Proposal: Online Voting System

Submitted by:

2024-CS-40 Abdul Ahad

2024-CS-35 Saad Akhtar

Supervised by:

Mr. Nazeef-ul-haq

Course:

Data Structures and Algorithms

Department of Computer Science

University of Engineering and Technology

CSC 200 – Data Structure & Algorithms

Final Term Project Guidelines

Project Scope

Develop an Online Voting System that facilitates secure, efficient electronic voting with features such as voter and candidate registration, vote casting, vote tallying, and result declaration. The system will ensure secure authentication and maintain the integrity of votes through robust data structure implementations.

Key Features

- Voter and candidate registration
- Secure vote casting with authentication
- Sorting candidates by the number of votes
- Searching for voters and candidates efficiently
- Undo feature to handle accidental vote entries
- Managing voting requests in real time
- Organizing constituencies hierarchically
- Analyzing voting influence and coalitions

Data Structures to Use

- Hashing: To securely store and authenticate voters and candidates.
- Trees: To represent constituencies and districts in hierarchical form.
- Sorting Algorithms: To rank candidates by votes efficiently.
- Searching Algorithms: To quickly find voter or candidate details.
- Stacks: To implement undo/redo functionality for vote entries.
- Queues: To manage incoming voting requests orderly.
- Linked Lists: To maintain voter history records for auditability.
- Graphs: To model connections between candidates, parties, and coalitions.

System Mechanics

- Initialization: Register eligible users and candidates with all necessary attributes securely.
- Voting: Allow authenticated voters to cast votes; votes are queued and processed securely.
- Vote Tallying: Scores and votes are counted and sorted regularly to display current standings.
- Undo/Redo: Allow voters/admin to undo vote submissions within a time window.
- Coalition Modeling: Map alliances and influence between candidates using graph data structures.
- Result Declaration: Once voting ends, results are declared sorted by total votes.

Project Phases

- **Phase 1:** User and Candidate Design

  Design classes to represent voters and candidates with attributes like ID, name, status, and credentials. Implement secure hashing for authentication.

- **Phase 2:** Constituency and Coalition Structure

  Build constituency hierarchy using tree data structures; model candidate alliances using graphs.

- **Phase 3:** Voting and Request Management

  Implement the voting mechanism using queues to manage requests and linked lists to maintain history.

- **Phase 4:** Vote Processing and Sorting

  Incorporate vote counting, sorting, and searching algorithms to analyze poll results efficiently.
- **Phase 5:** Undo/Redo and User Interface

  Add undo/redo functions with stacks to ensure robustness. Develop a simple command-line/user interface for practical interaction.

Validation and Testing

- Ensure only authenticated users can vote.
- Confirm that vote undo/redo features work as expected.
- Verify correct vote tallying and sorting.
- Test constituency tree and candidate coalition graph models.
- Validate searching features for rapid retrieval.

Submission Requirements

- Code: Well-documented with clear comments explaining implementations.
- ReadMe: Instructions on running the system, dependencies, and features.
- Test Cases: Demonstrate correctness of voting, searching, undo/redo, and result computation.
- Report: A comprehensive PDF documenting the project scope, design, data structures used, and testing results.