Institute Vision, Mission Statements

Vision

- To emerge as a centre of technical excellence, transforming the engineering aspirants into dynamic and socially responsible technocrats

Mission

- **IM1:** Implementing effective strategies for imparting quality education in a conducive academic ambience to upgrade the intellectual and professional dimensions of the learner's personality
- **IM2:** Facilitating skill development and research to fulfill societal needs
- **IM3:** Inculcating moral principles, environmental consciousness and social responsibility among Students
- **IM4:** Grooming the students to handle the career challenges successfully

Department of Computer Science & Engineering

Vision

- To evolve as a leading computer science and engineering center producing competent technocrats to meet the demands of ever-changing industry and society

Mission

- Impart quality education through innovative teaching learning processes
- Motivate the learners to upgrade technical expertise by promoting learner centric activities.
- Inculcate values and interpersonal skills in the learners towards overall development.
- Upgrade knowledge in cutting edge technologies keeping pace with industrial standards through collaborations.

TRUSTWORTHY DATA SHARING VIA MULTI-ATTRIBUTE BASED AUTHENTICATION MECHANISM

A Project Report Submitted to

JAWAHARLAL NEHRU TECHNOLOGICAL UNIVERSITY, ANANTAPUR

Submitted By

P. KUNDANA
(212U1A0583)

P. SOWMYA
(21R61A0582)

SK. ROSHNI
(212U1A0599)

SK. SOFIA
(212U1A05A1)

Under the Esteemed Guidance of

Mrs. CH. VASAVI
Assistant Professor
Department of Computer Science & Engineering

Project report submitted in partial fulfilment of the Requirements
For the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

GEETHANJALI INSTITUTE OF SCIENCE AND TECHNOLOGY

A Unit of USHODAYA EDUCATIONAL SOCIETY

(Approved by AICTE, New Delhi & Permanently Affiliated to JNTUA, Anantapuram)

NAAC 'A' Grade Accredited Institution, An ISO 9001:2015 certified Institution

Recognized under Sec. 2(f) & 12(B) of UGC Act,1956

3rd Mile, Bombay Highway, Gangavaram (V), Kovur(M), SPSR Nellore (Dt), Andhra Pradesh, India- 524137 Ph. No. 08622-212769, E-Mail: geethanjali@gist.edu.in, Website: www.gist.edu.in

(2021 – 2025)

Website : www.gist.edu.in
Email: csehod@gist.edu.in

Ph : 0862-212781
Fax: 08622-212778

GEETHANJALI INSTITUTE OF SCIENCE AND TECHNOLOGY

A Unit of USHODAYA EDUCATIONAL SOCIETY

(Approved by AICTE, New Delhi & Permanently Affiliated to JNTUA, Anantapuramu)

NAAC 'A' Grade Accredited Institution, An ISO 9001:2015 certified Institution Recognized under Sec. 2(f) & 12(B) of UGC Act, 1956

3rd Mile, Bombay Highway, Gangavaram (V), Kovur(M), SPSR Nellore (Dt), Andhra Pradesh, India- 524137

BONAFIDE CERTIFICATE

This is to certify that the project work entitled "**TRUSTWORTHY DATA SHARING VIA MULTI-ATTRIBUTE BASED AUTHENTICATION MECHANISM"** is a bonafide record done by P. KUNDANA (212U1A0583), P. SOWMYA (21R61A0582), SK. ROSHNI (212U1A0599), SK. SOFIA (212U1A05A5) in the department of **Computer Science & Engineering, Geethanjali Institute of Science and Technology, Nellore** and is submitted to Jawaharlal Nehru Technological University, Anantapur in the partial fulfilment for the award of **B. Tech degree in Computer Science & Engineering.** This work has been carried out under my supervision.

Mrs. CH. VASAVI
Assistant Professor
Department of CSE
GIST, NELLORE

Dr. V. GAYATRI
HOD
Department of CSE
GIST, NELLORE

Submitted for the Viva-Voce Examination held on _____

Internal Examiner **External Examiner**

(2021 – 2025)

ACKNOWLEDGEMENTS

The satisfaction that accompanies the successful completion of the project would be incomplete without the people who made it possible. Their constant guidance and encouragement crowned the efforts with success.

We express our deepest sense of gratitude to **Sri. N. SUDHAKAR REDDY Garu, Secretary and Correspondent**, Geethanjali Institute of Science and Technology, Nellore and other members of Management for providing all the facilities needed for this work.

We owe our gratitude to **Dr. G. SUBBA RAO, Director,** Geethanjali Institute of Science and Technology, Nellore, for his consistent help and valuable suggestions.

We express our sincere thanks to the honourable Principal **Dr. SUNDEEP KUMAR K, Principal,** Geethanjali Institute of Science and Technology, Nellore, for his consistent help and valuable suggestions.

Our special thanks to**, Dr. V. GAYATRI, Head of the Department,** Department of Computer Science & Engineering, Geethanjali Institute of Science and Technology, Nellore, for her timely suggestions and help during the progress of project work in spite of her busy schedule.

It is indeed our proud privilege to express our deep sense of gratitude and indebtedness to our guide, **Mrs. CH. VASAVI, Assistant Professor,** Computer Science & Engineering**,** Geethanjali Institute of Science and Technology, Nellore, for her keen interest, critical, constructive and skillful guidance and constant encouragement throughout the course and for successful completion of project.

During the entire course of dissertation work, we received valuable academic inputs as well as moral support from other departments, general teaching and non-teaching faculty **GEETHANJALI INSTITUTE OF SCIENCE AND TECHNOLOGY**, Nellore. We were motivated by the uphold and moral encouragement given to us by our beloved parents. Finally, we wish to express our sincere thanks for all those who helped me directly or indirectly to complete the work.

PROJECT ASSOCIATES

P. KUNDANA (212U1A0583)
P. SOWMYA (21R61A0582)
SK. ROSHNI (212U1A0599)
SK. SOFIA (212U1A05A1)

TABLE OF CONTENTS

ABSTRACT

In the current digital era, governments are digitizing citizens' data, necessitating secure data sharing among various organizations and departments. To address the risks associated with centralized data storage and third-party authentication servers, which are vulnerable to attacks and misuse by malicious employees, a novel approach employing IPFS and Blockchain is proposed. This method eliminates reliance on third-party servers and ensures data security through decentralized storage and verification mechanisms. Data requesters generate personal private keys and encrypt multiple attributes before storing them in IPFS, with the resulting hash code saved in Blockchain for verification. Data owners use Blockchain to verify requester authenticity via the stored hash codes. The approach leverages Blockchain's inherent resistance to tampering, as any alteration in data results in hash code discrepancies, ensuring data integrity. The system utilizes ECC for key generation and encryption, with an alternative lightweight algorithm, CHACHA20, proposed to reduce computational costs. Additionally, the system integrates address proof document uploads for enhanced authentication. Smart contracts coded in Solidity manage the storage and retrieval of data in Blockchain. This solution facilitates secure, efficient government data sharing, mitigating risks associated with traditional centralized systems.

LIST OF FIGURES

LIST OF ABBREVATIONS

S.NO	ACRONYM	ABBREVATIONS
1	IPFS	InterPlanetary File System
2	ECC	Elliptic Curve Cryptography
3	IA	Identity Authentication
4	PKI	Public Key Infrastructure
5	UML	Unified Modeling Language
6	HCL	Hardware Compatibility List
7	QA	Quality Assurance

CHAPTER - 1
INTRODUCTION

1. INTRODUCTION

1.1 OVERVIEW

The advent of the information age has promoted the development of data assets. It is a common goal of all government departments to accelerate interdepartmental data sharing and realize a digital service-oriented smart government. However, the efficiency of government data sharing is significantly affected due to data storage in separate departments, low security of shared information storage, and uncertainty of the sharer's identity. Furthermore, it is difficult to assign responsibility for government data, making it challenging to share the data fully. Therefore, a secure and efficient government data-sharing solution is urgently needed.

Identity Authentication (IA) is the first step in securing government departments. Security data and privacy protection mechanisms are needed to restrict illegal access and use of valuable government data. However, traditional authentication mechanisms are generally based on trust provided by a third party, such as the well-known Public Key Infrastructure (PKI), cloud-driven trusted certificate authority, and current internet address allocation strategy.

A conventional IA model, such as the simplified version, typically comprises three parties: data requester, authorization center, and data owner. The data requester requests identity registration from the identity authorization center. When data sharing is required, the data requester initiates a request to the data owner, who verifies the identity of the former through the authorization center. Conventional static password-based authentication methods are cost-effective, and their implementation is simple and fast because they use only passwords for authentication and do not require other resources; however, they are vulnerable to network attacks and have low security.

Thus, multi-factor authentication needs to be considered. Because the traditional IA is controlled by the central server rather than the users, the latter are forced to trust the authority. A trusted authority controls the user's keys, which may lead to potential key escrow. Such a highly centralized feature can render the whole system unable to operate and cause a risk of a single point of failure in case the authority fails.

1.2 MOTIVATION

Traditional identity authentication systems are prone to security vulnerabilities, inefficiencies, and a lack of transparency. Governments require a robust method to verify citizens' identities while ensuring data security and privacy. The key motivations for this project include:

- Enhancing Security: Prevent unauthorized access and data tampering using blockchain's decentralized nature.
- Improving Privacy: Securely encrypt and store identity attributes in IPFS to protect sensitive personal data.
- Reducing Costs: Eliminate expensive centralized storage solutions by leveraging distributed ledger technology.
- Ensuring Transparency: Enable immutable and verifiable identity transactions without reliance on third-party entities.

1.3 PROBLEM STATEMENT

1. Centralized data storage and third-party authentication servers are vulnerable to attacks and misuse by malicious employees, leading to data breaches.
2. Governments are digitizing citizens' data, necessitating secure sharing among various organizations and departments, but existing systems lack robust security measures.
3. Citizens are affected by data breaches, risking their privacy and personal information being compromised or altered.
4. Data breaches lead to loss of trust in government systems, financial losses, and potential misuse of personal information by unauthorized entities.
5. To address this, we will implement a decentralized system using IPFS and Blockchain, ensuring secure, efficient, and tamper-proof data sharing with enhanced authentication methods.

1.4 OBJECTIVE

The primary goal of this project is to develop a decentralized and highly secure identity

authentication system that overcomes the drawbacks of traditional methods. The system aims to:

- Eliminate central authority dependence by leveraging blockchain technology.
- Ensure data privacy through secure encryption and distributed storage in IPFS.
- Reduce identity fraud risks by implementing transparent and tamper-proof verification mechanisms.
- Improve computational efficiency by using CHACHA20 encryption for faster processing.
- Facilitate seamless authentication across various government sectors.

1.5 BLOCKCHAIN TECHNOLOGY IN IDENTITY AUTHENTICATION

Blockchain technology plays a crucial role in enhancing identity authentication by providing a secure, transparent, and tamper-proof system. It eliminates the need for centralized identity providers and reduces security risks associated with traditional authentication methods. The blockchain network consists of multiple nodes that validate and record transactions, ensuring data integrity and trust without requiring a single controlling entity. Immutability ensures that once identity data is recorded, it cannot be altered or deleted, making it highly secure. Decentralization prevents single points of failure, reducing the risk of cyberattacks. Smart contracts automate verification processes, allowing real-time identity authentication without manual intervention.

Transparency enables full auditability of authentication events, while permissioned access ensures that only authorized entities can view sensitive information. The authentication process involves encrypting user identity data, generating a cryptographic hash, and storing it on the blockchain. During verification, the system retrieves the stored hash and matches it with the user's credentials, ensuring authenticity. This decentralized trust model eliminates reliance on third-party identity providers and significantly enhances security.

1.6 INTERPLANETARY FILE SYSTEM (IPFS) FOR SECURE STORAGE

The InterPlanetary File System (IPFS) is a distributed file storage protocol that enhances security and efficiency in identity management. Instead of storing complete identity records on a centralized server, the proposed system encrypts and uploads identity attributes to

IPFS, generating a unique cryptographic hash for each file. This hash is then stored on the blockchain, ensuring verifiability without exposing sensitive details. By leveraging IPFS, the system achieves efficient storage by reducing dependency on large centralized databases, fast retrieval through distributed nodes, and enhanced security by preventing unauthorized data modifications.

The decentralized nature of IPFS ensures that no single entity controls identity storage, eliminating risks associated with data breaches. When a user registers, their encrypted identity data is stored on IPFS, and the corresponding hash is written to the blockchain. During authentication, the system retrieves the hash, fetches the data from IPFS, and verifies it against the user's credentials. This approach ensures both security and efficiency, making identity authentication seamless and tamper-proof.

1.7 CRYPTOGRAPHIC TECHNIQUES FOR SECURITY

To ensure secure identity authentication, the system integrates advanced cryptographic methods such as Elliptic Curve Cryptography (ECC) and the CHACHA20 encryption algorithm. ECC is a public-key cryptography technique that provides strong security with smaller key sizes, reducing computational overhead while maintaining high levels of encryption. It ensures faster processing, making authentication more efficient. CHACHA20 is a lightweight stream cipher that enhances encryption while reducing computational complexity, making it ideal for fast and secure authentication.

Unlike AES, which involves complex mathematical operations, CHACHA20 provides high-speed encryption and decryption with minimal processing power, ensuring optimal system performance. The authentication process involves encrypting user credentials using CHACHA20, securing digital signatures with ECC, and storing only hashed identity attributes on the blockchain. This end-to-end encryption ensures that sensitive data remains protected throughout storage and transmission. By integrating ECC and CHACHA20, the system achieves a balance between security and efficiency, making identity authentication highly resistant to cyber threats while maintaining fast processing speeds.

CHAPTER - 2
LITERATURE SURVEY

2. LITERATURE SURVEY

2.1 BlockSLAP: Blockchain-Based Secure and Lightweight Authentication Protocol for Smart Grid:

https://www.researchgate.net/publication/347948335_BlockSLAP_BlockchainBased_Secure_and_Lightweight_Authentication_Protocol_for_Smart_Grid

Due to intelligent electronic management, the smart grid has recently played a significant role in modern energy infrastructure. However, along with widespread deployment of the smart grid, many potential security threats (e.g., impersonation attack, replay attack, man-in-the-middle attack) rise to the surface. To defend against these possible attacks, numerous cryptography-based authentication schemes have been proposed for the smart grid. Most of the schemes investigate the secret key distribution problem, but the requirement of decentralized registration authority is neglected. In addition, over-complicated cryptographic primitives also strengthen the burden of authentication system. In contrast with previous researches, our proposed BlockSLAP utilizes cutting-edge blockchain technology as well as smart contract to decentralize the registration authority and reduce the interaction process to 2 steps. Moreover, our protocol is proved secure under computational hard assumption and informal security analysis. Finally, experimental results show that smart grid authentication performance in our protocol has been improved compared to the other existing ECC-related schemes.

2.2 A Lightweight Leakage-Resilient Identity-Based Mutual Authentication and Key Exchange Protocol for Resource-limited Devices:

https://www.researchgate.net/publication/352729296_A_Lightweight_Leakage-Resilient_Identity-Based_Mutual_Authentication_and_Key_Exchange_Protocol_for_Resource-limited_Devices

Identity (ID)-based mutual authentication and key exchange (IDMAKE) protocol for client-server environments is a critical cryptographic primitive. Recently, several leakage-resilient IDMAKE (LR-IDMAKE) protocols have been proposed to withstand a new type of attacks, named side-channel attacks. However, the existing LR-IDMAKE protocols have several drawbacks, especially, the performance problem for low-power clients. In this paper, a

lightweight LR-IDMAKE protocol well-suited for resource-limited devices is proposed. As compared with the previously proposed LR-IDMAKE protocols, our protocol has the following merits: (1) It is the first lightweight LR-IDMAKE protocol without on-line pairing operation for clients; (2) During life cycle of the proposed protocol, it is resilient to continuous key leakage and has totally unbounded leakage property; (3) Security analysis is formally made to demonstrate that it is strongly secure against adversaries in the continuous-leakage extended-Canetti–Krawczyk (CLeCK) model. Eventually, performance experiences on an IoT device with low-computing power, i.e., the Raspberry PI, are conducted to demonstrate that our protocol is well-suited for resource-limited devices.

2.3 Algorithmic Computability and Approximability of Capacity-Achieving Input Distributions:

https://arxiv.org/pdf/2202.12617

The capacity of a channel can usually be characterized as a maximization of certain entropic quantities. From a practical point of view it is of primary interest to not only compute the capacity value, but also to find the corresponding optimizer, i.e., the capacity-achieving input distribution. This paper addresses the general question of whether or not it is possible to find algorithms that can compute the optimal input distribution depending on the channel. For this purpose, the concept of Turing machines is used which provides the fundamental performance limits of digital computers and therewith fully specifies which tasks are algorithmically feasible in principle. It is shown for discrete memoryless channels that it is impossible to algorithmically compute the capacity-achieving input distribution, where the channel is given as an input to the algorithm (or Turing machine). Finally, it is further shown that it is even impossible to algorithmically approximate these input distributions.

2.4 A Survey of Authentication in Internet of Things-Enabled Healthcare Systems:

https://www.researchgate.net/publication/365710906_A_Survey_of_Authentication_in_Internet_of_Things-Enabled_Healthcare_Systems

The Internet of medical things (IoMT) provides an ecosystem in which to connect humans, devices, sensors, and systems and improve healthcare services through modern technologies.

The IoMT has been around for quite some time, and many architectures/systems have been proposed to exploit its true potential. Healthcare through the Internet of things (IoT) is envisioned to be efficient, accessible, and secure in all possible ways. Even though the personalized health service through IoT is not limited to time or location, many associated challenges have emerged at an exponential pace. With the rapid shift toward IoT-enabled healthcare systems, there is an extensive need to examine possible threats and propose countermeasures. Authentication is one of the key processes in a system's security, where an individual, device, or another system is validated for its identity. This survey explores authentication techniques proposed for IoT-enabled healthcare systems. The exploration of the literature is categorized with respect to the technology deployment region, as in cloud, fog, and edge. A taxonomy of attacks, comprehensive analysis, and comparison of existing authentication techniques opens up possible future directions and paves the road ahead.

2.5 Offline Access to a Vehicle via PKI-Based Authentication:

https://www.researchgate.net/publication/354110429_Offline_Access_to_a_Vehicle_via_PKI-Based_Authentication

Using modern methods to control vehicle access is becoming more common. At present, various approaches and ideas are emerging on how to ensure the access in use cases reflecting car rental services, car sharing, and fleet management, where the process of assigning car access to individual users is dynamic and yet must be secure. In this paper, we show that this challenge can be resolved by a combination of the PKI technology and an access management system. We implemented a vehicle key validation process into an embedded platform (ESP32) and measured the real-time parameters of this process to evaluate the user experience. Utilizing the SHA256-RSA cipher suite with the key length of 3072 bits, we measured the validation time of 46.6 ms. The results indicate that the user experience is not worsened by the entry delays arising from the limited computing power of embedded platforms, even when using key lengths that meet the 2020 NIST recommendations for systems to be deployed until 2030 and beyond.

CHAPTER - 3
SYSTEM ANALYSIS

3. SYSTEM ANALYSIS

3.1 EXISTING SYSTEM:

Existing systems address security concerns in data sharing and access control for unmanned aerial vehicles and interconnected environments by leveraging blockchain technology. A 5G data-sharing model employs smart contracts for intrusion avoidance and access control to deter malicious users. Secure and trusted access systems utilize blockchain combined with edge computing to ensure activity traceability, trusted authentication, fault tolerance, and resistance to attacks. In vehicle-to-grid environments, blockchain-based demand response management analyses privacy in information exchange, using miner nodes for intrusion avoidance, effectively reducing transmission delay. A key agreement protocol integrates elliptic curve cryptosystems with the Menezes-Qu-Vanstone mechanism for mutual authentication, enhancing resistance to denial of service and replay attacks while reducing communication costs. Distributed privacy protection schemes replace centralized key management centers to provide unlink ability and user anonymity through blind signatures, along with efficient pseudonym update or revocation functions using bloom filters for low-overhead revocation.

3.1.1 DISADVANTAGES OF EXISTING SYSTEM:

- Reliance on centralized key management increases vulnerability.
- Higher computational costs due to less efficient cryptographic methods.
- Limited tamper detection mechanisms.
- Potential for misuse by third-party authentication servers.
- Generic solutions may not fully address specific needs of government data sharing.

3.2 PROPOSED SYSTEM:

The proposed system leverages IPFS and Blockchain to enhance the security of government data sharing. Data requesters generate personal private keys and encrypt attributes, storing the encrypted data in IPFS, which returns a hash code saved in Blockchain for verification. Data owners verify requesters through these hash codes stored in Blockchain. This decentralized approach eliminates the need for third-party authentication servers, reducing vulnerability to attacks and misuse. Blockchain ensures data integrity, as any tampering results in mismatched

hash codes, making unauthorized alterations detectable. To optimize performance, the system uses ECC for key generation and encryption, with CHACHA20 as a lightweight alternative to reduce computational costs. The inclusion of address proof document uploads further strengthens authentication. Smart contracts in Solidity manage data storage and retrieval on Blockchain, ensuring secure and efficient data sharing across government departments and organizations.

3.2.1 ADVANTAGES OF PROPOSED SYSTEM:

1. A decentralized approach eliminates third-party vulnerabilities.
2. Enhanced tamper detection with Blockchain-stored hash codes.
3. Optimized performance using ECC and lightweight CHACHA20 encryption.
4. Robust authentication with address proof document uploads.
5. Tailored specifically for secure government data sharing.

3.3 FUNCTIONAL REQUIREMENTS

1. Admin
2. Data Owner
3. Requester

3.4 NON-FUNCTIONAL REQUIREMENTS

Non-Functional Requirement (NFR) specifies the quality attribute of a software system. They judge the software system based on Responsiveness, Usability, Security, Portability and other non-functional standards that are critical to the success of the software system. Example of nonfunctional requirement, *"how fast does the website load?"* Failing to meet non-functional requirements can result in systems that fail to satisfy user needs. Non- functional Requirements allows you to impose constraints or restrictions on the design of the system across the various agile backlogs. Example, the site should load in 3 seconds when the number of simultaneous users are greater than 10000. Description of non-functional requirements is just as critical as a functional requirement.

- Usability requirement
- Serviceability requirement

- Manageability requirement
- Recoverability requirement
- Security requirement
- Data Integrity requirement
- Capacity requirement
- Availability requirement
- Scalability requirement
- Interoperability requirement
- Reliability requirement
- Maintainability requirement
- Regulatory requirement
- Environmental requirement

3.5 FEASIBILITY STUDY

A feasibility study is a critical evaluation process that determines whether a project is viable, practical, and beneficial for an organization. It helps in identifying potential risks, analysing resource requirements, and ensuring that the proposed system aligns with business goals. The study involves a detailed examination of technical, economic, and social aspects to determine the likelihood of successful implementation.

The primary purpose of conducting a feasibility study is to assess whether the proposed blockchain-based identity authentication system can be effectively developed and integrated into real-world applications. It evaluates whether the available technology, financial resources,

and societal acceptance are sufficient for the project's execution. The study also helps in decision-making by providing insights into constraints, risks, and expected benefits.

By analysing feasibility, organizations can minimize project failure risks, optimize costs, and ensure smooth deployment. It serves as a foundation for project planning, helping stakeholders understand potential challenges and develop strategies for overcoming them.

3.5.1 Economic Feasibility

Economic feasibility evaluates the financial viability of the project by analysing its costs and benefits. It helps in determining whether the investment in the project is justified.

- **Cost of Hardware:** Implementing blockchain-based identity authentication requires reliable computational resources, especially for cryptographic operations and decentralized storage. However, utilizing cloud-based or distributed infrastructure can optimize costs.
- **Software Costs:** Open-source blockchain frameworks and cryptographic libraries reduce software licensing expenses. However, continuous updates and security maintenance may incur additional costs.
- **Deployment Costs:** Integrating the system with government or organizational databases may require development efforts, infrastructure upgrades, and ongoing maintenance.

3.5.2 Technical Feasibility

Technical feasibility assesses whether the project can be implemented with the available technology and infrastructure. It evaluates the hardware, software, and other technical requirements of the system.

- **Data Storage and Security:** The project uses blockchain for immutable record-keeping and IPFS for decentralized storage, requiring efficient data retrieval mechanisms and cryptographic security.
- **Computational Power:** Blockchain transactions, cryptographic operations, and identity verification processes demand computational efficiency. The optimized smart contracts and consensus mechanisms ensure smooth operation.
- **System Integration:** The system must be compatible with existing authentication mechanisms and databases, requiring seamless API integration and support for multi-platform accessibility.

3.5.3 Social Feasibility

Social feasibility examines the impact of the project on users, organizations, and society, ensuring ethical and practical acceptance.

- **Privacy and User Trust:** The system enhances user privacy by encrypting identity attributes and eliminating reliance on centralized authorities, increasing public trust.
- **Regulatory Compliance:** Blockchain-based identity verification must comply with data protection laws and government regulations to ensure legal acceptance.
- **Adoption Challenges:** Educating users and institutions about decentralized identity authentication is essential for widespread adoption and successful implementation.

CHAPTER - 4
METHODOLOGY

4.1 SYSTEM ARCHITECTURE:

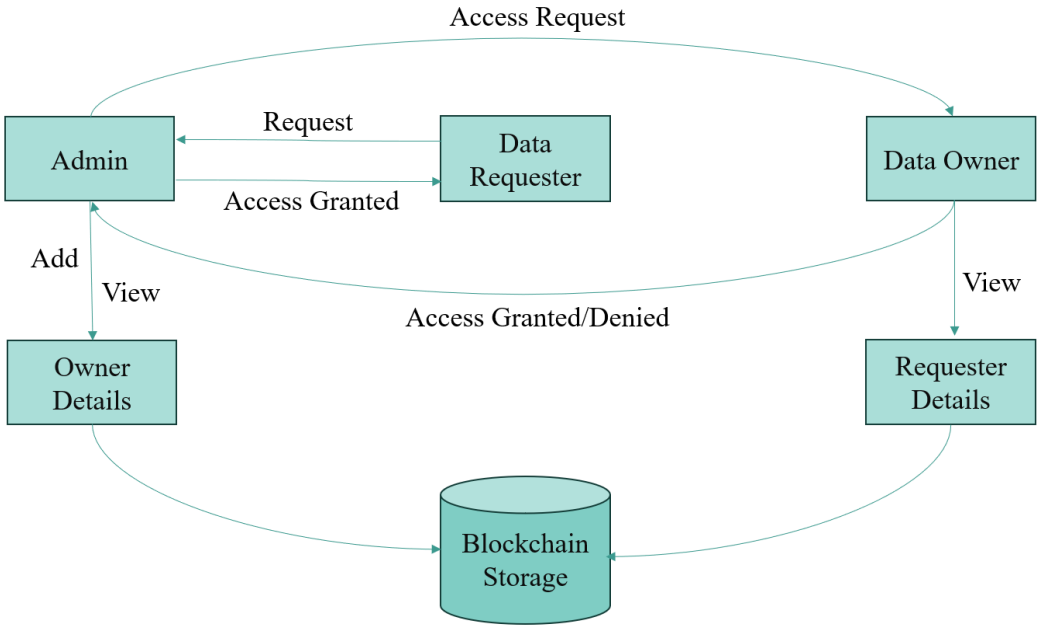


Fig. 4.1: System architecture

The system architecture for blockchain-based identity authentication is designed to ensure security, decentralization, and efficiency. It consists of multiple key components that work together to provide a seamless and tamper-proof identity verification process.

1. Admin Module: The admin is responsible for managing identity owners and requesters. They register new users, assign permissions, and oversee the overall system functionality.

2. Identity Owner Module: Users upload and secure their identity-related information. Instead of storing their sensitive data on a centralized database, the system encrypts the data and stores it on IPFS while keeping a record of its location on the blockchain.

3. Data Requester Module: Organizations or individuals who need to verify someone's identity submit authentication requests. They interact with the blockchain through smart contracts to ensure secure and verifiable access.

4. Blockchain Layer: This layer ensures data immutability and transparency by recording transactions, access requests, and verification results securely.

5. IPFS Storage: Unlike centralized storage systems, IPFS distributes data across multiple nodes, making it highly secure and resistant to cyberattacks.

6. Smart Contracts: These self-executing programs automate identity verification, eliminating the need for intermediaries and reducing the risk of fraud.

By combining these components, the system ensures privacy, security, and efficiency in identity management while removing reliance on centralized authorities.

4.2 Decentralized Identity Verification Using Blockchain

In traditional identity verification systems, centralized authorities control and manage identity records, creating vulnerabilities such as data breaches, unauthorized access, and single points of failure. This project leverages blockchain technology to decentralize identity verification, ensuring that identity records are secure, immutable, and resistant to tampering.

Blockchain provides a trust less environment, meaning that identity verification can be done without relying on a central authority. Each identity record is hashed, encrypted, and stored in a distributed ledger, ensuring that no single entity can alter or delete identity data. When a user registers their identity, the encrypted data is stored in the InterPlanetary File System (IPFS), and only a unique hash representing that data is recorded on the blockchain.

Core Features of Blockchain-Based Identity Verification:

1. Immutability:
    - Once an identity record is added to the blockchain, it cannot be altered or deleted.
    - This prevents fraud, identity manipulation, and unauthorized modifications.
2. Decentralization:
    - Unlike centralized systems, blockchain distributes identity records across multiple nodes, removing single points of failure.
    - This enhances reliability and security, ensuring that no central entity can control or exploit identity data.
3. Smart Contracts for Automation:
    - Smart contracts automate identity verification, reducing human intervention.

- When a requester submits an identity verification request, the smart contract validates the request, checks permissions, and grants or denies access automatically.

4. Privacy Protection & User Control:
    - Users own and control their identity attributes and decide who can access their data.
    - Blockchain ensures that only authorized entities can retrieve identity details, enhancing privacy and security.

When a government agency, bank, or employer requests to verify a user's identity, the blockchain validates the request through smart contracts. This ensures that only authorized requesters can access identity information, eliminating the risk of unauthorized access and data manipulation.

4.2.1 Working of Smart Contract-Based Authentication

Smart contracts play a critical role in automating the identity authentication process. Unlike traditional verification methods, where a centralized authority approves or denies access, smart contracts execute pre-defined conditions stored on the blockchain. This removes third-party intermediaries, improving efficiency, transparency, and security.

Authentication Workflow in Blockchain-Based Identity Verification:

1. Identity Registration:
    - A user submits their identity attributes, such as name, date of birth, and government-issued ID details.
    - These attributes are encrypted and stored on IPFS, ensuring decentralized and secure storage.
    - The system generates a unique hash for the stored identity data.
    - The Content Identifier is recorded on the blockchain, acting as a permanent reference for identity verification.
2. Access Request from Data Requesters:

- When a bank, employer, or government agency needs to verify the user's identity, they submit a verification request via the smart contract.
- The request includes authentication details and user consent to prevent unauthorized access.

3. Verification & Approval by Smart Contracts:
    - The smart contract checks the permissions associated with the identity data.
    - If the requester has valid authorization, the smart contract grants access and provides the IPFS hash for data retrieval.
    - If the requester is not authorized, access is denied, and an alert is generated to notify the identity owner.
4. Data Retrieval & Validation:
    - Once access is granted, the requester retrieves the encrypted identity data from IPFS using the provided Content Identifier.
    - The requester decrypts the data and verifies the identity information.

This decentralized model empowers users by giving them full control over their identity data, allowing them to grant or revoke access as needed. Moreover, organizations benefit from cost-efficient, fast, and fraud-resistant identity verification, which improves overall trust and security in digital interactions.

CHAPTER - 5
SYSTEM REQUIREMENTS AND SPECIFICATIONS

5. SYSTEM REQUIREMENTS AND SPECIFICATIONS

5.1 HARDWARE REQUIREMENTS

1) Processor: i5 and above

2) Ram: 8GB and above

3) Hard Disk: Minimum 25 GB free space in local drive

The most common set of requirements defined by any operating system or software application is the physical computer resources, also known as hardware, A hardware requirements list is often accompanied by a hardware compatibility list (HCL), especially in case of operating systems. An HCL lists tested, compatible, and sometimes incompatible hardware devices for a particular operating system or application. The following sub-sections discuss the various aspects of hardware requirements.

Architecture – All computer operating systems are designed for a particular computer architecture. Most software applications are limited to particular operating systems running on particular architectures. Although architecture-independent operating systems and applications exist, most need to be recompiled to run on a new architecture. See also a list of common operating systems and their supporting architectures.

Processing power – The power of the central processing unit (CPU) is a fundamental system requirement for any software. Most software running on x86 architecture define processing power as the model and the clock speed of the CPU. Many other features of a CPU that influence its speed and power, like bus speed, cache, and MIPS are often ignored. This definition of power is often erroneous, as AMD Athlon and Intel Pentium CPUs at similar clock speed often have different throughput speeds. Intel Pentium CPUs have enjoyed a considerable degree of popularity, and are often mentioned in this category.

Memory – All software, when run, resides in the random access memory (RAM) of a computer. Memory requirements are defined after considering demands of the application, operating system, supporting software and files, and other running processes. Optimal performance of other unrelated software running on a multi-tasking computer system is also considered when defining this requirement.

Secondary storage – Hard-disk requirements vary, depending on the size of software installation, temporary files created and maintained while installing or running the software, and possible use of swap space (if RAM is insufficient).

Display adapter – Software requiring a better than average computer graphics display, like graphics editors and high-end games, often define high-end display adapters in the system requirements.

Peripherals – Some software applications need to make extensive and/or special use of some peripherals, demanding the higher performance or functionality of such peripherals. Such peripherals include CD-ROM drives, keyboards, pointing devices, network devices, etc.

5.2 SOFTWARE REQUIREMENTS

1) Programming Languages: Python, JavaScript, Solidity
2) Web Framework: Django
3) Blockchain Tools: Ganache, MetaMask
4) Development Tools: Python IDLE, Node.js, Visual Studio Code

5.2.1 Programming Languages

The implementation of this blockchain-based identity authentication system requires multiple programming languages, each serving a specific function:

- Python: Used for backend development, handling server-side logic, and integrating blockchain functionalities.
- JavaScript: Essential for front-end development, ensuring a dynamic and user-friendly interface.
- Solidity: The primary language for writing smart contracts on the Ethereum blockchain, enabling automated and secure identity authentication.

Each of these languages plays a crucial role in ensuring an efficient, secure, and scalable software solution.

5.2.2 Web Framework

For efficient backend development, a powerful web framework is required:

- Django: A high-level Python web framework that simplifies backend development by providing:
    - Built-in security features to protect against vulnerabilities.
    - Scalability, making it suitable for handling large amounts of identity data.
    - Fast development capabilities, allowing for rapid prototyping and testing.

Django is chosen because of its robust security mechanisms, ease of integration with blockchain APIs, and efficient handling of authentication and database management.

5.2.3 Blockchain Tools

To interact with the Ethereum blockchain, multiple tools are essential for smart contract deployment, testing, and transaction execution:

- Ganache:
    - A personal Ethereum blockchain used for local testing of smart contracts.
    - Helps simulate real blockchain conditions without spending actual cryptocurrency.
- MetaMask:
    - A browser extension and crypto wallet that allows users to interact with blockchain applications securely.
    - Helps in signing transactions and authenticating users without a centralized authority.

These tools ensure secure deployment, testing, and execution of smart contracts, making blockchain-based identity verification reliable and accessible.

5.2.4 Development Tools

To enable efficient software development, debugging, and testing, several development environments and tools are used:

- Python IDLE:

  - Used for developing Python scripts that handle backend logic and integrate with the blockchain.

- Node.js:
  - Provides a runtime environment for JavaScript applications.
  - Essential for integrating blockchain functionalities, running Web3.js and Ethers.js libraries.
- Visual Studio Code (VS Code):
  - A powerful code editor that supports Python, JavaScript, and Solidity.
  - Ideal for full-stack development, offering features like syntax highlighting, debugging tools, and plugin support.

These tools enhance developer productivity, debugging efficiency, and seamless integration of blockchain and web technologies.

CHAPTER - 6
SYSTEM DESIGN

6. SYSTEM DESIGN

6.1 UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems. The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

Building Blocks of UML

UML has three fundamental building blocks:

1. Things in UML

"Things" refer to core elements that represent different parts of a system. They are categorized as:

- Structural Things: Represent system components (e.g., classes, objects, interfaces).
- Behavioral Things: Represent dynamic behavior (e.g., interactions, use cases).
- Grouping Things: Organize elements into logical units (e.g., packages).
- Annotational Things: Add notes and comments to explain different parts of the diagram.

2. Relationships in UML

Relationships define how different elements interact with each other in the system. UML includes:

- Dependency: One element relies on another for functionality.
- Association: Represents a connection between two elements.
- Generalization: Inheritance relationship between classes or objects.
- Realization: Implementation of an interface by a class.

3. Diagrams in UML

A UML diagram is a graphical representation of system components and interactions. The most commonly used UML diagrams include:

1. A Use-Case Diagram represents the interactions between users (actors) and the system. It helps in understanding the functional requirements by visually depicting how different users interact with various components of the system.
2. A Sequence Diagram shows the order of interactions between different objects in a system. It is particularly useful for modeling real-time and event-driven applications by representing the sequence of messages exchanged between objects over time.
3. A Class Diagram represents the static structure of a system using classes, attributes, and methods. It defines the relationships between different classes, making it essential for object-oriented design and implementation.

6.1.1 USE-CASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in

the system can be depicted. It helps in understanding user interactions, system boundaries, and functional requirements at a high level. Use case diagrams are essential in software development as they provide a clear visualization of the system's functionality. They help identify potential user roles and how they interact with the system. By mapping out interactions, they assist in defining system scope and dependencies. Additionally, they serve as a communication tool between developers, stakeholders, and business analysts to ensure a shared understanding of the system's requirements.

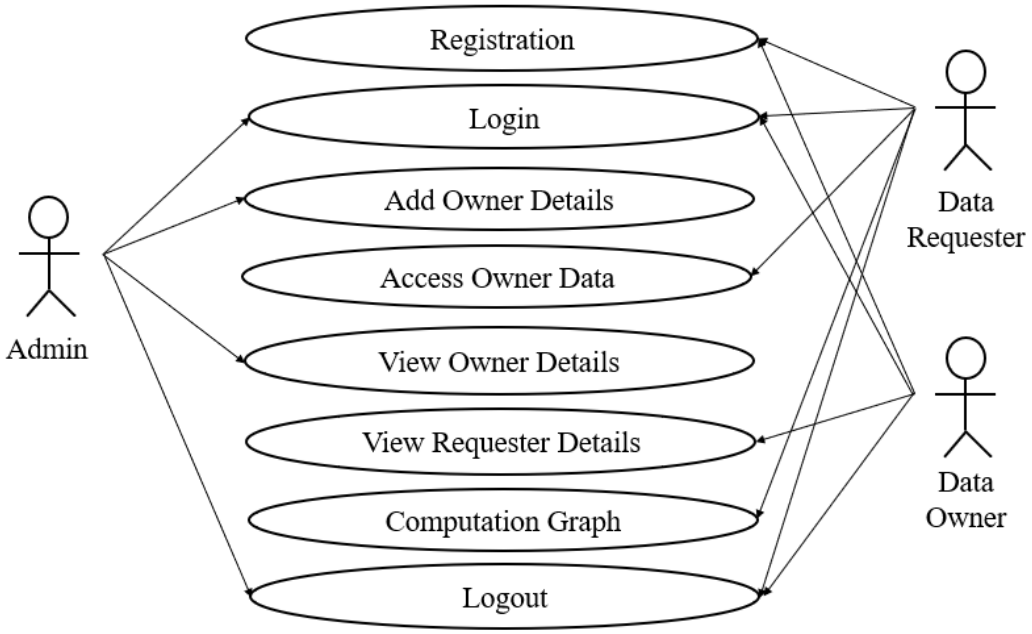


Fig.6.1.1 Use-Case diagram

6.1.2 SEQUENCE DIAGRAM

A sequence diagram represents the interaction between different objects in the system. The important aspect of a sequence diagram is that it is time-ordered. This means that the exact sequence of the interactions between the objects is represented step by step. Different objects in the sequence diagram interact with each other by passing "messages." These diagrams help visualize the dynamic behavior of a system, making it easier to understand complex processes. Sequence diagrams also assist in identifying potential bottlenecks, improving system efficiency, and validating system logic. They are widely used in software design to document workflows, user interactions, and system functionalities. Additionally, they aid in communication between developers, designers, and stakeholders to ensure a clear

understanding of system behavior. By illustrating object interactions over time, sequence diagrams provide valuable insights for debugging, testing, and system optimization.

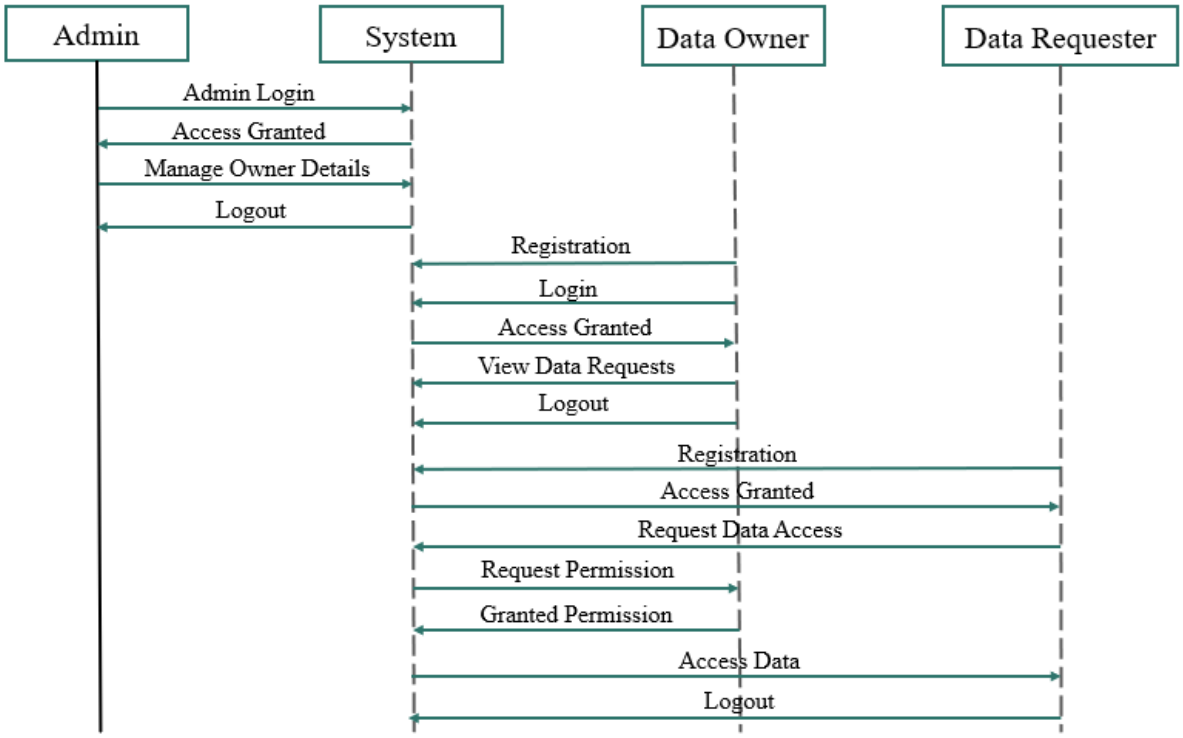


Fig.6.1.2 Sequence Diagram

6.1.3 CLASS DIAGRAM

The class diagram plays a crucial role in object-oriented design by defining the blueprint of the system. It helps developers understand how different classes interact and depend on each other, ensuring a well-structured architecture. By representing attributes and methods, class diagrams provide clarity on data encapsulation and inheritance, making it easier to maintain and scale the system. They also aid in database schema design, as the relationships between classes often correspond to table relationships in a database. The diagram serves as a bridge between conceptual system design and actual implementation by helping in code generation and system documentation. Additionally, it simplifies debugging and future modifications by clearly outlining the system's components. Since class diagrams offer a high-level overview of the system structure, they are widely used in both the analysis and design phases of software development. This makes them an essential tool for software engineers, architects, and developers working on complex applications.

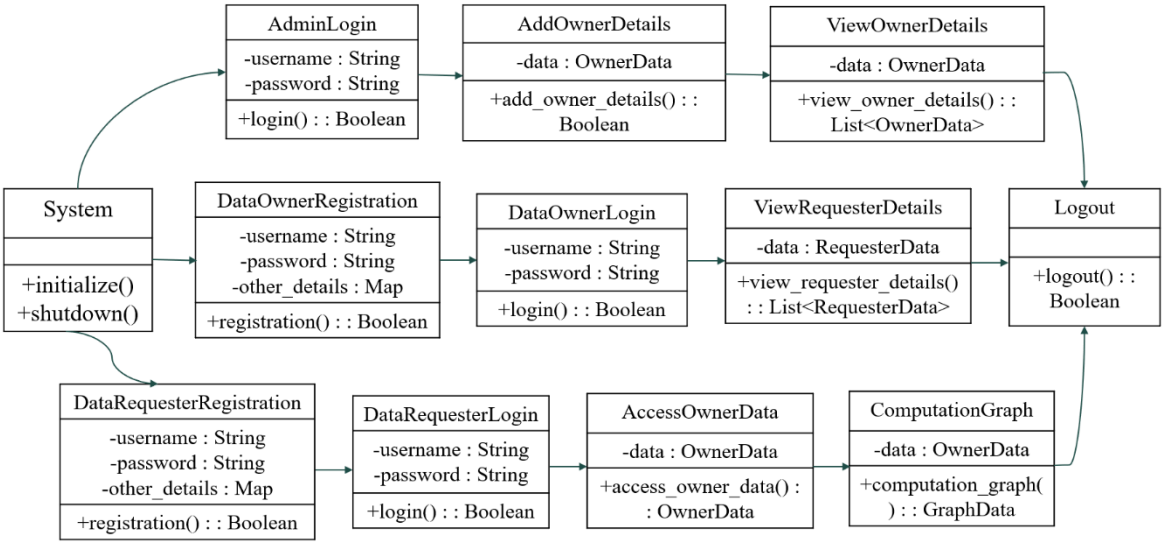


Fig.6.1.3 Class Diagram

CHAPTER - 7
IMPLEMENTATION

7. IMPLEMENTATION

Now-a-days all countries are digitizing their citizens data and different organizations or departments may require this data, for example bank or electricity departments may want to access this data to know about citizen capability of repayment. So, government will allow such different departments to access their citizen's data. In all existing applications all citizen's data will be maintained in a single centralized servers and then for DATA REQUESTER KEY authentication government will be using third party authentication servers.

Third party servers can be attack by attacker to steal all citizens private and public key to decrypt data and then all citizen's data privacy will be at risk. Sometime some malicious employees of third-party server managing employees can sale all citizen data to other companies or can alter citizen data and in such situations also citizen's privacy will be at risk.

To overcome from above issue author of this paper employing different technologies such as IPFS and Blockchain decentralized data storage and by using this technologies we are not required to depend on third party servers and data owner data will be secured.

In propose work author using following entities

1) Data Requester: data requester will request for private keys from authentication server and it just required to generate keys. Requester will generate personal private keys using private keys sent by authentication server. Data Requester will encrypt multiple attributes (requester name, contact no, department name) using personal keys and then saved data in IPFS system. IPFS will stored requester data and then returned hash code and this hash code will get saved in Blockchain to verify requester.

2) Data Owner: data owner will connect to Blockchain and verify requester and if requester hash code available in Blockchain then only data owner will give access to requester.

3) IPFS: Blockchain charge heavy gas values for storage so if we store all requester attributes in Blockchain then government has to pay huge charges so author of this paper saving Request attributes details in IPFS and the hash code returned by IPFS for those attributes will get saved in Blockchain.

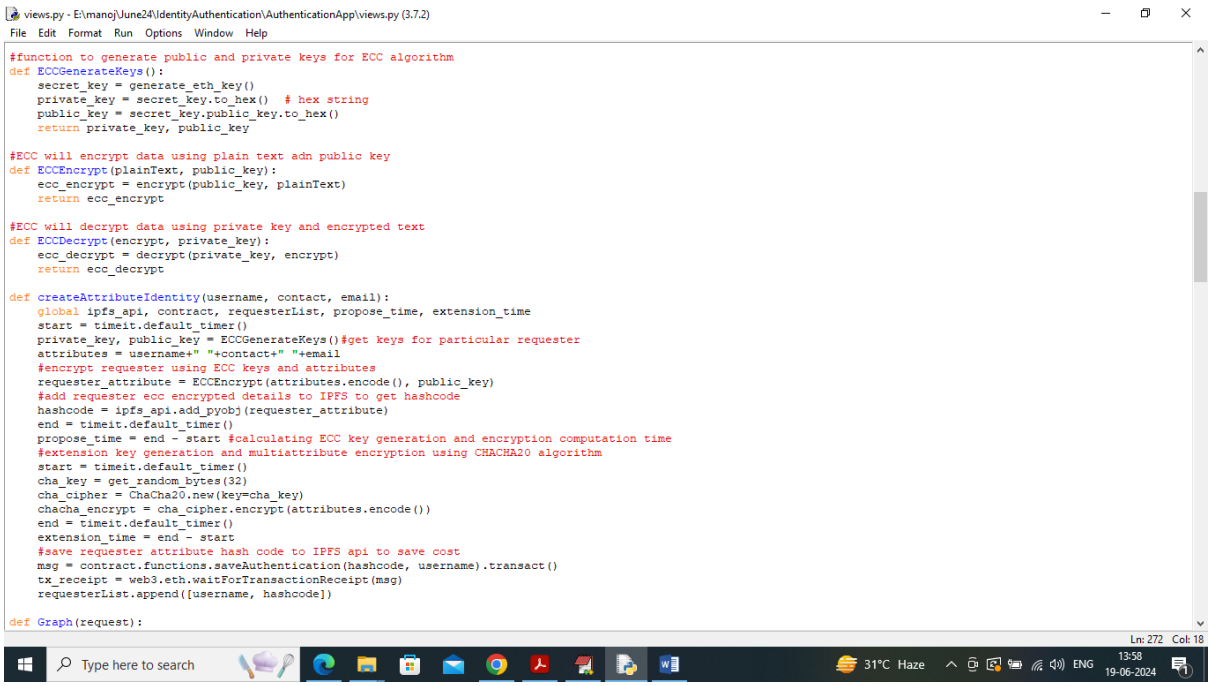
4) Blockchain: Blockchain smart contract just saved Requester hash code in Blockchain for future verification of Data Requester

To generate keys and for encryption author using ECC (Elliptic Curve Cryptography) and this algorithms key generation, initialization and encryption process we can read from base paper.

Extension 1: In this project author just using simple citizen details which will not sufficient to verify citizen authentication so as extension we are taking citizen details along with address proof document upload which can help in authenticating citizen as genuine or fake. This additional verification step strengthens identity authentication by providing verifiable proof, reducing the chances of identity fraud and unauthorized access.

Extension 2: In propose paper author using ECC algorithm which is very much heavy in computation and required heavy storage in IPFS and to reduce this computation cost we are employing another lightweight key generation and encryption algorithm called CHACHA20 as extension 2. This enhancement ensures faster encryption and decryption while maintaining high security, making the system more efficient and scalable.

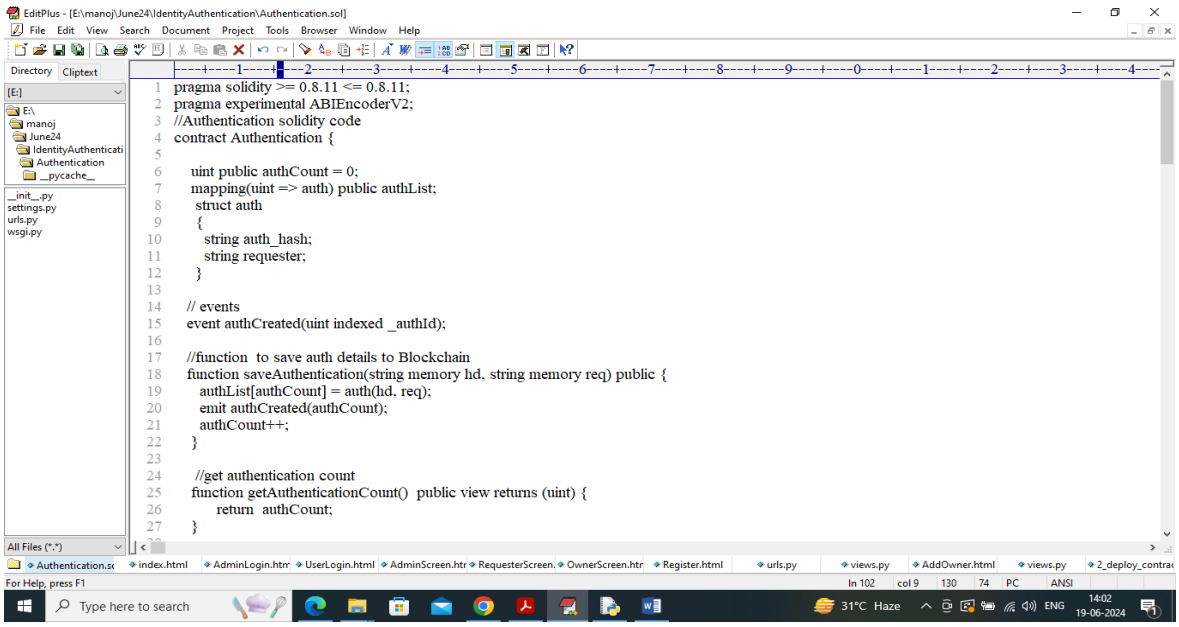
In below screen showing code for key generation and encryption using both ECC and CHACHA20.



In above screen read red colour comments to know about attributes encryption and key generation using ECC and extension algorithm.

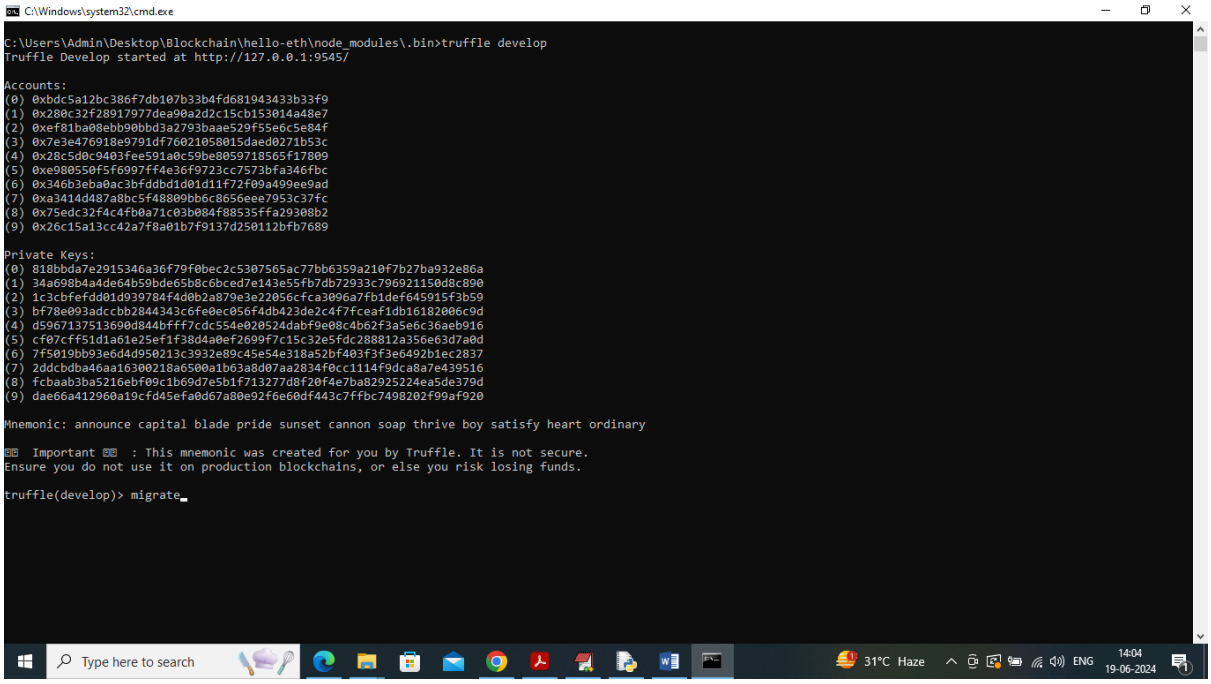
In propose paper main idea of using Blockchain is to avoid data tamper and leak without any attacks so Blockchain has inbuilt support for decentralized data storage (Blockchain record single data storage or transaction at multiple nodes) storage and verification. Blockchain store each record as block/transaction and associate each record with unique hash code and if any attacker or employee change data at any node then it will result into different hash code and verification get failed. So data in Blockchain cannot be alter in any manner. Additionally, blockchain ensures transparency by maintaining a tamper-proof ledger where every transaction is permanently recorded and cannot be erased. This enhances security, making it nearly impossible for unauthorized modifications to go unnoticed.

To store and retrieve record in Blockchain we need to utilize smart contracts code which can be designed using SOLIDITY programming and in below screen showing SMART Contract code to manage REQUESTER and Data Owner details.

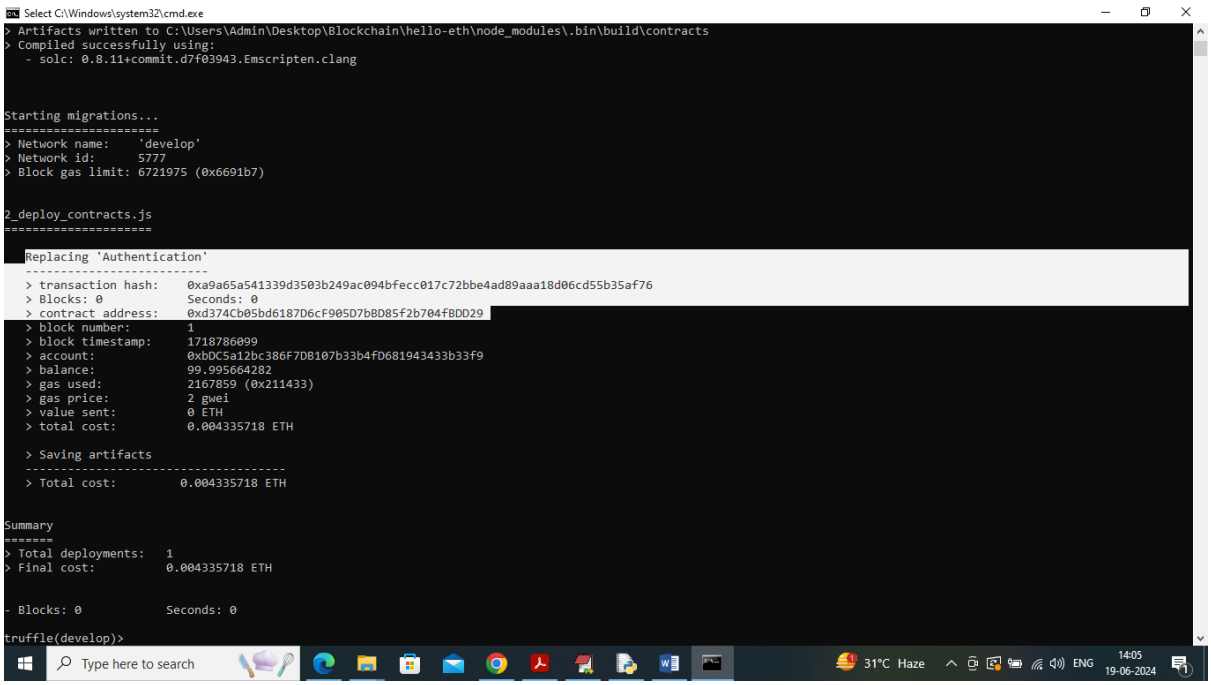


In above smart contract code defined functions to save and get data and now we need to deploy above contract in Blockchain ETHEREUM using below steps

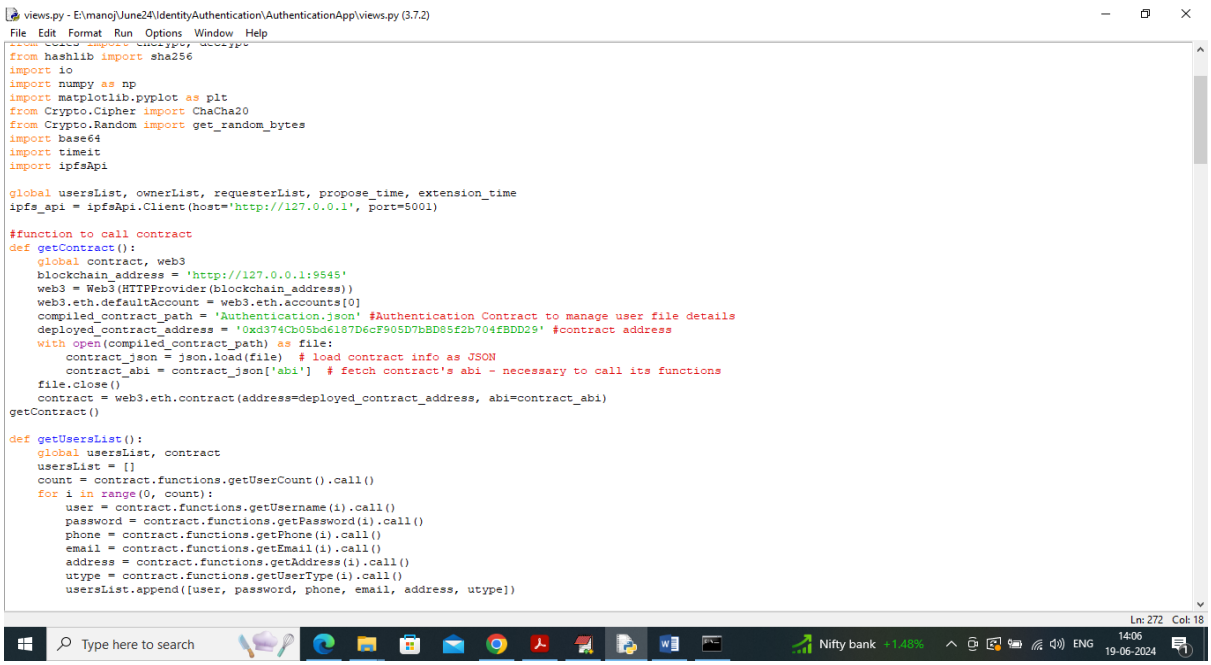
1) First go inside 'hello-eth/node-modules/bin' folder and then find and double click on 'runBlockchain.bat' file to get below page.



2) In above screen type command as ‘migrate’ and press enter key to get below page.



3) In above screen in white colour text can see ‘Authentication’ contract deployed and we got contract address also and this address need to specify in python code to call contract to save and get data.



4) In above screen read red colour comments to know about contract calling using contract address. In above screens can see contract deployed and running and let it run.

Modules Implementation

To implement this project we have designed following modules

1) Admin Module: admin can login to system using username and password as 'admin and admin'. After login admin will add all 'Data Owner or citizens' details. Admin will record only those owners whose has supported Address proof documents to upload. Admin can view all registered Data Owner details and can download their documents

2) User Register: using this module 'Data Owner and Data Requester' will register with the application. All requester attributes like Name, Contact number, address will get encrypted and saved in IPFS and Blockchain for future verification to access data.

3) Data Owner Login: data owner can login to system and then can see list of Requester along with their generated IPFS hash code which stored in Blockchain. Data owner can retrieve all hash code to verify their identity and then allow them to access data

4) Requester Login: Requester login to system and then Blockchain will verify Requester identity and upon successful identify verification then application allow 'Requester User' to access Data Owner details.

CHAPTER - 8
SOFTWARE ENVIRONMENT

8. SOFTWARE ENVIRONMENT

PYTHON LANGUAGE:

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed. Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

Python is a dynamic, high-level, free open source, and interpreted programming language. It supports object-oriented programming as well as procedural-oriented programming. In Python, we don’t need to declare the type of variable because it is a dynamically typed language. For example, x = 10 Here, x can be anything such as String, int, etc.

Features in Python:

There are many features in Python, some of which are discussed below as follows:

1. Free and Open Source

Python language is freely available at the official website and you can download it from the given download link below click on the Download Python keyword. Download Python Since

it is open-source, this means that source code is also available to the public. So you can download it, use it as well as share it.

2. Easy to code

Python is a high-level programming language. Python is very easy to learn the language as compared to other languages like C, C#, Javascript, Java, etc. It is very easy to code in the Python language and anybody can learn Python basics in a few hours or days. It is also a developer-friendly language.

3. Easy to Read

As you will see, learning Python is quite simple. As was already established, Python's syntax is really straightforward. The code block is defined by the indentations rather than by semicolons or brackets.

4. Object-Oriented Language

One of the key features of Python is Object-Oriented programming. Python supports object-oriented language and concepts of classes, object encapsulation, etc.

5. GUI Programming Support

Graphical User interfaces can be made using a module such as PyQt5, PyQt4, wxPython, or Tk in python. PyQt5 is the most popular option for creating graphical apps with Python.

6. High-Level Language

Python is a high-level language. When we write programs in Python, we do not need to remember the system architecture, nor do we need to manage the memory.

7. Extensible feature

Python is an Extensible language. We can write some Python code into C or C++ language and also we can compile that code in C/C++ language.

8. Easy to Debug

Excellent information for mistake tracing. You will be able to quickly identify and correct the majority of your program's issues once you understand how to interpret Python's error traces.

Simply by glancing at the code, you can determine what it is designed to perform.

9. Python is a Portable language

Python language is also a portable language. For example, if we have Python code for windows and if we want to run this code on other platforms such as Linux, Unix, and Mac then we do not need to change it, we can run this code on any platform.

10. Python is an Integrated language

Python is also an Integrated language because we can easily integrate Python with other languages like C, C++, etc.

11. Interpreted Language:

Python is an Interpreted Language because Python code is executed line by line at a time. like other languages C, C++, Java, etc. there is no need to compile Python code this makes it easier to debug our code. The source code of Python is converted into an immediate form called bytecode.

12. Large Standard Library

Python has a large standard library that provides a rich set of modules and functions so you do not have to write your own code for every single thing. There are many libraries present in Python such as regular expressions, unit-testing, web browsers, etc.

13. Dynamically Typed Language

Python is a dynamically-typed language. That means the type (for example- int, double, long, etc.) for a variable is decided at run time not in advance because of this feature we don't need to specify the type of variable.

14. Frontend and backend development

With a new project py script, you can run and write Python codes in HTML with the help of some simple tags <py-script>, <py-env>, etc. This will help you do frontend development work in Python like javascript. Backend is the strong forte of Python it's extensively used for this work cause of its frameworks like Django and Flask.

15. Allocating Memory Dynamically

In Python, the variable data type does not need to be specified. The memory is automatically allocated to a variable at runtime when it is given a value. Developers do not need to write int y = 18 if the integer value 15 is set to y. You may just type y=18.

GANACHE

- Ganache is a user-friendly interface for monitoring Ethereum blockchain activities. It simplifies tracking of accounts, transactions, and smart contracts, making it accessible even for users without in-depth blockchain expertise. Ganache offers detailed transaction information, including sender, receiver, amounts, gas usage, and success status, aiding debugging and ensuring transaction accuracy. It also tracks smart contract deployments, confirming correct deployment and functionality. This transparency simplifies monitoring and verification processes.
- Ganache lets us dive into the details of each block on the Ethereum blockchain. We can find out when a particular block was added, what transactions took place within it, and how much computing power (gas) was used. Ganache also enables data retrieval from stored blocks, allowing developers to access and analyze specific block information.

METAMASK

- Metamask is both an Ethereum wallet and a browser extension. It simplifies cryptocurrency management and provides direct access to DApps, making interactions with blockchain applications easier.
- In the project, Metamask ensures secure Ethereum transactions, promoting transparency by showing the deduction of ETH as fees. This transparency maintains accuracy and ensures confident, reliable financial interactions within the system.

LIBRARIES/PACKAGES:

TensorFlow

TensorFlow is a free and open-source software library for dataflow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine

learning applications such as neural networks. It is used for both research and production at Google.

TensorFlow was developed by the Google Brain team for internal Google use. It was released under the Apache 2.0 open-source license on November 9, 2015.

Numpy

Numpy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

It is the fundamental package for scientific computing with Python. It contains various features including these important ones:

- A powerful N-dimensional array object
- Sophisticated (broadcasting) functions
- Tools for integrating C/C++ and Fortran code
- Useful linear algebra, Fourier transform, and random number capabilities

Besides its obvious scientific uses, Numpy can also be used as an efficient multi-dimensional container of generic data. Arbitrary data-types can be defined using Numpy which allows Numpy to seamlessly and speedily integrate with a wide variety of databases.

Pandas

Pandas is an open-source Python Library providing high-performance data manipulation and analysis tool using its powerful data structures. Python was majorly used for data munging and preparation. It had very little contribution towards data analysis. Pandas solved this problem. Using Pandas, we can accomplish five typical steps in the processing and analysis of data, regardless of the origin of data load, prepare, manipulate, model, and analyze. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, Statistics, analytics, etc.

Matplotlib

Matplotlib is a Python 2D plotting library which produces publication quality figures in a

variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter Notebook, web application servers, and four graphical user interface toolkits. Matplotlib tries to make easy things easy and hard things possible. You can generate plots, histograms, power spectra, bar charts, error charts, scatter plots, etc., with just a few lines of code. For examples, see the sample plots and thumbnail gallery.

For simple plotting the pyplot module provides a MATLAB-like interface, particularly when combined with IPython. For the power user, you have full control of line styles, font properties, axes properties, etc, via an object oriented interface or via a set of functions familiar to MATLAB users.

Scikit – learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

CHAPTER - 9
TESTING

9. SYSTEM TESTING

System testing, also referred to as system-level tests or system-integration testing, is the process in which a quality assurance (QA) team evaluates how the various components of an application interact together in the full, integrated system or application. System testing verifies that an application performs tasks as designed. This step, a kind of black box testing, focuses on the functionality of an application. System testing, for example, might check that every kind of user input produces the intended output across the application.

Phases of system testing:

A video tutorial about this test level. System testing examines every component of an application to make sure that they work as a complete and unified whole. A QA team typically conducts system testing after it checks individual modules with functional or user-story testing and then each component through integration testing.

If a software build achieves the desired results in system testing, it gets a final check via acceptance testing before it goes to production, where users consume the software. An app-dev team logs all defects, and establishes what kinds and amount of defects are tolerable.

System testing ensures that the software meets business and technical requirements while functioning correctly under different conditions. It validates the complete system's behavior, performance, and security before deployment.

9.1 Software Testing Strategies:

Optimization of the approach to testing in software engineering is the best way to make it effective. A software testing strategy defines what, when, and how to do whatever is necessary to make an end-product of high quality. Usually, the following software testing strategies and their combinations are used to achieve this major objective:

Static Testing:

The early-stage testing strategy is static testing: it is performed without actually running the developing product. Basically, such desk-checking is required to detect bugs and issues that are present in the code itself. Such a check-up is important at the pre-deployment stage as it helps avoid problems caused by errors in the code and software structure deficits.

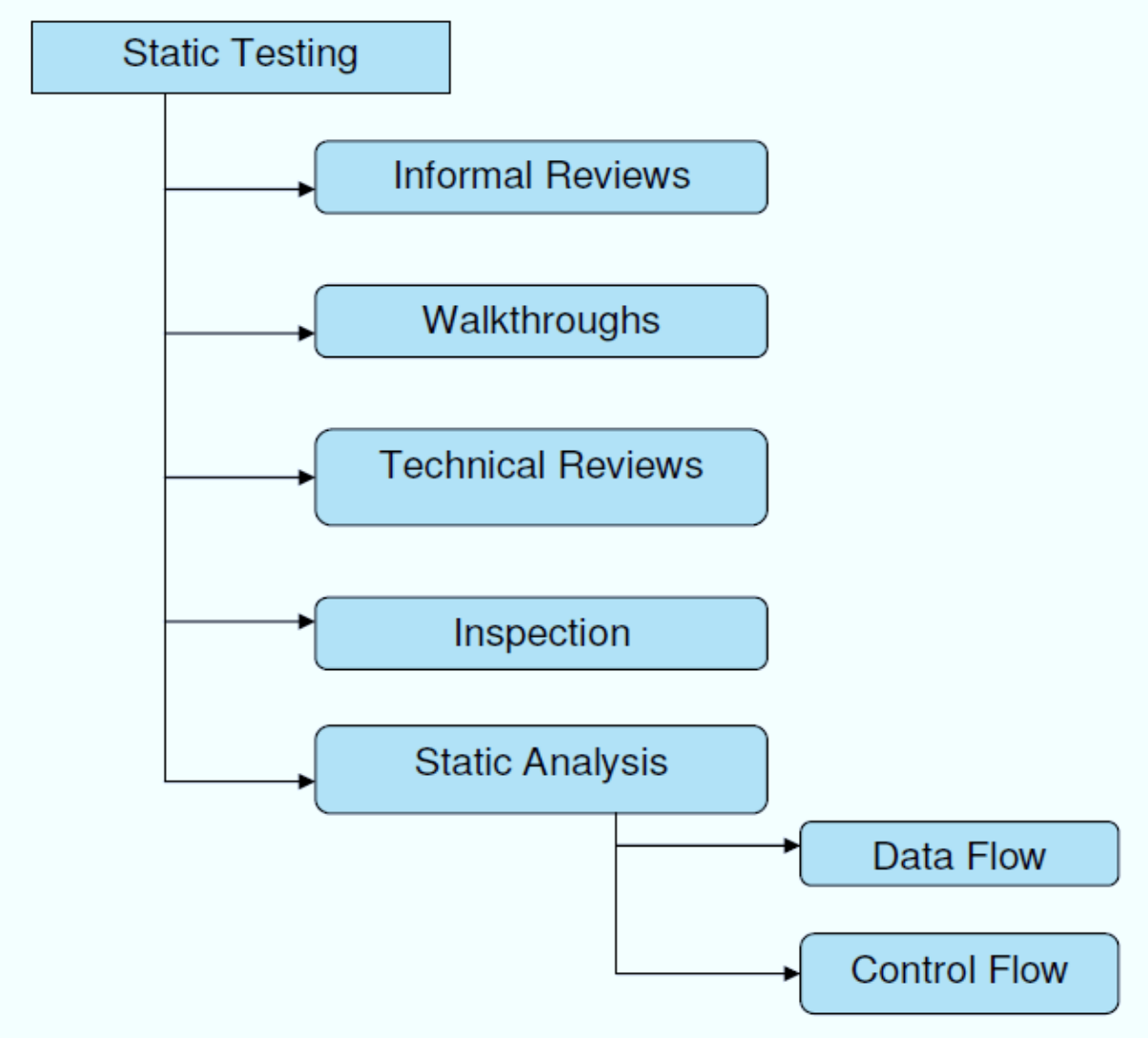


Fig. 9.1.1 Static Testing

Structural Testing:

It is not possible to effectively test software without running it. Structural testing, also known as white-box testing, is required to detect and fix bugs and errors emerging during the pre-production stage of the software development process. At this stage, unit testing based on the software structure is performed using regression testing. In most cases, it is an automated process working within the test automation framework to speed up the development process at this stage. Developers and QA engineers have full access to the software's structure and data flows (data flows testing), so they could track any changes (mutation testing) in the system's behavior by comparing the tests' outcomes with the results of previous iterations (control flow testing).

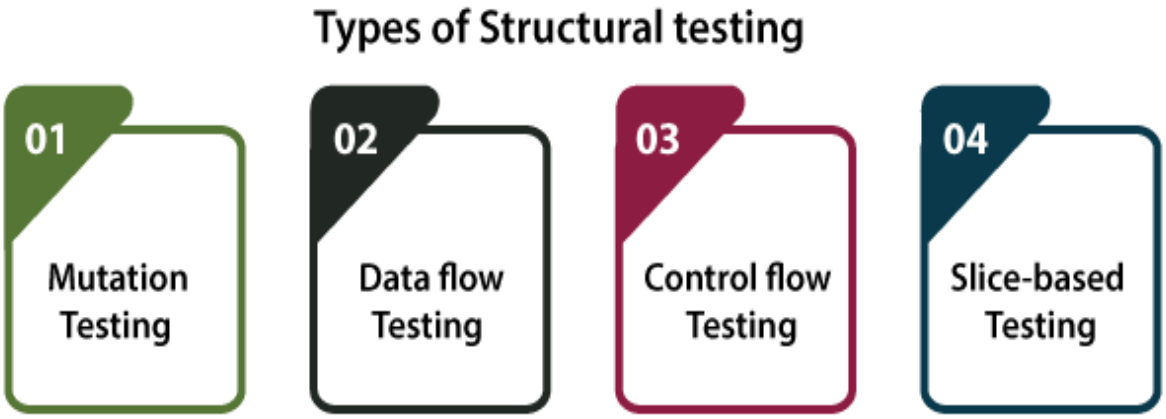


Fig. 9.1.2 Types of Structural Testing

Behavioral Testing:

The final stage of testing focuses on the software's reactions to various activities rather than on the mechanisms behind these reactions. In other words, behavioral testing, also known as black-box testing, presupposes running numerous tests, mostly manual, to see the product from the user's point of view. QA engineers usually have some specific information about a business or other purposes of the software ('the black box') to run usability tests, for example, and react to bugs as regular users of the product will do. Behavioral testing also may include automation (regression tests) to eliminate human error if repetitive activities are required. For example, you may need to fill 100 registration forms on the website to see how the product copes with such an activity, so the automation of this test is preferable.

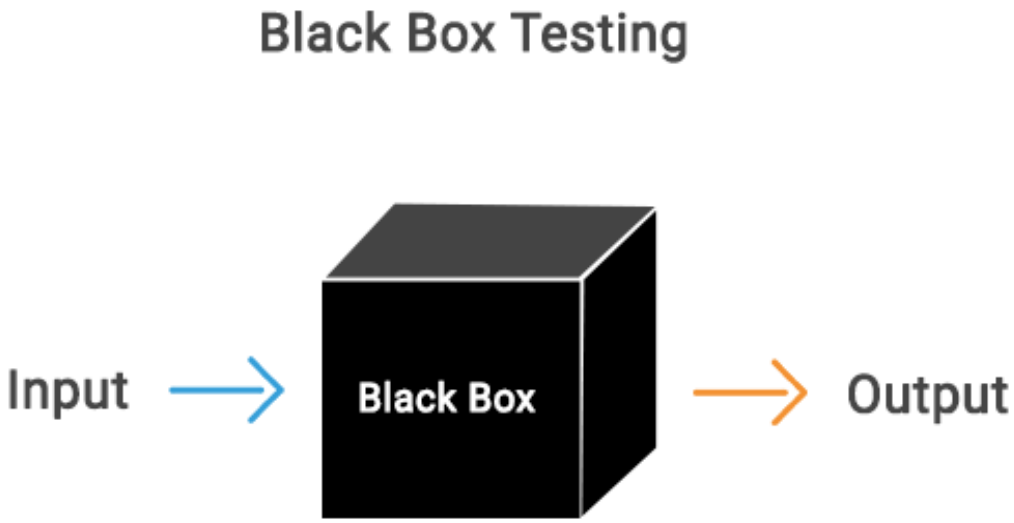


Fig. 9.1.3 Black Box Testing

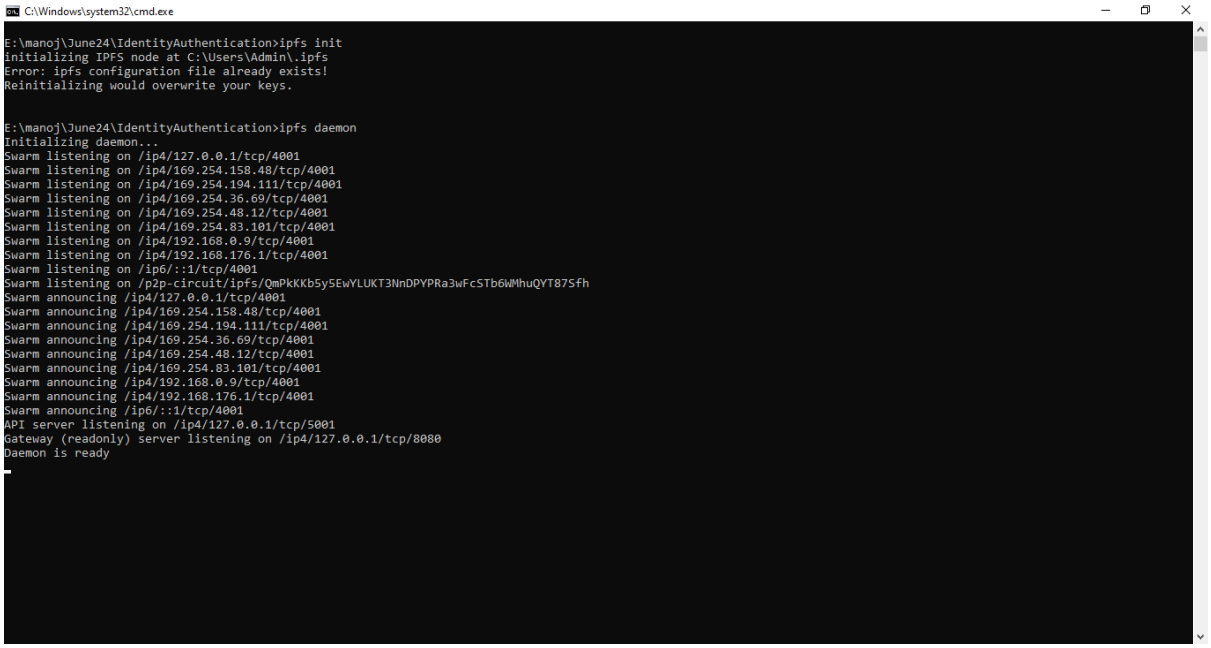
9.2 TEST CASES:

S.NO	INPUT	If available	If not available
1	Admin	Admin can enable secure login, data owner registration, document management, and access oversight for government data sharing.	There is no process
2	Data Requester	Data Requester can allow secure login, identity verification, and access to data from registered Data Owners.	There is no process
3	Data Owner	Data Owner can facilitate secure login, data management, and verification of access requests from authenticated Data Requesters.	There is no process

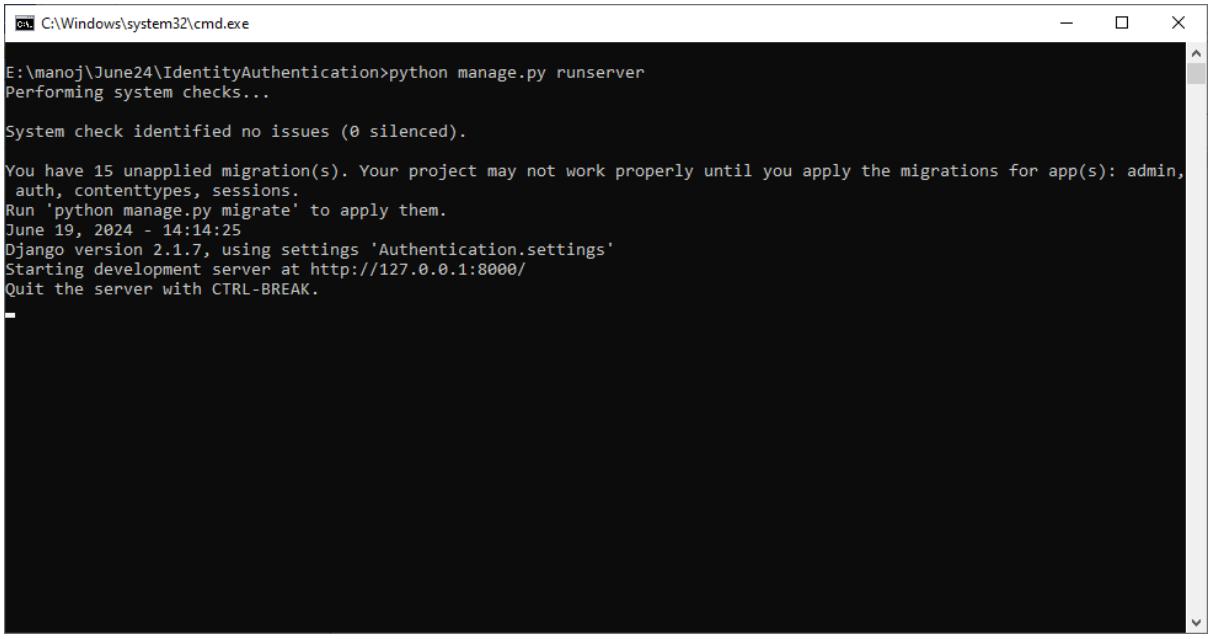
CHAPTER - 10
RESULTS

10. RESULTS

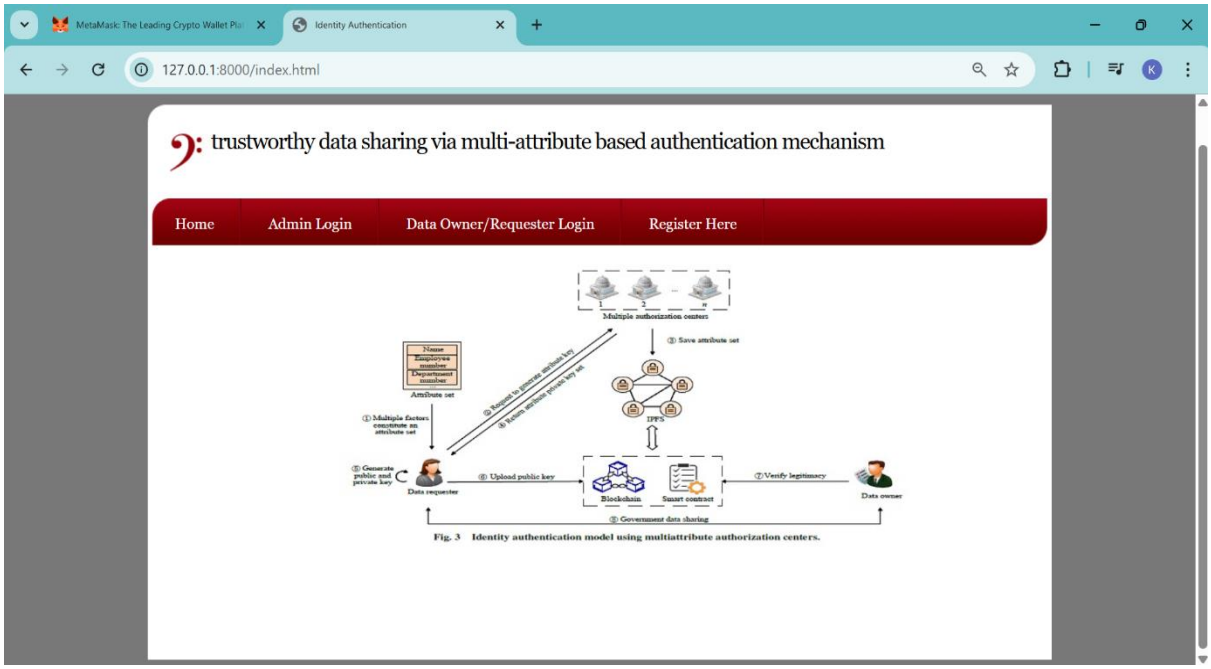
Double click on 'Start_IPFS.bat' file to start IPFS server and get below page



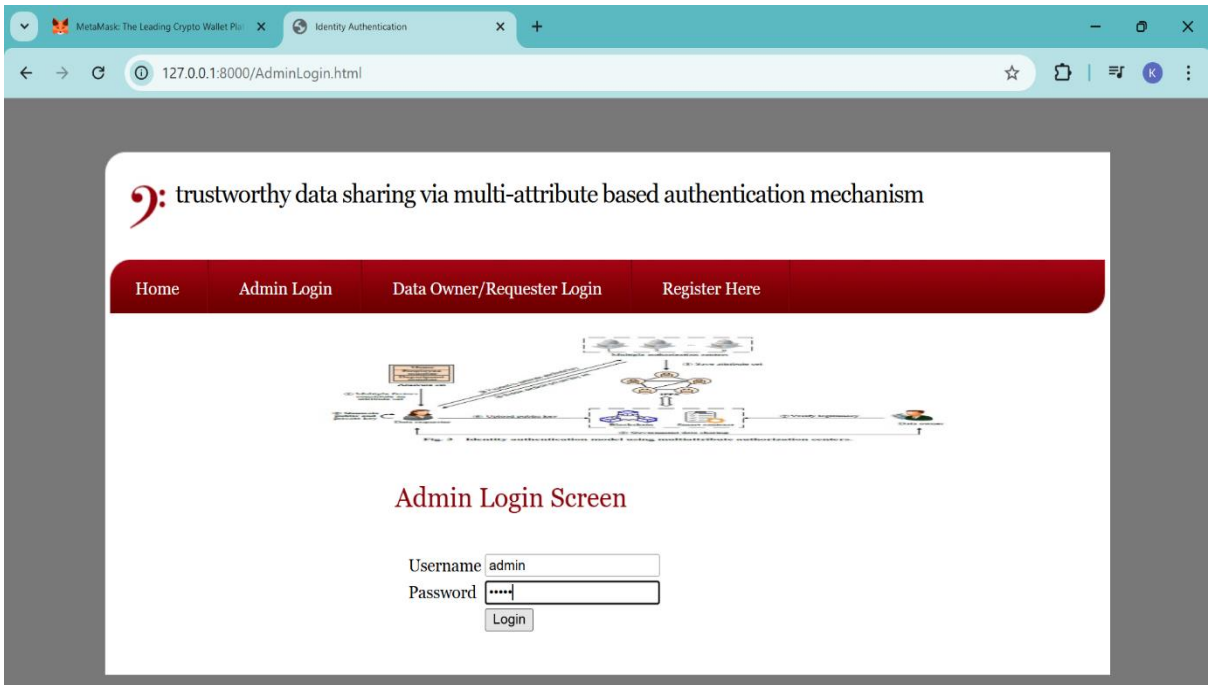
In above screen IPFS server started and now double click on 'runServer.bat' file to start python web server and get below page



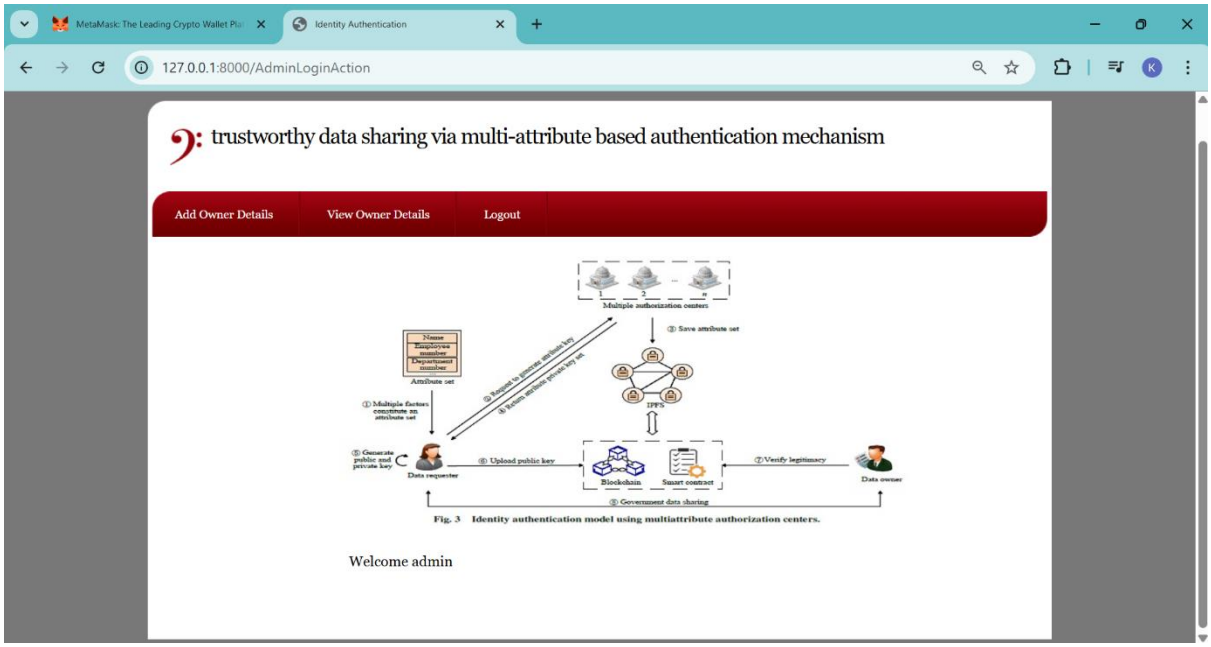
In above screen python server started and now open browser and enter URL as http://127.0.0.1:8000/index.html and press enter key to get below page



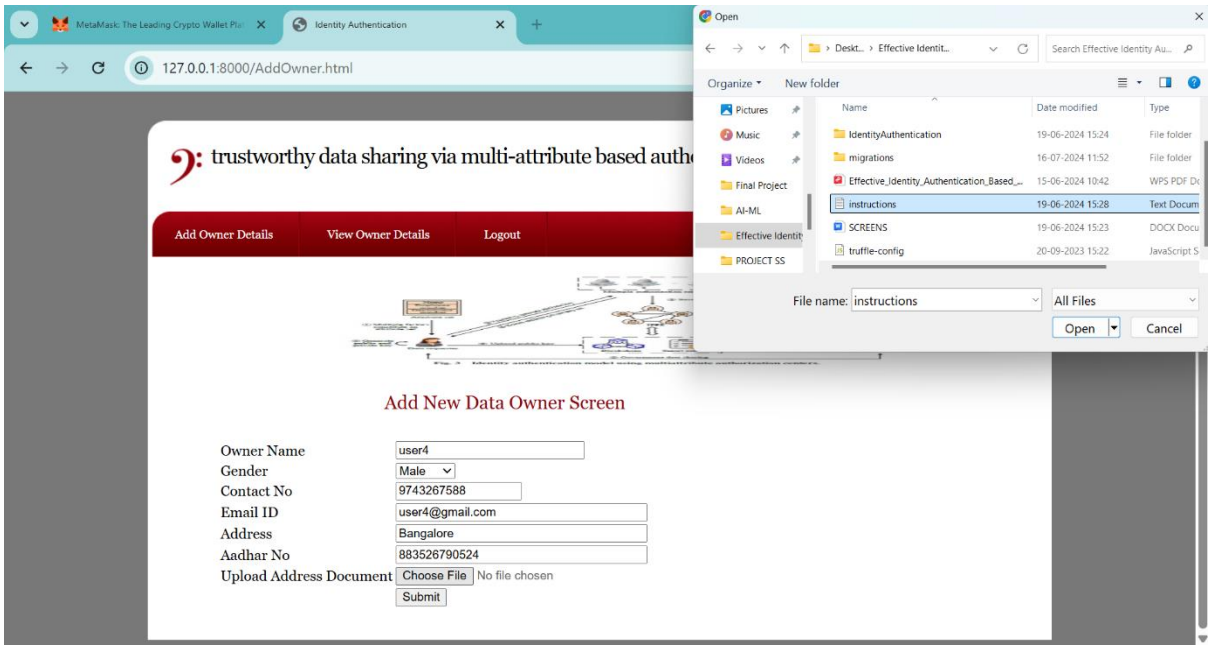
In above screen click on 'Admin Login' link to get below login screen



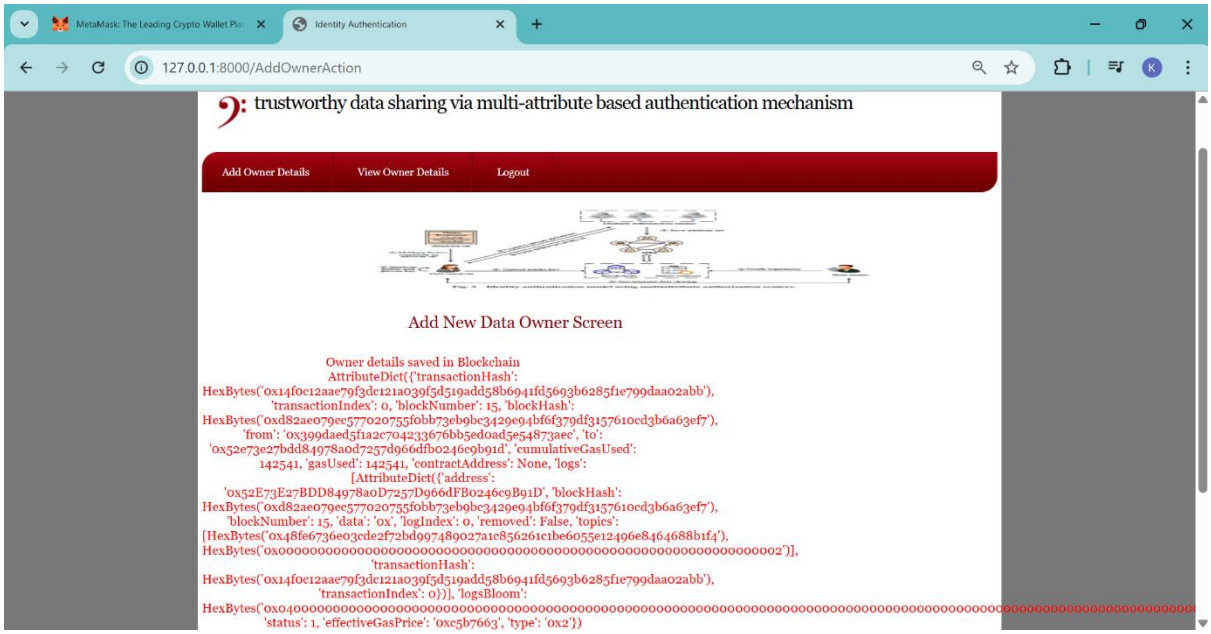
In above screen admin is login and after login will get below page



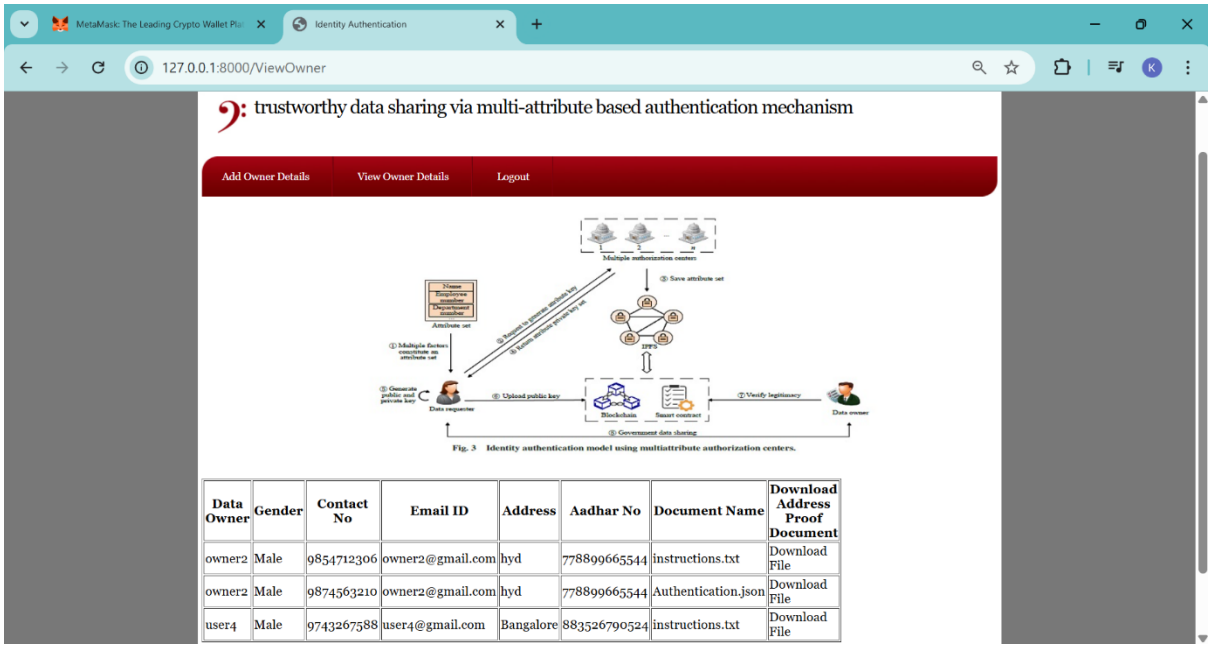
In above screen admin can click on 'Add Owner Details' link to get below page



In above screen admin adding 'Data Owner' details and then uploading supporting address proof document and then press button to saved data owner details in Blockchain and get below page

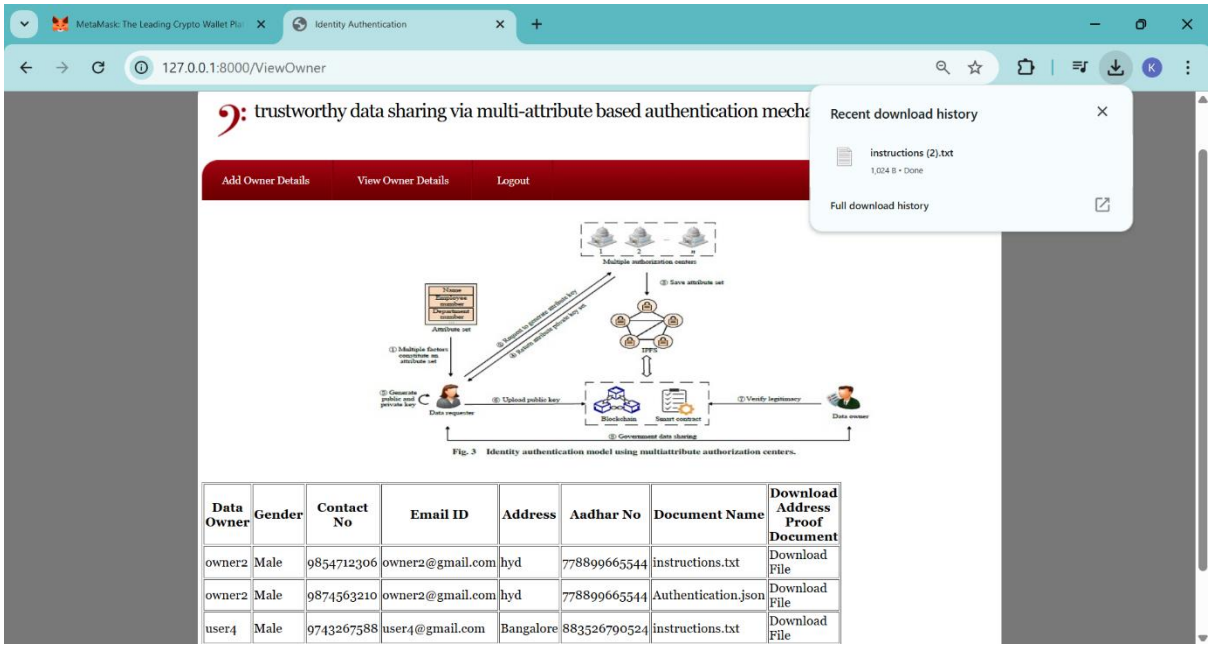


In above screen can see data owner details saved in Blockchain and then I am displaying entire logs obtained from Blockchain and this log contains information like Block No, Transaction No, Transaction hash code and many other details. Similarly by using above screen admin can add any number of 'Data Owner' details. Now click on 'View Owner Details' link to view all registered owner and get below page



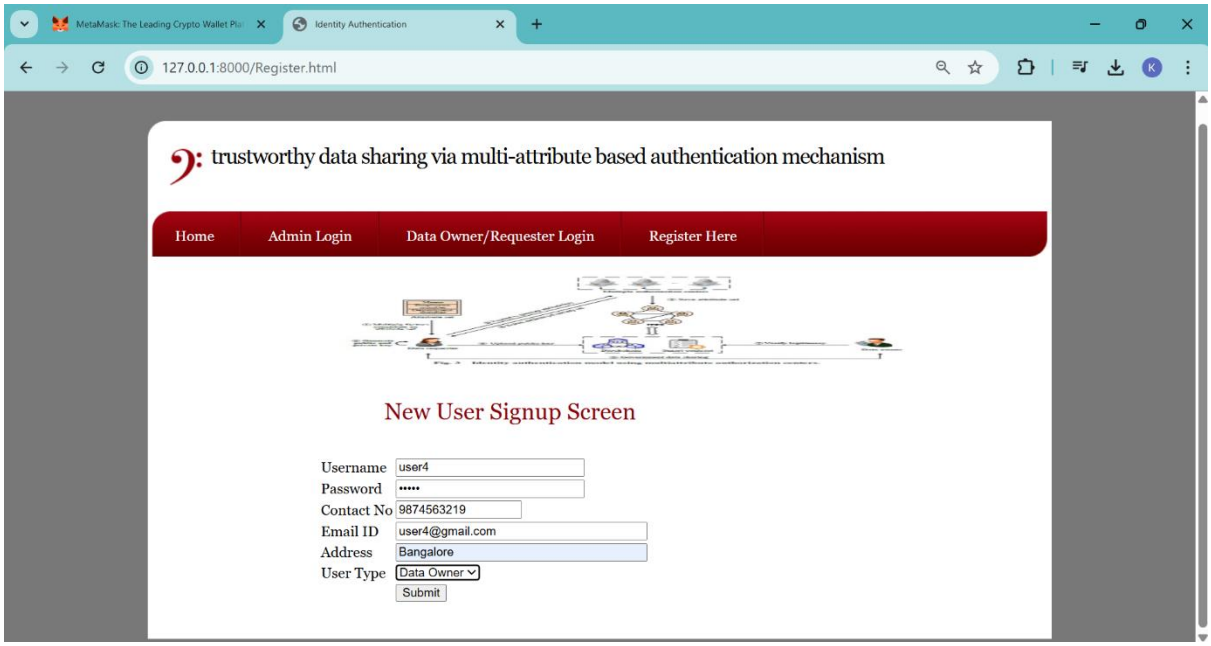
Data Owner	Gender	Contact No	Email ID	Address	Aadhar No	Document Name	Download Address Proof Document
owner2	Male	9854712306	owner2@gmail.com	hyd	778899665544	instructions.txt	Download File
owner2	Male	9874563210	owner2@gmail.com	hyd	778899665544	Authentication.json	Download File
user4	Male	9743267588	user4@gmail.com	Bangalore	883526790524	instructions.txt	Download File

In above screen admin can see list of added Blockchain owner details and can click on 'Download File' link to download file and get below page

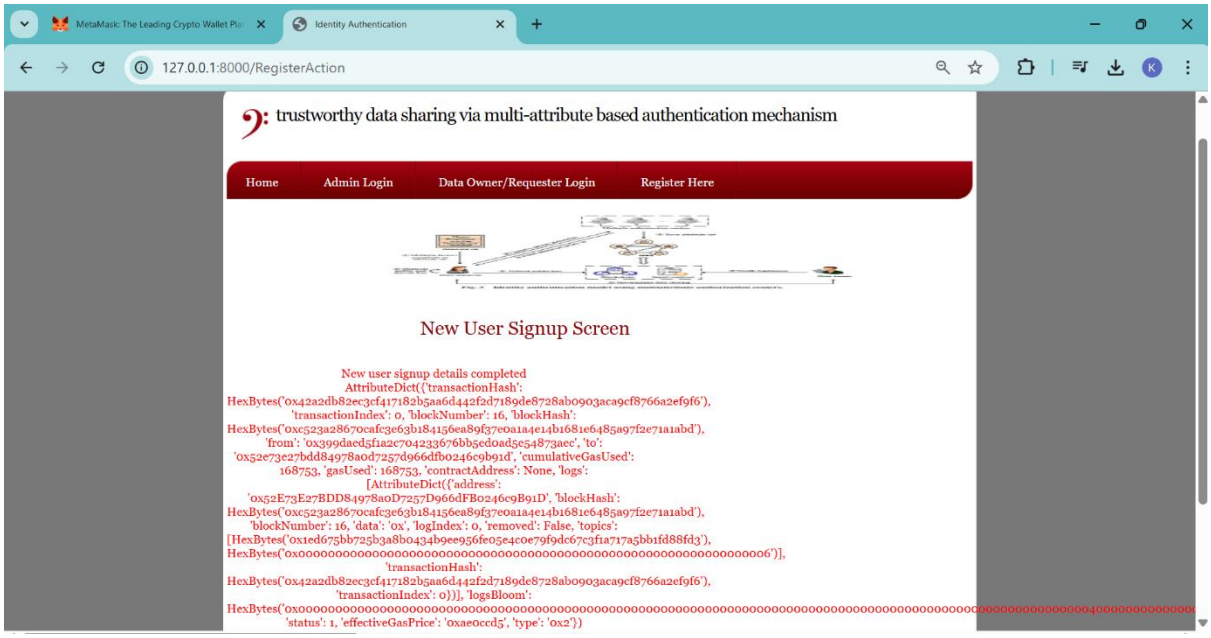


Data Owner	Gender	Contact No	Email ID	Address	Aadhar No	Document Name	Download Address Proof Document
owner2	Male	9854712306	owner2@gmail.com	hyd	778899665544	instructions.txt	Download File
owner2	Male	9874563210	owner2@gmail.com	hyd	778899665544	Authentication.json	Download File
user4	Male	9743267588	user4@gmail.com	Bangalore	883526790524	instructions.txt	Download File

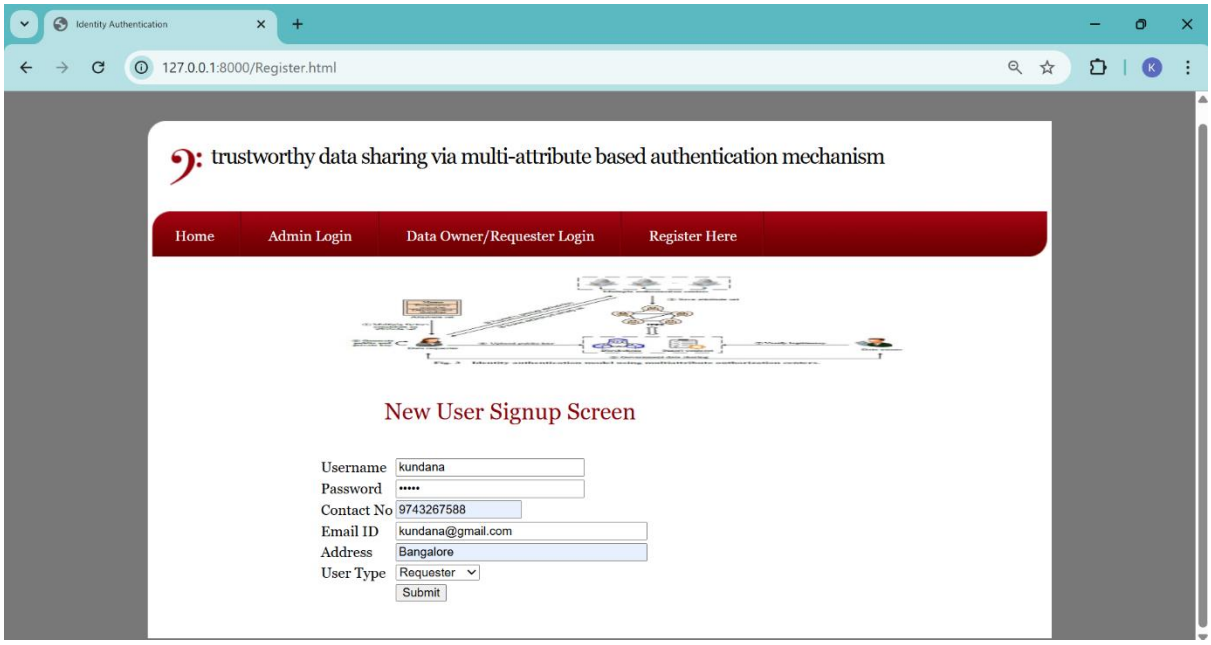
In above screen can see file is downloaded in top right browser side and now logout and sign up new user.



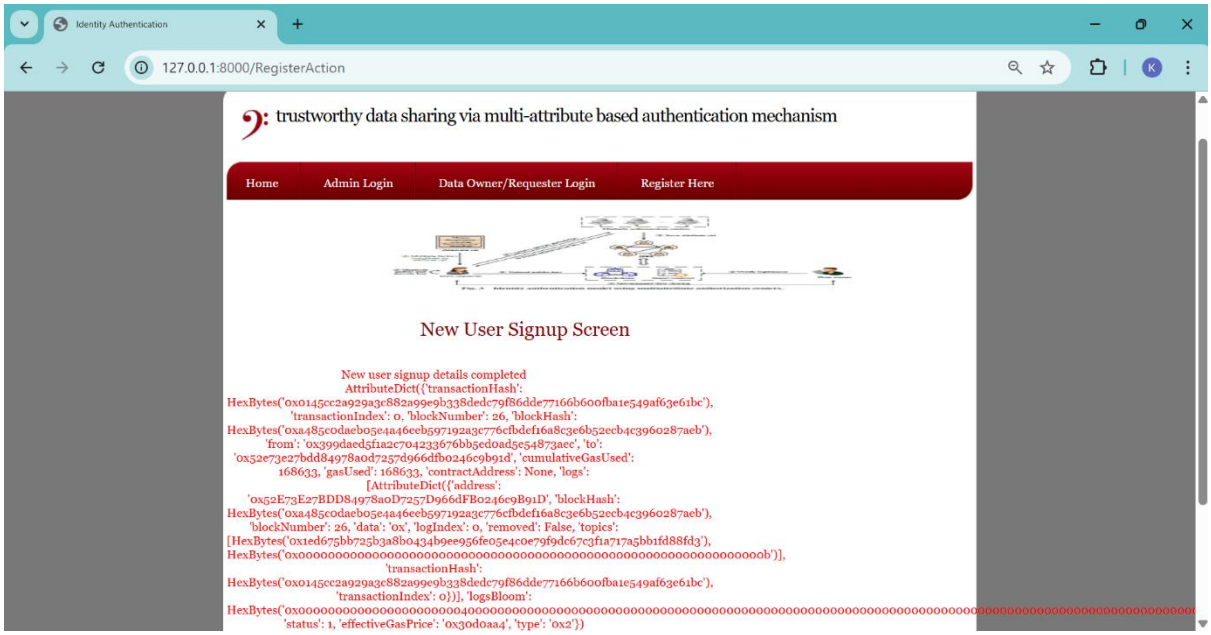
In above screen adding data owner details and then press button to save details in IPFS and Blockchain and then will get below output



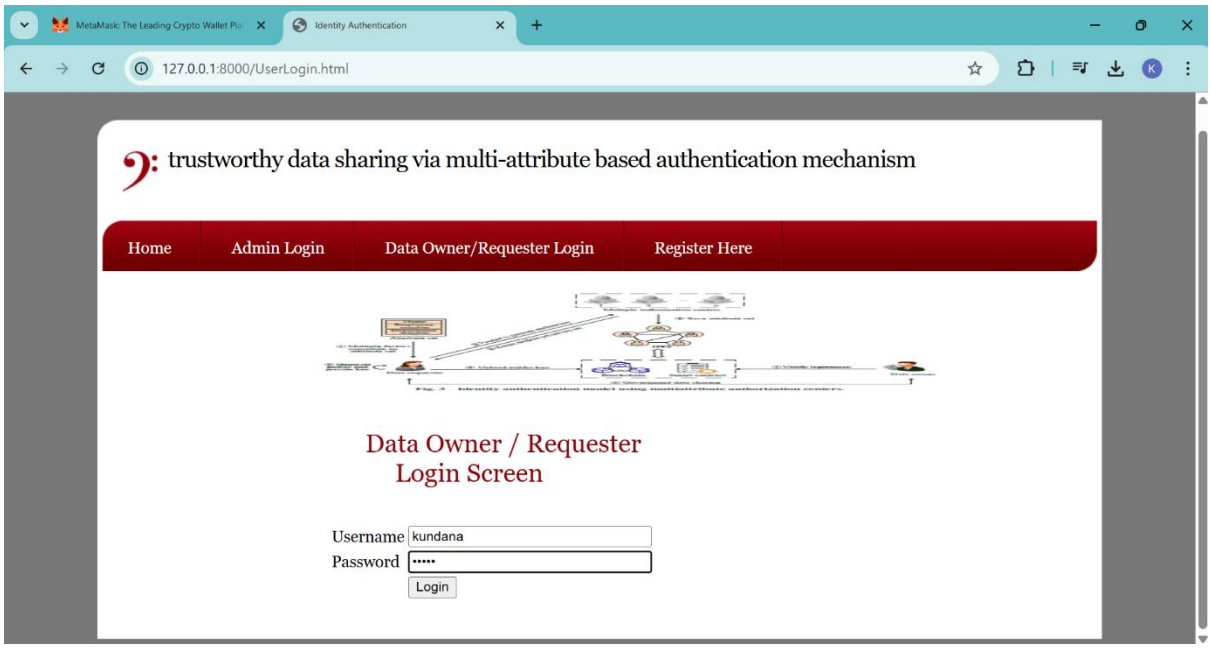
In above screen data owner details added in Blockchain and now similarly adding 'Requester' details also



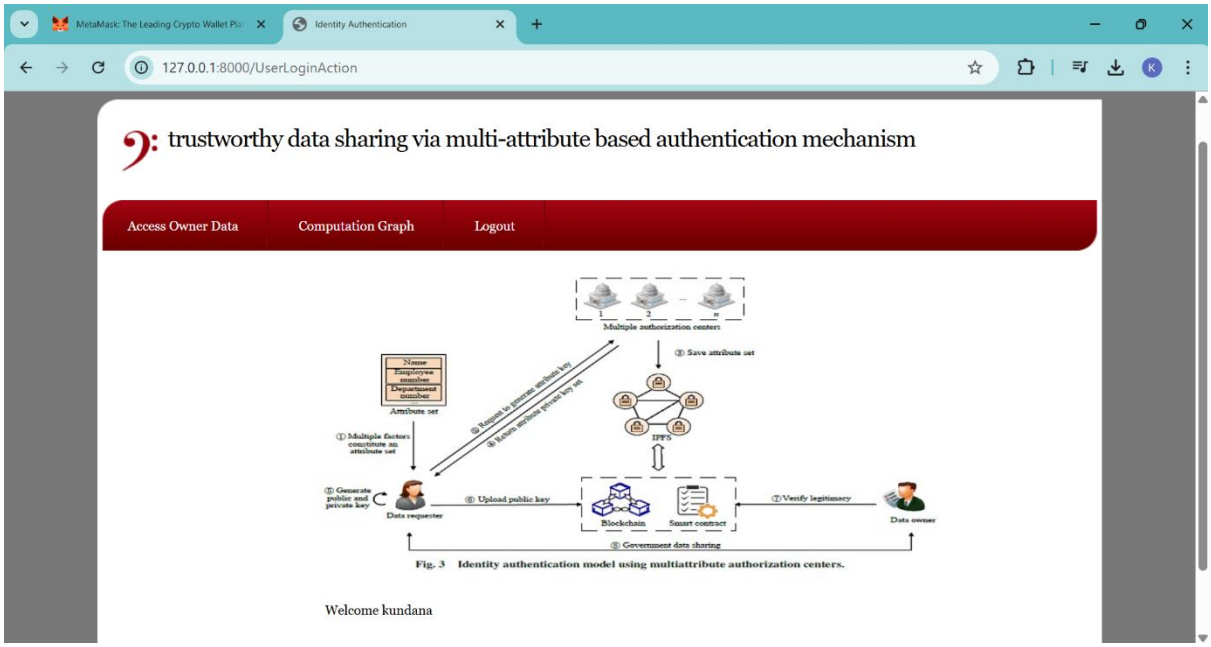
In above screen adding requester details and then press button to get below page



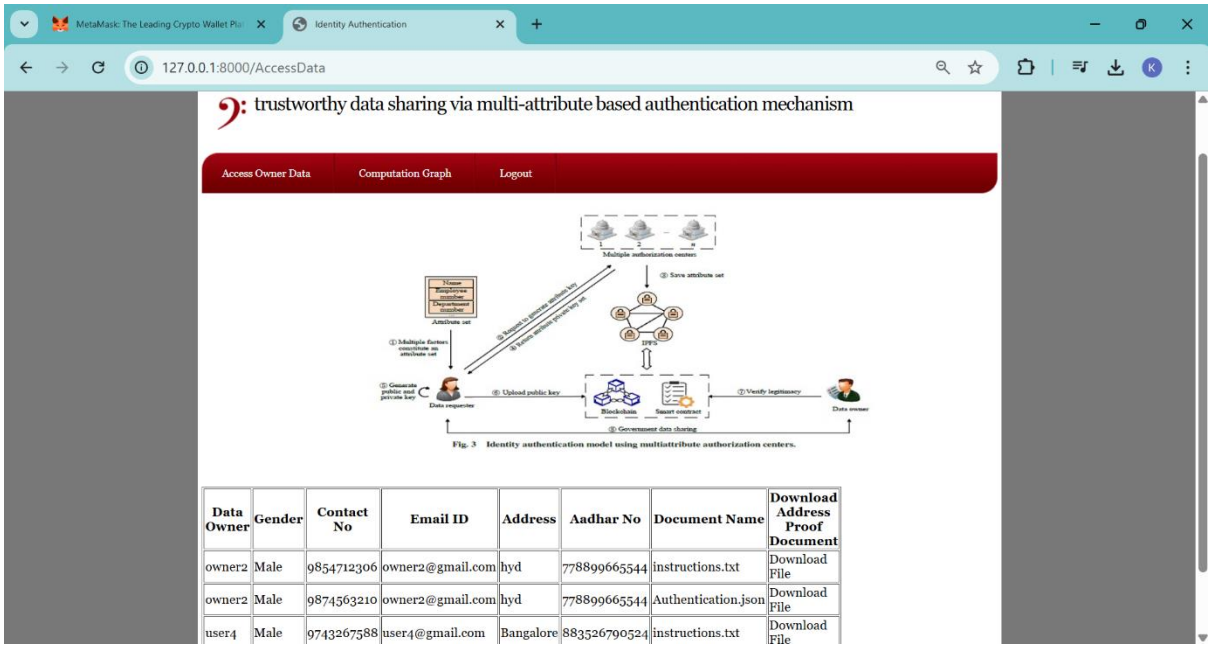
In above screen requester details added to IPFS and Blockchain and now click on 'Data Owner/Requester Login' link to get below screen



In above screen 'Requester' is login and after login will get below page

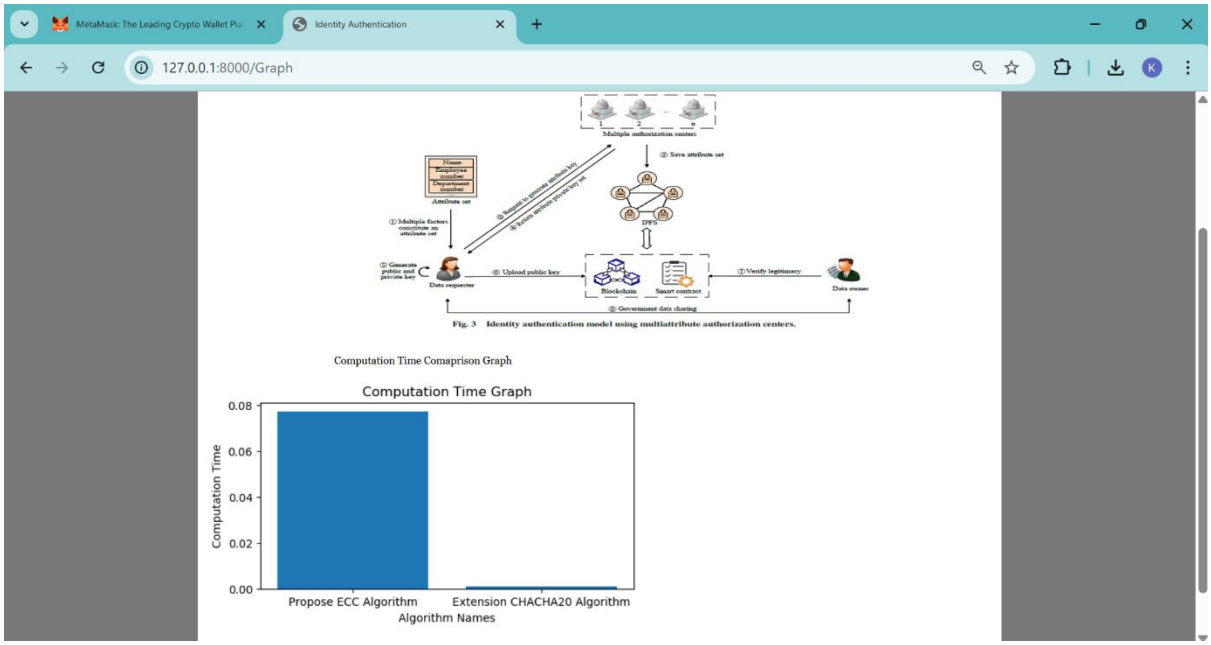


In above screen Requester can click on 'Access Owner Data' link to access all available Data Owner details and get below page

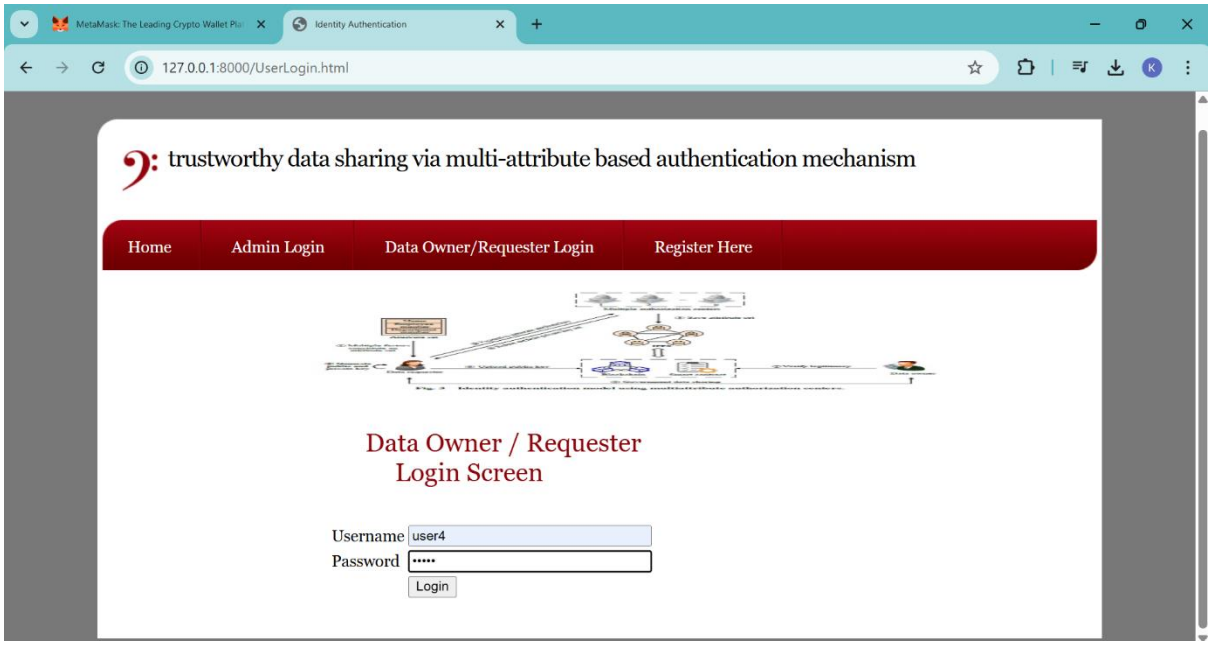


Data Owner	Gender	Contact No	Email ID	Address	Aadhar No	Document Name	Download Address Proof Document
owner2	Male	9854712306	owner2@gmail.com	hyd	778899665544	instructions.txt	Download File
owner2	Male	9874563210	owner2@gmail.com	hyd	778899665544	Authentication.json	Download File
user4	Male	9743267588	user4@gmail.com	Bangalore	883526790524	instructions.txt	Download File

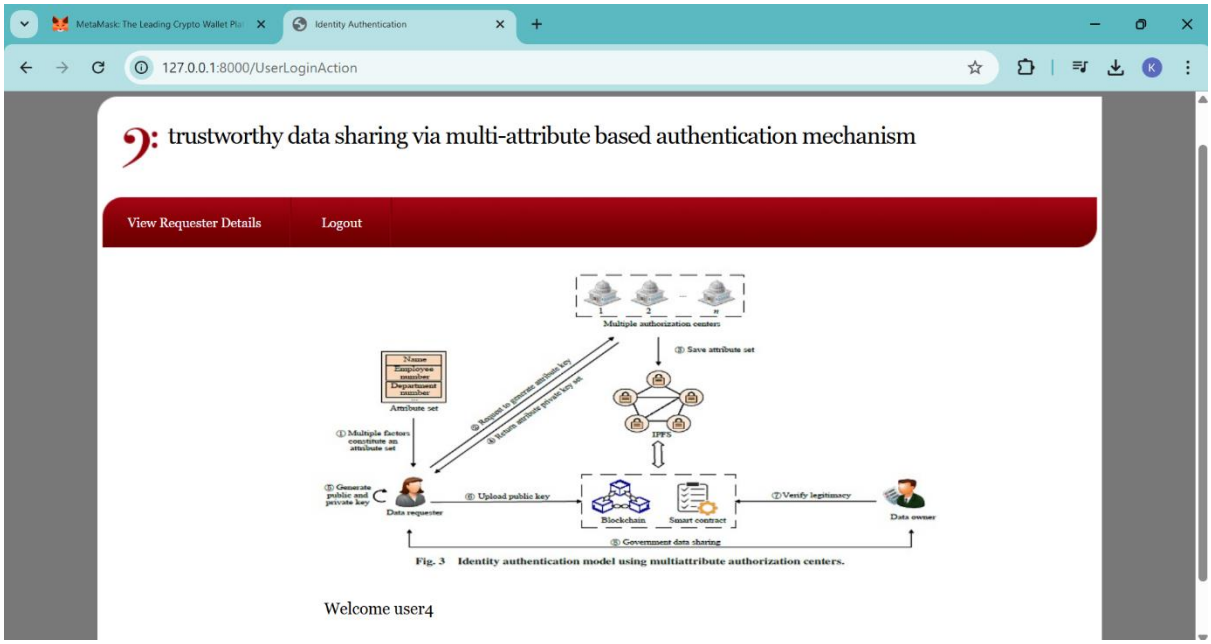
In above screen 'Requester' can access all details of 'Data Owner' and only those Requester whose keys are available in Blockchain can able access data and now click on 'Comparison Graph' link to get below graph



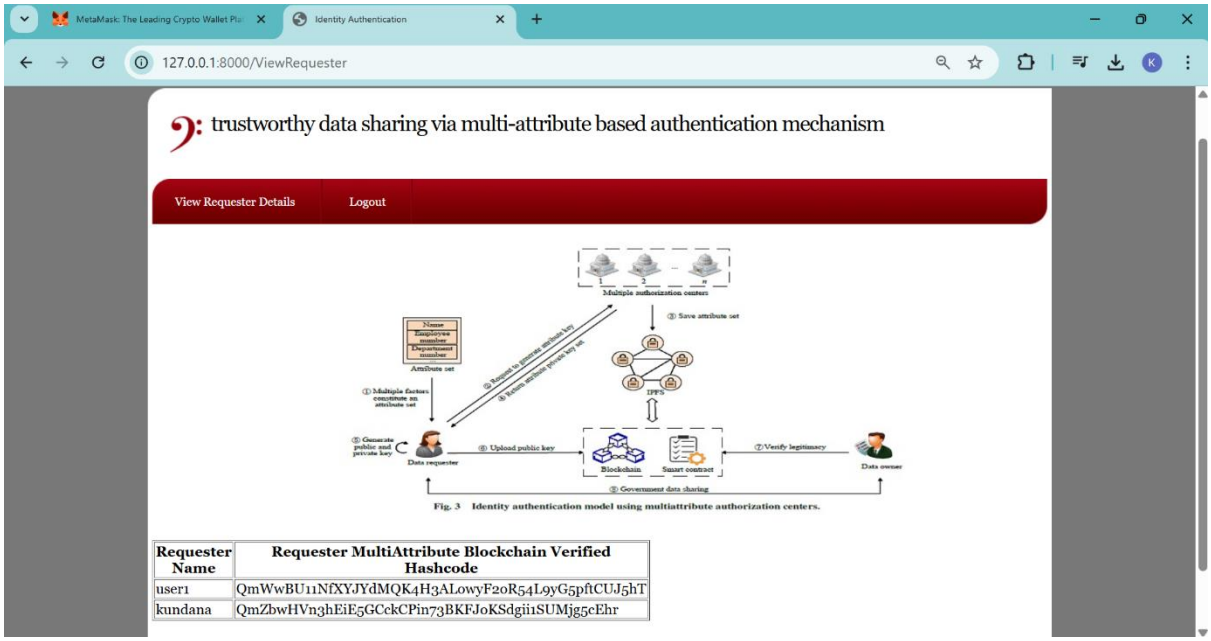
In above screen x-axis represents algorithm names and y-axis represents computation time and in both algorithms extension CHACHA20 took less computation compare to propose algorithm and now logout and login as 'Data Owner' to view all verified Requester Hash code.



In above screen 'Data Owner' is login and after login will get below page



In above screen 'Data Owner' can click on 'View Requester Details' link to get below page



In above screen 'Data Owner' can view different Requester details along with IPFS hash code to verify with Blockchain.

Similarly, by following above screen Government can share secured data with Requester.

CHAPTER - 11
CONCLUSION

10. CONCLUSION

The developed system provides a robust framework for secure government data sharing through advanced blockchain and IPFS integration. It ensures stringent authentication and access control mechanisms for both administrators and users. The Admin Module enables efficient management of data owner registrations and document uploads, while the Data Owner Module facilitates secure storage and controlled access to sensitive information. Concurrently, the Data Requester Module ensures authenticated access to authorized data, promoting transparency and accountability in data sharing practices. Overall, the system enhances data security, integrity, and privacy, crucial for maintaining trust in governmental data handling processes.

CHAPTER - 12
FUTURE SCOPE

12. FUTURE SCOPE

In the future, the system could integrate AI-based fraud detection to identify unusual login attempts and prevent fraudulent activities in real time. Additionally, implementing multi-factor authentication (MFA) with biometrics or OTP verification, along with blockchain-based authentication, could enhance security layers. These advancements would ensure robust protection against unauthorized access and strengthen overall system security. Furthermore, leveraging advanced machine learning models for continuous monitoring can help detect evolving fraud patterns with greater accuracy. The integration of AI and blockchain can also enhance user trust by providing a transparent and tamper-proof authentication framework.

CHAPTER - 13
BIBLIOGRAPHY

13. BIBLIOGRAPHY

[1] L. Qi, W. Lin, X. Zhang, W. Dou, X. Xu, and J. Chen, A correlation graph-based approach for personalized and compatible web APIs recommendation in mobile APP development, IEEE Trans. Knowl. Data Eng., vol. 35, no. 6, pp. 5444−5457, 2023.

[2] X. Zhou, W. Liang, W. Li, K. Yan, S. Shimizu, and K. I. K. Wang, Hierarchical adversarial attacks against graphneural-network-based IoT network intrusion detection system, IEEE Internet Things J., vol. 9, no. 12, pp. 9310− 9319, 2022.

[3] P. Zhang, M. Zhou, Q. Zhao, A. Abusorrah, and O. O. Bamasag, A performance-optimized consensus mechanism for consortium blockchains consisting of trust-varying nodes, IEEE Trans. Netw. Sci. Eng., vol. 8, no. 3, pp. 2147−2159, 2021.

[4] Z. Rahman, I. Khalil, X. Yi, and M. Atiquzzaman, Blockchain-based security framework for a critical industry 4. 0 cyber-physical system, IEEE Commun. Mag., vol. 59, no. 5, pp. 128−134, 2021.

[5] W. Diffie and M. Hellman, New directions in cryptography, IEEE Trans. Inf. Theory, vol. 22, no. 6, pp. 644− 654, 1976.

[6] W. Wang, H. Huang, L. Zhang, Z. Han, C. Qiu, and C. Su, BlockSLAP: Blockchain-based secure and lightweight authentication protocol for smart grid, in Proc. IEEE 19th Int. Conf. on Trust, Security and Privacy in Computing and Communications (TrustCom), Guangzhou, China, 2020, pp. 1332−1338.

[7] G. Li, X. Ren, J. Wu, W. Ji, H. Yu, J. Cao, and R. Wang, Blockchain-based mobile edge computing system, Inf. Sci., vol. 561, pp. 70−80, 2021.

[8] M. Di Mauro, G. Galatro, M. Longo, F. Postiglione, and M. Tambasco, HASFC: A MANO-compliant framework for availability management of service chains, IEEE Commun. Mag., vol. 59, no. 6, pp. 52−58, 2021.

[9] Y. M. Tseng, J. L. Chen, and S. S. Huang, A lightweight leakage-resilient identity-based mutual authentication and key exchange protocol for resource-limited devices, Comput. Networks, vol. 196, p. 108246, 2021.

[10] H. Boche, R. F. Schaefer, S. Baur, and H. V. Poor, On the algorithmic computability of the secret key and authentication capacity under channel, storage, and privacy leakage constraints, IEEE Trans. Signal Process., vol. 67, no. 17, pp. 4636−4648, 2019.

[11] M. A. Khan, I. U. Din, T. Majali, and B. S Kim, A survey of authentication in internet of things-enabled healthcare systems, Sensors, vol. 22, no. 23, p. 9089, 2022.

[12] Y. Yu, Y. Li, J. Tian, and J. Liu, Blockchain-based solutions to security and privacy issues in the internet of things, IEEE Wireless Commun., vol. 25, no. 6, pp. 12−18, 2018.

[13] M. Asghar, R. R. M. Doss, and L. Pan, A scalable and efficient PKI based authentication protocol for VANETs, in Proc. 28th Int. Telecommunication Networks and Applications Conf. (ITNAC), Sydney, Australia, 2018, pp. 1−3.

[14] F. Marino, C. Moiso, and M. Petracca, PKIoT: A public key infrastructure for the internet of things, Trans. Emerg. Telecommun. Technol., vol. 30, no. 10, p. e3681, 2019.

[15] H. Qiu, M. Qiu, and R. Lu, Secure V2X communication network based on intelligent PKI and edge computing, IEEE Network, vol. 34, no. 2, pp. 172−178, 2020.

[16] J. Arm, P. Fiedler, and O. Bastan, Offline access to a vehicle via PKI-based authentication, in Proc. Int. Conf. on Computer Safety, Reliability, and Security, York, UK, 2021, pp. 76−88.

[17] D. D. F. Maesa and P. Mori, Blockchain 3. 0 applications survey, J. Parallel Distrib. Comput., vol. 138, pp. 99−114, 2020.

[18] C. Feng, K. Yu, A. K. Bashir, Y. D. Al-Otaibi, Y. Lu, S. Chen, and D. Zhang, Efficient and secure data sharing for 5G flying drones: A blockchain-enabled approach, IEEE Network, vol. 35, no. 1, pp. 130–137, 2021.

[19] S. Guo, X. Hu, S. Guo, X. Qiu, and F. Qi, Blockchain meets edge computing: A distributed and trusted authentication system, IEEE Trans. Ind. Inf., vol. 16, no. 3, pp. 1972–1983, 2020.

[20] A. Barnawi, S. Aggarwal, N. Kumar, D. M. Alghazzawi, B. Alzahrani, and M. Boulares, Path planning for energy management of smart maritime electric vehicles: A blockchain-based solution, IEEE Trans. Intell. Transp. Syst., vol. 24, no. 2, pp. 2282–2295, 2023.

[21] S. Garg, K. Kaur, G. Kaddoum, J. J. P. C. Rodrigues, and M. Guizani, Secure and lightweight authentication scheme for smart metering infrastructure in smart grid, IEEE Trans. Ind. Inf., vol. 16, no. 5, pp. 3548–3557, 2020.

[22] J. S. Shin, S. Lee, S. Choi, M. Jo, and S. H. Lee, A new distributed, decentralized privacy-preserving ID registration system, IEEE Commun. Mag., vol. 59, no. 6, pp. 138–144, 2021.

[23] J. Liu, Z. Zhang, X. Chen, and K. S. Kwak, Certificateless remote anonymous authentication schemes for wireless Body area networks, IEEE Trans. Parallel Distrib. Syst., vol. 25, no. 2, pp. 332–342, 2014.

[24] M. Kumar and S. Chand, A lightweight cloud-assisted identity-based anonymous authentication and key agreement protocol for secure wireless body area network, IEEE Syst. J., vol. 15, no. 2, pp. 2779–2786,

[25] 2021. S. Jegadeesan, M. Azees, N. R. Babu, U. Subramaniam, and J. D. Almakhles, EPAW: Efficient privacy preserving anonymous mutual authentication scheme for wireless body area networks (WBANs), IEEE Access, vol. 8, pp. 48576–48586, 2020.

[26] X. Jia, D. He, N. Kumar, and K. K. R. Choo, A provably secure and efficient identity-based anonymous authentication scheme for mobile edge computing, IEEE Syst. J., vol. 14, no. 1, pp. 560–571, 2020.

[27] K. Zarour, O. A. Bounab, Y. Marir, and I. Boumezbeur, Blockchain-based architecture centred patient for decentralised storage and secure sharing health data, Int. J. Electron. Healthcare., vol. 12, no. 2, pp. 170–190, 2022.

[28] H. Chai, S. Leng, J. He, K. Zhang, and B. Cheng, CyberChain: Cybertwin empowered blockchain for lightweight and privacy-preserving authentication in internet of vehicles, IEEE Trans. Veh. Technol., vol. 71, no. 5, pp. 4620–4631, 2022.

[29] J. Jayabalan and N. Jeyanthi, Scalable blockchain model using off-chain IPFS storage for healthcare data security and privacy, J. Parallel Distrib. Comput., vol. 164, pp. 152–167, 2022.

[30] S. Nakamoto, Bitcoin: A Peer-to-Peer Electronic Cash System, http://bitcoin.org/bitcoin.pdf,2008.

[31] Q. Feng, D. He, S. Zeadally, M. K. Khan, and N. Kumar, A survey on privacy protection in blockchain system, J. Network Comput. Appl., vol. 126, pp. 45–58, 2019.

[32] R. Gennaro, S. Jarecki, H. Krawczyk, and T. Rabin, Secure distributed key generation for discrete-log based cryptosystems, in Proc. Int. Conf. on the Theory and Applications of Cryptographic Techniques, Prague, Czech Republic, 1999, pp. 295−310.

[33] R. L. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, Commun. ACM, vol. 21, no. 2, pp. 120–126, 1978.

[34] Cygwin: Linux environment emulator for windows, http://www.cygwin.com/,2022.

[35] J. A. Fernandez-Carrasco, T. Egues-Arregui, F. Zola, and R. Orduna-Urrutia, ChronoEOS: Configuration control system based on EOSIO blockchain for on-running forensic analysis, in Proc. Int. Congress on Blockchain and Applications, L'Aquila, Italy, 2022, pp. 37−47.

GEETHANJALI INSTITUTE OF SCIENCE & TECHNOLOGY

(AUTONOMOUS)

(Approved by AICTE, New Delhi & Permanently Affiliated to JNTU, Anathapuram Accredited with 'A" Grade by NAAC, Accredited by NBA-B. Tech (MECH, EEE, ECE)
3rd Mile, Bombay Highway, Gangavaram (V), Kovur(M), Nellore Dt-524 137, A.P.

Certificate of Appreciation

This is to certify

Ponaka Kundana

has **PRESENTED** Paper entitled

Trustworthy Data Sharing via Multi-Attribute Based Authentication Mechanism

for ***International Conference on Latest Trends in Electronics Communication and AI Technologies (ICLTECAT-2025)*** organized by ***Geethanjali Institute of Science & Technology, Nellore*** on 2nd- 3rd April 2025.

Convenor | **Program Chair** | **Principal**

GEETHANJALI INSTITUTE OF SCIENCE & TECHNOLOGY

(AUTONOMOUS)

(Approved by AICTE, New Delhi & Permanently Affiliated to JNTU, Anathapuram Accredited with 'A" Grade by NAAC, Accredited by NBA-B. Tech (MECH, EEE, ECE)
3rd Mile, Bombay Highway, Gangavaram (V), Kovur(M), Nellore Dt-524 137, A.P.

Certificate of Appreciation

This is to certify

Pappu Sowmya

has **PRESENTED** Paper entitled

Trustworthy Data Sharing via Multi-Attribute Based Authentication Mechanism

for ***International Conference on Latest Trends in Electronics Communication and AI Technologies (ICLTECAT-2025)*** organized by ***Geethanjali Institute of Science & Technology, Nellore*** on 2nd- 3rd April 2025.

Convenor | **Program Chair** | **Principal**

GEETHANJALI INSTITUTE OF SCIENCE & TECHNOLOGY

(AUTONOMOUS)

(Approved by AICTE, New Delhi & Permanently Affiliated to JNTU, Anathapuram Accredited with 'A" Grade by NAAC, Accredited by NBA-B. Tech (MECH, EEE, ECE)
3rd Mile, Bombay Highway, Gangavaram (V), Kovur(M), Nellore Dt-524 137, A.P.

Certificate of Appreciation

This is to certify

Shaik Roshni

has **PRESENTED** Paper entitled

Trustworthy Data Sharing via Multi-Attribute Based Authentication Mechanism

for ***International Conference on Latest Trends in Electronics Communication and AI Technologies (ICLTECAT-2025)*** organized by ***Geethanjali Institute of Science & Technology, Nellore*** on 2nd- 3rd April 2025.

Convenor | **Program Chair** | **Principal**

GEETHANJALI INSTITUTE OF SCIENCE & TECHNOLOGY

(AUTONOMOUS)

(Approved by AICTE, New Delhi & Permanently Affiliated to JNTU, Anathapuram Accredited with 'A" Grade by NAAC, Accredited by NBA-B. Tech (MECH, EEE, ECE)
3rd Mile, Bombay Highway, Gangavaram (V), Kovur(M), Nellore Dt-524 137, A.P.

Certificate of Appreciation

This is to certify

Shaik Sofia

has **PRESENTED** Paper entitled

Trustworthy Data Sharing via Multi-Attribute Based Authentication Mechanism

for ***International Conference on Latest Trends in Electronics Communication and AI Technologies (ICLTECAT-2025)*** organized by ***Geethanjali Institute of Science & Technology, Nellore*** on 2nd- 3rd April 2025.

Convenor | **Program Chair** | **Principal**

GEETHANJALI INSTITUTE OF SCIENCE & TECHNOLOGY

(AUTONOMOUS)

(Approved by AICTE, New Delhi & Permanently Affiliated to JNTU, Anathapuram Accredited with 'A" Grade by NAAC, Accredited by NBA-B. Tech (MECH, EEE, ECE)
3rd Mile, Bombay Highway, Gangavaram (V), Kovur(M), Nellore Dt-524 137, A.P.

Certificate of Appreciation

This is to certify

Ch. Vasavi

has **PRESENTED** Paper entitled

Trustworthy Data Sharing via Multi-Attribute Based Authentication Mechanism

for ***International Conference on Latest Trends in Electronics Communication and AI Technologies (ICLTECAT-2025)*** organized by ***Geethanjali Institute of Science & Technology, Nellore*** on 2nd- 3rd April 2025.

Convenor | **Program Chair** | **Principal**

Department of Computer Science & Engineering

PROGRAM OUTCOMES (POs)

Engineering Graduates will be able to:

PO1 **Engineering knowledge:** Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.

PO2 **Problem analysis:** Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.

PO3 **Design/development of solutions:** Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.

PO4 **Conduct investigations of complex problems:** Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.

PO5 **Modern tool usage:** Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the

PO6 **The engineer and society:** Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.

PO7 **Environment and sustainability:** Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.

PO8 **Ethics:** Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.

PO9 **Individual and team work:** Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.

PO10 **Communication:** Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.

PO11 **Project management and finance:** Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.

PO12 **Life-long learning:** Recognize the need for, and have the preparation and ability to engage in independent and lifelong learning in the broadest context of technological change.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO1 Apply the expertise in adaptive algorithms to develop quality software applications.

PSO2 Demonstrate the capabilities in basic and advanced technologies towards getting employed or to become an entrepreneur.

Department of Computer Science & Engineering

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

Graduates of B.Tech in Computer Science and Engineering Programme shall be able to

PEO1 Outperform in professional career or higher learning by upgrading skills in Computer Science and Engineering stream.

PEO2 Provide computing solutions for complex problems to meet industry demands and societal needs.

PEO3 Offer ethical, socially sensitive solutions as professionals and as entrepreneurs in Computer Science and other engineering disciplines.

PEO4 Leverage new computing technologies by engaging themselves in perpetual learning.