A

SYNOPSIS REPORT

On

ONLINE VOTING SYSTEM

Submitted in partial fulfilment of the requirements of the degree
of
BACHELOR OF TECHNOLOGY

Submitted by
VIPUL KUMAR YADAV (2309005370092)
AKASH KUSHWAHA (2309005370011)
SATYAM KUMAR PAL (2309005370071)
ANKIT KUMAR YADAV (2309005370020)
AYUSH YADAV (2309005370033)

Submitted to
MR. SUBHODH SHARMA

COMPUTER SCIENCE & ENGINEERING
INSTITUTE OF ENGINEERING & TECHNOLOGY,
VIVEKANAND CAMPUS, DBRAU, AGRA
DECEMBER 2024

SYNOPSIS INDEX

1. Introduction

In the modern era, leveraging technology for conducting fair and transparent elections is increasingly vital. This project, Online Voting System in C++, simulates a digital platform for
election management. The system facilitates voter registration, voting, and the display of election results. It is designed to be interactive, user-friendly, and ensure voter authentication
and voting integrity.
This project caters to small-scale elections where security and transparency are priorities. The
system demonstrates fundamental programming concepts and software design methodologies
that can be further enhanced for large-scale applications.

2. Objectives

The primary objectives of this project are:

- To create an automated voting process that is user-friendly and efficient.
- To ensure secure and unique voter registration.
- To prevent double voting by maintaining a record of voters who have already cast their vote.
- To maintain transparency in election results by displaying vote counts for all candidates.
- To showcase the implementation of C++ programming concepts like data structures, algorithms, and modular programming

3. Features of the System

3.1 Voter Registration

• The system allows users to register their unique voter ID.
• Duplicate registrations are restricted.
• The list of registered voters is stored in a vector<string> for easy access.

3.2 Voting

• The system presents voters with a list of candidates and their IDs.
• Only registered voters can cast a vote, and the system ensures a voter ID is used only once for voting.
• The vote is recorded in the candidate's record.

3.3 Display Results

· After votes are cast, users can view election results.
• Results are displayed with each candidate's name and the number of votes they have received.

3.4 Menu-Driven Interaction

• A menu system allows users to navigate the functionalities, including:
1. Registering as a voter.
2. Casting a vote.
3. Viewing election results.
4. Exiting the system.

4. Software Design

4.1 Data Structures

The program uses the following key data structures:

- vector<string>:
  - Stores registered voter IDs and the IDs of voters who have already voted.
- vector<Candidate>:
  - Maintains the list of candidates along with their details (ID, name, and vote count).

4.2 Core Functions

The functionality of the system is divided into modular functions:

- registerVoter: Adds a new voter ID to the list of registered voters if it is unique.
- checkVoterEligibility: Validates if a voter ID exists in the registered voters list.
- vote: Allows a voter to cast a vote for a candidate and updates the vote count.
- displayResults: Displays the voting results for all candidates.

5. Implementation Details

5.1 Development Environment

The system is implemented in C++ and can be compiled using popular compilers like GCC or
Visual Studio. It leverages standard libraries like <vector>, <string>, <algorithm>, and <iostream> for efficient coding.

5.2 Code Overview

The program comprises:

- A main function for the menu-driven interaction.
- A Candidate structure to represent candidates.
- Function definitions for registration, voting, and results display

6.Data Flow Diagram (DFD)

Level 0: Context Diagram

•**External Entities:**

- Voter (Interacts with the system for registration and voting)
- System Administrator (Views election results)

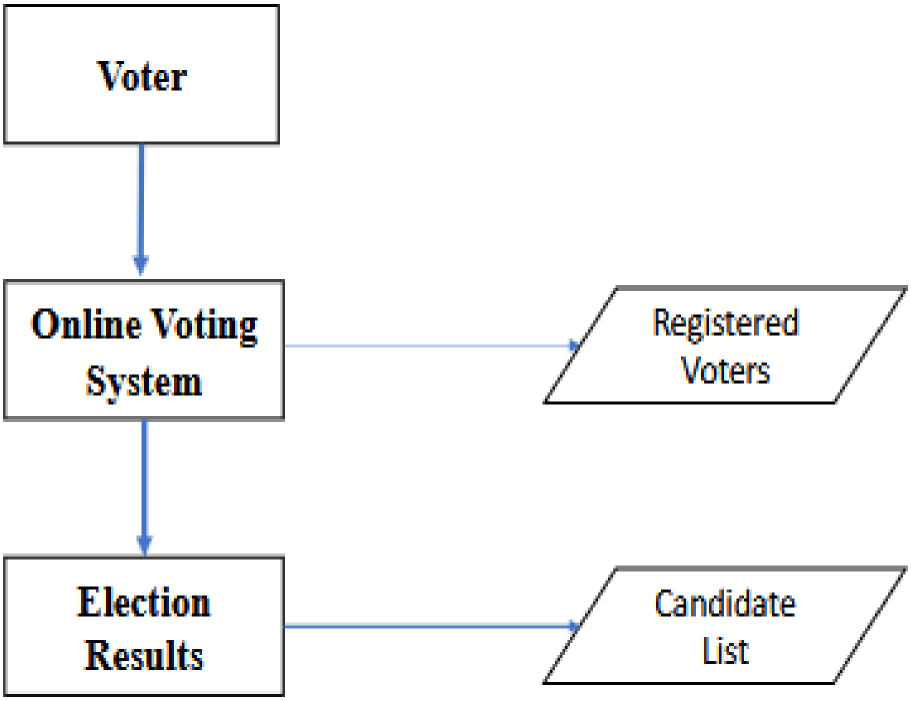
•**Processes**:

- Register Voter
- Cast Vote
- Display Results

Data Flow Steps:

•Voter provides Voter ID to the system.
•System registers the Voter ID into the database (or memory).
•Registered Voters cast their vote, and the system updates vote counts for candidates.
•System administrator or voters can request to view the election results.

Diagram Layout:



7. Flowchart

The flowchart represents the logical flow of the **Online Voting System**:

Steps:

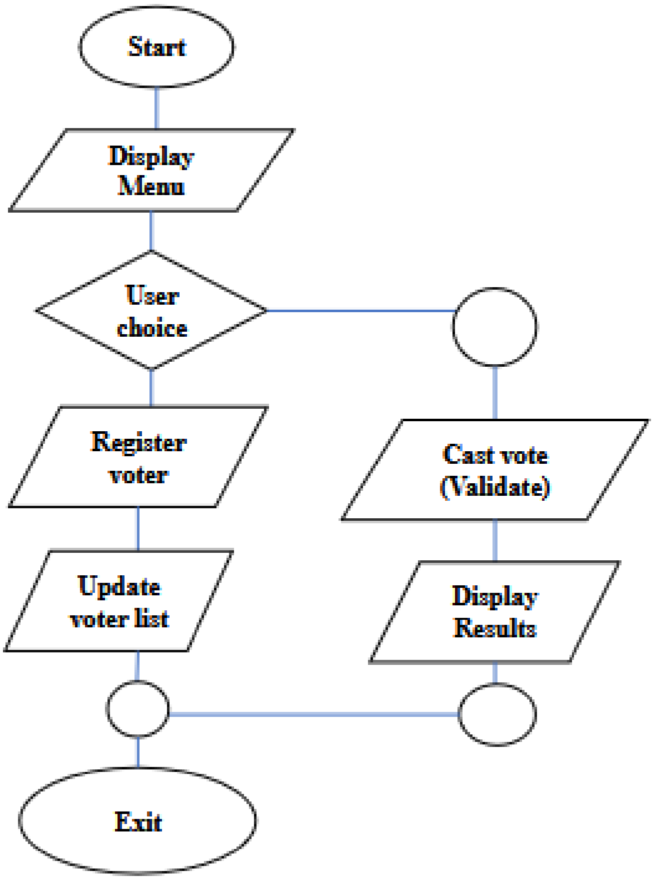
1.Start the program.

2.Display the menu:

1. **Option 1**: Register Voter
    1. Input: Voter ID
    2. Validate: Check if the Voter ID already exists.
    3. Output: Confirmation message.
2. **Option 2**: Cast Vote
    1. Input: Voter ID
    2. Validate: Check registration and voting status.
    3. Output: Record vote and update candidate's vote count.
3. **Option 3**: Display Results
    1. Output: Show all candidates and their vote counts.
4. **Option 4**: Exit
    1. Output: Exit message.

3.Repeat menu until user exits.

Flowchart Design:



8. Entity-Relationship (ER) Diagram

The **ER Diagram** depicts the relationships between entities in the system:

Entities and Attributes:

1.Voter:

- •Attributes: VoterID (PK)
- •Relationships: Can cast a vote (one-to-one relationship with Candidate through a voting event).

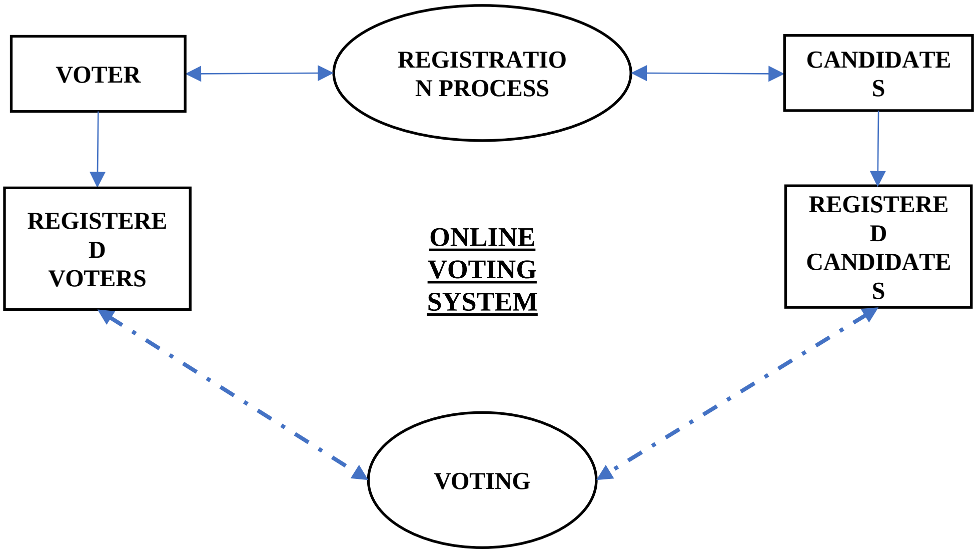
2.Candidate:

- •Attributes: CandidateID (PK), Name, VoteCount
- •Relationships: Receives votes (one-to-many relationship with Voters).

3.Voting Event:

- •Attributes: VoteID (PK), VoterID (FK), CandidateID (FK)
- •Relationships: Links Voter to Candidate.

Diagram Structure:



9. Conclusion

The Online Voting System in C++ demonstrates a basic yet functional model for conducting elections in a secure and transparent manner. By employing fundamental C++ concepts, the program lays a foundation for more complex systems that could be adopted in real-world scenarios. Future improvements, such as data persistence and enhanced security, can make the system robust enough for broader applications.

This project not only highlights the use of programming to solve real-world problems but also serves as an excellent educational tool for learning software development principles