Project Proposal

VOTING SYSTEM

General Questions

- Full Name: Nupur Joshi
- Roll Number: B24EE1049
- Short Bio/Overview : I'm a first year electrical engineering student at IIT-Jodhpur. I started coding in the first semester of college. I've learnt C language through the college coursework and a few other online resources. I have a keen interest in coding and want to enhance my skills in practical applications of coding like front end development etc.
- Contacts :
  - Email: b24ee1049@iitj.ac.in
  - GitHub Username: nupur2006
  - WhatsApp no : 9604175807
- Why are you interested in working on this project?

  I wish to work on this project as it is a great opportunity to learn about the real-world applications in the areas of security and DevOps. The project's goal of converting a traditional voting system into an enterprise grade application offers numerous learning opportunities across different domains.

Project Understanding

➔ **What problem the project solves and who faces that problem?**

The project solves the problem of an inefficient, manual, and insecure voting system by transforming it into an enterprise-grade, scalable, and secure system. The main issues addressed include:

1. Scalability: The current system may not handle a large number of voters effectively.
2. Security: To ensure the integrity and confidentiality of votes.
3. User Experience: Improving the voting interface to be more responsive.
4. Automation: Automating the management of voter data.

These problems are faced by: Institutions, Government Entities, Election Organizers

➔ **What do you plan to do with the project (list of features, user flow)?**

Features:

➔ Scalability:
  - ◆ Distributed Architecture: Use Kubernetes and cloud services (e.g., GCP, AWS) to ensure the system can scale.

➔ Admin Dashboard:
  - ◆ Election Management: Admins can create, schedule, and manage elections, customizing the process based on different requirements.
  - ◆ Voter Data Management: Admins can automate and update voter data using external APIs.
  - ◆ Real-Time Monitoring: Monitor election results, active votes, and user participation in real-time.

➔ Enhanced User Interface :

    - ◆ Responsive Design: Redesign the voting page for better user experience on different devices (mobile, tablet, desktop).
    - ◆ Guided Voting Process: step-by-step instructions for users unfamiliar with online voting systems.
    - ◆ Show users the status of their votes.
- ➔ Security & Authentication Enhancements:
    - ◆ Implement secure JWT authentication with RSA encryption for secure voter login and vote submission.
    - ◆ Data Integrity: Ensure that all votes are securely transmitted and stored with encryption.
- ➔ Real-Time Notifications:
    - ◆ Notify users upon successful vote submission or if their vote status changes .
- ➔ Automated Voter Data Sync:
    - ◆ External API Integration: Automatically fetch and update voter data (e.g., student lists) from external sources like the institution's ERP system.

User Flow:

- ➔ Admin Flow:
    - ◆ Login: Admin logs in using secure credentials.
    - ◆ Election Setup: Admin creates a new election, customizes voting options, and configures the schedule.
    - ◆ Voter Data Sync: Admin coordinates with the institution's ERP or external data source to automatically update voter data.
    - ◆ Election Monitoring: Admin monitors real-time vote counts, checks voter participation, and reviews status reports.
    - ◆ Post-Election Reports: After voting ends, the system generates and sends reports with detailed analysis.
- ➔ Voter Flow:
    - ◆ Login: Voter logs in using secure credentials (via OTP etc.).
    - ◆ Election Access: Voter accesses the election they are eligible to vote in.
    - ◆ Voting Process: Voter selects options or candidates and submits their vote.
    - ◆ Confirmation: Voter receives a confirmation notification (via email/SMS) after submitting their vote.

    - Vote Status Check: Voters can check the status of their vote at any time.

- **What is your tentative timeline for the progress of the project?**
    - Week 1: Setup & Learning
- Set up project structure.
    - Week 2: Core Backend & Authentication
- Implement JWT authentication for user login and registration.
- Develop backend for user management and vote submission.
    - Week 3: Frontend & Core Features
- Develop UI components for voter/admin interfaces.
- Connect frontend to backend APIs.
- Implement email/SMS notifications for vote submission.
    - Week 4: Security
- Implement encryption and secure communication.
- Conduct initial security and penetration testing.
    - Week 5: Real-Time Features
- Integrate Kubernetes for scaling.
- Refine UI and test real-time notifications.
    - Week 6: Testing and Optimization
- Conduct end-to-end testing and security checks.
- Optimize performance.
- Complete documentation and prepare for final execution.

- **Any open source alternatives to the project?**

- Helios voting
- OpenElections
- ElectionGuard

Technical Understanding

- **Suggest some tech stack options which you think we can use in this project?**

  Frontend: React.js, Angular, Vue.js

  Backend: Node.js, Django, Flask

  Database: PostgreSQL, MySQL

- **Which components of the tech stack do you know about?**

I don't have any prior experience with the specific components of the tech stack for this project, but I am eager to learn. I'm motivated to put in the effort to understand how these technologies work. I'm looking forward to learning from the team and gaining hands-on experience throughout this project.

Time Commitment

How much time (in hours per week) will you be able to commit to this project? Also mention any time phases when you will have other commitments (for example, end sems) and cannot devote the same time compared to your regular schedule.

I will be able to commit 12-14 hrs per week. This time commitment may reduce during end sems and mid sems.