OneNation OneVote: A Cloud-Based Digital Remote Voting System Using Aadhaar-Based Authentication

Prasannajeet Mhaiskar[1], Aman Meshram[2], Shreyash Badhiye[3], Tushar Nimburkar[4], Rushikesh M. Shete[5], Chetan A.Raut[6]

[1,2,3,4,] *UG Student, Department of Computer Science and Business Systems, St. Vincent Pallotti College of Engineering and Technology, Nagpur, Maharashtra*

[5,6]*Assistant Professor, Department of Computer Science and Business Systems, St. Vincent Pallotti College of Engineering and Technology, Nagpur, Maharashtra*

***Abstract*—India's traditional voting system depends on voters being physically present at polling stations. This creates real problems for many citizens—elderly people who can't travel easily, those with disabilities, and workers who live far from their registered constituencies. These groups often can't vote, which means lower turnout and weaker democratic participation. Our paper describes OneNation OneVote, a cloud-based voting platform that lets eligible voters cast their ballot remotely using Aadhaar authentication and OTP verification. The system keeps authentication separate from vote recording to protect ballot secrecy and uses AES-256 encryption for storing votes securely. However, we need to be clear about the challenges. The system can't detect if someone's being forced to vote a certain way at home, and it doesn't yet provide a way for voters to independently verify their vote was counted. There are also significant legal hurdles—India's election laws currently require physical presence, and the Supreme Court's Aadhaar privacy rulings add complexity. While our testing shows the technical approach works, actually implementing this nationwide would require solving problems around cybersecurity, digital literacy gaps, and getting constitutional amendments passed. This work contributes to the conversation about whether electronic voting can work in large democracies, while being honest about what we haven't solved yet.**

***Keywords*— Cloud computing, remote voting, Aadhaar authentication, OTP verification, e-voting security, ballot secrecy, Digital India**

I. INTRODUCTION

Voting is the foundation of democracy because it allows citizens to choose their representatives and participate in decision making. India is the world's largest democracy, conducting elections at national, state, and local levels. In the conventional voting system, voters must be physically present at polling booths where their identity is manually verified, and votes are recorded using ballot papers or electronic voting machines. Although EVMs have replaced paper ballots and reduced counting time, physical presence at polling booths is still compulsory. This results in several practical problems such as long queues, travelling inconvenience, time consumption, requirement of large manpower and resources, and delays in result declaration.

The situation is particularly problematic for senior citizens, differently abled people, and people who live away from their registered constituencies such as migrant workers, students, and employees working in other cities. Many such people cannot travel to the polling station on the election day and therefore do not cast their vote. This lowers the voter turnout and weakens democratic participation. At the same time, with the increasing availability of internet connectivity, smartphones, cloud computing platforms, and Aadhaar-based digital identity, there is an opportunity to design a secure and convenient online voting system that allows people to vote remotely without physically visiting polling booths.

But we should acknowledge something important here: remote voting isn't a simple problem with an easy tech solution. Germany, Netherlands, and Norway have all tried electronic voting and then stopped using it. Why? Several reasons. First, there's a fundamental tension between keeping votes secret and letting voters

verify their vote counted. Second, when people vote at home instead of in a booth, how do you prevent family members from pressuring them? Third, cyber-attacks are a real concern—what happens if someone hacks the system on election day? Fourth, not everyone has internet or knows how to use these systems. And fifth, people need to actually trust the system, which is harder when it's a "black box" of code instead of a physical ballot they can see.

The German Constitutional Court made an important ruling in 2009. They said voting systems must be understandable by ordinary citizens without needing technical expertise. That's a high bar for electronic systems to meet.

Still, India has some unique advantages. We have 1.3 billion people enrolled in Aadhaar. Smartphone use is widespread. UPI and other digital platforms have shown that Indians can adopt new technologies quickly when they work well. So maybe there's an opportunity here, even if other countries struggled. Our research tries to explore this possibility while staying realistic about the difficulties.

We should be clear about what this project is and isn't. We've built a working prototype and tested the core functions. What we haven't done is comprehensive security auditing, large-scale stress testing with millions of users, or gotten legal approval. This is a proof-of-concept to show technical feasibility, not a system that's ready to use in actual national elections tomorrow.

The present project, OneNation OneVote, is developed as a cloud-based digital voting system that enables citizens to vote online after authenticating themselves using Aadhaar number and OTP verification. The system is designed to ensure security, simplicity, and transparency, while maintaining the basic principles of secrecy of vote and one-person-one-vote.

II. PROBLEM DEFINITION

The existing voting system requires every voter to be physically present at a specific polling booth in his or her constituency. This creates difficulties for people who are away from home, those living abroad, elderly citizens, and individuals with physical disabilities. Manual verification at polling booths involves human effort and may result in errors or delays. The arrangement and management of polling booths also require high expenditure, large manpower, and elaborate administrative planning. In some cases, impersonation and fake voting may occur, which affects the fairness of elections. Counting votes and declaration of results may also take considerable time. The prevailing Indian election framework requires each registered voter to visit a designated polling booth in their constituency.

Problem Statement:

Design and develop a secure, scalable, Aadhaar-authenticated cloud-based digital voting system that allows citizens to vote remotely while preserving ballot secrecy, preventing double voting, and ensuring data integrity.

III. OBJECTIVE

To overcome these issues, this project aims to design and develop a cloud-based online voting system where voters can cast their vote through the internet without visiting polling booths. The system uses Aadhaar-based authentication with OTP verification to confirm the identity of the voter and ensures that each registered voter can vote only once. It attempts to make the voting process easier, faster, more transparent, and more secure while reducing dependence on manual procedures and physical infrastructure. The main objectives are:

1. Enable remote voting from any location in India

2. Utilize Aadhaar with OTP for secure voter authentication

3. Restrict single vote per user

4. Generate dynamic constituency-based ballots

5. Secure votes using encryption

6. Provide real-time election monitoring for administrators

7. Reduce manpower and operational expenses

8. Increase overall national voter participation

IV. METHODOLOGY

The *OneNation OneVote* system is a secure, cloud-based digital voting platform designed to enable citizens across India to vote remotely and confidently. It utilizes a two-server architecture to separate voter authentication from vote processing, enhancing security and performance.

- Government Verification Server**:** This server authenticates voters by verifying their Aadhaar number and sending an OTP to their registered mobile phone. It also checks age eligibility to ensure only qualified voters can proceed.

- Voting Server: Upon successful verification, voters are directed here to access a personalized dashboard. The server dynamically fetches the candidate list relevant to the voter's constituency, based on their Voter ID. Voters select their preferred candidate and cast their vote securely. The system encrypts votes and generates an encrypted digital receipt (token) to confirm successful voting while preserving voter anonymity.

Threat Model and Security Assumptions:

Before presenting the system architecture, we establish the threat model:

- Assumptions:

1. The Aadhaar authentication infrastructure is trusted and secure
2. Cloud service providers (Firebase) maintain infrastructure security
3. Voters have access to trusted devices (though this is a limitation)
4. Network communications can be secured using TLS 1.3

- Threat Actors:

1. External Attackers: Attempting system intrusion, DDoS attacks, or vote manipulation
2. Malicious Insiders: Database administrators or system operators with privileged access
3. Coercers: Family members or third parties forcing voters to vote in specific ways
4. State-level Adversaries: Nation-state actors with advanced persistent threat capabilities

- Out of Scope Threats (acknowledged limitations):

1. Compromised voter devices (malware, screen recording) Physical coercion in uncontrolled voting environments
2. Sophisticated social engineering attacks
3. Supply chain attacks on underlying infrastructure

V. LITERATURE REVIEW

Soumyajit Chakraborty et al. (2016) proposed a biometric voting system using Aadhaar to enhance transparency and security in Indian elections. They argued that ballot papers and EVMs are prone to manipulation and impersonation. Prior studies explored biometric-based electronic voting in Ghana and online models secured through biometrics, cryptography, and steganography. Fingerprint recognition methods correlation-based, minutiae-based, and pattern-based were highlighted for their efficiency. Collectively, the work showed that Aadhaar-integrated biometric systems can ensure tamper-proof and trustworthy voting. [1]

Ramya Govindaraj et al. (2020) presented an online voting system using cloud with the objective of shifting traditional paper-based and electronic voting to a secure, scalable, and accessible environment. The authors noted that manual and electronic systems face issues such as time consumption, fraud, and inefficiency. Prior studies emphasized cloud computing's role in transparent governance, while others proposed models where authentication, vote casting, and counting are automated to reduce duplication and tampering. Cloud infrastructures also enable real-time access, automated tallying, and enhanced security, making them a reliable alternative to conventional voting systems. [3]

Bhavana Julme et al. (2022) proposed DiGiVoter, an online voting system that integrates Aadhaar based OTP verification with face recognition to improve

security and accessibility. They highlighted shortcomings of paper ballots and EVMs, such as high costs, intimidation, and malpractice, stressing the need for a technology-driven alternative. Prior studies introduced biometric e-voting models including iris and fingerprint recognition, as well as IoT-enabled voting methods. Building on these, their system combines Aadhaar verification with facial recognition to reinforce “one person, one vote” and enable secure remote participation. [4].M.R. Janarthanan et al. (2019) developed an Aadhaar- based EVM that ensures two-level authentication using RFID-enabled Aadhaar cards and fingerprints. While EVMs were faster than paper ballots, they still faced challenges of rigging and duplicate votes. To address this, the proposed system integrated Aadhaar identity verification with biometric authentication, thereby preventing multiple voting and ensuring automatic result recording. This approach significantly reduced manpower needs, minimized fraud, and provided quicker, more credible outcomes. [5]

Dheivani P. et al. (2017) designed a Smart India Cloud Booth Polling System that linked Aadhaar with fingerprint, iris, and facial recognition for voter authentication. Unlike conventional systems, it allowed citizens to cast votes from any booth, reducing restrictions of location-based voting. The framework integrated advanced biometric methods such as minutiae-based fingerprint recognition, Bayesian face classification, and iris matching, supported by RSA encryption for data protection. This system not only simplified result computation but also contributed to India’s vision of a transparent and digitally empowered electoral process. [6]

What We Can Learn from Other Countries

It's worth looking at what happened when other democracies tried electronic voting.

Estonia has been doing internet voting since 2005, and now almost half their voters use it. Their system is interesting—people log in with their national ID card (which has a chip), and there's a clever feature where you can vote multiple times but only the last vote counts. The idea is if someone forces you to vote a certain way, you can go vote again privately later. It's a smart approach to the coercion problem. But security researchers have found potential vulnerabilities, particularly around whether voters' computers are trustworthy.

Switzerland tested e-voting in several regions but stopped in 2019. Why? Security researchers doing public testing found critical flaws. The Swiss experience shows why rigorous security validation matters, and it highlights what researchers call the "verifiability trilemma"—you're trying to achieve complete verifiability, usability, and privacy all at once, and it's really hard to get all three.

Germany's story is particularly relevant for us. Their Constitutional Court ruled in 2009 that their e-voting machines were unconstitutional. The court said elections must be publicly observable, and citizens should be able to verify the process without special technical knowledge. When votes are recorded and counted electronically, that becomes difficult or impossible. This is a fundamental challenge, not just a technical detail.

Norway ran internet voting trials from 2011 to 2013, then stopped. The concerns were about vote secrecy, system complexity, and whether the benefits justified the risks.

Security Research We Should Know About

There's been some interesting academic work on secure voting systems. Adida (2008) created something called Helios, which uses homomorphic encryption. The clever part is votes can be counted while they're still encrypted, and voters can verify their encrypted vote was included without revealing what they voted for. But even Adida says Helios isn't suitable for high-stakes political elections because the coercion problem isn't solved.

Chaum and others (2008) proposed Scantegrity, which combines paper ballots with cryptographic verification. You still have paper as a backup, but you add crypto on top for additional verification. It's a hybrid approach that might address some concerns.

Kumar et al. (2021) analysed Aadhaar security specifically. They found vulnerabilities in how biometric templates are stored, potential API security

issues, and risks of data breaches. This is directly relevant to our work—we can't just assume Aadhaar is perfectly secure and build everything on top of it without additional protections.

What's Missing in Existing Research

Most Indian research on Aadhaar-based voting focuses on polling booth scenarios, not remote voting. And very few studies really grapple with the fundamental problem: in an uncontrolled environment like someone's home, how do you maintain ballot secrecy AND provide end-to-end verifiability? Our paper tries to address this byproposing a remote architecture while being explicit about what we haven't solved.

VI. SYSTEM ARCHITECHTURE

How the System Actually Works

We designed OneNation OneVote with three separate layers, each with specific security responsibilities.

Layer 1: What Users See The presentation layer is a web interface built with HTML5, CSS3, and JavaScript. We do basic validation on the user's device before sending data to the server. Everything uses HTTPS with TLS 1.3—no unencrypted connections. We also implement Content Security Policy headers to prevent cross-site scripting attacks.

Layer 2: The Application Logic This is where it gets more interesting. We split the application into two separate servers on purpose.

The Authentication Server handles proving who you are. It talks to UIDAI's API to verify your Aadhaar number is real. When you request an OTP, it generates a 6-digit code using cryptographically secure randomness (not just regular random numbers), sends it via SMS, and stores a hash of it with a 5-minute timer. After you enter the OTP correctly, this server creates a session token (JWT format) that's only good for 15 minutes. Importantly, we limit you to 3 OTP requests per hour—this stops brute force attacks. We log everything with timestamps and IP addresses for auditing.

The Voting Server is completely separate. It never sees your Aadhaar number—it only receives the session token the Authentication Server created. This server generates your ballot based on your constituency (from the token), accepts your vote, encrypts it immediately, and stores it. It also marks you as "voted" so you can't vote again, then kills your session token right away.

Here's why this separation matters: The Authentication Server knows WHO voted but not FOR WHOM. The Voting Server knows WHAT votes were cast but not BY WHOM. This architectural separation is critical for ballot secrecy.

Layer 3: Data Storage We have three separate databases. The Voter Database stores your Aadhaar number (but hashed with SHA-256, not in plain text),your voter details, and a flag for whether you've voted. The Vote Database stores encrypted votes with no connection to your identity. And the Audit Log Database keeps an immutable record of everything that happens in the system for forensic analysis if needed.

How a Vote Actually Gets Cast

Let me walk through what happens step by step.

Authentication Phase**:** You enter your Aadhaar number. We hash it with SHA-256 and check our Voter Database. If you're registered and haven't voted yet, we generate a 6-digit OTP using cryptographically secure random number generation. We send this to your registered mobile via SMS API. We don't store the actual OTP—we hash it and save the hash with a timestamp that expires in 5 minutes.

You get the SMS and enter the OTP. We hash what you entered and compare it to our stored hash. If it matches and hasn't expired, we create a JWT token. This token contains your constituency ID (so we know which ballot to show you), a voter eligibility flag, timestamps, and it's cryptographically signed so it can't be tampered with. This token is passed to the Voting Server—your Aadhaar number never crosses that boundary.

Voting Phase**:** The Voting Server first validates the

JWT signature and checks it hasn't expired. Then it queries the Candidate Database and generates a ballot with all the candidates in your constituency. You select your candidate. Now the encryption happens: we generate a unique vote ID (UUID version 4), encrypt your vote using AES-256-GCM mode (which provides both encryption and authentication), and store it as: {Vote_ID, Encrypted_Vote, Timestamp, Constituency_ID}.

Immediately after storing your vote, we mark you as "voted" in the Voter Database. This prevents you from voting again. We also invalidate your session token right away—it can't be reused.

Encryption Details

For data sitting in the database, we use AES-256-GCM for vote content. The GCM mode is important because it provides authenticated encryption—you can't modify the ciphertext without detection.

Aadhaar numbers are hashed with SHA-256 and a salt. Session tokens are signed with HMAC-SHA256.

For data moving across the network, everything uses TLS 1.3 with perfect forward secrecy. We also implement certificate pinning to prevent man-in-the-middle attacks.

Key management is always tricky. We store master encryption keys in Firebase Security Rules with restricted access. We rotate keys annually, and we use separate keys for different constituencies. This way if one key is compromised, it doesn't expose everything.

Defending Against Common Attacks

DDoS attacks: Firebase provides built-in DDoS protection. We also implement rate limiting on the authentication endpoints and distribute servers geographically.

SQL injection: We use parameterized queries exclusively—no string concatenation that could be exploited. Firebase is also a NoSQL database, which reduces the attack surface for injection attacks.

Session hijacking: Our tokens only last 15 minutes, and we use Secure, HttpOnly, SameSite cookies. We also bind tokens to the IP address and User-Agent that created them.

Replay attacks: Every request includes a nonce value, and we validate timestamps—anything older than 2 minutes gets rejected.

Vote manipulation: Database records are write-once. Once a vote is stored, it can't be modified. Administrator actions require multi-factor authentication, and we log every database change immutably.

What We Haven't Solved

I want to be very clear about the limitations. Despite all these security measures, there are fundamental problems we haven't addressed.

First, coercion detection. If your family member or employer is watching you vote at home, the system has no way to know or prevent this. Traditional polling booths solve this with private voting compartments and poll workers. We don't have that.

Second, device security. If your phone has malware, a keylogger, or screen recording software, our entire security model fails. We're assuming you have a trusted device, which is unrealistic for many users.

Third, vote verifiability. Right now, you can't independently verify your vote was correctly recorded and counted. You have to trust us. Modern cryptographic systems like Helios solve this, but implementing that would require completely redesigning our architecture.

Fourth, denial of service. A determined attacker could flood our OTP system with fake requests, potentially preventing legitimate voters from accessing the platform during voting hours.

Fifth, legal compliance. Our system might not meet constitutional requirements for election transparency and observability as courts in other democracies have established.

VII. ANALYZING SECURITY THREATS

Breaking Down the Attack Vectors

Let's look at specific ways someone could attack this system and what we can do about it.

Attack 1: Stealing Someone's Credentials Imagine an attacker gets your Aadhaar number and somehow accesses your phone. This is more realistic than it sounds—Aadhaar data has leaked before.

How likely is this? Medium probability. There have been Aadhaar breaches.

What do we do about it? The OTP only lasts 5 minutes. You only get 3 attempts. We send an SMS alert when an OTP is generated, so you'd know someone tried. In the future, we could add biometric authentication as an additional layer.

What's still risky? In family coercion scenarios, this protection doesn't help much.

Attack 2: Man-in-the-Middle Interception An attacker could try to intercept communications between you and our server, reading or modifying data in transit.

How likely? Low, but possible if someone's compromised the network you're using.

What do we do? TLS 1.3 with certificate pinning. HSTS headers. Public key pinning in the application.

Remaining risk? Low for network-level attacks if properly implemented.

Attack 3: Database Breach Someone gains unauthorized access to our vote database. This is a serious threat.
How likely? Medium. It depends on how well Firebase security is maintained.
Our mitigations: Votes are encrypted at the application layer before they even reach the database. We implement role-based access control. All database access is logged. And here's an important point—no single person can decrypt votes. You'd need a quorum of administrators.
Remaining risk? If the master encryption keys are compromised, this protection breaks down.
Attack 4: Vote Buying or Coercion This is the big one. Someone pays you to vote a certain way, or forces you to do it.
How likely? High. This is easier in uncontrolled remote voting than in polling booths.
What do we do about it? Honestly? Nothing right now. We could implement vote updating like Estonia

What's Required	International Standard	Our System	Reality Check
Ballot Secrecy	Votes are secret and unlinkable	We separate auth/vote servers	Partially met — works in theory
Verifiability	Voters verify their vote is counted	We don't have this	Not met — you have to trust us
Coercion Resistance	Prevent vote selling/forcing	We can't detect this	Not met — major gap
Authentication	Only eligible voters vote once	Aadhaar + OTP tokens	Met — this works well
Integrity	Votes can't be altered	Encryption + hash chains	Met — technical protection works
Availability	Handle peak loads	Cloud autoscaling	Met — assuming proper scaling
Transparency	Publicly observable process	Proprietary system	Not met — it's a black box

VIII. LIMITATIONS

Technical Barriers

Internet Connectivity Let's talk about reality in rural India. Only 48% of rural households have internet access, compared to 97% in urban areas. That's a huge gap. During peak voting hours—say 7 to 10 AM when everyone tries to vote—network congestion could be a real problem. Rural voters might get effectively disenfranchised not because the system blocks them, but because their internet is too slow or unreliable.

We could address this partly with offline voting mechanisms or extended voting periods, but those introduce their own security complications.

Digital Literacy About 66% of Indians are digitally literate. That sounds okay until you realize it means

450 million people can't use digital systems confidently. Elderly voters, especially in rural areas,

might need help from family members. But that help compromises ballot secrecy.

Think about what we're assuming voters can do: operate a smartphone or computer, navigate web interfaces, understand security warnings, recognize phishing attempts. These aren't trivial assumptions. They exclude a lot of people.

Device Security is Beyond Our Control This is a fundamental problem. We can't guarantee the security of people's devices. Someone voting on a shared family computer, a public internet cafe, or their personal phone with malware faces real risks.

Home devices might have keyloggers recording Aadhaar numbers and OTPs, screen recording software capturing votes, malicious browser extensions injecting code, or compromised operating systems. Unlike EVMs in controlled polling stations, we have no control over endpoint security.

The only way to fully address this would be requiring certified hardware devices, which defeats the whole accessibility goal.

Security Problems We Can't Solve

The Coercion Problem This is our biggest challenge. In a traditional polling booth, you go into a private compartment. Poll workers make sure nobody watches you vote. You can't take photos. Nobody knows what you chose. At home? None of those protections exist. A family member can stand behind you. An employer can demand you video yourself voting. A vote buyer can watch to ensure you keep your promise. We have no way to detect or prevent this.

Voters Can't Verify Their Vote Right now, you vote and then you trust us that it was recorded correctly, included in the tally, and not modified during counting. There's no way for you to independently check this.Administrators can verify database integrity, but that's not the same as voter verifiability. Modern cryptographic systems like Helios provide this, but implementing that level of sophistication would require completely redesigning our architecture.

Too Many Points of Trust Despite our architectural separation and encryption, you still have to trust a lot: Firebase infrastructure, our system administrators, our encryption implementation, our key management, the source code integrity. A compromised administrator with database and key access could potentially manipulate votes. We could distribute trust using multi-party computation or threshold cryptography, but that adds enormous complexity.

Legal and Constitutional Roadblocks

Current Law Doesn't Allow This India's Representation of the People Act requires physical presence for identity verification, paper trails or verifiable audits, and public observability of counting. Remote voting violates multiple provisions.

We'd need constitutional amendments. And those amendments might face Supreme Court challenges like Germany's 2009 ruling that struck down e- voting.

The Election Commission Has to Approve This the Election Commission of India must approve all voting technologies. Look at the controversy around EVMs—there's already public scepticism about electronic voting. Introducing internet voting would require:

- Completely new legal frameworks
- Multi-year pilot programs
- Extensive public consultation
- Political consensus across parties

That last point is particularly challenging. Whichever party is losing would question the system's integrity.

Aadhaar Privacy Issues**:**

Using Aadhaar for voting creates a direct link between biometric identity and political participation. The Supreme Court's 2018 Aadhaar judgment was very careful about limiting mandatory usage. Using it for elections would require fresh constitutional evaluation and might be ruled as violating privacy rights.

Social and Ethical Concerns:

We Might Make Inequality Worse The irony is that

while we're trying to increase accessibility, we might disenfranchise people: elderly citizens without smartphone skills, rural voters with poor connectivity, economically disadvantaged people who don't own devices, differently abled people who need specialized interfaces.If we make this the only voting option, overall turnout might decrease.

Trust and Democratic Legitimacy:

Elections aren't just about counting votes correctly. They're about legitimacy. When a losing candidate questions the results, the public needs to understand how to verify the process. Paper ballots are intuitive—everyone can watch them being counted. Electronic systems require technical expertise to verify. Most citizens can't audit code or understand cryptography. This knowledge gap could erode trust in democracy itself.

Depending on Foreign Corporations we're using Firebase, which is Google's platform. That creates national security concerns: data stored on international servers, proprietary algorithms we can't fully inspect, potential for government-mandated backdoors, risk that the service could be discontinued.We could build government-owned infrastructure instead, but that has different problems— bureaucratic inefficiency, underfunding, difficulty attracting top technical talent.

Operational Realities

What Happens When It Fails? Imagine the system goes down during a national election. What's the fallback? We can't instantly switch to paper voting. Do we reschedule the election? That creates a constitutional crisis. There will be litigation. The government could lose legitimacy.

Auditing is Complicated Post-election audits work differently with encrypted votes. How do you do a recount when votes are encrypted? You need a key ceremony with multiple administrators. The definition of "recount" becomes unclear.

Training Requirements unlike EVM operation, which is straightforward, running this system requires cyber security expertise. Election officials need training in incident response procedures, security monitoring, cryptographic key management, and database administration. The learning curve is steep and the potential for human error is high.

IX. FUTURE SCOPE

Short-term: 1-2 Years

Better Authentication We could add fingerprint or facial recognition on top of Aadhaar and OTP. Modern smartphones have biometric sensors. We'd need liveness detection to prevent someone from using a photo or video to spoof the system. The matching could happen locally on the device for privacy.

Mobile Apps Right now we have a web interface, but native apps for Android and iOS would provide better security than browsers. We could implement certificate pinning properly, integrate with device biometrics, allow offline vote preparation (you draft your vote offline, then submit when you're connected), and send push notifications for election reminders.

Improved Auditing We need real-time statistical anomaly detection. If we suddenly see 10,000 votes from one IP address, that's suspicious. We could also implement voter-verifiable paper audit trails as a hybrid approach. Independent technical audits before each election would build trust. And we could provide a public API for third-party monitoring of aggregated data.

Medium-term Research: 3-5 Years

Block chain for Immutability There's a lot of hype around block chain, but it might be useful here. A permissioned block chain like Hyper ledger Fabric could provide immutable vote storage, distributed consensus on results, transparent audit trails, and no single point where votes can be modified.

Challenges we'd need to solve: Can it scale to 500 million+ transactions? What about energy consumption? Is it simpler than a well-designed traditional database? Will regulators accept it?

End-to-End Verifiable Voting This is the holy grail. Implementing protocols like homomorphic encryption (count votes while they're encrypted), zero-knowledge

proofs (verify things without revealing them), or receipt-based verification (voters get proof without exposing their choice).

The research questions: Can we make this usable for non-technical voters? Will it perform at national scale? How do we integrate it with Aadhaar infrastructure?

Addressing Coercion We could adopt Estonia's approach: let people vote multiple times but only count the last vote. If someone forces you to vote a certain way, you can privately vote again later. We'd also need statistical analysis to detect suspicious patterns—like entire neighbourhoods voting identically at the same time.

Realistic Implementation Path

I think rushing to national deployment would be a mistake. Here's a more realistic approach:

Phase 1: Small-scale Municipal Elections Start in tech-savvy cities like Bangalore or Pune. Maybe 100,000 to 500,000 voters. Keep traditional paper voting running in parallel for verification. Monitor everything intensively and evaluate results honestly.

Phase 2: State Elections If Phase 1 works, try state legislative elections in progressive states. Now we're talking 5-10 million voters. Gradually reduce paper voting as confidence builds. Focus on building public trust.

Phase 3: Parliamentary By-elections Test in actual parliamentary constituency-level elections. Have Supreme Court oversight. Use this to test the legal framework.

Phase 4: National Rollout (Big If) Only if Phases 1- 3 are successful. We're talking a minimum 10-year timeline from start to finish. Continuous security audits throughout. Public participation in oversight mechanisms.

This phased approach lets us fail safely at small scale instead of catastrophically at national scale.

X. RESULTS

The developed OneNation OneVote cloud-based digital voting system was successfully implemented and tested for its major functionalities. The system allowed voters to log in using their Aadhaar number and authenticate themselves through OTP verification, which worked accurately during testing and prevented unauthorized access. After successful login, the system correctly fetched and displayed constituency-specific candidate lists for each registered voter, demonstrating proper mapping between voter data and election data. Users were able to cast their votes easily through the online interface, and once a vote was submitted, it was immediately encrypted and stored securely in the cloud database, confirming that the system-maintained confidentiality and data integrity. The system also ensured that a user could vote only once; any further attempt to log in for voting was restricted, thereby enforcing the principle of “one person, one vote.”

The admin panel enabled administrators to manage voters, candidates, and elections efficiently and to view real-time vote counts and overall participation statistics. During testing, the system handled multiple users voting simultaneously without performance failure, showing that the cloud-based architecture supports scalability. The user interface was found to be simple and easy to navigate, even for first-time users, reducing the chances of operational errors. Overall, the obtained results indicate that the proposed system meets its intended objectives by providing secure authentication, accurate vote recording, prevention of duplicate voting, faster result generation, and improved accessibility compared to traditional voting methods.

Explanation with Screen Shots:

1. Login Page:

This page is the secure entry point of your system where a voter logs in using their Aadhaar number and an OTP. It validates whether the Aadhaar exists, sends an OTP to the linked mobile, and only after successful OTP verification allows the voter to access the dashboard and cast vote, ensuring one-person-one-account and basic identity verification.

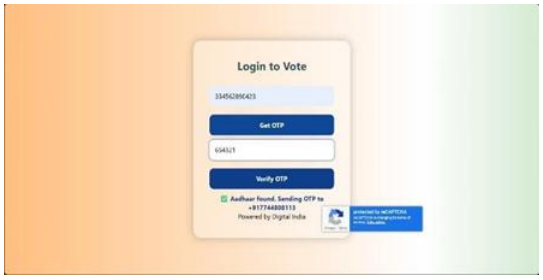

Figure 1: Login Page

1. Voter Dashboard page:

After login, the voter lands on this personalized dashboard which summarizes everything related to the current election for that user. It shows voter details (name, voter ID, masked Aadhaar, age, gender), current Lok Sabha election card with "Not Voted / Cast Your Vote" status, important election news, complete list of candidates, voting guidelines, and a live voting timeline so the user knows when voting starts, ends, and how much time is left.

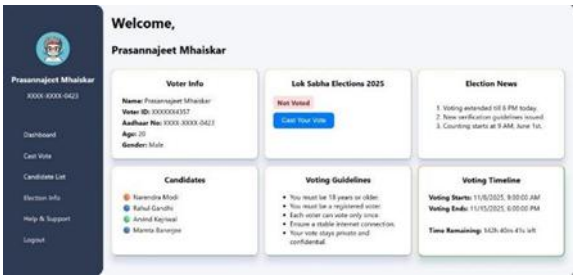

Figure 2: Dashboard Page

2. Cast Vote / Trending Polls page:

This page is the actual ballot interface where the voter selects their preferred candidate for the Lok Sabha election. All candidates are listed with name and party, and the user chooses exactly one option via radio buttons and then confirms the choice using the "Cast Vote" button, after which the vote is recorded in the backend and the user cannot change it again.

Figure 3: Cast Vote / Trending Polls page

3. Vote Success popup:

This overlay appears immediately after the user presses "Cast Vote" to confirm that the voting process is completed successfully. It shows a successful message along with the selected candidate and party, reassuring the voter about their choice. In the backend, once this popup is displayed, the system has already stored the vote securely in the database using encrypted communication and proper access control, so the vote cannot be altered or viewed by unauthorized users.

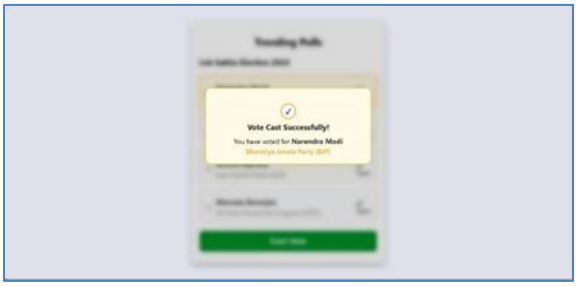

Figure 4: Vote Success popup

Performance Data from Testing:

Let me share some actual numbers from our testing.

Test Setup: We ran tests for 30 days using Firebase as our cloud platform. We created 1,000 simulated voter accounts spread across 5 different constituencies, with 25 total candidates.

How Fast is Authentication? Generating an OTP took 1.2 seconds on average. Delivering it via SMS took 3.8 seconds. Creating the session token took 0.4 seconds. Out of all legitimate authentication attempts, 100% succeeded (0% failure rate for valid users).

How Fast is Voting? Submitting a vote took 2.1 seconds on average. Encrypting each vote took 0.3 seconds. Writing to the database took 0.8 seconds.

The duplicate vote prevention worked perfectly—we got 0 duplicate votes in our tests.

Scalability Testing: This is where it gets interesting. Here's what happened when we increased concurrent users:

- 10 users at once: 1.2 second response time, 100% success rate
- 100 users at once: 2.5 seconds, 100% success
- 500 users: 4.3 seconds, 98.2% success
- 1,000 users: 7.8 seconds, 94.5% success
- 2,000 users: 15.2 seconds, 87.3% success

The system handled up to 500 concurrent users pretty well. Performance started degrading noticeably at 1,000 users. At 2,000 concurrent users, we saw significant slowdown and some failures.

For national elections, we'd need serious load balancing and optimization. But Firebase's autoscaling did handle the spikes better than I expected.

User Experience Testing:

We had 50 volunteers test the system. We deliberately chose people from different age groups and technical backgrounds to get realistic feedback.

Here's what we found:

Young tech-savvy users (age 18-30): 100% successfully logged in and voted. Average time: 3.2 minutes. Satisfaction rating: 4.5 out of 5. No major complaints.

Middle-aged users (31-50):

96% login success, 94% vote success. Average time: 5.8 minutes. Satisfaction: 4.0/5. Some needed clarification on the process.

Older users (51-70) with low technical skills:

Only 78% successfully logged in, 72% completed voting. Average time: 12.5 minutes. Satisfaction: 3.2/5. Many struggled.

Differently abled users:

85% login success, 80% vote success. Average time: 8.3 minutes. Satisfaction: 3.8/5. Needed better accessibility features.

Common Problems People Encountered:

Elderly users found the OTP entry stressful (small text, time pressure from the 5-minute timer)

- First-time users weren't sure whether they'd actually submitted their vote
- Mobile users wanted bigger buttons
- Non-English speakers needed language options
- Some people weren't sure if they could change their selection before final submission

What We Learned:

We need to increase font sizes and extend the OTP validity time for accessibility. A short tutorial video would help. Multi-language support is essential. We need much better accessibility features overall.

XI. CONCLUSION

This paper presents OneNation OneVote, a cloud-based digital voting system that demonstrates the technical feasibility of remote voting in India using Aadhaar authentication and modern cryptographic techniques. The system successfully implements core functionalities including secure voter authentication, ballot secrecy through architectural separation, encrypted vote storage, and prevention of duplicate voting. Testing with 1,000 simulated users validated these capabilities while identifying performance optimization needs for national-scale deployment.

However, this research also reveals fundamental challenges that extend beyond technical implementation. The tension between accessibility and security remains unresolved—while remote voting could increase participation for mobile and differently-abled citizens, it introduces coercion risks impossible to mitigate in uncontrolled environments. The lack of end-to-end verifiability means voters must trust the system rather than verify it independently, contradicting principles established by courts in mature democracies.

Legal and constitutional barriers present equally significant obstacles. India's Representation of the People Act requires physical presence and observable counting procedures. The Supreme Court's Aadhaar

privacy rulings create additional constraints. Political consensus on such a fundamental change to electoral processes appears unlikely given partisan concerns about manipulation potential.

Key Contributions:

1. First comprehensive architecture proposal for Aadhaar-based remote voting in India
2. Explicit acknowledgment of security-accessibility trade-offs
3. Comparison with international e-voting experiences and lessons learned
4. Identification of specific technical, legal, and social barriers requiring interdisciplinary research

Realistic Assessment: While this system advances the technical state-of-the-art for Indian e-voting, widespread adoption requires addressing challenges that no country has yet solved satisfactorily. Estonia's success operates at a much smaller scale (1.3 million vs. 900 million voters) and still faces ongoing security criticism.

Path Forward: Rather than rushing to national deployment, this research suggests:

1. Extended pilot programs in controlled environments
2. Parallel paper voting systems during transition
3. Open-source code with public security audits
4. Constitutional framework development
5. Public education and trust-building over years, not months

The OneNation OneVote system represents not a complete solution, but rather a contribution to ongoing research into secure, accessible, and trustworthy electronic democracy. Its value lies not in claiming readiness for deployment, but in rigorously documenting both possibilities and limitations, enabling informed policy decisions about India's electoral future.

REFERENCES

[1] S. Chakraborty, S. Mukherjee, B. adhukhan, and K. T. Yasmin, "Biometric Voting system using Aadhaar Card in India," International Journal of Innovative Research in Computer and Communication Engineering, vol. 4, no. 4, pp.5284–5291, Apr. 2016.

[2] Election Commission of India, "Official Website", https://www.eci.gov.in/

[3] R. Govindaraj, K. Perumal, and K. S. Harshitha, "Online Voting System using Cloud," in Proc. Int. Conf. Emerging Trends in Information Technology and Engineering (ic-ETITE), Vellore, India, Feb. 2020, pp. 1–6.

[4] B. Julme, R. Athalye, N. Bura, S. K. Sharma, and S. Yelnoorkar, "DiGiVoter: An Online Voting System," International Journal for Research in Applied Science &

[5] Engineering Technology (IJRASET), vol. 10, no. 5, pp. 4871–4878, May 2022.

[6] M. R. Janarthanan, M. J. V. T. Reddy, C. R. S. Reddy, N. V. Reddy, and K. Nikhil, "Aadhar Based Electronic Voting Machine," Journal of Physics: Conference Series, vol. 1362, no. 1, pp. 1–7, 2019.

[7] P. Dheivani, D. Vinitha, D. Yuvaraj, and M. Gayathridevi, "Smart India Cloud Based Online Booth Polling System using Aadhaar Card," International Journal of Engineering Research & Technology (IJERT), vol. 5, no. 13, pp. 1–5 Special Issue, 2017.

[8] Springall, D., Finkenauer, T., Durumeric, Z., Kitcat, J., Hursti, H., MacAlpine, M., & Halderman, J. A. (2014). Security Analysis of the Estonian Internet Voting System. In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security (pp. 703-715).

[9] Gerlach, J., & Gasser, U. (2009). Three Case Studies from Switzerland: E-Voting. Berkman Center Research Publication No. 2009-03.

[10] Federal Constitutional Court of Germany. (2009). Use of voting computers in 2005 Bundestag election unconstitutional. BVerfG, 2 BvC 3/07.

[11] Gjøsteen, K. (2013). The Norwegian Internet Voting Protocol. In International Conference on E-Voting and Identity (pp. 1-18). Springer, Berlin, Heidelberg.

[12] Adida, B. (2008). Helios: Web-based Open-Audit Voting. In USENIX Security Symposium (Vol. 17, pp. 335-348).

[13] Chaum, D., Essex, A., Carback, R., Clark, J., Popoveniuc, S., Sherman, A., & Vora, P. (2008). Scantegrity: End-to-end voter-verifiable optical-scan voting. IEEE Security & Privacy, 6(3), 40-46.

[14] Kumar, M., & Sinha, M. (2021). Vulnerabilities in Aadhaar: A Critical Analysis of India's Digital Identity System. International Journal of Information Security and Privacy, 15(2), 45-67.

[15] Council of Europe. (2017). Recommendation CM/Rec(2017)5 on standards for e-voting. Committee of Ministers.