E-Vote – Biometrics Based Secure Online Voting System

Muhammad Zaid Dildar

Saad Mehmood

Muhammad Ahmad

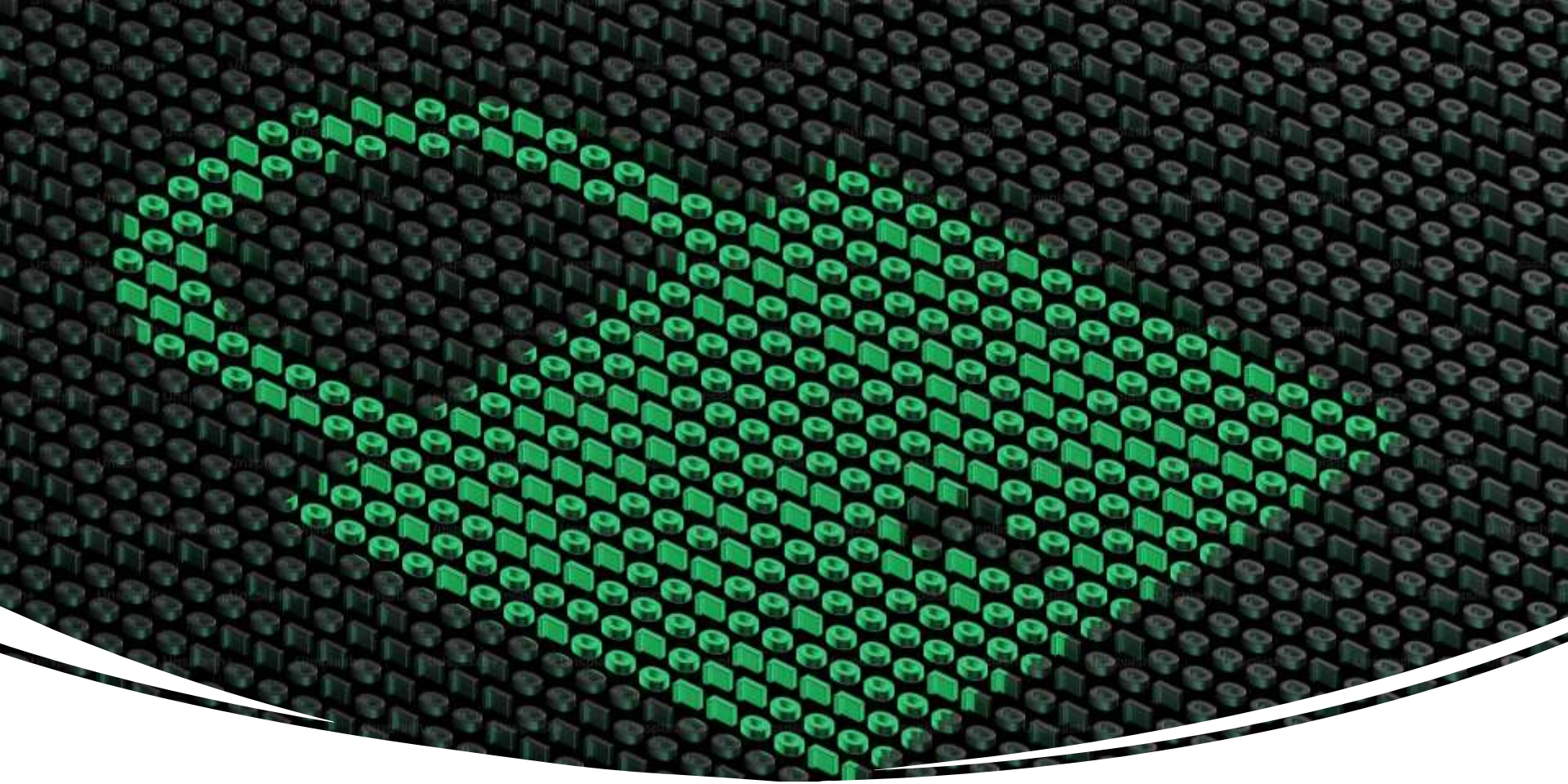
Supervised by:
Lecturer Fawad Khan

Agenda

- Problem & Motivation
- Our Proposed Solution
- Architecture Overview
- Implementation Details
- Testing & Evaluation
- SDGs & Future Scope

Problem Statement

- Time-consuming & error-prone
- Paper usage → not eco-friendly
- Inaccessible for off-campus students
- Hard to audit
- Low participation



Motivation

- Secure and accessible digital elections
- Transparency & trust
- Inspired by WebAuthn and real-time dashboards

Our Solution: E-Vote

- Biometric login using WebAuthn
- Admin, Auditor, Voter modules
- Real-time results & monitoring
- Built with Next.js, Express.js, MongoDB

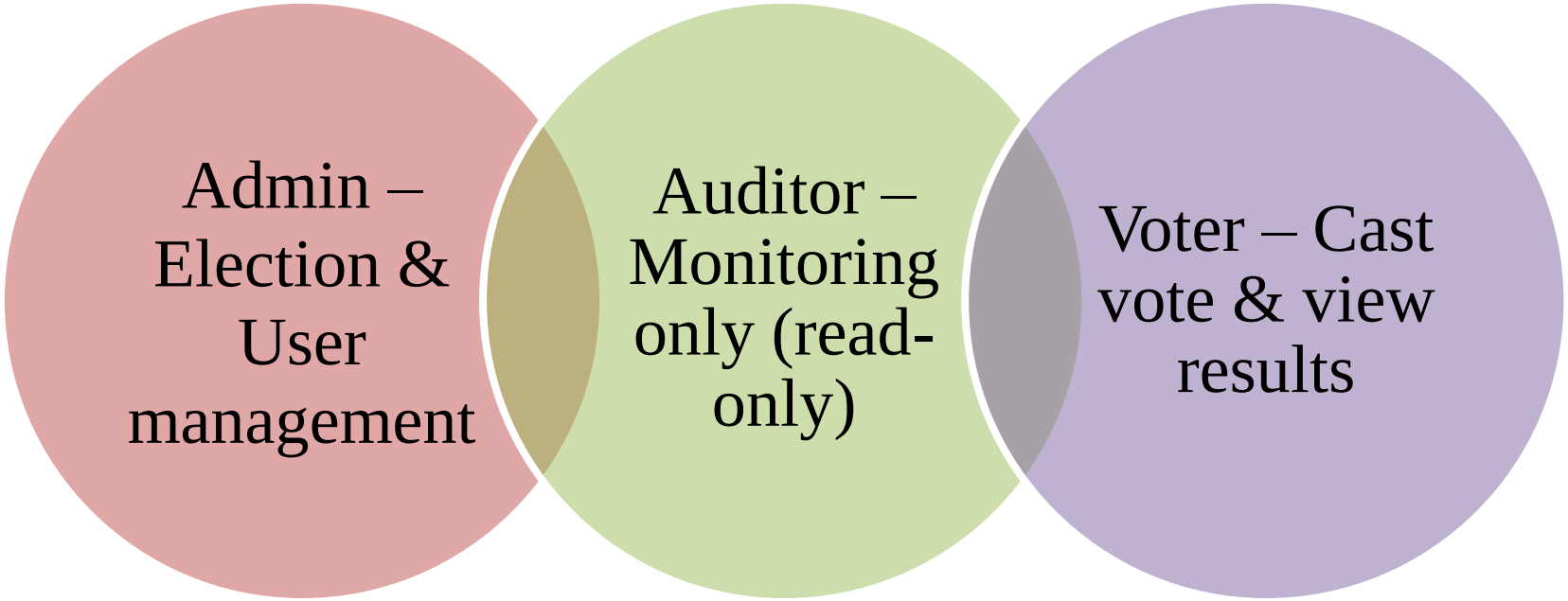
Objectives

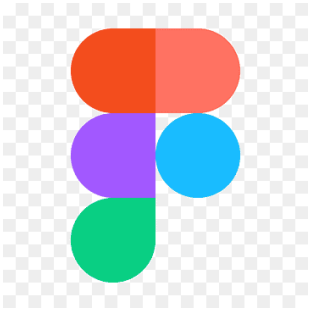
- Secure & transparent voting
- Real-time election analytics
- Simplified admin process
- Boost student engagement
- User Friendly

Scope of the System

- Covers full election lifecycle
- Role-based access
- Integrated analytics & audit logs

Key Modules


Admin – Election & User management
Auditor – Monitoring only (read-only)
Voter – Cast vote & view results



System Architecture

- UI/UX Design: Figma
- Frontend: Next.js
- Backend: Express.js
- Database: MongoDB
- Deployed on Vercel
- Role-based architecture

```
web/                                  # Next.js web app (TypeScript)
├── app/
│   ├── api/                          # API routes to communicate with backend
│   │   ├── admin/                    # API routes for admin
│   │   ├── audit/                    # API routes for auditor
│   │   └── user/                     # API routes for user
│   ├── admin/                        # Admin Module Pages
│   ├── audit/                        # Auditor Module Pages
│   ├── user/                         # Voter Module Pages
│   └── components/                   # Reusable UI components
├── next.config.json                  # next config
└── tailwind.config.json              # tailwind config
```

Frontend Structure

- Component-based React design
- Server-Side Rendering
- Dynamic routing
- Biometric API Integration

Backend APIs

- Models for users, elections and votes
- Service functions for users, elections and votes
- Controllers for users, elections and votes
- JWT-based access control

Database Design

- User: with biometric fields
- Election: candidates, timing
- Votes: secure + timestamped
- AuditLogs: trace every action

Biometric Authentication

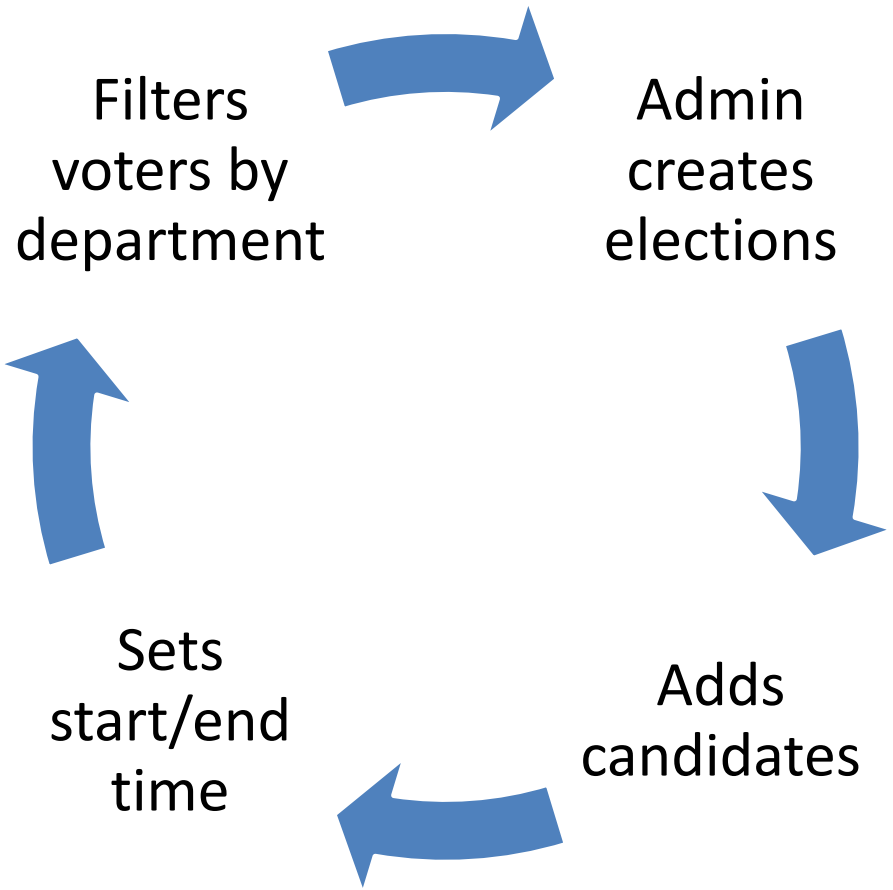
WebAuthn standard used

Biometric-only login enforced

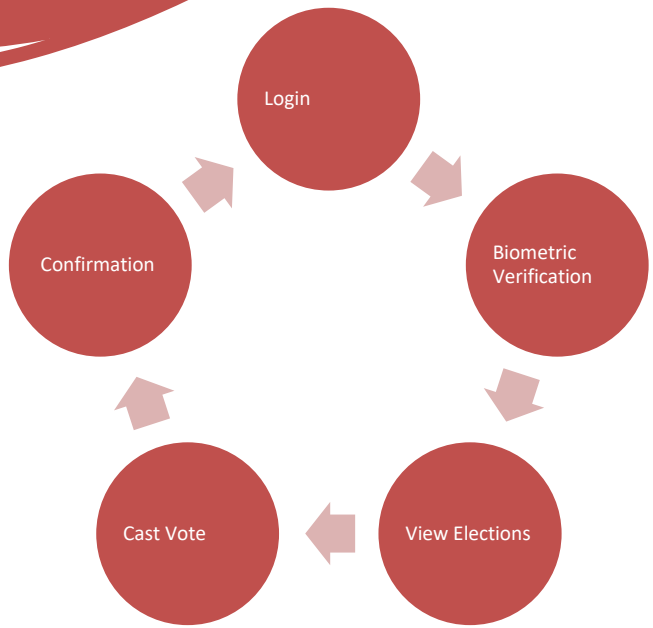
Prevents PIN-only logins

Public-key cryptography based

Election Management



Voting Flow



Login

Biometric Verification

View Elections

Cast Vote

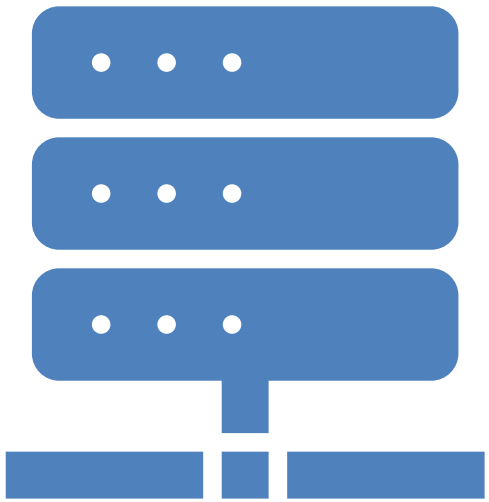
Confirmation

Auditor Features

- Real-time election observation
- Cannot modify data
- Advanced filtering
- Verify results & logs

Security Protocols

- HTTPS & JWT auth
- XSS, CSRF, SQLi protection
- Password hashing & biometric identity
- Audit trails encrypted

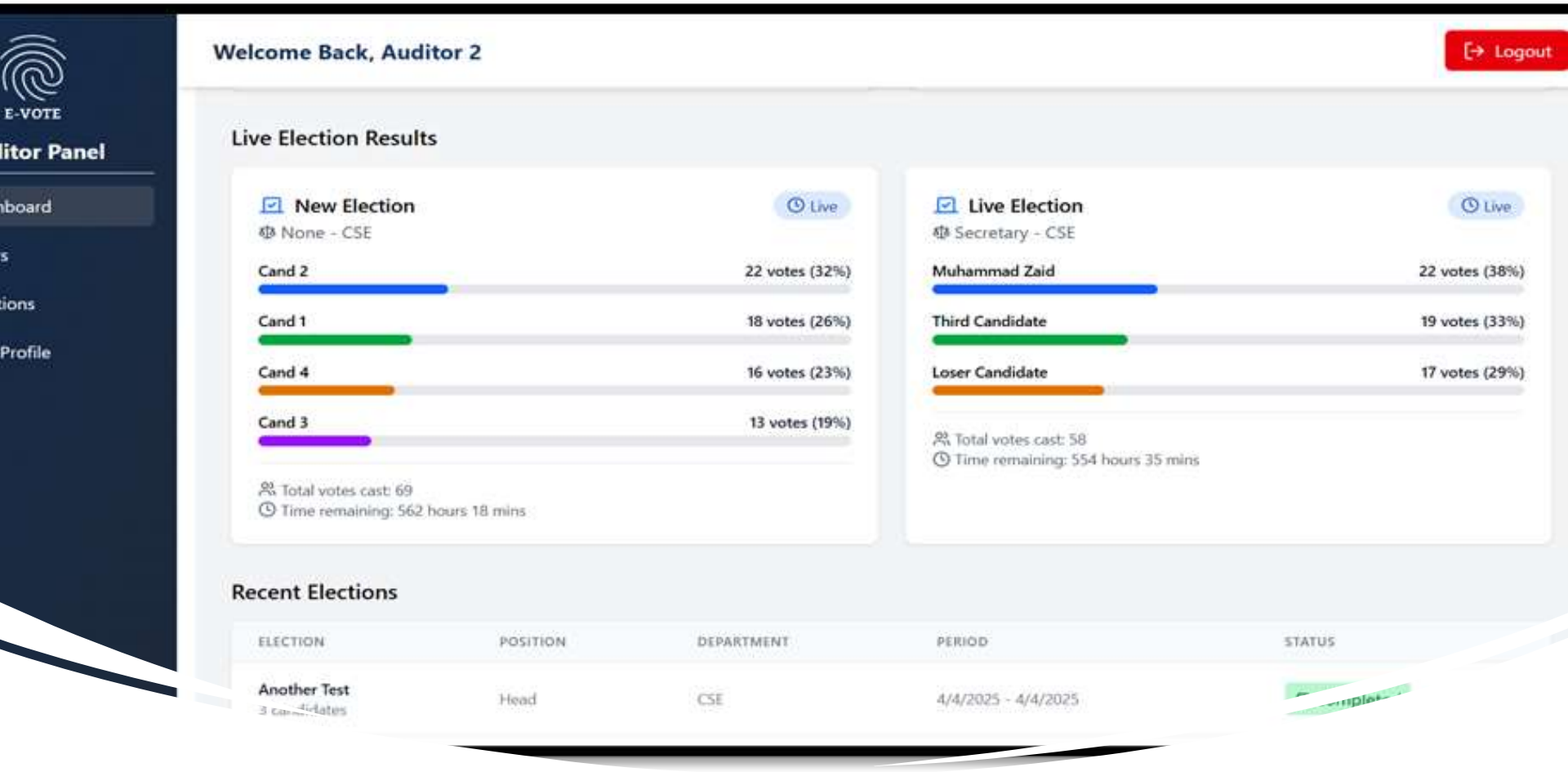


Testing Methodology

- Unit Testing (Jest)
- Integration Testing (API+DB)
- E2E Testing (Real scenario)
- Load Testing: 1000 users simulated

Performance Results

- 100 users: 320ms
- 500 users: 680ms
- 1000 users: 1.2s



Results Analytics Dashboard

- Live turnout per department
- Winning candidate graphs
- Raw vote breakdown

SDGs Covered

SDG 5: Gender Equality

SDG 9: Industry, Innovation & Infrastructure

SDG 16: Peace, Justice & Strong Institutions

Impact

75% ADMIN TIME REDUCED

BETTER PARTICIPATION

TRUSTED DIGITAL PROCESS

ENVIRONMENTALLY FRIENDLY

Future Work

MOBILE APP

BLOCKCHAIN INTEGRATION

PREDICTIVE ANALYTICS

AI-GENERATED REPORTS

Lessons Learned

Balancing security vs usability

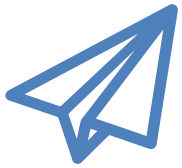
User-centered iterative design

Importance of documentation

Incremental rollouts helped refine

Thank You!

Visit: https://e-vote-x.vercel.app



Email: support@mcs.nust.edu.pk

Questions?