

LAND REGISTRATION SYSTEM USING BLOCKCHAIN TECHNOLOGY

A PROJECT REPORT

Submitted by,

SHAIK MEERAVALI

- 20201CBC0017

RAMIREDDY HARISH KUMAR REDDY- 20201CBC0007

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING (BLOCKCHAIN)

Under the guidance of,

Ms. Suma N G

Assistant Professor School of CSE



PRESIDENCY UNIVERSITY, BENGALURU

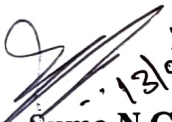
JANUARY 2024


PRESIDENCY UNIVERSITY


SCHOOL OF COMPUTER SCIENCE AND ENGINEERING


CERTIFICATE


This is to certify that the Project report "LAND REGISTRATION SYSTEM USING BLOCKCHAIN TECHNOLOGY" being submitted by "SHAIK MEERAVALI, RAMIREDDY HARISH KUMAR REDDY" bearing roll number "20201CBC0017, 20201CBC0007" in partial fulfillment of requirement for the award of degree of Bachelor of Technology in Computer Science and Engineering (Blockchain) is a bonafide work carried out under my supervision.


Ms. Suma N G
Project Supervisor,
Assistant Professor
School of CSE
Presidency University




Dr. S Senthilkumar
Professor & HOD
School of CSE
Presidency University


Dr. C. KALAIARASAN
Associate Dean
School of CSE&IS
Presidency University


Dr. SHAKKEERA L
Associate Dean
School of CSE&IS
Presidency University


Dr. Md SAMEERUDDIN KHAN
Dean
School of CSE&IS
Presidency University

Name and Signature of the Examiners

- 1)  [J. Venkata Giri]
- 2)  [Mary Diya Shamili D]

PRESIDENCY UNIVERSITY
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We hereby declare that the work, which is being presented in the project report entitled **LAND REGISTRATION SYSTEM USING BLOCKCHAIN TECHNOLOGY** in partial fulfillment for the award of Degree of **Bachelor of Technology in Computer Science and Engineering (BLOCKCHAIN)**, is a record of our own investigations carried under the guidance of **Ms. Suma N G, Assistant Professor, School of Computer Science and Engineering, Presidency University, Bengaluru.**

We have not submitted the matter presented in this report anywhere for the award of any other Degree.

SHAIK MEERAVALI 20201CBC0017

Shaik MeeraVali

RAMIREDDY HARISH KUMAR REDDY 20201CBC0007

R. Harish

ACKNOWLEDGEMENT

First of all, we indebted to the **GOD ALMIGHTY** for giving me an opportunity to excel in our efforts to complete this project on time.

We express our sincere thanks to our respected Dean **Dr. Md. Sameeruddin Khan**, School of Computer Science and Engineering & School of Information Science, Presidency University for getting us permission to undergo the project.

We record our heartfelt gratitude to our beloved Associate Deans **Dr. Kalaiarasan C** and **Dr. Shakkeera L**, School of Computer Science and Engineering & School of Information Science, Presidency University and **Dr. S Senthilkumar**, Professor & Head of the Department, School of Computer Science and Engineering, Presidency University for rendering timely help for the successful completion of this project.

We are greatly indebted to our guide **Ms. Suma N G**, Assistant Professor School of Computer Science and Engineering, Presidency University for his inspirational guidance, and valuable suggestions and for providing us a chance to express our technical capabilities in every respect for the completion of the project work.

We would like to convey our gratitude and heartfelt thanks to the University Project-II Coordinators **Dr. Sanjeev P Kaulgud**, **Dr. Mrutyunjaya MS** and also the department Project Coordinator **Dr. Srinivasan.T.R.**

We thank our family and friends for the strong support and inspiration they have provided us in bringing out this project.

SHAIK MEERAVALI

RAMIREDDY HARISH KUMAR REDDY

ABSTRACT

Blockchain technology serves as a decentralized, global ledger accessible to all participants within the blockchain network. Its potential lies in offering robust security measures for crucial information and data. The primary goal of this digital network is to establish an exceptionally secure database that is both traceable and operates on predefined protocols. In essence, each user within the blockchain network has access to a transparent record of the previous data, allowing for efficient management of their assets. These assets are underpinned by a single version of the truth, ensuring consistency and reliability. As users engage with the blockchain, their contributions and transactions are grouped together to form a block, thereby creating a secure foundation for establishing a comprehensive and immutable record. However, the challenge arises from the inherent complexity of maintaining a singular truth across a decentralized network. In certain instances, informational records may be incomplete or not fully digitized, leading to interactions that occur outside the digital realm, such as through voice communication or traditional paper documentation. This dual nature of the blockchain—providing a secure, traceable, and decentralized platform, while grappling with the occasional need for offline interactions—underscores the nuanced dynamics of its implementation. Blockchain's potential benefits in securing information and fostering trust are considerable, yet practical challenges persist in ensuring the completeness and digital nature of all transactions within the system.

LIST OF FIGURES

Sl. No.	Figure Name	Caption	Page No.
1	Figure 1.1	Use case Diagram	20
2	Figure 1.2	Class Diagram	22
3	Figure 1.3	Sequence Diagram	23
4	Figure 1.4	Collaboration Diagram	24
5	Figure 1.5	Deployment Diagram	25
6	Figure 1.6	Activity Diagram	27
7	Figure 1.7	Component Diagram	28
8	Figure 1.8	ER Diagram	29
9	Figure 1.9	DFD Diagram	31
10	Figure 1.10	Screenshots	46

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	i
	ACKNOWLEDGMENT	ii
	LIST OF FIGURES	iii
1.	INTRODUCTION	1-2
	1.1 Motivation	1
	1.2 Problem Statement	1
	1.3 Objective of the Project	1
	1.4 Scope	2
	1.5 Project Introduction	2
2.	LITERATURE REVIEW	3-4
	2.1 GENERAL	3,4
3.	RESEARCH GRAPHS OF EXISTING METHODS	5-6
	3.1 Existing Methods	5
	3.2 Challenges	6
4.	PROPOSED METHODOLOGY	7-8

5.	OBJECTIVES	9-
6.	SYSTEM DESIGN & IMPLEMENTATION	12-30
	6.1 Input Design Introduction	12-13
	6.2 Modules	14-15
	6.3 Functional and non-functional requirements	15-30
7.	TIMELINE FOR EXECUTION OF PROJECT	32
8.	OUTCOMES	33-35
9.	RESULTS AND DISCUSSIONS	36-38
10.	CONCLUSIONS	39-40

CHAPTER-1

INTRODUCTION

1.1 Motivation: The revolutionary impact of blockchain technology on asset registration lies in its creation of an inclusive, secure, and universally accessible ledger. Its ability to safeguard critical data inspires confidence in storing vital information. Through this digital network, a tamper-proof database is established, traceable via well-defined protocols. Users manage assets using cryptographic keys, ensuring a singular truth and enhanced security. Despite challenges like incomplete records, blockchain's adaptability accommodates physical interactions, bridging gaps through voice or paper. This transformative potential motivates the exploration of blockchain for robust, transparent, and reliable asset registration systems, ushering in a new era of secure asset management.

1.2 Problem Statement: Existing asset registration systems face challenges in maintaining a universally accepted truth. Current databases lack a seamless, secure, and globally accessible framework. Vital information often exists in fragmented, non-digital forms, requiring physical verification through voice or paper records. This discrepancy leads to inconsistent asset management and compromised data integrity. The absence of a unified, traceable system hampers efficient asset registration processes. This problem underscores the need for integrating blockchain technology to establish a robust, transparent, and universally accessible ledger for accurate and secure asset registration.

1.3 Objective of the Project: The aim of "Asset Registration Using Blockchain" is to leverage blockchain's distributed ledger technology to establish a secure, transparent, and immutable system for registering and managing assets. This seeks to streamline asset ownership, ensuring accuracy, and reducing

discrepancies through decentralized verification. By employing blockchain protocols, the objective is to create a universally accessible platform where users confidently register assets, overcoming challenges in maintaining a singular truth. The goal is to enhance reliability, traceability, and accessibility while seamlessly integrating both digital and physical records.

1.4 Scope: The scope of "Asset Registration Using Blockchain" encompasses establishing a robust, decentralized system for registering and managing assets. It involves implementing blockchain technology to create a secure and universally accessible ledger. This scope includes protocols for asset management, ensuring data traceability, and enabling users to interact within this secure network. The system addresses challenges in maintaining a singular truth, accommodating both digitized and physical asset records. Its reach extends across diverse industries seeking transparent, secure, and reliable asset registration mechanisms, leveraging the capabilities of blockchain.

1.5 Project Introduction: Blockchain serves as an emerging platform for decentralized applications and shared data storage among multiple parties throughout recorded transactions. Every transaction in the public ledger undergoes verification through consensus protocols involving a majority of system participants. New data emerges as blocks, which are created and encrypted using hashing algorithms. Once entered, information cannot be modified without consulting a legal administrator. Blockchain enables the creation of a ledger of events, transactions, and data through various IT processes with cryptographic guarantees. It is a distributed digital ledger that is open, shared, transparent, and highly secured, ensuring immutable and verifiable transactions. As the name suggests, blockchain allows a block of data to grow as new blocks are appended, each containing transaction information stored in a specially designed data structure.

CHAPTER-2

LITERATURE SURVEY

[1] In June 2018, N.S. Tinu published a survey on blockchain technology, exploring its taxonomy, consensus algorithms, and applications. Initially developed to support the digital currency Bitcoin, blockchain serves as a versatile platform, providing benefits beyond encrypted currency. It can be likened to an operating system running applications like Bitcoin and Ethereum. In essence, blockchain is a distributed digital ledger, ensuring openness, sharing, and high security, with records being both immutable and verifiable. The term "ledger" draws parallels to its use in banking operations, signifying its role as the official record-keeper for verifying data transactions over time.

[2] J. Michael Graglia, in a 2018 study, delves into the digitalization of land records using blockchain technology. The research focuses on case studies from Honduras and Georgia, examining the socio-political and technical factors influencing the Information Systems (IS) readiness of public organizations. The success or failure of blockchain initiatives in land registry is influenced by factors such as political support, comprehensive national registries, and modern digital land records. The cases highlight the need for process redesign and technological readiness, with success contingent on the interaction between socio-political and technological factors.

[3] Yashwanth Madak, in a 2018 journal, discusses a blockchain application using Hyperledger Fabric with Angular Frontend. The goal is to create a secure and traceable database activated through protocols. Users manage assets within this digital network, relying on a singular truth supported by blockchain assets. These assets collaborate to form blocks where users register and establish their

base, ensuring a completely secure database.

[4] In 2019, Madakam and S. presented a study on land purchasing using blockchain, aiming to strengthen the land registration process and reduce property fraud. The blockchain-based Land Registration system becomes a distributed system recording all transactions during land purchases. Blockchain, known for its innovation in tracking transactions, employs cryptographic hash methods to protect data blocks. The study also highlights blockchain's applications in various fields, emphasizing its role in ensuring the integrity and security of transactions.

These summaries provide insights into different aspects of blockchain technology, including its origins, applications in land registration, and the utilization of specific frameworks like Hyperledger Fabric.

CHAPTER-3

RESEARCH GAPS OF EXISTING METHODS

3.1 Existing Method:

The utilization of blockchain technology presents a promising avenue for establishing a decentralized, secure, and tamper-proof asset registration system. The versatility of blockchain applications in asset registration showcases its potential to significantly improve transparency, security, efficiency, and mitigate fraud risks in various industries.

3.2 Challenges:

1. **Incomplete Records:** A notable challenge arises when blockchain encounters incomplete or undigitized records, particularly when dealing with physical assets or legacy systems. This issue can potentially hinder the accurate representation of all assets within the blockchain network. The incapacity to fully integrate non-digital information poses a hurdle to achieving a comprehensive and reliable asset registration system.
2. **Limited Interoperability:** Achieving interoperability between different blockchain platforms and systems proves to be a challenging aspect. The complexity of ensuring seamless communication and data sharing among entities using distinct blockchain networks may result in fragmented systems. This limitation hampers the potential for a

unified and cohesive asset registration infrastructure, particularly in scenarios where diverse blockchain solutions are in use.

3. Scalability Concerns: The scalability of blockchain becomes a concern when faced with an increasing volume of transactions and users. As the demand for transaction processing grows, limitations may arise, leading to slower processing times and potentially elevated costs. This scalability challenge is particularly pertinent in scenarios where the asset registration system needs to accommodate a large number of transactions simultaneously.

4. Security Risks: Despite its decentralized nature and robust cryptographic protocols, blockchain is not impervious to security risks. Various vulnerabilities, such as smart contract bugs, the potential for 51% attacks on smaller blockchain networks, and human errors in coding, can expose the system to security breaches. While blockchain is generally acknowledged for its security measures, these risks highlight the need for ongoing vigilance and advancements in security protocols to maintain the integrity of the asset registration system.

In summary, while the utilization of blockchain for asset registration offers numerous advantages, addressing challenges related to incomplete records, limited interoperability, scalability concerns, and security risks is crucial for ensuring the effectiveness and reliability of the proposed decentralized asset registration system. Overcoming these challenges will contribute to the successful implementation of blockchain technology in revolutionizing asset registration processes.

CHAPTER-4

PROPOSED MOTHODOLOGY

The proposed methodology for asset registration through blockchain revolves around the creation of a decentralized ledger that ensures openness and accessibility for all participants within the network. This decentralized framework is intricately designed to serve as a secure and transparent management system capable of handling various types of assets. At the core of this approach is the integration of smart contracts, a feature that facilitates automated asset transfers, thereby eliminating the need for intermediaries and significantly improving overall operational efficiency.

The decentralized ledger ensures equal access for all participants, fostering a transparent and tamper-resistant environment within the asset registration system. The incorporation of smart contracts not only automates asset transfers but also enhances transaction speed and accuracy. This innovative use of blockchain technology seeks to revolutionize traditional asset registration processes.

A fundamental advantage of this proposed system lies in the permanence of records stored in the blockchain. The immutability of the distributed ledger guarantees data integrity, as once a record is added to the blockchain, any attempt to alter or delete it becomes practically impossible. This feature ensures a reliable and auditable history of asset registrations over time, enhancing the credibility of the entire system.

To fortify the security of sensitive information, cryptographic measures are implemented within the blockchain system. These measures encompass encryption techniques and digital signatures, providing robust safeguards for the confidentiality and integrity of data stored in the decentralized ledger. This comprehensive security infrastructure instills confidence in users and

stakeholders, assuring them of the protection of their valuable assets and associated information.

The primary objective of this proposed asset registration system is to modernize and streamline the registration process. By capitalizing on the inherent advantages of blockchain technology—such as decentralization, transparency, and security—the system aims to establish a tamper-resistant, easily accessible, and efficient mechanism for registering a diverse range of assets. This approach has the potential to revolutionize asset management across various industries, laying the groundwork for trust and efficiency in the registration and transfer of assets.

Continuing with the detailed description:

Furthermore, the proposed asset registration system aims to address the challenges inherent in traditional registration processes by leveraging blockchain technology. Traditional systems often rely on centralized databases, which may be susceptible to manipulation, unauthorized access, or data breaches. In contrast, the decentralized nature of the blockchain ensures that no single entity has control over the entire system, reducing the risk of fraud and enhancing overall security.

The integration of smart contracts is a pivotal feature that not only automates asset transfers but also introduces programmable logic into the registration process. This enables the execution of predefined actions based on specific conditions, streamlining complex workflows and minimizing the potential for errors. Smart contracts add a layer of efficiency and autonomy, allowing for a seamless and trustless asset transfer mechanism.

Moreover, the proposed system is designed to cater to a diverse range of assets, accommodating both digital and physical forms. This versatility is crucial in today's dynamic business environment, where assets can take various forms, from digital tokens to real estate properties. By providing a unified platform for

registering and managing these diverse assets, the system promotes interoperability and simplifies the overall asset management landscape.

The transparency offered by the decentralized ledger extends beyond mere accessibility. It engenders trust among participants, as they can independently verify the ownership and history of any registered asset. This transparency not only reduces the likelihood of disputes but also contributes to a more accountable and auditable asset registration ecosystem.

Additionally, the proposed system introduces a level of disintermediation, eliminating the need for intermediaries such as banks or brokers in asset transactions. This not only reduces transaction costs but also accelerates the process, making asset registration and transfer more agile and cost-effective. The removal of intermediaries also aligns with the decentralized ethos of blockchain technology, empowering users and reducing dependencies on centralized authorities. In summary, the proposed asset registration system, harnessing the power of blockchain technology, transcends traditional norms, providing a holistic remedy to the inherent challenges of conventional registration methods. By embracing decentralization, ensuring transparency, incorporating smart contracts, and implementing robust cryptographic security measures, the envisioned system aims to establish a robust, streamlined, and globally accessible platform for the registration and oversight of a wide array of assets.

CHAPTER-5

OBJECTIVES

The core objective behind the implementation of "Asset Registration Using Blockchain" is to leverage the inherent capabilities of blockchain's distributed ledger technology. The primary goal is to establish a robust and efficient system that ensures security, transparency, and immutability in the registration and management of diverse assets. The focus of this initiative is to streamline the process of asset ownership, emphasizing precision and minimizing discrepancies through the adoption of decentralized verification methods.

By harnessing blockchain protocols, the overarching objective is to create a platform that offers universal accessibility. This approach aims to instill confidence in users, allowing them to register assets securely and with trust in the system's integrity. The initiative addresses the challenges associated with maintaining a singular truth in asset records, a critical aspect in asset management. The distributed ledger ensures that all participants within the network have access to the same set of information, promoting transparency and eliminating the risk of conflicting records.

This initiative acknowledges the importance of accommodating scenarios where asset data may exist in nondigital formats. By doing so, it strives to bridge the gap between physical and digital records, offering a comprehensive solution for asset registration. The ultimate aspiration is to seamlessly integrate both types of records within the asset registration framework, providing a unified and standardized approach.

The primary benefits of this approach include boosting reliability, traceability, and accessibility in the asset registration process. The immutability of records

within the blockchain ensures a trustworthy and tamper-resistant history of asset ownership. The decentralized nature of the system minimizes the reliance on central authorities, reducing the risk of manipulation or errors in the registration process.

In summary, the "Asset Registration Using Blockchain" initiative aims to revolutionize the asset management landscape by creating a secure, transparent, and universally accessible platform. By seamlessly integrating both physical and digital records, the system strives to address challenges associated with maintaining accuracy and consistency in asset ownership information. The ultimate goal is to establish a trustworthy and efficient framework that enhances reliability and traceability in the registration and management of assets.

CHAPTER-6

SYSTEM DESIGN & IMPLEMENTATION

6. REQUIREMENT ANALYSIS

6.1 Input Design Introduction:

In the realm of information systems, input serves as the raw data processed to generate meaningful output. Input design involves careful consideration of devices such as PCs, MICR, OMR, etc. The quality of system input is pivotal in determining the overall output quality. Well-crafted input forms and screens exhibit certain characteristics:

- Specific purpose effectiveness, encompassing storage, recording, and retrieval of information.
- Ensuring accurate and proper completion.
- Ease of filling and straightforwardness.
- Focused on user attention, consistency, and simplicity.

Objectives for Input Design:

The key objectives of input design include:

- Designing data entry and input procedures.

- Reducing input volume.
- Designing source documents for data capture or exploring alternative data capture methods.
- Crafting input data records, data entry screens, user interface screens, etc.
- Implementing validation checks and developing effective input controls.

Output Design:

The design of output is a critical task within any system. During output design, developers identify the necessary output types and consider essential output controls and prototype report layouts. Objectives of output design encompass:

- Developing an output design serving its intended purpose while eliminating unwanted output.

Creating an output design that aligns with end user requirements.

- Delivering an appropriate quantity of output.
- Formatting the output correctly and directing it to the right person.
- Ensuring timely availability of output to facilitate informed decision-

making.

Implementation:

6.2 Modules:

Buyer:

- Register: Buyers undergo a registration process on the page.
- Login: Buyers access the system by logging in after completing registration.
- View Selling Land Details: Post-login, buyers can peruse the details of land listings.
- Communicate with the Seller: Buyers can easily engage in communication with sellers after reviewing the details.

Seller:

- Signup: Sellers need to sign up on the page.
- Sign In: After signing up, sellers log in to the system.
- Add Land Details: Sellers can input and add details of the land they wish to sell.

- View Buyer Requests: Once land details are added, sellers can view requests from potential buyers.
- Communicate with the Buyer: Sellers can initiate communication with interested buyers.

This system design aims to facilitate seamless interaction between buyers and sellers, ensuring a user-friendly experience throughout the registration, login, and communication processes.

6.3 Functional and non-functional requirements

Requirement analysis is a crucial step in evaluating the success of a system or software project. Requirements are typically categorized into two main types: Functional and non-functional requirements.

Functional Requirements: These are the specific functionalities that end users explicitly request as essential features the system must provide. All these functionalities are integral parts of the contract and must be incorporated into the system. Functional requirements are expressed in terms of the input required by the system, the operations it should perform, and the expected output. Unlike non-functional requirements, these are directly observable in the final product, representing the user's explicit demands.

Functional Requirements Examples:

1. User Authentication: Ensuring the system authenticates users during login processes.
2. Cyber-attack Response: Implementing a system shutdown in the event of a cyber-attack.
3. Registration Verification: Sending a verification email to users upon their first-time registration in a software system.

Non-functional Requirements Explanation:

Non-functional requirements, also known as quality constraints or non-behavioral requirements, define the criteria that the system must meet according to the project contract. These aspects vary in priority across projects and encompass factors such as portability, security, maintainability, reliability, scalability, performance, reusability, and flexibility.

Non-functional Requirements Examples:

1. Email Latency: Ensuring emails are sent with a latency not exceeding 12 hours from a specific activity.
2. Request Processing Time: Ensuring each request is processed within a maximum time frame of 10 seconds.

3. Website Load Time: Specifying that the site should load within 3 seconds when the number of simultaneous users exceeds 10,000.

Software and Hardware Requirements:

System Specifications:

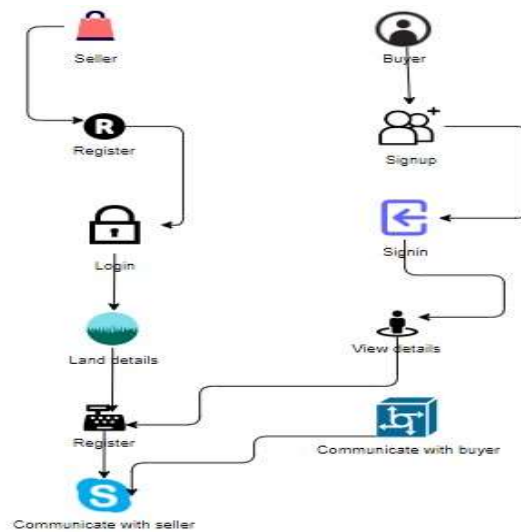
Hardware Specifications:

- Processor: I5/Intel Processor
- RAM: Minimum 8GB
- Hard Disk: 128 GB

Software Specifications:

- Operating System: Windows 10
- Front End: React JS
- Backend Technology: Python
- IDE: Visual Studio Code

Architecture:



SYSTEM DESIGN

Input Design in System Design:

In information systems, input constitutes the raw data processed to generate output. During input design, developers must consider various input devices such as PCs, MICR, OMR, etc. The quality of system input directly influences the quality of output. Well-designed input forms and screens possess certain key properties:

- Specific Purpose: Effectively serves purposes like storing, recording, and retrieving information.
- Accuracy: Ensures proper completion with accuracy.
- Simplicity: Easy to fill and straightforward.

- User-Focused: Focuses on user attention, consistency, and simplicity.

Input design achieves these objectives through a sound understanding of basic design principles related to required inputs and end users' responses to different elements of forms and screens.

Objectives for Input Design:

- Designing data entry and input procedures.
- Reducing input volume.
- Designing source documents or alternative data capture methods.
- Designing input data records, screens, and user interfaces.
- Implementing validation checks and effective input controls.

Output Design in System Design:

Output design is a critical task in system development. During this phase, developers identify the required output types, consider necessary output controls, and create prototype report layouts.

Objectives of Output Design:

- Developing output designs that serve intended purposes and avoid producing unnecessary output.
- Meeting end user requirements through appropriate output designs.
- Delivering the correct quantity of output.

- Formatting output appropriately and directing it to the right recipient.
- Making output available in a timely manner for informed decision-making.

UML Diagrams, Specifically Use Case Diagram:

A Use Case Diagram in the Unified Modeling Language (UML) is a behavioral diagram derived from Use-case analysis. It provides a graphical overview of a system's functionality, presenting actors, their goals (use cases), and dependencies between use cases.

Key Points about Use Case Diagram:

- Illustrates what system functions are performed for each actor.
- Depicts the roles of actors within the system.
- Aids in understanding the interactions and relationships between different functionalities within the system.

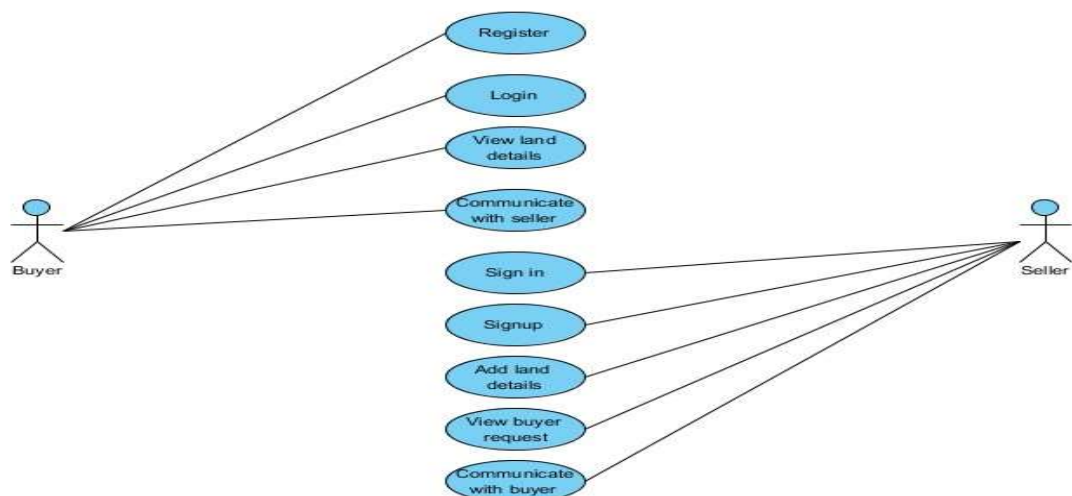


Fig-1.1 Use case Diagram

CLASS DIAGRAM

Class Diagram in Software Engineering and UML:

In the realm of software engineering, a class diagram within the Unified Modeling Language (UML) serves as a static structure diagram. Its primary purpose is to depict the system's structure by illustrating classes, their attributes, operations (or methods), and the relationships existing among these classes. Essentially, it provides insights into how information is organized within different classes of the system.

Key Aspects of Class Diagrams:

- **Depiction of System Structure:** Describes the organization and composition of a system through classes.
- **Class Components:** Highlights attributes, operations (methods), and other pertinent characteristics of each class.
- **Relationships Representation:** Illustrates the connections and associations between different classes in the system.

In simpler terms, a class diagram offers a visual representation of the building blocks of a software system, showcasing the classes, their properties, and the ways they are interconnected. It serves as a valuable tool for understanding the structural aspects of software design.

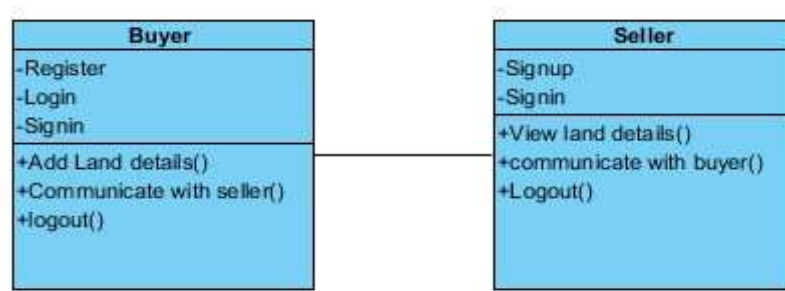


Fig-1.2 Class Diagram

SEQUENCE DIAGRAM

Sequence Diagram in UML:

In Unified Modeling Language (UML), a sequence diagram is a type of interaction diagram designed to illustrate the order and manner in which processes interact. It effectively portrays the flow of operations between different elements of a system.

Key Characteristics of Sequence Diagrams:

- Interaction Depiction: Offers a visual representation of how processes within a system interact with one another.
- Order of Operations: Illustrates the sequential order in which various processes or components operate.
- Derived from Message Sequence Chart: It is a conceptualization derived from Message Sequence Charts, providing a graphical representation of system dynamics.
- Alternative Names: Sequence diagrams are alternately referred to as event

diagrams, event scenarios, and timing diagrams.

In simpler terms, a sequence diagram in UML serves as a dynamic visual tool, capturing the flow and order of operations between different elements or processes within a system. It aids in understanding the chronological aspects of how components interact and collaborate.

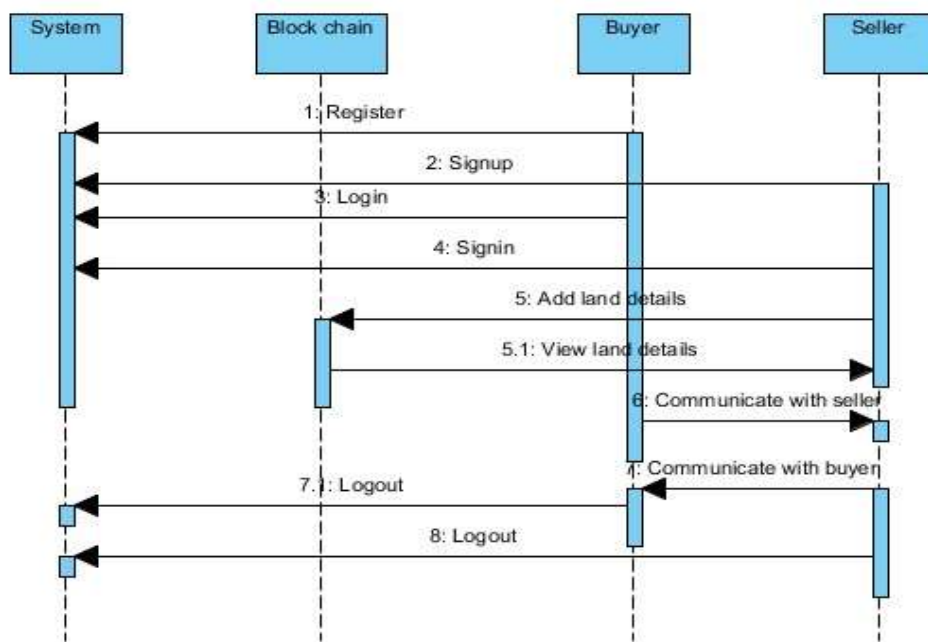


Fig-1.3 Sequence Diagram

COLLABORATION DIAGRAM:

Collaboration Diagrams in Object-Oriented Modeling:

Collaboration diagrams in object-oriented modeling employ a numbering system to represent the sequence of method calls, as demonstrated below. Each number corresponds to the order in which methods are executed, offering a clear depiction of the method call sequence. Using the example of an order management system for illustration, it's important to note that while collaboration diagrams share similarities with sequence diagrams in terms of

method calls, a significant distinction lies in their focus.

Key Characteristics of Collaboration Diagrams:

- Method Call Sequence: Utilizes a numbering technique to convey the sequential order of method calls.
- Object Organization: Uniquely emphasizes the organization and relationships among objects within the system.
- Comparison with Sequence Diagrams: In contrast to sequence diagrams, collaboration diagrams go beyond illustrating method sequences; they explicitly showcase the structural arrangement of objects.

In summary, collaboration diagrams serve as dynamic visual tools that not only illustrate the chronological flow of method calls but also provide valuable insights into the organizational structure of objects within the system. This dual focus enhances their utility in comprehending both temporal and structural aspects of object interactions.

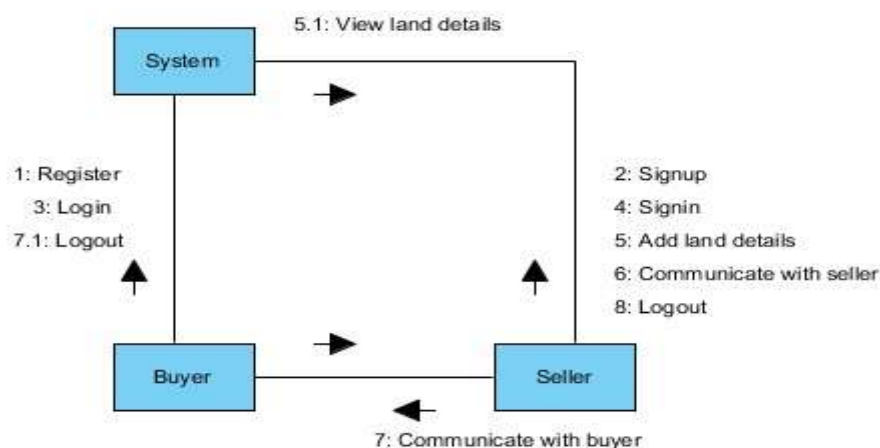


Fig-1.4 Collaboration Diagram

DEPLOYMENT DIAGRAM

Deployment Diagrams in System Modeling:

A deployment diagram serves as a visual representation of the deployment perspective within a system, closely linked to the component diagram. In essence, it illustrates how components are physically deployed in the system. The key elements of a deployment diagram are nodes, representing the actual hardware entities used for deploying the application.

Key Aspects of Deployment Diagrams:

- Deployment View: Provides a visual portrayal of how a system is physically deployed.
- Relationship with Component Diagrams: Depicts the deployment of components, establishing a connection with the component diagram.
- Node Representation: Nodes, integral to deployment diagrams, signify the physical hardware entities employed for application deployment.

In simpler terms, a deployment diagram offers a clear understanding of the physical arrangement of a system, showcasing the relationship between components and the hardware nodes supporting their deployment. This visual tool aids in comprehending the logistical aspects of how software components are distributed and interact within the underlying hardware infrastructure.

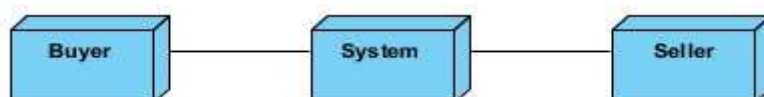


Fig-1.5 Deployment Diagram

ACTIVITY DIAGRAM:

Activity Diagrams in System Modeling:

Activity diagrams serve as visual representations illustrating stepwise workflows, actions, and activities within a system. They provide graphical depictions that support the representation of choices, iterations, and concurrent processes. Within the Unified Modeling Language (UML), activity diagrams find application in describing the sequential workflows of business and operational components within a system.

Key Features of Activity Diagrams:

- Workflow Representation: Graphically portrays the stepwise progression of activities and actions within a system.
- Support for Logic: Includes elements for representing choices, iterations, and concurrent processes in the workflow.
- UML Application: In the Unified Modeling Language, activity diagrams effectively describe business and operational workflows.
- Flow of Control: Offers a visual representation of the overall flow of control within the system.

In essence, an activity diagram provides a dynamic and intuitive way to convey the procedural aspects of a system, illustrating how various activities and actions unfold, making it a valuable tool for understanding the flow of control in a system's components.

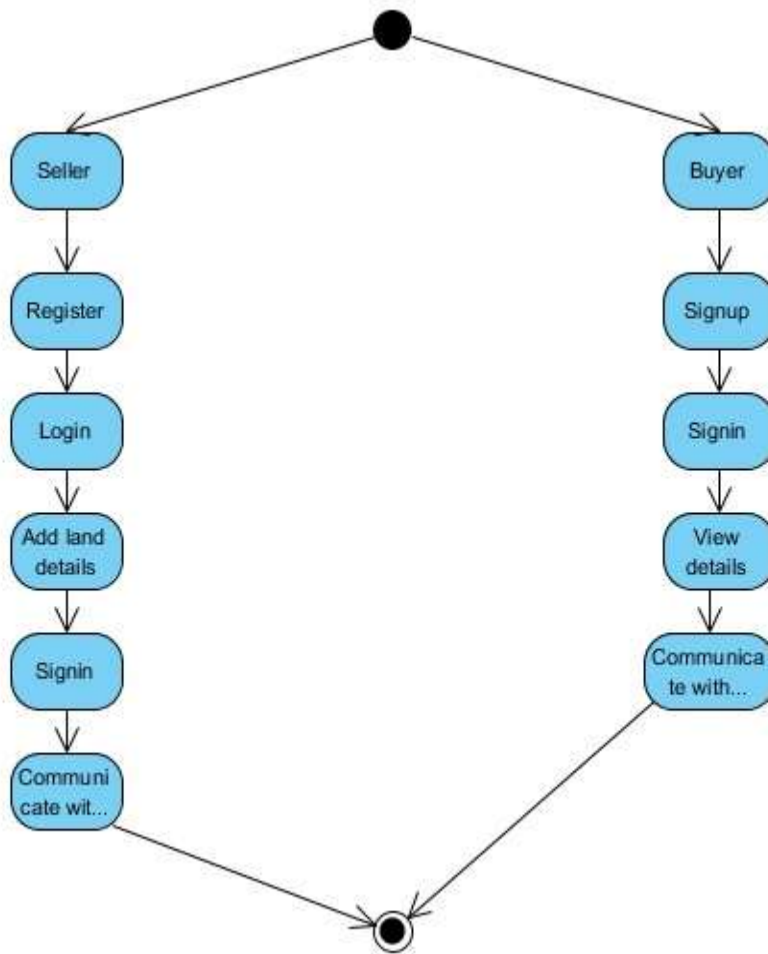


Fig-1.6 Activity Diagram

COMPONENT DIAGRAM:

****Component Diagrams in System Modeling:****

A component diagram, or UML component diagram, provides a visual representation that elucidates the structure and interconnection of physical components within a system. It serves as a tool to depict how various elements of the system are organized and connected. Typically, component diagrams play a crucial role in modeling implementation details, ensuring that the planned development comprehensively covers all the necessary

functions required by the system.

Key Aspects of Component Diagrams:

- ****Structural Organization:**** Illustrates how physical components are organized within the system.
- ****Interconnection Representation:**** Depicts the wiring and connections between different components.
- ****Implementation Modeling:**** Often utilized to model the specifics of system implementation.
- ****Validation Tool:**** Acts as a means to double-check that the planned development covers all the required functions of the system.

In simpler terms, a component diagram offers a visual blueprint of the physical composition of a system, aiding in understanding how components are structured, connected, and ensuring that the development plan adequately addresses the system's functional requirements.

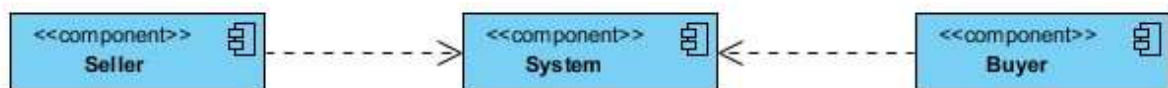


Fig-1.7 Component Diagram

ER DIAGRAM:

****Entity-Relationship Modeling and ER Diagrams:****

An Entity-Relationship (ER) model serves as a structural representation of a database, depicted through an Entity Relationship Diagram (ER Diagram).

This model acts as a design or blueprint for a database, laying the groundwork for its eventual implementation.

Key Points about Entity-Relationship Modeling:

- ****Components of ER Model:**** The primary elements of the ER model include entity sets and relationship sets.
- ****Entity Relationship Diagram (ER Diagram):**** A graphical representation showcasing the relationships among entity sets.
- ****Entity Set Definition:**** An entity set represents a grouping of similar entities, each with its own attributes.
- ****Attributes in DBMS:**** In the context of a Database Management System (DBMS), an entity corresponds to a table or an attribute of a table.

An ER diagram provides a comprehensive view of the logical structure of a database by illustrating relationships among tables and their attributes. Essentially, it captures the essence of how entities are related, offering a visual guide to the complete structure of the database.

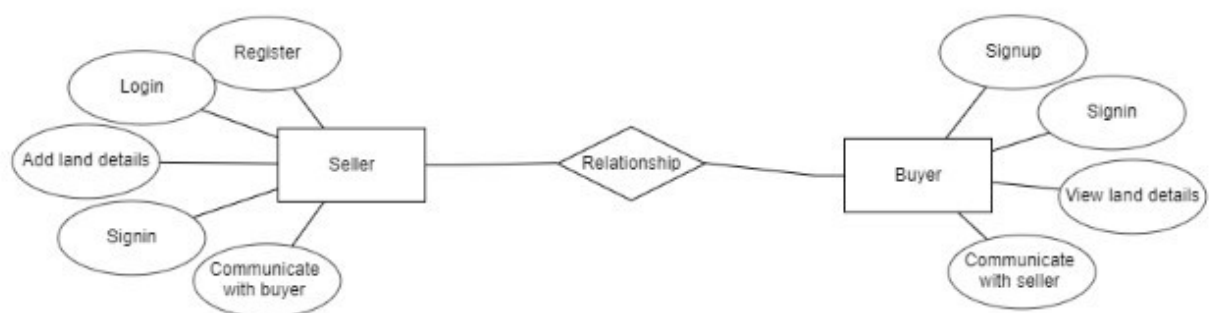


Fig-1.8 ER Diagram

DFD DIAGRAM:

****Data Flow Diagrams (DFD) for System Visualization:****

A Data Flow Diagram (DFD) serves as a conventional method for visually representing information flows within a system. When well-constructed, a DFD provides a concise and clear visualization of significant system requirements. DFDs can depict various systems, whether they operate manually, through automation, or via a combination of both approaches. These diagrams effectively illustrate how information flows into and out of the system, the points where information undergoes modifications, and the locations where information is stored.

Key Characteristics of Data Flow Diagrams:

- ****Information Flow Visualization:**** Offers a visual representation of how information moves within a system.
- ****System Requirements:**** A well-designed DFD encapsulates a substantial portion of the system's requirements.
- ****Manual or Automated Processes:**** Suitable for illustrating systems that operate manually, automatically, or through a blend of both methods.
- ****System Boundaries and Scope:**** Provides insights into the overall scope and boundaries of the system.

In essence, the primary purpose of a DFD is to present a holistic view of a system, showcasing how information circulates, changes, and is stored. It serves as a valuable tool for understanding the comprehensive dynamics of a system's information flow, making it a crucial element in system analysis and design.

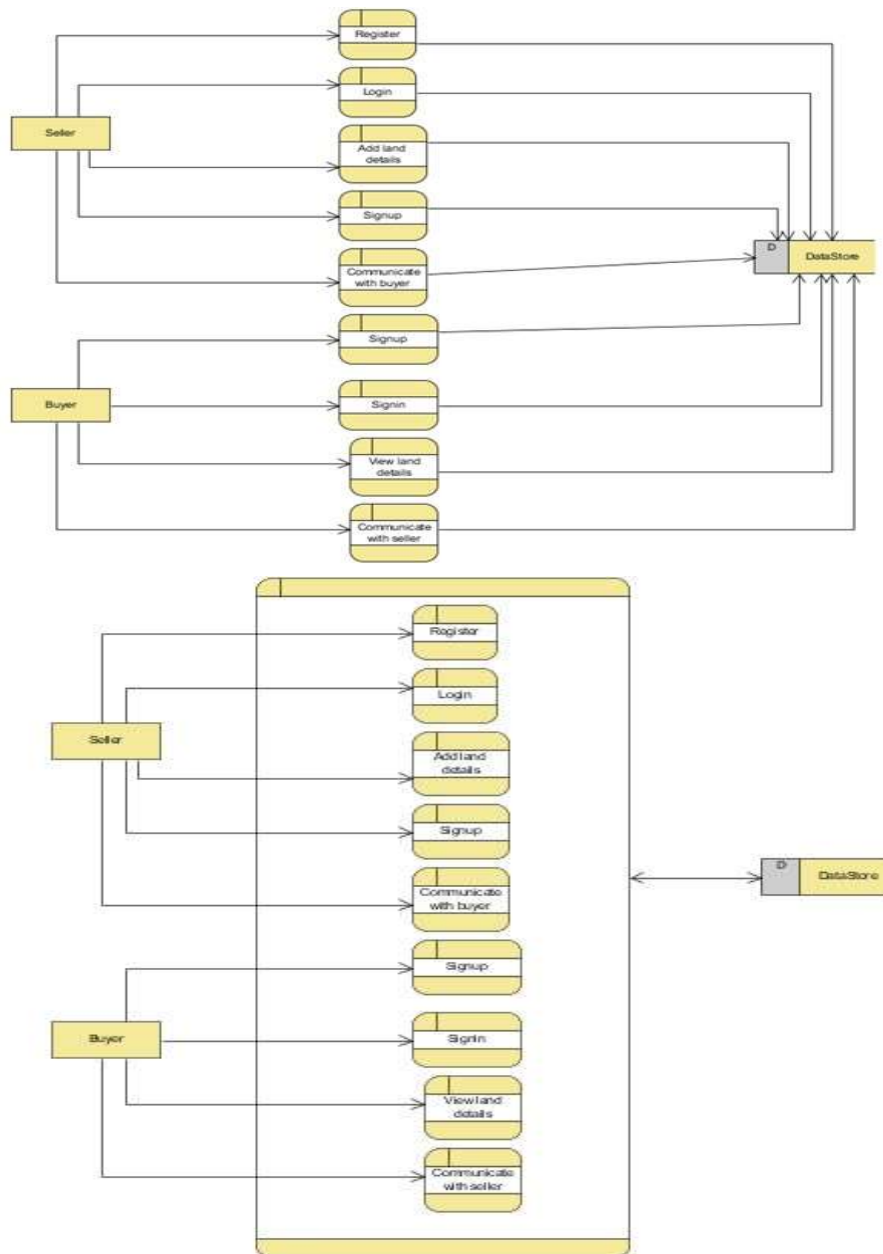
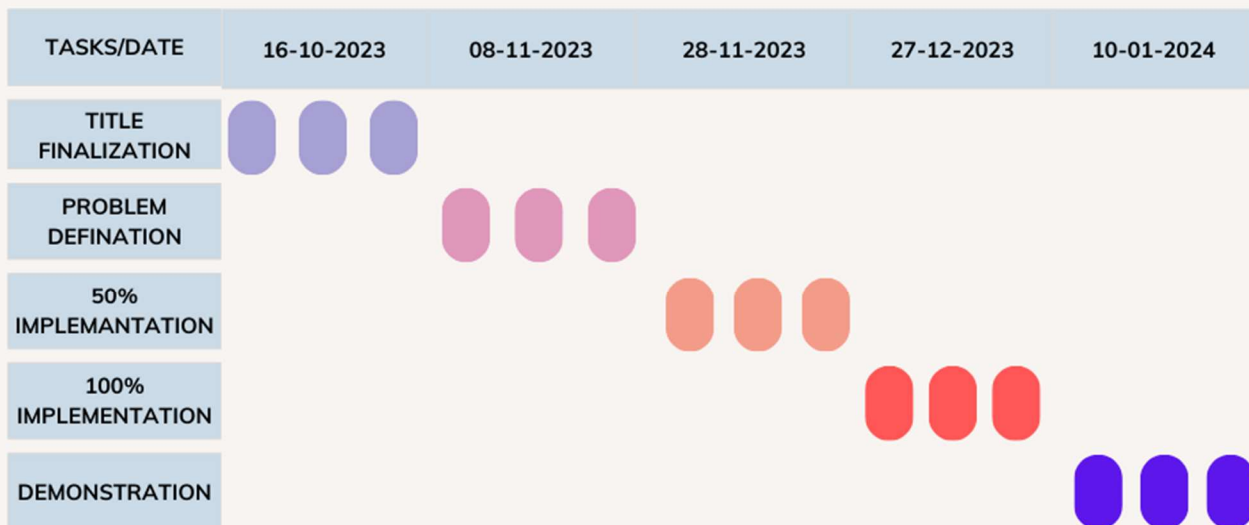


Fig-1.9 DFD Diagram

CHAPTER-7

TIMELINE FOR EXECUTION OF PROJECT (GANTT CHART)

GANTT CHART



CHAPTER-8

OUTCOMES

During the requirement analysis phase of a software project, the identification and delineation of both functional and non-functional prerequisites emerge as crucial determinants for the project's ultimate success. Functional requirements pertain to specific functionalities and operations that the software must perform. For instance, aspects like user authentication during login, the system's response to a potential cyber-attack, and the automatic dispatch of verification emails upon user registration fall within the domain of functional requirements. These specifications define the software's capabilities and functionalities in response to user interactions and system events. On the other hand, non-functional requirements impose constraints on the system's overall performance and behavior. These constraints go beyond specific functionalities and focus on parameters such as response time, scalability, and reliability. Examples of non-functional requirements include limiting email latency to 12 hours, ensuring that each request is processed within 10 seconds, and maintaining a site load time of 3 seconds even under the load of 10,000 simultaneous users. Non-functional requirements are essential for guaranteeing the system's efficiency, reliability, and user satisfaction. The subsequent section of the project provides detailed specifications for both hardware and software components. Hardware specifications include requirements such as an I5/Intel Processor, a minimum of 8GB RAM, and a 128 GB hard disk. These specifications outline the necessary computational resources to support the software's functionality. On the software side, specifications include Windows 10 as the operating system, React Js for the front end, Python for technology/backend, and VS Code as the integrated development environment. These choices not only define the technology stack but also set the environment in which the software will be developed and deployed. While specific architectural details are not provided in this overview,

the system design section underscores the importance of both input and output design. Input design focuses on enhancing system output by creating user-friendly and consistent input forms and screens. It prioritizes elements such as user attention, simplicity, and objectives like designing efficient data entry procedures and reducing input volume. On the other hand, output design is geared towards producing purposeful output that meets user requirements. It involves eliminating unnecessary information and ensuring the timely availability of output for effective decision-making. The subsequent exploration delves into various Unified Modeling Language (UML) diagrams as integral components of the system design. These diagrams include Use Case, Class, Sequence, Collaboration, Deployment, Activity, Component, Entity-Relationship (ER), and Data Flow Diagrams (DFD). Each diagram offers a unique perspective, collectively providing a holistic view of the software development process. These visual representations aid in understanding the system's structure, interactions, and data flow, contributing to effective communication among stakeholders and facilitating the overall design and development process.

In the realm of software project requirement analysis, the meticulous identification and delineation of both functional and non-functional prerequisites stand as pivotal determinants for the project's ultimate success. Functional requirements delve into the specific functionalities and operations that the software must perform, serving as the backbone for user interactions and system responses. Examples include user authentication, cyber-attack response mechanisms, and automated verification emails upon user registration. These specifications form the cornerstone of the software's capabilities, aligning it with the users' needs and system events.

Conversely, non-functional requirements transcend specific functionalities, focusing on overarching constraints that shape the system's overall performance and behavior. Parameters like response time, scalability, and reliability fall under

this category. Ensuring email latency within a 12-hour limit, processing each request within 10 seconds, and maintaining a site load time of 3 seconds under a load of 10,000 simultaneous users exemplify the critical role non-functional requirements play in guaranteeing the system's efficiency, reliability, and user satisfaction.

Moving into the subsequent phase of the project, detailed specifications for both hardware and software components are provided. Hardware specifications, including an I5/Intel Processor, a minimum of 8GB RAM, and a 128 GB hard disk, outline the essential computational resources necessary to support the software's functionality. On the software side, specifications encompass the choice of Windows 10 as the operating system, React Js for the front end, Python for technology/backend, and VS Code as the integrated development environment. These choices not only define the technology stack but also set the environment in which the software will be developed and deployed, crucial factors influencing the project's trajectory.

While specific architectural details may not be expounded upon in this overview, the system design section underscores the significance of both input and output design. Input design centers on optimizing system output by crafting user-friendly and consistent input forms and screens. Key considerations include user attention, simplicity, and objectives such as designing efficient data entry procedures and reducing input volume. In parallel, output design aims at generating purposeful output that meets user requirements. This involves eliminating extraneous information and ensuring timely availability of output for effective decision-making.

The subsequent exploration delves into various Unified Modeling Language (UML) diagrams, integral components of the system design. These diagrams, ranging from Use Case and Class to Sequence, Collaboration, Deployment, Activity, Component, Entity-Relationship (ER), and Data Flow Diagrams (DFD), offer unique perspectives, collectively software development process.

CHAPTER-9

RESULTS AND DISCUSSIONS

System requirement analysis is a critical step in evaluating the success of a software project, involving the categorization of requirements into functional and non-functional domains. Functional requirements, stemming from user demands, constitute the core features the system must provide, while non-functional requirements revolve around quality aspects like security, reliability, and performance. Examples of functional requirements include user authentication, system response to cyber-attacks, and the sending of verification emails for user registration. Non-functional requirements cover elements like email latency, request processing time, and site loading speed under specific user loads.

Transitioning to hardware and software specifications, the system demands an Intel i5 processor, a minimum of 8GB RAM, and a 128GB hard disk. Software prerequisites involve Windows 10 as the operating system, React JS for the front end, and Python for the technology/backend, integrated with VS Code.

System design encompasses both hardware and software considerations, detailing an Intel i5 processor, 8GB RAM, and a 128GB hard disk. Software specifications encompass Windows 10, React JS, Python, and VS Code.

Within system design, the focus extends to input and output design. Input design concentrates on refining raw data by considering input devices, aiming to create purposeful, user-friendly forms and screens. Objectives include designing data entry procedures, reducing input volume, and implementing validation checks. Output design becomes pivotal for generating necessary outputs, controls, and report layouts, with objectives centering on meeting user requirements and ensuring timely availability for decision-making.

The system design phase incorporates UML diagrams like use case, class, sequence, collaboration, deployment, activity, component, ER, and DFD

diagrams. Each diagram serves a unique purpose, presenting functionality, illustrating structure, showcasing interactions, depicting object organization, outlining deployment, detailing workflows, specifying physical components, and mapping database structures.

In summary, requirement analysis underscores the dichotomy of functional and non-functional requirements, hardware and software specifications delineate the technological foundation, and system design entails meticulous considerations of input and output design, complemented by a suite of UML diagrams capturing diverse aspects of the system architecture.

System requirement analysis is a pivotal phase in the software development lifecycle, serving as a compass for the project's success. The process involves the careful categorization of requirements into two main domains: functional and non-functional. Functional requirements form the core features dictated by user demands, delineating what the system must accomplish. Examples include user authentication, cyber-attack response mechanisms, and the automation of verification emails during user registration. On the other hand, non-functional requirements focus on quality aspects such as security, reliability, and performance, encompassing elements like email latency, request processing time, and site loading speed under specific user loads.

When transitioning to hardware and software specifications, a detailed understanding of the system's technological needs is imperative. For hardware, the system demands an Intel i5 processor, a minimum of 8GB RAM, and a 128GB hard disk. Software prerequisites include Windows 10 as the operating system, React JS for the front end, and Python as the backend technology, integrated seamlessly with Visual Studio Code (VS Code).

System design encapsulates both hardware and software considerations, providing a comprehensive blueprint for the system's architecture. Reiterating the hardware specifications of an Intel i5 processor, 8GB RAM, and a 128GB hard disk, the software specifications include Windows 10, React JS, Python,

and VS Code. This holistic approach ensures a robust foundation that aligns with the project's objectives.

Within the system design phase, a nuanced focus on input and output design becomes paramount. Input design involves refining raw data by considering various input devices and aims to create purposeful, user-friendly forms and screens. Objectives within this phase include designing efficient data entry procedures, reducing input volume, and implementing validation checks to enhance data accuracy. Meanwhile, output design plays a pivotal role in generating necessary outputs, controls, and report layouts. Objectives center around meeting user requirements and ensuring timely availability of information for effective decision-making.

The system design phase also incorporates Unified Modeling Language (UML) diagrams, serving as visual tools to capture different facets of the system architecture. These diagrams include use case diagrams for functionality representation, class diagrams for illustrating structure, sequence and collaboration diagrams for showcasing interactions, deployment diagrams for outlining system deployment, activity diagrams for detailing workflows, component diagrams for specifying physical components, and entity-relationship (ER) and data flow diagrams (DFD) for mapping database structures.

In summary, the requirement analysis phase underscores the dichotomy of functional and non-functional requirements, hardware and software specifications provide the technological foundation, and the system design phase entails meticulous considerations of input and output design. The inclusion of a diverse suite of UML diagrams enriches the documentation, offering a comprehensive representation of various aspects of the system architecture.

CHAPTER-10

CONCLUSION

In a holistic assessment, the application of blockchain technology in land registration emerges as a widely acclaimed and forward-looking strategy, celebrated for its intrinsic security features and immutable characteristics. By serving as a public digital ledger, the data entered becomes indelible, ensuring a transparent and tamper-resistant system that addresses the complexities inherent in customary land processes. This analysis emphasizes the adoption of blockchain technology, particularly leveraging the capabilities of Hyperledger in the system architecture, to address longstanding challenges in land registration processes in India.

The transformative potential of this innovative approach is particularly evident in its ability to revolutionize traditional practices associated with property transactions, such as buying and selling. By introducing a more pragmatic and reliable framework, blockchain technology lays the foundation for a modernized land registration system. The incorporation of smart contracts stands out as a key feature, significantly enhancing security measures and expediting transaction processes within the land registration system. This augmentation not only fortifies the system against potential vulnerabilities but also contributes to operational efficiency surpassing that of traditional counterparts.

Looking forward, there is a recognition of the need for continuous improvement and development to enhance the system's functionality. The exploration of seamless integration with relevant Application Programming Interfaces (APIs) is a proactive strategy with the potential to accelerate operational processes. This forward-looking approach aims to create a more streamlined, user-friendly, and efficient land registration system, ensuring adaptability to evolving

technological landscapes.

The envisaged benefits of this blockchain-driven approach extend beyond the immediate scope of land registration. Beyond providing a secure and transparent platform for property transactions, the systemic improvements have the potential for a positive and transformative impact on society in the long term. Improved operational efficiency, heightened security, and an enhanced user experience collectively position blockchain as a catalyst for positive change in the realm of land registration, promising a more equitable and technologically advanced future.

REFERENCES

The cited references cover various aspects of blockchain technology and its applications, particularly in the context of land registration:

1. In a survey by n.s.tinu (2018), the taxonomy, consensus algorithms, and applications of blockchain technology are explored.
2. J. Michael Graglia and Christopher Mellon (2018) discuss the intersection of blockchain and property, marking a significant point in its evolution.
3. Raquel Benbunan-Fich and Arturo Castellanos (2018) focus on the digitalization of land records, transitioning from traditional paper-based systems to blockchain.
4. IBM and State Street Corp.'s collaboration resulted in "Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains," detailing the features and functionalities of Hyperledger Fabric.
5. Miroslav Stefanovic, Djorde Przulj, and Darko Stefanovic (2018) examine the potential applications and limitations of blockchain in land administration.
6. Tsung-Ting Kuo, Hugo Zavaleta Rojas, and Lucila Ohno-Machado (2019) conduct a systematic review comparing various blockchain platforms, with a focus on healthcare examples.
7. Yacov Manevich, Artem Barger, and Yoav Toc explore service discovery for Hyperledger Fabric, shedding light on the operational aspects of this blockchain framework.

8. Hyperledger (2019) provides an introduction to Hyperledger Fabric, offering insights into its core functionalities and structure.

<https://hyperledgerfabric.readthedocs.io/en/release-1.4/whatis.html>

9. Another Hyperledger resource from 2017 delves into the transaction flow within Hyperledger Fabric, providing a detailed understanding of how transactions are processed.

In essence, these references collectively contribute to a comprehensive understanding of blockchain technology, its applications in land registration, and the specific features of Hyperledger Fabric.

<https://hyperledger-fabric.readthedocs.io/en/release-1.1/txflow.htm>

APPENDIX-A

PSUEDOCODE:

Here's a pseudocode representation of the provided Django web application code:

```
```plaintext
```

```
Import required modules and libraries
```

```
Define constant file names for HTML templates
```

```
Define functions for various views and actions in the application
```

```
function index(request):
```

```
 # Render index page
```

```
function login_user(request):
```

```
 if request.method == 'POST':
```

```
 # Extract user login details from the request
```

```
 # Authenticate user
```

```
 # Redirect to seller or buyer home page based on user type
```

```
 # Display warning message if login details are not valid
```

```
function register_user(request):
```

```
 if request.method == 'POST':
```

```
 # Extract user registration details from the request
```

```
 # Check password match and duplicate username
```

```
 # Create and save a new user
```

```
 # Display success or error messages
```

```
function logout_user(request):
 # Flush user session and redirect to the index page

function add_land(request):
 # Initialize land form
 if request.method == 'POST':
 # Extract land details and owner information from the request
 # Generate QR code and save land details
 # Display success message and redirect to the add land page

function sellerhome(request):
 # Redirect to the seller's home page

function view_land(request):
 # Render a page displaying land details

function request_land(request, id):
 if request.method == 'POST':
 # Extract land purchase request details from the request
 # Create a new land purchase request
 # Display success message and redirect to view land page

function view_land_requests(request):
 # Render a page displaying pending land purchase requests to a seller

function update_request(request, id):
 # Update the status of a land purchase request based on user action
 # Redirect to view request page
```

```
function buyer_landrequest(request):
 # Render a page displaying land purchase requests made by a buyer

function add_registration(request):
 # Retrieve accepted land purchase requests
 # Render a page for adding land registrations

function seller_lands(request):
 # Retrieve lands owned by the seller
 # Render a page displaying seller's lands

function register_land(request, id):
 # Register a land after a successful purchase
 # Generate a PDF receipt and save it
 # Update the status of the land purchase request
 # Redirect to add registration page

function delete_land(request, id):
 # Delete a land entry
 # Redirect to seller's lands page

function buyer_registrations(request):
 # Retrieve land registrations made by a buyer
 # Render a page displaying buyer's registrations
...
```

This pseudocode outlines the structure and flow of the Django web application without diving into specific implementation details. It provides a high-level

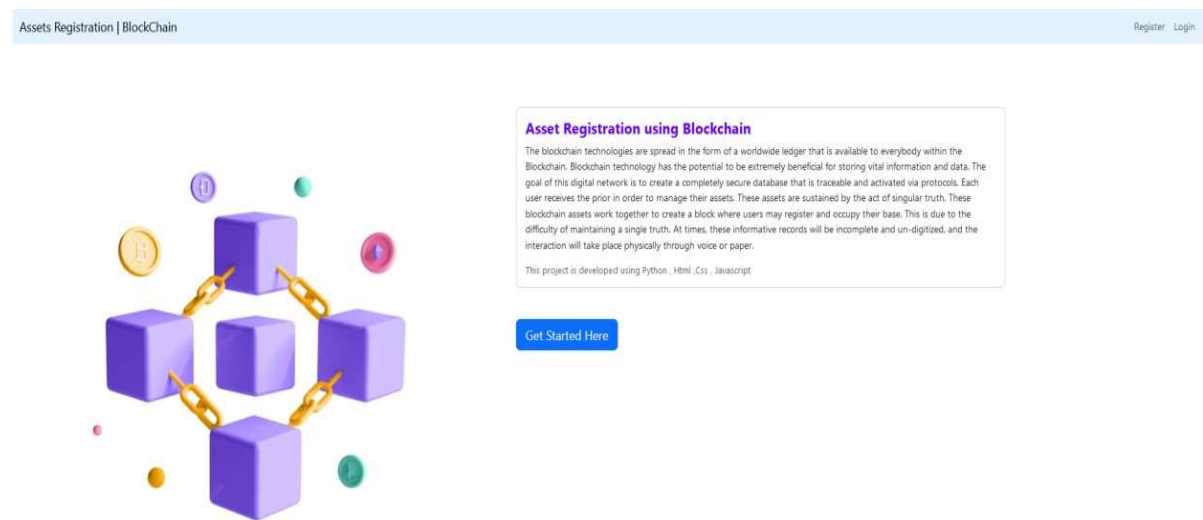
overview of the application's functionality.

## APPENDIX-B

### SCREENSHOTS

**Fig-1.9**

#### Home page:



#### Registration form:

### Create an account

Select User Type

Name  
Enter name

Username or Email  
Enter username or email

Password  
Enter password

Confirm Password  
Confirm password

Contact Number  
Contact Number

Address

Register [Already have an account?](#)

### Login Form:

### Login to your account

Username or Email  
Enter username

Password  
Enter password

Login [Don't have an account? Register](#)

### Sellers Home page:





## Add land Details:

### Add Land Details

Address\*

Land area\*


Landmark\*

Cost unit\*

Photo\*

## View land details:

## Land Details



Owner Name : sellertype  
 Owner Email : sellertype@gmail.com  
 Contact Number : 3322116655  
 Landmark : bangalore  
 Address : bangalore  
 Area : 470  
 Cost per Unit : 15

Make Request

Welcome!

Owner Name : nakku  
 Owner Email : nakku@gmail.com  
 Contact Number : 9685745896  
 Landmark : 20x40  
 Address : bangalore  
 Area : 20  
 Cost per Unit : 3

Make Request

### View land Details:

Land Registration System Using Blockchain Technology

[View Land Details](#) [My Registrations](#) [Logout](#)

## Land Details

Submit

Unit Cost : 3  
 Available Land : 20  
 Address : bangalore  
 Owner Name: nakku  
 Owner Email: nakku@gmail.com

Welcome!

### View request:

S.No	Buyer Name	Buyer Email	Buyer Phone	Land Address	Land Requested	Land Available	Buyer Price	Actions
2	preeti	preeti@gmail.com	8574589654	bangalore	2 sq.ft	20 sq.ft	6 /-	<a href="#">Accept</a> <a href="#">Reject</a>

## View land details:

## Land Details

# Welcome!

**Oner Name** : nakku  
**Owner Email** : nakku@gmail.com  
**Contact Number** : 9685745896  
**Landmark** : 20x40  
**Address** : bangalore  
**Area** : 18  
**Cost per Unit** : 3

[Delete](#)

## report for final year project plag 1

### ORIGINALITY REPORT

<b>22%</b>	<b>16%</b>	<b>11%</b>	<b>18%</b>
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

### PRIMARY SOURCES

<b>1</b>	<b>Submitted to Presidency University</b> Student Paper	<b>8%</b>
<b>2</b>	<b>Sai Apurva Gollapalli, Gayatri Krishnamoorthy, Neha Shivaji Jagtap, Rizwana Shaikh. "Land Registration System Using Block-chain", 2020 International Conference on Smart Innovations in Design, Environment, Management, Planning and Computing (ICSIDEMPC), 2020</b> Publication	<b>3%</b>
<b>3</b>	<b>Submitted to The University of the West of Scotland</b> Student Paper	<b>2%</b>
<b>4</b>	<b><a href="http://www.geeksforgeeks.org">www.geeksforgeeks.org</a></b> Internet Source	<b>1%</b>
<b>5</b>	<b><a href="http://docshare.tips">docshare.tips</a></b> Internet Source	<b>1%</b>
<b>6</b>	<b>Submitted to Sreenidhi International School</b> Student Paper	<b>1%</b>
<b>7</b>	<b><a href="http://www.ijcspub.org">www.ijcspub.org</a></b>	



### **The Project work carried out here is mapped to SDG-11 Sustainable Cities and Communities:**

Leveraging the power of blockchain, the project aims to revolutionize land registration processes, fostering transparent, efficient, and secure urban development. By deploying decentralized ledger technology, it enhances accessibility, ensuring a more inclusive approach to land management. The system's resilience and transparency contribute to building smart and resilient urban infrastructure, while also minimizing environmental impact through reduced reliance on paper-based systems. Empowering communities, the project encourages active participation in decision-making processes related to land use. Through this innovative application of blockchain, the project exemplifies a commitment to sustainable, inclusive, and technologically advanced urban development, addressing key challenges and fostering a resilient foundation for the cities and communities of the future.

## Certificate:

