

A
Mini Project Report
On
**SECURE E-VOTING SYSTEM USING ETHEREUM
BLOCKCHAIN TECHNOLOGY AND SMART
CONTRACTS**

Submitted to JNTU HYDERABAD
In Partial Fulfillment of the requirements for the Award of Degree of
**BACHELOR OF TECHNOLOGY
IN
INFORMATION TECHNOLOGY**

Submitted
By

BURRA MALATHI	(218R1A1213)
BOKKENA SANGHAVI	(218R1A1212)
BATHULA CHARAN THEJA	(218R1A1211)
GADDE SHIVA	(218R1A1222)

Under the Esteemed guidance of
Mr. K. ANIL
Assistant Professor, Department of IT
Department of Information Technology



**CMR ENGINEERING COLLEGE
(UGC AUTONOMOUS)**

(Accredited by NAAC & NBA, Approved by AICTE NEWDELHI, Affiliated to JNTU, Hyderabad)
(Kandlakoya, MedchalRoad, R.R.Dist. Hyderabad-501 401)

(2024-2025)

CMR ENGINEERING COLLEGE

(UGC AUTONOMOUS)

(Accredited by NAAC & NBA, Approved by AICTE NEWDELHI, Affiliated to JNTU, Hyderabad)

(Kandlakoya, Medchal Road, R.R. Dist, Hyderabad-501 401)

Department of Information Technology



CERTIFICATE

This is to certify that the project entitled “**Secure E-Voting System Using Ethereum Blockchain Technology And Smart Contracts**” is a bonafide work carried out by

BURRA MALATHI	(218R1A1213)
BOKKENA SANGHAVI	(218R1A1212)
BATHULA CHARAN THEJA	(218R1A1211)
GADDE SHIVA	(218R1A1222)

In partial fulfillment of the requirement for the award of the degree of **BACHELOR OF TECHNOLOGY** in **INFORMATION TECHNOLOGY** from CMR Engineering College, affiliated to JNTU, Hyderabad, under our guidance and supervision.

The results presented in this project have been verified and are found to be satisfactory. The results embodied in this project have not been submitted to any other university for the award of any other degree or diploma.

Internal Guide
Mr. K. ANIL
Assistant Professor
Department of IT
CMREC, Hyderabad

Head of the Department
Dr. MADHAVI PINGILI
Professor & HOD
Department of IT
CMREC, Hyderabad

DECLARATION

This is to certify that the work reported in the present project entitled **“Secure E-Voting System Using Ethereum Blockchain Technology And Smart Contracts”** is a record of bonafide work done by us in the Department of Information Technology, CMR Engineering College, JNTU Hyderabad. The reports are based on the project work done entirely by us and not copied from any other source. We submit our project for further development by any interested students who share similar interests to improve the project in the future.

The results embodied in this project report have not been submitted to any other University or Institute for the award of any degree or diploma to the best of our knowledge and belief.

BURRA MALATHI	(218R1A1213)
BOKKENA SANGHAVI	(218R1A1212)
BATHULA CHARAN THEJA	(218R1A1211)
GADDE SHIVA	(218R1A1222)

ACKNOWLEDGEMENT

We are extremely grateful to **Dr. A. Srinivasula Reddy**, Principal and **Dr. Madhavi Pingili**, HOD, **Department of IT, CMR Engineering College** for their constant support.

We are extremely thankful to **Mr. K. ANIL**, Assistant Professor, Internal Guide, Department of IT, for his constant guidance, encouragement and moral support throughout the project.

We will be failing in duty if we do not acknowledge with grateful thanks to the authors of the references and other literatures referred in this Project.

We express our thanks to all staff members and friends for all the help and co-ordination extended in bringing out this project successfully in time.

Finally, we are very much thankful to our parents who guided us for every step.

BURRA MALATHI	(218R1A1213)
BOKKENA SANGHAVI	(218R1A1212)
BATHULA CHARAN THEJA	(218R1A1211)
GADDE SHIVA	(218R1A1222)

CONTENTS

TOPIC	PAGE NO
ABSTRACT	I
LIST OF FIGURES	II
LIST OF TABLES	III
1. INTRODUCTION	1
1.1. Introduction	1
1.2. Project Objectives	2
1.3. Purpose of the project	2
1.4. Existing System with Disadvantages	2
1.5. Proposed System with features	3
1.6. Input and Output Design	5
2. LITERATURE SURVEY	8
3. SOFTWARE REQUIREMENT ANALYSIS	12
3.1. Problem Specification	12
3.2. Modules and their Functionalities	12
3.3. Functional Requirements	13
3.4. Non-Functional Requirements	13
3.5. Feasibility Study	14
4. SOFTWARE & HARDWARE REQUIREMENTS	16
4.1. Software requirements	16
4.2. Hardware requirements	16
5. SOFTWARE DESIGN	17
5.1. System Architecture	17
5.2. Dataflow Diagrams	17
5.3. UML Diagrams	19

6. CODING AND IMPLEMENTATION	24
6.1. Source code	24
6.2. Implementation	28
6.2.1. Python	28
6.2.2. Modules used in project	29
7. SYSTEM TESTING	31
7.1. Types of System Testing	31
7.2. Test Cases	35
8. OUTPUT SCREENS	39
9. CONCLUSION	44
10. FUTURE ENHANCEMENTS	45
11. REFERENCES	46

ABSTRACT

In the digital era where hacking and bypassing a system is easy, tampering of data is always possible which causes data loss or financial loss at larger scale. Blockchain is used to store data which is nearly impossible to change or tamper with as it is much secured in nature. Election is a process of establishing Government in the Democratic Country by voting which is an essential event and if votes get miscalculated or tampered by any external source it will leads to chaotic situations in the Country disrupting the peace and harmony and also effects the entire democratic system. To avoid such kind of situations and making it more secure blockchain technology comes into picture. This project proposes a decentralized national e-voting system based on blockchain technology. It includes an admin panel to schedule the voting, manage the candidates and declare the results. The web application will provide the users with an interface to enter their credentials and a photo of user at the time of voting. The eligibility of the voter will be checked immediately after entering their credentials. During voting, voters will be monitored through a webcam/front camera. The votes will be stored in a blockchain and any kind of tampering would be easily detected. The address of the voter and his respective constituency will be checked in the backend. This project aims at identifying the strategies and the guidelines as well as providing a comprehensive end-to-end electronic voting system based on blockchain by the help of cryptographic techniques such as zero-knowledge proofs to improve privacy. Our purpose is to provide key elements for organizations to design a secured electronic voting system based on blockchain technology. It proposes a novel electronic voting system based on blockchain that addresses some of the limitations in existing systems and evaluates some of the popular blockchain frameworks for the purpose of constructing a blockchain - based e - voting system. In particular, we evaluated the potential of distributed ledger technologies through the description of a case study; namely, the process of an election, and the implementation of a blockchain - based application, which improves the security and also decreases the cost of conducting a election nationwide. We propose a cryptographic technique for an authenticated, end-to-end verifiable and a secret ballot election.

Keywords: Electronic Voting, Blockchain, Zero Knowledge Proofs, Ethereum, Smart Contracts, Authentication, Distributed Ledger.

LIST OF FIGURES

S.NO	FIGURE NO	DESCRIPTION	PAGE NO
1	5.1	System Architecture	17
2	5.2	Data Flow diagram	18
3	5.3.1	Class diagram	20
4	5.3.2	Use case diagram	21
5	5.3.3	Sequence diagram	22
6	5.3.4	Activity diagram	23
7	7.2.1	Test Case 1	36
8	7.2.2	Test Case 2	36
9	7.2.3	Test Case 3	37
10	7.2.4	Test Case 4	37
11	7.2.5	Test Case 5	38
12	8.1	Homepage of e- voting	39
13	8.2	Admin login page	39
14	8.3	Admin Homepage	40
15	8.4	Add party candidate page	40
16	8.5	View votes page	41
17	8.6	Voter registration page	41
18	8.7	User image capturing	42
19	8.8	Sign up process page	42
20	8.9	Cast your vote page	43

LIST OF TABLES

S.NO	TABLE NO	DESCRIPTION	PAGE NO
1	7.2	Test Cases	35

1. INTRODUCTION

1.1 Introduction

In modern democracies, ensuring the integrity and security of voting processes is crucial. Traditional voting systems, whether paper-based or electronic, face various challenges, including the risks of fraud, tampering, and lack of transparency. With the rise of digital transformation, there is a growing need for more secure, transparent, and efficient voting systems. There is no doubt that the revolutionary concept of the blockchain, which is the underlying technology behind the famous cryptocurrency Bitcoin and its successors, is triggering the start of a new era in the Internet and the online services. While most people focus only at cryptocurrencies; in fact, many administrative operations, fintech procedures, and everyday services that can only be done offline and/or in person, can now safely be moved to the Internet as online services. What makes it a powerful tool for digitalizing everyday services is the introduction of smart contracts, as in the Ethereum platform. Smart contracts are meaningful pieces of codes, to be integrated in the blockchain and executed as scheduled in every step of blockchain updates. Ethereum, a leading blockchain platform, supports smart contracts that automate and enforce complex processes in a secure and transparent manner. Leveraging these technologies for e-voting can revolutionize the way elections are conducted, providing enhanced security, transparency, and trust. In the digital age, the integrity, transparency, and efficiency of voting processes are more crucial than ever. Traditional voting systems, whether paper-based or electronic, have faced numerous challenges including fraud, tampering, and inefficiencies. E-voting on the other hand, is another trending yet critical, topic related to the online services. The blockchain with the smart contracts, emerges as a good candidate to use in developments of safer, cheaper, more secure, more transparent, and easier-to-use e-voting systems. Ethereum and its network is one of the most suitable ones, due to its consistency, widespread use, and provision of smart contracts logic. Blockchain technology, particularly Ethereum, offers a decentralized, immutable ledger that can fundamentally transform how votes are cast, recorded, and verified. By leveraging Ethereum's blockchain and smart contracts, it is possible to create a secure, transparent, and efficient e-voting system that addresses many of the shortcomings of existing voting methods. An e-voting system must be secure, as it should not allow duplicate votes and be fully transparent, while protecting the privacy of the voters. In this project, we have implemented and tested a sample e-voting application using Ethereum Blockchain Technology and Smart Contracts.

1.2 Project Objectives

The objective of this project is to develop a secure e-voting system utilizing Ethereum blockchain technology and smart contracts. This system aims to provide a transparent and tamper-proof voting process, ensuring that every vote is accurately recorded and that the election results are only verifiable by admins.

1.3 Purpose of the Project

The purpose of developing a secure e-voting system using Ethereum blockchain technology and smart contracts is multifaceted, addressing several critical issues in traditional voting systems and leveraging the strengths of blockchain technology to enhance the electoral process.

1.4 Existing System with Disadvantages

Current voting systems, whether traditional paper-based, electronic, or hybrid, have several limitations that can undermine the integrity, efficiency, and accessibility of the electoral process. Below are some of the common disadvantages associated with existing voting systems:

1. Traditional Paper-Based Voting

- **Risk of Fraud and Tampering:**
 - **Ballot Manipulation:** Paper ballots can be altered or destroyed, leading to potential vote tampering or fraud.
 - **Counting Errors:** Manual counting of paper ballots is prone to human error and can be influenced by biases.
- **Lack of Transparency:**
 - **Limited Verification:** The process of counting and verifying paper ballots is often opaque, with limited mechanisms for independent verification.
 - **Fraud Detection:** The absence of a digital trail makes it difficult to detect and address fraudulent activities effectively.
- **Operational Challenges:**
 - **Logistical Issues:** Handling, storing, and transporting paper ballots can be cumbersome and expensive, especially in large-scale elections.
 - **Voter Accessibility:** Paper-based systems may not be accessible to all voters, particularly those with disabilities or those who are geographically distant from polling stations.

2. Electronic Voting Machines (EVMs)

- **Security Vulnerabilities:**
 - **Hacking Risks:** EVMs can be susceptible to cyber-attacks or tampering if not properly secured.
 - **Software Flaws:** Vulnerabilities in the software or firmware of EVMs can potentially be exploited to manipulate election outcomes.
- **Lack of Auditability:**
 - **Opaque Systems:** Many EVMs do not provide a verifiable paper trail, making it challenging to audit and confirm the accuracy of the recorded votes.
 - **Technical Failures:** Malfunctions or errors in electronic systems can lead to incorrect vote counts or system failures.
- **Accessibility and Usability Issues:**
 - **Complex Interfaces:** EVMs may have complex interfaces that can be difficult for some voters to use effectively.
 - **Maintenance and Support:** EVMs require regular maintenance and technical support, which can be costly and logistically challenging.

1.5 Proposed System with Features

The proposed secure e-voting system leverages Ethereum blockchain technology and smart contracts to provide a modern, reliable, and transparent voting solution. This system aims to address the shortcomings of traditional and existing electronic voting systems by offering enhanced security, transparency, and efficiency. In this project, we present a blockchain based e-voting platform, which can be used for any kind of voting. It is fully utilized by blockchain and all processes can be handled within it. After the start of the voting, the platform behaves as fully independent and decentralized without possibilities to affect the voting process. The data are fully transparent, but the identity of voters is secured by SHA-256 encryption. We have tested and compared our solution in three different blockchains. The results show, that both public and private blockchains can be used with only a little difference in the speed. The key novelty of our solution is a fully decentralized management of e-voting platform through blockchain, transparency of the whole process and at the same time security and privacy of the voters thanks to SHA-256 encryption.

Advantages of a Secure E-Voting System

Implementing a secure e-voting system using Ethereum blockchain technology and smart contracts offers numerous advantages over traditional voting methods and even other electronic voting

systems. Here's a comprehensive look at the benefits:

1. Enhanced Security

- **Immutability:** Once votes are recorded on the blockchain, they cannot be altered or deleted. This immutability ensures that the vote count remains accurate and prevents tampering or fraud.
- **Cryptographic Protection:** SHA-256 and other cryptographic techniques used in Ethereum ensure that data is securely encrypted, protecting it from unauthorized access and manipulation.

2. Transparency and Trust

- **Public Ledger:** Ethereum's blockchain is a public ledger that allows anyone to verify the results of the election. This transparency fosters trust among voters and stakeholders, as the entire voting process can be audited by anyone.
- **Real-Time Tracking:** Voters and election authorities can track the status of votes in real-time, providing immediate visibility into the voting process and results.

3. Automation and Efficiency

- **Smart Contracts:** Smart contracts automate key aspects of the voting process, including voter registration, vote casting, and result tabulation. This automation reduces the risk of human error and ensures that election rules are consistently enforced.
- **Reduced Administrative Burden:** By automating processes and minimizing manual intervention, the administrative workload is significantly reduced, making the overall election process more efficient.

SHA-256 Algorithm in Blockchain

SHA-256 (Secure Hash Algorithm 256-bit) is integral to the functioning of blockchain technology. Its role is pivotal in maintaining the security, integrity, and immutability of blockchain systems. SHA-256 is a cryptographic hash function that produces a fixed-size 256-bit hash value from an input of arbitrary size. Its key properties include:

- **Deterministic:** The same input always produces the same hash output.
- **Fast Computation:** Efficient to compute.
- **Pre-image Resistance:** It is infeasible to retrieve the original input from the hash output.
- **Collision Resistance:** It is infeasible to find two different inputs that produce the same hash output.
- **Avalanche Effect:** A slight change in input drastically changes the output.

SHA-256 Algorithm Steps

The SHA-256 algorithm processes data in a series of steps. Here's a high-level overview:

1. Message Padding

- **Initial Padding:** The input message is padded to ensure its length is congruent to 448 modulo 512. Padding involves appending a single 1 bit followed by a series of 0 bits.
- **Length Encoding:** Append a 64-bit representation of the original message length to the end of the padded message. The total length of the padded message must be a multiple of 512 bits.

2. Initialize Hash Values

- SHA-256 initializes eight 32-bit hash values (H0 to H7) with specific constants. These initial values are derived from the fractional parts of the square roots of the first eight prime numbers.

3. Process Message in Blocks

- The padded message is divided into 512-bit blocks. Each block is further divided into 16 words of 32 bits each.
- **Message Schedule:** Extend these 16 words into 64 words using bitwise operations and constants.
- **Compression Function:** Each block is processed through 64 rounds of operations, which include bitwise logical operations, modular additions, and the application of constants. This involves updating the hash values (H0 to H7) using the extended message schedule.

4. Compute Final Hash

- After processing all blocks, the final hash value is obtained by concatenating the eight 32-bit hash values into a 256-bit (32-byte) output.

1.6 Input and Output Design

Input Design

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The

input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

Objectives

- Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.
- It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.
- When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus, the objective of input design is to create an input layout that is easy to follow.

Output Design

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

- Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.
- Select methods for presenting information.
- Create document, report, or other formats that contain information produced by the system.

- The output form of an information system should accomplish one or more of the following objectives.
- Convey information about past activities, current status or projections of the Future.
- Signal important events, opportunities, problems, or warnings.
- Trigger an action.
- Confirm an action.

2. LITERATURE SURVEY

Hajian Berenjestanaki, M.; Barzegar, H.R.; El Ioini, N.; Pahl, C. Blockchain-Based E-Voting Systems: A Technology Review. Electronics 2024, 13, 17.

The evolution of blockchain-based e-voting systems from 2017 to 2023 has been marked by significant advancements, as evidenced by research papers from this period. Significant studies emerged, proposing a novel approach to utilizing blockchain technology for recording votes for different voting scenarios.

Hossain Faruk, M.J., Alam, F., Islam, M. et al. Transforming online voting: a novel system utilizing blockchain and biometric verification for enhanced security, privacy, and transparency. Cluster Comput (2024).

A novel, web-based online voting system that utilizes blockchain technology and biometric identification techniques to improve the security, privacy, and transparency of elections. The results of the evaluation demonstrate that the proposed system provides seamless access to voters to conduct online voting.

El Kafhali, "Blockchain-Based Electronic Voting System: Significance and Requirements", Mathematical Problems in Engineering, vol. 2024, Article ID 5591147, 17 pages, 2024.

This paper mostly focuses on a review study of blockchain-based voting systems. It aims at identifying the strategies and the guidelines as well as provides a comprehensive end-to-end electronic voting system based on blockchain, with the help of cryptographic techniques such as zero-knowledge proofs to improve privacy.

Y. A. F. Ali, O. T. M. Ahmed, M. A. M. Diab, M. A. E. Sayed, M. K. Abd Elaziz and B. W. Aboshosha, "Blockchain-Based Online E-voting System," 2023 International Conference on Smart Computing and Application (ICSCA), Hail, Saudi Arabia, 2023, pp. 1-8, doi: 10.1109/ICSCA57840.2023.10087767.

Smart contracts are being employed in a new blockchain based voting system that uses machine learning to secure the privacy of voters. The various obstacles that limit the widespread use of electronic voting systems can be removed using this technology. Additionally, it creates a level playing field for election transparency.

S. N, G. S, S. E and V. K, "E-Voting Using Blockchain," 2023 2nd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA), Coimbatore, India, 2023, pp. 1-4,

The Blockchain technology, introduced by Satoshi Nakamoto using the cryptographic currency Bitcoin in 2008, opens up possibilities of designing and developing a secure, transparent and

decentralized system with the absence of a third party for access and control, in the election procedure of casting and counting of votes.

A. -M. Stanciu, H. Ciocârlie and C. -P. Julean, "Electronic Voting System Based on the Blockchain Technology," 2023 International Conference on Electrical, Computer and Energy Technologies (ICECET), Cape Town, South Africa, 2023, pp. 1-6, doi: 10.1109/ICECET58911.2023.10389415.

Blockchain technology has shown promise in revolutionizing the voting process by providing transparency, immutability, and decentralization. It offers potential solutions to address some of the challenges associated with traditional voting systems, such as fraud, tampering, and lack of trust.

S. Al-Maaitah, M. Qatawneh and A. Quzmar, "E-Voting System Based on Blockchain Technology: A Survey," 2021 International Conference on Information Technology (ICIT), Amman, Jordan, 2021, pp. 200-205, doi: 10.1109/ICIT52682.2021.9491734.

Trust in the E-voting systems is considered vital and should be used to reduce fraud during the election process. The BC technology is most preferred to tackle this problem and aid in this context by tracking each step and making sure the whole process is very secure.

Benny, Albin, Blockchain-based E-voting System (July 11, 2020).

This utilizes smart contracts to enable secure and cost-effective elections while guaranteeing voters' privacy. They have shown that blockchain technology offers a new possibility to overcome the limitations and adoption barriers of electronic voting systems which ensures election security and integrity and lays the ground for transparency.

S. Drakshayani, U. Vijayalakshmi, S. R. Sri, A. Srivani and A. Vyshnavi, "Online Voting System Using Blockchain," 2022 International Conference on Electronics and Renewable Systems (ICEARS), Tuticorin, India, 2022, pp. 886-891, doi: 10.1109/ICEARS53579.2022.9752044.

The concept of E-casting ballot layout attacks has now gained traction in urban areas with sophisticated surveying methods. Every element's trust is processed and stored in a Blockchain so that their consistent behavior can be examined when thought about, thus resolving the security and protection flaws. An online voting system using blockchain technology offers significant advancements over traditional voting methods by enhancing security, transparency, and accessibility.

M. J. H. Faruk et al., "Development of Blockchain-based e-Voting System: Requirements, Design and Security Perspective," 2022 IEEE International Conference on Trust, Security

and Privacy in Computing and Communications (TrustCom), Wuhan, China, 2022, pp. 959-967, doi: 10.1109/TrustCom56396.2022.00132.

It proposes a framework for voting systems by adopting Hyperledger Fabric, a permissioned blockchain. To identify and validate the voters, biometric identification technology was utilized including fingerprint and face recognition. It is also more convenient due to efficiency where voters can vote simply by accessing a browser.

Yaga, D.; Mell, P.; Roby, N.; Scarfone, K. Blockchain technology overview. arXiv 2019, arXiv:1906.11078.

Blockchains are tamper evident and tamper resistant digital ledgers implemented in a distributed fashion (i.e., without a central repository) and usually without a central authority (i.e., a bank, company, or government). At their basic level, they enable a community of users to record transactions in a shared ledger within that community, such that under normal operation of the blockchain network no transaction can be changed once published. This document provides a high-level technical overview of blockchain technology.

Schinckus, C. The good, the bad and the ugly: An overview of the sustainability of blockchain technology. Energy Res. Soc. Sci. 2020, 69, 101614.

Blockchain technology has the promise to change all existing business models and make financial services cheaper contributing therefore to a better financial inclusion and, even a better economic wealth distribution. Numerous studies optimistically praise such potential societal benefits by listing all processes that could be optimized through this technology.

Kim, T.; Ochoa, J.; Faika, T.; Mantooth, A.; Di, J.; Li, Q.; Lee, Y. An overview of cyberphysical security of battery management systems and adoption of blockchain technology. IEEE J. Emerg. Sel. Top. Power Electron. 2020.

Blockchain is a buzzword describing the current excitement for an innovative technology that could change and disrupt major industries and economic sectors. Blockchain technology has the promise to change all existing business models and make financial services cheaper contributing therefore to a better financial inclusion and, even a better economic wealth distribution.

Chang, V.; Baudier, P.; Zhang, H.; Xu, Q.; Zhang, J.; Arami, M. How Blockchain can impact financial services—The overview, challenges and recommendations from expert interviewees. Technol. Forecast. Soc. Chang. 2020, 158, 120166.

This article suggests how financial services should respond to this new technology and how to manage knowledge sharing in a more structured way. This article contributes to the literature related to the current entrepreneurial finance landscape for Blockchain.

Gao, S.; Zheng, D.; Guo, R.; Jing, C.; Hu, C. An Anti-Quantum E-Voting Protocol in Blockchain with Audit Function. IEEE Access 2019, 7, 115304–115316.

As an important method of making democratic decisions, voting has always been a topic of social concern. Compared with the traditional, e-voting is widely used in various decision scenarios because of the convenience, easy to participate and low cost.

Racsko, P. Blockchain and Democracy. Soc. Econ. 2019, 41, 353–369.

It aimed to explore whether the blockchain technology is suitable for voting or elections in large communities and the issues to be addressed for real world applications to leverage democratic rights. Our final conclusion is that there are both theoretical and practical obstacles in the way of such direct applications.

Hang, L.; Kim, D.-H. Design and implementation of an integrated iot blockchain platform for sensing data integrity. Sensors 2019, 19, 2228.

With the rapid development of communication technologies, the Internet of Things (IoT) is getting out of its infancy, into full maturity, and tends to be developed in an explosively rapid way, with more and more data transmitted and processed.

Liu, Y.; Wang, Q. An E-voting Protocol Based on Blockchain. IACR Cryptol. Eprint Arch. 2017, 2017, 1043.

In this paper, based on the blockchain technology, it proposes a decentralized e-voting protocol, without the existence of a trusted third party. Furthermore, we provide several possible extensions and improvements that meet the requirements in some specific voting scenarios.

Shahzad, B.; Crowcroft, J. Trustworthy Electronic Voting Using Adjusted Blockchain Technology. IEEE Access 2019, 7, 24477–24488.

Electronic voting (e-voting), which uses electronic systems to aid casting and counting votes in an election, has been a research topic of interest for the past few decades in cryptography. In comparison with the traditional paper-based voting, remote e-voting is environmentally friendly, real-time counting and processing, less error-prone

RaCullen, R.; Houghton, C. Democracy online: An assessment of New Zealand government web sites. Gov. Inf. Q. 2000, 17, 243–267

Blockchains are tamper evident and tamper resistant digital ledgers implemented in a distributed fashion (i.e., without a central repository) and usually without a central authority (i.e., a bank, company, or government). At their basic level, they enable a community of users to record transactions in a shared ledger within that community, such that under normal operation of the blockchain network no transaction can be changed once published.

3. SOFTWARE REQUIREMENTS ANALYSIS

3.1 Problem Statement

This project aims at E-voting systems which have gained prominence as an alternative to traditional paper-based voting, offering the potential for increased efficiency, accessibility, and convenience. However, traditional e-voting systems face significant challenges related to security, transparency, and trust. Issues such as vote tampering, data breaches, lack of transparency, and administrative inefficiencies continue to plague many existing systems. Blockchain technology, particularly Ethereum, presents an opportunity to address these challenges through its decentralized, immutable, and transparent nature. By leveraging Ethereum's blockchain and smart contracts, it is possible to design an e-voting system that enhances security, ensures transparency, and improves the overall voting process.

3.2 Modules and Their Functionalities

Data Preprocessing

After collecting datasets from various resources. Dataset must be pre-processing before training to the model. The data pre-processing can be done by various stages, begins with reading the collected dataset the process continues to data cleaning. In data cleaning the datasets contain some redundant attributes, those attributes are not considering for phishing detection. So, we have to drop unwanted attributes and data sets containing some missing values we need to drop these missing values in order to get better accuracy.

- Getting the dataset
- Importing libraries
- Importing datasets
- Splitting dataset into training and test set
- Train the classifier
- Test the classifier
- Evaluate

Splitting the Dataset into the Training set and Test set

In machine learning data pre-processing, we divide our data set into a training set and testset. This is one of the crucial steps of data pre-processing as by doing this; we can enhance the performance of our machine learning model. Suppose if we have given training to our machine learning model by a data set and we test it by a completely different dataset. Then, it will create difficulties for our

model to understand the correlations between the models. If we train our model very well and its training accuracy is also very high, but we provide a new dataset to it, then it will decrease the performance. So we always try to make a machine learning model which performs well with the training set and also with the test dataset. Here, we can define these datasets as:



Training Set: A subset of dataset to train the machine learning model, and we already know the output.

Test set: A subset of dataset to test the machine learning model, and by using the test set, model predicts the output.

Train the classifier:

Training the classifier with the training data by specifying the value of k . Use $k=3$ for binary classification, i.e., two labels classification. If used $k=1$ then it is simply a nearest neighbor classifier.

Test the classifier: Testing the classifier with the testing data.

Evaluate: Evaluating the classifier using confusion matrix and its evaluation metrics i.e., accuracy, precision, recall, etc.

3.3 Functional Requirements

It provides the users a clear statement of the functions required for the system in order to solve the project information problem it contains a complete set of requirements for the applications. A requirement is condition that the application must meet for the customer to find the application satisfactory. A requirement has the following characteristics:

- It provides a benefit to the origination.
- It describes the capabilities the application must provide in business terms.
- It does not describe how the application provides that capability.
- It is stated in unambiguous words. Its meaning is clear and understandable.
- It is verifiable.

3.4 Non-Functional Requirements

Career recommendation non-functional requirements, like interests he has, how hours he can work likewise, with today's IT projects, to determine non-functional requirements, like availability, the approach requires that the designer 1st determine the scope: does the whole solution or only part of

it need to be architected to meet minimum levels?

- This is done through 4 steps:
- Identify the critical areas of solutions
- Identify the critical components within each critical area.
- Determine each components availability and risk.
- Model worst-case failure scenarios.

3.5 Feasibility Study

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

- Economical Feasibility
- Technical Feasibility
- Social Feasibility

Economical Feasibility

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

Technical Feasibility

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

Social Feasibility

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

4. SOFTWARE AND HARDWARE REQUIREMENTS

4.1 Software Requirements

The functional requirements or the overall description documents include the product perspective and features, operating system and operating environment, graphics requirements, design constraints and user documentation.

The appropriation of requirements and implementation constraints gives the general overview of the project in regard to what the areas of strength and deficit are and how to tackle them.

Operating system	:	Windows 08
Coding Language	:	Python
Tool	:	PyCharm
Database	:	MYSQL
Server	:	Flask
Client Side Technologies	:	HTML, CSS, JavaScript

4.2 Hardware Requirements

Minimum hardware requirements are very dependent on the particular software being developed by a given Enthought Python/Canopy/VS Code user. Applications that need to store large arrays/objects in memory will require more RAM, where as applications that need to perform numerous calculations or tasks more quickly will require a faster processor.

System	:	Intel i3 or above.
Hard Disk	:	40 GB.
FloppyDrive	:	1.44 MB
Monitor	:	15'' LED
Input Devices	:	Keyboard, Mouse
Ram	:	4 GB

5. SOFTWARE DESIGN

5.1 System Architecture

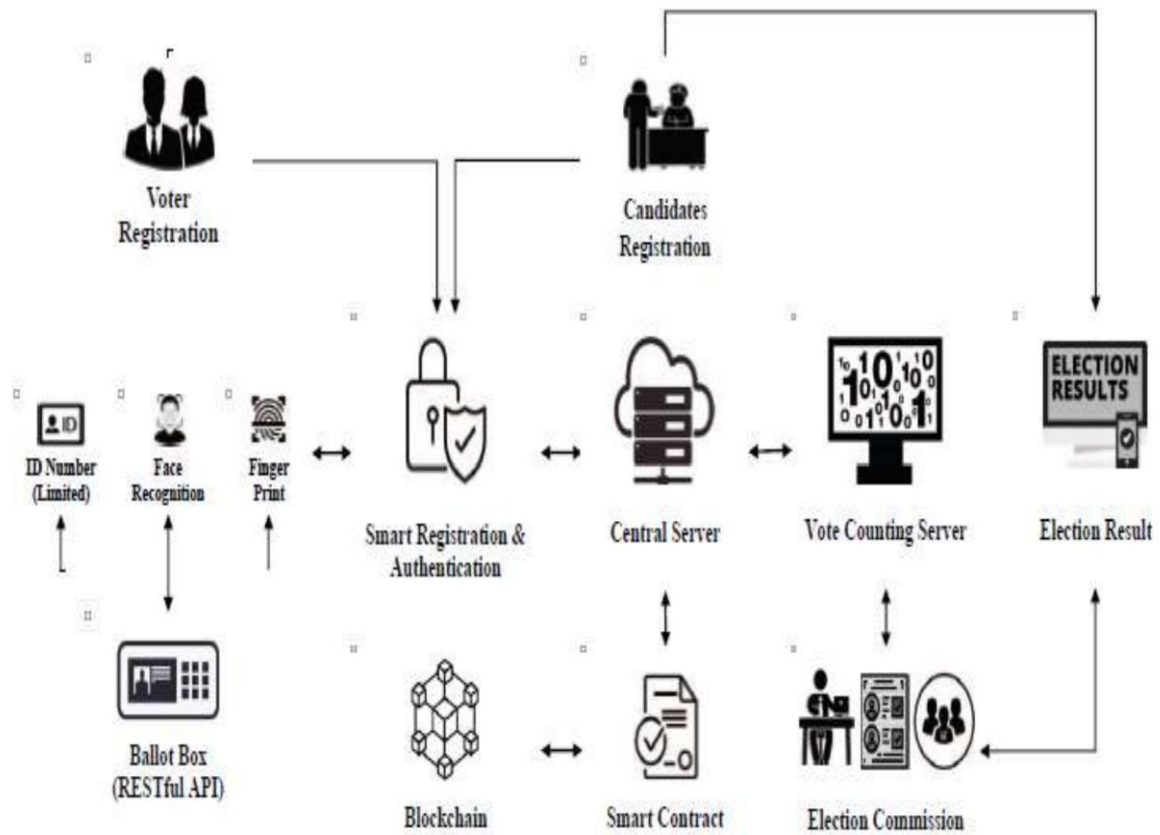


Figure: 5.1 System Architecture

The proposed system architecture for a secure e-voting system using Ethereum blockchain technology and smart contracts aims to create a reliable, transparent, and efficient voting process. By integrating secure interfaces, leveraging blockchain's immutability, and automating processes through smart contracts, the system addresses key challenges in traditional voting methods and provides a modern solution for secure digital elections.

5.2 Data flow Diagram

1. The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.

2. The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
3. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.
4. DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.

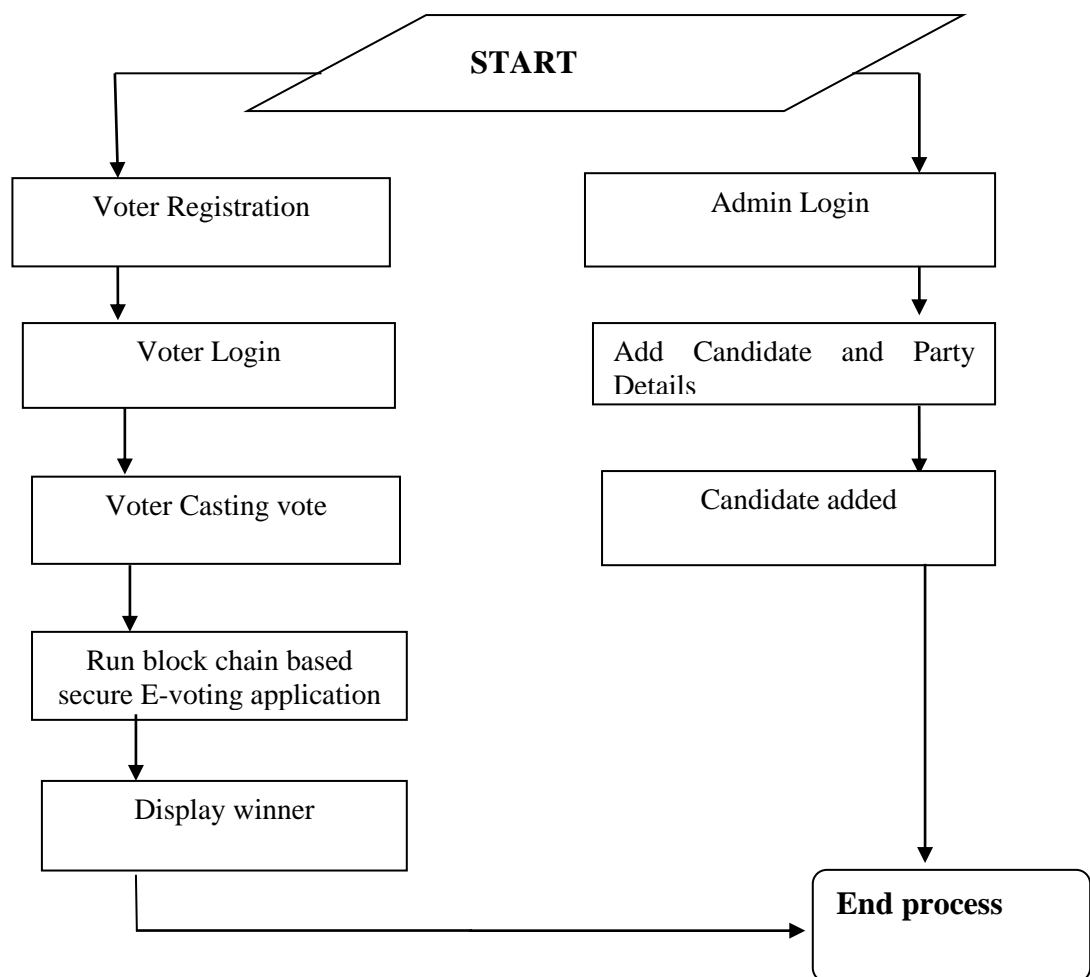


Figure: 5.2 Data flow Diagram

5.3 UML Diagrams

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997. There are several types of UML diagrams and each one of them serves a different purpose regardless of whether it is being designed before the implementation or after (as part of documentation). UML has a direct relation with object-oriented analysis and design. After some standardization, UML has become an OMG standard. The two broadest categories that encompass all other types are:

- Behavioral UML diagram and
- Structural UML diagram.

As the name suggests, some UML diagrams try to analyses and depict the structure of a system or process, whereas other describe the behavior of the system, its actors, and its building components.

Goals: The Primary goals in the design of the UML areas follows:

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

The different types are as follows:

- Class diagram
- Use case diagram
- Sequence diagram
- Activity diagram

Class Diagram

In software engineering, a class diagram in the Unified Modelling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information.

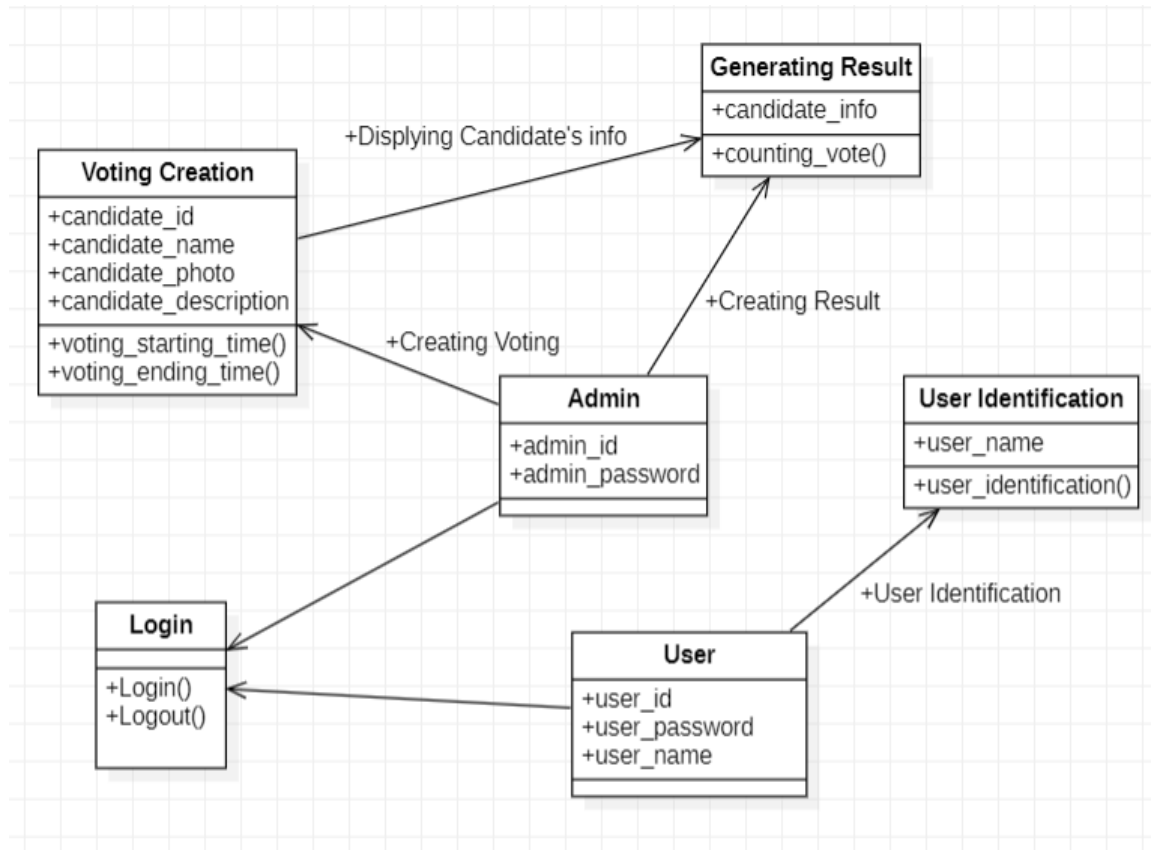


Figure 5.3.1 Class Diagram

Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use case in which the user is involved. A use case diagram is used to structure of the behavior thing in a model. The use cases are represented by either circles or ellipses.

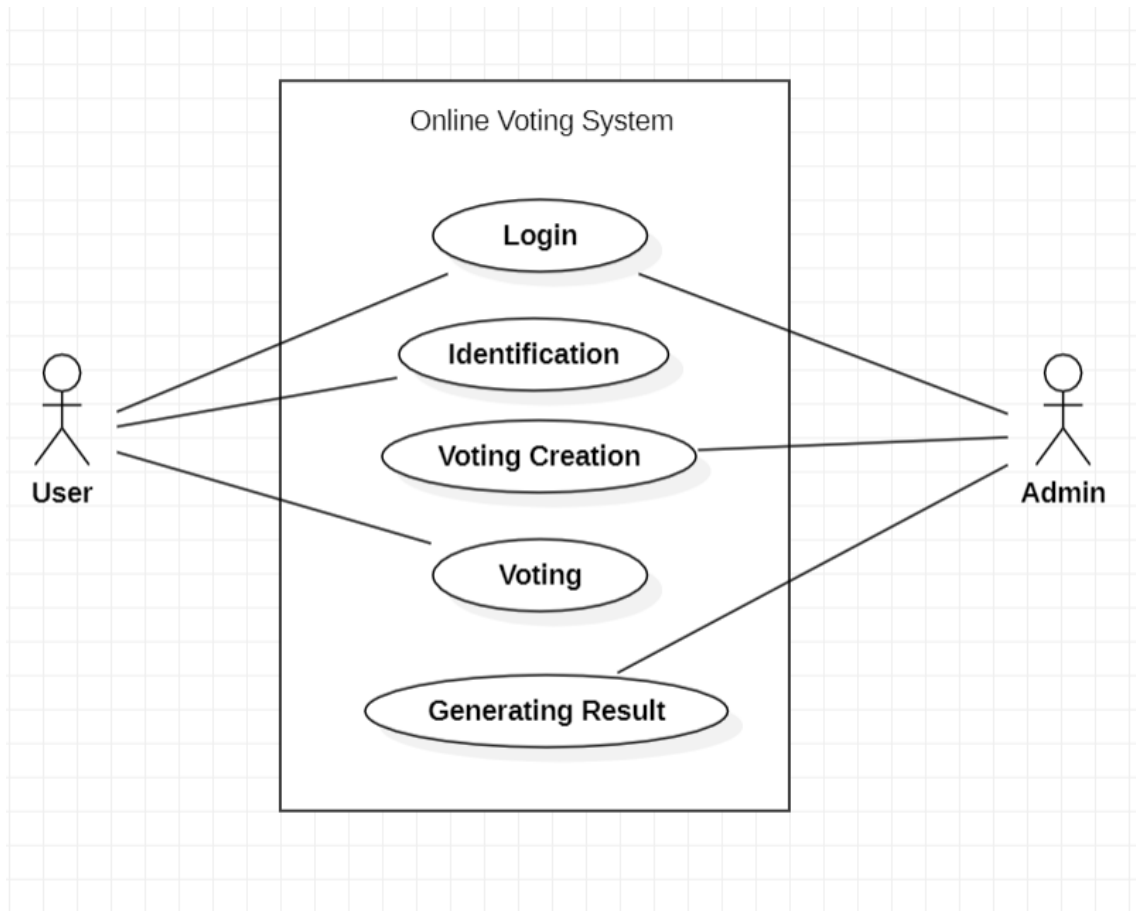


Figure 5.3.2 Use Case Diagram

Sequence Diagram

A sequence diagram simply depicts interaction between objects in a sequential order i.e., the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

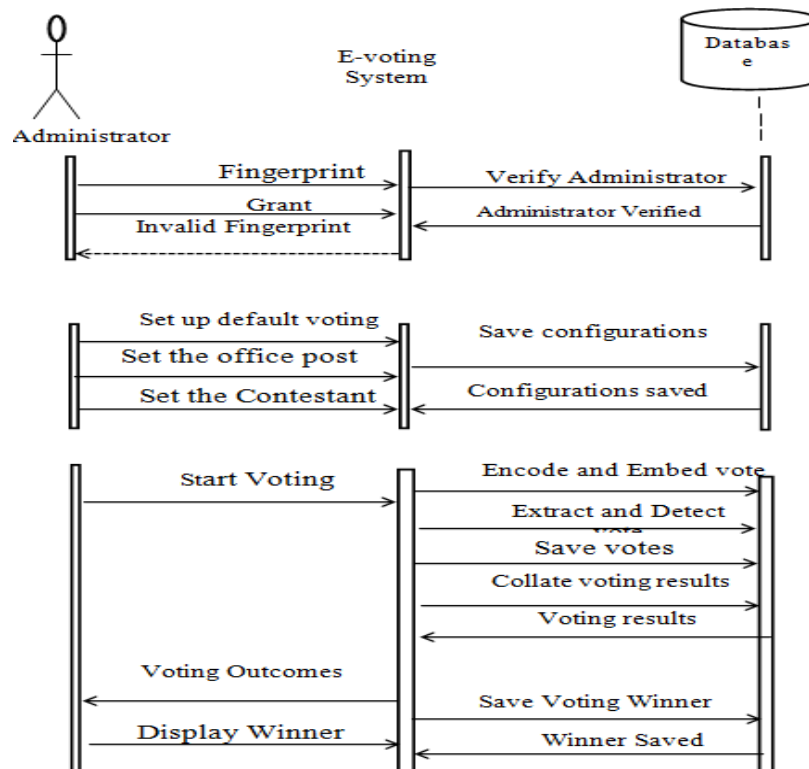


Figure 5.3.3 Sequence Diagram

List of Action

Users:

User need to press any of the given three then he will get the output accordingly.

System:

System will give the output as he enters according to the given data.

Result:

As per maximum no. of users enters data the candidate gets selected.

Activity Diagram

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc.

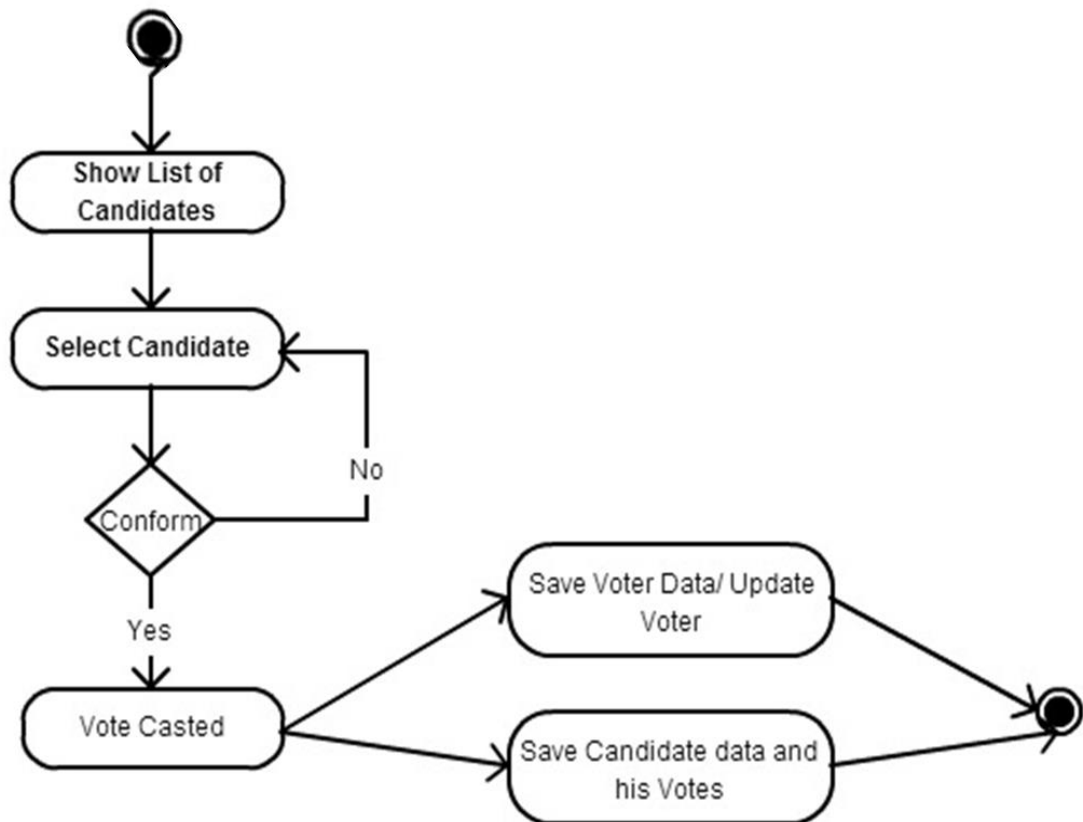


Figure 5.3.4 Activity Diagram

6. CODING AND ITS IMPLEMENTATION

6.1 Source code

```
from hashlib import sha256
import json
import time

class Block:
    def __init__(self, index, transactions, timestamp, previous_hash):
        self.index = index
        self.transactions = transactions
        self.timestamp = timestamp
        self.previous_hash = previous_hash
        self.nonce = 0

    def compute_hash(self):
        """
        A function that return the hash of the block contents.
        """
        block_string = json.dumps(self.__dict__, sort_keys=True)
        return sha256(block_string.encode()).hexdigest()

from hashlib import sha256
import json
import time
import pickle
from datetime import datetime
import random
import pyaes, pbkdf2, binascii, os, secrets
import base64

class Block:
    def __init__(self, index, transactions, timestamp, previous_hash):
```

```

        self.index = index
        self.transactions = transactions
        self.timestamp = timestamp
        self.previous_hash = previous_hash
        self.nonce = 0

    def compute_hash(self):
        block_string = json.dumps(self.__dict__, sort_keys=True)
        return sha256(block_string.encode()).hexdigest()

class Blockchain:
    # difficulty of our PoW algorithm
    difficulty = 2 #using difficulty 2 computation

    def __init__(self):
        self.unconfirmed_transactions = []
        self.chain = []
        self.create_genesis_block()
        self.peer = []
        self.translist = []

    def create_genesis_block(self): #create genesis block
        genesis_block = Block(0, [], time.time(), "0")
        genesis_block.hash = genesis_block.compute_hash()
        self.chain.append(genesis_block)

    @property
    def last_block(self):
        return self.chain[-1]

    def add_block(self, block, proof): #adding data to block by computing new and previous
hashes
        previous_hash = self.last_block.hash

```

```

    if previous_hash != block.previous_hash:
        return False

    if not self.is_valid_proof(block, proof):
        return False

    block.hash = proof
    #print("main "+str(block.hash))
    self.chain.append(block)
    return True

def is_valid_proof(self, block, block_hash): #proof of work
    return (block_hash.startswith('0' * Blockchain.difficulty) and block_hash ==
block.compute_hash())

def proof_of_work(self, block): #proof of work
    block.nonce = 0

    computed_hash = block.compute_hash()
    while not computed_hash.startswith('0' * Blockchain.difficulty):
        block.nonce += 1
        computed_hash = block.compute_hash()

    return computed_hash

def add_new_transaction(self, transaction):
    self.unconfirmed_transactions.append(transaction)

def addPeer(self, peer_details):
    self.peer.append(peer_details)

def addTransaction(self,trans_details): #add transaction

```

```

self.translist.append(trans_details)

def mine(self):#mine transaction
    if not self.unconfirmed_transactions:
        return False

    last_block = self.last_block

    new_block = Block(index=last_block.index + 1,
                       transactions=self.unconfirmed_transactions,
                       timestamp=time.time(),
                       previous_hash=last_block.hash)

    proof = self.proof_of_work(new_block)
    self.add_block(new_block, proof)

    self.unconfirmed_transactions = []
    return new_block.index

def save_object(self,obj, filename):
    with open(filename, 'wb') as output:
        pickle.dump(obj, output, pickle.HIGHEST_PROTOCOL)

#!/usr/bin/env python
import os
import sys

if __name__ == '__main__':
    os.environ.setdefault('DJANGO_SETTINGS_MODULE', 'EVoting.settings')
    try:
        from django.core.management import execute_from_command_line

```

```

except ImportError as exc:
    raise ImportError(
        "Couldn't import Django. Are you sure it's installed and "
        "available on your PYTHONPATH environment variable? Did you "
        "forget to activate a virtual environment?"
    ) from exc
execute_from_command_line(sys.argv)

create database evoting;
use evoting;

create table register(username varchar(30) primary key,
password varchar(30),
contact varchar(12),
email varchar(30),
address varchar(40));
create table addparty(candidatename varchar(30),
partyname varchar(50),
areaname varchar(100),
image varchar(100));

```

6.2 Implementation

6.2.1 Python

Python is an interpreted high-level programming language for general-purpose programming. Created by Guido van Rossum and first released in 1991, Python has a design philosophy that emphasizes code readability, notably using significant white space.

Python features a dynamic type system and automatic memory management. It supports multiple programming paradigms, including object-oriented, imperative, functional and procedural, and has a large and comprehensive standard library.

- Python is Interpreted – Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- Python is Interactive – you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.

Python also acknowledges that speed of development is important. Readable and terse code is part of this, and so is access to powerful constructs that avoid tedious repetition of code. Maintainability also ties into this may be an all but useless metric, but it does say something about how much code you have to scan, read and/or understand to troubleshoot problems or tweak behaviors. This speed of development, the ease with which a programmer of other languages can pick up basic Python skills and the huge standard library is key to another area where Python excels. All its tools have been quick to implement, saved a lot of time, and several of them have later been patched and updated by people with no Python background -without breaking.

6.2.2 Modules Used in Project

NumPy

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

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

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

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

Django

Django is a powerful and high-level Python web framework that enables developers to build robust and scalable web applications quickly and efficiently. It follows the "batteries-included" philosophy, offering a comprehensive suite of tools and features out of the box to streamline web development. Its key features are:

- It follows the Model-Template-View (MTV) architectural pattern.
- It provides a built-in admin interface that allows developers to perform CRUD operations on the database easily.
- Django's ORM allows you to interact with the database using Python code instead of SQL queries, making database operations more intuitive.
- It has a powerful and flexible URL routing system that maps URLs to views,

allowing for clean and readable URL structures.

Django includes built-in protection against common security threats such as SQL injection, cross-site scripting (XSS), cross-site request forgery (CSRF), and more. It is designed to scale and handle large amounts of traffic and data efficiently.

Pytest

Pytest is a popular testing framework for Python that simplifies writing and running tests. It provides powerful features and capabilities for both simple and complex testing scenarios, making it a versatile tool for developers who want to ensure their code is reliable and correct. It is easy to use, simple syntax for writing tests. Tests can be written in plain Python functions with assert statements. It provides a way to set up and tear down test environments, improving test organization and reusability. It allows you to run tests with multiple sets of parameters, making it easy to test various input scenarios. It is extensible through plugins, allowing additional functionality like HTML reports, coverage checks, and more. It has enhanced assertion introspection helps provide detailed and informative failure messages. It automatically discovers and runs tests based on naming conventions and directory structures.

SQLAlchemy

SQLAlchemy is a popular and comprehensive SQL toolkit and Object-Relational Mapping (ORM) library for Python. It provides both a high-level ORM for database interaction and a low-level SQL expression language for more fine-grained control. SQLAlchemy is known for its flexibility, performance, and ability to integrate with various databases. Its key features of SQLAlchemy allows you to map Python classes to database tables and perform database operations using Python objects, abstracting away the complexities of raw SQL. It provides a powerful and flexible way to construct SQL queries using Python code, giving you full control over the generated SQL. It also includes built-in connection pooling to efficiently manage database connections and improve performance. It manages transactions and ensures that database operations are atomic, consistent, isolated, and durable (ACID properties).

Scikit-learn

Scikit-learn is a widely used machine learning library in Python that provides simple and efficient tools for data analysis and modeling. It is built on top of NumPy, SciPy, and matplotlib, and offers a range of functionalities for supervised and unsupervised learning, model selection, and data preprocessing. Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use.

7. SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

7.1. Types of Tests

Unit Testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration Testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional Test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

- Valid Input : identified classes of valid input must be accepted.
- Invalid Input : identified classes of invalid input must be rejected.
- Functions : identified functions must be exercised.
- Output : identified classes of application outputs must be exercised.
- Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. You cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Type of Testing Used

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach:

Field testing will be performed manually and functional tests will be written in detail.

Test objectives:

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

Test strategy and approach:

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test objectives:

- Ensure that different modules or services interact correctly.
- Detect and resolve issues related to the interfaces.
- Identify integration issues as early as possible in the development process.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Functional Testing

Functional testing for blockchain technology which is critical due to the decentralized and often

immutable nature of blockchain systems. Ensuring that the system is secure against a range of threats and vulnerabilities is paramount. It is essential to approach security comprehensively to safeguard against the diverse range of threats and vulnerabilities that can affect blockchain systems.

Test strategy and approach:

Static analysis is made using Mythx tool to analyze smart contract code for common vulnerabilities where as Dynamic analysis is made using Ganache for Ethereum to test the smart contracts in simulated environment.

Test objectives:

- Check that the application produces the correct outputs.
- Verify that the system integrates correctly with external systems.
- Validate that typical user scenarios and workflows operate as intended.

Test Results: All the test cases mentioned above passed successfully. No defects encountered

7.2 Test Cases:

S.no	Test Case	Excepted Result	Result	Remarks(IF Fails)
1.	Data uploading	Initially data need to be uploaded.	Pass	Executed successfully
2.	Data Splitting	After Uploading the Data, it should divide into train and test data.	Pass	Algorithm implemented
3.	Giving Input	Input should be Uploaded.	Pass	Input uploaded successfully
4.	Fraud Detection	After trying to login with same email id and different photographs find out whether it is detecting fraud or not	Pass	If photograph is different from registered, it will show access denied.
5.	Data uploading	Initially data need to be Uploaded.	Fail	Cannot be executed
6.	Fraud Detection	Fraud detection needs to be done.	Fail	Fraud is not detected

Table no 7.2 Test Cases

Test Case 1:

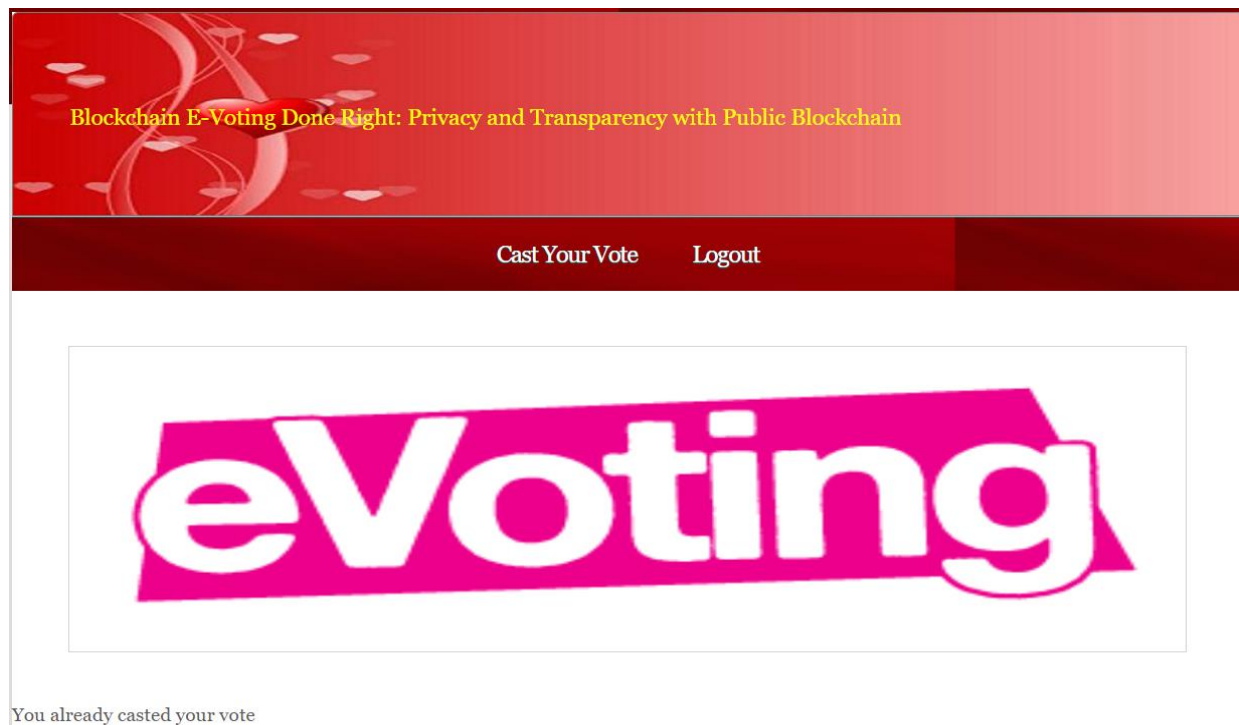


Figure 7.2.1: Test Case 1

Test Case 2:

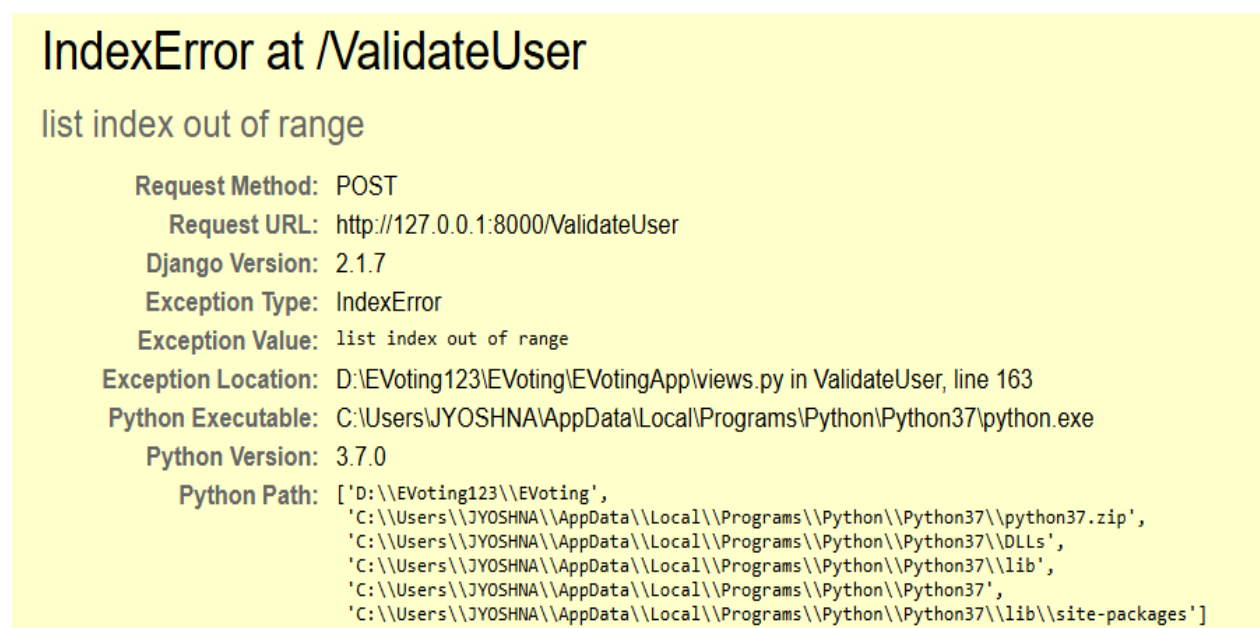
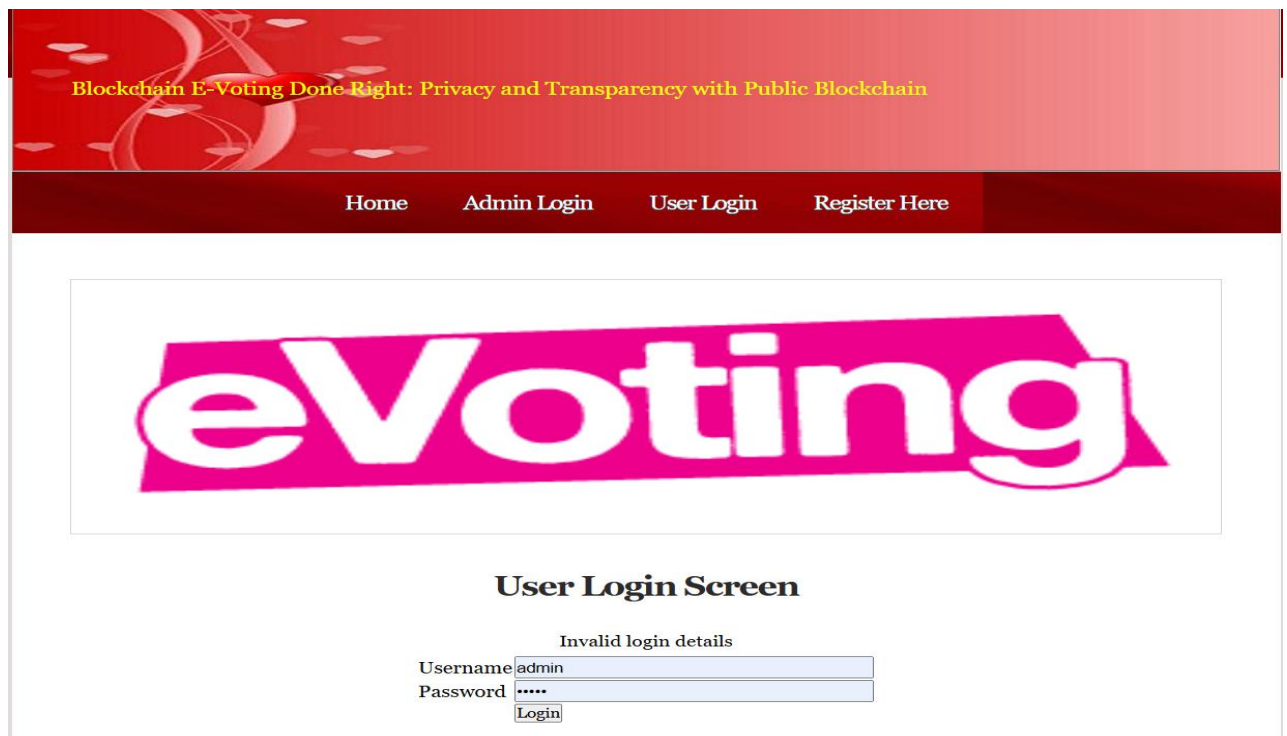


Figure 7.2.2: Test Case 2

Test Case 3:



Blockchain E-Voting Done Right: Privacy and Transparency with Public Blockchain

Home Admin Login User Login Register Here

eVoting

User Login Screen

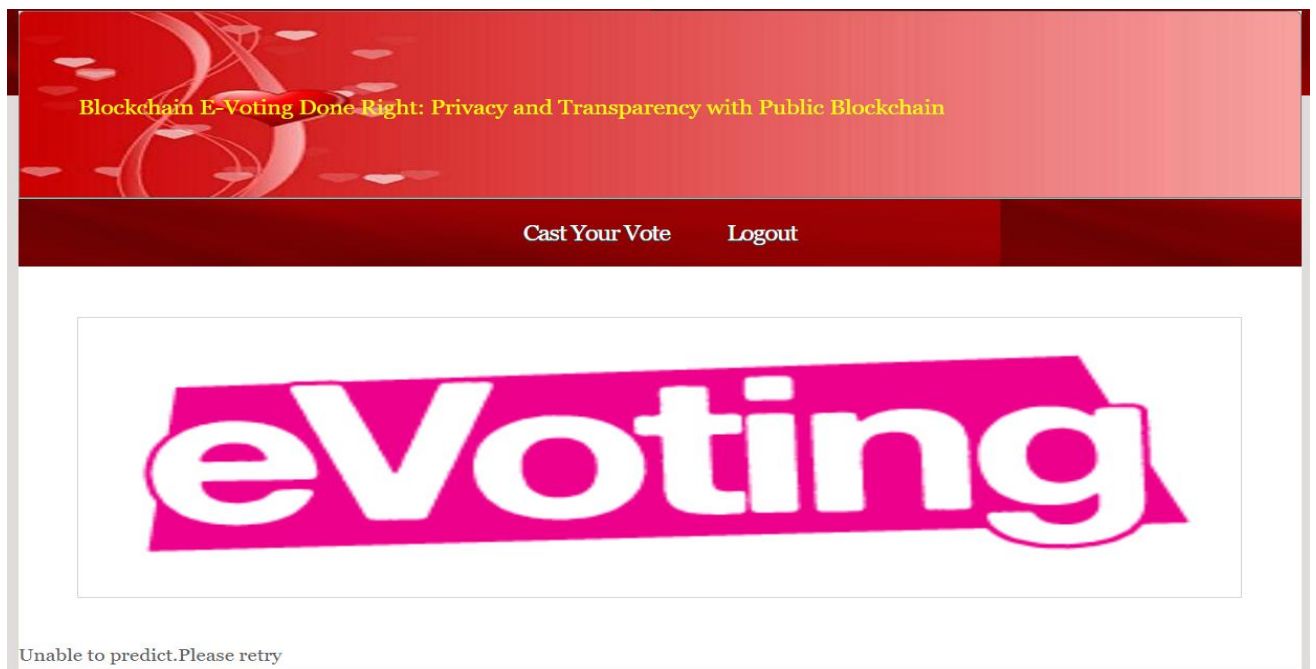
Invalid login details

Username

Password

Figure 7.2.3: Test Case 3

Test Case 4:



Blockchain E-Voting Done Right: Privacy and Transparency with Public Blockchain

Cast Your Vote Logout

eVoting

Unable to predict. Please retry

Figure 7.2.4: Test Case 4

Test Case 5:

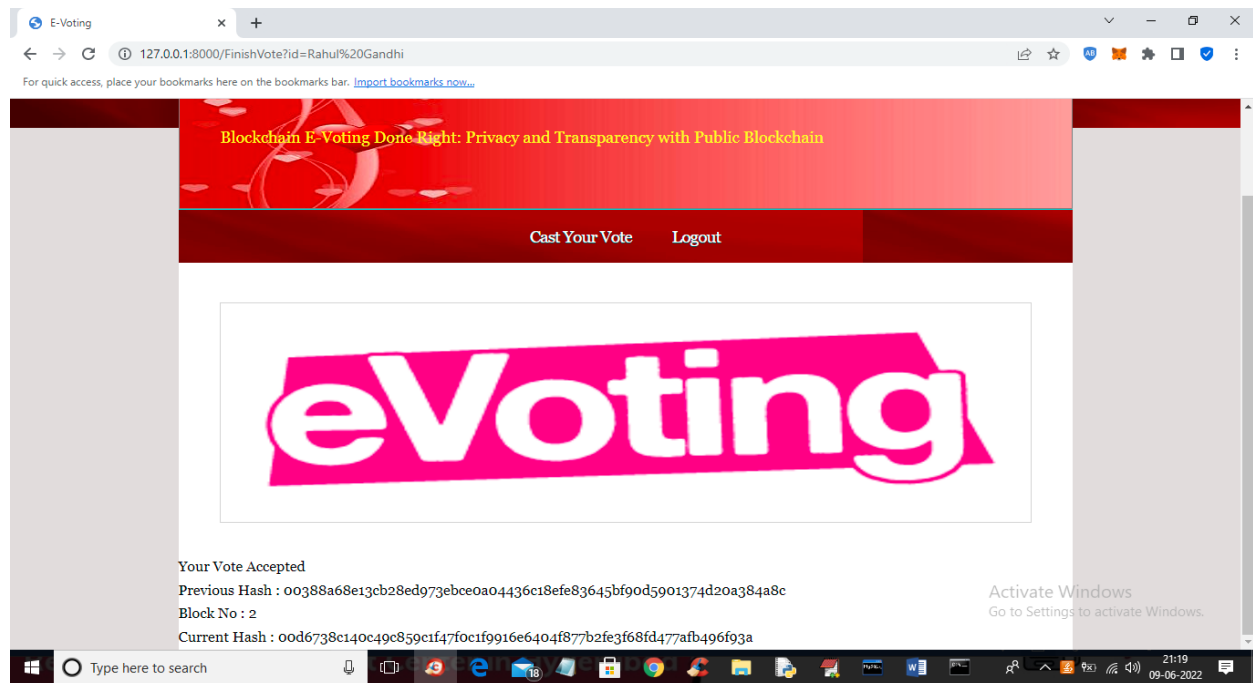


Figure 7.2.5: Test Case 5

8. OUTPUT SCREENS

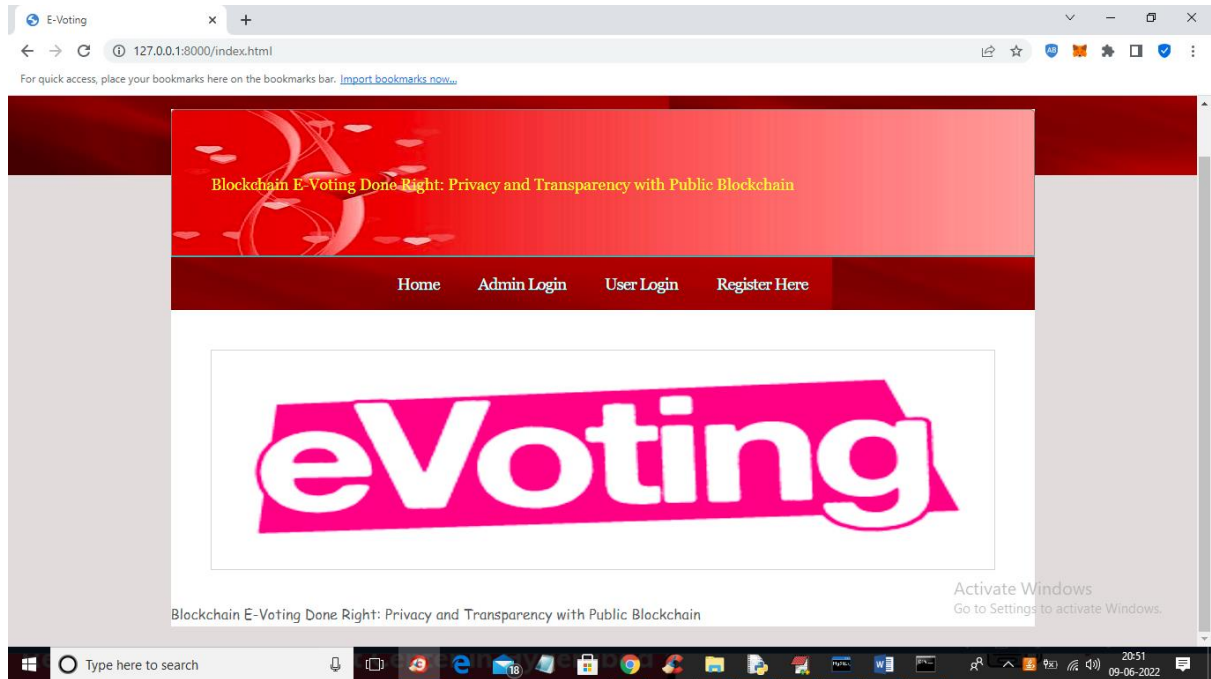


Figure 8.1: Homepage of eVoting

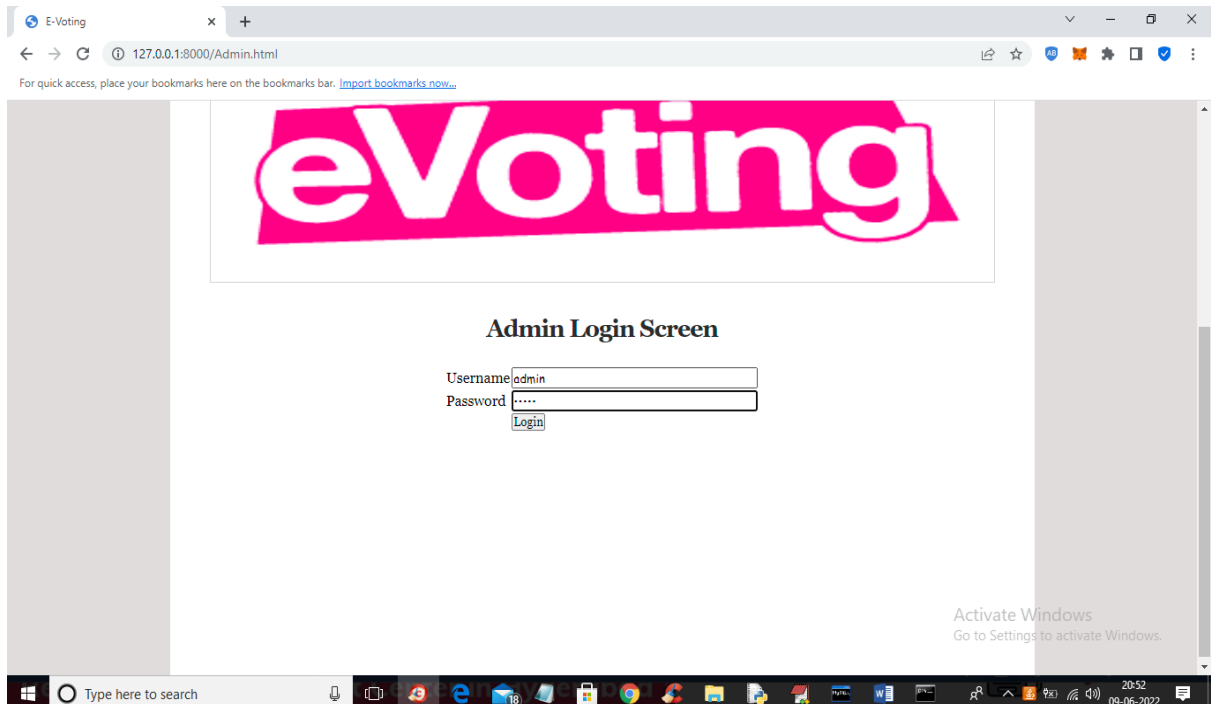


Figure 8.2: Admin Login Page

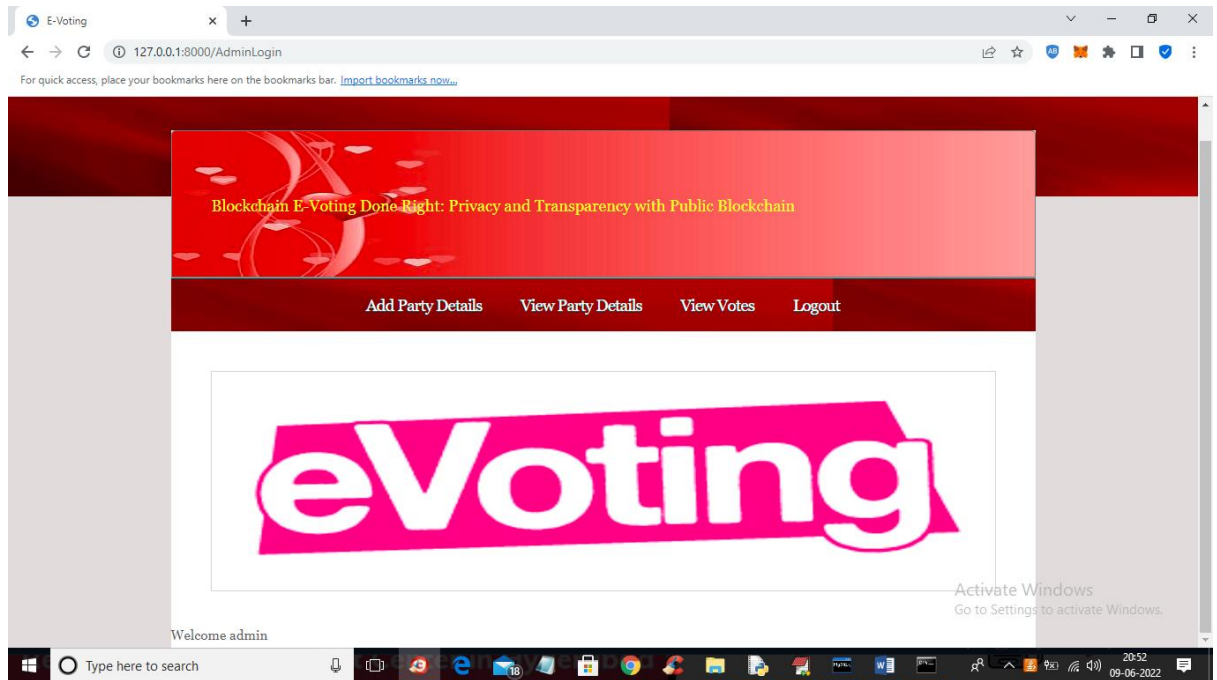


Figure 8.3: Admin Homepage

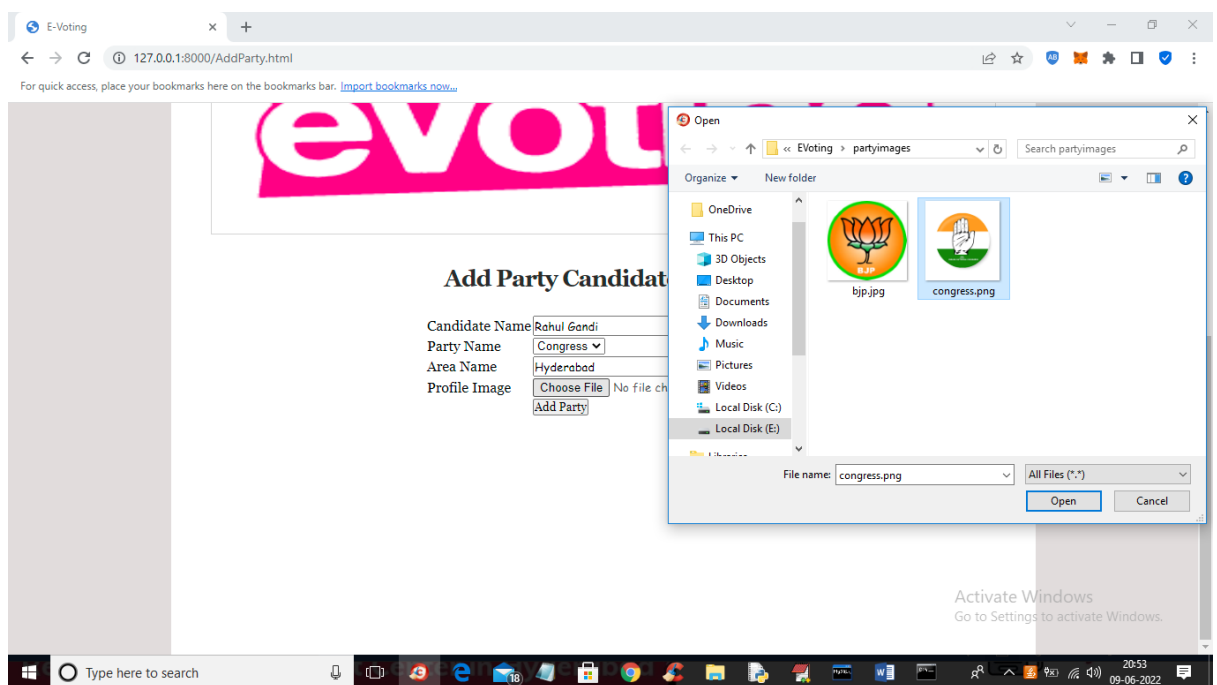


Figure 8.4: Add party candidate page

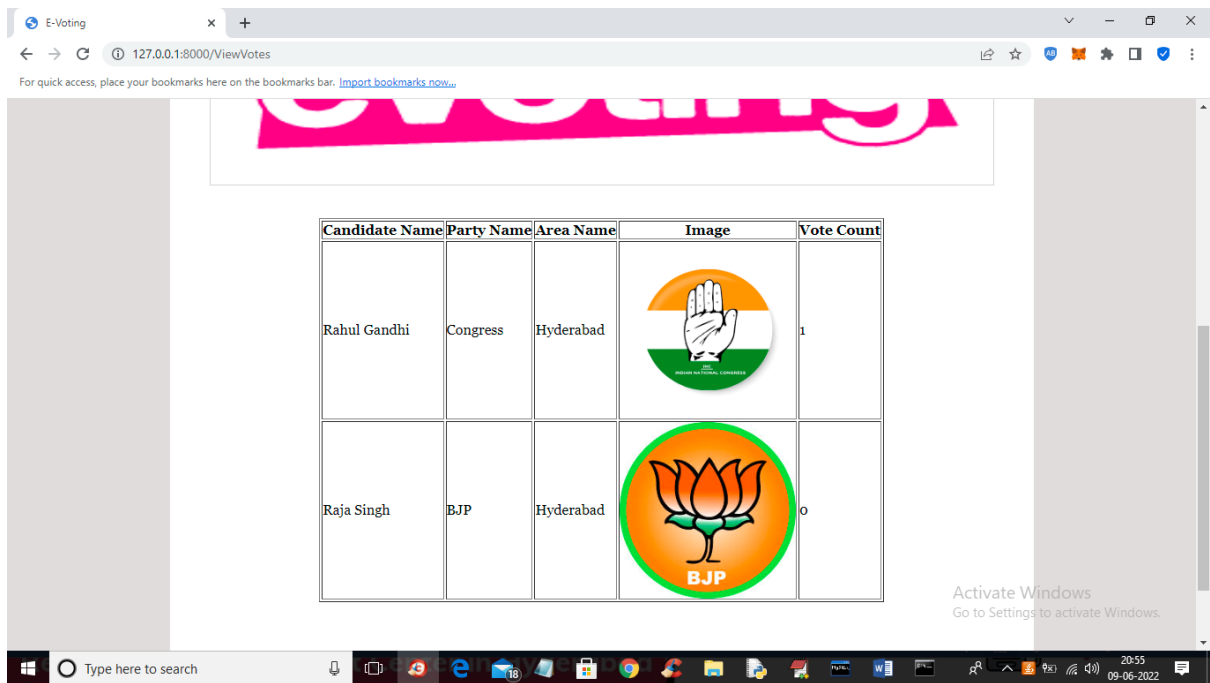


Figure 8.5: View votes page

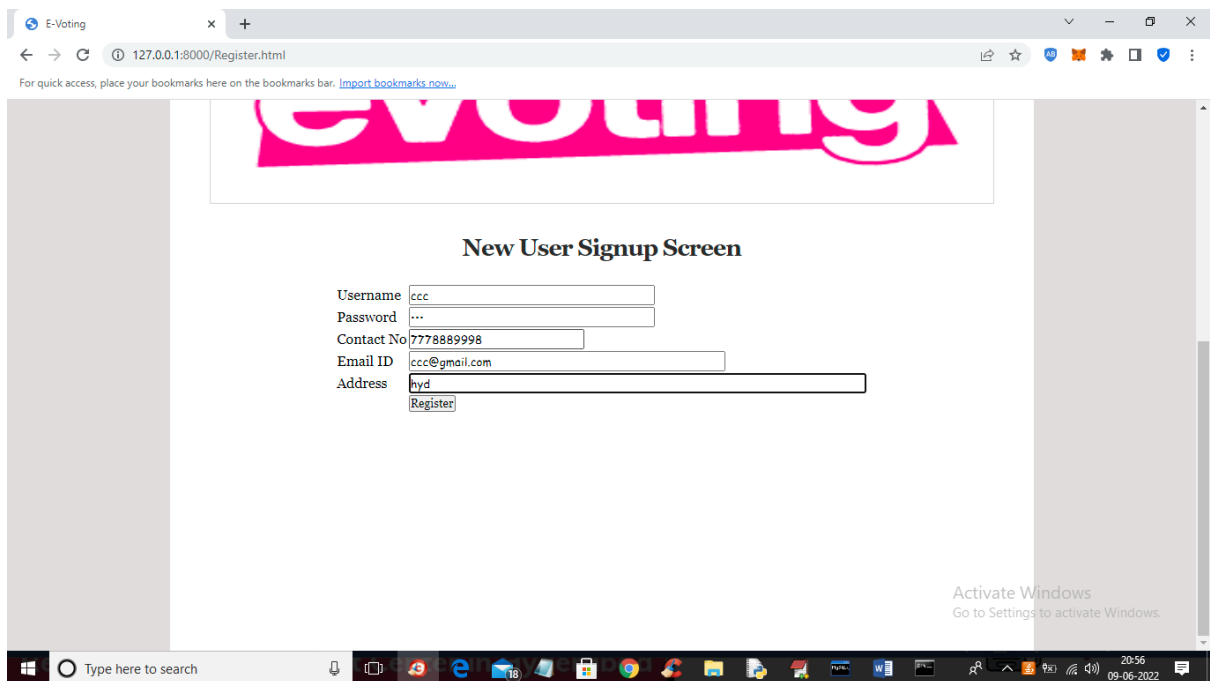


Figure 8.6: Voter registration page

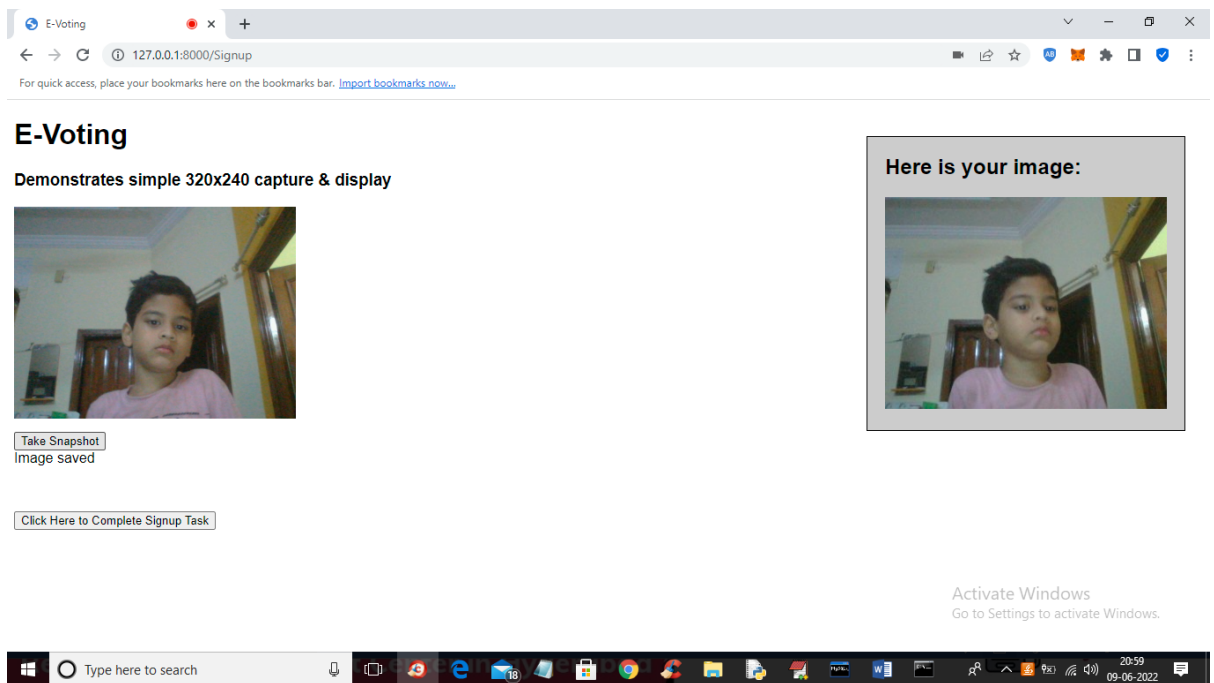


Figure 8.7: User image Capturing

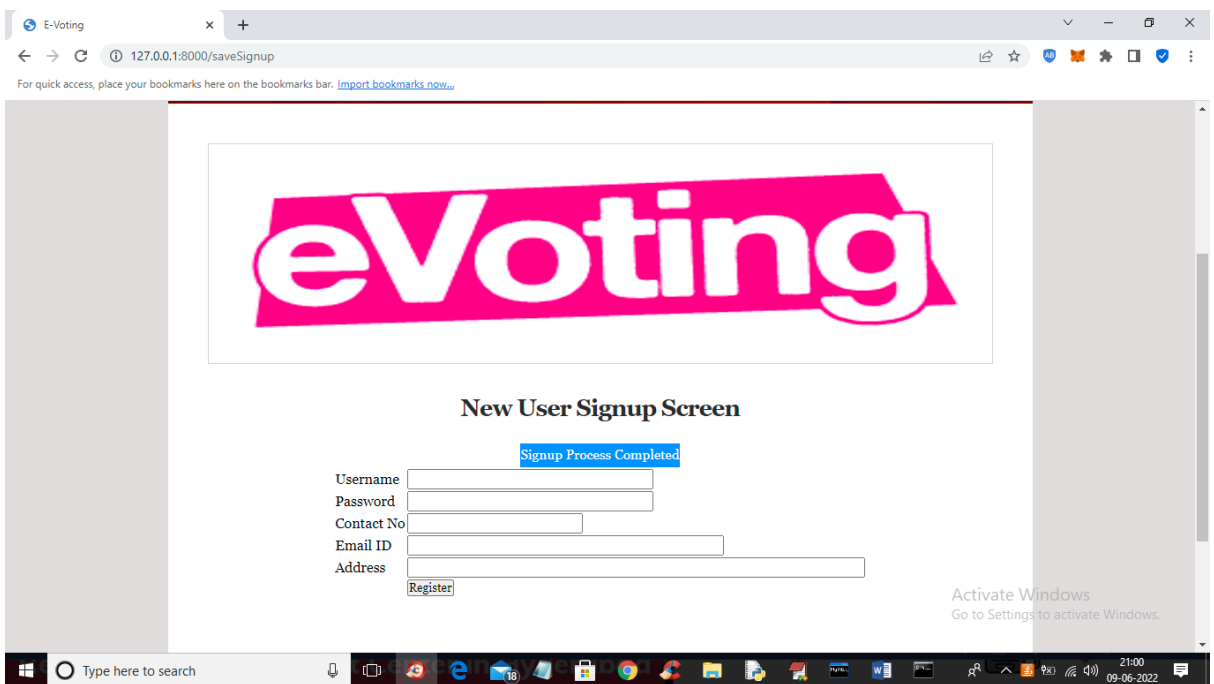


Figure 8.8: Signup process completed

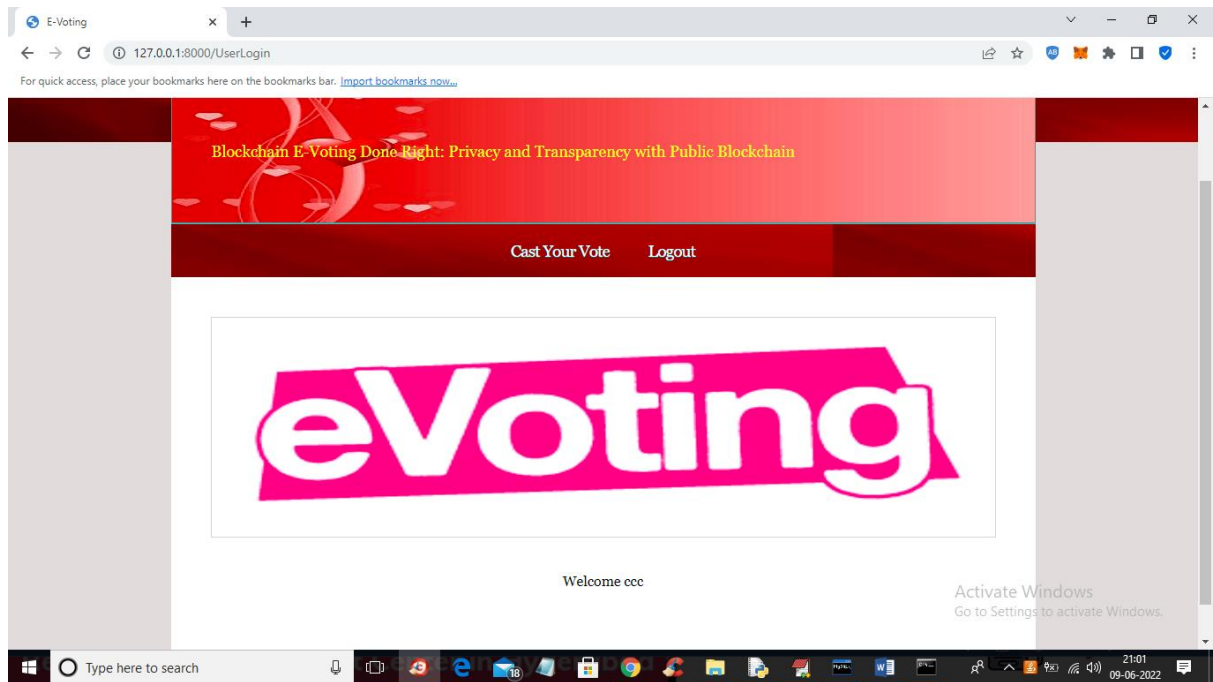


Figure 8.9: Caste your vote page

9. CONCLUSION

The development and implementation of a secure e-voting system using Ethereum blockchain technology and smart contracts represent a significant advancement in ensuring the integrity, transparency, and security of voting processes. Here's a comprehensive conclusion that summarizes the key aspects and benefits of the proposed system: By leveraging Ethereum's blockchain technology, the proposed e-voting system ensures the highest level of security and data integrity. Each vote is recorded as a transaction on the blockchain, providing an immutable and tamper-proof ledger. The use of blockchain technology facilitates complete transparency in the voting process. Since all transactions are publicly recorded on the blockchain, the system allows for real-time verification and auditing of votes. This transparency builds trust among voters and stakeholders, as anyone can independently verify the accuracy and fairness of the voting results. Smart contracts automate the voting process by executing predefined rules and conditions without manual intervention. This automation reduces the likelihood of errors, ensures consistency in vote handling, and accelerates the overall process. Smart contracts handle tasks such as vote counting, result aggregation, and eligibility verification, streamlining the voting process and enhancing efficiency. The e-voting system is designed to respect voter privacy while ensuring the anonymity of individual votes. Using cryptographic techniques, votes are encrypted and anonymized, protecting voter identities from being disclosed. The adoption of a blockchain-based e-voting system increases accessibility for voters, allowing them to cast their votes remotely and securely. This inclusivity is particularly beneficial for individuals who may face challenges accessing traditional voting locations, such as those with disabilities or those residing abroad. Implementing a blockchain-based e-voting system can lead to significant cost savings compared to traditional voting methods. The automation of voting processes, reduction in paper usage, and decreased need for physical infrastructure contribute to lower operational costs. Additionally, the elimination of manual counting and verification processes further reduces expenses. Optimizing blockchain performance and exploring layer-2 solutions can mitigate these concerns. Ensuring that voters are familiar with the e-voting system and its processes is crucial for successful adoption. Providing clear instructions and support can help overcome initial resistance or confusion. The secure e-voting system utilizing Ethereum blockchain technology and smart contracts presents a transformative approach to voting, offering unparalleled security, transparency, and efficiency. By addressing current challenges and incorporating future advancements, this system has the potential to revolutionize the way elections are conducted, ensuring that democratic processes are conducted with the utmost integrity and trustworthiness.

10. FUTURE ENHANCEMENTS

As technology and the field of digital voting continue to evolve, there are several areas where the secure e-voting system using Ethereum blockchain technology and smart contracts can be enhanced. These enhancements aim to improve system performance, scalability, user experience, and compliance with emerging standards. Here are some key future enhancements to consider: Integrate biometric authentication methods (e.g., fingerprint or facial recognition) to ensure that only eligible voters can access the system, adding an additional layer of security. Explore decentralized identity solutions that leverage blockchain technology to verify and manage voter identities in a secure and privacy-preserving manner. Implement layer-2 scaling solutions such as Optimistic Rollups or zk-Rollups to enhance transaction throughput and reduce latency, enabling the system to handle large-scale elections efficiently. Investigate Ethereum sharding techniques to partition the blockchain into smaller, manageable pieces, improving overall network performance and scalability. Continuously refine the user interface (UI) and user experience (UX) to make the voting process more intuitive and accessible, ensuring that voters of all technical levels can navigate the system with ease. Offer multilingual support to accommodate diverse populations and ensure that users can interact with the system in their preferred language. Integrate advanced cryptographic techniques such as Zero-Knowledge Proofs (ZKPs) to enhance voter privacy while maintaining the integrity of the voting process. Investigate compatibility with other blockchain networks to enable cross-chain voting systems and broaden the scope of blockchain-based elections. Ensure that the system complies with evolving legal and regulatory frameworks for electronic voting, engaging with regulatory bodies to incorporate their guidelines and requirements. Follow industry best practices and security standards for blockchain and smart contracts, such as those outlined by organizations like the ISO (International Organization for Standardization). Implement more sophisticated audit trails and reporting features to facilitate comprehensive auditing and verification of the voting process, ensuring transparency and accountability. Integrate support for assistive technologies to accommodate users with disabilities, ensuring that the system is inclusive and accessible to all voters. Future enhancements to the secure e-voting system using Ethereum blockchain technology and smart contracts focus on advancing the system's scalability, privacy, user experience, and regulatory compliance. By implementing these enhancements, the system can continue to evolve and address the needs of modern democratic processes, ensuring a secure, transparent, and accessible voting experience for all users.

11. REFERENCES

- [1] Hajian Berenjestanaki, M.; Barzegar, H.R.; El Ioini, N.; Pahl, C. Blockchain-Based E-Voting Systems: A Technology Review. *Electronics* 2024, 13, 17.
- [2] Said El Kafhali, "Blockchain-Based Electronic Voting System: Significance and Requirements", *Mathematical Problems in Engineering*, vol. 2024, Article ID 5591147, 17 pages, 2024.
- [3] S. N, G. S, S. E and V. K, "E-Voting Using Blockchain," 2023 2nd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA), Coimbatore, India, 2023, pp. 1-4, doi: 10.1109/ICAECA56562.2023.10199732
- [4] Y. A. F. Ali, O. T. M. Ahmed, M. A. M. Diab, M. A. E. Sayed, M. K. Abd Elaziz and B. W. Aboshosha, "Blockchain-Based Online E-voting System," 2023 International Conference on Smart Computing and Application (ICSCA), Hail, Saudi Arabia, 2023, pp. 1-8 doi:10.1109/ICSCA57840.2023.10087767.
- [5] S. Drakshayani, U. Vijayalakshmi, S. R. Sri, A. Srivani and A. Vyshnavi, "Online Voting System Using Blockchain," 2022 International Conference on Electronics and Renewable Systems (ICEARS), Tuticorin, India, 2022, pp. 886-891 doi:10.1109/ICEARS53579.2022.9752044
- [6] M. J. H. Faruk et al., "Development of Blockchain-based e-Voting System: Requirements, Design and Security Perspective," 2022 IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom), Wuhan, China, 2022, pp. 959-967, doi:10.1109/TrustCom56396.2022.00132.
- [7] Hossain Faruk, M.J., Alam, F., Islam, M. et al. Transforming online voting: a novel system utilizing blockchain and biometric verification for enhanced security, privacy, and transparency. *Cluster Comput* (2024).
- [8] A. -M. Stanciu, H. Ciocârlie and C. -P. Julean, "Electronic Voting System Based on the Blockchain Technology," 2023 International Conference on Electrical, Computer and Energy Technologies (ICECET), Cape Town, South Africa, 2023, pp. 1-6. doi:10.1109/ICECET58911.2023.10389415
- [9] S. Al-Maaitah, M. Qatawneh and A. Quzmar, "E-Voting System Based on Blockchain Technology: A Survey," 2021 International Conference on Information Technology (ICIT), Amman, Jordan, 2021, pp. 200-205, doi: 10.1109/ICIT52682.2021.9491734.
- [10] Benny, Albin, Blockchain-based E-voting System (July 11, 2020).
- [11] Liu, Y.; Wang, Q. An E-voting Protocol Based on Blockchain. *IACR Cryptol. Eprint Arch.* 2017, 2017, 1043.

- [12] Shahzad, B.; Crowcroft, J. Trustworthy Electronic Voting Using Adjusted Blockchain Technology. *IEEE Access* 2019, 7, 24477–24488.
- [13] Racsco, P. Blockchain and Democracy. *Soc. Econ.* 2019, 41, 353–369.
- [14] Yaga, D.; Mell, P.; Roby, N.; Scarfone, K. Blockchain technology overview. *arXiv* 2019, arXiv:1906.11078.
- [15] Cullen, R.; Houghton, C. Democracy online: An assessment of New Zealand government web sites. *Gov. Inf. Q.* 2000, 17, 243–267.
- [16] Schinckus, C. The good, the bad and the ugly: An overview of the sustainability of blockchain technology. *Energy Res. Soc. Sci.* 2020, 69, 101614.
- [17] Gao, S.; Zheng, D.; Guo, R.; Jing, C.; Hu, C. An Anti-Quantum E-Voting Protocol in Blockchain with Audit Function. *IEEE Access* 2019, 7, 115304–115316.
- [18] Kim, T.; Ochoa, J.; Faika, T.; Mantooth, A.; Di, J.; Li, Q.; Lee, Y. An overview of cyber-physical security of battery management systems and adoption of blockchain technology. *IEEE J. Emerg. Sel. Top. Power Electron.* 2020.
- [19] Hang, L.; Kim, D.-H. Design and implementation of an integrated iot blockchain platform for sensing data integrity. *Sensors* 2019, 19, 2228. [Green Version]
- [20] Chang, V.; Baudier, P.; Zhang, H.; Xu, Q.; Zhang, J.; Arami, M. How Blockchain can impact financial services—The overview, challenges and recommendations from expert interviewees. *Technol. Forecast. Soc. Chang.* 2020, 158, 120166.

