IMPLEMENTING BLOCKCHAIN BASED INFORMATION FLOW
A CASE STUDY OF KOM CONSULT

KEVIN MUTUGI KITHINJI

SCT221-0415/2020

A proposal submitted to the Department of Information Technology in partial fulfillment of the requirement for the award of the Degree of Bachelor of Science Information Technology at Jomo Kenyatta University of Agriculture and Technology.

APRIL 2024

DECLARATION

I Kevin Mutugi Kithinji declare that this is my project proposal and has never been submitted to this or any other University for the award of Degree or any other award.

Student: Kevin Mutugi Kithinji

Sign……. Date…….

Supervisor: Francis Thiong'o

Sign…… Date…….

ABSTRACT

The management of construction projects requires adequate techniques to support the continual exchange of information across the stakeholders. Onsite assembly is a critical stage for modular construction. Its success or failure depends on accurate information sharing among numerous stakeholders who unfortunately, often possess unsynchronized information. The aim of this project is to integrate blockchain and smart contracts into information flows used across the life cycle stage of a construction project.

The proposed solution involves developing smart contracts, developing a Decentralized Application (DApp) to access data in the blockchain using smart contacts. DApps are able to interact using smart contracts with blockchain and allow users to perform operations through a web user interfaces.

The research objective will be to develop a system that will facilitate the efficient and timely relay of information needed in critical decision making in buildings construction.

The specific objectives include conducting survey using interviews, designing a system prototype and validating and testing the developed prototype.

The project methodology involves developing prototypes that can be tested and user feedback received and if any changes are need then they are implemented.

The scope of the project is limited to developing a prototype of a web based interface that the users can interact with and developing smart contracts to run on the blockchain and evaluating its effectiveness.

The project's budget includes costs associated with requirements gathering, computational resources and project management.

The project requirements include access to building construction experts, expertise in blockchain technology and developing smart contracts, and access to computational resources.

The project schedule involves problem identification, requirements gathering, data collection, data analysis, design, prototype development, prototype testing, refinement, final implementation and documentation.

Output of this research is to determine the effectiveness of the new prototype in reducing the repetitive paper work, raising the trustworthiness of the approvals, and timely relay of instructions and approvals and disapprovals of requested changes.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

DApp - Decentralized Application is an application built on a decentralized network that combines a smart contract and a frontend user interface.

DLT – Distributed Ledger Technology is a type of data structure that exits across multiple computing devices, called nodes, which are generally spread over locations or regions throughout the internet.

TCP/IP - Transmission Control Protocol/Internet Protocol is a suite of communication protocols used to interconnect network devices on the internet.

POW - Proof of Work is a decentralized consensus mechanism used in blockchain technology that requires network members to expend significant computational effort to solve an encrypted hexadecimal number. This process is also called mining, and miners are rewarded for their work. Prof of work allows for secure peer-to-peer transaction processing without needing a trusted third party

POS - Proof of Stake is a consensus mechanism used in blockchain technology to validate transactions and create new blocks in a blockchain. Under PoS, validators are chosen based on the number of staked coins they have. Validators hold and stake tokens for the privilege of earning transaction fees.

P2P - Peer to Peer is a computing or networking decentralized platform whereby two or more network members interact directly with each other, without intermediation by a third party.

Definition of terms

Smart Contract - It is a program that is stored on a blockchain that runs when predetermined conditions are met.

Blockchain - It is a distributed database that is maintained across computers linked in a network that maintains a continuously growing list of ordered records called blocks that are linked using cryptographic hashes. Each block contains a cryptographic hash of the previous block, a timestamp and transaction data.

CHAPTER ONE
INTRODUCTION

1.1 Research Background

In the buildings construction industry when a construction project is being carried out there is a book called instructions book. This book keeps a record of instructions written down by the construction's stakeholders mainly being the structural engineer(in charge of ensuring the structure of the building is intact), architect, quantity surveyor(in charge of making the cost estimates of the construction of the building), project manager(in charge of ensuring all other stakeholders work as a team), mechanical engineer(in charge of water pipes, drainage, firefighting utilities) and electrical engineer(in charge of electricity cabling of the building). When a stakeholder goes to the site and there are instructions given to the workers on the ground, these instructions must be written in the instructions book. Currently the engineer must have his/her own copy, the architect must also have his/her own copy, and one copy must remain at the site. This instructions book is important in that if it's the engineer who was at the site, the architect can come and read what the engineer has instructed, if it's the architect who was at the site the engineer can come and read what the architect has instructed, or if it is both the engineer and architect who were at the site, the owner/client can come and read what they have instructed. Also the county inspectors use this instructions book when they occasionally come to inspect if the building construction is being done according to set standards. Current method used to document these instructions is through a manual book. Sometimes someone can remember of an instruction they forgot to write in the book and they are far away from the site and decide to write these instructions in a WhatsApp group that they may have formed for the project. The proper recording and storage of these instructions whether in the manual book or WhatsApp messages is vital because they are admissible in court should anything happen and there need to be proof of them being recorded. The issue with this mode of record keeping is it prone to loss or damage and also duplication of records. As much as WhatsApp is a modern technology some people may not see the messages, for instance if they have so many groups and contacts, some people overlook checking messages in some groups or they tend to just focus on the top messages only. Others will just take a peek at the

message and assume that if it is from a certain stakeholder, then it does not concern them and they wouldn't read the whole message.

When the client requests any type of changes in the building, these changes must be assessed and approved by all the stakeholders especially the architect and the engineer. The instructions and the approval from each stakeholder must be documented in the instructions books. There are cases where some stakeholders may give their approval through WhatsApp and as stated earlier this can be challenging as an effective mode of communication.

KOM Consult is a consultancy firm that deal with supervision and construction of buildings that range from homes, mansions, apartments to high-rise buildings. They provide the engineering expertise in buildings constructions. While there are systems to support document management, communication between them is limited and mainly involves activities susceptible to human error.

Owing to its decentralized consensus mechanism, blockchain has the potential to improve information sharing effectiveness on construction management. Blockchain can store data securely and easily for query on the chain, providing support for the long construction cycle and reducing unnecessary work. The focus of the proposal is on reducing human error and increasing the reliability and transparency of decision making process on construction sites. Blockchain is an important research area that can be used to improve information sharing effectiveness on construction management.

Blockchain technology belongs to the wider digital ledger family, of which there are three fundamental types: centralized, decentralized and distributed. The research area will be on distributed ledger technology. The Distributed Ledger Technology (DLT) is a type of data structure that exits across multiple computing devices, called nodes, which are generally spread over locations or regions throughout the internet (IP/TCP), which acts as the base technology for information sharing. The ledger contains records i.e. transactions, collected into blocks, which are linked using cryptography (V. Ciotta et al., 2021).

A blockchain has four interdependent core layers:

1. Ledger which is a record of transactions grouped into blocks.
2. Peer to Peer network.
3. Protocol comprising governance (consensus rules).

4. Application layer which contains relations (e.g. smart contracts) that are allow information to flow through the system.

Permission less blockchain use proof based consensus algorithms, including Proof of Work (POW) and Proof of Stake (POS), which are the most common. These blockchains are also public (e.g. Bitcoin and Ethereum) since anyone can join the network (V. Ciotta et al., 2021).

The main properties of blockchain include:

Distribution – information is recorded by distributing it among several nodes to ensure IT security and system resilience.

Traceability – each transaction on the register is traceable in every respect and can be mapped back to its precise origin.

Disintermediation – blockchain platforms allow the management of transactions without intermediaries: in other words without the presence of trusted central bodies.

Transparency – the content of the register is transparent and visible to everyone in the public blockchain as well as easily accessible and verifiable.

Immutability –once written into the register, the data cannot be changed without the network consent.

Trust – this is built by the P2P network via the consensus mechanism, with no need for intermediaries, even though there is no trust among the parties involved.

Opportunity to program transactions – it is possible to schedule actions that take place when certain conditions occur on the blockchain using smart contracts (V. Ciotta et al., 2021).

In the context of construction information flow, blockchain play a crucial role in enabling to bypass the need for emails and other more traditional transmission channels during construction project. This is achieved by certifying all the information containers exchanged and their corresponding information flows on the blockchain. This produces a universal and reliable source of information for the stakeholders both during and following the construction process.

1.2 Problem Statement

There is a lack of proper communication between the client, contractor, architect and engineer, especially when a client wishes to make changes to certain elements of the structure under construction and it is implemented without the consultation of either the architect or the engineer or in some cases both. This can lead

to comprises in the integrity of the structure. There is a need to come up with a system that bypasses obsolete and incomplete data exchange processes, while concurrently providing a blockchain tool to create an immutable, trustworthy source that assembles the entire storyline of the structural safety information exchanges that take place during the building process.

1.3 Proposed Solution

The proposed solution is to develop a web user interface that will be deployed as a decentralized application that communicates using smart contracts running in the blockchain. The system will involve a message delivery system that will alert all users of any new requests, recommendations and current progress in the field and on site.

1.4 Objectives

1.4.1 Main objective

The main objective of this research is to develop a system that will facilitate the efficient and timely relay of information needed in critical decision making in buildings construction.

1.4.2 Specific objectives

The specific objectives of this research are as follows:

i. To design the functions and databases for the application where only the authorized users are allowed to log in the system.
ii. To develop smart contracts for handling information flow.
iii. To develop a web user interface that will allow for entering details and viewing information/notifications.
iv. To implement and test the system to ensure that the designed functions can satisfy the designed goals.

1.5 Research Questions

The research questions for this project proposal are:

i. How can a decentralized messaging platform be created to ensure quick communication, while ensuring privacy?
ii. How can users be granted control over their personal information while maintaining compliance with relevant regulations and ensuring the security of their data?
iii. What are the key components required to establish a robust and secure blockchain infrastructure that enables seamless peer-to-peer transactions within the information flow platform?
iv. How can we leverage blockchain technology and decentralized application to provide a user-centric and inclusive information flow experience?

1.6 Justification

The development of an information flow application using blockchain technology will facilitate effective information flow in buildings construction industry. The proposed solution will also contribute to the advancement of blockchain technology research, particularly in the area of construction industry.

1.7 Scope

The scope of this project is limited to the development of a web user interface that will communicate with smart contracts that will run in blockchain. The application will be tested and evaluated for effectiveness.

CHAPTER TWO
LITERATURE REVIEW

2.1 Introduction

Blockchain is a distributed database which is shared among and agreed upon a peer to peer network. It consists of a linked sequence of blocks, holding time stamped transactions that are secured by public key cryptography and verified by the network community. Once an element is appended to the blockchain, it cannot be altered, turning a blockchain into an immutable record of past activity (Partala, J., 2018).

Once the block is full, nodes simultaneously perform Proof of Work (mathematical operations that are difficult to solve but whose correct solution is easy to verify. These mathematical operations are indispensable to the operation of the system, as they force the verifying nodes to expend processing power which would be wasted if they included any fraudulent or invalid transactions. The first node that succeeds in solving a proof of work problem broadcasts the solution, along with the block of transactions and when 51% of the processing power of the network votes to approve a block, nodes begin recording new transactions to a new block, amending them to all previous blocks(CoinMarketCap, 2018,).

The blockchain technology solves double-spend problem with the help of public-key cryptography, whereby each user is assigned a private key, and a public key is shared with all other users. The main idea of the blockchain is a distributed database comprising records of transactions that are shared among participating parties. Every transaction is verified by the consensus of most of the participants in the system, making fraudulent transactions unable to pass collective verification. Once a record is created and accepted by the blockchain, it can never be altered (Zhao et al., 2016).

This allows for the creation of a jointly generated electronic time stamp that all participants can trust, even if they do not trust one another. In this manner it is easy to verify the origin and accuracy of the information whatever its source. No external intermediary (such as a centralized server) trusted by all the parties is required to validate the data (Seebacher et al., 2017).

2.2 Performance Evaluation

Factor	Variable
Functionality	Security levels No of users Permission management Error reporting scalability
User friendliness	Ease of use of graphical interface Ease of learning User manuals
Time	Duration Availability of necessary modules Completeness Interoperability

Table 1: Performance Evaluation

2.3 Research Scope and Limitation

Considering the construction industry, blockchain technology has been employed in different fields, such as supply chain management, risk management, smart contracts, logistics, carbon estimation, building information modeling (BIM), the IoT and sustainability. Blockchain's potential in the construction industry should be thoroughly analyzed to glean insights from various stakeholders' points of view. The use of Blockchain to enhance construction procedure management and service delivery necessitates identifying new trends, providing research findings, and suggesting prospective future research pathways.

2.4 Significance of Study

Blockchain technology provide a traceable and transparent track for products and materials, but it can also give vital information that may be utilized in decision-making as a foundation plan for the decommissioning of a structure and the reuse of the materials in every construction project through the appropriate

understanding of their properties and composition. Smarter and more sustainable methods may be implemented due to technological innovation and improvements in the building sector. Blockchain technology has the potential to improve several areas, including information flows, and the simplicity with which approvals are tracked in real time.

2.5 Expected Research Output

Output of this research is to develop a web based application that will interact with smart contracts running in blockchain.

2.6 Research and System Methodology

The proposed research methodology approach for the DApp combines qualitative and quantitative research methods to gain a comprehensive understanding of the decentralized application ecosystem and the adoption of the DApp. Here's an outline of the research methodology:

i. Research Design: A mixed-methods research design combining qualitative and quantitative approaches will be used to gather both subjective and objective data.
ii. Qualitative Research (Exploratory): Conduct interviews and focus groups with users, developers, and stakeholders to explore their perceptions, experiences, and expectations related to decentralized application. This qualitative research will provide insights into user needs, challenges, and potential improvements.
iii. Quantitative Research (Evaluative): Collect and analyze quantitative data through surveys, user analytics, and platform usage metrics. This data will help measure and evaluate the adoption, usability, and effectiveness of the DApp ecosystem.
iv. Merge and integrate the qualitative and quantitative findings to gain a comprehensive understanding of the adoption process, user experiences, and the impact of the DApp ecosystem.
v. Compare and contrast qualitative and quantitative results to identify converging themes, validate findings, and provide a holistic interpretation of the research outcomes.

vi. Discuss the limitations of the study, such as sample size or potential biases, and provide recommendations for future research in the field of decentralized application adoption.

2.6.1 Key Tools and Techniques

i. Requirements Elicitation and Analysis
ii. Iterative Development
iii. Prototyping and User Feedback
iv. Test-Driven Development (TDD)
v. Continuous Integration and Deployment

2.6.2 Proposed System Implementation Methodology

The proposed system implementation methodology for the DApp is Object Oriented System Analysis and Design Methodology (OOSADM). OOSADM is a systematic approach that focuses on the analysis, design, and implementation of object-oriented systems.

i. Analysis Phase:
    a) Identify the requirements and objectives of the DApp.
    b) Gather insights from stakeholders through interviews and consultations.
    c) Use modeling techniques (e.g., use case diagrams, activity diagrams) to visualize system requirements.
    d) Define the scope, boundaries, and constraints of the DApp.
ii. Design Phase:
    a) Create a conceptual model using object-oriented analysis techniques.
    b) Represent entities and their relationships with class diagrams.
    c) Design a user-friendly interface based on user research and feedback.
iii. Implementation Phase:
    a) Code the DApp using object-oriented programming principles.
    b) Implement features based on design specifications.
    c) Perform rigorous testing and debugging for correctness and reliability.
iv. Deployment and Maintenance Phase:
    a) Deploy the DApp on a suitable hosting platform.
    b) Monitor performance and promptly address issues.
    c) Continuously update and enhance the application based on user feedback.

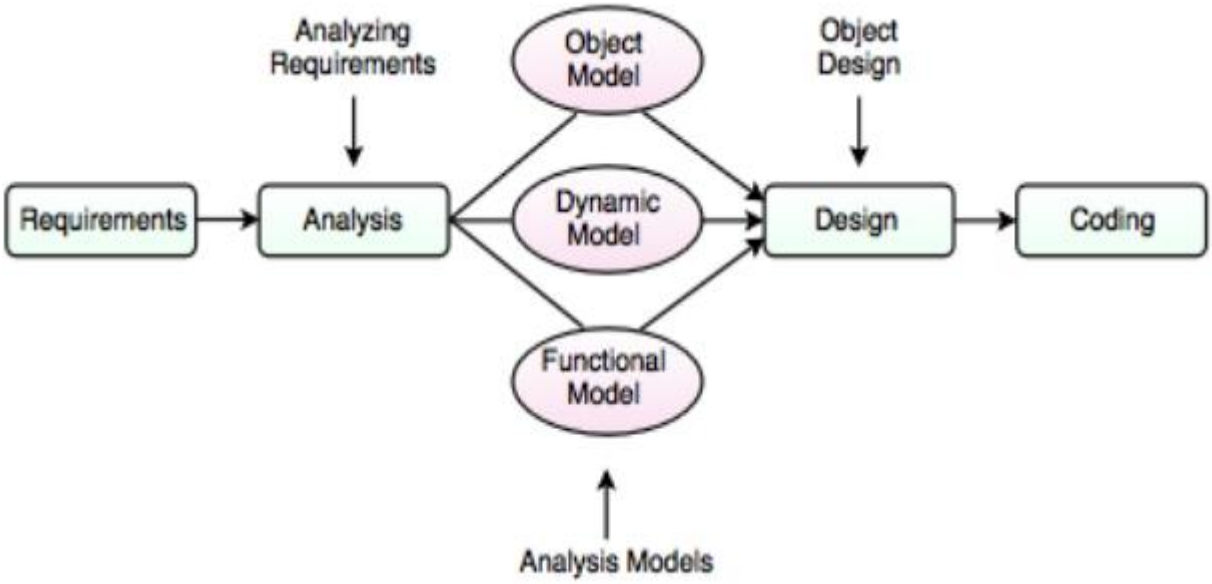

Figure 1: System Methodology

2. 6. 3 Conceptual Framework

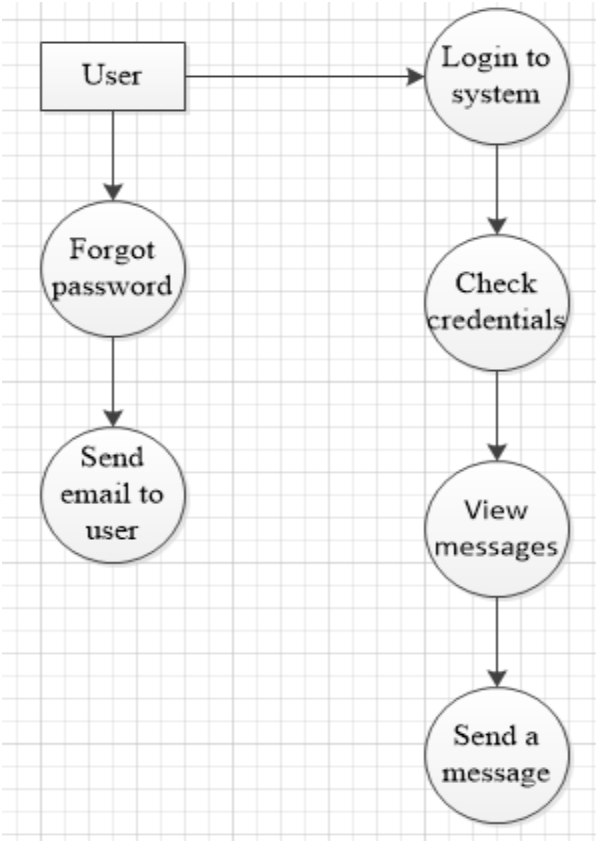

Figure 2: Conceptual Diagram

CHAPTER THREE
SYSTEM ANALYSIS AND DESIGN

3.1 Introduction

The chapter on system analysis and design will cover the following topics:

i. Identify the requirements: This involves understanding the needs of the users and determining what the application will do to meet those needs. It's the foundational step where I gain a deep insight into the DApp landscape, engaging with stakeholders, and exploring the multifaceted expectations of my user base.

ii. Modeling the System: Once the requirements are meticulously gathered, I move on to system modeling. This is where I take abstract ideas and transform them into a tangible and structured representation of my DApp application. Through modeling techniques, I will craft a roadmap that outlines the architecture, components, and interactions within the system.

iii. Enhancing User Experience: In the world of DApp, user experience (UX) is a key driver of adoption. Therefore, I will explore the principles of effective UX design, focusing on creating an intuitive and user-friendly interface. By placing the user at the center of my design decisions, I enhance accessibility and usability, making my DApp application appealing and functional for all.

iv. Security by Design: Security is a non-negotiable facet of DApp. Throughout this chapter, I will underscore the importance of security considerations at every stage of the system analysis and design process. From threat modeling to robust access controls, I will explore strategies to safeguard user assets and data.

3.2 System Development Methodology

For my DApp project, I will adopt the Object-Oriented System Analysis and Design Methodology (OOSAD) as the central approach to guide my development process. OOSAD is ideally suited to address the intricacies and unique challenges posed by the decentralized application (DApp) landscape. It provides a structured framework that emphasizes the modeling of objects and their interactions, ensuring a secure and scalable application architecture.

OOSAD is founded on key principles that align seamlessly with my DApp project's objectives:

i. Object-Centric Approach: OOSAD centers its approach around objects, modeling the various components and entities within my application, such as smart contracts, and user profiles. This method creates a modular and comprehensible design that simplifies system development.
ii. Abstraction and Encapsulation: OOSAD prioritizes abstraction and encapsulation, allowing me to conceal internal object details while exposing only the necessary functionalities. This not only enhances security but also promotes code reusability and maintainability.
iii. Inheritance and Polymorphism: By leveraging inheritance and polymorphism, OOSAD enables me to establish hierarchies of objects and define common behaviors. This proves invaluable in DApp, where multiple components may share similarities in functionality.
iv. Use Case Modeling: OOSAD incorporates use case modeling to outline how users will interact with the system. This ensures that my application is designed with the end-users in mind, aligning closely with their needs and expectations.

Finally in the realm of decentralized application (DApp), where innovation and security converge, the adoption of the Object-Oriented Analysis and Design (OOSAD) signals a new era of systematic development.

3.3 Feasibility Study

This feasibility study examines the financial, economic, and technical aspects of my DApp (Decentralized Application) application project. In addition, a crucial focus is placed on security measures to ensure the protection of user assets and data. The application would need to allow listing, sending, receiving and notification of messages. In order to be technically feasible, the application would need to be cross platform compatible with a variety of browser. The application would also need to integrate digital wallets such as metamask or coinbase.

3.3.1 Technical Feasibility

The proposed application is technically feasible. The required hardware and software are readily available and the necessary development tools are well understood. The developer has the required skills and experience to develop the application.

The application can be designed to meet the needs of the target users, and the developer has a clear understanding of the development process. The application will be developed using industry standard tools and techniques and will be tested thoroughly before release.

3.3.2 Financial Feasibility

i. Development Costs: The development of my DApp system application is financially feasible. An estimated cost of Ksh 4,800 has been allocated for the development phase. This cost encompasses various components, including labor, materials, and overhead expenses.
ii. Labor Costs: A significant portion of the budget is allocated to skilled developers, engineers, and other professionals who will be instrumental in bringing my DApp application to life. Their expertise and dedication are key factors contributing to the financial feasibility of the project.
iii. Materials and Resources: Beyond labor costs, my budget includes resources such as hardware, software licenses, and other essential tools required for development. These investments are essential to ensure that my application is built on a solid foundation and adheres to the highest technical standards.
iv. Overhead Expenses: Overhead expenses are also factored into the budget. This includes costs associated with project management, administrative support, and other operational needs. These expenses are necessary to keep the project running smoothly and efficiently.
v. Return on Investment (ROI): Based on my financial projections, I expect the project to yield a positive return on investment. Charging for application usage, implementing subscription models, or partnering with relevant businesses can provide sustainable revenue streams to surpass the initial development costs, ensuring that the project becomes financially self-sustaining over time.

ROI on Feasibility Study = (Net Benefits - Cost of Feasibility Study) / Cost of Feasibility Study

Net Benefits = Ksh 40,000

Cost of Feasibility Study= Ksh 4,800

ROI = (40000-4800)/4800

ROI= Ksh 7.3333

This means that for every Ksh 1 spent on the feasibility study, you gained Ksh 7.3333 in net benefits. This is a positive ROI, indicating that the feasibility study generated substantial value compared to its cost, and it's roughly 7.33 times the initial investment.

Conclusion

The proposed Decentralized application is feasible from a technical, financial point of view. Development of the application is recommended.

3.4 Requirements Elicitation

3.4.1 Data Collection

My data collection tools served as the backbone of my efforts to gather comprehensive requirements for my DApp (Decentralized Application) application. It was meticulously designed to facilitate the structured capture of information from various sources, including users, stakeholders, and experts in the field. The tools were versatile, adaptable to different data collection methods, and carefully crafted to ensure precision in data acquisition.

3.4.2 Preparation and Administration

i. Identification of Data Sources: The preparation phase commenced with the identification of key data sources. I recognized that the insights from end-users, stakeholders, domain experts, and existing documentation were invaluable in shaping my project's requirements.

ii. Tool Design: With an understanding of my data sources, I embarked on designing the data collection tool. It was created to accommodate multiple data collection methods, including interviews, observations, and questionnaires. The questions and prompts within the tool were thoughtfully structured to guide discussions and inquiries efficiently.
iii. Pilot Testing: I conducted pilot testing with a small group of respondents to validate the effectiveness of the data collection tool. This phase allowed me to identify any potential issues, ambiguities, or redundancies in the questions and prompts. The insights gained from pilot testing informed refinements and improvements.

3.4.3 Methods of Administration

I employed various data collection methods based on the nature of the data sources and research objectives

i. Interviews: One-on-one interviews were conducted with key stakeholders, domain experts, and potential end-users. These interviews were guided by the structured questions within the tool and allowed for in-depth discussions to uncover nuanced insights.
ii. Observation: In cases where direct observation was relevant, I employed this method to gather data. For instance, I observed user interactions with existing DApp platforms to gain insights into user behavior and pain points.
iii. Questionnaires: Questionnaires were distributed to a broader audience of potential users. These structured surveys contained questions aligned with my research objectives and were administered digitally or in print, depending on respondent preferences.
iv. Document Analysis: Existing documentation within the DApp space, such as whitepapers, reports, and academic literature, was analyzed to extract relevant insights and industry trends.

3.5 Data Analysis

In the following sections, we will delve into our data analysis with a particular focus on the viewpoints expressed by respondents concerning the system's non-functional requirements and functional requirements.

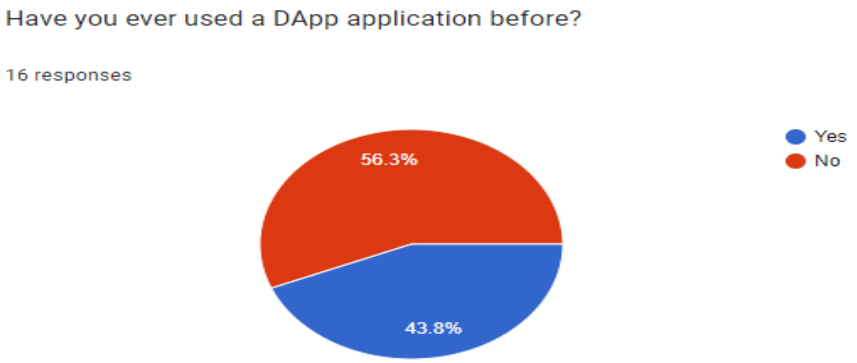

Figure 3

The purpose was to evaluate the respondents' prior experience and familiarity with decentralized application technologies. The analysis reveals that a significant majority of respondents, constituting 56%, have reported using DApp applications. Conversely, the remaining respondents, totaling 43%, have not engaged with DApp applications.

What is your level of familiarity with decentralized Finance?

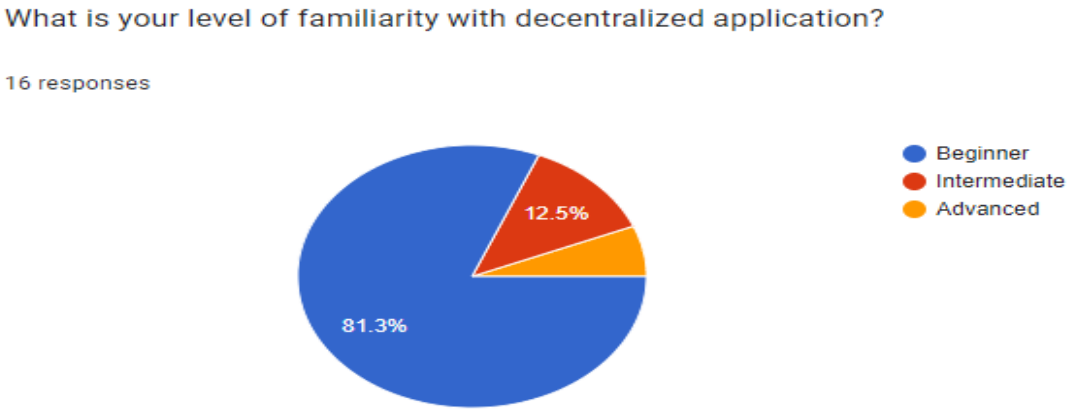

Figure 4

The purpose of inquiring about respondents' familiarity with DApp (Decentralized Application) is to gather insights into their prior knowledge and experience in the domain of decentralized application technologies. A majority of respondents,

accounting for 81%, fall into the beginner familiarity category, followed by 12% of respondents who report an intermediate understanding of DApp. Additionally, none of the respondents identify themselves as advanced in the realm of DApp.

How often do you use DApp application?

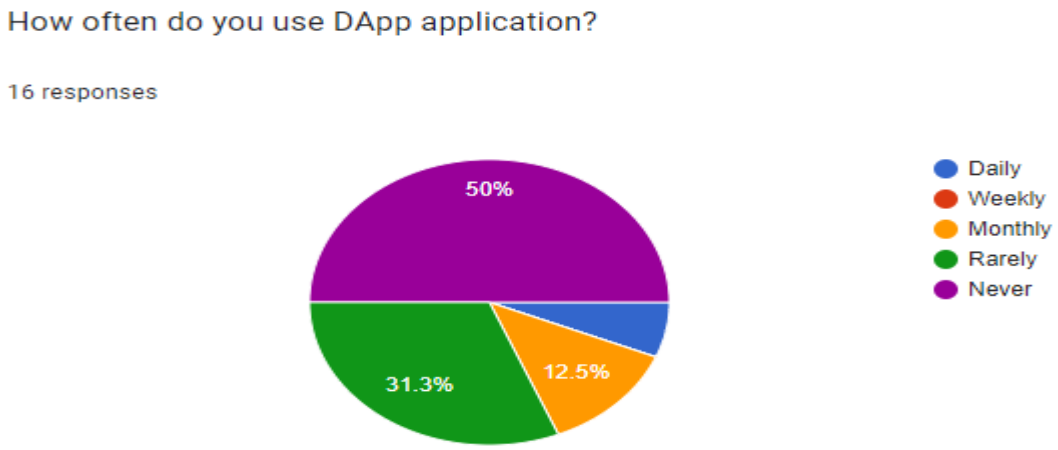

Figure 5

The purpose of the question "How often do you use a DApp (Decentralized Application) application?" is to evaluate the frequency and extent to which respondents engage with decentralized application technologies on a timely basis.

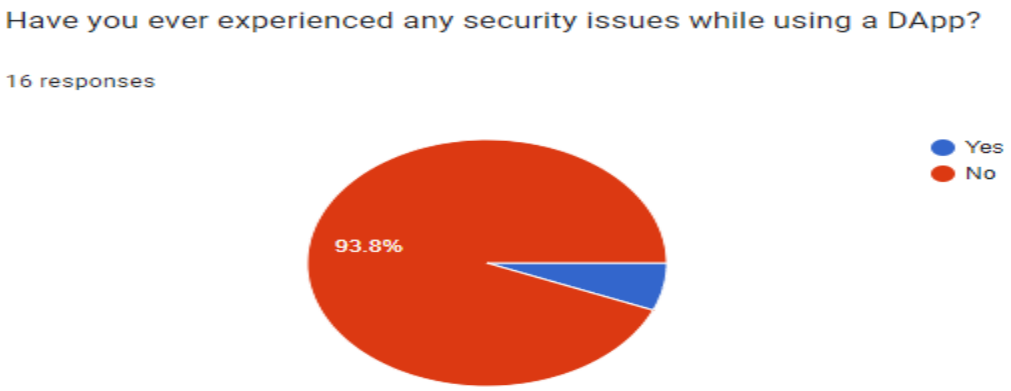

Figure 6

The intent of the question is to assess the respondents' personal experiences and concerns related to the security of DApp applications. This information is valuable for gaining insights into the potential risks and challenges users have encountered in the

DApp ecosystem. It helps in identifying areas that may require improvement in terms of security measures and user education within the DApp space.

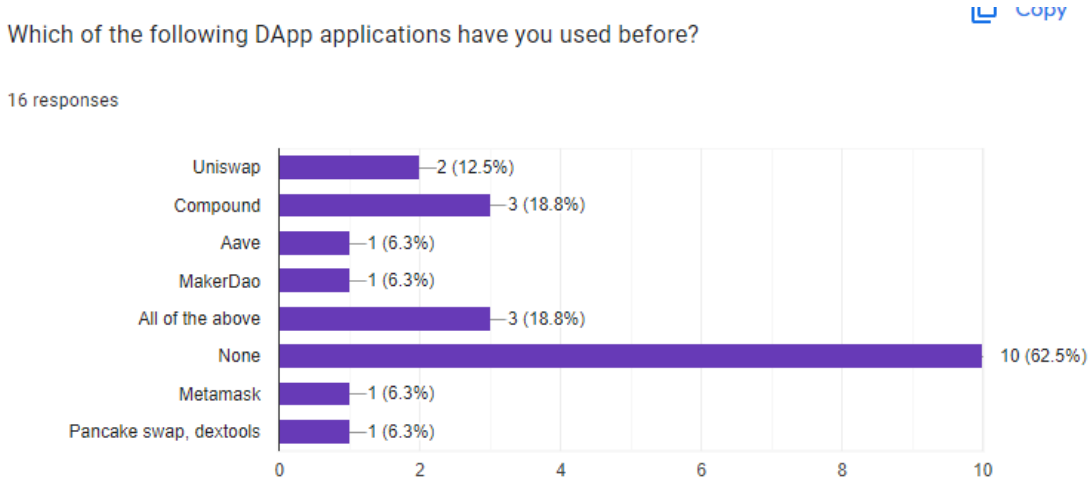

Figure 7

The Purpose of the question is to gather information about the respondents' specific experiences and usage of DApp platforms. This question aims to identify and understand the range of DApp applications that respondents have interacted with, helping to assess the diversity and popularity of various DApp services within the target audience.

How much do you trust DApp application in terms of security?

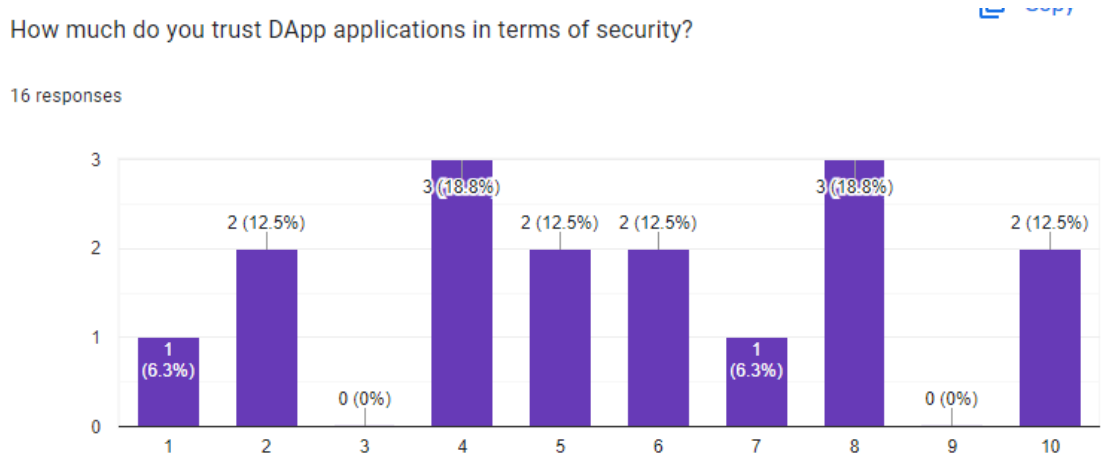

Figure 8

The intent of the question is to gauge the level of confidence and trust that respondents have in the security measures of DApp platforms. This question aims to assess the perception of security within the DApp ecosystem from a user's perspective.

3.6 System specification

3.6.1 Functional Requirements

In this section, we present the essential functional requirements that outline the system's expected behavior and operational capabilities. These requirements serve as a detailed guide for the actions the system must perform to fulfill user expectations and meet operational objectives. The functional requirements for the application are as follows:

i. The system must allow users to log into their account using metamask.
ii. The system must be able to connect and communicate with the smart contract.
iii. The system must allow a user to logout.
iv. The system must send messages to a user.
v. The system must allow a user to receive messages.

3.6.2 Non-Functional Requirements

Non-functional requirements describe the attributes of the system and not its actions. Non-Functional requirements are as follows:

i. Cross-Browser and device Compatibility
    a) The application should be compatible with any major web browsers i.e. Chrome, Firefox.
    b) The Application should adapt to various screen sizes and resolutions.
ii. Internet Connection

a) The application requires internet connection to connect with the blockchain nodes.

iii. Availability and Reliability

a) The application should not experience a long time of downtime.

iv. Usability and User Experience

a) The user interface must be user-friendly.

v. Security

a) Implement robust authentication measures, including encryption, to safeguard user data and assets.

3.7 Logical design

3.7.1 Use Case Diagram

The use Case Diagram depicts how users interact with the system and one another and how messages moves from sender to the recipients.

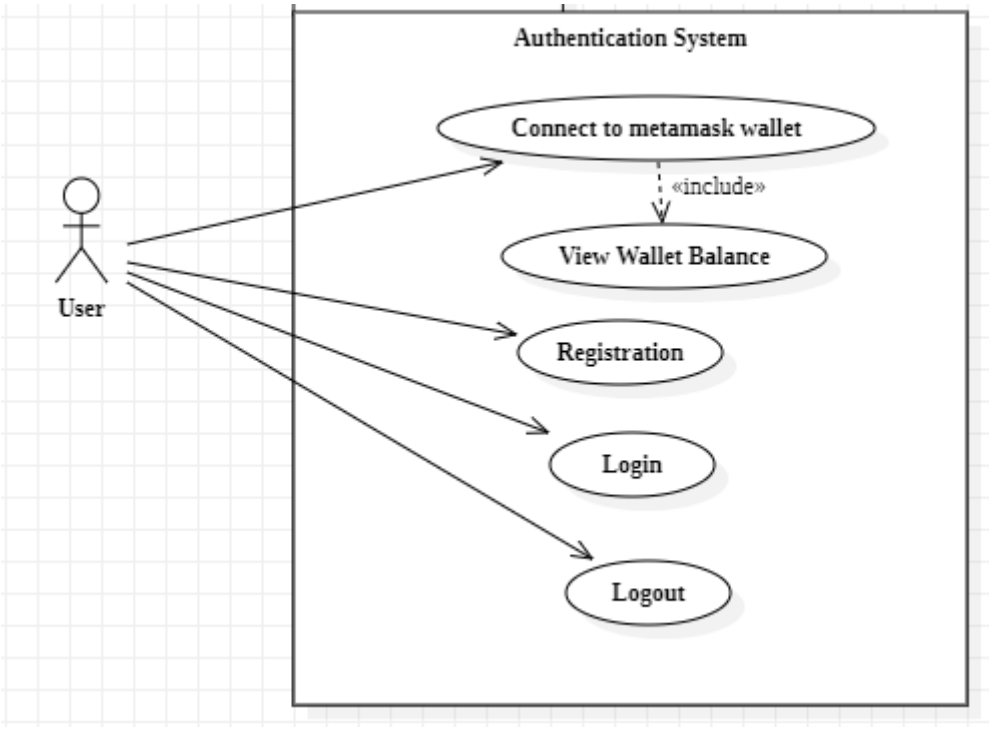

Figure 9: Authentication Use Case Diagram

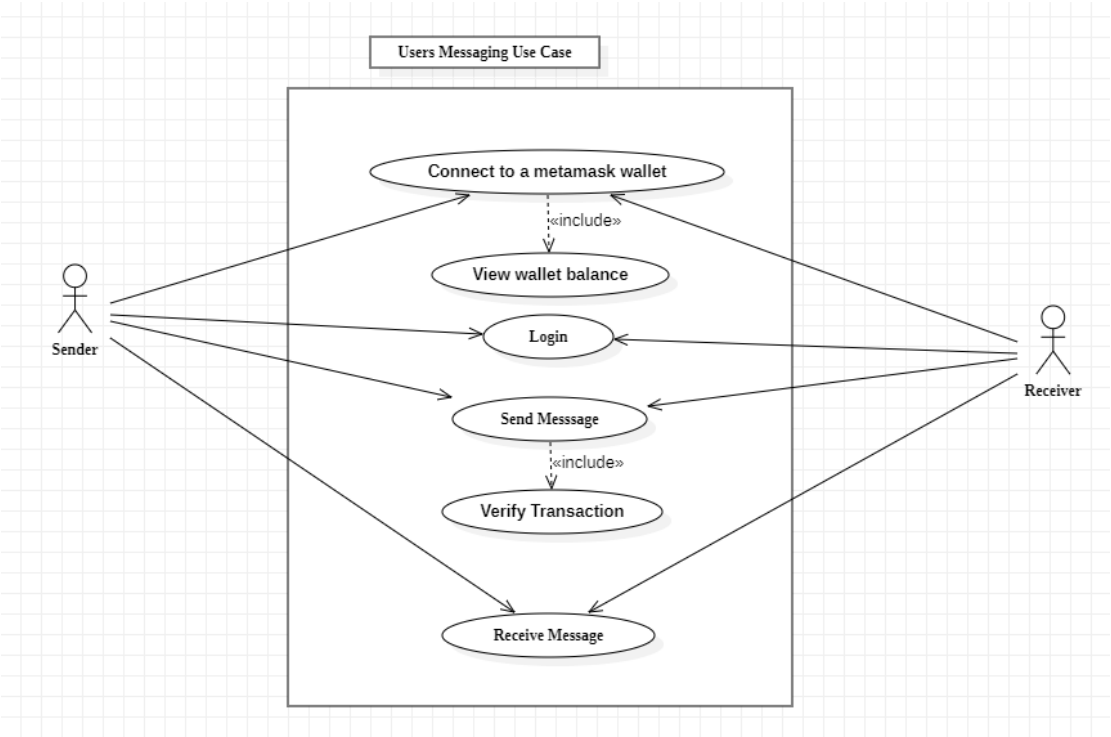

Figure 10: Messaging Use Case Diagram

3.7.2 Class Diagram

The class diagram shows how different classes interact with one another.

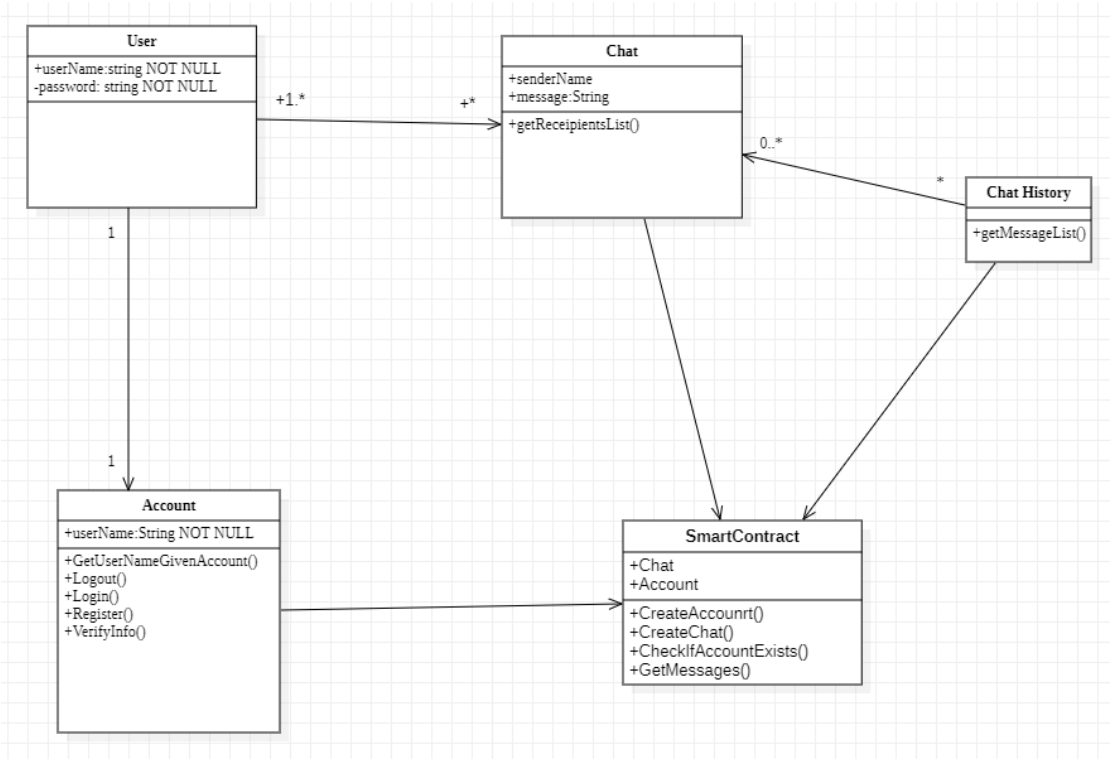

Figure 11: Class Diagram

3.7.3 Activity Diagram

Activity diagrams are visual tools that simplify complex processes by breaking them down into steps and decisions, aiding in process optimization and communication. Here it shows how different processes move from one point to another.

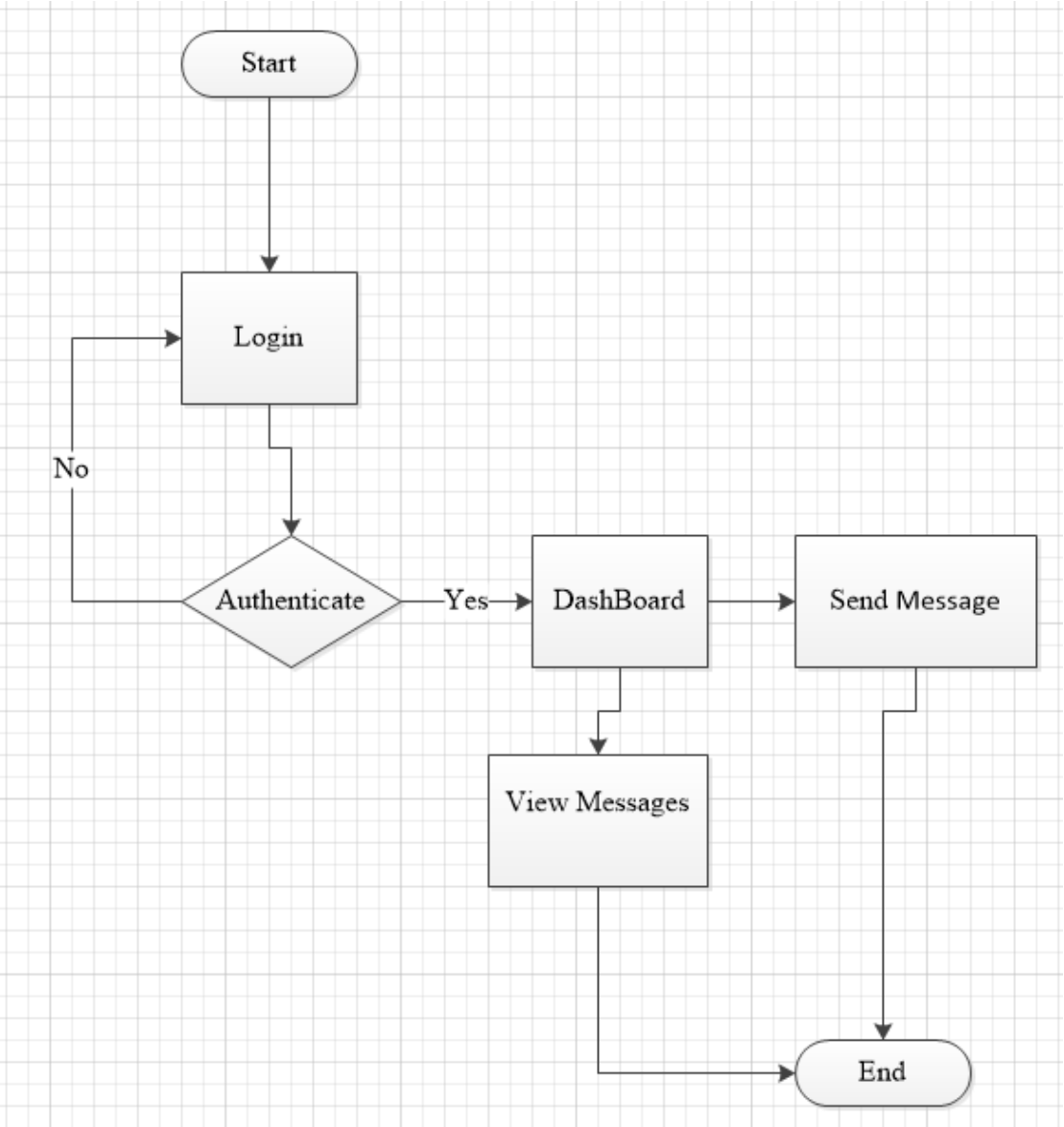

Figure 12: Activity Diagram

3.7.4 Sender Sequence Diagram

It shows the interaction of the Sender (user) and the system.

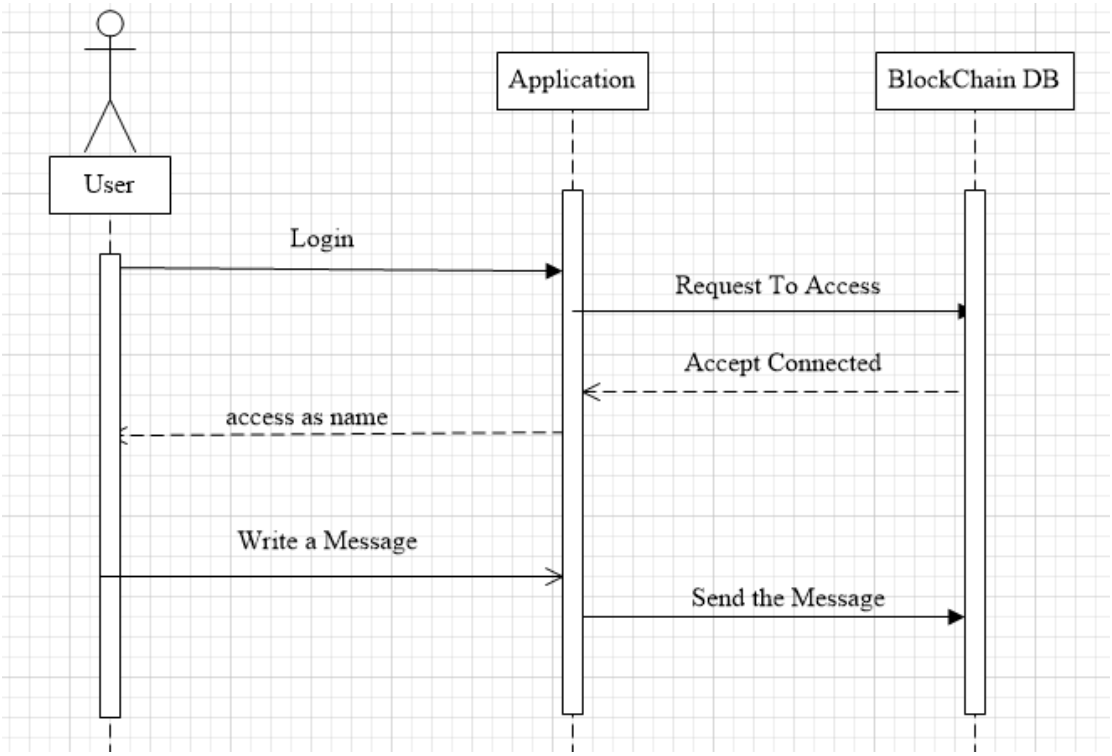

Figure 13: Sender Sequence Diagram

3.7.5 Receiver Sequence Diagram

It shows the interaction of the Receiver (user) and the system.

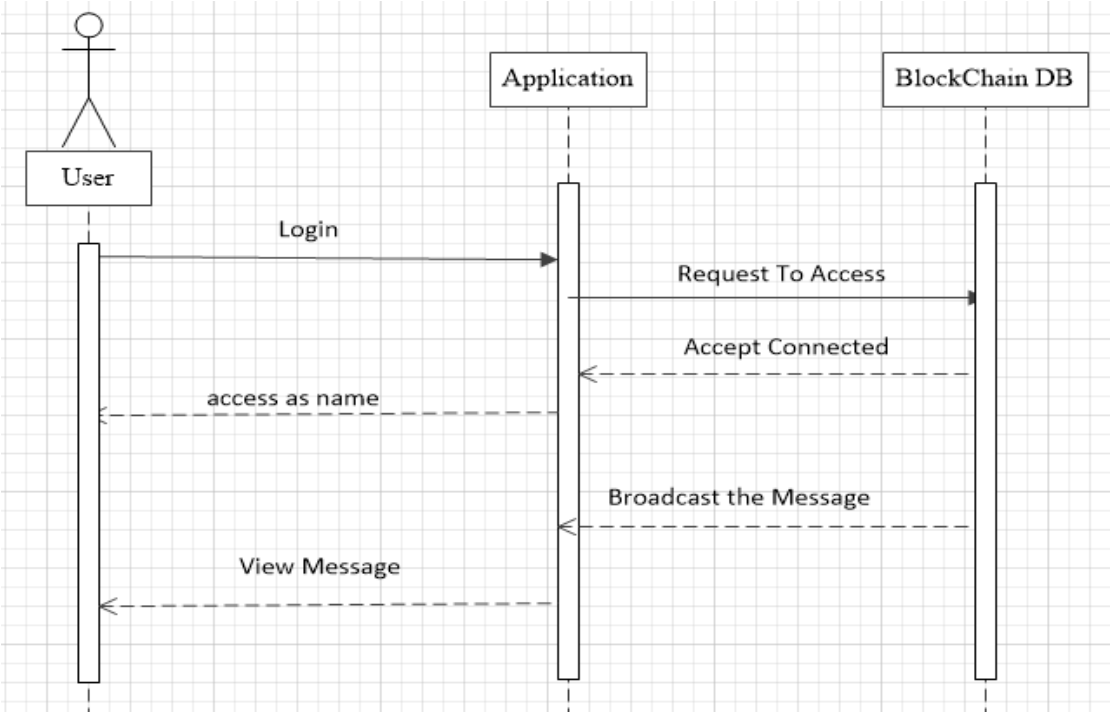

Figure 14: Receiver Sequence Diagram

3.8 Physical design

3.8.1 Entity Relationship Diagram

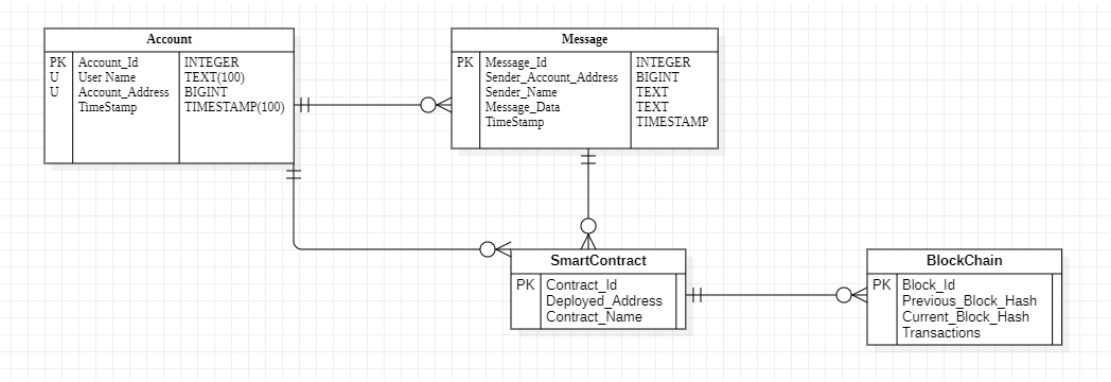

Figure 15: ERD Diagram

3.8.2 System Wireframes

These wireframes provide a visual blueprint, outlining the layout and structural elements of the user interface.

Figure 16: Read Messages Screen

Figure 17: New Message Screen

3.9 System Architecture

3.9.1 Architecture

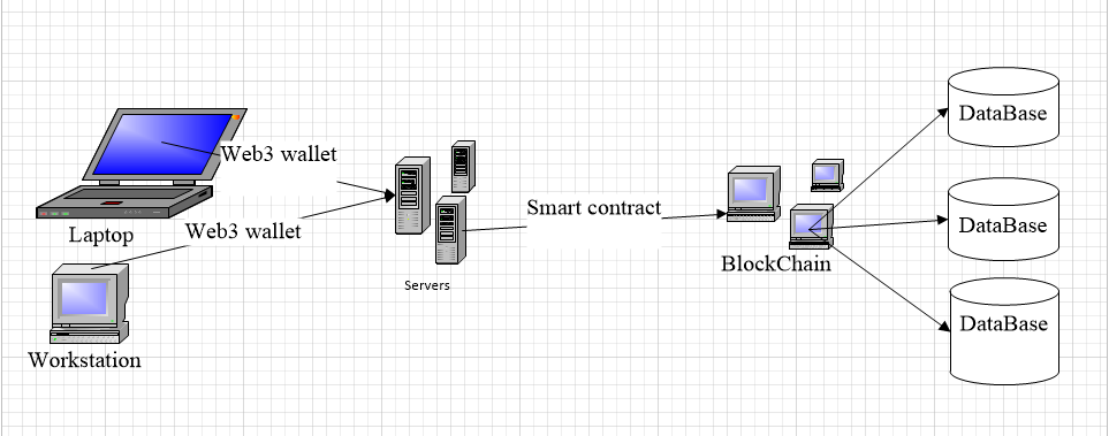

Figure 18: System Architecture

CHAPTER FOUR
SYSTEM IMPLEMENTATION AND TESTING, CONCLUSIONS AND RECOMMENDATIONS

4.1 Introduction

The chapter focuses on the implementation and testing of the proposed system. The implementation part, explores different parts of the system, how they were implemented and how they function. The testing section of this chapter focuses on usability testing and functional testing to verify if the system attains the objectives of the proposed solution.

4.2 Environment and Tools

i. Ganache - a popular Ethereum development blockchain that allows you to create a local network with pre-funded accounts for testing smart contracts without using real Ether.
ii. Truffle - a development framework that simplifies the process of building, testing, and deploying smart contracts on Ethereum.
iii. Web3.js - a backend JavaScript library that allows you to interact with the Ethereum blockchain from your application's frontend.
iv. Solidity Compiler: Required to compile Solidity smart contracts before deployment.
v. HTML, CSS, and JavaScript to create the front-end components and interact with the backend using Web3.js.
vi. MetaMask extension on the browser.

4.3 System Code Generation

The project's codebase has been meticulously developed using React library and Solidity as foundational technologies. Employing the React JavaScript framework, I achieved seamless integration of the model into the web application. Every facet, ranging from interfaces to functions and Smart Contracts, was meticulously synchronized during the coding phase. The integration process was orchestrated through the Smart Contracts, establishing a cohesive link between the interfaces and the blockchain. Presented herewith are illustrative snapshots providing insight into the systematic integration of these technologies within the project infrastructure.

```
// Login to MetaMask and check if the user exists else creates one
async function login() {

    try {
        //check if browser supports metamask
        let res = await connectToMetamask();
        if( res === true ) {
            //get the provider
            provider = new ethers.providers.Web3Provider( window.ethereum );
            provider.on("disconnect", disconnect_user);
            window.ethereum.on("accountsChanged", async()=>{localStorage.removeItem("account")})
            //force metamask to allow user to choose account
            const permissions = await window.ethereum.request({
                method: "wallet_requestPermissions",
                params: [
                    {
                        eth_accounts: {},
                    },
                ],
            });
            setHasWalletPermissions(permissions);
            console.log("has_wallet_permissions", has_wallet_permissions);
            //set the signer used when creating the contract
            signer = provider.getSigner();
            //get the metamask accounts
            let accounts = await provider.send("eth_requestAccounts", []);
            console.log(accounts.length);
            //if there is atleast one account the user has selected the first one
            if(accounts.length){
                console.log("You are connected to account : ", accounts[0]);
                setMyAccount(accounts[0]);
                localStorage.setItem("account", myAccount);
            } else {
                console.log("Metamask is not connected.");
            }
            console.log("CONTRACT_ADDRESS ", CONTRACT_ADDRESS);
            console.log("contractABI ", contractABI);
            console.log("signer ", signer);
            //construct the contract given an address, application binary interface and the signer
            const contract = new ethers.Contract( CONTRACT_ADDRESS, contractABI, signer );

            if(contract === null) return;

            console.log("contract created successfully. ", contract);
            console.log("contract ", contract);

            setMyContract( contract );
```

Figure 19: MetaMask Login Logic

```
//check if user is registered
let present = await contract.check_user_exists_given_account(address);

console.log("present ", present);

let username;
if( present )
    //if user is registered get the user name and login in
    username = await contract.get_user_name_given_account( address );
else {
    //else prompt for a user name and create a new account.
    username = prompt('Enter a username', 'Guest');
    if( username === '' ) username = 'Guest';

    console.log("calling contract...");
    //create a new account.
    await contract.create_new_account(username);

    console.log("account created in block chain.");

    //listen for create account event
    await contract.on("user_created_event", (message, user_address, timestamp, username) => {
        console.log("create account event raised.");
        console.log(message, user_address, timestamp, username);
    });

    const bal = await provider.getBalance(address) //balance in wei
    const balance = ethers.utils.formatEther(bal) // wei balance convert to eth balance

    console.log("balance ", balance);

}
//set the global variables upon login.
setMyName( username );
setMyPublicKey( address );
setShowConnectButton( "none" );
setShowLogoutButton("block");
set_App_Connection(true);
```

Figure 20: Registration/Login logic

```
// Sends message to users
async function sendMessage(message) {
    try{
        //check if user is logged in
        if(!is_app_connected) return;
        //validate the message is not blank
        if(message.length < 1)
        {

        }else{

            console.log("calling contract...");
            //save the message in blockchain
            await myContract.create_new_message(message);

            console.log("message created in block chain.");

            //listen for new message created event
            await myContract.on("message_created_event",  (message, user_address, timestamp, message_sent) => {
                console.log("create message event raised.");
                console.log(message, user_address, timestamp, message_sent);
            });

            //reload the newly created record to all users.
            setReloadData(true);
        }
    }catch(error)
    {
        console.log(error);
    }
}
```

Figure 21: Send Message logic

```
//create user event
event user_created_event(string msg, address sender, uint256 timestamp, string user_name);

//create message event
event message_created_event(string msg, address sender, uint256 timestamp, string message_data);

//error message event
event error_message_event(string msg, address sender, uint256 timestamp);

// It checks whether a user(identified by its public key)
// has created an account on this application or not
ftrace | funcSig
function check_user_exists_given_account(address pubkey1) public view returns(bool) {
    return bytes(user_list[pubkey1].user_name).length > 0;
}

// Registers the caller(msg.sender) to our app with a non-empty username
ftrace | funcSig
function create_new_account(string calldata user_name1) external {

    emit  user_created_event("validating..", msg.sender, block.timestamp, user_name1);

    require(check_user_exists_given_account(msg.sender) == false, "User already exists!");
    require(bytes(user_name1).length > 0, "Username cannot be empty!");

    //user_struct memory new_user = user_struct(last_user_id, msg.sender, block.timestamp);

    last_user_id++;

    emit  user_created_event("new account index created.", msg.sender, block.timestamp, user_name1);

    user_list[msg.sender].id = last_user_id;
    user_list[msg.sender].user_address = msg.sender;
    user_list[msg.sender].user_name = user_name1;
    user_list[msg.sender].timestamp = block.timestamp;

    user_addresses_list[last_user_id] = msg.sender;

    total_users_count++;

    emit  user_created_event("new account created..", msg.sender, block.timestamp, user_name1);
}
```

Figure 22: Create Account Backend logic

```
// Makes Cards for each Message
const Messages = activeChatMessages ? activeChatMessages.map( (message) => {
    let margin = "5%";
    let sender_name = message.sender_name;

    if( message.sender_address === myPublicKey ) {
        margin = "15%";
        sender_name = "You";
    }

    return (
        <Message key={ message.id } marginLeft={ margin } sender_name={ sender_name } msg={ message.msg } timestamp={ message.timestamp } />
    );

}) : null;
```

Figure 23: List Messages logic

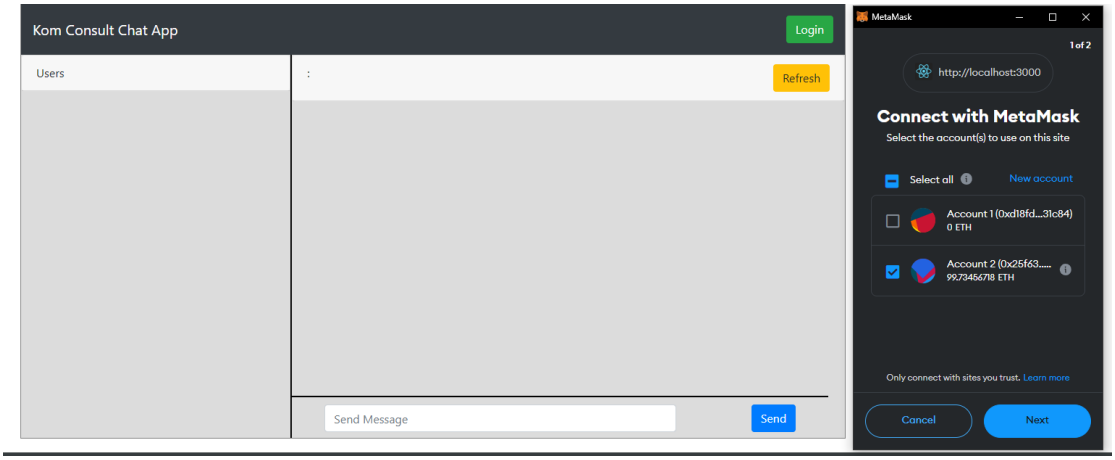

Figure 24: Connect Metamask Account screen

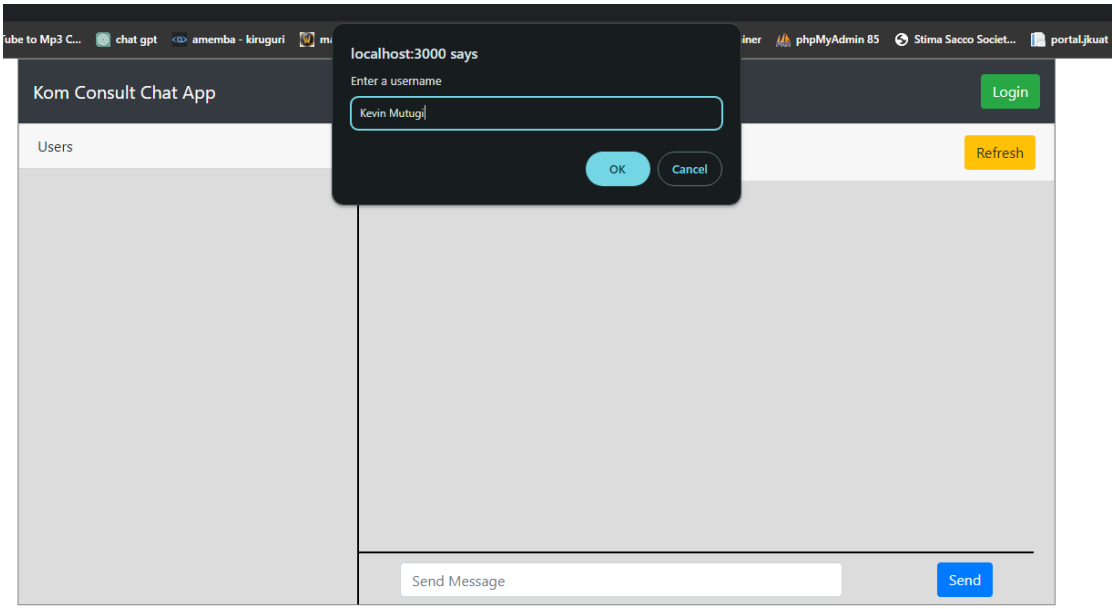

Figure 25: Register User screen

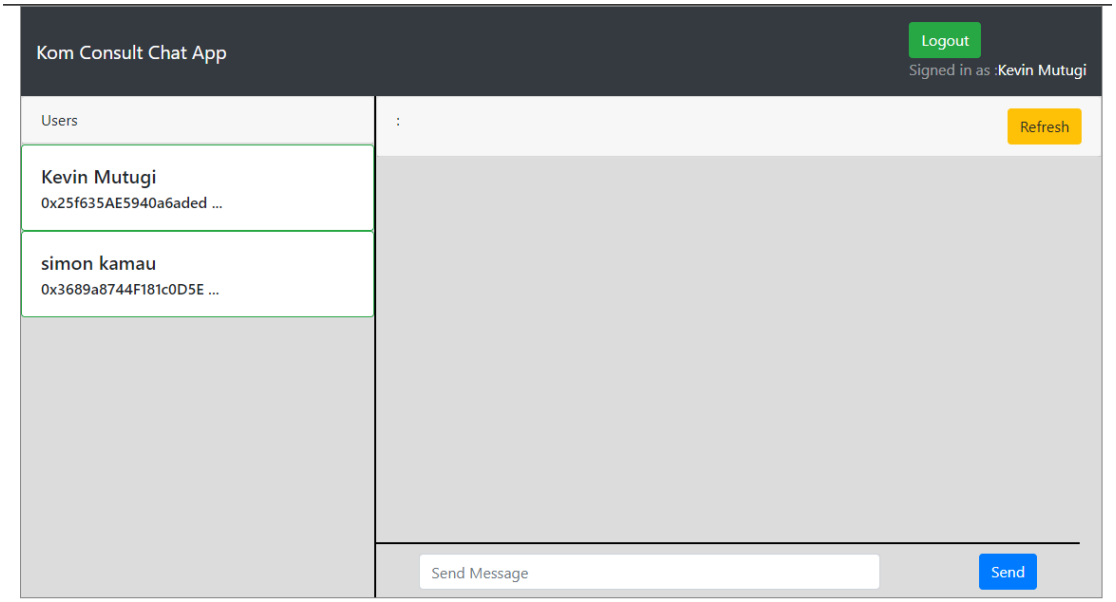

Figure 26: Landing Page screen

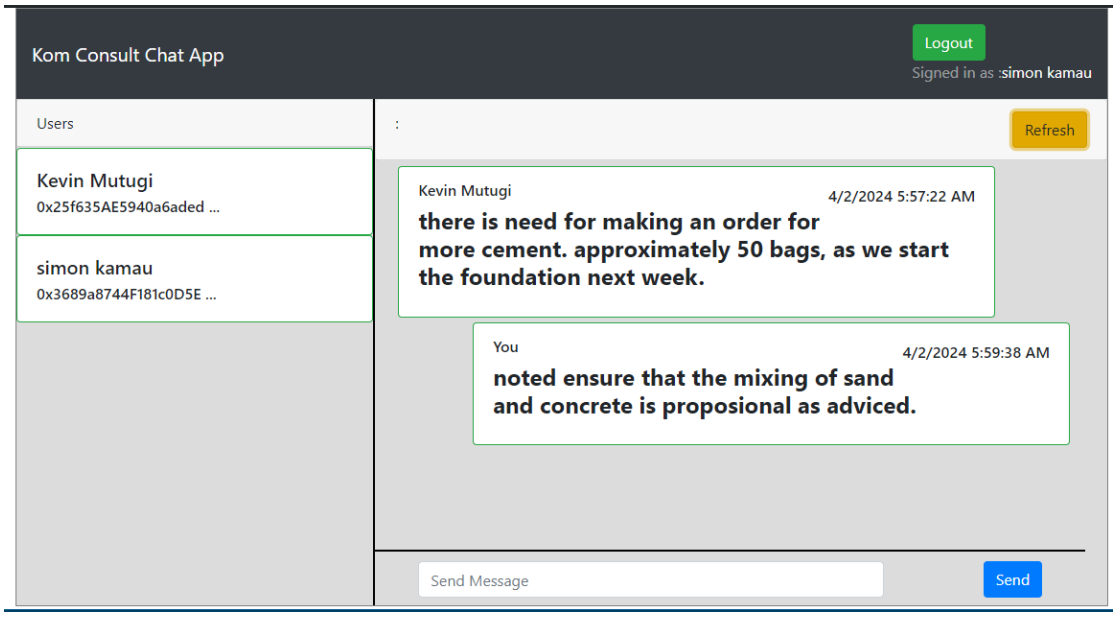

Figure 27: Messages List screen

4.4 Testing

The testing phase served as a critical checkpoint in my project. Through a series of rigorous tests, I aimed to validate the functionality, security, usability, and performance of the implemented system. Results indicate that the system met all functional requirements and performed as expected, and challenges such as performance bottlenecks in certain modules were navigated with the bottlenecks being addressed through code optimization and resource allocation. Types of tests include:

4.4.1. Functional Testing

The authentication functionalities through metamask wallet work seamlessly. However, during testing, a bug was identified in the authentication module where user account was not being captured correctly by the system. The issue has been logged and a fix completed.

Task To be Tested	Authentication through Metamask Wallet
Test Case	Ability to Authenticate a user into the system
Test Plan	● Navigate to the home page ● Click Connect button ● Metamask wallet pops up ● Accept connection
Expected Results	The system should display user wallet address and allow to perform system functionality
Actual Results	The system authenticated the user and display the user wallet address on the navbar

Table 2: Functional Testing

4.4.2. Usability Testing

User feedback highlighted a navigation challenge in the Dashboard section. A redesign is recommended to improve overall user experience and reduce friction in accessing the related features.

Task To be Tested	Log in Process
Test Case	Ability to authenticate users and navigate as well as result in the access of appropriate interfaces.
Test Plan	● Navigate Page to page ● Smooth login Through metamask ● View messages
Expected Results	The system should grant user access to it if they have metamask wallet.
Actual Results	The users were able to login into the system and view messages.

Table 3 : Usability Testing

4.4.3. Validation Testing

Task To be Tested	Input validation
Test Case	Validate empty inputs
Test Plan	● Don’t input value ● Submit an empty inputs value
Expected Results	The system should show a notification and reject empty data submission
Actual Results	The system shows a notification if an empty value is passed

Table 4 : Validation Testing

4.4.4. Integration Testing

In the integration testing phase, the collaboration between the smart contracts and the frontend components demonstrated successful integration, ensuring a cohesive and functional interaction. Specifically, the incorporation of smart contract functionalities, such as posting transactions from the user wallet to smart contract, seamlessly interfaced with the frontend user interface. Users were able to effortlessly execute these transactions through the frontend, with the smart contracts accurately processing and recording the actions on the blockchain.

However, during testing, a disparity was identified in the time taken to fetch the current transaction between the smart contracts and the frontend. This could potentially impact the accuracy of displayed information and lead to user confusion.

4.5 Conclusions

In concluding the testing phase, the results indicate that the system has met the specified testing objectives. The positive outcomes in functional, security, usability, performance, and integration testing affirm the robustness and reliability of the implemented solution.

4.6 Limitation

- Skills: Limited expertise in performance optimization impacted the depth of analysis in load testing. While the system's responsiveness under normal conditions was assessed, more extensive stress testing could have provided deeper insights into performance bottlenecks under extreme load.
- Tools: Budget constraints limited access to certain testing tools, influencing the thoroughness of UI automation testing. While manual testing provided valuable feedback on the user interface, automated testing tools could have accelerated the testing process and identified potential usability issues more efficiently.
- External Factors: Unforeseen delays in third-party integrations impacted the timing of testing for specific modules. While these delays were beyond our control, they influenced the overall testing schedule and required adjustments to the testing plan.

4.7 Recommendations

Building on the successful testing phase, recommendations for further improvement include:

- Investing in continuous monitoring: Implement continuous monitoring tools to ensure ongoing security and performance assessment. These tools should provide real-time insights into system behavior, allowing for proactive identification and resolution of potential issues. Regular security audits should also be conducted to maintain a high level of protection against evolving threats.
- User feedback integration: Establish a feedback loop with users for continuous usability improvements. This feedback loop can be facilitated through surveys, user interviews, and usability testing sessions. By actively gathering user feedback, the development team can identify pain points, prioritize usability enhancements, and ensure that the system meets the evolving needs of its users.
- Skills enhancement: Encourage team members to enhance skills in performance optimization for more comprehensive testing. This could involve attending training workshops, participating in online courses, or pursuing certifications in performance testing methodologies.

4.8 Appendix

To provide transparency and a deeper understanding of my data collection process, I have included the comprehensive data collection tool as an appendix to this report. This appendix contains detailed information about the tool's structure, specific questions, and prompts used, as well as the format for capturing data.

This data collection tool, along with the methods employed, forms the foundation upon which the development and implementation phases of my DApp project are built, aligning with my mission to deliver a user-centric and secure decentralized application.

4.8.1 Project Schedule

Activities	Duration (Weeks)	Proposed Start Date	Actual Start Date	Proposed End Date	Actual End Date	Deliverables
Problem Identification	1	02/10/2023	09/10/2023	15/09/2023	16/10/2023	Initial Project Idea
Requirements Gathering & Specification	2	17/10/2023	19/10/2023	20/10/2023	20/10/2023	Requirements Specification Document
Proposal Initiation	2	23/10/2023	24/10/2023	02/11/2023	03/11/2023	Project Concept Document
Proposal Writing	3	30/10/2023	31/10/2023	13/11/2023	13/11/2023	Project Proposal Document
System Design	3	25/11/2023	27/11/2023	4/12/2023	15/12/2023	Wireframes and Mock-ups
Implementation	8	18/12/2023	18/12/2023	19/02/2024	20/02/2024	Front-end and Back-end Codes
Testing	2	21/02/2024	21/02/2024	08/03/2024	10/03/2024	Test plan and Test Report
Review and Final Release	2	11/03/2024	12/03/2024	25/03/2024	25/03/2024	User Feedback Reports
Documentation	3	27/03/2024	27/03/2024	10/04/2024	12/04/2024	Final Project Document

Table 5 : Project Schedule

4.8.2 Gantt Chart

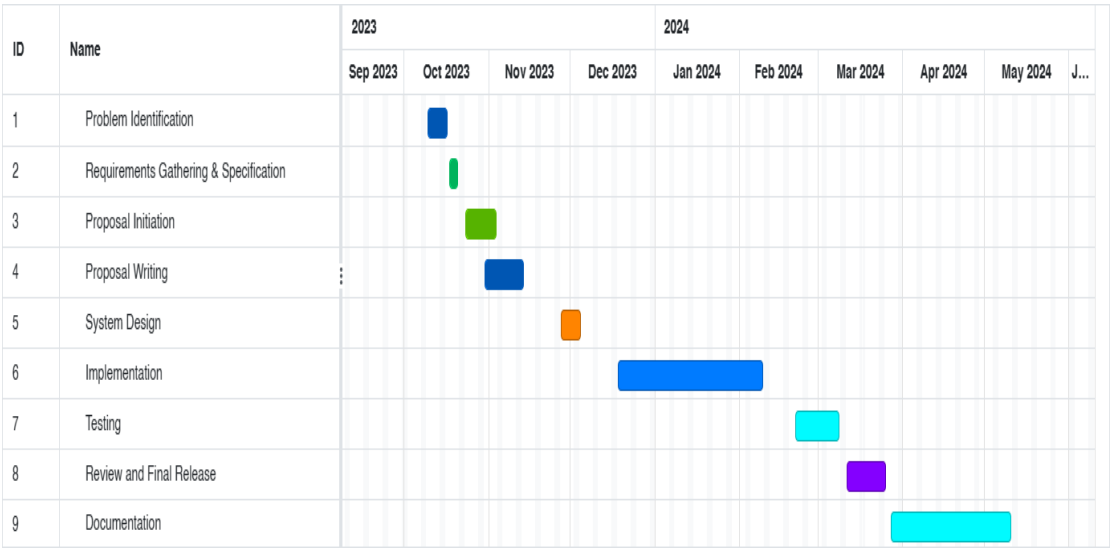

Figure 28 : Gantt Chart

4.8.3 Budget

Number	Item Description	Quantity	Amount (Ksh)
1	Laptop	1	50,000(provided)
2	External Hard disk	1	8000(provided)
3	Operating system	64 bits	provided
4	Bundles	10GB	3500
5	Transport		2000
6	Printing		2000
	Total		7,500

Table 6: Budget

4.8.4 Project requirements

4.8.4.1 Hardware requirements

The hardware requirements for this project include:

i. A computer with a minimum of 12 GB RAM and 500 GB storage.
ii. Flash disk with minimum 64 GB storage.

4.8.4.2 Software Requirements

The software requirements for this project include:

i. React Javascript programming language.
ii. Solidity programming language.
iii. Ganache for blockchain development environment.
iv. Vs Code for smart contract development.
v. Node js for web application development.
vi. Git for version control.
vii. Microsoft Office for documentation (MS word), analysis (MS Excel), Presentation (MS Power Point)

REFERENCES

CoinMarketCap. (n.d.). Retrieved May 10, 2018, from https://coinmarketcap.com/

Crosby, M., Pattanayak, P., Verma, S., & Kalyanaraman, V. (2006, 2). Blockchain Technology: Beyond Bitcoin. *Appl. Innov.*, 71.

Czepluch, J., Lollike, N., & Malone, S. (n.d.). The Use of Block Chain Technology in Different Application Domains Bachelor Project in Software Development. Retrieved March 13, 2018

Dobrovnik, M., Herold, D., Fürst, E., & Kummer, S. (2018, 2 18). Blockchain for and in Logistics: What to Adopt and Where to Start. *Logistics*.

Fernando, Y., & Saravannan, R. (2021, 4). Blockchain technology: Energy efficiency and ethical compliance. *J. Gov. Integr.*, 88–95.

Gatteschi, V., Lamberti, F., DeMartini, C., Pranteda, C., & Santamaria, V. (2018, 10 20). Blockchain and Smart Contracts for Insurance: Is the Technology Mature Enough? *Future Internet*.

Hassani, H., Huang, X., & Silva, E. (2018, 2 34). Big-Crypto: Big Data, Blockchain and Cryptocurrency. *BDCC*.

Partala, J. (2018, 2 ,18). Provably Secure Covert Commmunication on Blockchain. In *Cryptography*.

Seebacher, S., & Schüritz, R. (2017). Blockchain Technology as an Enabler of Service Systems: A Structured Literature Review. *Springer Nature*, 279, 12–23.

V. Ciotta, G. M. (2021). Integration of blockchains and smart contracts into construction information flows: Proof-of-concept,. *Automation in Construction,, Volume 132,*. Retrieved from https://www.sciencedirect.com/science/article/pii/S0926580521003769

Zhao, J., Fan, S., & Yan, J. (2016, 2 12). Overview of business innovations and research opportunities in blockchain and. *Finan. Innov.*

Zheng, Z., Xie, S., Dai, H.-N., Chen, X., & Wang, H. (n.d.). Blockchain challenges and opportunities: A survey. *Int. J. Web Grid Serv.*, 352–375. Retrieved 2018