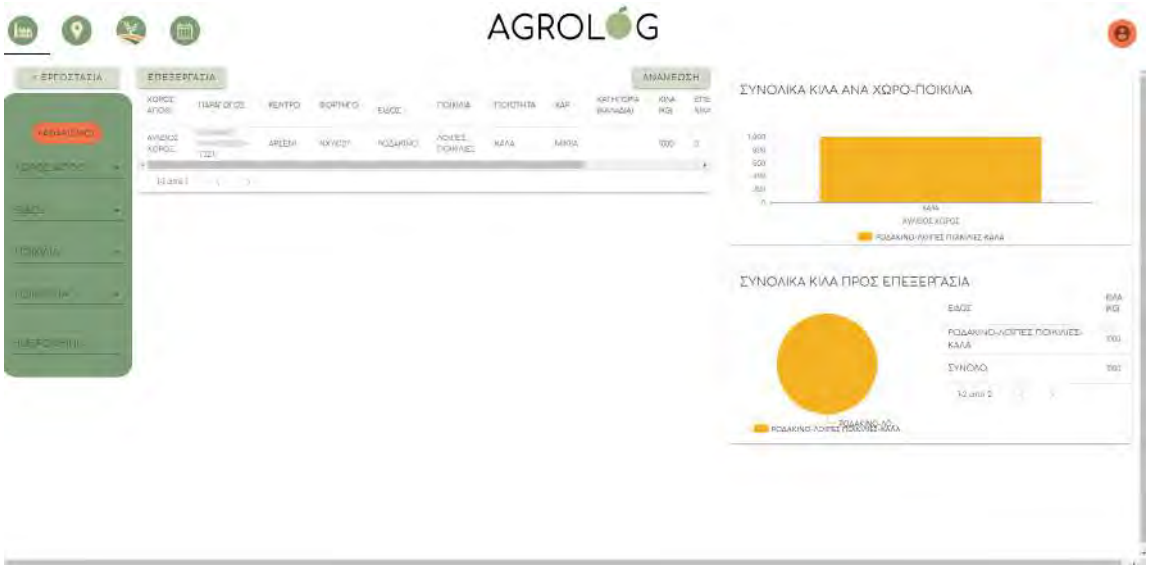

Figure 3.23: AGROLOG Factory Storage after storing crops

5. **Process of the Crop**

   The final step is the process the stored crops. To do that the user must first select the crop, or crops, that will be processed as shown in figure 3.24. After that, a form must be filled out stating the customer the product will be produced for, the production line, the crop to be processed, the amount of the selected crop, the packaging, the packaging

category, and type, as well as the packaging quality as depicted in figure 3.25. Finally, the user has an overview of the final products where he can check for potential errors before finalizing the submission. The process view after a crop has been processed is depicted in figure 3.26.

Figure 3.24: AGROLOG Factory Process form selecting crops from storage

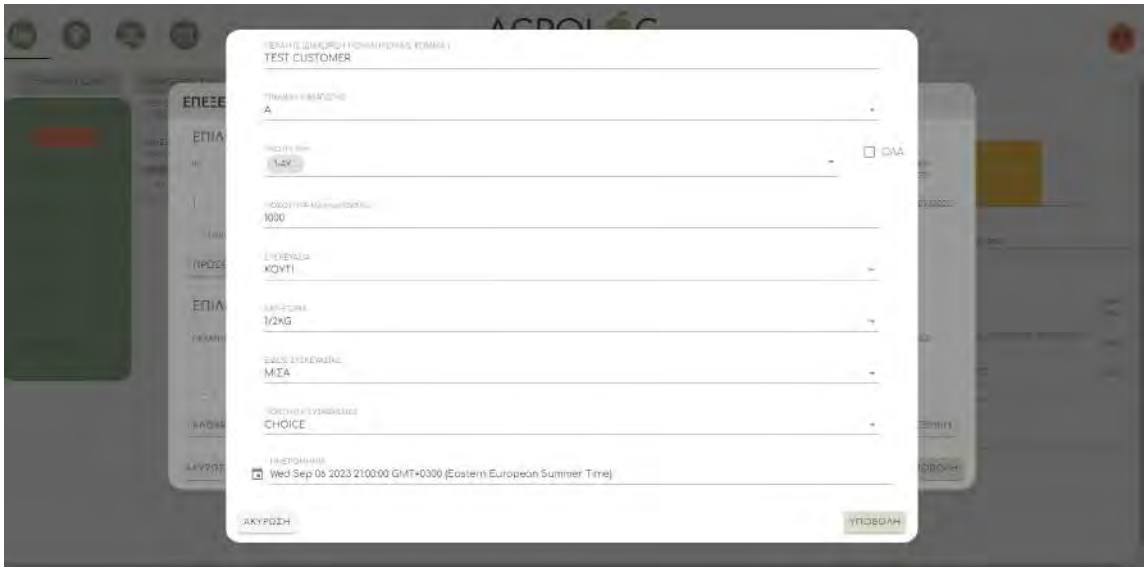

Figure 3.25: AGROLOG Factory Process form about the final products information

Figure 3.26: AGROLOG Factory Process after crops have been processed into products

When the user clicks on the product the product history is displayed displaying all of the intermediate steps the crop passed through as shown in figure 3.27.

Figure 3.27: AGROLOG Factory Process product History

Chapter 4

Blockchain

4.1 Blockchain Technology

Blockchain technology is a distributed ledger database mechanism that allows transparent information sharing within a business network. A blockchain stores data in blocks linked together in a chain, hence the name. The data is chronologically consistent because you cannot delete or modify the chain without consensus from the network. As a result, you can use blockchain technology to create an unalterable or immutable ledger for tracking orders, payments, accounts, and other transactions. The system has built-in mechanisms that prevent unauthorized transaction entries and create consistency in the shared view of these transactions.[15]

4.1.1 History

The idea of blockchain was first proposed by cryptographer David Chaum in a 1982 dissertation named ”Computer Systems Established, Maintained, and Trusted by Mutually Suspicious Groups”. Then further work was described in 1991 by Stuart Haber and W. Scott Stornetta where they wanted to implement a tamperproof timestamp system. In 1992 Haber, Stornetta, and Dave Bayer incorporated Merkle trees into the design enhancing the efficiency of the system.[2]

In 2008, an individual or group going by the name ”Satoshi Nakamoto” introduced the concept of a decentralized blockchain. Nakamoto enhanced this design by employing a method similar to Hashcash for timestamping blocks without needing a trusted signature. Additionally, a difficulty parameter was introduced to maintain a consistent block addition rate. In

2009, this design was incorporated into the foundation of the Bitcoin cryptocurrency. While Satoshi Nakamoto's initial paper referred to "block" and "chain" separately, the term "blockchain" became widely accepted as one word by 2016.[2]

4.2 Strucure and Design

Blockchain is a digital ledger. It's made up of records, known as blocks, which document transactions across numerous computers. Once a block is added, it can't be changed without modifying all the following blocks. This structure lets users independently and cost-effectively verify and audit transactions. The blockchain database operates on its own, facilitated by a peer-to-peer network and a shared timestamping system.[2]. The figure below displays a Blockchain formation. The main chain (black) consists of the longest series of blocks from the genesis block (green) to the current block. Orphan blocks (purple) exist outside of the main chain.

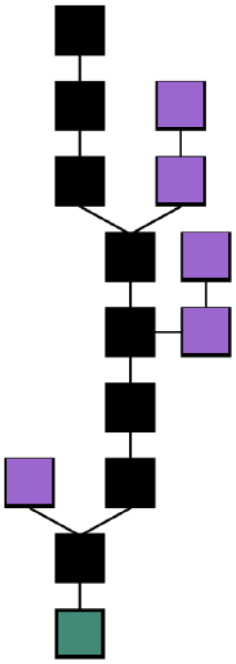

Figure 4.1: Blockchain Formation [2]

Generally, the blockchain consists of several layers such as :

- data (blocks, transactions)
- consensus (proof of work, proof of stake)

- application (smart contracts/decentralized applications)

4.2.1 Blocks

Blocks contain groups of verified transactions, which are then hashed and structured into a Merkle tree. Every block carries the cryptographic hash of its preceding block, creating a connected chain. This method ensures the authenticity of each block, tracing back to the very first one, termed the genesis block. For added security and to validate the information within, blocks are typically digitally signed.[2]

At times, multiple blocks can be created at once, leading to a temporary split or fork. Every blockchain not only relies on a secure hash history but also has a specific method to rank different history versions. The one with a higher rank is chosen over the rest. Blocks that don't make it to the chain are termed orphan blocks. The peers maintaining the database occasionally have varying history versions. They retain the top-ranked version they know. When a peer gets a version with a higher rank (typically the previous version with an added block), they update or replace their database and share this update with others. It's important to note that there's no absolute assurance that a specific entry will always stay in the most accepted history version. Blockchains are designed to build the rank of newer blocks on older ones, and there's an incentive to add new blocks rather than modify existing ones. [2]

4.2.2 Decentralization

The decentralized nature of blockchain networks allows storage of data across the network thus eliminating the risks of storing data in a centralized database utilizing ad hoc message parsing and distributed networking. To ensure security blockchain employs the use of public-key cryptography while data stored in the network is generally considered incorruptible. Therefore while blockchain is a decentralized network it provides great security against malicious actions. [2]

4.2.3 Public and Private Blockchains

Blockchain networks can be divided into two categories, public and private. Public blockchains have no access restrictions and anyone can access the network and invoke transactions while also participating in the consensus protocol. Private blockchains, on the other hand, are per-

missions, and access is restricted and only available through invitation from the network administrators. [2]

4.2.4 Consensus

The consensus problem is fundamental in distributed computing where multiple processes must agree. One approach is for all processes to agree to a majority value which requires one more than half of available votes (where each process is given a vote). However, one or more faulty processes may tamper with the election process so that consensus may not be reached or reached incorrectly. The decentralized nature of blockchain calls for a consensus protocol so that the peers of the network come to a common agreement. This way, consensus algorithms achieve reliability in the Blockchain network and establish trust between unknown peers in a distributed computing environment. Essentially, the consensus protocol makes sure that every new block that is added to the Blockchain is the only version of the truth that is agreed upon by all the nodes in the Blockchain.[16]

4.2.5 Smart Contracts

A smart contract is a self-executing contract with the terms of the agreement between buyer and seller being directly written into lines of code. It exists on a blockchain platform and automatically enforces and executes the terms of a contract when certain conditions are met. Essentially, smart contracts allow for transactions and agreements to be carried out without the need for intermediaries, making processes more transparent, secure, and efficient.

4.3 Key Features of Blockchain

Blockchain provides the following key benefits :

1. **Decentralization**: One of the core advantages of blockchain is its decentralized nature. Instead of relying on a central authority or intermediary, blockchain allows for a peer-to-peer network where multiple participants validate and maintain the integrity of the system. This decentralization enhances transparency, security, and resilience.

2. **Immutable and Transparent**: Blockchain provides an immutable and transparent ledger. That means that any transactions committed to the ledger cannot be altered

thus providing full transparency and preventing fraudulent actions.

3. Enhanced Security: The cryptographic techniques employed by Blockchain ensure secure data and transactions. Each block in the chain is linked to the previous block through cryptographic hashes, making it highly challenging for malicious actors to tamper with the data.

4. **Improved Traceability and Auditability**: Blockchain's transparent and immutable nature allows for improved traceability and auditability of transactions or events. This is particularly valuable in supply chain management such as the Food Supply Chain.

5. **Data Integrity and Ownership**: Blockchain provides a means to securely record and verify ownership or authenticity of assets, digital identities, or intellectual property rights. It enables individuals to have greater control and ownership over their data and reduces the risk of unauthorized modifications or unauthorized use.

4.4 Blockchain Technology in Food Supply Chain Systems

The vast majority of traditional logistic information systems in Agriculture and Food Supply Chains merely track and store orders and deliveries, without providing features such as transparency, traceability, and auditability. While Internet of Things (IoT) devices make remote monitoring of the Food Supply Chains more feasible, the majority of these IoT solutions still rely on centralized infrastructures that lack transparency and fail to prevent data tampering [17]

To address these issues, the use of blockchain technology is proposed as a solution. Blockchain's decentralized nature, along with other benefits such as transparency and data immutability, makes it an ideal candidate for transforming the Food Supply Chain industry. Each step, from production to distribution, can be recorded on the blockchain, ensuring transparency and allowing easy traceability. This level of visibility not only helps identify and address issues such as contamination or fraud but also enables faster and more targeted recalls in the event of food safety concerns. In addition, the incorporation of data from IoT devices such as temperature, humidity, and location provides accurate verifiable information from trusted sources that is tamperproof.

The decentralized and transparent nature of blockchain, combined with the capabilities of

IoT devices, can revolutionize the industry by providing transparency, traceability, and trust throughout the entire Food Supply Chain ecosystem.

Chapter 5

System Archtecture

5.1 Hyperledger Fabric

Hyperledger Fabric is an open-source enterprise-grade **permissioned** distributed ledger technology (DLT) platform, designed for use in enterprise contexts, that delivers some key differentiating capabilities over other popular distributed ledger or blockchain platforms.[18] Hyperledger Fabric offers a permissioned blockchain platform, originally contributed by IBM and Digital Asset, providing a highly modular infrastructure including configurable consensus, execution of Smart Contracts called ”Chaincode” as well as membership services. The Hyperlegder Fabric project belongs under the Linux Foundation’s Hyperledger umbrella that started in 2015. Some of Hyperledger’s key concepts are analyzed in the following sections.

5.1.1 Ledger

A ledger is one of the most fundamental concepts in Hyperledger Fabric. The ledger is used to store important information about business objects including not only the object’s attributes and current values but also a history of transactions that resulted in these values. The ledger is composed of two distinct parts, a world state, and a blockchain as shown in figure 5.1.

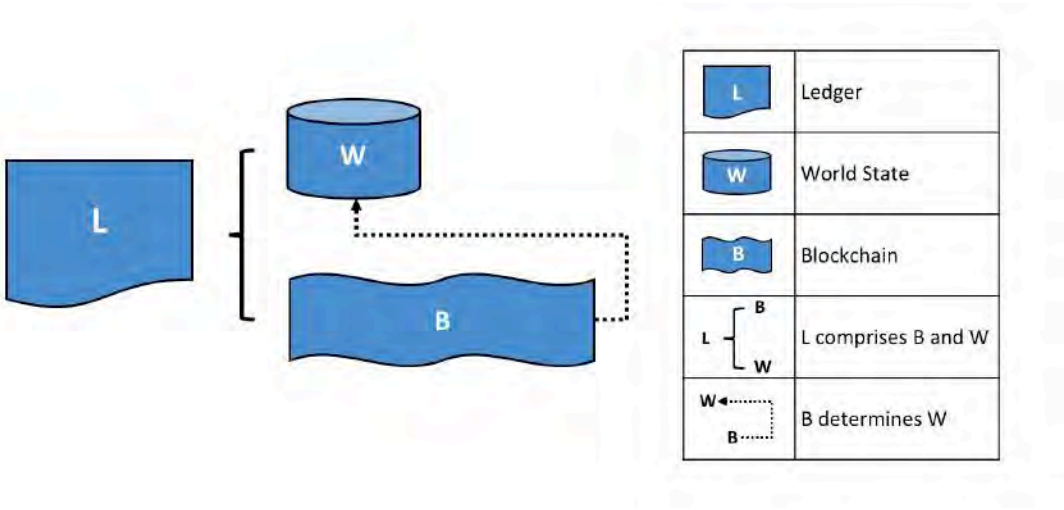

Figure 5.1: A Ledger L compromised of Blockchain B and Wolrd state W [3]

The **world state** acts as a database that holds the current values of the objects. The world state makes it easier to access an object's values directly instead of traversing the transaction history to determine the current state of the object. The world state is often accessed and changed frequently since states can be created, updated, read, or deleted. The world state is implemented as a database that efficiently manages, queries, and retrieves ledger states. The Ledger states are stored as key-value pairs as depicted in figure 5.2.[3]

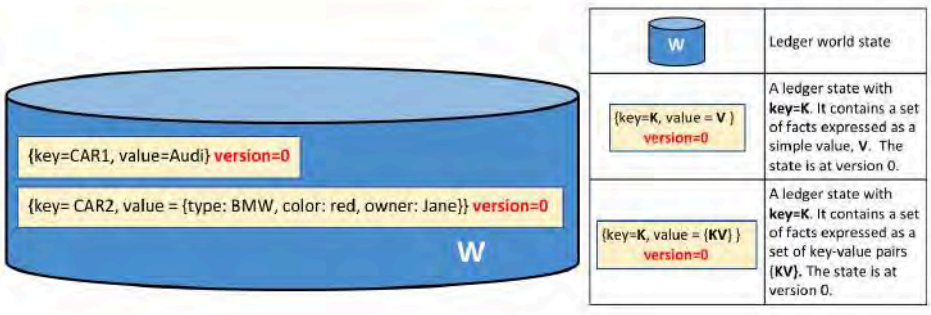

Figure 5.2: A world state containing two ledger states, stored as key-value pairs [3]

The **blockchain** on the other hand is a transaction log that stores the changes that occurred and resulted in the current world state. The blockchain is useful as it provides insight into the changes that have happened. The blockchain differs significantly from the world state since it is **immutable** and once written it cannot be modified. The blockchain consists of interlinked blocks, where each block stores a transaction. Transactions represent queries or updates to

the world state thus providing insight as to the changes that occurred to the ledger state. Each block header contains a hash that links it directly to the previous block so that all transactions are sequenced and cryptographically linked together. In contrast to the world state the blockchain is commonly implemented as a file and appending to it is the most frequent action. The first and most important block of the blockchain is called the **genesis block** and does not contain any transaction data but instead, it contains a configuration transaction containing the initial state of the network channel as shown in figure 5.3.[3]

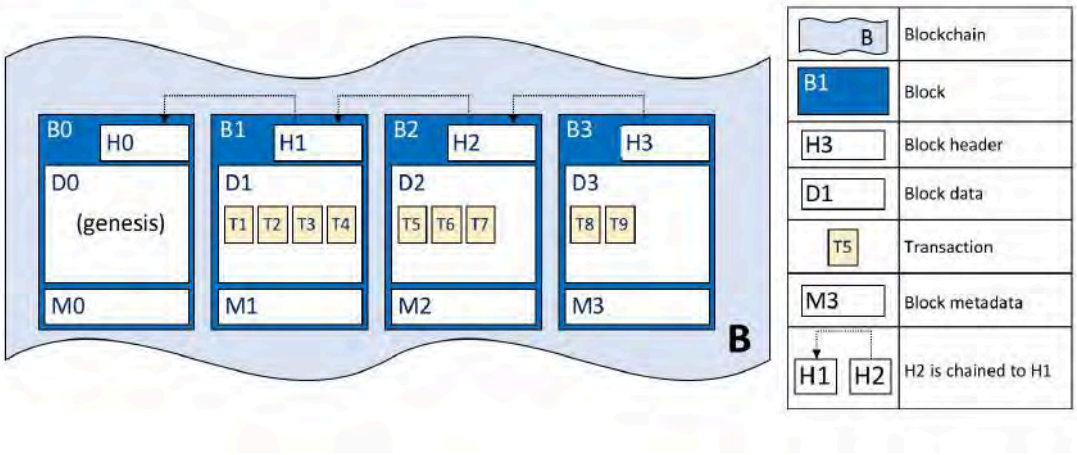

Figure 5.3: A blockchain B containing multiple blocks as well as the genesis block B0 [3]

5.1.2 Peers

Peers is another fundamental component of a Hyperledger Fabric blockchain network. Peers manage ledgers and smart contracts and in current versions also manage transaction proposals and endorsements. A peer hosts instances of both the chaincode and the ledger but also can host more than one ledger or chaincode at a time as shown in the figure below. Peer P1 hosts 2 ledger instances L1 and L2. It also hosts instances of chaincodes S1 and S2. Ledger L1 is accessed using chaincode S1 whereas ledger L2 is accessed using either chaincode S1 or S2 as shown in figure 5.4.[19]

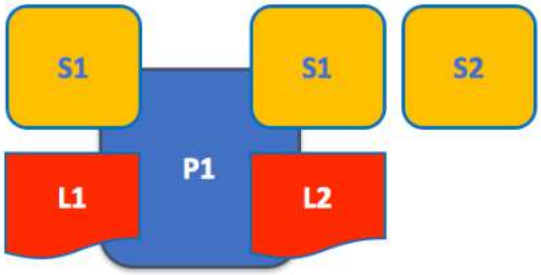

Figure 5.4: Peer P1 hosting ledgers L1, L2 and chaincodes S1,S2 [3]

5.1.3 Ordering Service

Hyperledger Fabric consists also of nodes called orderers, also known as ”orderer nodes”, which form an **ordering service**. Orderers are responsible for providing a total order for published transactions, promoting finality, and guaranteeing that all ledgers are synchronized. Orderers also maintain a list of organizations known as the ”consortium” that are allowed to create channels on the network. Moreover, orderers enforce essential access control for these channels restricting who can read, write, or configure them. [20]

5.1.4 Network Structure

A blockchain network is a technical infrastructure that provides ledger and smart contract (which are packaged as part of a “chaincode”) services to applications. Primarily, smart contracts are used to generate transactions, which are distributed to every peer node in the network where they are immutably recorded on their copy of the ledger. The users of applications might be end users using client applications or blockchain network administrators. In most cases, several organizations come together to form a channel on a blockchain network on which transactions occur and where permissions are determined by a set of policies agreed upon during the creation of the channel by the participating organizations. [4]‘

In the below figure 5.5 a sample network is presented consisting of three organizations R0, R1, and R2. The policies of the network are determined by the organizations who form the network for example, who can add a new organization. The three initial organizations have agreed on the creation of a network. This network consists of a network configuration, CC1, which they have all agreed on and lists the definition of each organization as well as

the policies and the role of each organization on the channel. On this particular channel, R1 and R2 join one peer each, P1 and P2, to channel C1 while R0 owns O which is the ordering service of the channel. All of these nodes contain a copy of the ledger, L1, where the transactions are recorded. Each peer also contains a copy of the state database, S5, excluding the ordering service O. P1 and P2 interact with the network through the applications A1 and A2. Finally, each organization has a Certificate Authority, CA 0 to 2, that generates the necessary cryptographic materials for the nodes, admins, organization definitions, and applications of its organization. [4]

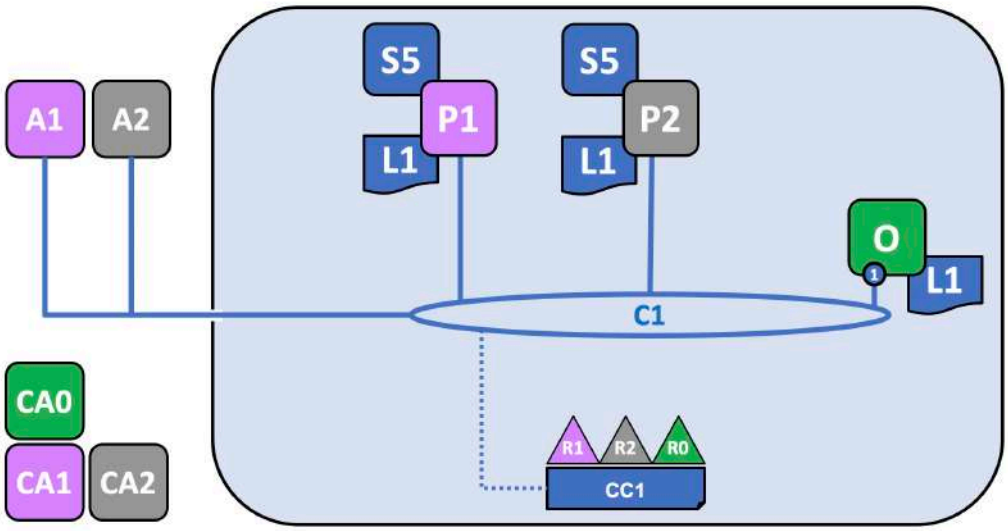

Figure 5.5: Sample Hyperledger Fabric Network [4]

The terms network and channel are sometimes used interchangeably but refer to different concepts within Hyperledger Fabric. The network consists of multiple organizations, each having its own set of peers, orderers, and certificate authorities. These peers and other network components work together to create and maintain the shared ledger. On the other hand, a channel is a private communication pathway within a Hyperledger Fabric network. It allows a subset of network participants to create and maintain a separate ledger that is only visible and accessible to those participants. Channels enable different parties to transact privately and securely without the need for all network participants to see or validate every transaction. Each channel has its own ledger, chaincode (smart contracts), and access control policies. In conclusion, a channel is a subset of participants within that network who collaborate on a separate ledger. A depiction of this is shown in figure 5.6

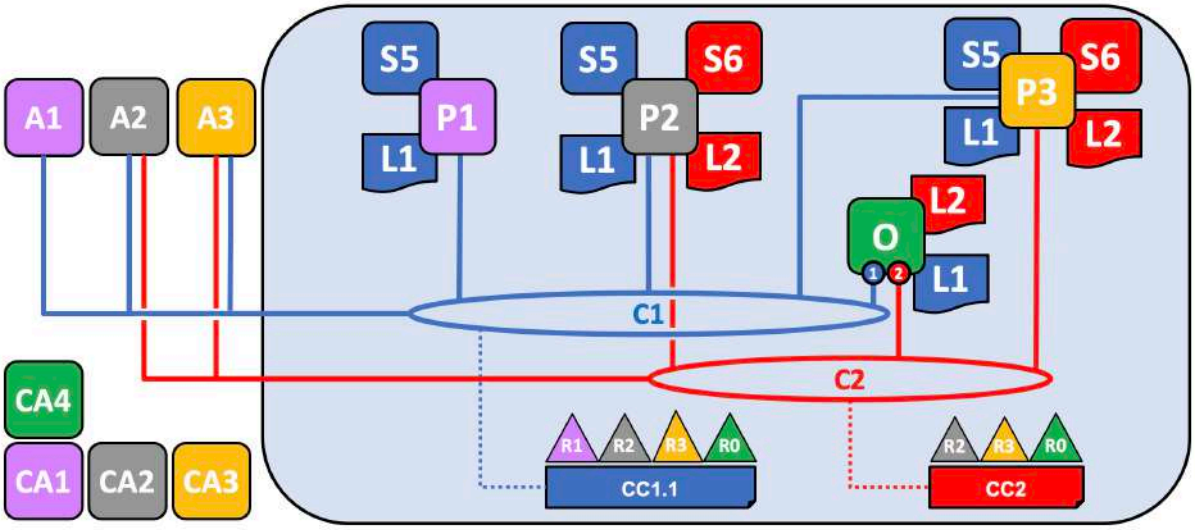

Figure 5.6: Network with multiple channels [4]

5.1.5 Transaction Flow

Transaction Flow describes the steps a proposed transaction goes through until it is submitted to the blockchain. The scenario describes two clients, A and B, who are buying and selling radishes and wish to transact with each other. [5] The transaction goes through the following steps :

1. **Client A initiates the transaction**

   Client A wishes to purchase some radishes and therefore initiates a transaction that targets peerA and peerB, who represent Client A and Client B respectively. In this step, the transaction proposal, which is nothing more than a request to invoke a chaincode function with specific parameters, is constructed.

2. **Endorsing peers verify the signature and execute the transaction**

   The endorsing peers must first verify that (1) the transaction proposal is well-formed, (2) the validity of the signature, (3) the transaction proposal has not already been submitted and (4) the submitter is authorized to perform operations on the channel. After the verification of the above the chaincode is then executed, and the result is passed back as a ”proposal response”.

3. **Proposal responses Inspection**

   The peer then verifies that the proposal responses sent back match each other before submitting the transaction

4. **The Peer assembles endorsements into a transaction**

   The peer sends the transaction proposal as well as the proposal response to the ordering service. The ordering service receives the transactions, orders them, and creates the transaction block of the ledger.

5. **Transaction Validation and Commit**

   The ordering service sends the transaction block to each peer on the channel. The transaction contained within the block is validated to ensure the endorsement policy is fulfilled and that the ledger state was not updated since the read operation of the transaction which would result in false data. The transaction is then marked as valid or invalid.

6. **Ledger Update**

   Each peer appends the block to the channel's chai. The results of the write operation are committed to the world state database. Finally, an event is triggered to notify client applications that the block has been appended to the chain, as well as a notification of whether the transaction was validated or not.

Figure 5.7 demonstrates the transaction flow.

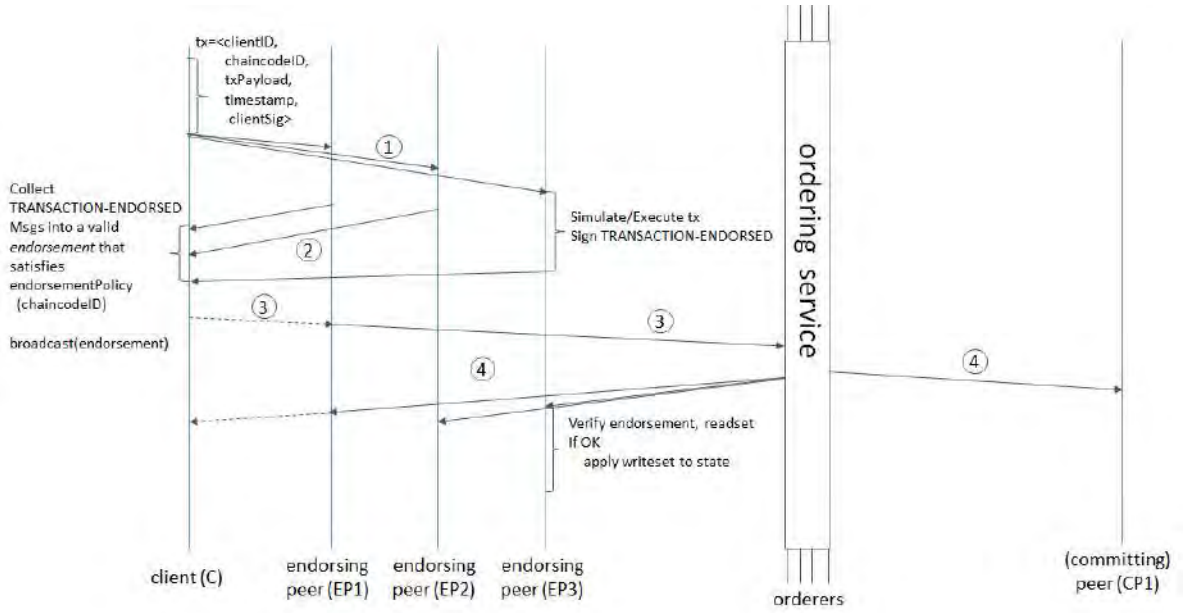

Figure 5.7: Hyperledger Fabric Transaction Flow [5]

5.1.6 Hyperledger Fabric Gateway

Hyperledger Fabric Gateway was introduced in version 2.4 and provides a simple API that is used for submitting transactions within a Fabric network. The Gateway service manages the evaluation of proposals, endorsements, and transaction submissions making the client application development easier. SDKs in three languages, Go, Node, and Java, are provided in order to provide a programming interface for the developers. Among other things, the Gateway service offers retry attempts, error handling, and timeouts. Finally, it provides a simplified API for client applications to listen for events that occur in the blockchain network.[21]

5.2 Docker

Docker(5.8) is a platform that allows the packaging of applications in packages called containers which are lightweight and can work efficiently in different environments and isolation. It was first introduced in 2013 and was developed as a way to isolate an app from its environment thus eliminating the common issue of ”it works on my machine”. [6]

Figure 5.8: Docker Logo [6]

5.2.1 Docker Containers

A Docker container is a unified software package that bundles code and its dependencies, ensuring the application operates seamlessly across different computing environments. Docker container images are streamlined, software packages encompassing all essentials for running an application: from code and runtime to system tools, libraries, and configurations. Compared to traditional virtual machines containers are more portable and efficient since they virtualize the operating system instead of the hardware.

Containers are an abstraction at the app layer and thus share the OS kernel with other containers as well. For that reason, containers take less space than virtual machines and can handle more applications. Virtual machines on the other hand are an abstraction of physical hardware. Using the hypervisor multiple virtual machines can run on a single machine, where each virtual machine contains a full copy of an operating system making it take up more space as shown in figure 5.9.[7]

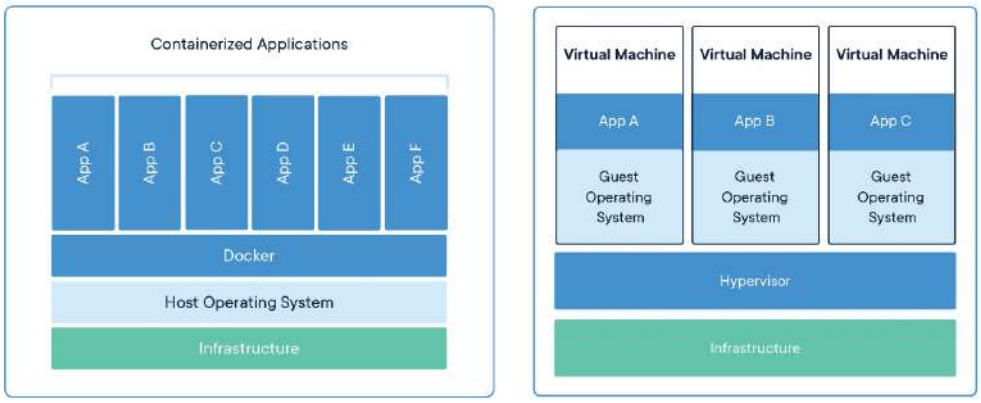

Figure 5.9: Container and Virtual Machine architecture [7]

5.2.2 Docker Swarm

Docker Swarm is a management tool that runs on Docker applications and provides native clustering functionality for Docker containers. Each container within the Docker Swarm can be accessed by nodes belonging to the same cluster in that way providing a way of node communication. Nodes can also dynamically enter and leave the Swarm. Finally, Swarm among other things provides decentralized access, high security, autoload balancing, and high scalability

5.3 NITOS

NITOS Future Internet Facility is an integrated facility, developed by NITLab, supporting heterogeneous testbeds focusing on experimentation and research in the fields of wired and wireless networks. NITOS has been used hundreds of times by researchers worldwide and provides around-the-clock remote access to the research community. It is worth mentioning that the testbed is based on open-source software allowing the design and implementation

of new algorithms where protocols and applications can be easily evaluated in real-world scenarios. The NITOS architecture is shown in figure 5.10.[8] NITOS provides the following experimental components:

- **A wireless experimentation testbed**, consisting of 100 powerful nodes featuring multiple wireless interfaces and heterogeneous wireless technologies such as Wi-Fi, WiMAX, LTE, and Bluetooth

- **A Cloud infrastructure**, consisting of 7 HP blade servers and 2 rack-mounted ones providing 272 CPU cores, 800 GB of RAM, and 22TB of storage capacity. Through the HP 5400 series modular Openflow switch, network connectivity of 10GB Ethernet connectivity amongst the cluster's modules and 1Gb between the cluster and GEANT is achieved.

- **A wireless sensor network testbed**, consisting of a controllable testbed deployed in UTH's offices, a city-scale sensor network deployed in Volos city, and a city-scale mobile sensing infrastructure that relies on bicycles of volunteer users. All the sensors were developed by UTH and use several wireless technologies for communication such as ZigBee, Wi-Fi, LTE, Bluetooth, and IR.

- **A Software Defined Radio (SDR) testbed**, that consists of Universal Software Radio Peripheral (USRP) devices attached to the NITOS wireless nodes

- **A Software Defined Networking (SDN) testbed**, consisting of multiple switches using OpenFlow technology enabling experimentation with switching and routing networking protocols.

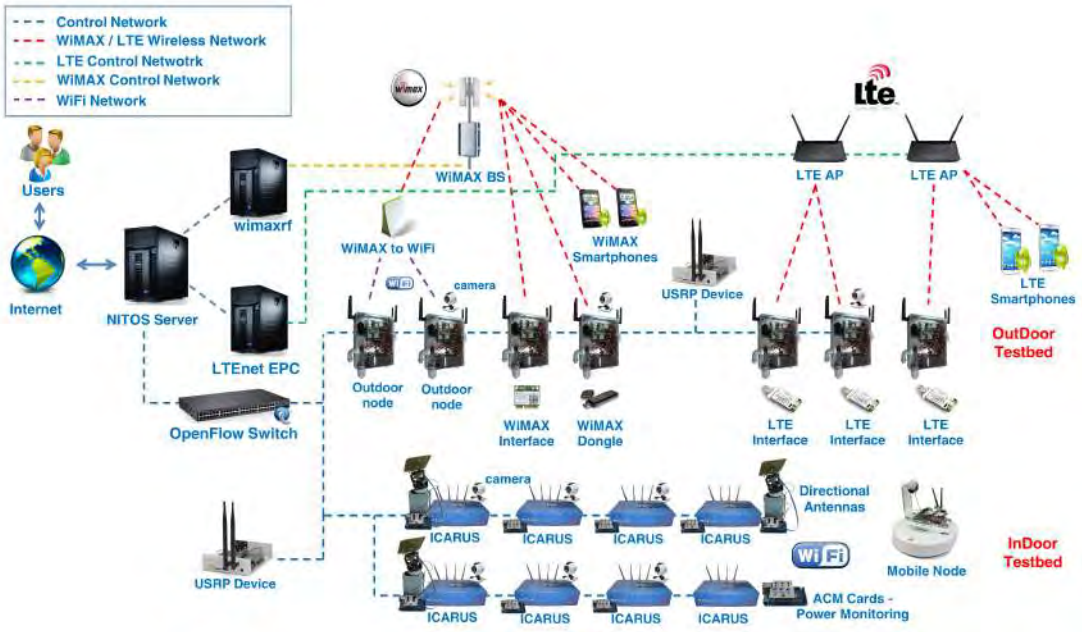

Figure 5.10: NITOS architecture [8]

NITOS is compromised of three deployments: **the Outdoor Testbed**, **the Indoor RF Isolated Testbed**, and **the Office Testbed**.

5.3.1 Outdoor Testbed

The NITOS Outdoor Tesbed deployment consists of 50 powerful nodes featuring multiple wireless interfaces. It is deployed at the exterior of a University of Thessaly (UTH) campus building. The nodes are deployed on the roof of the building, 25 of them in a grid topology and 25 of them spread across the building floors as depicted in figures 5.11 and 5.12.[8]

Figure 5.11: Outdoor Testbed [8]

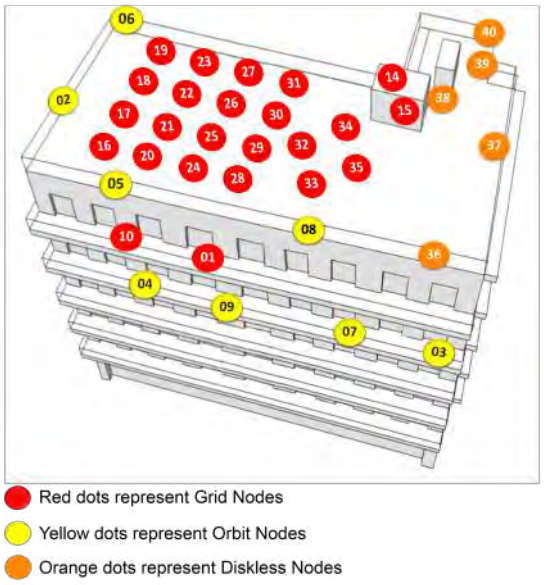

Figure 5.12: Outdoor Testbed Topology [8]

5.3.2 Indoor RF Isolated Testbed

NITOS RF Isolated Indoor Deployment is composed of 50 Icarus nodes featuring multiple wireless interfaces, Wi-Fi, WiMAX, and LTE, and is deployed in an isolated environment at a University of Thessaly's campus building. The Icarus nodes are placed symmetrically around the isolated environment forming a grid topology as shown in figures 5.13 and 5.14. The distance amongst the nodes is fixed at 1.2 meters and the height level is identical for all of them as well. This results in creating a uniform environment.[8]

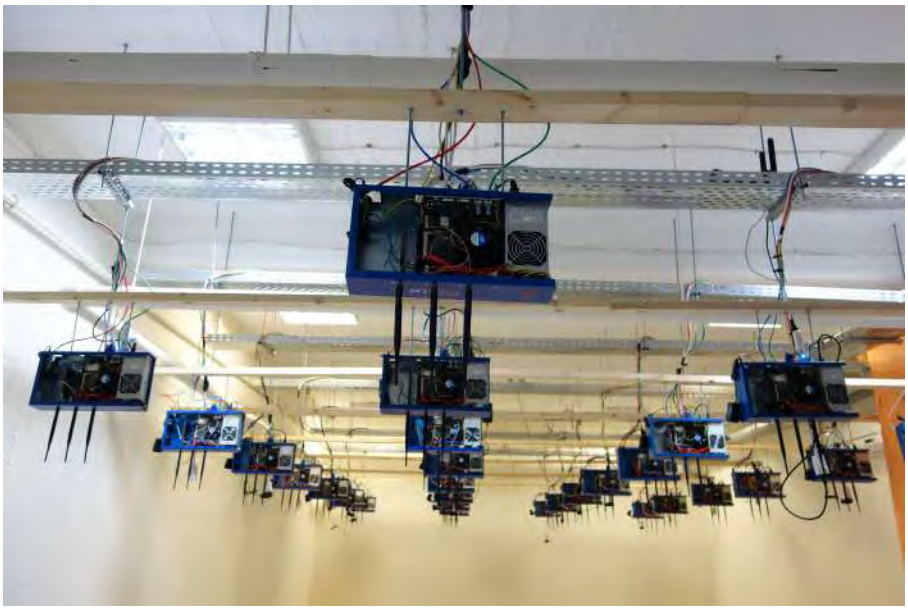

Figure 5.13: Indoor RF Isolated Testbed [8]

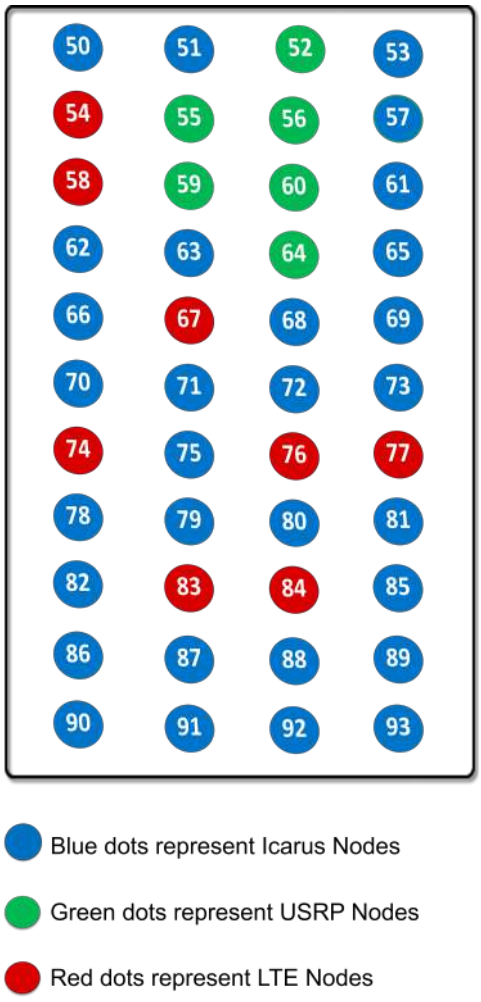

Figure 5.14: Indoor RF Isolated Testbed Topology [8]

5.3.3 Office Testbed

The Office Testbed consists of 10 powerful second-generation ICARUS nodes. The nodes feature heterogeneous technologies, such as WI-FI, WiMAX, and LTE, allowing for experimentation and execution under real-life scenarios while providing a deterministic office environment. The testbed is located at the premises of CERTH's ITI laboratory in the city of Volos and is spread throughout the entire 3rd floor and is depicted in figure 5.15.[8]

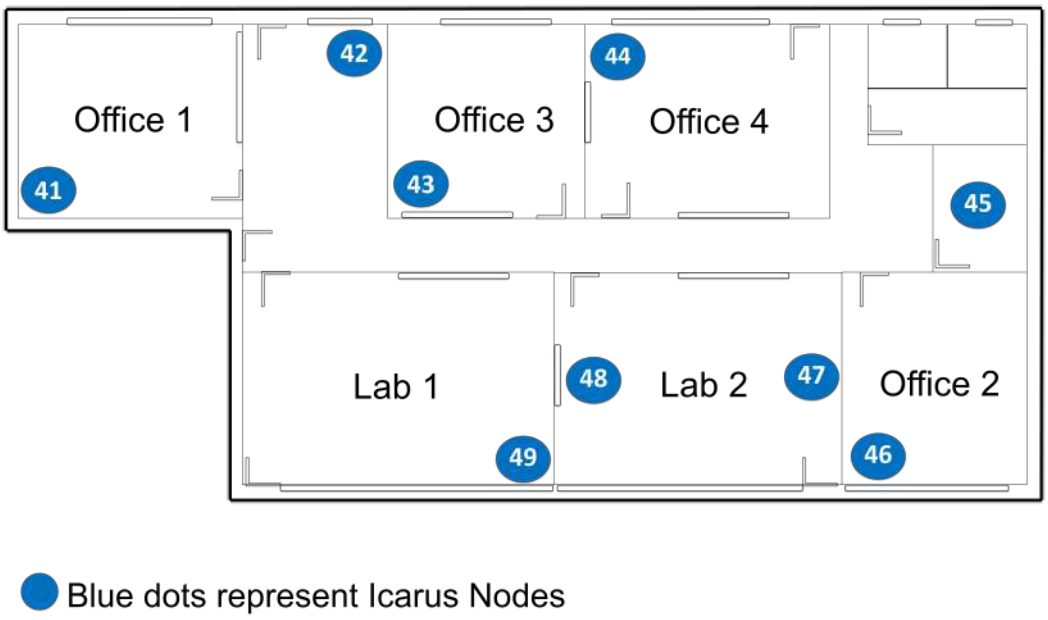

Figure 5.15: Office Testbed Topology [8]

5.3.4 How was NITOS used

In this Thesis, NITOS was used as an experimental tool to help set up a blockchain network consisting of multiple distinct nodes. The deployed blockchain network consisted of two organizations and one orderer. For this purpose, four nodes were used hosting four peers of the blockchain network, two peers per organization in particular. One of the nodes also hosted the orderer service required. Its around-the-clock remote access and network capabilities contributed to easier development of the blockchain network thus making it a crucial part of this research.

Chapter 6

Blockchain Network setup

6.1 Network Setup

In this section, the process of creating a blockchain network is discussed. The network consists of one orderer organization and 2 peer organizations. The orderer organization contains a single ordering service node (orderer). The peer organizations, Org1 and Org2, contain two peers each.

6.1.1 Hosts

In this research 4 host nodes were used, hosts 1 to 4, to set up the blockchain network. The steps for bringing up the blockchain network are the following :

1. **Create an overlay network using Docker Swarm**
   On host 1 we initially create a docker swarm network and then generate a token that the rest of the hosts can use to join this network as managers. After the hosts have joined the Swarm, an overlay network can be created allowing docker containers, running on each host, to communicate with each other seamlessly.

2. **Bring up the Hosts**
   By using docker-compose we can easily bring up docker containers running on each host. Host 1 contains 3 containers. These containers are the CLI container which is used to issue commands to the blockchain network for example creating a channel, the peer0.org1 container which is the first peer of Org1 and the orderer container. The other hosts bring up containers running peer1.org1, peer0.org2, and peer1.org2 respectively.

3. **Bring up the channel and join all peers**

   The next step is to generate the channel's genesis block. This is accomplished through the CLI running on host 1. Once the genesis block has been generated the peers can now join the channel.

Once the blockchain network and channel have been set up the next step is to install the appropriate chaincode described in the next section.

6.2 Chaincode

Chaincode is a program, written in Go [22], Node.js [23], or Java [24], that implements an interface. The chaincode is executed in an isolated docker container from the peers and its main purpose is the initialization and management of the ledger state. Chaincode is responsible for handling business logic and thus is considered a smart contract. In this section, the chaincode implementation and installation are discussed.

6.2.1 Chaincode Implementation

The chaincode was implemented using the Node.js language. The records created by the chaincode and inserted into the blockchain are mentioned as ”assets”. Initially, an asset class is implemented that defines the asset's class fields. The most important field is the ID since it is used as a unique identifier to query the blockchain and access a particular asset. After the class has been implemented the functions for creating, reading, updating, and deleting assets are implemented. On creation a new asset with a unique ID is created, if the ID already exists the operation fails. Reading of an asset is achieved through querying the blockchain using the asset's ID, the process fails if the ID does not exist. An update of an asset is also achieved through two operations, read and then update. The asset is first read using the read operation and an ID and then the desired fields are updated to new ones while preserving the rest of the fields. Finally, deletion is a simple operation that only requires the asset's ID.

6.2.2 Chaincode Installation

Once the network and channel have been set up the next step is to install the desired chaincode on the channel. The first step is to package the chaincode into a compressed file.

Once the packaging has been completed the next step is to install the chaincode to the participating peers of the channel. This is achieved through the CLI that runs on host 1. Once the chaincode has been installed each organization, through an admin peer, must approve the chaincode definition. When the approval has been completed the final step is to commit the chaincode to the channel using the orderer service. After the installation process has been completed the chaincode is ready to be utilized on the channel.

6.3 Client Application and REST API setup

After the blockchain network and channel establishment, and the chaincode installation the next step is to provide a seamless way to access the network. This is achieved through creating a client application from which users can easily access the network and invoke transactions.

6.3.1 Client Application Implementation

The client application is developed using Node.js. The Fabric Gateway service is used to connect to the blockchain network. To achieve the connection, peer data and private key must be provided as well as the channel and chaincode name.. After the Fabric Gateway service has connected to the network, chaincode functions mentioned in the section above can be invoked providing a seamless way of interacting with the blockchain network.

6.3.2 REST API and Endpoints

To simplify the process of invoking transactions even further, a REST API is provided. REST, Representational State Transfer, is a software architectural style that defines how the architecture of a distributed system should behave. It uses the HTTP protocol for communication. By defining endpoints, the REST API can trigger transactions on the blockchain network. By providing a REST API users that do not have physical access to the hosts running the containers, and thus the blockchain network, can invoke transactions and query the blockchain network. For that purpose, the following endpoints are provided :

1. **createAsset**: **POST** request that creates and adds a new asset to the blockchain. The body of the request includes the asset's fields.

2. **readAsset**: **GET** request that queries the blockchain for an asset with an ID that matches an id that is passed via the URL, the response is the query's result.

3. **updateAsset**: **POST** request that updates an asset's data on the blockchain. The body of the request includes the asset's updated fields.

4. **deleteAsset**: **DELETE** request, that deletes and asset from the blockchain.

5. **getAllAssets**: **GET** request that fetches all assets on the blockchain. The response is the query result.

6.3.3 Transactions Handling

Although the REST API provides an easy and seamless way for external users to access the blockchain network, response times when submitting transactions to the blockchain network can be problematic. An approach to dealing with this obstacle is, instead of waiting for the transaction to complete, immediately send back a response indicating acceptance of the request and then process the request at a later time. To achieve this functionality when receiving a request that mutates the state of an asset, create, update, or delete, the operation is represented as a job and is added to a wait queue until it is time to be processed while the job ID is sent back with the response. The job ID returned can be used with another endpoint, "**jobs/"**, to check on the status of the job and get the transaction results. One by one the jobs from the wait queue are processed and are added to another queue, named the processing queue. To ensure the correct order of transaction execution, for example, an update does occur before an asset creation, jobs whose asset ID is present in the processing queue are not selected from the wait queue. This handling enables faster response times as well as a way to fetch the results of a transaction after it has been completed

Chapter 7

Blockchain Integration to AGROLOG

7.1 Transferring Data from AGROLOG to the Blockchain

Since the AGROLOG app and the blockchain are decoupled from each other a way of syncing data between AGROLOG and the blockchain network must be established. This is achieved easily by using the REST API described in the previous chapter. Through API usage, data submitted to AGROLOG can be submitted to the blockchain network as well without adding any extra overhead. Since data submitted to AGROLOG are accompanied by a UUID(universally unique identifier), this UUID can be used as the asset's ID on the blockchain network. When creating, updating, or deleting data from AGROLOG the appropriate transactions can be invoked via the API therefore submitting the data to the blockchain network as well. By the use of the described API, the AGROLOG app can now integrate the Blockchain technology seamlessly providing a way of assuring data integrity and veracity. By the use of blockchain technology, all data submitted to the AGROLOG app can be verified using a tamper-proof system, eliminating food fraud and adulteration. Moreover, with the transparency of the stages, the raw materials pass through, consumer trust can increase significantly since the consumers can track and view each stage while also maintaining trust thanks to blockchain technology.

7.2 Off-chain Data

The method described above creates and manages the same data in two different places, the AGROLOG database and the blockchain network. Although this might seem counter-

intuitive, it is absolutely necessary in order to ensure that the performance of the AGROLOG application does not degrade. The nature of the blockchain network does not guarantee fast transaction handling since the transaction must pass through multiple stages before its results are committed. Since AGROLOG is an app that requires real-time data this can be problematic. The solution to this issue is to store the data in two places, the AGROLOG database which is used for fast storage and analysis of the data, and the blockchain network which can be later used to ensure the validity and integrity of the data. Since the data is not stored only on the blockchain this method is called 'Off-chain data'. The use of Off-Chain data allows for better overall performance of the app without losing any key features.

7.3 Data Verification Using Blockchain

To verify the integrity of data in the AGROLOG app the blockchain network is used. Upon request, the appropriate data from the blockchain network is fetched and its fields are compared with the fields of the data present in the AGROLOG database. If all of the fields match exactly the verification of the data is successful and an appropriate message is displayed. If even a single field fails to match, verification fails. On failure, an indication is displayed and a message indicating which field caused the failure is displayed, therefore informing the end-user about possible fraud or alteration of data. This is vital since end-users can easily view and verify the data that is displayed. Product managers can use this information to recall a batch of products early or track down the stage where something went wrong or adulteration occurred. Final product consumers can also benefit from this functionality since they can verify the product label thus increasing trust. In figure 7.2 blockchain verification failure is shown. The red icon indicates an error during the verification and the displayed message informs the user of the reason the verification failed. In this example, verification failed due to the fields of the crop quality in the AGROLOG database and the blockchain network did not match. A possible reason for this is that someone altered the crop quality field in the database. In figure 7.1 blockchain verification success is shown. The green icon indicates verification success and the message displayed also informs the user that the data is verified using the blockchain network.

Figure 7.1: Blockchain Verification Success

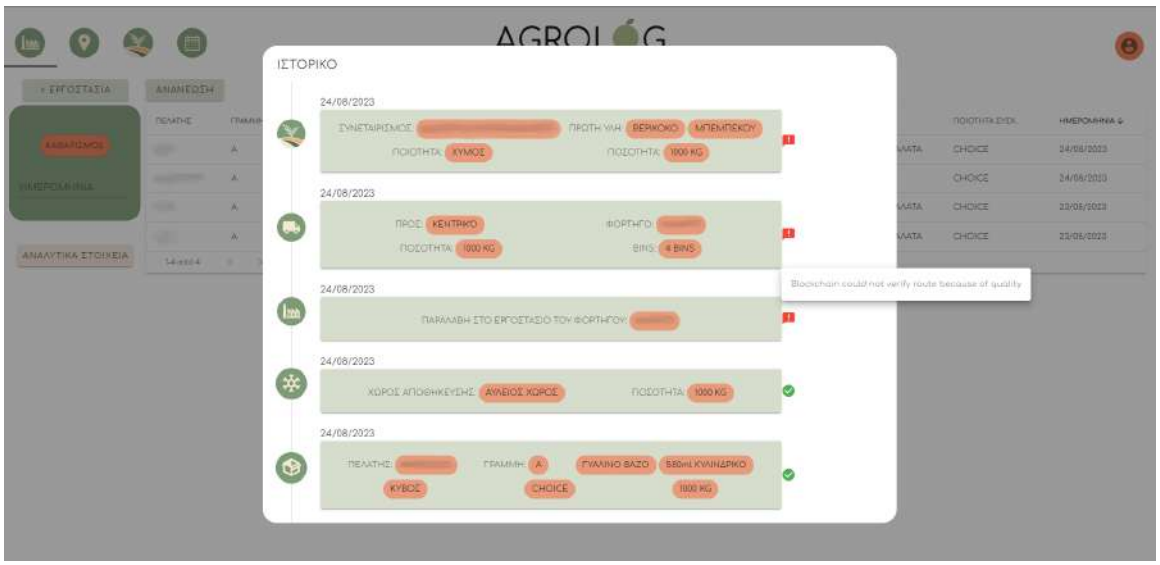

Figure 7.2: Blockchain Verification Failure

7.4 Evaluation

The proposed solution was tested under real-world conditions, with multiple users concurrently submitting data to the AGROLOG app. Leveraging off-chain data storage, the performance and efficiency of the AGROLOG app showed no signs of degradation. This underscores the robustness of the application and its ability to handle concurrent user interactions as intended.

Furthermore, the blockchain network that supported the AGROLOG app performed admirably. Even when accommodating numerous peers and users connecting simultaneously, the transaction completion time exhibited an expected increase. However, it's important to note that this increase did not compromise any of the app's fundamental functionalities or features. The blockchain network's performance was assessed according to the Hyperledger Blockchain Performance Metrics white paper. [25]The evaluation included four key metrics:

- **Read Latency = Time when response received – submit time**, which is defined as the time between when the read request is submitted and when the reply is received.
- **Transaction Latency = (Confirmation time @ network threshold) – submit time**, which measures the time a transaction is submitted, up to the point the result of the transaction is available in the network.
- **Transaction Throughput = Total committed transactions / total time in seconds @ #committed nodes**, which measures the rate at which valid transactions are committed to the network in a defined time period.
- **Read Throughput = Total read operations / total time in seconds**, which measures the number of read operations completed in a defined time period.

The test setup consisted of 4 distinct nodes each running a peer of the blockchain network. The first node of the setup also included the orderer node. The transaction time per number of assets present in the blockchain network is depicted in figure 7.3. The figure demonstrates that the number of documents in the blockchain network does not affect the transaction time. The first transaction shows a greater latency but this is expected since there is a slight overhead of initializing the gateway that is used for the transactions whereas the next transactions reuse the same gateway. Figure 7.4 depicts the read latency of a single asset as well as the read latency of all the assets in the network. The read latency of a single asset remains low as the number of assets in the network does not affect it. On the other hand, there is a slight, although steady, increase in the read latency of all of the assets. This is expected since more and more assets must be fetched and sent over as a response from the blockchain network thus affecting the read latency. Finally, figure 7.5 depicts the Transaction and Read Throughput per number of assets in the network. Both metrics remain relatively steady while increasing the number of assets in the network which is expected.

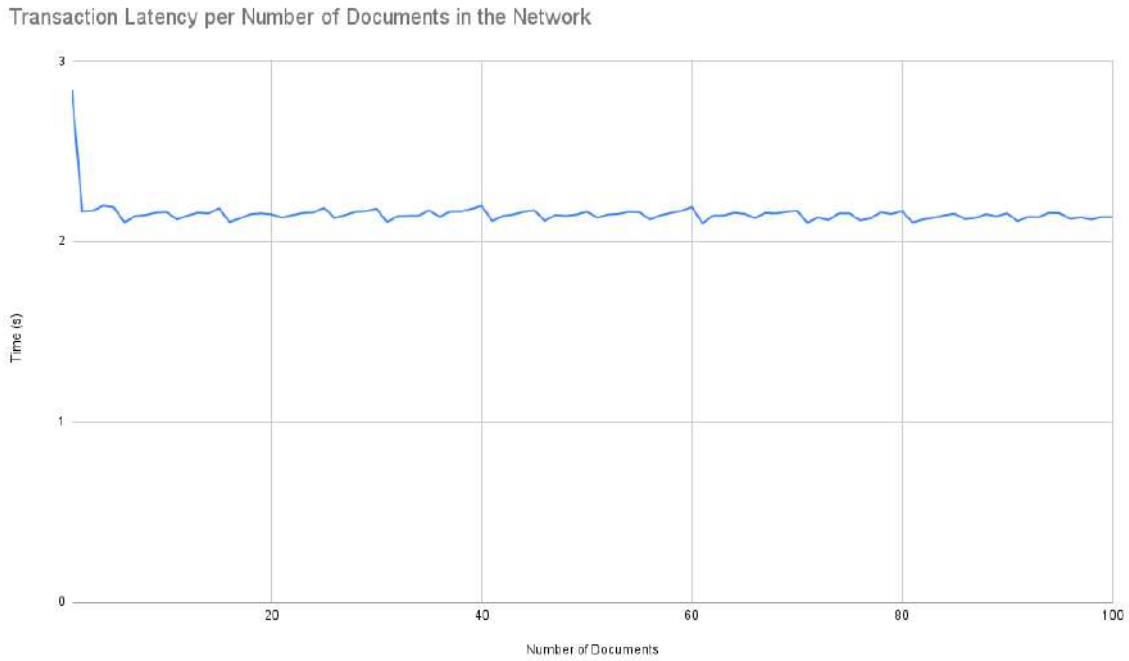

Figure 7.3: Transaction Latency per Number of Assets in the Network

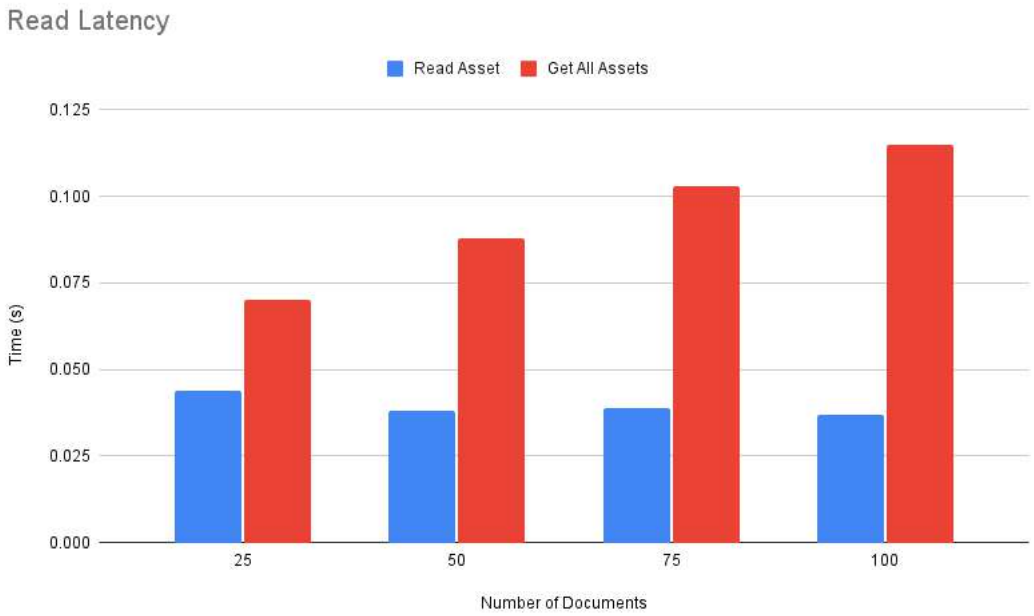

Figure 7.4: Read Latency

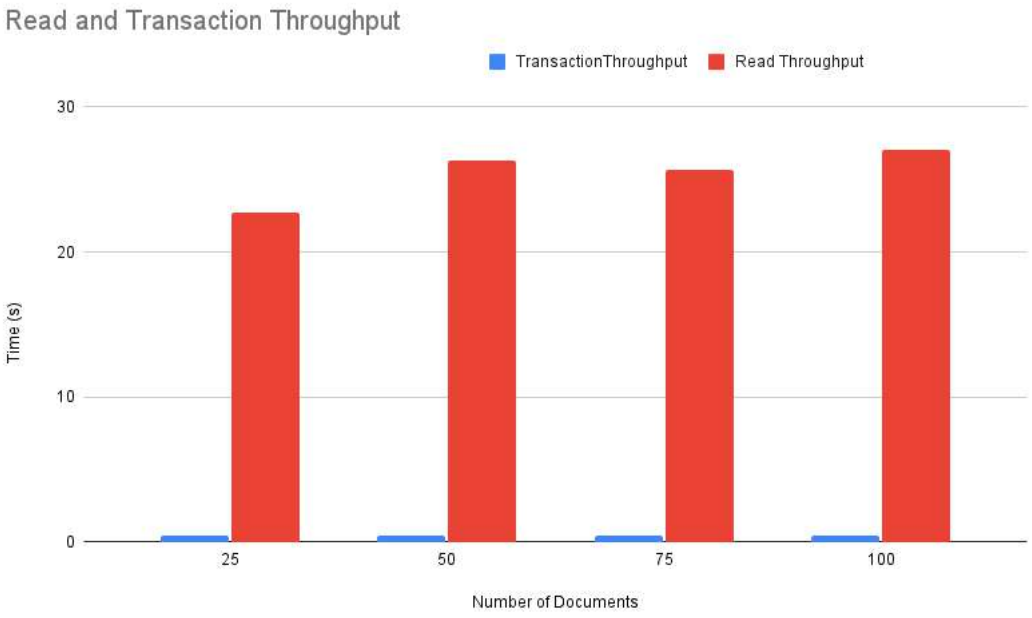

Figure 7.5: Read and Transaction Throughput

In summary, the extensive testing of the solution not only demonstrated the resilience of the AGROLOG app's off-chain data management but also highlighted the blockchain network's scalability and reliability, even under heavy user load.

Chapter 8

Conclusions

8.1 Conclusion and Results

Food Supply Chains are a great aspect of modern society, providing safe food and ingredients. The lack of transparency and integrity present in the Food Supply Chains due to high fragmentation makes it susceptible to fraud and adulteration. To address those issues blockchain technologies can be utilized. Blockchain provides several benefits such as traceability and data integrity making it an ideal candidate. The Blockchain technology was integrated into AGROLOG, a Food Supply Chain tracking app, in order to test the efficiency and practicality of the proposed idea. Through the use of blockchain, the data submitted to AGROLOG can be verified in order to ensure data integrity. Blockchain and smart contract technologies were proven to provide data integrity, veracity, interoperability, transparency, and traceability in Food Supply Chain systems solve many of the major issues while enhancing fraud resistance.

8.2 Future Work

Several aspects of this Thesis can be improved by using more advanced tools or improving existing methods, for example, the submitting of off-chain data. A blockchain event listener can be put into place so that every time an asset is modified on the blockchain, created, updated, or deleted, an event is triggered and the same data can be submitted to the off-chain AGROLOG database. Moreover, advanced orchestration tools, such as Kubernetes can be utilized. Kubernetes services are more portable and productive and provide load balancing

while simplifying container management on multiple hosts allowing for greater scalability. Finally, due to limited resources, our solution was tested on 4 distinct nodes. Future experiments could include more nodes running numerous peers in order to gain a greater view of the network performance under heavy usage.

Bibliography

[1] Mevn stack. https://vteams.com/blog/miscellaneous/mevn-stack/. Date of Access: 20-06-2023.

[2] Blockchain. https://en.wikipedia.org/wiki/Blockchain. Date of Access: 15-06-2023.

[3] Hyperledger fabric ledger. https://hyperledger-fabric.readthedocs.io/en/latest/ledger/ledger.html#. Date of Access: 15-06-2023.

[4] Hyperledger network. https://hyperledger-fabric.readthedocs.io/en/release-2.5/network/network.html. Date of Access: 15-06-2023.

[5] Hyperledger fabric transaction flow. https://hyperledger-fabric.readthedocs.io/en/release-2.5/txflow.html. Date of Access: 15-06-2023.

[6] Docker. https://www.docker.com/. Date of Access: 20-06-2023.

[7] What is a docker container. https://www.docker.com/resources/what-container/. Date of Access: 15-06-2023.

[8] Nitos. https://nitlab.inf.uth.gr/NITlab/nitos. Date of Access: 20-06-2023.

[9] What is the food supply chain? | 5 food supply chain stages. https://www.bluecart.com/blog/what-is-the-food-supply-chain. Date of Access: 15-06-2023.

[10] Nitlab. https://nitlab.inf.uth.gr/. Date of Access: 20-06-2023.

[11] Nuxt.js. https://www.java.com/en/. Date of Access: 20-06-2023.

[12] Vue.js. https://www.java.com/en/. Date of Access: 20-06-2023.

[13] Express.js. https://expressjs.com/. Date of Access: 20-06-2023.

[14] Mongodb. https://www.mongodb.com/. Date of Access: 20-06-2023.

[15] What is blockchain technology. https://aws.amazon.com/what-is/blockchain/?aws-products-all.sort-by=item.additionalFields.productNameLowercase&aws-products-all.sort-order=asc. Date of Access: 15-06-2023.

[16] Consesus (computer science). https://en.wikipedia.org/wiki/Consensus_(computer_science). Date of Access: 15-06-2023.

[17] Miguel Pincheira Caro, Muhammad Salek Ali, Massimo Vecchio, and Raffaele Giaffreda. Blockchain-based traceability in agri-food supply chain management: A practical implementation. In *2018 IoT Vertical and Topical Summit on Agriculture - Tuscany (IOT Tuscany)*, pages 1–4, 2018.

[18] What is hyperledger fabric. https://hyperledger-fabric.readthedocs.io/en/release-2.5/whatis.html. Date of Access: 15-06-2023.

[19] Hyperledger fabric peers. https://hyperledger-fabric.readthedocs.io/en/latest/ledger/ledger.html#. Date of Access: 15-06-2023.

[20] Hyperledger fabric ordering service. https://hyperledger-fabric.readthedocs.io/en/release-2.2/orderer/ordering_service.html. Date of Access: 15-06-2023.

[21] Hyperledger fabric gateway service. https://hyperledger-fabric.readthedocs.io/en/release-2.5/gateway.html. Date of Access: 15-06-2023.

[22] The go programming language. https://go.dev/. Date of Access: 20-06-2023.

[23] Node.js. https://nodejs.org/en. Date of Access: 20-06-2023.

[24] Java. https://www.java.com/en/. Date of Access: 20-06-2023.

[25] Hyperledger blockchain performance metrics. `https://www.hyperledger.org/learn/publications/blockchain-performance-metrics`. Date of Access: 20-06-2023.