SecureVote

SUBMITTED IN PARTIAL FULFILLMENT FOR THE REQUIREMENT OF THE AWARD OF DEGREE OF

BACHELOR OF TECHNOLOGY

IN

COMPUTER SCIENCE

Submitted by

RHYTHM GARG (2100290120141)

AASHISH GUPTA (2100290120001)

Supervised by:

MRS. ARTI SHARMA

ASSISTANT PROFESSOR

Session 2024-25

DEPARTMENT OF COMPUTER SCIENCE

KIET GROUP OF INSTITUTIONS, GHAZIABAD

(Affiliated to Dr. A. P. J. Abdul Kalam Technical University, Lucknow, U.P., India)

May 2025

DECLARATION

We hereby declare that this submission is our own work and that, to the best of our knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or other institute of higher learning, except where due acknowledgment has been made in the text.

Date:-

Signature

Name:- Rhythm Garg

Roll No.:- 2100290120141

Signature

Name:- Aashish Gupta

Roll No.:- 2100290120001

CERTIFICATE

This is to certify that Project Report entitled “SecureVote” which is submitted by Rhythm Garg and Aashish Gupta in partial fulfillment of the requirement for the award of degree B. Tech. in Department of Computer Science of Dr. A.P.J. Abdul Kalam Technical University, Lucknow is a record of the candidates own work carried out by them under my supervision. The matter embodied in this report is original and has not been submitted for the award of any other degree.

.

Date: **Supervisor**

Mrs. Arti Sharma

(Assistant Professor)

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of the B. Tech Project undertaken during B. Tech. Final Year. We owe special debt of gratitude to Professor Arti Sharma, Department of Computer Science, KIET, Ghaziabad, for her constant support and guidance throughout the course of our work. Her sincerity, thoroughness and perseverance have been a constant source of inspiration for us. It is only her cognizant efforts that our endeavors have seen light of the day.

We also take the opportunity to acknowledge the contribution of Dr. Ajay Kumar Shrivastava, Head of the Department of Computer Science, KIET, Ghaziabad, for his full support and assistance during the development of the project. We also do not like to miss the opportunity to acknowledge the contribution of all the faculty members of the department for their kind assistance and cooperation during the development of our project.

Last but not the least, we acknowledge our friends for their contribution in the completion of the project.

Date:

Signature

Name:- Rhythm Garg

Roll No.:- 2100290120141

Signature

Name:- Aashish Gupta

Roll No.:- 2100290120001

ABSTRACT

Electronic voting (e-voting) has clear advantages over traditional paper-based systems, such as improved efficiency and reduced errors. However, gaining widespread public trust in e-voting systems remains a significant challenge, particularly in addressing their potential vulnerabilities. Blockchain technology has emerged as a promising solution to enhance the security and reliability of e-voting systems. This project explores how blockchain's transparency and cryptographic capabilities can create a secure, efficient, and user-friendly e-voting infrastructure. Using the Multichain platform, it aims to design and implement an e-voting mechanism that meets critical requirements like legality, accuracy, and security. Despite these advancements, digital voting still faces obstacles that may limit its acceptance. Blockchain’s decentralized nature and end-to-end verification provide a strong foundation for tackling these issues, offering a pathway to more trustworthy and robust electronic voting systems.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ABBREVIATIONS

API: Application Programming Interface.

AUT: Application Under Test.

ETH: Ethereum cryptocurrency.

PoW: Proof of Work (Ethereum's consensus mechanism).

Rinkeby: Ethereum's test network.

SDG MAPPING WITH JUSTIFICATION

SDG 8: Decent Work and Economic Growth

Fair and transparent elections are key to honest leadership and good governance. SecureVote helps make elections more trustworthy, which in turn builds a stable environment for businesses, investments, and jobs. When people have faith in their government, it sets the stage for smarter economic decisions and long-term growth.

SDG 9: Industry, Innovation and Infrastructure

By using blockchain and smart contracts, SecureVote pushes forward innovation in how critical systems like voting are built. It's not just about tech for tech's sake, it's about creating smart, secure digital infrastructure that could be used in many other sectors too. That kind of innovation helps drive sustainable progress in the long run.

SDG 16: Peace, Justice and Strong Institutions

SecureVote is designed to boost trust in democratic systems by making elections tamper-proof, transparent, and verifiable. This helps create more peaceful and fair societies, where people feel confident that their voices are heard. It also supports stronger institutions by reducing the risks of manipulation or fraud in elections.

CHAPTER 1
INTRODUCTION

1.1 Introduction

In today's digital world, eventually getting our voting systems up to speed is just a given – it's got to happen. We've used the old-school pen and paper thing forever, and yeah, people trust it because it's familiar. But let's be real, trying to run huge elections with millions of people using just paper? It's getting pretty strained. Think about it – paper ballots can get messed up by simple human mistakes, or worse, maybe even tampered with. Counting them takes ages, like, seriously long delays waiting for results, and the whole process costs a ton of money to manage. Naturally, people everywhere are starting to look around and ask, "Isn't there a better way?" We need something digital for voting that's not just faster, but actually secure, can handle massive numbers of voters, and is totally transparent.

Now, jumping to electronic voting systems (or e-voting) definitely sounds like the answer, right? The promise is huge: super-fast voting, way more efficient than shuffling paper, potentially cheaper in the long run, and a whole lot easier for folks who find it tough to get to a polling place, like if you're voting from abroad or have mobility issues. But – and it's a big "but" – a lot of people, from regular voters to tech whizzes, are super skeptical. And honestly, fair enough. The big worries are things like systems being vulnerable to hackers changing votes, huge data breaches that could mess everything up, or just not really being able to see how the votes are counted. At the end of the day, the major sticking point for adopting e-voting is trust. Voters absolutely have to believe that their vote is totally private and that it's counted correctly, without anyone messing with it.

This is where blockchain technology steps in – you know, the stuff that first popped up as the backbone for cryptocurrencies like Bitcoin? It turns out its core design is actually pretty perfect for solving those trust issues in e-voting. Blockchain is basically a shared, digital ledger or

record book that isn't stored in just one place (that's the decentralized bit). Once something is recorded on it, it's incredibly difficult – practically impossible – to change or delete it without everyone seeing (that's the immutable part). And everything is protected using some seriously strong cryptography, making it super secure. See why that fits elections? When you need absolute trust, total transparency (in how the system works, not who voted for whom!), and the ability to prove everything later (auditability), blockchain checks a lot of boxes. It seems perfectly built for something as critical as making sure every vote is recorded correctly and securely, while keeping the voter's choice private.

So, rolling all that together, our project, which we're calling "SecureVote," is putting forward an e-voting system specifically built on blockchain. The whole goal here is to seriously amp up how transparent, secure, and generally solid democratic elections are. We're building it using a platform called Multichain, and we're really leaning on the cryptographic tricks blockchain provides to create a voting setup that you can verify, that's tamper-proof, and that's actually easy for people to use. We're putting a lot of focus on making sure it follows all the necessary laws, is totally auditable after the fact, and is super resistant to any kind of sneaky or fraudulent behaviour. And, crucially, we're designing it to keep voter privacy locked down tight, using tech like cryptographic hashing and even biometric checks to confirm identity without linking your name directly to your ballot on the public record.

1.2 Project Category:

At its core, the SecureVote system fits squarely into the realm of building new systems and applications using what we call 'emerging technologies.' It's not just a single piece of software; rather, it's a project that brings together expertise from several different, but related, technical fields.

Specifically, the system draws significantly upon:

1) Blockchain Technology: We're really leveraging the fundamental principles here – the idea that the record is decentralized (not controlled by one single entity), secured through complex math (cryptography), and essentially unchangeable once information is added (immutability). This combination is key to making sure that votes are stored securely and that the entire record of the election can be verified publicly without fear of tampering.

2) Cybersecurity and Cryptography: This is absolutely vital for protecting the voters themselves. We use techniques from these fields to help ensure voter privacy is maintained, prevent anyone from messing with the system or the ballots, and guarantee that the data—the votes themselves—remain accurate and complete.

3) Web Application Development: While the underlying technology is complex, the voter experience needs to be simple. This involves building a user-friendly web interface that makes it easy and accessible for voters to cast their ballot, and also provides administrators with the tools they need to manage the election smoothly.

4) Biometric Authentication: To address concerns about voter identity and preventing duplicate voting, the system incorporates methods for biometric verification. This adds an extra layer of security by helping to confirm that the person casting a vote is indeed the registered voter they claim to be, strengthening the 'one person, one vote' principle in a digital context.

This project started with significant research, exploring the theoretical possibilities and challenges of applying these advanced technologies to elections. However, it didn't stop there; we've actually taken that theoretical work and translated it into a functional prototype. So, you could say it's a blend – combining deep academic exploration with the practical work of building

a real-world system. By applying these innovative solutions, which are rooted in some of the newest computing ideas, the project directly tackles the critical challenge of electoral security. Therefore, it's most accurately described as 'Research-Oriented System Development,' specifically within the important domains of E-Governance (using technology to improve government processes) and Digital Security (protecting crucial digital infrastructure).

1.3 Objectives:

What we're trying to do here is build a new system for voting online, something that's really secure, open, and doesn't mess things up. The main tech behind it is blockchain – you've probably heard of it. We're using what blockchain can do to try and fix a whole load of problems you get with the old paper ballot ways, and even some of the digital voting systems out there already.

To make this all happen, we've got a few main things we're shooting for:

1) Making Sure Voters are Legit: First thing's first, we have to make sure everyone who's casting a vote is actually, like, allowed to. So we're figuring out a really solid way to check people's IDs online. This might mean using official ID documents, maybe even some biometeric stuff like fingerprints down the line. The big idea is just to confirm everyone's who they say they are, so no one sneaks in who shoudnt be voting.
2) Keeping Your Vote Secret: This is a big one. Your vote is your business, period. We're using some clever crypto tricks, like hashing, to make sure that even though the system can count all the votes accurately and everyone can check the final tally, nobody can ever link your actual name to how you voted. It stays totally private.
3) Stopping Cheats and Double Votes: We're also dead set on making sure every eligible voter gets one vote, and only one. The system's being built with stuff in it to stop any funny business, like people trying to vote more then once. We might use biometrics here too to spot any duplicates, and the blockchain itself is pretty good at showing if someone's trying to pull a fast one with dodgy votes.
4) Letting You See it's All Fair: People need to trust this system, right? So, we're making it so you can actually check for yourself that the whole election process is on the level. Voters, and the people running the election, can sort of trace the journey of votes – from when they're cast to when they're counted – all without ever seeing who voted for who, of course. This should help everyone feel a lot more confident that the results are legit and accurate.

And speaking of blockchain, one of its best tricks is that it's un-changeable, or immutable. That's huge for us. All the vote records get locked onto the blockchain, making a kind of permanent,

tamper-proof diary. Once a vote is in and recorded, nobody – no matter who they are – can go back and change it, delete it, or slip in fake ones. It's like, super secure in that way.

Beyond just letting you check the votes, we really want the whole thing to be open and trustworthy. We're aiming for a system where people who need to can look at how parts of the election are running, maybe even in real time. We reckon if people can see more of what's going on, they'll have more faith in the final outcome.

We also want to make voting easier and more efficient, especially if you can't get to a polling station. So, a big part of the project is building a website for voting that's actually easy to use. This means you could vote securely from your own computer or phone, which is great if there's some kind of emergency, if you live miles away from where you're supposed to vote, or if you're out in a rural spot. Its good for people who are far away.

And of course, none of this matters if it costs an arm and a leg. For this system to really take off, it needs to be something places can actually afford. So, we're also looking hard at how to roll this out in ways that are scalable but don't break the bank for different government offices or even other kinds of organizations.

So yeah, by trying to achieve all these bits and pieces, SecureVote is really hoping to make a proper difference. We want to help make elections more fair, more secure, and more trustworthy for everyone, all by using some of this cool new technology.

.

1.4 Structure of Report:

This report is organized into eight comprehensive chapters, each focusing on a key aspect of the SecureVote project—from conceptualization to implementation and final evaluation. Below is an overview of what each chapter contains:

1) Chapter 1: Introduction

- Provides a background on the need for secure and transparent electronic voting systems, followed by an introduction to the SecureVote project. It also discusses the project category, its objectives, and an outline of the entire report.

2) Chapter 2: Literature Review

- Reviews existing research and systems related to e-voting and blockchain technologies. It identifies the gaps in current solutions and formulates the core problem the project aims to address.

3) Chapter 3: Proposed System

- Describes the architecture, design, and unique features of SecureVote. It explains how the system is built, what technologies are used, and how the proposed design overcomes known limitations in current e-voting methods.

4) Chapter 4: Requirement Analysis and System Specification

- Covers the feasibility of the project from technical, economic, and operational perspectives. It also includes the detailed Software Requirement Specification (SRS), system design, data flow diagrams, and database design.

5) Chapter 5: Implementation

- Introduces the tools and technologies used to build SecureVote, such as Solidity, Hardhat, Ethers.js, and React.js. This chapter outlines the development environment and explains the system's implementation process.

6) Chapter 6: Testing and Maintenance

- Details the testing strategy and test cases used to ensure that the system functions securely and correctly. It includes quality objectives, test methodologies, and tools used for functional, integration, and security testing.

7) Chapter 7: Results and Discussions

- Summarizes the key findings and insights gained during the development and evaluation phases. It reflects on the effectiveness of the blockchain-based design and highlights any challenges faced.

8) Chapter 8: Conclusion and Future Scope

- Concludes the report by summarizing achievements and suggesting future enhancements. It discusses how SecureVote can be scaled and adapted for real-world deployment.

Additional sections include the list of references in IEEE format, proof of research paper acceptance, and other supporting documents.

CHAPTER 2
LITERATURE REVIEW

2.1 Literature Review:

Electronic voting, often called e-voting, has drawn a lot of attention as a way to make our democratic processes more efficient and easier for everyone to access. While the idea of smoother voting is attractive, there are still big challenges in making sure these systems are highly secure, completely transparent, and can earn the widespread trust of voters. However, new developments in technology, especially blockchain—with its key features of being decentralized, unchangeable, and secured by cryptography—show good promise for tackling these older issues. Plenty of research and a number of real-world trials have looked at how e-voting and blockchain can connect, and this work forms important background for the SecureVote project.

Traditional e-voting systems, while often faster and more convenient than paper ballots, often aren't very transparent. They can also be open to problems like tampering, unauthorized access, or issues if a central part of the system fails, which can throw the whole vote into question. For example, Javaid (2014) pointed out some major security weaknesses in early e-voting designs. These included poor ways to check voter ID and vote storage systems that you couldn't independently verify. Naturally, these problems made them less reliable and not ideal for important elections where public trust is really crucial.

To deal with these downsides of older e-voting methods, several groups have suggested blockchain-based approaches. The goal is to use blockchain's special features to make voting safer and more open:

1) Follow My Vote: This group created a blockchain platform where voters can actually check in real-time that their vote was recorded correctly. This is key for letting voters see their vote

counted. Their system is built for voting from anywhere and uses math to prove the results are accurate, which helps make the whole process more transparent.

2) Agora: Agora took a significant step by using a custom blockchain voting system in parts of Sierra Leone's 2018 presidential election. Their setup combined ways to agree on vote counts (consensus mechanisms), token-based rewards, and public checks, all designed to handle many voters and offer some transparency that could be audited.

3) Polys: Backed by Kaspersky Lab, Polys has focused on helping communities and organizations run secure, decentralized polls. Their work showed how blockchain could cut costs and make administration smoother, while keeping things very transparent and encouraging people to take part.

4) Voatz: Voatz developed a mobile voting app that uses blockchain and biometric checks (like fingerprints) for security. It's set up to allow remote and anonymous voting and includes ways for voters to make sure their ballots were received and counted right. This aims to make things easier and boost voter trust.

Even with cool ideas like blockchain, current online voting systems still hit some tricky snags we need to fix before lots of people can use them confidently:

1) Privacy vs. Checking Votes: It's hard to keep your vote totally secret and let everyone verify the whole election happened correctly. Getting these two big things right is tough, both tech-wise and ethically.
2) Secure Digital IDs: We need strong, secure ways for voters to prove who they are online. It has to be easy for real voters but really block anyone trying to fake it or vote when they shouldn't.
3) Setting Up & Running Costs: Blockchain systems can be pricey to build and maintain. We need to find ways to keep costs down so they're an option for all kinds of elections, even smaller ones.
4) Cyber Attack Worries: Blockchain isn't attack-proof. Things like trying to crash the system (DDoS) or finding flaws in the code that runs the votes (smart contracts) are real risks hackers might try.

5) Getting People to Trust It: A big step is getting the public comfortable with digital voting. We need simple ways for people to audit (check) the system themselves to really build trust that it's safe and fair.

   Luckily, researchers have been digging into these problems. Folks like Benny (in 2020) and Taherdoost (more recently in 2023) have highlighted how crucial things like smart contracts and smart crypto tools are for making the voting and counting process automatic and secure. They've looked at how tech like SHA-256 (for data fingerprints), Proof of Stake (a blockchain method), and Byzantine Fault Tolerance (systems staying reliable even if parts fail) help ensure votes are correct and agreed upon without one central boss.

   Plus, newer studies are checking out using biometrics (like fingerprints) and having multiple checks for voters. These look like good ways to stop specific issues like someone voting as another person, voting twice, or getting into the system without permission. We're building these smart security and checking methods into our SecureVote project based on what we've learned from all this research, aiming to make our e-voting plan more reliable, secure, and fair to tackle the real challenges of online democracy.

2.2 Research Gaps

Even though there’s been a lot of progress in blockchain-based voting systems, there’s still a bunch of areas where current solutions fall short or don’t go far enough. While some platforms like Agora or Follow My Vote have done decent work, they mostly either stay theoretical or don't scale well in real-world conditions. This leaves a clear gap between what’s possible and what’s actually being used today. One of the main problems is usability. A lot of existing systems aren’t really user-friendly, especially for people who aren’t tech-savvy. If the interface is confusing or hard to access, it discourages voter participation. Not to mention, some platforms assume people are already familiar with how blockchains or cryptographic hashing work, which really isn’t the case for most voters.

Another issue is voter privacy vs. transparency. It’s hard to find that sweet spot where a vote is verifiable but still anonymous. Many systems use hashes or tokens, but if these aren’t handled carefully, there's always the risk of linking votes back to users — either accidentally or maliciously.

Also, there's not a lot of open-source, customizable frameworks available. Most existing systems are closed off or too generalized, so organizations can’t tweak things to fit their specific needs or legal standards. This limits their practical use. Security-wise, sure, blockchain is pretty secure, but not bulletproof. Threats like DDoS attacks, phishing during voter registration, or flaws in biometric modules are often overlooked. Plus, many of these systems don’t have strong enough methods for real-time verification or fraud detection during voting.

Finally, the cost and infrastructure barrier is still pretty high. Even though blockchain promises cost reduction in the long run, the initial deployment—especially if it includes biometric gear, secure hosting, and training—can be expensive and technically overwhelming for small institutions or local governments.

So yeah, there's a lot of room for improvement. SecureVote aims to fill these gaps by not just using blockchain as a buzzword, but really thinking through how to make voting fair, simple, secure, and actually usable.

2.3 Problem Formulations

Despite advances in digital technology, modern-day voting systems—both electronic and traditional—still face major challenges that threaten the integrity of democratic processes. Many existing electronic voting (e-voting) platforms fail to meet core requirements like voter privacy, vote integrity, tamper resistance, and public verifiability, all in one single solution.

Right now, there's no widely adopted system that offers a fully secure, transparent, and verifiable voting process that is also easy to use and scalable to large populations. Centralized systems are prone to hacking, manipulation, or internal tampering. On the other hand, some blockchain-based voting systems exist, but they either stay too theoretical, lack proper identity verification, or don't balance transparency with anonymity well enough.

Another major concern is the lack of trust. If voters can't verify that their vote was counted correctly—or worse, if they think it could be traced back to them—they're less likely to participate. And let's not forget how important authentication is. Without a solid way to make sure only real, eligible voters are casting ballots (and only once), the whole process can fall apart.

Given these issues, the specific problem this project aims to solve is:
How can we design and implement a blockchain-based electronic voting system that ensures voter authentication, vote anonymity, and tamper-proof recording, while also making the entire process transparent, auditable, and accessible for general users?

The system needs to satisfy the following conditions:

1) Only eligible voters should be able to vote, and only once.
2) The vote should be recorded immutably on the blockchain and remain untampered.
3) Voters must remain anonymous, even to the system administrators.
4) The system must allow public verification of results without revealing voter identities.
5) It should be easy to use, even for non-technical users, and scalable for large-scale elections.
6) The infrastructure should be secure, resisting common cyber threats and privacy violations.

CHAPTER 3
PROPOSED SYSTEM

3.1 Proposed System:

SecureVote is built using a platform called Multichain. This basically lets us create our own private, permissioned blockchain network. Why private? Because we only want the authorized folks – like the official election administrators and the verified voters – to be able to participate and interact with the system. But even though it's private, you still get all the fantastic benefits of blockchain, like its incredible security and the ability to audit everything because the record is permanent.

Voters interact with SecureVote through a simple website. The front end is designed to be straightforward and easy to navigate, so you don't need to be a tech expert to cast your ballot. The voting process is broken down into a few simple steps: you'll register, prove you are who you say you are, cast your vote digitally, and if you're curious, you can even check later to make sure your vote was recorded – all without giving up your anonymity.

1) Signing Up and Proving It's Really You: First things first, we need to verify that you're an eligible voter. You'll register using your official government ID, and there's even a biometric check involved, like a fingerprint scan, to make sure the person registering is actually you. Once you're verified, the system creates a unique, anonymous digital ID for you using some clever cryptographic magic. This is how you're identified on the blockchain – not by your name, but by this secure, one-way digital code.
2) Casting Your Vote, Totally Secretly: When it's time to vote, you get a digital ballot that's encrypted. Your actual vote choice is then converted into another kind of digital fingerprint (a hash) and recorded on the blockchain. The crucial part here is that your real identity is linked directly to this vote record in a way that could trace it back to you. It also supports something

called "receipt-freeness," which means you can't easily prove to someone else who you voted for, helping prevent things like vote buying or coercion. Your vote is truly your own business.

3) The Unbreakable Record Keeper: Every single vote cast is recorded as a transaction on the blockchain ledger. Once that vote is added, it's permanent. You absolutely cannot change it, tamper with it, or delete it. This immutability is one of blockchain's biggest strengths and eliminates the risk of anyone messing with the vote count after ballots are cast.
4) Tools for the Election Crew: Election officials get a dashboard with tools to manage the process, like setting up different voting areas or seeing the overall number of people who have voted. They can verify the final results. However, and this is super important for trust, they cannot see how individual people voted, nor can they alter any of the data once it's on the blockchain. Their role is to manage the election, not see your private choice or change anything.
5) You Can Check Your Vote: After you cast your ballot, you'll get a specific transaction ID via an NFT, like a tracking number for your vote. You can use this ID to look up your vote's record on the blockchain ledger. This lets you personally verify that your vote was indeed recorded correctly and exists in the system – without revealing what your actual vote was. It's a great way to build confidence in the process.
6) Keeping the Bad Guys Out: Security is layered throughout SecureVote. We use strong stuff like SHA-256 for hashing, SSL encryption to keep data safe while it travels over the internet, and digital signatures to make sure votes are authentic and haven't been messed with. Plus, there are standard defenses against automated attacks and fraud, like limiting failed login attempts, using CAPTCHA tests to block bots, and those initial biometric locks to ensure legitimate access.

SecureVote has a few more neat features:

1) Easy to Audit: Since the blockchain ledger is transparent (while keeping votes anonymous), independent auditors can easily review the entire record of the election. It makes the whole process much more accountable without exposing private voter data.
2) Vote From Anywhere: As long as you have an internet connection, you can vote! This system supports remote voting from pretty much any device. It's great for making voting more accessible, especially for people who might live far from a polling station or have mobility issues.

3) Flexible Design: The system is built modularly, which means it's not just for massive national elections. We can adapt and customize it for different needs, whether it's for a city council election, a university's student body president vote, or something else entirely.
4) Ready to Grow: The Multichain platform is designed to handle a lot of activity, so scaling up SecureVote to accommodate huge numbers of voters shouldn't be a problem and shouldn't bog down the system.

Just a quick look at the main technologies making SecureVote run:

1) Blockchain: We're using Multichain with an Ethereum-based blockchain. (picked partly because it's good for setting up those private networks).
2) Frontend: Built with standard web technologies like HTML/CSS and JavaScript.
3) Backend: Could be running on Node.js or Python Flask – depends on the specific setup.
4) Security Magic: Uses SHA-256 and digital signatures for keeping things safe and verified.
5) Other Data: For information that doesn't need to live directly on the blockchain, we'd use a database like PostgreSQL.
6) Identity Checks: Integrating biometric systems and government ID verification.

3.2 Unique Features of The System

While there are definitely other e-voting systems out there, SecureVote brings together a number of features that really make it stand apart, not just from the older digital voting platforms but even from many of the newer blockchain ones. These aren't just bells and whistles; they're features designed to get to grips with some of the tough, real-world problems we see with trust, security, and just how easy (or not!) elections are to use.

So, one of the big things with SecureVote is how it uses biometric checks – think fingerprint scanning – when people sign up to vote. This adds a serious extra layer of checking who's who, making it a lot tougher for someone to create fake accounts or for bots to try and mess with the system. Using biometrics helps ensure that each real person can only sign up once and can't pretend to be somebody else. It’s a solid step towards making sure voters are genuine.

Now, some blockchain systems out there can accidentally leave little clues about a voter's identity on the actual blockchain. SecureVote handles this differently. When you register, it creates a unique cryptographic hash, basically a special, scrambled code. This hash keeps your identity completely seperate and anonymous, but still lets the system make sure your vote is counted correctly and the overall election is fair. The bottom line? Nobody, not even the system administrators, can connect your specific vote back to you. That means your vote stays truly private.

After you've cast your vote, you'll get a transaction ID sent to your email and also an NFT will be generated for you, which you can find on platforms like OpenSea. This little ID is your way of checking that your vote definitely got recorded on the blockchain. You can see it's there, but it doesn't show what your actual vote was. This is all about building up confidence in the system and being transparent, while still keeping the actual content of your vote under wraps.

Every single vote is then written to what's called an immutable ledger. It’s a digital record book that can’t be tampered with. This is powered by the Multichain blockchain platform. Once your vote is in that ledger, it can't be changed, deleted, or swapped out by anyone. This makes trying

to fiddle with the results after the election pretty much impossible, which means you can have a lot more trust in the outcome.

We've all seen how some secure systems can be a real pain to use, right? SecureVote really tries to be different here. It offers a clean, user-freindly web app that should work smoothly even if you're not a tech wizard. This is super important for making sure everyone can vote, especially people voting online for the first time or maybe older folks who aren't as comfortable with complicated tech.

When it comes to checking if an election was run properly, old systems often mean you need full admin rights to look at the data. SecureVote's blockchain ledger changes that. It allows for public auditability. This means any autorized observer can verify that the election was run with integrity, and they can do it without needing to get into private voter information or sensitive background data.

Thanks to putting biometric ID checks together with those digital voter hashes we mentioned, the system is also really good at stopping anyone from voting more than once. It'll automatically block attempts, even if someone tries to be sneaky and use different computers or internet connections. This really shores up the integrity of the final count.

SecureVote isn't meant to be a rigid, one-size-fits all kind of deal either. The platform is modular. That means it can be adapted fairly easily for smaller things, like student council elections, or scaled up for bigger stuff like national or state-level votes. The tools for admins and the backend features can also be tweaked to fit what's needed.

And here's another important security point: even if, somehow, someone managed to get unauthorized access to the main hosting server, they still couldn't tamper with the votes. Why? Because all the actual vote data isn't just sitting on that server; it's stored on the blockchain. This makes the vote records independent of whether that central server is secure or not, adding another strong layer of protection agaisnt insider threats or direct attacks on the server.

CHAPTER 4
REQUIREMENT ANALYSIS AND SYSTEM SPECIFICATION

4.1 Feasibility Study (Technical, Economical, Operational):

From a technical point of view, SecureVote is definitely implementable with current tools and technologies. It uses components like:

Multichain blockchain (which is open-source and well-documented),

Biometric authentication (which can be integrated via existing APIs or hardware modules),

Standard web development stacks like HTML, CSS, JavaScript, and Node.js or Python.

The project doesn't rely on futuristic or experimental tech. All the building blocks already exist and can be connected using known methods. Blockchain itself is a mature technology at this point, and Multichain simplifies a lot of the setup and permission handling. The only real challenge is syncing biometric data securely with the blockchain records—but even that can be handled with proper encryption and architecture design.

One concern with blockchain-based solutions is cost. While blockchain promises long-term savings (like reduced fraud, faster counting, less manpower), the initial setup can be a bit heavy. Costs may include:

Setting up secure servers,

Hosting and maintaining the blockchain nodes,

Biometric devices (if used in physical booths),

Development time and skilled personnel.

However, by using open-source tools and cloud infrastructure, we can cut down the budget significantly. Also, for small to medium-scale elections (student bodies, institutions, etc.), the cost is very manageable. Over time, as more organizations adopt the system, the per-unit cost will drop further.

Operationally, SecureVote is designed to be simple enough for real users and admins to manage. With a clean interface, minimal training requirements, and straightforward workflows for both voters and election authorities, it doesn’t require deep technical knowledge to operate.

Voters can log in, verify themselves using biometric ID, cast their vote in just a few clicks, and verify it using a transaction ID. Meanwhile, admins can manage constituencies, view analytics, and publish results securely. Since most of the backend logic (like vote recording and verification) is handled by the blockchain automatically, there’s less chance of human error.

The system is also flexible—it works in both remote and on-site voting scenarios. And since the votes are stored on a decentralized ledger, there's no single point of failure that can compromise the operation.

4.2 Software Requirement Specification

This section outlines the functional and non-functional requirements, as well as the hardware and software dependencies, necessary to build and run the SecureVote e-voting system. The goal is to clearly define what the system must do and the environment it will run in.

These are the features the software must support for it to work as intended.

1) User Registration
- Voters must be able to register using valid ID documents and biometric verification.
- The system should generate a unique cryptographic hash for each verified voter.

2) Voter Authentication
- Voters should log in using a secure authentication process (e.g., password + fingerprint).
- Only authenticated and verified users can access the voting interface.

3) Vote Casting
- The system must allow each authenticated voter to cast a vote for a candidate of their choice.
- Only one vote should be allowed per user.

4) Vote Anonymity
- Votes must be anonymized before being written to the blockchain (e.g., using hashing).
- No personal data should be traceable to the vote.

5) Blockchain Transaction
- Every vote should be recorded as a transaction on the blockchain.
- The system must handle blockchain communication for writing and retrieving vote data.

6) Vote Verification
- After voting, the user should receive a transaction ID to verify their vote was recorded.
- Users must be able to view their transaction on a blockchain explorer.

7) Admin Panel
- Admins can create elections, manage constituencies, and approve/reject voter registrations.

- Admins cannot view individual votes or alter them.

8) Result Tallying
- The system must count votes from the blockchain and display real-time or final results.

- Results must be publicly viewable without compromising vote privacy.

These define the quality and constraints of the system beyond basic functionality.

1) Security:

- All data in transit must be encrypted using SSL.
- Biometric and user data should be securely stored and hashed.
- System should defend against common threats like SQL injection, XSS, and DDoS.

2) Performance:

- The system should handle hundreds (or thousands) of concurrent users.
- Voting transactions should complete in under 2 seconds on average.

3) Usability:

- User interface should be intuitive and require minimal training.
- The platform should support multiple devices (desktop, tablet, mobile).

4) Scalability:

- The system should scale to support elections of different sizes—from small universities to large government polls.

5) Availability:

- The system should maintain 99.9% uptime during active election periods.

6) Maintainability:

- Codebase should be modular and well-documented for future updates or improvements.

Hardware Requirements:

Component	Minimum Specification		
Client Device	Smartphone / Laptop with browser		
Server (Node Host)	4-core CPU	8 GB RAM	100 GB SSD
Biometric Device	Fingerprint or retinal scanner		

Software Requirements:

Software Component	Description
Operating System	Windows/Linux (for server and dev)
Frontend	JavaScript, React
Backend	Node.js, Solidity

Blockchain Platform	Multichain
Database (optional)	PostgreSQL (for voter data)
Version Control	Git

CHAPTER 5
IMPLEMENTATION

5.1 Introduction Tools and Technologies Used:

To build a secure, transparent, and decentralized e-voting platform, our system leverages a modern technology stack composed of both blockchain-specific tools and general-purpose web development frameworks. Each tool was chosen based on its suitability for the requirements of our project, including smart contract development, blockchain interaction, front-end design, and backend logic handling.

Below is a breakdown of the core tools and technologies used:

1. Solidity

Solidity is the primary programming language used to write smart contracts for the Ethereum blockchain. It enables the creation of logic for vote casting, voter registration, and ensuring tamper-proof vote records. Its contract-oriented nature makes it ideal for defining voting rules and storing votes immutably on-chain.

2. React.js

React.js was used to develop the frontend of the web application. Its component-based architecture allowed for building a responsive and interactive UI for voters and administrators. React enabled a smooth user experience, especially during actions like login, vote submission, and transaction tracking.

3. Node.js

Node.js powered the backend server that communicates with the blockchain and handles operations like wallet management, API routing, and verification processes. Its event-driven, non-blocking architecture makes it highly efficient for handling real-time voting interactions and network events.

4. Hardhat

Hardhat is a development environment and testing framework for Ethereum smart contracts. It was used to compile, deploy, and test our Solidity contracts. Hardhat's local blockchain simulation and plugin ecosystem greatly accelerated the development process by allowing us to catch issues before going live.

5. Ethers.js

Ethers.js is a lightweight JavaScript library used to interact with the Ethereum blockchain. It connects our React frontend with the deployed smart contracts and handles wallet functions, transaction signing, and blockchain queries. Its modular design makes it well-suited for building secure DApps.

6. Remix IDE

Remix IDE was used during the early stages of smart contract development for rapid prototyping and debugging. Its browser-based interface allowed us to write, test, and deploy smart contracts quickly without setting up a full development environment initially.

CHAPTER 6

TESTING AND MAINTENANCE

6.1 Testing Techniques and Test Cases Used:

Features Slated for Testing:

The testing process will cover several key aspects of the SecureVote system:

- The functionality of smart contracts written in Solidity.
- Aadhar-based biometric authentication for voter verification.
- The entire workflow: from vote casting and subsequent validation through to the final computation of results.
- System integration, focusing on how different components interact, including communication between blockchain nodes.
- The usability and accessibility of the voting interface for the end-user.

Key Quality Objectives:

- Ensure all smart contracts are functioning correctly and meet specified operational requirements, including aspects like performance and security.
- Thoroughly validate the security measures implemented within the blockchain network.
- Confirm the integrity and authenticity of all voter data throughout the process.

Test Methodology:

For SecureVote, our testing methodology is built on an Agile framework. This means we're committed to iterative and continuous testing throughout the development cycle, rather than only at the end. Each feature or component is tested incrementally as it's developed. This approach allows us to identify and address any potential issues in a timely manner. The overall aim is to ensure that all parts of the system function as intended, both when tested individually and when they are integrated into the complete solution.

1. Unit Testing: Validating the functionality of individual smart contract functions.

2. Integration Testing: Ensuring seamless interaction between the authentication system, blockchain network, and user interface.

3. System Testing: Verifying the entire system, including end-to-end voting, authentication, and result calculation.

4. Security Testing: Assessing the system's resistance to vulnerabilities such as double-voting, unauthorized access, and data breaches.

Test Completeness

Testing will be deemed complete when:

- All smart contract functions achieve 100% code coverage.
- Security vulnerabilities are resolved.
- All critical bugs are fixed, and non-critical bugs are documented for future releases.
- The voting system passes performance benchmarks for up to 1 million users.

Test Deliverables

- Comprehensive test cases for smart contract operations.
- Bug reports and resolution logs.
- Performance metrics for voter throughput and system latency.
- Security audit report for the blockchain and authentication system.

Test Cases for Aadhar Authentication:

Boundary Value Analysis (BVA) is a testing technique used to validate inputs at the edge of acceptable ranges, ensuring the system works correctly for both valid and invalid boundaries.

An Aadhar Card number is a 12-digit numeric identifier issued in India. The valid range of digits is 0000 0000 0001 to 9999 9999 9999.

Test Case ID	Test Input	Expected Result	Reason
TC1	000000000001	Valid	Minimum Valid Value
TC2	999999999999	Valid	Maximum Valid Value

TC3	000000000000	Invalid	Below Lower Boundary
TC4	1000000000000	Invalid	Above upper boundary
TC5	12345678901	Invalid	Less than 12 digits
TC6	1234567890123	Invalid	More than 12 digits
TC7	1234A5678901	Invalid	Alphanumeric Input
TC8	12@#$5678901	Invalid	Special Characters Input
TC9	“”	Invalid	Empty Input
TC10	“ “	Invalid	Input with spaces only

Decision Table to allow/reject voting based on several conditions:

Rule ID	C1: 12 Digits	C2: Numeric Only	C3: Valid Range	C4: Already Voted	Action
R1	Yes	Yes	Yes	No	Allow Voting
R2	No	-	-	-	Reject Due to Invalid Aadhar
R3	Yes	No	-	-	Reject Due to Invalid Aadhar
R4	Yes	Yes	No	-	Reject Due to Invalid Aadhar

R5	Yes	Yes	Yes	Yes	Reject Due to Duplicate Vote
R6	No	No	-	-	Reject Due to Invalid Aadhar

Resource & Environment Needs: Testing Tools

Ganache: For local blockchain simulation.

Truffle: For deploying and testing smart contracts.

Postman: For API testing of authentication endpoints.

Selenium: For automated testing of the voting interface.

OWASP ZAP: For security testing and vulnerability analysis.

CHAPTER 7
RESULTS AND DISCUSSION

7.1 Key Findings of The Project:

Throughout the design, development, and testing of the SecureVote e-voting system, several important insights and outcomes emerged. These key findings highlight both the strengths of using blockchain for electoral processes and the practical considerations that must be taken into account when implementing such systems in real-world scenarios.

1. Blockchain Significantly Improves Vote Integrity

By using a decentralized and immutable ledger, we found that blockchain virtually eliminates the possibility of vote tampering, unauthorized changes, or centralized control. Every vote recorded on-chain is permanent, traceable (without compromising anonymity), and verifiable.

2. Voter Anonymity and Transparency Can Co-Exist

Through the use of cryptographic hashing, wallet abstraction, and smart contract design, the system successfully maintains voter anonymity while still allowing each vote to be transparently verified on the blockchain. This confirms that it is technically possible to build a system that is both private and auditable.

3. Smart Contracts Provide Trustless Automation

The logic of elections—such as vote validation, duplicate prevention, and result computation—was embedded directly into smart contracts using Solidity. This removes reliance on third parties or manual oversight, making the voting process more automated, trustless, and tamper-resistant.

4. User Experience Matters for Adoption

While the backend was secure and technically sound, early feedback showed that voters (especially non-technical ones) require an intuitive and minimal-click interface. Integrating

blockchain features (like MetaMask or wallet confirmations) in a user-friendly way was essential to build trust and encourage usage.

5. Development Tools like Hardhat Streamlined Testing

Using Hardhat during the development cycle helped us test smart contracts in a controlled environment before deploying them to a live network. It also allowed for time-travel debugging, unit testing, and automatic contract deployment, reducing the risk of critical bugs.

6. Real-World Implementation Requires Infrastructure Support

Despite being technically feasible, deploying this system in a real-world government election would require additional layers like KYC integration, regulatory compliance, hardware-level security (e.g., biometric devices), and offline voting support for rural areas.

7. Voter Confidence Increases with Verifiable Proof

Providing each voter with a transaction hash after voting greatly increased trust in the system. Voters appreciated being able to verify their vote on an Ethereum block explorer, proving that transparency directly contributes to confidence in digital voting.

In conclusion, the SecureVote project confirmed that blockchain-based e-voting systems are not only possible but offer a promising path forward for more secure, fair, and transparent elections—provided proper attention is given to usability, scalability, and legal integration.

CHAPTER 8
CONCLUSION AND FUTURE SCOPE

Conclusion:

The SecureVote project aimed to tackle the growing need for secure and transparent digital voting systems by leveraging blockchain technology. Through careful design and implementation, the project successfully built a functional prototype that ensures voter authentication, vote anonymity, tamper-proof recording, and verifiable results. Using tools like Solidity, React.js, Node.js, Hardhat, and Ethers.js, the system was able to offer a seamless experience for both voters and administrators. Key features such as biometric-backed registration, cryptographic vote hashing, and immutable blockchain storage ensured both security and trust in the process.

SecureVote confirms that blockchain-based e-voting is not only technically feasible but also more secure and auditable than traditional systems, making it a strong candidate for future adoption in institutional and even public-sector elections.

Future Scope:

While the current prototype meets many critical requirements, there are several areas where the system can be further improved:

- Mobile App Deployment: Developing a secure mobile version of SecureVote could significantly improve accessibility and ease of use.
- Offline and Hybrid Voting Support: Supporting offline vote collection with later blockchain syncing would increase inclusivity in low-connectivity regions.
- Advanced Privacy Techniques: Incorporating methods like Zero-Knowledge Proofs (ZKPs) could enhance privacy without sacrificing verifiability.
- Legal and Regulatory Alignment: Further research into compliance with electoral laws and data privacy regulations would help prepare the system for real-world deployments.

REFERENCES

[1] Mayuri Gomte, Harshal Chaudhari, Abhishek Tapare, and Dr. Y. M. Patil. (2023, April 28). "IoT-Based E-Voting System to Prevent Fraudulent Voting." International Journal of Advanced Research in Science, Communication and Technology, 289–293. https://doi.org/10.48175/ijarsct-9576

[2] Benny, A. (2020, August 11). "Blockchain-Based E-Voting System." Blockchain Based E-voting System by Albin Benny: SSRN. https://doi.org/10.2139/ssrn.3648870

[3] Taherdoost, H. (2023, February 13). "Smart Contracts in Blockchain Technology: A Critical Review." MDPI. https://doi.org/10.3390/info14020117

[4] Javaid, M. A. (2014). "Electronic Voting System Security." SSRN Electronic Journal. https://doi.org/10.2139/ssrn.2393158

[5] Khan, I., and Shahaab, A. (2021, February 2). "A Peer-To-Peer Publication Model on Blockchain." *Frontiers in Blockchain*, https://doi.org/10.3389/fbloc.2021.615726

[6] "Smart Voting Platform (Secured Digital Voting System) Utilizing Blockchain Technology and Biometric Authentication." (2023, May 17). *International Journal of Science and Engineering Applications*, 105–113. https://doi.org/10.7753/ijsea1205.1029

[7] "Decentralized Online Voting System." (2023, May 31). *International Research Journal of Modernization in Engineering Technology and Science.* https://doi.org/10.56726/irjmets40446

[8] Virani, H., and Kyada, M. (2022, December 9). "A Systematic Literature Review on Smart Contracts Security." *arXiv.org.* https://arxiv.org/abs/2212.05099v1

[9] Khan, S. N., Loukil, F., Ghedira-Guegan, C., Benkhelifa, E., and Bani-Hani, A. (2021, April 18). "Blockchain Smart Contracts: Applications, Challenges, and Future Trends." *PubMed Central (PMC).* https://doi.org/10.1007/s12083-021-01127-0

[10] Yao, J., Wei, L., and Liu, T. (2020, July 6). "Blockchain-Based Voting System." *Computer System Networking and Telecommunications*, 3(1). https://doi.org/10.18063/csnt.v3i1.1146

TURNITIN PLAGIARISM REPORT

PCS25-57-11

ORIGINALITY REPORT

11%	10%	5%	7%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.ijraset.com Internet Source	3%
2	www.coursehero.com Internet Source	2%
3	Submitted to KIET Group of Institutions, Ghaziabad Student Paper	1%
4	Submitted to ABES Engineering College Student Paper	1%
5	it.gndec.ac.in Internet Source	<1%
6	Submitted to University of Westminster Student Paper	<1%
7	www.scirp.org Internet Source	<1%
8	fr.slideshare.net Internet Source	<1%
9	powerknot.com Internet Source	<1%
10	technodocbox.com Internet Source	<1%

*% detected as AI

AI detection includes the possibility of false positives. Although some text in this submission is likely AI generated, scores below the 20% threshold are not surfaced because they have a higher likelihood of false positives.

Caution: Review required.

It is essential to understand the limitations of AI detection before making decisions about a student's work. We encourage you to learn more about Turnitin's AI detection capabilities before using the tool.

Disclaimer
Our AI writing assessment is designed to help educators identify text that might be prepared by a generative AI tool. Our AI writing assessment may not always be accurate (it may misidentify writing that is likely AI generated as AI generated and AI paraphrased or likely AI generated and AI paraphrased writing as only AI generated) so it should not be used as the sole basis for adverse actions against a student. It takes further scrutiny and human judgment in conjunction with an organization's application of its specific academic policies to determine whether any academic misconduct has occurred.

Frequently Asked Questions

How should I interpret Turnitin's AI writing percentage and false positives?
The percentage shown in the AI writing report is the amount of qualifying text within the submission that Turnitin's AI writing detection model determines was either likely AI-generated text from a large-language model or likely AI-generated text that was likely revised using an AI-paraphrase tool or word spinner.

False positives (incorrectly flagging human-written text as AI-generated) are a possibility in AI models.

AI detection scores under 20%, which we do not surface in new reports, have a higher likelihood of false positives. To reduce the likelihood of misinterpretation, no score or highlights are attributed and are indicated with an asterisk in the report (*%).

The AI writing percentage should not be the sole basis to determine whether misconduct has occurred. The reviewer/instructor should use the percentage as a means to start a formative conversation with their student and/or use it to examine the submitted assignment in accordance with their school's policies.

What does 'qualifying text' mean?
Our model only processes qualifying text in the form of long-form writing. Long-form writing means individual sentences contained in paragraphs that make up a longer piece of written work, such as an essay, a dissertation, or an article, etc. Qualifying text that has been determined to be likely AI-generated will be highlighted in cyan in the submission, and likely AI-generated and then likely AI-paraphrased will be highlighted purple.

Non-qualifying text, such as bullet points, annotated bibliographies, etc., will not be processed and can create disparity between the submission highlights and the percentage shown.

RESEARCH PAPER ACCEPTANCE PROOF

RHYTHM GARG <rhythm.2125cs1135@kiet.edu>

IEEE ICCSAI 2025-Technical Presentation Schedule-Day 2_OFFLINE

1 message

Microsoft CMT <email@msr-cmt.org> Wed, Apr 2, 2025 at 1:49 PM
Reply-To: ICCSAI 2025 <iccsai@galgotiasuniversity.edu.in>
To: Rhythm Garg <rhythm.2125cs1135@kiet.edu>
Cc: iccsai@galgotiasuniversity.edu.in

Dear Rhythm Garg,

We hope this message finds you well. We are delighted to inform you that your manuscript has been accepted for oral presentation at the 2025 3rd International Conference on Communication, Security, and Artificial Intelligence, scheduled to be held from April 4-6, 2025, at Galgotias University, Greater Noida, U.P., India in hybrid mode.
Your presentation is scheduled in OFFLINE mode on 5th April 2025

Your insights and expertise will undoubtedly contribute to the success of our event. To facilitate your participation, we kindly request that you download the technical paper presentation schedule for ICCSAI 2025 using the following link:

Download using the link for: https://drive.google.com/file/d/1Joe06bfxde1i7u5dkEAPraSIrq6OAbm_/view?usp=sharing

The technical paper presentations for your "Paper ID- 97" - and "Title- SecureVote: Enhancing Electoral Integrity Using Blockchain-Based E-Voting" will be conducted as per schedule. Reach at the venue 30 minutes before schedule.
We encourage you to review the details and notify us of any changes or specific requirements on or before April 02, 2024, till 05:00 P.M.

Please do not request us to change the mode.

Your contribution to ICCSAI 2025 is highly valued, and we are looking forward to an engaging and insightful presentation from you. If you have any questions or concerns, please do not hesitate to reach out to us at iccsai@galgotiasuniversity.edu.in.

Additionally, we invite you to join the official WhatsApp group for conference updates:https://chat.whatsapp.com/GISw8870sjD9dbqpkPCLm6
For regular updates, please visit the conference website at https://iccsai.in
Note:

- Ensure that the similarity score (plagiarism) of your manuscript is not more than 10% (excluding bibliography) to be eligible for inclusion in the IEEE Xplore digital library.
- Complete and submit the copyright form to ensure your paper's appearance in the conference proceedings and enrolment in IEEE Xplore.
- Presentation certificates will be released only after the submission of the required documents.
- All accepted and presented papers will be submitted to IEEE for inclusion in the IEEE Xplore Digital Library.

Thank you for your valuable contribution, and we look forward to hosting you at ICCSAI 2025.

For any queries, please contact:

Dr. Ajeet Singh - 9560393898
Dr. Gaurav Aggarwal - 7011363441
Dr. K K Aggarwal - 7398032912
Dr. Shachi Mall- 99846 01301
Dr. Arpesh - 88608 88444

RESEARCH PAPER PRESENTATION PROOF

GALGOTIAS UNIVERSITY

ICCSAI

IEEE UP SECTION (INDIA)

Certificate of Presentation

This is to certify that

Dr./Mrs./Ms./Mr. Rhythm Garg; Aashish Gupta; Arti Sharma

has presented the paper entitled

SecureVote: Enhancing Electoral Integrity Using Blockchain-Based E-Voting

in 2025 3rd International Conference on Communication, Security and Artificial Intelligence, 04th-06th April 2025, Organised by Galgotias University, Greater Noida, UP.

Dr. Avadhesh Kumar	Dr. Nitin Kumar Gaur	Dr. K. Mallikharjuna Babu
General Chair	Registrar	Patron, ICCSAI-2025
Pro-VC, Galgotias University (U.P)	Galgotias University (U.P.)	VC, Galgotias University (U.P.)