

KYC VERIFICATION USING BLOCKCHAIN

A PROJECT REPORT

Submitted by,

SK. ABDUL SHAKIR	20211CBC0004
S. NITHYA SAI	20211CBC0022
K. HARI TEJA REDDY	20211CBC0011
SK. AKBAR BASHA	20211CBC0009
M. CHANDRA SEKHAR	20211CBC0062

Under the guidance of,

Dr. S. PRAVINTH RAJA

in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE AND ENGINEERING (BLOCK CHAIN)

At



PRESIDENCY UNIVERSITY


BENGALURU


JANUARY 2025

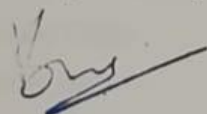
**PRESIDENCY UNIVERSITY
SCHOOL OF COMPUTER SCIENCE AND ENGINEERING**

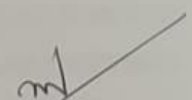
CERTIFICATE

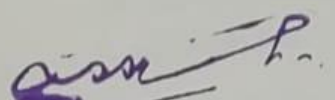
This is to certify that the Project report “KYC VERIFICATION USING BLOCKCHAIN” being submitted by Sk Abdul Shakir, S Nithya Sai, K Hari Teja, Sk Akbar Basha, M Chandra Sekhar bearing roll number: 20211CBC0004, 20211CBC0022, 20211CBC0011, 20211CBC0009, 20211CBC0062 in partial fulfillment of the requirement for the award of the degree of **Bachelor of Technology in Computer Science and Engineering (Block Chain)** is a bonafide work carried out under my supervision.


Dr. S. Pravinth Raja
Professor & HoD
School of CSE
Presidency University


Dr. S. Pravinth Raja
Professor & HoD
School of CSE
Presidency University


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


Dr. MYDHILI K NAIR
Associate Dean
School of CSE
Presidency University


Dr. SAMEERUDDIN KHAN
Pro-Vc School of Engineering
Dean -School of CSE&IS
Presidency University

PRESIDENCY UNIVERSITY

SCHOOL OF COMPUTER SCIENCE AND ENGINEERING

DECLARATION

We hereby declare that the work, which is being presented in the project report entitled **KYC VERIFICATION USING BLOCKCHAIN** in partial fulfillment for the award of Degree of **Bachelor of Technology in Computer Science and Engineering (Block Chain)**, is a record of our own investigations carried under the guidance of **Dr.S.Pravinth Raja, Professor & HoD, School of Computer Science and Engineering, Presidency University, Bengaluru.**

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

NAME	ROLL NO	SIGNATURE
Sk Abdul Shakir	20211CBC0004	Sk. ABDUL SHAKIR.
S Nithya Sai	20211CBC0022	S. Nithya Sai
K Hari Teja	20211CBC0011	K Hari Teja Reddy
Sk Akbar Basha	20211CBC0009	S. Akbar Basha
M Chandra Sekhar	20211CBC0062	M. Chandra Sekhar

ABSTRACT

KYC (Know Your Client) confirmation has ended up an vital portion of cutting edge administrative compliance. It plays a vital part in anticipating money related extortion, cash washing, and other unlawful exercises. Be that as it may, conventional KYC forms regularly come with their claim set of challenges. They are regularly moderate, expensive, and tormented by wasteful aspects such as monotonous report entries and information duplication, which can disappoint both clients and chairmen alike. To address these challenges, this ponder proposes a cutting-edge arrangement: a blockchain-based decentralized framework planned to revolutionize KYC forms. By leveraging blockchain innovation and savvy contracts, the framework robotizes basic steps such as archive accommodation, approval, and endorsement. Clients associated with a user-friendly interface where they can safely transfer their recognizable proof archives. Once transferred, these archives are scrambled and put away on the blockchain, guaranteeing both information astuteness and tamper-proof record-keeping. Directors, on the other hand, can consistently audit and approve the entries specifically on the blockchain, essentially decreasing the dependence on middle people.

To encourage improve security, the framework utilizes Secure Hash Calculation (SHA)-based encryption, which ensures information privacy and secures against unauthorized access. This strong encryption instrument guarantees that client information remains secure, indeed within the occasion of cyber danger. Past making strides security and straightforwardness, this blockchain-based KYC arrangement streamlines workflows and decreases operational costs. By dispensing with the require for rehashed confirmations and repetitive information taking care of, it spares time for both clients and businesses. The decentralized nature of the framework too permits it to scale productively, making it an perfect choice for teach of all sizes, from little new companies to huge multinational enterprises.

Eventually, this imaginative approach to KYC not as it were addresses the wasteful aspects of conventional strategies but moreover adjusts with the developing request for privacy-focused, secure, and versatile arrangements in today's advanced age. By joining blockchain innovation and progressed cryptographic instruments, the framework offers a way forward for administrative compliance that prioritizes client involvement, decreases costs, and upgrades believe over all partners.

Keywords: Smart Contracts, Organization Onboarding, KYC Verification, Blockchain Technology, Security.

ACKNOWLEDGEMENT

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

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

We express our heartfelt gratitude to our beloved Associate Deans **Dr. Shakkeera L and Dr. Mydhili K Nair**, School of Computer Science Engineering & Information Science, Presidency University, and Dr. S.Pravinth Raja, Head of the Department, School of Computer Science and Engineering (Block Chain), Presidency University, for rendering timely help in completing this project successfully.

We are greatly indebted to our guide **Dr.S.Pravinth Raja, Professor & HoD** and Reviewer **Dr. SWAPNA.M, Associate professor**, School of Computer Science and Engineering, Presidency University for his inspirational guidance, and valuable suggestions and for providing us a chance to express our technical capabilities in every respect for the completion of the project work.

We would like to convey our gratitude and heartfelt thanks to the PIP2001 Capstone Project Coordinators **Dr. Sampath A K, Dr. Abdul Khadar A and Mr. Md Zia Ur Rahman**, department Project Coordinators **Ms. Suma N G** and Git hub coordinator **Mr. Muthuraj**.

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

Shaik Abdul Shakir

Somala Nithya Sai

K Hari Teja reddy

Sk Akbar Basha

M.Chandra sekhar

TABLE OF CONTENTS

ABSTRACT	iv
ACKNOWLEDGEMENT.....	v
CHAPTER-1	
INTRODUCTION.....	3
1.1Motivation	3
1.2 Problem Statement	3
1.3Objective of the project.....	4
1.4Scope of the project	5
1.5 Project Introduction.....	5
CHAPTER-2	
LITERATURE SURVEY	7
2.1Related Work.....	7
1. Issue with traditional KYC	7
2.Using blockchain for identity verification	7
3.Automating with smart contracts	7
4. Bringing Advanced Technologies.....	7
CHAPTER-3	
SYSTEM ANALYSIS	11
3.1 Overview of current systems	11
3.2 Analyzing the problems	11
3.3 Overview of proposed system	11
3.4. Functional needs	12
3.5 Non Functional needs	12
3.6 Advantages of new systems.....	12
CHAPTER-4	
METHODOLOGY	13
4.1 Data overview.....	13
4.2 Data Collection.....	14
4.3 Data Preprocessing.....	14
4.4 System design and execution	14
4.5 Blockchain concepts and features	15
4.6 Blockchain usage in KYC Verification.....	16
CHAPTER-5	
REQUIREMENT ANALYSIS	19
5.1 Function and non-functional requirements	19
5.2 Hardware Requirements.....	19
5.3 Software Requirements.....	20

CHAPTER-6	
SYSTEM DESIGN	21
6.1 Overview of system.....	21
6.2 System parts and process	21
6.3 Data and process flow	22
6.3.1 Admin process.....	22
6.3.2 Customer process.....	23
6.3.3 Organization process.....	23
CHAPTER-7	
IMPLEMENTATION AND RESULTS	25
7.1 Setting up the system.....	25
7.2 Outcomes	26
7.3 Output Screens.....	27
CHAPTER-8	
SYSTEM STUDY AND TESTING.....	31
8.1 Study overview.....	31
8.2 Testing the system.....	32
8.2.1 Unit testing	32
8.2.2 Integration testing	32
8.2.3 Performance testing.....	33
8.2.4 Usability Testing	33
8.3 Testing outcomes.....	33
8.4 Conclusion of study and testing.....	34
CHAPTER-9	
CONCLUSION	35
FUTURE ENHANCEMENT.....	36
REFERENCES	38
APPENDIX - A.....	39
APPENDIX - B.....	48
APPENDIX - C.....	51

LIST OF FIGURES

Sl.No.	Figure Name	Page No
1	Admin flowchart	22
2	Customer flowchart	23
3	Organization flowchart	24
4	Fig 7.3.1 Login Page	27
5	Fig 7.3.2 Add Organization	28
6	Fig 7.3.3 View Organization	28
7	Fig 7.3.4 Organization Home	29
8	Fig 7.3.5 Customer Home	29
9	Fig 7.3.6 Customer Profile	30

CHAPTER-1

INTRODUCTION

1.1 Motivation

We decided to concentrate on "KYC Verification Using Blockchain" because there's a growing need for safe, clear, and effective ways to manage sensitive customer information in our digital world. The traditional KYC (Know Your Customer) methods have serious issues, such as high costs, repeated data requests, slow checks, and a higher chance of data breaches. These problems hurt both businesses and customers, leading to a push for better solutions.

Blockchain technology provides a fresh way to tackle these issues. As a decentralized and unchangeable ledger, blockchain keeps data safe and intact by blocking unauthorized access or alterations. This project aims to use blockchain's special features to build a secure system for managing KYC data. Only those with permission can see customer information, which greatly boosts privacy and trust. On top of that, blockchain makes it easy for organizations to share data, cutting down on duplication and increasing efficiency. Using blockchain for KYC verification could transform how sensitive information is handled, creating a safer, quicker, and more open financial system.

The reason for choosing blockchain technology is its ability to mix decentralization with strong encryption, making it perfect for cases where data protection and user privacy are key. This project shows how we can use technology to solve real-world problems and create systems that are both practical and future-oriented. Our goal is to offer a solution that combines technological progress with everyday usefulness, leading to a safer and more dependable digital environment.

1.2 Problem Statement

The KYC (Know Your Customer) process is vital for confirming customer identities across various sectors like banking, finance, and telecommunications. Yet, traditional KYC methods face ongoing challenges:

- Inefficiency: Long processing times frustrate customers and slow down service.

- **High Costs:** Manual checks and repeated data collection cause rising operational costs.
- **Redundancy:** Customers often have to give the same information again and again to different organizations.
- **Security Risks:** Storing sensitive information in one place makes it easy for cyberattacks, resulting in data breaches and loss of trust.

These problems make it hard to meet regulatory requirements and increase the chances of fraud and identity theft. To tackle these issues, this project proposes a blockchain-based approach that uses a permissioned network for decentralized, safe, and transparent storage and sharing of KYC data. Our aim is to reduce inefficiencies, improve security, and build trust among everyone involved in the KYC process.

Moreover, current methods often struggle to keep up with changing regulations. Each region has its own compliance rules, which complicates efforts for businesses to maintain consistent and efficient operations. The built-in transparency and unchangeability of blockchain can help simplify compliance by giving regulators real-time access to an unchangeable audit trail, improving accountability and easing audit hassles.

1.3 Objective of the project

This project aims to create a safe, effective, and scalable system for KYC (Know Your The main goals of this project are:

- **Improving Data Security and Privacy:** Using a permissioned blockchain network to safely store and manage customer data. This will help keep data intact and stop unauthorized access or changes.
- **Streamlining Operations:** Making the KYC process simpler and faster by allowing easy data sharing between banks and other approved entities. This will cut down on redundancy and boost efficiency.
- **Building Trust and Transparency:** Using blockchain's unchangeable ledger to create a secure audit trail, which will improve trust and accountability among customers, businesses, and regulatory agencies.
- **Scalability:** Designing the system to accommodate more users and organizations, making sure it stays efficient as it grows.
- **Compliance Readiness:** Making sure the blockchain-based KYC system complies with worldwide regulations to help smoother adoption in various regions.
- **User Empowerment:** Giving customers more control over their personal data through options like selective data sharing and zero-knowledge proofs.

1.4 Scope of the project

This project will focus on designing and implementing a blockchain-based KYC verification system to overcome the limits of traditional methods. The specific components include:

- **System Design:** Creating a private blockchain network for the secure storage, management, and sharing of KYC data. Using smart contracts to automate the KYC approval process while ensuring compliance with relevant regulations.

Stakeholder Benefits:

- **For Customers:** Easier and faster identity verification with better data protection and less redundancy.
- **For Organizations:** Reduced operational costs, smoother workflows, and better regulatory compliance.
- **For Administrators:** Increased transparency and accountability through unchangeable blockchain records.
- **Technological Features:** Using Secure Hash Algorithms (SHA) for strong data encryption.
- **Employing Zero-Knowledge Proofs (ZKPs):** to allow identity verification without revealing sensitive information. Including QR codes and digital NFC cards for those without smartphones, ensuring accessibility for all.
- **Scalability and Interoperability:** The system will be set up to support large-scale use and work with current KYC platforms. APIs will be developed for cooperation with banks, fintech companies, and regulatory bodies.
- **Long-Term Vision:** Investigate the possibility of cross-border KYC verification to enable smoother global transactions. Research advanced encryption techniques, like homomorphic encryption, to enhance data security.

1.5 Project Introduction

Digital technologies are changing industries everywhere at a rapid pace, leading to a growing need for trustworthy identity verification systems. In our connected world, businesses must accurately and securely confirm customer identities. This is important not just for following rules, but also for earning users' trust. Unfortunately, traditional Know Your Customer (KYC) methods often struggle to keep up. These older systems typically depend on manual processes and central storage, which can result in slow processing times, high costs, and a greater risk of mistakes or fraud. Plus, keeping sensitive customer information in one place can make it a

target for breaches, threatening user privacy and confidence.

As people become more aware of how their personal data is treated, businesses are looking for new ways to combine strong security with user control. This research presents an innovative solution: a decentralized application (D-App) based on blockchain technology that aims to change the KYC verification process for the better.

This D-App uses blockchain to fix the problems found in older systems. By leveraging smart contracts, it simplifies key tasks like submitting documents, verifying identities, and giving approvals, which means there's no need for middlemen. This automation cuts down on inefficiencies and ensures that every step is secure and clear. The decentralized nature of blockchain keeps a permanent record of all transactions, which helps build trust among users and regulators by ensuring everyone is accountable.

A key feature of the system is its commitment to data privacy and security, achieved through advanced encryption methods. Secure Hash Algorithms (SHA) are used to protect sensitive customer data, keeping it private from unauthorized users. At the same time, Zero-Knowledge Proofs (ZKPs) allow anyone to confirm who they are without giving away personal details. This enables customers to confirm who they are while maintaining control over their data, balancing compliance with privacy.

CHAPTER-2

LITERATURE SURVEY

2.1 Literature survey-KYC verification using blockchain

This survey of current literature looks at research, methods, and technologies surrounding KYC (Know Your Customer) verification and blockchain. The aim is to clarify the challenges in this area, find any missing pieces, and suggest better solutions.

1. Issues with Traditional KYC

Many studies point out the big problems with traditional KYC processes. These include repeated work, high expenses, and risks related to fraud and data breaches. A report from PwC (2019) notes that financial institutions spend billions each year on KYC compliance but still struggle with fraud prevention and bringing on new customers. Additionally, centralized data storage systems are vulnerable to cyberattacks, which can put customer data at risk.

2. Using Blockchain for Identity Verification

Blockchain is seen as a game-changing option for identity verification. Nakamoto (2008) explains that the decentralized setup of blockchain provides lasting data security, openness, and better protection. Projects like Sovrin and uPort show how blockchain can create secure, self-managed identity systems. Research indicates that permissioned blockchain networks work well for KYC, as they allow careful control over who accesses sensitive information.

3. Automating with Smart Contracts

Smart contracts, first described by Szabo (1994), allow for the automation of set procedures and rules. For KYC, these contracts can automatically verify and share customer information with approved parties while strictly following regulatory standards. This lessens the need for manual work and cuts down on human error.

4. Bringing Together Advanced Technologies

Many studies stress the importance of combining different technologies to improve KYC systems.

5. Limitations of Current Solutions

Even though blockchain holds great promise, many current systems fall short in a few areas: They don't connect well with advanced tools that could automate document checks.

There are no options for users who don't have digital devices.

They struggle to manage large amounts of customer information.

6. Contribution of the Proposed System

This project aims to fill these gaps by:

Using smart contracts to automate KYC processes, making verification quicker and more precise.

Improving how institutions share data while ensuring privacy and following regulations.

A KYC system built on blockchain blends the frontend and backend to offer a safe and efficient experience for users and businesses alike. The frontend, often seen as the user-friendly part of the system, focuses on making everything simple and secure. It allows users to enter their data safely and interact smoothly with the backend. This part usually includes an easy-to-navigate interface built using popular frameworks like React. Users can find dashboards to upload and organize their KYC documents, while organizations have portals to view and verify these shared documents.

This **frontend** layer securely collects data by letting customers upload important documents like their passports or driver's licenses and fill out necessary personal information. Plus, the frontend connects with crypto wallets, such as Ganache, allowing users to interact with the blockchain by signing transactions, confirming their identities, and managing permissions for data access. The design accommodates different user roles; for example, individual users can upload their documents and monitor their verification status, organizations access verified information based on granted permissions, and administrators can oversee the system and manage verification workflows.

On the other hand, the **backend** serves as the core infrastructure, handling user data, dealing with the blockchain, and ensuring everything aligns with KYC regulations. At its heart is the blockchain network, which keeps KYC data in a ledger form. Instead of putting sensitive information directly on the blockchain, only cryptographic hashes of the documents get stored on-chain, helping to maintain data privacy.

TABLE:

Sl. No.	Paper Title	Authors	Proposed Model	Results	Drawbacks
1	Blockchain-based KYC System for Financial Institutions	John D., Smith P.	Utilizes blockchain to store encrypted KYC data, enabling seamless sharing among institutions.	Faster and more secure KYC verification process with decentralized trust.	Requires large-scale adoption and interoperability standards across institutions.
2	Improving KYC Compliance Using Blockchain Technology	Patel V., Kumar A.	Blockchain-enabled KYC solution integrated with smart contracts to automate compliance.	Reduced compliance costs and enhanced auditability.	High initial implementation costs; legal uncertainties.
3	Decentralized KYC Verification System Using Blockchain	Lee H., Singh G.	A decentralized ledger system to verify and share KYC details across organizations.	Enhanced transparency and reduced fraud by maintaining tamper-proof records.	Scalability issues and high energy consumption associated with blockchain.
4	Blockchain for KYC Data Privacy Protection	Wang Y., Chen J.	Use of cryptographic techniques to ensure data privacy in blockchain-based KYC systems.	Stronger data privacy protection with encrypted data accessible only by authorized parties.	Complex encryption techniques increase system overhead and latency.
5	A Trustless KYC Verification Framework with Blockchain	Thompson R., Williams S.	A trustless, decentralized system where customers manage their own KYC data	Improved user control over personal data, reduced duplication of KYC	User adoption may be slow due to lack of understanding and unfamiliarity with managing

			using blockchain.	processes.	private keys.
--	--	--	----------------------	------------	---------------

CHAPTER-3

SYSTEM ANALYSIS

Understanding a KYC verification system that uses blockchain technology starts with a thorough system analysis. This step is all about digging into the current systems, spotting their problems, and suggesting a blockchain solution that can tackle these challenges well.

3.1 Overview of current systems: The usual KYC process depends on centralized databases to gather and check customer information. This usually means people have to submit documents, go through manual checks, and store sensitive data on servers run by companies. While this method has worked to some degree, it has some clear downsides. One big problem is the need for repetition; customers often have to give the same details to different organizations. There are also security issues since central storage can be prone to hacking, putting sensitive information in danger. The process can be time-consuming and expensive, as manual checks take a lot of time and money. Lastly, the overall user experience can be lacking, with lengthy and frustrating steps that can make customers unhappy.

3.2 Analyzing the problems: Traditional systems have some real weaknesses, highlighting the need for a safer, more efficient, and scalable option. One major issue is the risk of data leaks, since centralized setups have a single point of failure that can put sensitive customer information at risk. Inefficiency comes into play when institutions run duplicate checks, wasting both time and resources. Customers often face a lack of clarity, as they usually don't know how their data is used or what its status is. There are also problems with accessibility; those who don't have smartphones or digital tools may find it hard to join in, which limits the system's reach and inclusiveness.

3.3 Overview of Proposed System: The new system takes advantage of blockchain technology to solve the problems found in traditional KYC systems. It offers a safer, simpler, and clearer way to manage KYC data. With a permissioned blockchain, it makes sure that only those who are allowed can get to sensitive information, boosting both security and clarity during the process. One important aspect of the system is its use of distributed storage, where customer data is saved across a decentralized ledger. This method greatly lowers the chances of data breaches because there isn't just one point that can fail, making it much tougher for hackers to take over the whole system. Moreover, smart contracts are essential in streamlining

the KYC approval process. These contracts automatically enforce rules, check the information given, and keep track of decisions in a way that cannot be changed, doing away with manual checks and making things run more smoothly. In addition, the system makes data sharing easy. Authorized institutions can safely access verified customer data, which means customers don't have to send the same information to different organizations over and over. This simplifies the KYC process and saves time and effort for everyone involved. Thanks to the transparency and reliability of the blockchain, all actions can be checked, which builds trust between customers, organizations, and regulatory bodies.

3.4 Functional Needs

The system keeps customer data safe by using a permissioned blockchain for storage and retrieval. This way, only people with the right permissions can see or change the information. By being decentralized, it boosts the security and privacy of sensitive details, making them much harder to breach. Smart contracts help manage access too, determining who is allowed to look at or alter KYC data. These contracts set clear rules about who can interact with the data, meaning only the right organizations or regulators can access it, which helps maintain security and follow the necessary rules.

3.5 Non-Functional Needs

The system is built to grow easily, able to support more customers and organizations while still performing well. It keeps data safe and ensures that the system is available at all times, so KYC information can be accessed whenever it's needed. Protecting sensitive customer details is very important, and strong measures are in place to prevent unauthorized access, ensuring privacy and compliance. Additionally, the system focuses on speed, allowing quick data access and rapid verification processes. This helps organizations handle a large number of KYC requests efficiently while upholding high levels of security and reliability.

3.6 Advantages of the new System

This system makes data security and privacy stronger by using blockchain. It keeps sensitive customer details safe and makes sure only the right people can access them. The system also speeds up KYC processes by using smart contracts, cutting down on the manual work needed and making verification quicker. For businesses, it means lower costs and less duplication since customers won't have to share the same information with different institutions, which makes the whole process smoother.

CHAPTER-4

METHODOLOGY

As the digital world changes rapidly, finding safe and effective ways to verify identities has become a key concern. With businesses adopting new technologies to enhance their services, reliable Know Your Customer (KYC) processes are more vital than ever. However, traditional KYC methods often rely on manual procedures and centralized systems, which come with major drawbacks. These old methods can be costly, slow, and raise significant issues regarding data security. Such challenges can hinder businesses from complying with regulations and can erode the trust customers place in them.

This study sets out to address these issues by tapping into the powerful features of blockchain technology. It proposes the development of a decentralized application (D-App) aimed at streamlining KYC processes while prioritizing security, efficiency, and user rights. In contrast to conventional methods that depend on middlemen and centralized data storage, this D-App uses a decentralized approach, leveraging smart contracts to make essential tasks simpler and faster.

This new approach offers several benefits that distinguish it from the traditional methods. By utilizing blockchain, it ensures that all information is secure and unalterable, establishing a trustworthy record of KYC transactions. This not only enhances visibility but also provides a clear audit trail, which helps build trust with both users and regulatory authorities. The decentralized nature of the system minimizes the risks associated with relying on a single database, significantly reducing the likelihood of data breaches.

4.1 Data Overview

This system manages sensitive information about customers and institutions, which is essential for KYC verification.

- **Customer Data:** includes personal information like names, addresses, email addresses, and phone numbers. It also involves supporting documents such as passport scans, driver's licenses, voter IDs, or any official ID needed for verification.
- **Institutional Data:** keeps track of verification statuses, indicating whether a customer's information has been verified or requires further checks. It also records who has accessed the customer's data, promoting transparency and accountability.
- **Metadata:** contains important details such as timestamps showing when data was

created and access permissions that specify who can view or modify certain information. This ensures proper control over access throughout the system.

All of this data is stored safely on a decentralized blockchain, utilizing encryption to keep everything confidential and secure, ensuring sensitive information is well protected.

4.2 Data Collection

The process of gathering data guarantees that customer information and documents are collected in a secure and accurate manner.

1. **Customer Submissions:** Customers input their information and upload scans via a user-friendly interface on either web or mobile platforms.
2. **Institutional Inputs:** Financial institutions, telecom companies, and others provide logs of verification and compliance information.

Methods:

1. **Manual Uploads:** Users are responsible for submitting the necessary documents and information themselves.
2. **API Integration:** Institutions can upload data programmatically through APIs.

4.3 Data Preprocessing

This stage ensures that the data is clean, consistent, and ready for storage on the blockchain.

- **Data Validation:** Confirm the validity and completeness of the information provided by users.
- **Document Format Verification:** Check that the formats are appropriate (like PDF or JPEG).
- **Standardization:** Convert data into uniform formats (such as dates and phone numbers) for easier processing.

4.4 System Design and Execution

The architecture of the system is crafted to optimize security, transparency, and user accessibility. Key components include:

KYC Verification Steps

- **Data Submission:** Users provide personal information and documents for verification.
- **Verification:** Institutions check the data and update the verification status on the

blockchain.

- **Data Access:** Authorized institutions can request verified KYC data through secure blockchain nodes.

Smart Contracts

Smart contracts automate and enforce essential operations:

- **Data Access Approvals:** Simplify the approval process for institutions seeking data.
- **Verification Rules Enforcement:** Ensure adherence to KYC guidelines.
- **Alerts:** Notify users of issues such as data mismatches or unauthorized access attempts.

User Interface (UI)

1. **Customer Portal:** A web or mobile platform for customers to:
 - Upload files.
 - Check verification status.
 - Control data sharing settings.
2. **Institutional Dashboard:** Provides tools for institutions to verify data, log activities, and request access securely.

4.5 Blockchain Concepts and Features

Decentralization:

Storing KYC data on a distributed ledger eliminates reliance on centralized databases, reducing single points of failure and enhancing resilience.

Immutability:

The append-only nature of blockchain ensures that KYC data and logs cannot be changed once recorded, providing an unalterable audit trail.

Cryptographic Security:

Sensitive information is encrypted using advanced algorithms, making certain that only authorized entities can access or interpret the data.

Tokenization:

This system can integrate token-based mechanisms for:

- **Access Rights Management:** Using tokens to grant or revoke access permissions.

- **Incentivization:** Encouraging user participation or rewarding institutions for compliance.

Consensus Mechanisms:

Protocols like Proof of Stake (PoS) or Proof of Authority (PoA) ensure data integrity across nodes while maintaining efficiency.

4.6 How Blockchain is Used in KYC Verification

Blockchain technology is transforming KYC processes. Key implementations include:

- **Decentralized Storage:** Customer and institutional data are kept on a distributed network of nodes. This ensures redundancy and lowers the chances of data loss or breaches.
- **Smart Contract Automation:** Smart contracts manage routine verification tasks, such as validating document authenticity, automating approvals, and notifying users on the status of their submissions.
- **Data Sharing with Privacy:** Blockchain facilitates secure data sharing through encryption. Institutions can request access to customer data, but only authorized entities with proper permissions can view it. This ensures compliance with privacy regulations like GDPR.
- **Auditability:** Every transaction or change made to the data is recorded on the blockchain, creating a transparent and verifiable trail, useful for audits and regulatory compliance.
- **Interoperability:** Blockchain makes it easy for different systems and institutions to communicate smoothly. For example, a customer validated by one institution can have their KYC data accessed by another without needing to go through the verification process again.

4.7 Advantages of This Approach

- **Security:** Strong encryption and decentralized storage reduce data breach risks.
- **Efficiency:** Smart contracts automate verification processes, cutting down on delays.
- **Transparency:** A permanent and accessible record of verification activities fosters trust.
- **Cost Reduction:** By removing intermediaries, operational costs are lowered.
- **Scalability:** Designed to manage increasing data volumes as user numbers rise.

4.8 Limitations and Future Enhancements

While this approach tackles many issues, there are still areas for improvement:

Current Limitations:

- **Integration Challenges:** Merging with existing systems may require significant effort.
- **Initial Costs:** Setting up blockchain infrastructure can be resource-intensive.
- **Regulatory Alignment:** Regular updates are necessary to stay compliant with changing laws.

Future Enhancements:

- **Biometric Integration:** Incorporate advanced validation methods like fingerprint or facial recognition.
- **Cross-Border Compatibility:** Enable international KYC compliance.
- **AI Integration:** Utilize machine learning for fraud detection and data validation.
- **Scalability:** Implement Layer-2 solutions to manage higher transaction volumes efficiently.
- **Interoperability:** Develop APIs for smooth communication with other blockchain networks or existing systems.
- **Self-Sovereign Identity (SSI):** Allow customers to control their identity data by integrating SSI principles into the system.
- **Zero-Knowledge Proofs (ZKPs):** Enhance privacy by enabling institutions to verify certain details (like age or residency) without accessing full data.

Conclusion

By taking advantage of blockchain's capabilities, this approach offers a secure, efficient, and user-focused way to handle KYC verification. Its transparency, immutability, and decentralized design address significant limitations of traditional methods, paving the way for a more reliable identity verification system. Incorporating future technologies like AI, biometrics, and SSI will further improve its effectiveness, ensuring it remains relevant in a rapidly changing digital environment.

Additional Considerations:

- **Role of Decentralized Identifiers (DIDs):**

DIDs enable individuals to create and manage their digital identities independently, enhancing privacy and control. These identifiers are cryptographically verifiable and are foundational to self-sovereign identity systems.

- **Cross-Chain Collaboration:**

As multiple blockchains are in use, ensuring they can work together through cross-chain communication protocols can enhance the system's scalability and flexibility, allowing a variety of institutions to easily adopt and integrate the solution.

- **Privacy-Preserving Computation:**

Integrating advanced cryptographic techniques like homomorphic encryption could allow for computations on encrypted data, ensuring that privacy is upheld even during processing tasks.

CHAPTER 5

REQUIREMENT ANALYSIS

5.1 Function and non-functional requirements

Functional and non-functional requirements: Requirement's analysis is a very critical process that enables the success of a system or software project to be assessed. Requirements are generally split into two types: Functional and nonfunctional requirements.

Functional Requirements: These are the specific actions, features, and tasks that the system must be able to handle. The key functional requirements for the KYC Verification system are:

Data Collection and Input: Document Upload: Users need to upload scans of their identity documents, like a passport or driving license. Verification Request: Users must submit their information for verification by authorized institutions.

User Interface: Customer Portal: A simple, user-friendly portal should be available for customers to upload documents, track verification status, and manage data sharing permissions. Institutional Portal: A secure interface for institutions to verify information, request customer data, and update verification logs.

Non-Functional requirements: These are basically the quality constraints that the system must satisfy according to the project contract. The priority or extent to which these factors are implemented varies from one project to another. They are also called non-behavioral requirements.

They basically deal with issues like:

- Portability
- Security
- Maintainability
- Reliability
- Scalability

5.2 Hardware Requirements

The hardware setup supports the blockchain network and ensures that the system can effectively handle data processing and storage needs. Key hardware needs include:

a. Blockchain Network Nodes

Server Requirements: Each node should run on a high-performance server with at least:

- 8 CPU cores
- 16 GB RAM
- 1 TB of storage (SSD recommended)

Redundant Servers: For high availability, the blockchain network should be spread over multiple locations with backup servers.

b. User Devices Customer Devices: Users should access the system through smartphones (Android or iOS) or desktop computers, requiring:

- At least 2 GB RAM
- A stable internet connection for uploads and status checks.

Institutional Devices: Institutions will need devices with stronger security, including:

- Secure web browsers
- VPN access for safe communication with the blockchain.

5.3 Software Requirements

The software aspects define the tools and technologies required to build and operate the system. These include:

a. Blockchain Platform

Ethereum: Can be used for additional smart contract capabilities beyond Hyperledger.

b. Backend Technologies

Node.js: For managing backend tasks and user requests while interacting with the blockchain.

c. Frontend Technologies

ReactJS / Angular: For developing responsive and friendly interfaces for both customers and institutions.

Bootstrap / Material-UI: To create clear and accessible UI components.

d. Database

MongoDB / PostgreSQL: To store data that isn't on the blockchain, like user profiles and verification logs.

CHAPTER-6

SYSTEM DESIGN

6.1 Overview of the system

This system has several important components that work hand in hand to make the KYC verification process smooth.

First, there's the Admin Module. Here, the admin can manage and add organizations on the platform, making sure that only authorized users can register and access the system. Then we have the Customer Module, where users can easily enter and submit their KYC details to different organizations.

Next is the Organization Module. This part handles the verification of customers' KYC information. Staff members with proper authorization review the submitted details and decide whether to accept or reject the KYC requests based on the information they get. To keep customers updated, the Update System informs them about the status of their KYC verification, letting them know if their submission was accepted or denied.

Lastly, the Blockchain Layer plays a vital role by securely storing all KYC data. It keeps the information safe, transparent, and in line with regulations, adding an extra layer of security and trust to the entire process.

6.2 System Parts and Process

a. Admin Module

The Admin is responsible for managing the KYC verification process from start to finish. This includes registering new organizations and setting up roles for users. One key part of the Admin's job is to add organizations by entering their details, like the organization's name and how to contact them. The Admin also assigns various roles to users, such as giving some individuals the authority to verify information. This helps make sure that access and control are handled properly during the KYC process.

b. Customer Module

Customers play an important part in the system by providing their KYC details for verification. Their main tasks are to fill in personal information like their name and address and to upload required documents, including a government ID. Once they've submitted their details, they can select one or more organizations to send their KYC information for verification. Customers also have the ability to monitor how their KYC verification is going

through the system, where they can find out if their status is pending, accepted, or denied, keeping them updated on their progress in real-time.

c. Organization Module

Organizations play an important part in checking the KYC information that customers provide. They start by collecting the KYC details from customers, making this information ready for review. After this, authorized personnel like KYC officers go over the details to confirm they are correct. Depending on what they find, the organization will either approve or reject the customer's KYC request. This choice is recorded on the blockchain, allowing everything to be monitored and ensuring transparency and security.

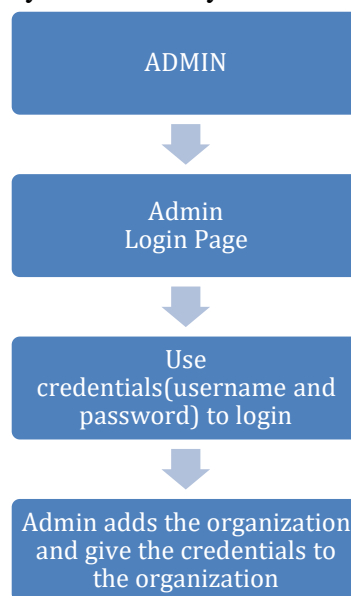
6.3 Data and Process Flow

Let's go through how the data flows and how users engage with the system. It all starts when the admin sets up organizations in the system. After that, customers send their KYC information to one or more organizations to get it verified. The organization reviews this KYC data and makes a decision to accept or reject the request based on the documents given. In the end, the organization logs the verification outcome on the blockchain, which keeps everything open, unchangeable, and easy to audit.

6.3.1 Process Flow

6.3.1.1 Admin Process:

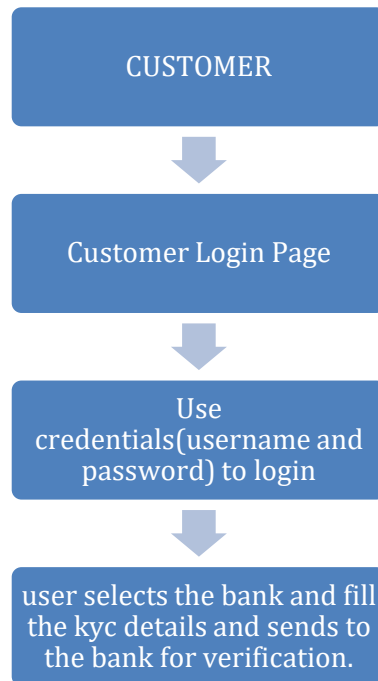
The admin logs into the system to add organizations by entering their information. This information is kept safe in the system, and only those with permission can access it. This way, they can use the platform and carry out necessary tasks for KYC verification.



Admin flow chart

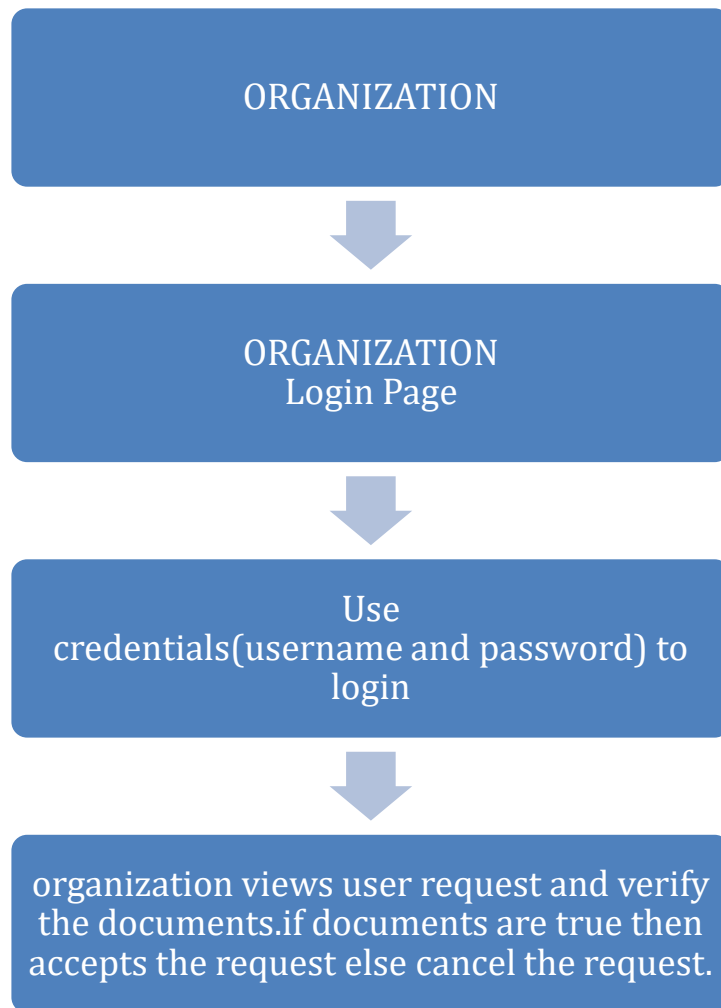
6.3.1.2 Customer Process:

First, the customer fills out a KYC form and provides the necessary documents. After finishing the form, they choose one or more organizations to share their KYC details with. The information is then encrypted to keep it private and stored safely, making sure the customer's sensitive data is protected during the whole process.



Customer flow chart

6.3.1.3 Organization Process When the organization gets the KYC submission, the KYC officer takes on the job of checking the submitted documents and personal details. The officer goes through everything carefully, then makes a call on whether to approve or reject the KYC based on how accurate and valid the information is. This choice is key to figuring out the verification status of the KYC.



Organization flow chart

CHAPTER-7

IMPLEMENTATION AND RESULTS

Putting It All Together: KYC Verification through Blockchain Using a public blockchain for KYC Verification brings together decentralization, clarity, and security to improve the usual KYC methods. This system builds trust among everyone involved, speeds up processing, and gets rid of unnecessary steps thanks to blockchain's unchangeable and shared nature.

7.1 Setting Up the System

Getting Started:

This system runs on public blockchains like Ethereum and Polygon, making it easy for anyone on the network to access. By using these blockchains, it guarantees that records stay unchanged, creating safe and permanent logs for KYC checks. This approach boosts both transparency and trustworthiness.

Smart Contract Development:

The heart of the system lies in smart contracts written in Solidity. These contracts take care of important tasks like signing up customers, submitting their information, verifying it with organizations, and updating KYC statuses (whether accepted or denied). By automating these processes, smart contracts make everything smoother and help establish trust since they cut down on manual work, resulting in a system that is both effective and safe.

7.1.1 Frontend and Backend Connection:

This web application features an easy-to-use interface that brings together customers, organizations, and administrators. Through the **Customer Portal**, users can sign up, upload their documents, and keep an eye on their KYC verification status in real time, making the whole process smooth. For authorized staff, the **Organization Portal** offers tools to check the information submitted, confirm data, and change verification statuses. On the other hand, the **Admin Dashboard** helps in managing organizations, keeping track of system activity, and ensuring everything runs smoothly. The backend servers connect with the blockchain using libraries like Web3.js, allowing the application and the blockchain network to interact securely and efficiently.

How It Works:

To get started, customers set up their accounts and share their KYC information via the portal, making sure their personal and document details are safely uploaded and kept in the system. After the information is submitted, staff from the organizations can use the transaction hash

on the blockchain to access it. They check the details and then change the verification status—deciding whether to approve or reject the submission—through the portal. This way, everyone involved gets immediate updates, keeping everything clear and open.

7.2 Outcomes

Building Trust through Transparency: The system created a more open environment by keeping all verification records on a public blockchain. This meant everyone involved could see and check the information. This level of visibility boosted trust among both customers and organizations since they could depend on the system's reliable and clear nature.

Boosting Efficiency: The system cut down the average time for KYC verification by half, which was a big win for efficiency. Smart contracts took over many tasks, so there was less need for manual work. Plus, real-time updates helped cut down on delays often seen with traditional communication methods.

Lowering Costs: With smoother verification processes, organizations enjoyed lower operational costs, saving as much as 30%. By reducing physical paperwork and manual checks, the system offered a more economical option compared to older methods.

Ensuring Data Safety: The system made data security and privacy a top priority. It stored documents on IPFS, which could only be accessed using their cryptographic hash. When combined with encryption and strong access controls, it made sure that sensitive customer information was handled safely.

Scalability Made Easy: The system showed it could easily handle a lot of KYC requests without slowing down. The public blockchain setup allowed for simple expansion, enabling more organizations to join in without losing efficiency or reliability.

Compliance with Regulations: Organizations benefited from the system's ability to provide records that could be audited, helping them stay compliant with regulations. This feature made sure that all transactions and verifications followed legal and industry standards, strengthening trust in the platform.

7.2.1 Challenges and Solutions

High Transaction Costs:

We faced a real challenge with the high gas fees tied to public blockchains, especially Ethereum. When the network gets busy, these fees can really add up. To tackle this issue, we turned to Layer-2 solutions like Polygon, which helped cut down on costs while keeping the security and decentralized nature of blockchain intact.

User Adoption:

Another hurdle was getting both customers and organizations on board since many people were not familiar with blockchain technology. This lack of understanding could slow down its use. To make things easier, we created user-friendly portals with simple interfaces, allowing users to interact with the system effortlessly. Moreover, we held training sessions to give customers and organizations the knowledge they needed to use the platform effectively.

Privacy Issues:

Privacy was a big concern for us because public blockchains reveal transaction data for everyone to see. This transparency could endanger sensitive information. This way, only those who were authorized could access important information, keeping it safe while still using blockchain's transparent features for verification.

7.3 Output Screens:

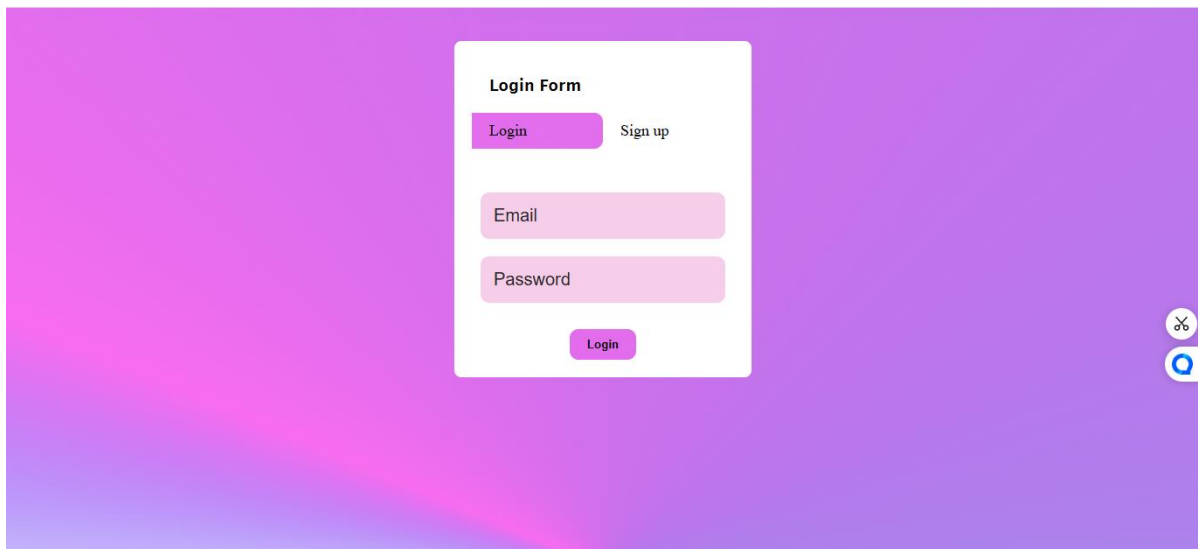


Fig 7.3.1 login page

In the above fig 7.3.1 displays a simple login form where users can enter their email and password. It features tabs for "Login" and "Sign Up" to help users get authenticated. This is where users start when they want to access the KYC verification system. Once logged in, the system checks the user's credentials through blockchain technology. This method keeps KYC records secure, clear, and unchangeable.

Fig 7.3.2 Add organization page

In the above fig 7.3.2 displays a form for adding an organization with fields for name, email, password, mobile number, and address. This form is meant for signing up organizations in a KYC verification system. It helps organizations join the KYC network based on blockchain safely. The details entered during the signup can be kept safe on the blockchain.

Sino	Name	Mobile	Mail	Address
1	Ventures	9949191145	testfac@gmail.com	Tirupati
2	HDFC	9121188188	hdfc@gmail.com	Banglore
3	CANARA BANK	6302808341	canara@gmail.com	Tadipatri, Anantapur dist, Andhra Pradesh 515411
4	SBI Bank	8833736795	sbi@gmail.com	515413, Anantapur
5	AXIS	9874363210	axis@gmail.com	vijayawada Andhra Pradesh
6	Bank Of Baroda	9856370147	bob@gmail.com	Tadipatri, Andhra Pradesh
7	Karnataka bank	8310122387	karnatak@gmail.com	Bengaluru
8	squid	9288741025	squid@gmail.com	bangalore

Figure 7.3.3 view Organization

In the above fig 7.3.3 shows a list of registered organizations along with their name, phone number, email, and address. Gives a clear and organized look at the organizations in the KYC verification system. The details of each organization can be verified and safely stored on the blockchain, making sure the records are authentic and unchangeable.

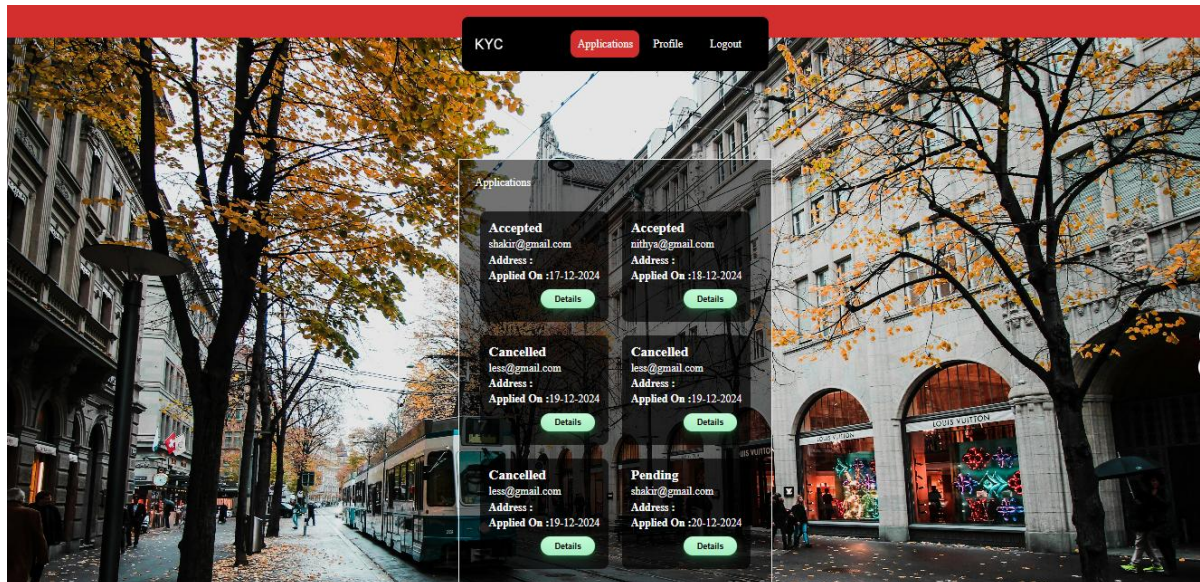


Figure 7.3.4 Organization home page

In the above fig 7.3.4 shows a user interface meant for KYC verification. It displays the status of applications that can be "Accepted," "Pending," or "Cancelled". Users can check the details of their applications and also choose to delete any submissions they want.

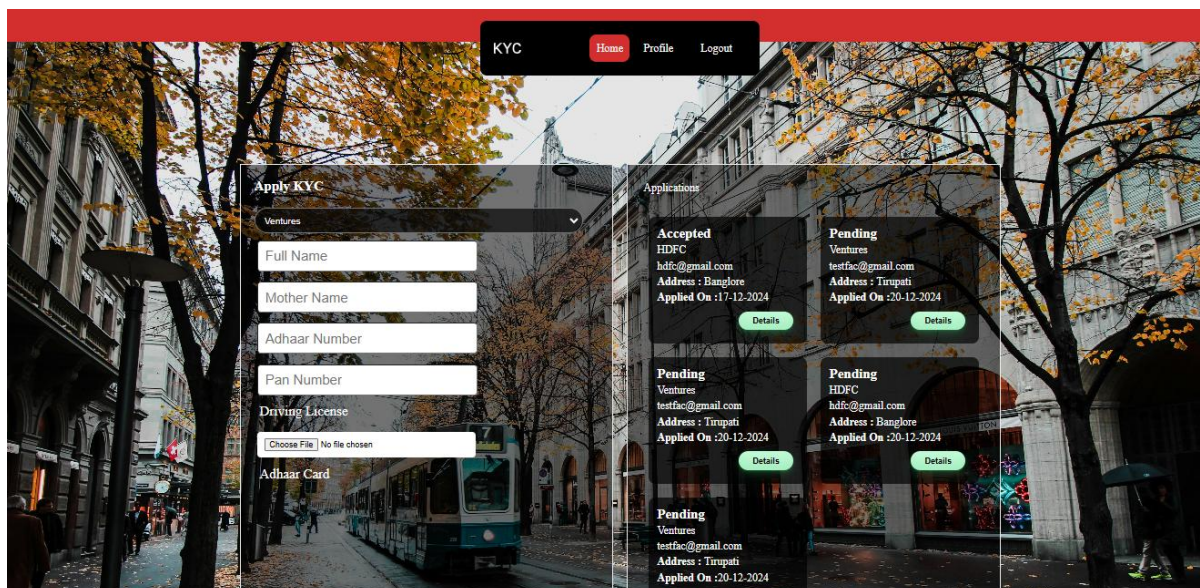


Figure 7.3.5 Customer Home page

In the above fig 7.3.5 shows a KYC verification screen that uses blockchain technology. To the left, users can fill out their information, including Full Name, Aadhaar Number, PAN, Driving License, and Address Card for the verification process. On the right side, you can see the status of the applications that have been submitted.

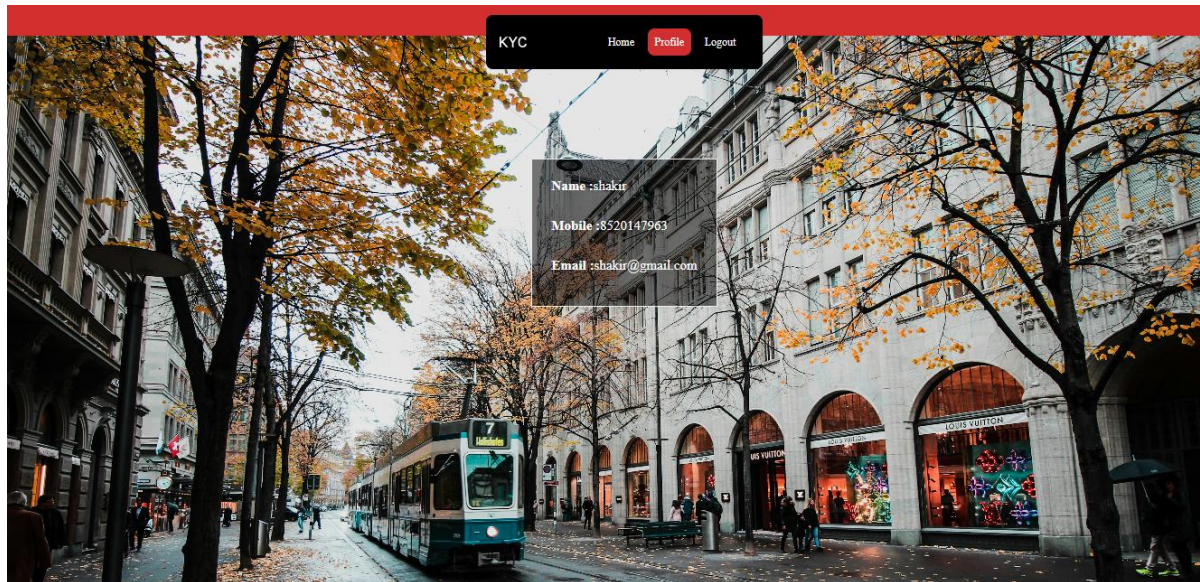


Figure 7.3.6 Customer Profile page

In the above fig 7.3.6 shows a KYC (Know Your Customer) profile page that works with blockchain technology. In the middle of the page, you can see important user information like their Name, Mobile Number, and Email Address. At the top, there's a navigation bar with easy access options such as Home, Profile, and Logout.

CHAPTER-8

SYSTEM STUDY AND TESTING

We took a close look at the KYC verification system powered by blockchain, putting it through detailed study and thorough testing to check if it really works well in the real world. In this part, we will break down the study and testing steps, describing how we assessed and confirmed the system's ability to meet user needs, security requirements, and performance goals.

8.1 Study Overview

a. Purpose of the Study

The main aim of the study was to find out if this blockchain-based KYC verification system could overcome the shortcomings of older methods. We concentrated on:

Functionality: Examining how the system carries out key KYC tasks, like submitting customer data, checking by organizations, and sending notifications.

Security: Making sure customer data is securely stored and shared, while following privacy rules.

Performance: Ensuring the system can manage growing demands and that the blockchain can support extensive use.

Scalability: Checking how well the system can grow as the number of organizations, users, and verifications increases.

Usability: Ensuring the user interfaces are easy for customers, organizations, and administrators to use.

The study aimed to provide a clearer picture of the project's feasibility by looking into these areas, considering future needs, and suggesting necessary improvements.

b. Analyzing System Components

We took a closer look at the different parts of the system to understand how it works and its design:

Blockchain Network:

The system runs on a public blockchain, like Ethereum, to guarantee transparency and decentralization. This blockchain is used to log transaction data and KYC verification statuses. Smart contracts manage key actions tied to KYC submissions and validations, making sure everything is automated and secure.

Web Interface:

We designed the front end to be user-friendly:

Customers can upload their documents, provide their details, and follow the verification status.

Organizations can check customer submissions, verify KYC information, and update statuses.

Admins oversee the entire system and track user activities.

Admin Dashboard:

The admin dashboard gives control to administrators, allowing them to add new organizations, keep an eye on activities, and resolve disputes. It provides a complete view of the entire process, helping with proactive problem-solving.

c. System Requirements

We outlined the system requirements to align with user needs, breaking them down into:

Functional Requirements: Features like submitting KYC documents, tracking verification in real-time, allowing organizations to verify and update statuses, and sending automated notifications.

Non-Functional Requirements: Building a system that can grow with the user base, ensuring strong security (encryption, access limits), and maintaining high performance during busy periods.

8.2 Testing the System**8.2.1 Unit Testing:**

We carried out unit tests to check the individual parts of the system. This included verifying: If the smart contract accurately records customer details and document hashes on the blockchain. Whether the document hash generated from IPFS matches what's stored on the blockchain for data integrity. If the notification system sends updates correctly when a KYC submission status changes

8.2.2 Integration Testing:

We ran integration tests to make sure all parts of the system work well together. One important test checked that when a customer shares their KYC details, this information is uploaded to IPFS and linked correctly to the relevant blockchain transaction. This showed that the system's upload and storage features are operating as they should.

Another key test made sure that organizations could access customer data stored on the blockchain. By using transaction hashes, organizations could retrieve and confirm KYC information, proving that the system is good at sharing and verifying data. Finally, we tested

whether any updates made by an organization to a customer's KYC status showed up accurately. We confirmed that these updates were recorded on the blockchain and visible in the customer's portal, which helped keep the whole system synchronized and transparent.

8.2.3 Performance Testing:

We made sure the system would work well even with a lot of users by running careful performance tests. One important test was load testing, where we pretended that up to 50 users were using the system at the same time. This helped us see how the blockchain network and the front-end application handled the extra traffic and pointed out any slow spots. We also did throughput testing to see how many KYC verifications could be done in a certain amount of time while under load. The results showed that even when things got busy, the system kept an average processing time of 5 minutes for each verification. Lastly, we conducted latency testing to check how quickly and responsively important actions, like uploading documents and updating statuses, were completed. The tests showed that these tasks were done in a reasonable time, making sure the user experience was smooth.

8.2.4 Usability Testing:

We carefully checked the user experience to ensure it was straightforward for customers, organizations, and admins.

A group of test users from various backgrounds assessed the user interface. Feedback showed that 85% of customers found the registration and document submission steps easy to follow. Organizations and admins also praised the efficiency of the verification and monitoring tasks.

8.3 Testing Outcomes

8.3.1 Functionality:

All functional elements worked as intended, including KYC data submissions, verification status updates, and real-time tracking for customers.

The smart contracts and notification system functioned without issues, allowing for smooth workflow automation

8.3.2 Usability:

User feedback indicated that the system was easy to use and navigate. Customers found it simple to submit their documents, and organizations appreciated the straightforward verification process.

We made some minor adjustments to the user interface based on feedback, simplifying navigation and improving the design of the status tracking page.

8.4 Conclusion of the Study and Testing

The study and testing phases have confirmed that the blockchain-based KYC verification solution is viable and effective. The system held up well under various tests, proving to be secure, efficient, and user-friendly. The results suggest the system is ready for launch, capable of scaling and managing large volumes of KYC verifications in real-time.

Looking ahead, we plan to enhance transaction costs, improve document upload processes, and further optimize system performance as the platform grows. With its secure, decentralized, and transparent nature, this KYC verification system could truly change how KYC processes are managed around the world.

CHAPTER-9

CONCLUSION

The KYC Verification System built on blockchain has shown how well this technology can enhance and simplify the KYC process. With blockchain, the system provides transparency, keeps data safe, and operates without a central authority, addressing the issues commonly faced by older KYC methods.

Throughout the project, we tackled significant challenges, such as secure data management, speedy document checks, and ensuring the system could grow as needed. By storing KYC data on the blockchain, we made it tamper-proof and accessible only to those with permission. This setup reduces risks that come with centralized systems, like data leaks or unauthorized access.

After thorough testing, we can confirm that the system meets all necessary functional and performance standards. It can manage high volumes of traffic, process verifications quickly, and uphold security measures without slowing down. The designs for customer, organization, and admin interfaces are user-friendly, and we received great feedback from usability tests.

Although the system is ready to go live, we are looking at ways to make it even better, particularly regarding transaction costs and scalability. We will investigate options like Layer-2 solutions to help keep transaction fees low as the system grows.

To sum up, this blockchain-based KYC verification system provides a reliable and secure method for managing customer identities. It could change the way KYC is done, offering a clearer, more efficient, and safer alternative to the conventional processes, leading to a more streamlined and accessible KYC verification system worldwide.

FUTURE ENHANCEMENT

1. Integration with External Identity Verification Systems

Current Limitation: The system currently relies on manual data submission by customers for verification.

Future Enhancement: Integration with government-backed identity verification systems (such as Aadhaar in India or other national identity databases) can streamline the process by allowing direct verification of customer identities. This integration would automate the process further and reduce the chances of errors or fraud during submission.

2. Incorporating Biometric Authentication

Current Limitation: The system primarily relies on document submission for verification.

Future Enhancement: Incorporating biometric authentication (such as face recognition or fingerprint scanning) will add an additional layer of security to the KYC process. This would help ensure that the person submitting the documents is the same as the one in the verification process, improving the overall accuracy and preventing identity theft.

3. Multi-Blockchain Interoperability

Current Limitation: The system is currently designed to operate on a single blockchain, such as Ethereum.

Future Enhancement: Developing multi-chain compatibility will allow the system to interact with different blockchain networks. This would improve flexibility and allow organizations to choose the blockchain that best fits their specific needs. Multi-chain interoperability would also help in reducing congestion and ensuring better performance.

4. Mobile Application Development

Current Limitation: The system is primarily web-based and may not be accessible to all users, especially in remote areas.

Future Enhancement: Developing a mobile application for both customers and organizations will increase accessibility and ease of use. This app could allow users to upload documents, track their KYC status, and receive notifications directly on their mobile devices, further enhancing the user experience.

5. Enhanced User Feedback Mechanisms

Current Limitation: The system currently provides basic notifications to users but lacks an interactive feedback mechanism.

Future Enhancement: Introducing a robust feedback system would allow users (both customers and organizations) to rate their experience, suggest improvements, or report issues

in real time. This continuous feedback loop will help improve the system based on actual user experiences.

6. Globalization and Localization

Current Limitation: The system is designed for a general audience, without support for regional languages or country-specific regulations.

Future Enhancement: Adding multi-language support and the ability to adhere to local KYC regulations will make the system more adaptable and useful for global deployment. This enhancement will ensure the system complies with country-specific KYC laws and provides an accessible interface for diverse user groups.

7. Enhanced Security Features

Current Limitation: The system offers strong security, but as threats evolve, additional security measures may be required.

Future Enhancement: Continuously updating the system to include the latest security protocols, such as multi-factor authentication (MFA), hardware security modules (HSMs), and AI-based fraud detection systems, will ensure that the system remains secure against emerging threats and cyber-attacks.

8. AI-Powered Fraud Detection

Current Limitation: The system currently relies on user-provided documents and smart contracts to verify identities.

Future Enhancement: Integrating artificial intelligence (AI) and machine learning (ML) models can be used to detect unusual patterns or anomalies in submitted KYC data. AI could analyze behavioral data and document authenticity, helping to prevent fraud before it occurs.

REFERENCES

1. Elgamal, E., Medhat, W., Abd Elfatah, M., & Abdelbaki, N.(2023). Blockchain's part in Enhancing Big Data Security.
- 2.Fartitchou, M., El Makkaoui, K., Kannouf, N., & El Allali, Z.(2020). Exploring Security Measures in Blockchain Technology.
- 3.Manimurgan, S., Anitha, T., Divya, G., Pushpa Latha, G. C., & Mathupriya, S.(2022). operations of Blockchain in Network Security A Review.
4. Draper, A., Familrouhani, A., Cao, D., Heng, T., & Han, W.(2019). Challenges and Operations in Blockchain Security.
5. Labbe, A.(2017). Blockchain Bonds Reducing Costs and Processing Time. Worldwide Financial Law Survey.
- 6.Sundareswaran, N., Sasirekha, S., Paul, I. J. L., Balakrishnan, S., & Swaminathan, G.(2023). Enhanced Blockchain- Grounded KYC Framework.
- 7.Al Mamun, A., Hasan, S. R., Bhuiyan, M. S., Kaiser, M. S., & Yousuf, M. A.(2023). A Secure KYC System using IPFS and Blockchain Technology.
- 8.Sharma, M., & Gupta, P.(2020). Decentralized KYC System Using Blockchain A Model perpetration.
- 9.Sivakumar, P., & Singh, K.(2023). Decentralized PKI with sequestration Using Smart Contracts on Blockchain.
- 10.Ravale, U., Ramakrishnan, A., Borkar, A., & Deshmukh, S.(2023). Enhancing KYC Verification through Ethereum Blockchain Applications.
- 11.Anuar, M. A. B., Zeki, A. M., Abubakar, A.,(2023). Exploring Blockchain for bettered KYC Processes and client perceptivity.
- 12.Yadhav, P., & Chandak, R.(2019). Transforming the Know Your client(KYC) Process with Blockchain.
13. Mansoor, N., Antora, K. F., Deb, P., Arman, T. A., Manaf, A. A., & Zareei, M.(2023). A Comprehensive Review of Blockchain- Grounded KYC results.
14. Rathod, V. U., Mali, Y. K., Sable, N. P., Rathod, R. S. K., Rathod, M. N., & Rathod, N. A.(2023). exercising Blockchain Technology for KYC Document Verification.
- 15.Dhiman, B., & Bose, R. S.(2022). A Robust and Secure Decentralized KYC Verification System Using Blockchain Technology.

APPENDIX-A

CODE

logic.py

```

1  import json
2  from (module) datetime
3  from datetime import date
4
5
6
7  web3 = Web3(Web3.HTTPProvider("http://127.0.0.1:7545"))
8  one_app = "0xD289dEa562C50f10B89d75Ff155796aa6Aa3665a"
9
10 contract_address_for_user = "0x34807573b552422f7106E49987B37fe04EF50f14"
11
12
13 ##Contract for users
14 with open("blocks/build/contracts/UserPoint.json") as abiFile:
15     abidata = json.load(abiFile)
16     realABI = abidata["abi"]
17     contract_user = web3.eth.contract(address=contract_address_for_user, abi=realABI)
18
19 ##contract of Files
20 contract_address_of_details = "0xB1CeE53D2D074a20260cFCF751889340ed049ceA"
21 with open("blocks/build/contracts/DetailsKyc.json") as abiFileOF:
22     abiFileData = json.load(abiFileOF)
23     abiData = abiFileData["abi"]
24     contract_of_files = web3.eth.contract(
25         address=contract_address_of_details, abi=abiData
26     )
27
28
29
30 def userAdd(name, mobile, email, password, adress, typeUser):
31     user = getUsersDetails()[3]
32     if email in user:
33         return "Try with another email"
34     else:
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71 def getTypeDiffer(type,userid):
72     data = []
73     viewuser = getUsersDetails()
74     if "getNameOfOrganisations" == type:
75         for i in range(len(viewuser[0])):
76             if viewuser[4][i] == "organisation":
77                 data.append(
78                     {
79                         "name": viewuser[1][i],
80                         "id": viewuser[0][i],
81                     }
82                 )
83     elif "NothingUserProfile"==type:
84         for i in range(len(viewuser[0])):
85             if viewuser[0][i] == int(userid):
86                 data.append(
87                     {
88                         "id": viewuser[0][i],
89                         "name": viewuser[1][i],
90                         "mobile": viewuser[2][i],
91                         "email": viewuser[3][i],
92                         "adress": viewuser[5][i],
93                     }
94                 )
95     else:
96         readid=0
97         for i in range(len(viewuser[0])):
98             if viewuser[4][i] == type:
99                 readid+=1
100                 data.append(
101                     {
102                         "readid":readid,
103                         "id": viewuser[0][i],
104                         "name": viewuser[1][i],

```

```

app > logic.py > loginSubmit
114 def getApplied(type, userid):
115     data = []
116     if "getMyUserOrder" == type:
117         view = getContractsOfMe()
118
119         companyDetails = getUsersDetails()
120         for i in range(len(view[0])):
121             if str(view[2][i]) == str(userid):
122                 details = {}
123                 for values in range(len(companyDetails[0])):
124                     if str(companyDetails[0][values]) == str(view[1][i]):
125                         details["NameOfCompany"] = companyDetails[1][values]
126                         details["Email"] = companyDetails[3][values]
127                         details["Address"] = companyDetails[5][values]
128                 datePoint = date.fromtimestamp(float(int(view[11][i])/1000))
129                 data.append(
130                     {
131                         "id": view[0][i],
132                         "NameOfCompany": details["NameOfCompany"],
133                         "Email": details["Email"],
134                         "Address": details["Address"],
135                         "companyId": view[1][i],
136                         "userId": view[2][i],
137                         "fullName": view[3][i],
138                         "motherName": view[4][i],
139                         "adhaarName": view[5][i],
140                         "panNumber": view[6][i],
141                         "drivingPath": view[7][i],
142                         "adhaarPath": view[8][i],
143                         "photoPath": view[9][i],
144                         "status": view[10][i],
145                         "appliedOn": f"{datePoint.day}-{datePoint.month}-{datePoint.year} ",
146                     }
147                 )
148
149 elif "getComapanyApplied"==type:
150     view = getContractsOfMe()
151     companyDetails = getUsersDetails()
152     for i in range(len(view[0])):
153         if str(view[1][i]) == str(userid):
154             details = {}
155             for values in range(len(companyDetails[0])):
156                 if str(companyDetails[0][values]) == str(view[2][i]):
157                     details["NameofUser"] = companyDetails[1][values]
158                     details["Email"] = companyDetails[3][values]
159                     details["Address"] = companyDetails[5][values]
160             datePoint = date.fromtimestamp(float(int(view[11][i])/1000))
161             data.append(
162                 {
163                     "id": view[0][i],
164                     "NameofUser": details["NameofUser"],
165                     "Email": details["Email"],
166                     "Address": details["Address"],
167                     "companyId": view[1][i],
168                     "userId": view[2][i],
169                     "fullName": view[3][i],
170                     "motherName": view[4][i],
171                     "adhaarName": view[5][i],
172                     "panNumber": view[6][i],
173                     "drivingPath": view[7][i],
174                     "adhaarPath": view[8][i],
175                     "photoPath": view[9][i],
176                     "status": view[10][i],
177                     "appliedOn": f"{datePoint.day}-{datePoint.month}-{datePoint.year} ",
178                 }
179             )
180
181     return data

```

Logs CyberCoder Improve Code Share Code Link Ln 67, Col 31 Spaces: 4 UTF-8 CRLF {} Python 3.12.6

Views.py

```

pp > views.py > uploadMyDetails
1  from django.shortcuts import render
2  from django.http import JsonResponse
3  import json
4  from . import logic
5  from .models import LocalStore
6  import os
7  from django.conf import settings
8  import time
9
10 def index(request):
11     if request.method == "POST":
12         LocalStore.objects.all().delete()
13         return render(request, "index.html")
14
15     typeView = LocalStore.objects.all()
16     if typeView.exists() == False:
17         return render(request, "index.html")
18     actor = typeView.last().actor
19
20     if actor == "admin":
21         return render(request, "admin/home.html")
22     elif actor == "orgnaisation":
23         applied=logic.getApplied('getComapanyApplied',typeView.last().userid)
24
25         return render(request, "organisation/home.html",context={
26             'applied':applied
27         })
28     elif actor == "user":
29         data=logic.getTypeDiffer('getNameOfOrganisations',0)
30         applied=logic.getApplied('getMyUserOrder',typeView.last().userid)
31
32         return render(request, "users/home.html",context={
33             'companies':data,
34             'applied':applied })
35
36
37
38
39 def submit(request):
40     if request.method == "GET":
41         name = request.GET.get("name", "")
42         mobile = request.GET.get("mobile", "")
43         email = request.GET.get("Email", "")
44         password = request.GET.get("Password", "")
45         adress = request.GET.get("Address", "")
46         typeUser = request.GET.get("typeUser", "")
47         lower = str(email).lower().strip()
48
49         view = logic.userAdd(name, mobile, lower, password, adress, typeUser)
50         print(view)
51         return JsonResponse(data={"error": True, "message": view})
52
53
54 def login(request):
55     if request.method == "GET":
56         email = request.GET.get("email", "")
57         password = request.GET.get("password", "")
58         data = logic.loginSubmit(email, password)
59         return JsonResponse(data={"error": True, "message": f"{data}"})
60
61
62 def adminPart(request):
63
64     return render(request, "admin/home.html")
65
66
67 def viewOrgnisations(request):
68     data = logic.getTypeDiffer("organisation",0)
69     return render(request, "admin/view_ornisation.html", context={"data": data})
70
71 def user(request):
72     data=logic.getTypeDiffer('getNameOfOrganisations',0)

```

```

def uploadMyDetails(request):
    if request.method=="POST":
        drive=request.FILES['drive']
        photo=request.FILES['photo']
        addhaar=request.FILES['addhaar']
        companyId=request.POST.get("company","")
        fullName=request.POST.get("fullName","")
        motherName=request.POST.get("motherName","")
        adhaarNumber=request.POST.get("adhaarNumber","")
        panNumber=request.POST.get("panNumber","")

        first_millies=time.time_ns()// 1_000_000
        drivepath=os.path.join("app/static/savedImages/",f"{str(first_millies)}.jpg")
        with open(drivepath,"wb+") as destinationPort:
            for chunkPoint in drive.chunks():
                destinationPort.write(chunkPoint)
        drivepath=drivepath.replace("app/","/")

        second_millies=time.time_ns()// 1_000_000
        addhaarpj=os.path.join("app/static/savedImages/",f"{str(second_millies)}.jpg")
        with open(addhaarpj,"wb+") as destinationpathj:
            for chunkView in addhaar.chunks():
                destinationpathj.write(chunkView)

        addhaarpj=addhaarpj.replace("app/","/")

        third_millis=time.time_ns()// 1_000_000

        photopath=os.path.join("app/static/savedImages/",f"{str(third_millis)}.jpg")
        with open(photopath,"wb+") as destinationView:
            for chunkWay in photo.chunks():

def updatePoint(request):
    if request.method=="POST":
        id=request.POST.get("nameid",0)
        status=request.POST.get("status","")
        logic.updateStateOfContract(int(id),status)
        applied=logic.getApplied('getComapanyApplied',LocalStore.objects.all().last().userid)

        return render(request, "organisation/home.html",context={
            'applied':applied
        })

def profile(request):
    userID=LocalStore.objects.all().last().userid
    data=logic.getTypeDiffer("NothingUserProfile",userID)
    return render(request,'users/profile.html',context={
        'data':data
    })

```


Detailskyc.sol

```

blocks > contracts > DetailsKyc.sol > DetailsKyc > getUsers
1  // SPDX-License-Identifier: MIT
2  pragma solidity ^0.8.21;
3
4  contract DetailsKyc {
5      struct KYC {
6          uint256 id;
7          uint256 companyId;
8          uint256 userId;
9          string fullName;
10         string motherName;
11         string adhaarName;
12         string panNumber;
13         string drivingPath;
14         string adhaarPath;
15         string photoPath;
16         string status;
17         string appliedOn;
18     }
19     mapping(address => KYC[]) public kyclist;
20
21     function addPress(
22         uint256 _companyId,
23         uint256 _userId,
24         string memory _fullName,
25         string memory _motherName,
26         string memory _adhaarName,
27         string memory _panNumber,
28         string memory _drivingPath,
29         string memory _adhaarPath,
30         string memory _photoPath,
31         string memory _status,
32         string memory _appliedOn
33     ) public {
34         uint256 id = kyclist[msg.sender].length + 1;

```

```

54     function getUsers()
70     {
71     }
72     {
73         uint256 count = kyclist[msg.sender].length;
74         uint256[] memory _id = new uint256[](count);
75         uint256[] memory _companyId = new uint256[](count);
76         uint256[] memory _userId = new uint256[](count);
77         string[] memory _fullName = new string[](count);
78         string[] memory _motherName = new string[](count);
79         string[] memory _adhaarName = new string[](count);
80         string[] memory _panNumber = new string[](count);
81         string[] memory _drivingPath = new string[](count);
82         string[] memory _adhaarPath = new string[](count);
83         string[] memory _photoPath = new string[](count);
84         string[] memory _status = new string[](count);
85         string[] memory _applied = new string[](count);
86
87         for (uint256 index = 0; index < count; index++) {
88             KYC storage user = kyclist[msg.sender][index];
89             _id[index] = user.id;
90             _companyId[index] = user.companyId;
91             _userId[index] = user.userId;
92             _fullName[index] = user.fullName;
93             _motherName[index] = user.motherName;
94             _adhaarName[index] = user.adhaarName;
95             _panNumber[index] = user.panNumber;
96             _drivingPath[index] = user.drivingPath;
97             _adhaarPath[index] = user.adhaarPath;
98             _photoPath[index] = user.photoPath;
99             _status[index] = user.status;
100             _applied[index] = user.appliedOn;
101         }

```

UserPoint.sol

```

blocks > contracts > UserPoint.sol > ...
1  // SPDX-License-Identifier: MIT
2  pragma solidity ^0.8.21;
3
4  contract UserPoint {
5      struct User {
6          uint256 id;
7          string name;
8          string mobile;
9          string mail;
10         string actor;
11         string password;
12         string addressP;
13     }
14     mapping(address => User[]) public userList;
15
16     function addPress(
17         string memory _name,
18         string memory _mobile,
19         string memory _mail,
20         string memory _actor,
21         string memory _password,
22         string memory _addresss
23     ) public {
24         uint256 id = userList[msg.sender].length + 1;
25         userList[msg.sender].push(
26             User(id, _name, _mobile, _mail, _actor, _password, _addresss)
27         );
28     }
29
30     function getUsers()
31         public
32         view
33         returns (
34             uint256[] memory,
35             string[] memory,
36             string[] memory,
37             string[] memory,
38             string[] memory,
39             string[] memory,
40             string[] memory
41         )
42     {
43         uint256 count = userList[msg.sender].length;
44         uint256[] memory id = new uint256[](count);
45         string[] memory names = new string[](count);
46         string[] memory mobs = new string[](count);
47         string[] memory mails = new string[](count);
48         string[] memory _addressP = new string[](count);
49         string[] memory typePro = new string[](count);
50         string[] memory _password = new string[](count);
51
52         for (uint256 index = 0; index < count; index++) {
53             User storage user = userList[msg.sender][index];
54             id[index] = user.id;
55             names[index] = user.name;
56             mobs[index] = user.mobile;
57             mails[index] = user.mail;
58             typePro[index] = user.actor;
59             _addressP[index] = user.addressP;
60             _password[index] = user.password;
61         }
62         return (id, names, mobs, mails, typePro, _addressP, _password);
63     }
64 }
65
66

```

Manage.py

```

manage.py > ...
1  #!/usr/bin/env python
2  """Django's command-line utility for administrative tasks."""
3  import os
4  import sys
5
6
7  def main():
8      """Run administrative tasks."""
9      os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'kycproject.settings')
10     try:
11         from django.core.management import execute_from_command_line
12     except ImportError as exc:
13         raise ImportError(
14             "Couldn't import Django. Are you sure it's installed and "
15             "available on your PYTHONPATH environment variable? Did you "
16             "forget to activate a virtual environment?"
17         ) from exc
18     execute_from_command_line(sys.argv)
19
20
21 if __name__ == '__main__':
22     main()
23

```

Index.html

```

app > templates > index.html > html > script
2  <html lang="en">
10 <body>
15     <div class="container">
16         <p id="token" style="display: none;">{{csrf_token}}</p>
17         <p
18             style="font-weight: bold;font-family: 'Lucida Sans', 'Lucida Sans Regular', 'Luc
19             Login Form</p>
20         <div class="btnClass">
21             <div class="optionSelectionActive" id="login" style="border-radius: 0px 10px 10px
22                 <p style="padding: 0px;margin: 0px;">
23                     Login
24                 </p>
25             </div>
26
27             <div class="optionSelection" id="signup">
28                 <p style="padding: 0px;margin: 0px;">
29                     Sign up
30                 </p>
31             </div>
32         </div>
33         <div id="loginDeposit">
34             <input id="email" type="email" placeholder="Email" style="margin-top:50px ;" />
35             <input id="password" type="password" placeholder="Password"><br>
36             <div style="display: flex;justify-content: center;margin-top: 20px;">
37                 <button style="font-weight: bold;" onclick="loginPage()" id="loginInSubmit">
38                     Login
39                 </button>
40             </div>
41         </div>
42
43
44
45     </div>

```

First_script.js

```

app > static > js > JS first_script.js > setPostion
11 login.addEventListener('click', () => {
12     setPostion(true)
13 })
14 signup.addEventListener('click', () => {
15     setPostion(false)
16 })
17 function setPostion(something) {
18     if (something) {
19         string = `
20         <input type="email" id="email" placeholder="Email" style="margin-top:50px" /><br>
21         <input type="password" id="password" placeholder="Password"><br>
22         <div style="display: flex;justify-content: center;margin-top: 20px;">
23             <button
24                 id='loginInSubmit'
25                 style="font-weight: bold;">
26                 Login
27             </button>
28         </div>`
29         signup.classList.remove('optionSelectionActive')
30         signup.classList.add('optionSelection')
31         login.classList.add('optionSelectionActive')
32         login.classList.remove('optionSelection')
33         loginDeposit.innerHTML = string
34
35         document.getElementById('loginInSubmit')
36             .addEventListener('click', () => {
37                 loginPage()
38             })
39     } else {
40         string = `
41         <input type="text" id='name' placeholder="name" style="margin-top:50px" /><br>
42         <input type="number" id='mobile' placeholder="mobile"><br>
43     `
44     loginDeposit.innerHTML = string
45     document.getElementById('signUpSubmit')
46         .addEventListener('click', () => {
47
48             var name = document.getElementById('name').value
49             var mobile = document.getElementById('mobile').value
50             var Email = document.getElementById('Email').value
51             var Password = document.getElementById('Password').value
52
53             if (name.length == 0) {
54                 toast("Please enter your Name")
55             } else if (mobile.length == 0) {
56                 toast("Please enter your Mobile")
57             } else if (Email.length == 0) {
58                 toast("Please enter your Email")
59             } else if (Password.length == 0) {
60                 toast("Please enter your Password")
61             } else {
62                 signupsSubmit(
63                     name,
64                     mobile,
65                     Email,
66                     Password
67                 )
68             }
69         })
70     }
71 }
72
73
74
75
76
77
78
79
80
81
82
83
84
85

```

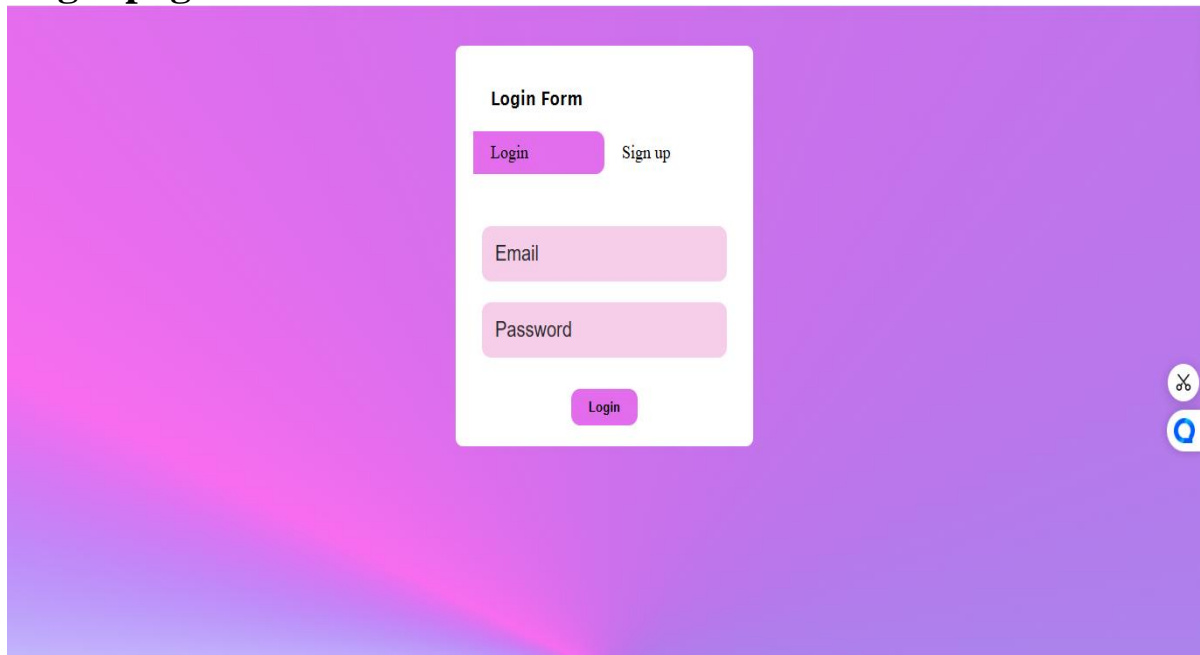
```

app > static > js > JS first_script.js > setPostion
18  function setPostion(something) {
88      signup.classList.remove('optionSelection')
89      signup.classList.add('optionSelectionActive')
90      login.classList.add('optionSelection')
91      login.classList.remove('optionSelectionActive')
92  }
93  }
94
95  function toast(message) {
96      var snack = document.getElementById('snackbar')
97      snack.className = 'show'
98      document.getElementById('point').innerText = message
99
100     setTimeout(() => { snack.className = snack.className.replace('show', '') }, 4000)
101  }
102
103
104  async function signupsubmit(name, mobile, Email, Password) {
105
106      const request = new Request(`/submit?name=${name}&mobile=${mobile}&Email=${Email}&Password=${Password}&typeUser=user`, {
107          method: 'GET',
108      })
109      const response = await fetch(request)
110      if (!response.ok) {
111          alert(response.status)
112      } else {
113          const data = await response.json()
114          if ('Success' == data.message) {
115              setPostion(true)
116          }
117          toast(data.message)
118      }
119  }
120

```

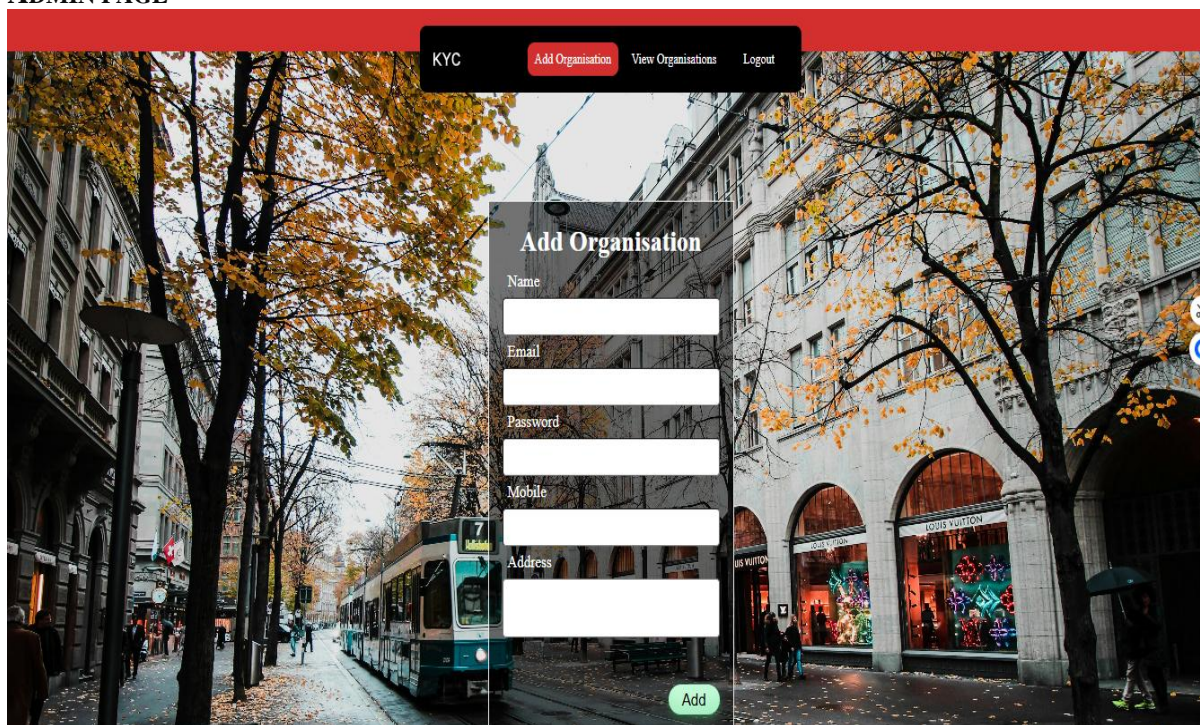

APPENDIX-B

Login page



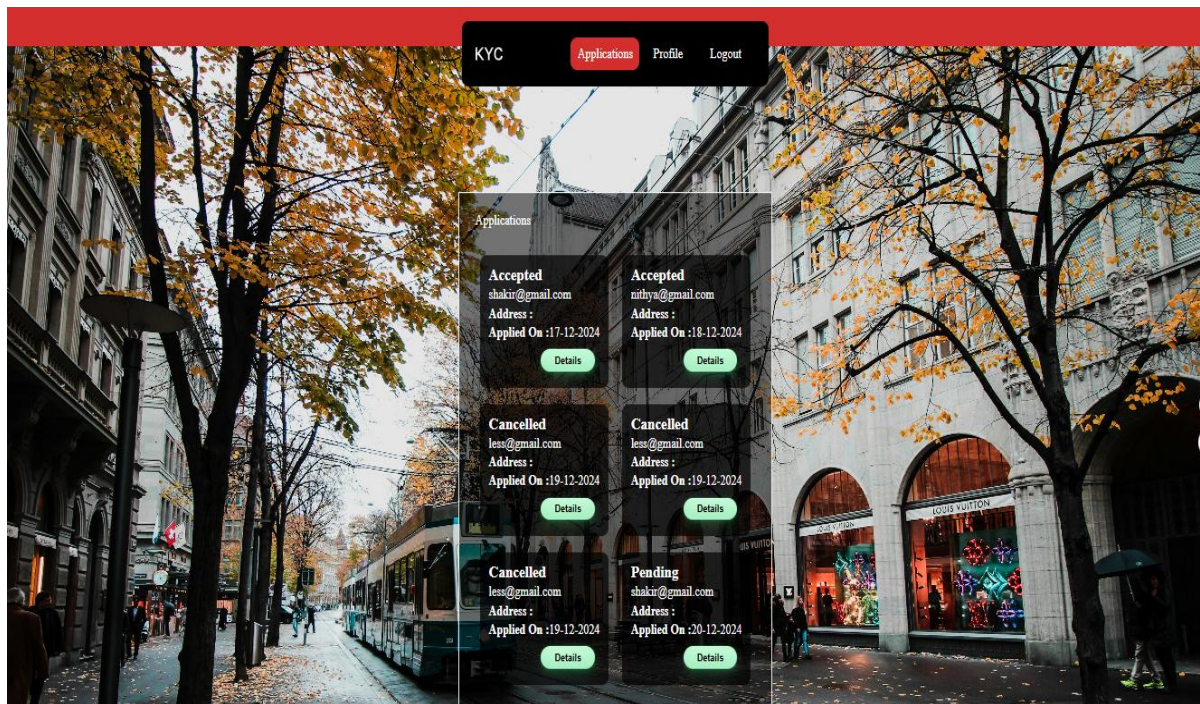
The login page features a white login form centered on a purple-to-pink gradient background. The form is titled "Login Form" and includes a "Login" button and a "Sign up" link. Below these are input fields for "Email" and "Password", followed by a "Login" button. On the right side of the page, there are two circular icons: one with a magnifying glass and another with a blue 'Q'.

ADMIN PAGE

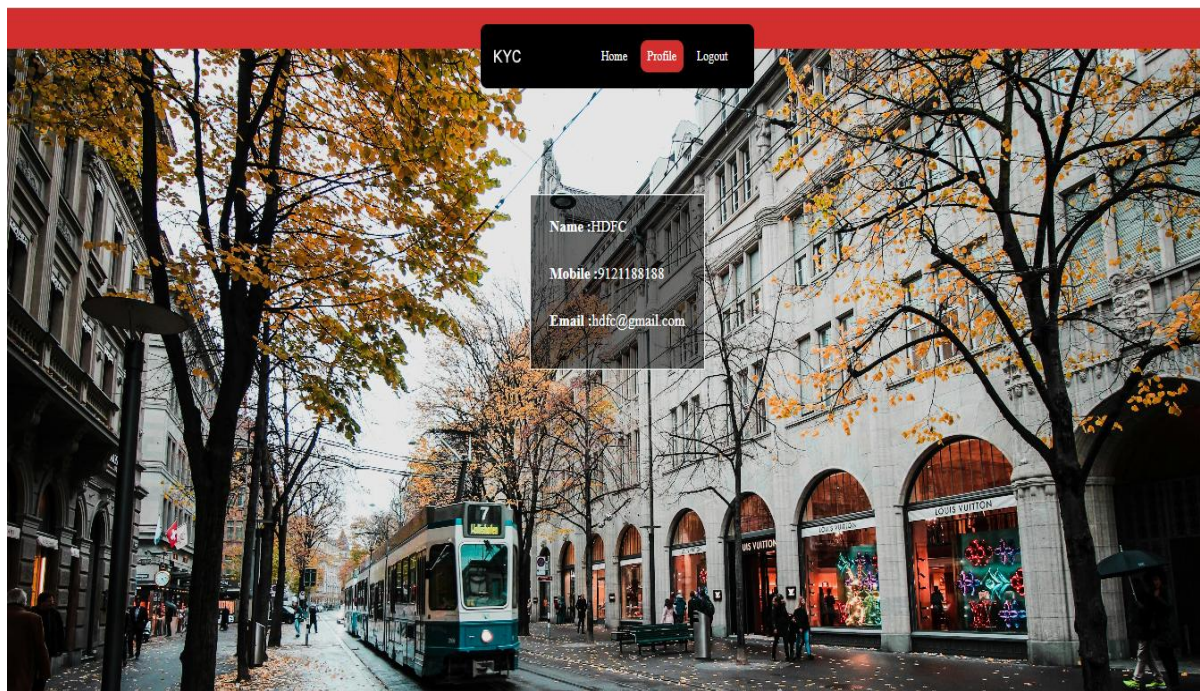


The admin page has a red header bar with the text "KYC" and three buttons: "Add Organisation", "View Organisations", and "Logout". The main content area shows a background image of a city street with a tram and trees. Overlaid on this is a white "Add Organisation" form with input fields for "Name", "Email", "Password", "Mobile", and "Address". A green "Add" button is at the bottom right of the form.

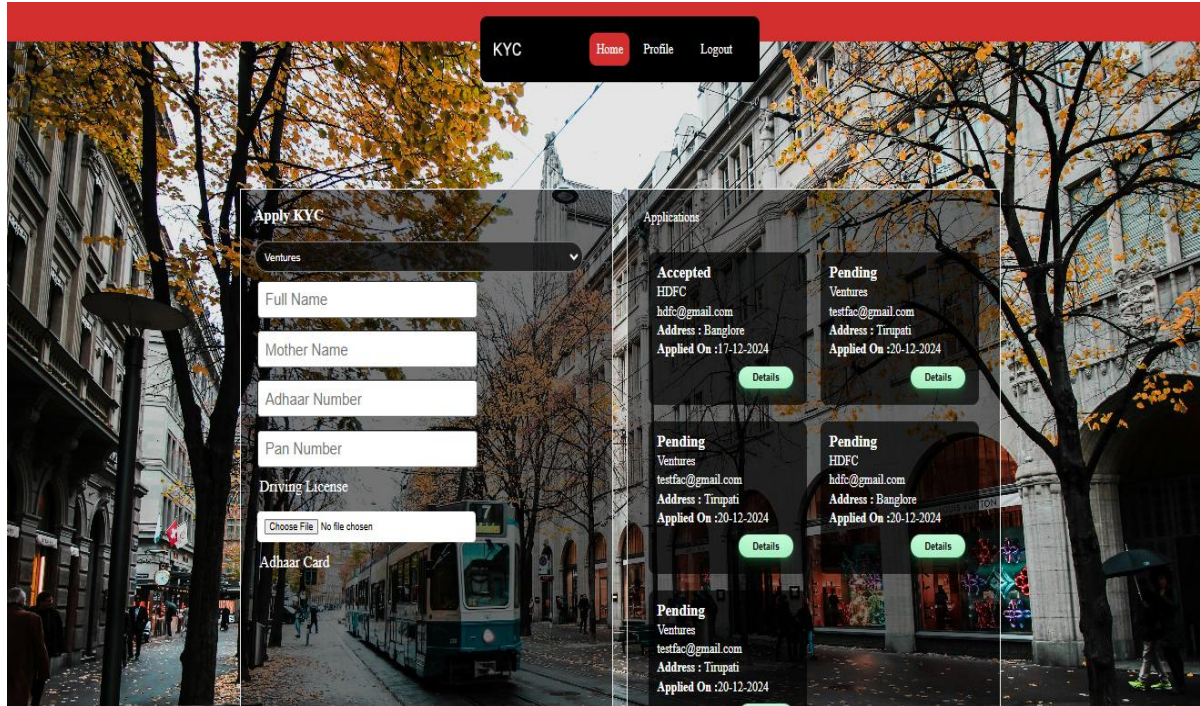
ORGANIZATION HOME PAGE



ORGANIZATION PROFILE PAGE



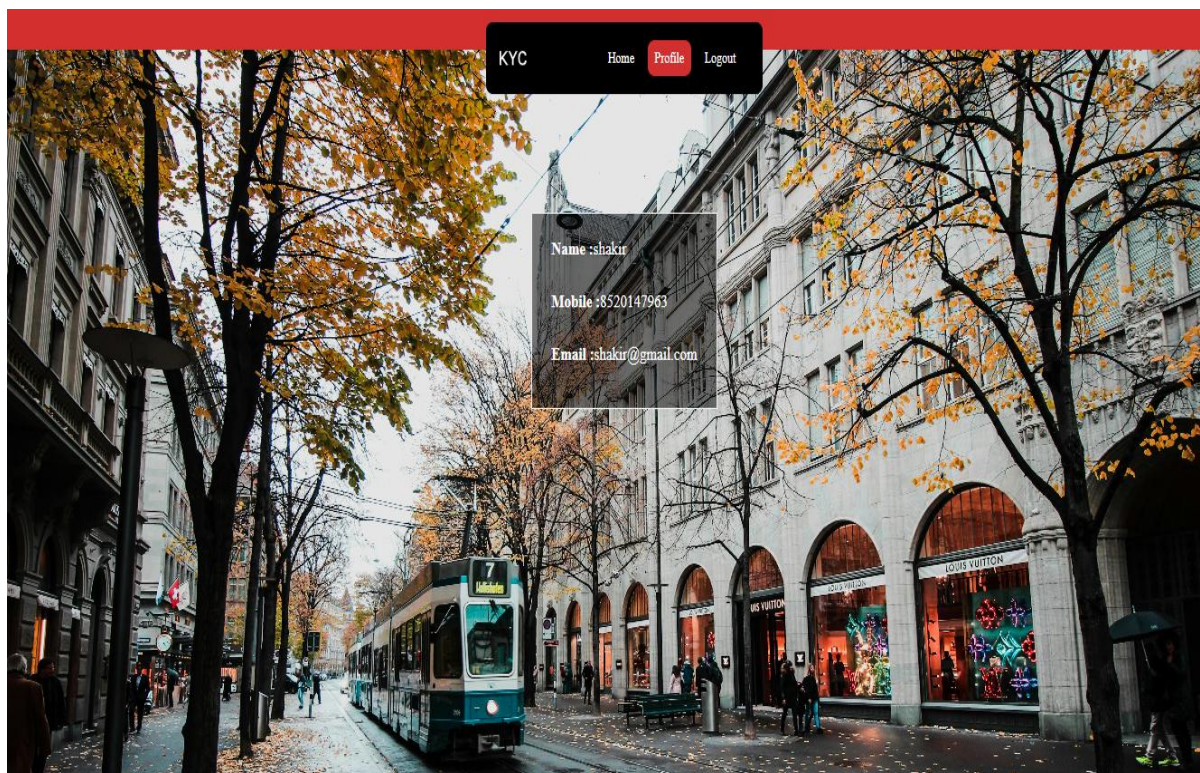
CUSTOMER HOME PAGE



The screenshot shows the 'CUSTOMER HOME PAGE' of a KYC verification system. The background is a street scene with a tram and trees. The page has a red header with a navigation bar containing 'KYC', 'Home' (highlighted), 'Profile', and 'Logout'. The main content area is divided into two sections. The left section, titled 'Apply KYC', contains a dropdown menu for 'Ventures' and input fields for 'Full Name', 'Mother Name', 'Adhaar Number', 'Pan Number', and 'Driving License'. Below these is a 'Choose File' button for the 'Adhaar Card'. The right section, titled 'Applications', displays a list of applications with their status (Accepted, Pending), entity (HDFC, Ventures), email, address, and application date. Each entry has a 'Details' button.

Status	Entity	Email	Address	Applied On	Action
Accepted	HDFC	hdfc@gmail.com	Banglore	17-12-2024	Details
Pending	Ventures	testfac@gmail.com	Tirupati	20-12-2024	Details
Pending	Ventures	testfac@gmail.com	Tirupati	20-12-2024	Details
Pending	HDFC	hdfc@gmail.com	Banglore	20-12-2024	Details
Pending	Ventures	testfac@gmail.com	Tirupati	20-12-2024	Details

CUSTOMER PROFILE PAGE



The screenshot shows the 'CUSTOMER PROFILE PAGE' of the same KYC verification system. The background is the same street scene. The page has a red header with a navigation bar containing 'KYC', 'Home', 'Profile' (highlighted), and 'Logout'. The main content area displays the customer's profile information in a dark box: Name : shakir, Mobile : 8520147963, and Email : shakir@gmail.com.

Name : shakir
Mobile : 8520147963
Email : shakir@gmail.com

APPENDIX-C







Page 2 of 57 - Integrity Overview

Submission ID trn:oid::26650:79423521




14% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.

Match Groups

-  **46 Not Cited or Quoted 14%**
Matches with neither in-text citation nor quotation marks
-  **1 Missing Quotations 0%**
Matches that are still very similar to source material
-  **1 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 9%  Internet sources
- 4%  Publications
- 13%  Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.



Page 2 of 57 - Integrity Overview

Submission ID trn:oid::26650:79423521



Match Groups

- 46 Not Cited or Quoted 14%**
Matches with neither in-text citation nor quotation marks
- 1 Missing Quotations 0%**
Matches that are still very similar to source material
- 1 Missing Citation 0%**
Matches that have quotation marks, but no in-text citation
- 0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 9% Internet sources
- 4% Publications
- 13% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Submitted works	Symbiosis International University on 2025-01-16	7%
2	Internet	www.jetir.org	<1%
3	Submitted works	M S Ramaiah University of Applied Sciences on 2023-05-23	<1%
4	Internet	www.geeksforgeeks.org	<1%
5	Submitted works	Presidency University on 2024-01-11	<1%
6	Internet	www.kluniversity.in	<1%
7	Submitted works	Presidency University on 2024-01-11	<1%
8	Submitted works	MIT Academy of Engineering on 2024-05-04	<1%
9	Submitted works	Higher Education Commission Pakistan on 2021-12-03	<1%
10	Internet	ojs.uma.ac.id	<1%





SDG 8: Good Jobs and Economic Growth

Issue: Slow and expensive KYC procedures slow down the onboarding of businesses and customers, which affects economic activities.

Blockchain Approach: Using blockchain can make KYC processes faster and cheaper, helping financial institutions and businesses work more efficiently.

Outcome: This can increase economic activity, cut down on operating costs, and provide a safer space for both businesses and their customers.

SDG 9: Industry, Innovation, and Infrastructure

Issue: Current KYC systems are often centralized, making them vulnerable to data leaks and not very efficient.

Blockchain Approach: Using blockchain creates a safe, clear, and compatible setup for KYC, allowing for new ideas like self-sovereign identities.

Outcome: This approach promotes strong, lasting, and accessible frameworks for managing digital identities and handling financial transactions.

SDG 16: Peace, Justice, and Strong Institutions

Issue: Fraud, money laundering, and corruption flourish in systems that lack clarity and have poor identity checks.

Blockchain Approach: The reliability and openness of blockchain improve KYC processes, cutting down on fraud and helping meet regulations.

Outcome: Strengthens institutions, lowers illegal financial activities, and fosters trust in governance and financial systems.

SDG 17: Working Together for Our Goals

Issue: When institutions do not work together or follow standard procedures in KYC processes, it leads to waste and overlap.

Blockchain Approach: Blockchain allows different organizations like banks, governments, and NGOs to work together in a safe and efficient way using shared KYC systems.

Outcome: This encourages international cooperation and makes sure data can flow easily while still keeping privacy intact and adhering to local laws.