

**Blockchain-based postal system**

Submitted September 2023,  
in partial fulfilment of the conditions for the award of the degree   
**MSc Computer Science**

**Hongmiao Yang  
20471959**

I hereby declare that this dissertation is all my own work, except as indicated in the text

**Supervised by Ying He**

School of Computer Science  
University of Nottingham

Contents

[Abstract 3](#_Toc144892490)

[Acknowledgements 4](#_Toc144892491)

[Chapter 1: Introduction 5](#_Toc144892492)

[1.1 Background 5](#_Toc144892493)

[1.2 Motivation 5](#_Toc144892494)

[1.3 Aims and Objectives 5](#_Toc144892495)

[1.4 Description of the Work 6](#_Toc144892496)

[Chapter 2: Literature Review 6](#_Toc144892497)

[2.1 The Decentralized Nature of Blockchain 6](#_Toc144892498)

[2.2 The Security of Blockchain 7](#_Toc144892499)

[2.3 The Core Component of Blockchain: Consensus Algorithm 7](#_Toc144892500)

[2.4 Applications of Blockchain 8](#_Toc144892501)

[2.5 Summary 8](#_Toc144892502)

[Chapter 3: Methodology 9](#_Toc144892503)

[3.1 System Overview 9](#_Toc144892504)

[3.2 Blockchain Design and Implementation 10](#_Toc144892505)

[3.2.1 Blockchain Structure and Initialization 11](#_Toc144892506)

[3.2.2 Node Registration in the Decentralized System 12](#_Toc144892507)

[3.2.3 Ensuring Blockchain Integrity 12](#_Toc144892508)

[3.2.4 Decentralized Conflict Resolution 13](#_Toc144892509)

[3.2.5 Block and Transaction Management 14](#_Toc144892510)

[3.2.6 Cryptography and Proof-of-Work 15](#_Toc144892511)

[3.3 Blockchain Network 15](#_Toc144892512)

[3.3.1 Transaction Route 16](#_Toc144892513)

[3.3.2 Mining Route 16](#_Toc144892514)

[3.3.3 Retrieving the Full Chain & Node Management 17](#_Toc144892515)

[3.3.4 Ensuring Consensus 17](#_Toc144892516)

[3.4 Postal System 17](#_Toc144892517)

[3.4.1 Method of Data Storage 18](#_Toc144892518)

[3.4.2 Creating a New Transaction 18](#_Toc144892519)

[3.4.3 Mine 19](#_Toc144892520)

[3.4.4 Sending Mail & Receive Mail 19](#_Toc144892521)

[3.4.5 Parsing QR codes 20](#_Toc144892522)

[3.4.6 Register & Login 20](#_Toc144892523)

[3.5 GUI 20](#_Toc144892524)

[Chapter 4: Evaluation 21](#_Toc144892525)

[4.1 Testing Intra-Blockchain Performance 21](#_Toc144892526)

[4.2 Overall System Response Time 22](#_Toc144892527)

[Chapter 5: Conclusion and Reflections 23](#_Toc144892528)

[5.1 Contribution 23](#_Toc144892529)

[5.2 Gap 23](#_Toc144892530)

[5.3 Limitations 23](#_Toc144892531)

[5.4 Future Work 24](#_Toc144892532)

[Reference 24](#_Toc144892533)

# Abstract

This study aims to explore the potential of integrating blockchain technology into postal systems, striving to establish a decentralized and transparent mechanism for mail operations. The research emphasizes the pivotal role of postal systems in the digital age, highlighting the demand for more efficient, transparent, and secure services. We developed a prototype integrating blockchain technology with the basic functionalities of mail sending and receiving. Transactions were stored on the blockchain, ensuring decentralized data, transparency, and immutability. Preliminary tests demonstrated promising capabilities, particularly in terms of data consistency and traceability. However, limitations related to the system's simplified functionalities and data storage methods were identified. These findings provide a foundation for future research, underscoring the need for exploring wider postal business scenarios, more sophisticated consensus algorithms, and synergies with other advanced technologies.

# Acknowledgements

I would like to take this opportunity to express my profound gratitude to those who offered their unwavering support and assistance throughout the course of completing this thesis, as well as to my cherished pets.

First and foremost, I am immensely grateful to my supervisor, Professor Ying. Throughout the course of this research, her invaluable guidance and enduring patience have been instrumental in shaping the direction and content of this work. Next, I would like to thank my parents. Their continual support has been the backbone, allowing me to immerse myself wholeheartedly in my academic pursuits. Lastly, my heartfelt appreciation goes to my two beloved cats, Bao Bao and Xue Er, who, regrettably, departed from this world due to illnesses while I was in my studies. It saddens me deeply that I could not be with them during their final moments. I will forever cherish the ten years they graced my life with their presence.

# Chapter 1: Introduction

## 1.1 Background

In the contemporary digital era, with the explosive growth of data volumes and the widespread adoption of the internet, concerns over data privacy are intensifying. Traditional postal systems, serving as crucial channels for information and logistics exchange, are confronting multifaceted challenges arising from technological advancements, market dynamics, and evolving user demands. Simultaneously, the centralized architectural nature of these systems poses potential security vulnerabilities, rendering them susceptible to attacks. Consequently, determining how to harness emerging technologies to enhance the efficiency, security, and transparency of the postal systems has become an urgent issue. Historically, postal systems predominantly relied on centralized management and operations. In today's digitized context, such centralized modalities are increasingly failing to meet user demands. Specifically, when considering data security, transparency, and efficiency, conventional postal systems are revealing inherent shortcomings (Jaag, 2014). Blockchain technology, with its decentralized, transparent, and immutable attributes, offers a novel solution for the postal system's contemporary challenges.

In fact, as technology advances, it is not only the postal systems that are exploring the potential applications of blockchain technology (Li & Wu, 2022). From supply chain management to financial transactions, and from healthcare to the Internet of Things, blockchain technology is progressively manifesting its unique value and potential. For postal systems, blockchain not only enhances service transparency and security but also facilitates genuine decentralized operations, significantly improving system efficiency and reliability (Eggrickx et al., 2020). Additionally, Winkler (2022) underscores that the modernization and efficiency-driven evolution of postal services are pivotal to the current developmental trajectory of postal systems. Achieving this objective undeniably hinges on technological innovation. In this regard, blockchain technology presents an unparalleled opportunity for postal systems. By integrating blockchain, postal systems can not only offer more secure and transparent services but can also achieve authentic decentralized operations, thus catering to the elevated standards of postal services expected by contemporary society (Dutta et al., 2020).

## 1.2 Motivation

The motivation behind this research stems from the intricate challenges contemporary postal systems face, particularly in the realms of data privacy, centralization tendencies, and transparency (Eggrickx et al., 2020). In today's digital age, with growing public concern for data privacy, ensuring the reliability of data within postal systems has become paramount. Traditional postal systems grapple with issues of security and efficacy, often hindered by centralized organizational structures that negatively impact operational efficiency (Che Hashim, 2018). Such centralization introduces vulnerabilities, making the system more susceptible to disruptions and attacks. Tallyn et al. (2021) harnessed blockchain and smart contracts to not only enhance precision, coordination, and accountability in postal and logistics operations—thereby improving the working experience for delivery personnel—but also to offer a more efficient and secure solution for the system. Nonetheless, challenges persist within the postal sector, notably uncertainties surrounding the security and efficacy of mail (Che Hashim, 2018). Such uncertainties can foster distrust among users, potentially affecting their willingness to engage. Jaag (2014) and Rogowski et al. (2022) posit that the core issues encountered by traditional postal systems are primarily attributed to centralization. As a decentralized technology, blockchain holds promise as a potential solution to these challenges. Thus, delving deeper into how blockchain can be leveraged to refine the postal system, achieving greater decentralization and transparency, has become a pivotal direction in current research.

## 1.3 Aims and Objectives

Consequently, building upon the above-mentioned motivation, this study delineates the following specific research objectives:

* Objective 1: To explore and develop a blockchain-based decentralized postal system. This objective seeks to address the centralization drawbacks of conventional postal systems while proposing solutions to enhance the security and efficiency of mail transmissions.
* Objective 2: To examine the mail transmission mechanisms underpinned by blockchain technology. The focal point of this objective is to ascertain transparent and effective means of storing each mail transaction on the chain.
* Objective 3: To explore and implement storage and retrieval processes for mail QR codes within the blockchain framework, ensuring the efficacy and legibility of QR codes contained in mails.
* Objective 4: To comprehensively evaluate the performance of the system by gauging its response time. By measuring how swiftly the system reacts to user inputs and requests, to ascertain its efficiency and scalability, ensuring that it meets the needs of real-world applications.

## 1.4 Description of the Work

This study delves deep into the convergence of the conventional mail exchange function with the avant-garde blockchain technology. By integrating these, we aspire to introduce a new dimension to the postal system, one that is rooted in the core principles of decentralization. This involves not just the basic exchange of mail but extends to recording intricate transactional information on the blockchain, ensuring a transparent and immutable record of every action, be it mail dispatch or receipt.

Our proposition is not merely an academic exploration; it represents a tangible step forward. We aim to make a seminal contribution to the envisioning and realization of a blockchain-based postal system, setting the stage for a revolutionary shift in how mail systems function. This lays the groundwork for what could eventually transform into an intelligent postal system, augmenting traditional operations with the power of decentralized ledger technology.

What sets our work apart and underscores its novelty is the unique amalgamation of conventional mail operations with a cutting-edge technology framework. While others have primarily explored the theoretical potentials of such a merger, our study focuses on the pragmatic realization and implementation aspects. It highlights the distinctiveness of our approach and its potential to reshape the landscape of postal communications.

In the subsequent sections, we will detail the objective focused on evaluating the system's performance using response time, emphasizing our project's innovative contributions

Chapter 2: Literature Review

## 2.1 The Decentralized Nature of Blockchain

Since the birth of blockchain technology, it has been seen as an important force that could disrupt the entire Internet landscape. In the early days of blockchain, people always confused blockchain with Bitcoin, and research by Tasatanattakool & Techapanupreeda (2018) clarified the differences between blockchain and Bitcoin. Due to the unique characteristics of blockchain, such as security, transparency, decentralization, and immutability, it has given it a wide range of application prospects, covering multiple fields such as finance, healthcare, and supply chain (Li & Wu, 2022). Dinh et al. (2018) provided a comprehensive interpretation and analysis of the blockchain, from four aspects, namely distributed ledger, cryptography, consensus protocol and smart contract, and provided a detailed analysis of the blockchain structure, helping us to understand the underlying technology of blockchain more clearly. Among them, the concept of decentralization has become the core of blockchain's transformative potential. Both Zarrin et al. (2021) and Zheng et al. (2018) talked about the importance of consensus algorithms, analysed their applications and challenges, and considered consensus algorithms to be one of the core components of blockchain technology that can implement the principle of decentralization. In past research, Aini et al. (2020) have explored the feasibility of using blockchain and QR codes to verify academic certificates, making full use of the decentralization and robustness of blockchain networks, and improving the trust and transparency of academic certificates in the education system. In addition, the decentralized nature of blockchain is now its distributed structure. Ruan et al. (2021) conducted twin studies on blockchain and distributed databases and discussed the security of blockchains and the performance of distributed databases. Not only that but Kiayias et al. (2022) also proposed the CSBDC protocol according to the distributed nature of blockchain, which not only circumvents the single point of failure problem but also solves the privacy and regulatory problems of digital currency. In addition, based on the distributed ledger characteristics of blockchain, Graf et al. (2021) propose a design that covers not only blockchain but also non-blockchain distributed ledgers, providing a new idea of data storage.

## 2.2 The Security of Blockchain

With the integration of blockchain technology in different fields, its security is particularly important. Zamani et al. (2020) highlight that while blockchain design itself is secure and has potential, security risks can still arise when interacting with systems and software. Therefore, we cannot assume that the system is secure as long as blockchain technology is used in the system. The study of Qin et al. (2022) also proves this view. Their research shows that there are malicious traders in the blockchain who exploit vulnerabilities in decentralized trading platforms to attack and arbitrage. In addition, Lewis-Pye and Roughgarden (2021) also pointed out the synchronization problems that the longest chain protocol may face by analysing the security of permissionless blockchain protocols. In addition, regarding the security of postal services, Korepanova et al. (2019) have proposed a novel framework for using permissive blockchain technology to prevent stamp fraud. They implemented a blockchain-based system to combat stamp forgery and improve the traceability of stamp circulation. Not only that, in order to solve the significant loss of revenue caused by the illegal copying of stamps to the postal service, they also designed a permissive blockchain solution with timestamps to guarantee the validity of stamps and prevent reuse. And through the certified QR code scanner, stamp transactions are automatically recorded on the blockchain. In addition, Duan et al. (2022) introduce an inspection technique to avoid and detect safety issues. Also, this technique allows for custom safety specifications and can identify safety problems. This reminds us that we need to raise awareness of security vulnerabilities and the means to be strong security measures in blockchain systems.

## 2.3 The Core Component of Blockchain: Consensus Algorithm

In the field of blockchain technology, consensus algorithms play an indispensable role in determining the security, reliability and effectiveness of the entire network. Although the underlying consensus algorithms may be relatively simple by design, they are critical to ensuring the consistency and trustworthiness of transactions in distributed systems. Although the problem of unguaranteed consistency may occur in unreliable networks, Dingyu et al. (2020) propose a consensus mechanism to solve the problem of ensuring consistency and correctness in unreliable network environments. On the other hand, when discussing consensus mechanisms, one must mention Proof-of-Work (PoW). It is a widely used mechanism that guarantees the legitimacy of transactions by solving complex mathematical puzzles, subsequently leading to the creation of new blocks. Capponi et al (2023) verify the expectation that the proof-of-work consensus protocol can support distributed cryptocurrency mining through a game theory model. While the model may lead to centralized mining, greater mining rewards lead to greater decentralization. In addition, Saleh (2020) proposed a proof-of-stake alternative to the proof-of-stake scheme based on the fact that the proof-of-work consumes a lot of energy. Not only that, Schinckus (2021) raised the issue of proof-of-work on energy consumption, showing that Bitcoin has been underestimated in the past in research assessments of its ecological impact. Nevertheless, we continue to use proof-of-work in this study. Despite the energy efficiency advantages of proof-of-stake, challenges persist in its implementation and safety. Between the scope and objectives of the study, we chose to maintain the use of proof-of-work mechanisms to mitigate some potential risks and uncertainties.

## 2.4 Applications of Blockchain

On the other hand, integrating blockchain technology with postal, logistics and supply chain systems has become an important avenue for innovation. Che Hashim (2018) has raised concerns about the security and effectiveness of services such as email due to technical issues in the design of electronic communications. Not only that, both Jaag (2014) and Rogowski et al. (2022) also analysed that the essence of the problems existing in traditional postal services is due to excessive monopoly and excessive centralization. With continuous development and exploration, Wu et al. (2019) addressed the limitations of traditional centralized supply chain management mechanisms, such as lack of transparency, data retrieval delay, Product traceability checks, etc., and strengthened data management. With the popularization and application of blockchain in various industries, Dutta et al. (2020), Eggrickx et al. (2020) and Winkler (2022) successively analysed and discussed the application of blockchain in supply chain and postal service through case analysis and discussion. and logistics industry opportunities and prospects are given a positive attitude. In addition, Dey et al. (2021) proposed a framework for applying blockchain technology to the digitization of food production information, that is, to write food information in the form of two-dimensional codes to make it easy for consumers to obtain and track. On the other hand, Hasan et al. (2022) discussed how to solve the problems of the former centralized supply chain system in protecting product authenticity, and transaction privacy and There are weaknesses in security. Not only that, they directly record product transactions on the blockchain through certified IoT QR code scanners to solve key challenges in the supply chain like disinformation and lack of transparency. It is worth mentioning that Tallyn et al. (2021) improved the user experience of couriers during the delivery stage by improving accuracy, synergy and accountability through blockchain and smart contracts. In order to make up for the lack of trust guarantees in decentralized systems, Chen et al. (2022) proposed real-time arbitration based on smart contracts through blockchain technology to solve such problems, although there are currently users who may not recognize the arbitration results of smart contracts This leads to the need for a second offline arbitration, but this is also the direction that this research will continue to challenge in the future. In addition, Bhatia & Albarrak (2023) found that due to the lack of detection and authentication mechanisms in the grain supply chain process, they use blockchain-based encryption schemes to verify communication tokens, and can quickly detect information through QR codes. This minimizes public health hazards and enhances efficient decision-making for suppliers. Its current disadvantages are that there is a sensor failure that will cause effective information loss, and the blockchain delay will be caused by the consensus algorithm under high throughput. In order to provide enough ideas for these studies, in the postal system, the mail and the Internet of Things can be combined through the QR code, and the QR code can be stored as data in the blockchain to ensure that the data cannot be tampered with.

## 2.5 Summary

In this part, we can see from the discussion of the decentralized characteristics of blockchain, the exploration of its core function consensus mechanism, and the application and case analysis in postal, logistics and supply chains that although the consensus mechanism and security of blockchain technology need to be discussed and in-depth research, it has sufficient potential in the postal industry due to its decentralization and transparency. At present, blockchain technology is mainly used in the supply chain and logistics industry, and there are still not enough applications in the postal system. Therefore, my research is to provide an attempt to integrate and evaluate blockchain technology and the sending and receiving functions of mail in the postal system and QR codes.

# Chapter 3: Methodology

## 3.1 System Overview

For postal systems, the benefits of using blockchain can be realized in transactions between sellers and buyers, where an intermediary role is usually required in traditional centralized supply and logistics systems. In systems such as Amazon, this intermediary mechanism ensures rights and communications between both buyers and sellers. However, the rules and treaties that govern the operation of the system and that buyers and sellers face are defined by the intermediary.  
The system eliminates the intermediary mechanism by means of blockchain node registration and conflict resolution, distributed database storage, and blockchain inspection. Node registration and conflict resolution build a decentralized network and avoid the risk of the blockchain being controlled by a single central node. Distributed databases are used to store and manage mail transmissions and transaction data, which are shared by multiple nodes and confirmed and verified for consistency by consensus algorithms. Blockchain validation can be used to verify the validity of other nodes' chains by reviewing and verifying mail transmissions and transactions through multiple nodes. Each node is empowered to participate in the validation process, thus reducing the reliance on centralized structures.

[Figure 1](#figure1) depicts the overall architecture of the system. At the foundation lies the hardware or physical layer. Above this is the data layer, responsible for data storage. The third tier is the blockchain layer, dedicated to executing blockchain-related functions. The fourth stratum serves as the business layer, implementing the specific functionalities of the postal system. The fifth and final layer is the application layer, visualizing the capabilities provided by the business layer.

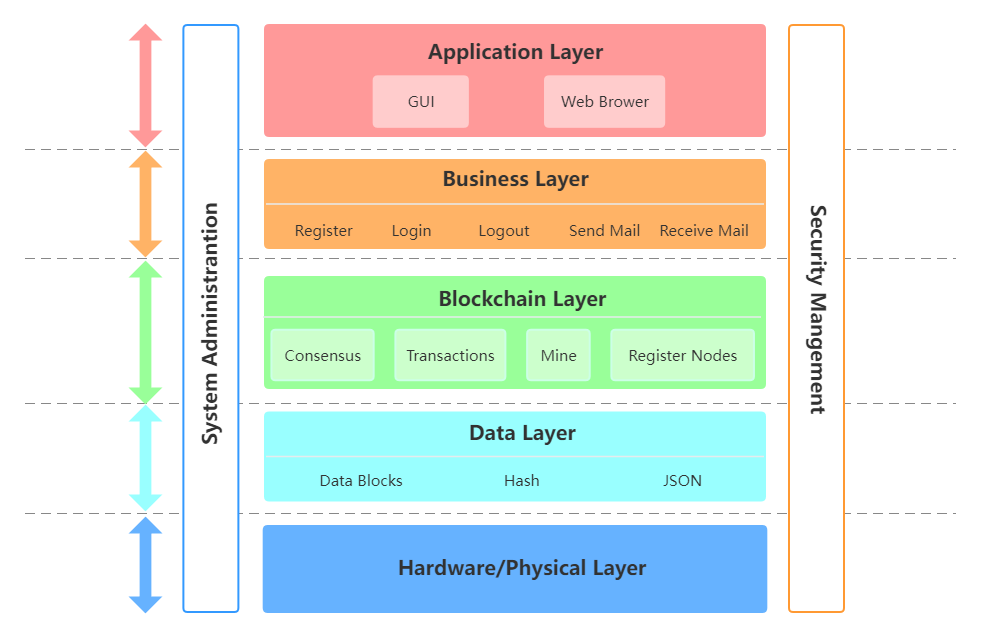


Figure : Postal System Architecture

The process of using the proposed blockchain-based postal system is as follows:

1. User can register a new account by providing a username and password. The system will check whether the username and password are registered or not. A pound of registered users can login using the username and password.
2. User creates mail transactions. Users can create new mail after login. The user can provide the username of the recipient, subject and content of the mail. In addition, the user has the option to generate and upload a QR code and the data will be automatically and securely stored in the blockchain (Hasan, 2022). It will also be sent to the recipient as part of the mail.
3. Mail delivery and mining. After a user creates a mail, the mail is added to the user's blockchain as a transaction. Not only that, but the user can choose to perform a mining operation, and there will be incentives to encourage incoming users to perform this operation. This operation will pack the pending transaction into a new block.
4. Receiving and viewing mails. After logging into the system, the recipient can view the mails in their inbox. The system will extract the mail transaction from the blockchain and display the mail subject and sender information.
5. Mail content and QR code access. Users can select a specific mail and view the integrity of the mail, including the text and the attached QR code. In addition, the user can scan the QR code with a smart device, such as a cell phone, to access the information.

The implementation architecture of the above system is based on Python, HTTP, Flask, Pyzbar, and JSON. each technology stack plays a different role in the overall system architecture and works together to realize the system's features.

* Python is used for back-end development and logic implementation, such as data processing, server creation, implementation of business logic and interaction between the postal system and the blockchain.
* HTTP (Hypertext Transfer Protocol) is used to visualize and transfer data in the decentralized postal system, including data exchange between users and servers, and data synchronization between blockchain nodes.
* Flask is a lightweight Python-based Web application framework. It mentions enough basic tools and features needed to build web applications, such as routing, request handling, template rendering, etc.
* Pyzbar is a Python library for decoding QR codes and barcodes. In the system, it is used to decode user-input QR code information, such as mail content, transaction records, and so on. It can effectively parse and visualize this data.
* JSON is a lightweight data exchange format. Due to its flexibility and ease of use, it is used in this system to store and transfer various information, such as user account information, mail content, transaction records, blockchain data, and so on.

Four key mechanisms in the above approach are described below. These include the mail sending and receiving process based on blockchain technology, decentralized trust mechanism, data encryption security guarantee, and blockchain transaction verification and confirmation.

## 3.2 Blockchain Design and Implementation

This part pertains to the blockchain layer of [Figure 1](#figure1). Within this segment, we established a foundational structure and functionality for the blockchain. To ensure the integrity and immutability of postal data, we record all postal data on a private blockchain. Whenever a new mail or parcel enters the system, a new block is generated which contains all the information of that mail or parcel. At the same time, this newly generated block will be linked to the previous block to ensure the continuity and integrity of the data.

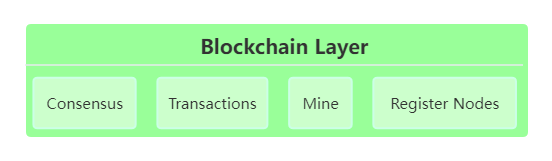


Figure : blockchain layer

[Figure 2](#figure2) is a section of the blockchain layer in [Figure 1](#figure1). It displays the primary functional structures of the blockchain we implemented, such as consensus algorithms, transactions, mining, and node registration.

The blockchain design we adopt follows the traditional data structure and working principle. Each block contains a timestamp, transaction information, proof of work, and the hash value of the previous block. Additionally, in order to maintain integrity and tamper-proofness, each block generates a hash value through its content. In order to ensure the security of the system and to avoid double spending, a proof-of-work mechanism is used. This involves solving a mathematical problem to ensure that the creation of a new block requires some computational work. Our verification condition is to look up a number that, when hashed with the proof of work of the previous block, results in the first four characters of '0000'. In addition, in order to enable multiple nodes to work in a distributed environment and maintain chain integrity, we implemented a node registration function and a consensus algorithm. If a node discovers that another node's chain is longer, it replaces its own chain.

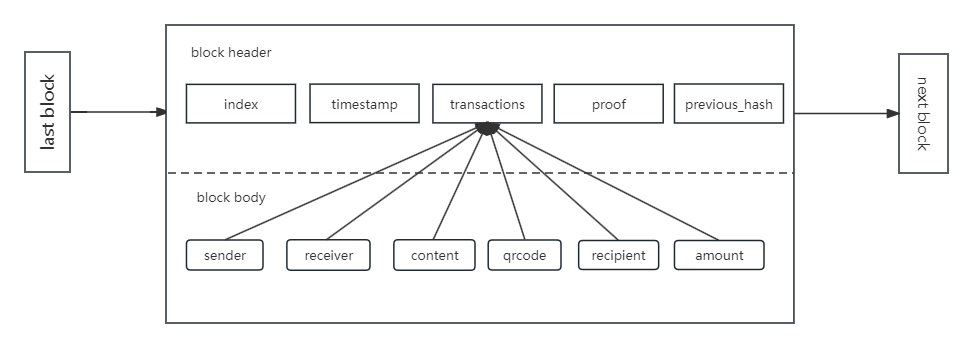
[Figure 3](#figure3) illustrating the internal structure of an individual block within the blockchain. Within the block header, fundamental data such as the index, timestamp, transactions, proof, and the hash value of the preceding block are present. Within the block body, business-related data is stored, encompassing fields like the sender, receiver, content, and QR code.

Figure : Intra-blockchain structure

### 3.2.1 Blockchain Structure and Initialization

The Blockchain framework represents a fundamental structure upon which cryptocurrency systems and other distributed ledger technologies are built. Our design starts with the initialization of the Blockchain class. Upon creation:

* A list, chain, is instantiated. This list functions as the primary ledger, holding all the blocks in the blockchain.
* Another list, current-transaction, is used to hold ongoing or unconfirmed transactions. It is initialized to hold transactions waiting for block placement. It is imperative to separate the ongoing or unconfirmed transactions from those embedded in the blockchain for integrity and accuracy.
* The node-set is used to store nodes in the network. During initialization, it is used to store the addresses of network participants, ensuring decentralization.
* To initiate the blockchain, a genesis block is created using the new\_block method with a default proof of 100 and a previous\_hash of 1.

|  |  |
| --- | --- |
| Algorithm 1 Pseudocode for Blockchain Initialization | |
| 1 | **procedure** INIT() |
| 2 | chain ← [] |
| 3 | currentTransaction ← [] |
| 4 | nodes ← set() |
| 5 | NEW\_BLOCK(proof=100, previous\_hash=1) |

### 3.2.2 Node Registration in the Decentralized System

For the blockchain to be truly decentralized, a mechanism for adding new nodes or participants is necessary. Our system provides the register\_node function, which parses and validates a given node address. The parsed address is then added to the nodes set, ensuring that each node is unique within the network. A given node's address is parsed to validate its structure. This is important for ensuring that only valid URLs are integrated into the network. Depending on the URL structure, either the network location (netloc) or the path is added to the node-set. This flexibility accommodates varying address formats. If neither a valid netloc nor a path is identified, an error is raised, preventing erroneous or malicious URLs.

|  |  |
| --- | --- |
| Algorithm 2 Pseudocode for Blockchain Registration Node | |
| 1 | **procedure** REGISTER\_NODE(address) |
| 2 | parsed\_url ← parse the address |
| 3 | **if** parsed\_url has netloc **then** |
| 4 | add parsed\_url.netloc to nodes |
| 5 | **else if** parsed\_url has path **then** |
| 6 | add parsed\_url.path to nodes |
| 7 | **else** |
| 8 | raise error "Invalid URL" |

### 3.2.3 Ensuring Blockchain Integrity

The validity of the blockchain is and its importance. valid\_chain function is also the basis of the entire blockchain. It internally traverses each block in the chain through a loop. Compare the previous\_hash of the current block with the hash of the previous block to ensure that the link is not broken. The valid\_proof method is also called to validate the proof of work for each block. The loop continues until each block is validated, ultimately confirming the integrity of the entire chain.

The valid\_chain(chain) method checks each block of the blockchain to ensure that it has the correct hash reference to the previous block and that its proof-of-work is correct. The method returns False if any of the blocks do not meet these conditions, and it returns True if all of the blocks meet the conditions. it validates by comparing the current block's previous\_hash to the previous block's hash, and it also validates the proof of work. At the beginning of the program, two variables last\_block and current\_index are initialized to represent the first block in the blockchain and the index value of 1. After that, the while is used to traverse the entire blockchain, and in the loop, the hash of the previous block is checked to see if the hash of the previous block matches with the 'previous\_hash' of the current block and the proof of work is used valid\_proof(last\_block['proof'], block['proof']) to verify that the proof of work for the current block was generated based on the proof of work for the previous block. If correct, iterate to the next block, current\_index+1, and check the next block until the end of the loop.

|  |  |
| --- | --- |
| Algorithm 3 Pseudocode for Blockchain Valid Chain | |
| 1 | **procedure** VALID\_CHAIN(chain) |
| 2 | last\_block ← chain[0] |
| 3 | current\_index ← 1 |
| 4 | **while** current\_index < length of chain **do** |
| 5 | block ← chain[current\_index] |
| 6 | **if** block's 'previous\_hash' ≠ HASH(last\_block) **then** |
| 7 | **return** False |
| 8 | **if** not VALID\_PROOF(last\_block's 'proof', block's 'proof') **then** |
| 9 | **return** False |
| 10 | last\_block ← block |
| 11 | current\_index ← current\_index + 1 |
| 12 | **return** True |

### 3.2.4 Decentralized Conflict Resolution

One of the challenges faced by decentralized systems is data integrity across nodes. resolve\_conflicts function solves this problem as it retrieves the chains of all known network nodes. This is important in decentralized systems this is due to the fact that different nodes may have slightly different chains due to propagation delays or network issues. In addition, it adheres to the consensus principle by evaluating the length (which represents the length of the blockchain) and validity, the system ensures that the longest valid chain in the network is used. If a longer valid chain is found, it replaces the current chain.

The purpose of resolve\_conflicts () is to guarantee that the current node's blockchain is the longest and valid blockchain in the network. This is a strategy for achieving distributed blockchain consensus and is often referred to as the "longest chain principle." It resolves blockchain conflicts between nodes by ensuring that the chain on each node is the longest and most up-to-date. First, the program assigns values to Neighbours all nodes registered on the current blockchain instance and initializes a new\_chain variable to hold valid blockchains that may be found longer than the current node. In addition, the program also obtains the length of the blockchain on the current node and assigns it a value to the max\_length. The program then loops through all neighbouring nodes, compares their chain lengths, and selects the longest valid chain as the new chain. In the loop, an HTTP GET request is made to each neighbour node to get the complete blockchain of that node. If the response code status\_code == 200, the request is successful. The program parses the length of the blockchain and the actual blockchain data from the response. At this point, the program determines whether the blockchain length of this neighbour node is greater than the currently known maximum length and verifies that it is a valid valid\_chain chain. If the above conditions are met, the known maximum length is updated and the entire longer blockchain is assigned to 'new\_chain'. It checks if a new, longer valid blockchain is found, and if so, replaces the current node's blockchain and returns True. Otherwise, return False, indicating that our blockchain has not been replaced.

|  |  |
| --- | --- |
| Algorithm 4 Pseudocode for Blockchain Conflicts Resolution | |
| 1 | **procedure** RESOLVE\_CONFLICTS() |
| 2 | neighbours ← nodes |
| 3 | new\_chain ← None |
| 4 | max\_length ← length of chain |
| 5 | **for** node **in** neighbours **do** |
| 6 | response ← get request to 'http://{node}/chain' |
| 7 | **if** response is successful **then** |
| 8 | length ← response's length |
| 9 | chain ← response's chain |
| 10 | **if** length > max\_length and VALID\_CHAIN(chain) **then** |
| 11 | max\_length ← length |
| 12 | new\_chain ← chain |
| 13 | **if** new\_chain exists **then** |
| 14 | chain ← new\_chain |
| 15 | **return** True |
| 16 | **return** False |

### 3.2.5 Block and Transaction Management

Each blockchain manages blocks and transactions securely and transparently. Where the new\_block method is used to handle block creation. In the design of the blockchain, each block is timestamped, indexed, and contains a series of transactions. In addition, each block has a proof from a proof-of-work algorithm and a hash of the previous block, thus ensuring the continuity and tamper-proofness of the blockchain. Additionally, transactions are represented as a dictionary with details such as sender, receiver, and amount, which is added to the current-transaction list via the new-transaction method.

The main purpose of new\_block(proof, previous\_hash=None) is to create a new block and add it to the blockchain. It does this by creating a new block and adding it to the chain. Input a proof of work and the hash of the previous block (if any) and return the newly created block. The method will have a method block for storing data such as index, timestamp, transactions, proof, previous\_hash, etc. Additionally, it will reset the current transaction list at the end of the method, as all pending transactions have been packed into the new block.

The purpose of the new\_transaction(sender, receiver, content, recipient, amount) method is to add a new transaction to the blockchain. It adds the new transaction to the 'currentTranscation' list, returning the index of the block that will hold the transaction. It has a total of 5 inputs which are sender(sender address), receiver(receiver address), content(message content), recipient(receiver name) and amount(transaction amount). Also, in ‘currentTransaction.append()’, a new transaction is created as a dictionary and added to this list. Each key-value pair of the dictionary corresponds to a parameter. This dictionary contains all the information related to a transaction. The method finally returns the index of the new block that is about to be created.

|  |  |
| --- | --- |
| Algorithm 5 Pseudocode for Blockchain Transaction Management | |
| 1 | **procedure** NEW\_BLOCK(proof, previous\_hash) |
| 2 | block ← { |
| 3 | 'index': length of chain + 1, |
| 4 | 'timestamp': current time, |
| 5 | 'transactions': currentTransaction, |
| 6 | 'proof': proof, |
| 7 | 'previous\_hash': previous\_hash or HASH(last block in chain) |
| 8 | } |
| 9 | currentTransaction ← [] |
| 10 | append block to chain |
| 11 | **return** block |
| 12 | **procedure** NEW\_TRANSACTION(sender, receiver, content, recipient, amount) |
| 13 | transaction ← { |
| 14 | 'sender': sender, |
| 15 | 'receiver': receiver, |
| 16 | 'content': content, |
| 17 | 'recipient': recipient, |
| 18 | 'amount': amount |
| 19 | } |
| 20 | append transaction to currentTransaction |
| 21 | **return** index of last block + 1 |

### 3.2.6 Cryptography and Proof-of-Work

The hash method returns the SHA-256 hash of a block. SHA-256, a cryptographic hashing function, ensures that each block's content is converted to a fixed-size string of bytes, appearing random. Any slight change in input drastically changes the hash, enforcing content integrity.

The proof-of-work mechanism is implemented through the proof\_of\_work and valid\_proof methods, which require work to be computed to create a new block. It blocks spam and unauthorized changes. A proof-of-work method. Simply put, it finds a number 'proof' such that the first four characters of the hash value after connecting it to the previous proof of work are '0000'. This ensures that it takes some work to create the block, thus making it more difficult to maliciously modify the chain. Proof-of-work is a consensus algorithm used in blockchain that requires participants to perform computationally intensive tasks to reach a consensus, thus ensuring the security of the blockchain. In the proof\_of\_work(self, lastProof) method, it accepts a parameter lastProof representing the proof-of-work value of the previous block. Inside the method, proof = 0 indicates that a variable named proof is initialized and assigned a value of 0. This variable will be used to find a valid new proof-of-work value. Finding a valid new proof of work is achieved by a while loop which will keep executing until a valid proof of work value is found.

|  |  |
| --- | --- |
| Algorithm 6 Pseudocode for Blockchain Cryptography and Proof-of-Work | |
| 1 | **procedure** HASH(block) |
| 2 | block\_string ← convert block to json string, sorted by keys |
| 3 | **return** sha256 hash of block\_string |
| 4 | procedure PROOF\_OF\_WORK(lastProof) |
| 5 | proof ← 0 |
| 6 | **while** not VALID\_PROOF(lastProof, proof) **do** |
| 7 | proof ← proof + 1 |
| 8 | **return** proof |
| 9 | **procedure** VALID\_PROOF(lastProof, proof) |
| 10 | guess ← concatenate lastProof and proof |
| 11 | guessHash ← sha256 hash of guess |
| 12 | **return** first 4 characters of guessHash == '0000' |

## 3.3 Blockchain Network

This part, like Part 3.2, pertains to the blockchain layer within [Figure 1](#figure1). The construction of the blockchain network is based on the Flask framework in Python, a micro web framework, serving the purpose of testing whether the foundational blockchain functionalities operate properly. Within the network, it is essential to identify each node with a unique identifier. Hence, we opted to utilize the uuid4() function to generate a distinct node identifier, ensuring the identifier is unique and random in the form of a string. This ensures that nodes within the network are also unique and randomized. Subsequently, we initialized an instance of the blockchain class previously described.

### 3.3.1 Transaction Route

The routing and processing function new\_transaction() allows the client to add a new transaction to the blockchain via a POST request and tells the client which block the transaction will be added to. Its purpose is to add a new transaction to the blockchain. It takes three parameters: sender, recipient, and amount. If the transaction is successful, a message is returned telling the client to which block the transaction will be added.

In this case, @app.route() is a Flask decorator that defines a new route. The URL of the route defined here is /transactions/new and only the POST method is allowed. And values = request.get\_json() uses Flask's request object to get the JSON data from the POST request and saves it to the values variable.

In addition, we define a list of required fields, which are fields that must be included in the transaction data. If there are missing fields, we return an error message 'Missing values' and a status code 400 indicating that the request was formatted incorrectly. If the fields are correct, we extract the mandatory fields and pass them to the blockchain.new\_transaction method. This method adds a new transaction to the blockchain and returns the index of the block this transaction will be added to. This index value is stored in the index variable. Then, we create a response message that contains a message field that tells the client which block the transaction will be added to. The format of the response message is converted to JSON.

### 3.3.2 Mining Route

The ‘/mine’ route is dedicated to mining new blocks on the blockchain. The process entails: Retrieving the last block and its proof. Computing a new proof-of-work. Creating a new transaction that awards a miner a reward. Adding the new block to the blockchain. Returning details of the newly mined block as a response.

The logic implemented by mint() is to find a new proof of work, add a new transaction for the miner (i.e., the node that performs the mining), and then create a new block and return information about the new block. The code builds the blockchain's mining process and provides a way to start the process through Flask routing. Mining is a computationally intensive process that involves finding a number (proof of work) that means the right answer to a given question. We have used Flask's decorator to define a new route with the URL /mine and accessed via the GET method. After that we fetch the last block on the blockchain and store it in last\_block. Extract the proof of work from the last block and store it in last\_proof. Also, the program will be called the blockchain. And proof\_of\_work method to calculate the new proof of work. It is a computationally intensive process as a specific number (new proof of work) needs to be found. After that, the program adds a new transaction to the new block representing the miner's reward. The sender of this transaction is "0" (indicating that it is a newly mined coin) and the receiver is the identifier of the current node with an amount of 1. In addition, the program stores the hash value of the last block calculated in previous\_hash and creates a new block, which includes the new proof of work and previous\_hash. After that the program responds to the client with a creation of a response dictionary that includes the message, index, transaction list, proof of work, and the hash of the previous block by converting it to JSON. At the same time, the HTTP status code 200 is returned, indicating that the request has been successfully processed.

### 3.3.3 Retrieving the Full Chain & Node Management

To facilitate transparency and verification, the /chain endpoint is available. It retrieves the entire blockchain and returns both the chain and its length, offering a quick overview of the blockchain's current state.

Blockchain decentralization is achieved through nodes. To accomplish this, we route /nodes/register to allow adding new nodes to the network. A POST request containing a list of nodes is processed so that each node is registered with our blockchain instance. The program returns feedback about the successful addition of a new node and an overview of all nodes.

Registers new nodes for the blockchain. The input is a list of nodes, each of which is added to the nodes collection of the Blockchain. We define a new route in the program for registering new nodes in the blockchain network. Nodes can be considered as participants or servers in the blockchain network. The main function of this route is to allow new participants to inform the network about themselves and join the network. The program defines a new route through a Flask‘s decorator @app.route(). It has a URL path of /nodes/register and accepts requests from the POST method. The program needs to get the JSON data from the HTTP request and store it in the values variable. In values, the program will try to get information about the field named 'nodes' which should contain a list of nodes. These nodes will be added to the blockchain network. If the nodes field is not provided in the request, or if it has the value None, an error message is returned along with the HTTP status code 400 (Bad Request). If the 'nodes' field is not empty, the program traverses the list of nodes. In the loop, for each node in the list, the blockchain.register\_node() method needs to be called to register it to the blockchain network. After the traversal is complete, the program returns a message to the client that the resource has been created successfully and returns the HTTP status code 201.

### 3.3.4 Ensuring Consensus

Given the decentralized nature of our system, it is critical to ensure data consistency across nodes. The /nodes/resolve route is therefore designed for this purpose. It compares our chain with the chains of other nodes in the network and attempts to resolve potential conflicts. If our chain is replaced because a longer valid chain was found, the response is displayed. Conversely, if our chain is considered authoritative (longest and valid), it will also be broadcast to let us know.

We do this by ensuring that all nodes have the same, longest chain. If the current node's chain is replaced, return the new chain, otherwise return the current chain. This logic implements the consensus algorithm. Therefore, the main function of the /nodes/resolve route in the program is also to perform the consensus algorithm to ensure that the current node's version of the blockchain is up-to-date and consistent with the other nodes in the network. If another node's chain is found to be longer and valid, the current node's chain is replaced. This is one way to achieve consensus in a distributed blockchain system. The program checks the other nodes in the network by calling the blockchain.resolve\_conflicts() method to confirm their chain versions, if there is a longer and valid chain, the current node's chain will be replaced, and the method returns True; otherwise, it returns False. if the resolve\_conflicts() method returns If resolve\_conflicts() returns True, the chain of the current node has been replaced, then the program needs to give a response to the client, which needs to inform the news that the chain of the current node has been replaced. If the return is False, it means that the chain of the current node is authoritative and has not been replaced. At this point, the client needs to be sent a message that the current node's chain is authoritative. Thus, the main purpose of this method is to ensure that the current node's version of the blockchain is consistent with the other nodes in the network. If a longer and valid chain exists, the current node's chain will be replaced, otherwise, it will remain unchanged. The message in the response tells the client the state of the chain and the current chain data.

## 3.4 Postal System

In this part, we focus on the implementation of the business layer. [Figure 4](#figure4) represents the business layer segment from [Figure 1](#figure1), which is the method to be implemented in this section. The approach not only encompasses the functionalities of sending and receiving mails but also integrates the mail content and associated QR codes into the blockchain. Additionally, foundational user functions, such as registration, logging in, and logging out, have been established in the program. In order to build a postal system, we chose to use a web application built using the Flask framework. The main function is similar to mail communication, but unlike traditional mail, here the mail is stored and exchanged through a blockchain implementation. It is worth mentioning that for the logistics information, the goods will be delivered by QR code and mail. The recipient can get the latest messages by scanning the QR code or getting mail on the page. In addition, there are registration and login functions in the system.



Figure : business layer

### 3.4.1 Method of Data Storage

Before implementing the functionalities of the business layer, we have to address the data layer. As illustrated in [Figure 1](#figure1), the data layer encompasses various components, prominently blockchain storage and user data storage. For the sake of development efficiency, we eschewed the use of conventional relational databases such as MySQL, opting instead for JSON-based user data storage. It is imperative to clarify that this approach does not entail the erasure of existing data with each termination and subsequent restart of the project. While the JSON storage methodology might not be universally applicable, it is particularly suited for our application. This preference is attributed to JSON's schema-free design, lightweight nature, and seamless integrative capabilities. The absence of a fixed schema obviates the need for tasks related to table instantiation and complex join operations. Moreover, being a standard data interchange format in web development, JSON ensures facile integration with APIs and server-side components. Its inherent lightweight characteristic further eliminates the complexities often encountered with the setup and troubleshooting of conventional relational databases in our system (Bourhis et al. 2020).

### 3.4.2 Creating a New Transaction

The new\_transaction() method allows the client to add a new transaction to the blockchain via a POST request. The program will notify the client which block the transaction will be added to. The purpose of this method is to add a new transaction to the blockchain, and to accomplish this three parameters are required, sender, receiver, and amount. If the transaction is successful, the method returns a message telling the client to which block the transaction will be successfully added. In addition, to ensure the integrity of the message, we define a list of required fields, and before creating a new transaction, the program needs to check the integrity of the fields. A new transaction can only be successfully created if none of the three fields are empty. To create a new transaction, the new\_transaction() method of the blockchain class is called, and when it is created successfully, it returns the index value of the block to which the transaction was added, which is assigned to the index variable and returned to the client as a response message from the client. Considering that the message needs to be serialized, we use JSON as the format of the response.

### 3.4.3 Mine

The mining feature is designed to incentivize miners to actively participate while maintaining the proper functioning and ensuring the security of the entire blockchain ecosystem. As the blockchain is composed of a series of blocks, transaction records are stored in each block. When a new transaction record appears it is necessary for a miner to go out and validate the pending transaction record while packing it into a new block. On the other hand, the security and decentralized nature of blockchain relies on the consensus mechanism such as proof of work. In proof of workload, the miners need to prove that they have done some work for the blockchain network and hence they have to solve a complex mathematical puzzle. Since too complex a mathematical puzzle increases the computational cost, in order to ensure the integrity and efficiency of the feature, here we reduce the difficulty of the proof of workload. That is, we set one of the verification conditions as the first four characters of the hash value must be 0000, and the algorithm will try to satisfy the value of the verification condition by continuously increasing the size of the PROOF value until it finds the answer that finally satisfies the condition. It is through this large amount of computing power, makes it more difficult for malicious attackers to control the network, making the network more secure. In addition, the mining function will also realize the decentralization idea, it is through the decentralized nodes to participate in mining, that the blockchain network can avoid centralized control, increasing the resistance to single points of failure and potential attacks.

In the method implementation, the incoming parameter includes 4 parameters, namely sender address, receiver address, memory and QR code image path. The program will first get the most recent block which contains information such as its workload proof. The value of this workload proof is then assigned to the variable last\_proof, which will be used as the basis for the calculation of the next block. Immediately after that, the program passes it into the blockchain's proof-of-work method and uses that method to get the proof-of-work value for the new block, saving it with the proof variable. This process involves mining to find a value that satisfies a specific condition, which is used to guarantee the legitimacy of the new block and the security of the network. If the proof-of-work value for the new block is successfully obtained, the block.new\_transaction() method is called to create a new transaction with the sender, receiver, content, payee address, amount, and possibly the path to the QR code image. This transaction will be included in the new block. In addition, before creating the new block, the hash value of the previous block is also calculated as the previous hash of the new block. after ensuring that all the above data is obtained, the new block is created.

### 3.4.4 Sending Mail & Receive Mail

This function realizes the main functions in the postal system online, including getting mail, sending mail and viewing blockchain. Users can get the content of received mail by clicking the corresponding button, thus aspect to view the latest content of the mail. Meanwhile, users can send mail by filling in the recipient, mail content and uploading the QR code image. The system converts the QR code image into a stream of bytes and embeds it in the blockchain to ensure that the content of the mail is not tampered with. On the other hand, the user is also equipped with the ability to view the blockchain data, which allows the user to view the entire block of messages, including the length of the blockchain and the content of each block by clicking the corresponding button. This provides users with transparency and traceability of the heap system data and also strengthens their trust in the system operation.

In the get mail function, the user first needs to click on the 'mail-get' button to get the input mail content 'mail-get-textarea' from the form. Then the program will call the 'get\_content(username)' function to get the content, which internally traverses each block in the blockchain to find the corresponding user's mail. If it finds a corresponding message, it returns a list of the contents of the message and information about the sender. In addition, the system will try to get a QR code image. If the byte data of the QR code image can be successfully parsed, it will be converted into mail content. Finally, depending on whether the QR code image is successfully fetched or not, the system will use the 'render\_template' function to render the mail page and pass the relevant data, such as the content of the mail, the username, the sender's information and whether the QR code image is available or not. In this case, 'render\_template' is a function in the Flask framework that renders the HTML template and generates the final HTML page. In Flask, the template engine can be used to create dynamic web pages, embedding data and logic into the HTML page for more flexible page generation and rendering.

In the Send Mail function, the user sends mail by clicking on the 'mail-post' button. The program will get the content of the mail from 'mail\_containt\_post'. At the same time, the program reads the data from the uploaded QR code image file, saves it in the specified path, and converts the image to a string for storage in the blockchain. Usually at the initial stage, the system will first get the username of the mail recipient and check whether the mail recipient exists by traversing the list of users. If it exists, it indicates that the mail can be sent. After the above is done, the system will execute the mining function to create a new block and store the mail content, recipient, sender, QR code and other information in it. The 'dump\_bu(username)' function is called after successful creation to save the updated blockchain data to a file.

In the view blockchain function, the user checks if a de-blockchain instance already exists by using 'new\_chain'. If not, a new blockchain object will be created and the previous blockchain data (if any) will be loaded from the local file. Alternatively, the 'get-chain' button will similarly check if a de-blockchain instance already exists. Unlike 'new\_chain', if there is no blockchain instance, it will redirect to the mail page. If a blockchain instance exists, it displays the blockchain information and length in JSON format on the page, which is displayed through the rendering of 'render\_template'. This implementation provides enough transparency and traceability to the user.

### 3.4.5 Parsing QR codes

We have implemented parsing and recognizing QR codes through Python's Pyzbar library. When a sender sends a message to a receiver, it needs to upload a QR code, which stores information related to the goods. This data is stored in the blockchain. On the receiver's page, not only the sender's message is displayed but also the information stored in the attached QR code is read out and displayed in the interface.

### 3.4.6 Register & Login

On the other hand, we have implemented the account function for the purpose of distinguishing the sender from the receiver. Therefore, we provide functions for registering an account, logging in, and logging out. Normally, we should implement the data persistence functionality only through a database. However, since the account function is not the main purpose of the study, in order to save unnecessary development time, JSON is chosen as the data storage method.

## 3.5 GUI

In this part, we address the application layer as depicted in [Figure 1](#figure1), primarily designed to realize the graphical user interface of the entire program. The entire graphical interface of the program is constructed using HTML. The advantages of HTML are its universality and versatility. It complies with web standards and is supported by almost all devices and browsers, signifying that the user interface can function across diverse device platforms. Furthermore, its versatility accommodates the Python-based framework, Flask. Embedding Python code within HTML becomes feasible, facilitating the creation of dynamic web pages with ease. Additionally, Flask streamlines the process of handling server-side requests and dispatching data to the HTML interface.

There are four interfaces in total: the homepage, login page, registration page, and the mail page. The primary function of the homepage is to load static resources. The registration page presents a form requiring username and password details, and users can submit the form using the 'sign up' button. The login page, in a similar fashion, provides a form for user account login, with submissions being made through the 'sign in' button. There is also a link for account creation, redirecting users to the registration page. Lastly, the mail page encompasses three segments: the mail sending section, mail receiving section, and the blockchain functionality section.

In the mail sending section, users input recipient details, the content of the mail, and have the option to upload a QR code image - all of which are mandatory fields. Mail submission is achieved by clicking the 'mail-post' button. In the mail receiving segment, users can retrieve mail by clicking the 'mail-get' button. The program displays the sender's details and allows users to view the content of two received mails in two separate read-only text areas. Moreover, if present, the QR code image is displayed. Additionally, in the blockchain functionality segment, users have the following options:

* Initiate a new chain via the 'new-chain' button, which is imperative during the first mail transmission.
* View pertinent blockchain information through the 'get-chain' button.

This layered structure ensures that the user interface remains intuitive while offering functionalities related to blockchain-based postal services.

# Chapter 4: Evaluation

We conducted preliminary testing on the entire program to assess the functionality and performance of the blockchain-based postal system. Our evaluation methodology drew inspiration from the study by Vazquez & Landa-Silva (2021), which evaluated blockchain-based ride-sharing systems. In their research, they specifically measured Computational Effort and response time. For our context, response time emerged as the more relevant metric. Hence, our assessment focused on two primary aspects:

1. Executing tests specifically aimed at gauging the response time of the blockchain component.
2. Conducting a comprehensive response time test for the entire system.

The performance of our system was subsequently evaluated based on the response times obtained from these tests.

## 4.1 Testing Intra-Blockchain Performance

We established a specialized network for assessing the functionalities of the blockchain, built upon the Flask framework. Various blockchain features such as new transaction creation, node registration, mining, obtaining the full chain, and conflict resolution were designed as routes. In this study, Postman was employed as a pivotal tool to test these routes. Postman, widely used as an API testing and development platform, offers a range of capabilities that allow developers to seamlessly construct, test, and debug APIs. Postman was utilized to send diverse HTTP requests such as GET, POST, and PUT, effectively simulating interactions between users and the system. Through Postman, API response data, status codes, and header information were visually accessible, facilitating the validation of system functionality and performance.

Firstly, we tested the blockchain's new transaction creation functionality. Using Postman, we sent a POST request based on JSON data to the '/transactions/new' test route, aiming to validate the feasibility of creating new blockchain transactions. This route would return a successful request along with a message indicating the impending creation of a new transaction.

Subsequently, the testing of blockchain mining functionality was imperative. This process aimed to validate the consensus mechanism, security, and successful creation of new blocks within the network. By sending a GET request to the '/mine' route without parameters, we effectively tested the mining process. The response successfully returned complete block data.

Furthermore, the functionality to view the entire blockchain was also examined. By utilizing a GET request to the '/chain' route, we accessed the complete blockchain. The response accurately displayed the entire blockchain and its corresponding length.

Node registration functionality was equally indispensable. Through a POST request with JSON-formatted parameters sent to the '/nodes/register' route, we tested the ability to add a new node to the network's node list. The node address typically adheres to network addresses, such as "[http://127.0.0.1:5000](http://127.0.0.1:5000/)". The response from the route indicated a successful request.

To assess the program's ability to resolve blockchain conflicts, we employed a GET request to the '/nodes/resolve' route. When no conflicts exist within the blockchain, the response would return a chain structure data and a message stating "Our chain is authoritative". Conversely, in the presence of blockchain conflicts, the response would display a new chain structure data along with the message "Our chain was replaced".

[Figure 5](#figure5) visually depicts the average response times of these five fundamental blockchain functionalities across 10,000 requests. The average response time for creating transactions is approximately 4ms, while the average response time is 76ms. Additionally, the average response time for node registration and conflict resolution routes is around 5ms, and the least demanding function is obtaining the full chain, requiring an average response time of 4ms.

Figure : Average Response Time for Basic Blockchain Functions

## 4.2 Overall System Response Time

We conducted a performance evaluation of the entire system based on response times. For each functionality within the system, we conducted tests ranging from 10 to 10,000 requests.

On the one hand, we tested the non-blockchain functionalities, which encompass registration, login, and logout operations. Evidently, regardless of the level of requests, the response times for these operations remained within acceptable ranges. Figure 2 illustrates that the response time for registration operations consistently ranged from 3 to 6ms across 10 to 10,000 requests, displaying a subtle linear growth trend. Furthermore, the response times for login and logout operations exhibited relative stability, averaging at 4ms and 2ms for 10 requests, respectively, and reaching up to 6ms and 5ms for 10,000 requests.

On the other hand, we integrated blockchain functionalities with the postal system's send-and-receive mail features. Consequently, sending and receiving mail involved core blockchain functionalities such as creating new blocks, resolving chain conflicts, and proof of work. [Figure 6](#figure6) notably illustrates that as the number of requests increased, the response times for sending and receiving mail gradually rose. Specifically, the response time for sending mail increased from 6ms for the initial 10 requests to 12ms for 10,000 requests. Similarly, the response time for receiving mails increased from the initial 16ms to 52ms for the same request range.

Figure :Response Time Performance of a Blockchain-based Postal System

# Chapter 5: Conclusion and Reflections

## 5.1 Contribution

This study pioneers the exploration of applying blockchain technology to the postal system. Against this backdrop, we successfully designed and implemented a prototype of the postal system based on blockchain. This not only facilitates the foundational mail functions, such as sending and receiving but also capitalizes on the decentralized characteristics of blockchain, ensuring data integrity, transparency, and security. Additionally, we incorporated an innovative node registration simulation, allowing for the representation of various network participants and ensuring data consistency in a distributed environment. To evaluate the actual performance of the prototype, we conducted a series of experiments, simulating real-user interactions with the system and assessing its response time and overall feasibility.

## 5.2 Gap

Postal systems, ubiquitously used worldwide, serve as core conduits for information and logistics. Their efficiency and transparency in the digital age are paramount. However, traditional postal systems often grapple with inefficiencies and opacity of data. While blockchain technology is widely acknowledged for its potential to revolutionize traditional industries, its specific application within the postal system remains an under-researched territory.

## 5.3 Limitations

Despite the preliminary successes of our research, there are evident limitations. Firstly, the current prototype's functionalities remain relatively simplistic, majorly concentrating on basic mail operations. More intricate postal business scenarios, such as logistics tracking and identity verification, have not been embraced yet. Moreover, while we have integrated foundational blockchain functionalities, there is a myriad of advanced consensus algorithms and techniques awaiting exploration. In terms of data storage, our reliance on JSON might inhibit efficiency in handling and querying large-scale data. The absence of smart contracts also curtails the automation potential of our system.

## 5.4 Future Work

Moving forward, it is evident that our prototype, while foundational, has vast potential for enhancement. Firstly, there is a pressing need to expand the prototype's functionalities. This would involve encompassing more intricate postal business scenarios, such as logistics tracking and identity verification. Building on this, a deeper dive into the world of consensus algorithms is imperative. Our current model, while functional, could greatly benefit from the implementation of advanced consensus algorithms, which would undoubtedly enhance the system's overall efficiency and security.

Equally important is the matter of data structure. Our existing reliance on JSON has served its purpose, but as the scope of our project broadens, researching more efficient data storage and querying methods becomes essential. This would not only streamline the data handling process but also pave the way for a more responsive system.

Moreover, the incorporation of smart contract capabilities is a path we are keen to explore. Smart contracts offer a level of automation that our current model lacks. By harnessing their potential, we can further optimize our system, making transactions and operations seamless and more efficient.

Lastly, as technology continues to evolve, it is vital that our system remains adaptable. To this end, there is a compelling case for exploring the fusion of blockchain technology with other emerging domains, like artificial intelligence and big data. Such integration could usher in innovative solutions for postal operations, redefining the landscape of the industry.

# Reference

Aini, Q., Rahardja, U., Tangkaw, M. R., Santoso, N. P. L., & Khoirunisa, A. (2020). Embedding a Blockchain Technology Pattern Into the QR Code for an Authentication Certificate. *Jurnal Online Informatika*, *5*(2), 239. <https://doi.org/10.15575/join.v5i2.583>

Bhatia, S., & Albarrak, A. S. (2023). A Blockchain-Driven Food Supply Chain Management Using QR Code and XAI-Faster RCNN Architecture. *Sustainability (Switzerland)*, *15*(3). <https://doi.org/10.3390/su15032579>

Bourhis, P., Reutter, J. L., & Vrgoč, D. (2020). JSON: Data model and query languages. *Information Systems*, *89*, 101478. <https://doi.org/https://doi.org/10.1016/j.is.2019.101478>

Capponi, A., Ólafsson, S., & Alsabah, H. (2023). Proof-of-Work Cryptocurrencies: Does Mining Technology Undermine Decentralization? *Management Science*. <https://doi.org/10.1287/mnsc.2023.4840>

Che Hashim, R. (2018). Postal rule in acceptance via email. *Commonwealth Law Bulletin*, *44*(1), 111–127. <https://doi.org/10.1080/03050718.2018.1552523>

Chen, F., Wang, J., Jiang, C., Xiang, T., & Yang, Y. (2022). Blockchain Based Non-repudiable IoT Data Trading: Simpler, Faster, and Cheaper. *Proceedings - IEEE INFOCOM*, *2022-May*. <https://doi.org/10.1109/INFOCOM48880.2022.9796857>

Dey, S., Saha, S., Singh, A. K., & McDonald-Maier, K. (2021). FoodSQRBlock: Digitizing food production and the supply chain with blockchain and QR code in the cloud. *Sustainability (Switzerland)*, *13*(6). <https://doi.org/10.3390/su13063486>

Dingyu, W., Dingmin, W., Lupin, Q., & MinLong, X. (2020). A Byzantine consensus based on proof-of-work of nodes’ behaviors. *Journal of Physics: Conference Series*, *1684*(1), 12049. <https://doi.org/10.1088/1742-6596/1684/1/012049>

Dinh, T. T. A., Liu, R., Zhang, M., Chen, G., Ooi, B. C., & Wang, J. (2018). Untangling Blockchain: A Data Processing View of Blockchain Systems. *IEEE Transactions on Knowledge and Data Engineering*, *30*(7), 1366–1385. <https://doi.org/10.1109/TKDE.2017.2781227>

Duan, Y., Zhao, X., Pan, Y., Li, S., Li, M., Xu, F., & Zhang, M. (2022). Towards Automated Safety Vetting of Smart Contracts in Decentralized Applications. *Proceedings of the ACM Conference on Computer and Communications Security*. <https://doi.org/10.1145/3548606.3559384>

Dutta, P., Choi, T.-M., Somani, S., & Butala, R. (2020). Blockchain technology in supply chain operations: Applications, challenges and research opportunities. *Transportation Research Part E: Logistics and Transportation Review*, *142*, 102067. <https://doi.org/https://doi.org/10.1016/j.tre.2020.102067>

Eggrickx, B., Lefort, M., & Roset, A. (2020). Blockchain and Postal Digital Services: Opportunities and Drawbacks. In P. L. Parcu, T. J. Brennan, & V. Glass (Eds.), *The Changing Postal Environment: Market and Policy Innovation* (pp. 151–165). Springer International Publishing. <https://doi.org/10.1007/978-3-030-34532-7_12>

Graf, M., Rausch, D., Ronge, V., Egger, C., Küsters, R., & Schröder, D. (2021). A Security Framework for Distributed Ledgers. *Proceedings of the ACM Conference on Computer and Communications Security*. <https://doi.org/10.1145/3460120.3485362>

Hasan, A. S. M. T., Sabah, S., Haque, R. U., Daria, A., Rasool, A., & Jiang, Q. (2022). Towards Convergence of IoT and Blockchain for Secure Supply Chain Transaction. *Symmetry*, *14*(1). <https://doi.org/10.3390/sym14010064>

Jaag, C. (2014). Postal-sector policy: From monopoly to regulated competition and beyond. *Utilities Policy*, *31*, 266–277. <https://doi.org/10.1016/j.jup.2014.03.002>

Kiayias, A., Kohlweiss, M., & Sarencheh, A. (2022). PEReDi: Privacy-Enhanced, Regulated and Distributed Central Bank Digital Currencies. *Proceedings of the ACM Conference on Computer and Communications Security*. <https://doi.org/10.1145/3548606.3560707>

Korepanova, D., Kruglik, S., Madhwal, Y., Myaldzin, T., Prokhorov, I., Shiyanov, I., Vorobyov, S., & Yanovich, Y. (2019). Blockchain-Based Solution to Prevent Postage Stamps Fraud. *2019 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, 171–175. <https://doi.org/10.1109/BLOC.2019.8751495>

Lewis-Pye, A., & Roughgarden, T. (2021). How Does Blockchain Security Dictate Blockchain Implementation? *Proceedings of the ACM Conference on Computer and Communications Security*. <https://doi.org/10.1145/3460120.3484752>

Li, X., & Wu, W. (2022). Recent Advances of Blockchain and Its Applications. *Journal of Social Computing*, *3*(4), 363–394. <https://doi.org/10.23919/JSC.2022.0016>

Qin, K., Zhou, L., & Gervais, A. (2022). Quantifying Blockchain Extractable Value: How dark is the forest? *Proceedings - IEEE Symposium on Security and Privacy*, *2022-May*. <https://doi.org/10.1109/SP46214.2022.9833734>

Rogowski, J. C., Gerring, J., Maguire, M., & Cojocaru, L. (2022). Public Infrastructure and Economic Development: Evidence from Postal Systems. *American Journal of Political Science*, *66*(4), 885–901. <https://doi.org/10.1111/ajps.12594>

Ruan, P., Dinh, T. T. A., Loghin, D., Zhang, M., Chen, G., Lin, Q., & Ooi, B. C. (2021). Blockchains vs. Distributed Databases: Dichotomy and Fusion. *Proceedings of the ACM SIGMOD International Conference on Management of Data*. <https://doi.org/10.1145/3448016.3452789>

Saleh, F. (2021). Blockchain without Waste: Proof-of-Stake. *The Review of Financial Studies*, *34*(3), 1156–1190. <https://doi.org/10.1093/rfs/hhaa075>

Schinckus, C. (2021). Proof-of-work based blockchain technology and Anthropocene: An undermined situation? *Renewable and Sustainable Energy Reviews*, *152*, 111682. <https://doi.org/https://doi.org/10.1016/j.rser.2021.111682>

Tallyn, E., Revans, J., & Morgan, E. (2021). Enacting the last mile: Experiences of smart contracts in courier deliveries. *Conference on Human Factors in Computing Systems - Proceedings*. <https://doi.org/10.1145/3411764.3445525>

Tasatanattakool, P., & Techapanupreeda, C. (2018). Blockchain: Challenges and applications. *2018 International Conference on Information Networking (ICOIN)*, 473–475. <https://doi.org/10.1109/ICOIN.2018.8343163>

Vazquez, E., & Landa-Silva, D. (2021). Towards blockchain-based ride-sharing systems. *ICORES 2021 - Proceedings of the 10th International Conference on Operations Research and Enterprise Systems*, 446–452. <https://doi.org/10.5220/0010323204460452>

Winkler, H. (2022). *Blockchain-Based Digitalization of Logistics Processes—Innovation, Applications, Best Practices* (H. Winkler, Ed.). Basel MDPI Books.

Wu, H., Cao, J., Yang, Y., Tung, C. L., Jiang, S., Tang, B., Liu, Y., Wang, X., & Deng, Y. (2019). Data Management in Supply Chain Using Blockchain: Challenges and a Case Study. *2019 28th International Conference on Computer Communication and Networks (ICCCN)*, 1–8. <https://doi.org/10.1109/ICCCN.2019.8846964>

Zamani, E., He, Y., & Phillips, M. (2020). On the Security Risks of the Blockchain. *The Journal of Computer Information Systems*, *60*(6), 495–506. <https://doi.org/10.1080/08874417.2018.1538709>

Zhang, C., Xu, C., Wang, H., Xu, J., & Choi, B. (2021). Authenticated keyword search in scalable hybrid-storage blockchains. *Proceedings - International Conference on Data Engineering*, *2021-April*. <https://doi.org/10.1109/ICDE51399.2021.00091>

Zheng, Z., Xie, S., Dai, H.-N., Chen, X., & Wang, H. (2018). Blockchain challenges and opportunities: a survey. *International Journal of Web and Grid Services*, *14*(4), 352–375. <https://doi.org/10.1504/IJWGS.2018.095647>