C Programming Mini Project Documentation

Title Page

- **Project Title : Online Voting System**
- **Student Name : Gokilaa A**
- **Register Number : 921724102050**
- **Department : CSE**
- **College Name : SIT**
- **Course : Programming in C**
- **Guide / Trainer Name : Pugazh Mara**
- **Submission Date : 25.02.2026**

1.Abstract

The Online Voting System is a console-based application developed in C to automate the voting process, prevent duplicate voting, and provide accurate vote counts. It allows voters to vote using unique Voter IDs, stores votes in a file for permanent record, and provides secure admin authentication to view results. This project demonstrates the use of structured programming, arrays, functions, loops, conditional statements, and file handling while solving a practical real-world problem.

2.Introduction

The Online Voting System is designed to solve the problem of manual vote collection and counting, which can be time-consuming, error-prone, and vulnerable to duplicate voting. Traditional systems require supervision and effort to ensure fairness.

This project automates the process, ensuring that each voter votes only once, calculating vote percentages automatically, and storing results permanently in a file.

It is useful for schools, colleges, clubs, small organizations, and local events where a simple and secure voting method is required.

3. Objectives

- **Understand and apply structured programming in C.**
- **Implement loops and conditional statements to control program flow.**
- **Use arrays and functions for efficient data handling.**
- **Improve logical thinking and problem-solving skills.**
- **Learn file handling to store and retrieve voting data permanently.**
- **Understand basic security concepts such as preventing duplicate votes and admin authentication.**

4. Tools & Technology

- **Programming Language: C**
- **IDE / Compiler: Turbo C, Code::Blocks, Dev-C++, or GCC**
- **Operating System: Windows / Linux**

5. System Requirements

- **File Handling Functions: fopen(), fprintf(), fclose()**
- **Hardware Requirements: Basic computer system, 4GB RAM minimum, keyboard for input**

6. Methodology / Algorithm

Methodology

The program uses structured programming to divide the system into logical modules: voter authentication, vote casting, vote counting, and result display. Arrays are used to store voter IDs, preventing duplicate voting. File handling stores results permanently. Conditional statements and loops control the voting process.

Algorithm

1. **Start the program.**
2. **Initialize vote counters for all candidates to zero.**
3. **Input total number of voters.**
4. **For each voter:**

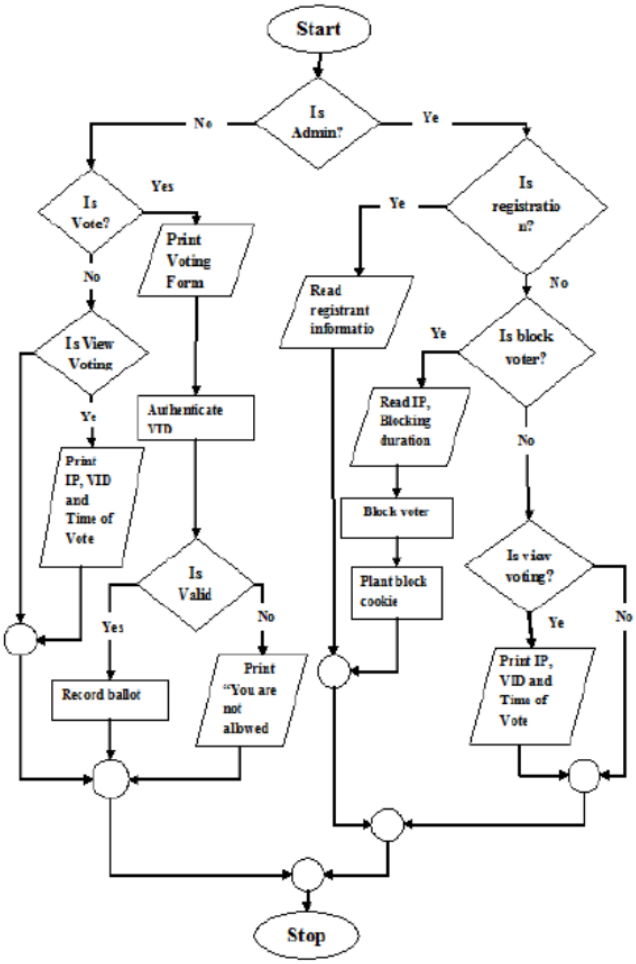
- **Input Voter ID.**
- **Check for duplicate Voter ID.**
- **If duplicate → deny voting and repeat.**
- **If unique → display candidates and accept vote.**
- **Update the corresponding candidate's vote count.**

5. **After voting is complete, store results in a file.**
6. **Ask for admin password:**

- **If correct → display total votes, percentage, and winner.**
- **If incorrect → display "Access Denied".**

7. **End program.**

7. Flowchart



8. Program Code

```
#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#define MAX 100

struct Voter {

   int voterID;

};
```

```
int main() {
   struct Voter voters[MAX];
   int votedIDs[MAX];
   int voteCount = 0;
   int vote1 = 0, vote2 = 0, vote3 = 0;
   int totalVoters, i, j;
   int choice, id;
   int password;
   int duplicate;
   FILE *fp;
   printf("Enter total number of voters: ");
   scanf("%d", &totalVoters);
    for(i = 0; i < totalVoters; i++) {
    printf("\n--- Voter %d ---\n", i+1);
    printf("Enter Voter ID: ");
    scanf("%d", &id);
    duplicate = 0;
      for(j = 0; j < voteCount; j++) {
         if(votedIDs[j] == id) {
            duplicate = 1;
```

```
            break;
        }
    }
if(duplicate) {
        printf("You have already voted! Access Denied.\n");
        i--;
        continue;
    }
  votedIDs[voteCount++] = id;
   printf("1. Candidate A\n");
    printf("2. Candidate B\n");
    printf("3. Candidate C\n");
    printf("Enter your vote (1-3): ");
    scanf("%d", &choice);
    switch(choice) {
        case 1: vote1++; break;
        case 2: vote2++; break;
        case 3: vote3++; break;
        default:
            printf("Invalid vote! Try again.\n");
```

```
            voteCount--;
            i--;
        }
    }
    fp = fopen("votes.txt", "w");
    fprintf(fp, "Candidate A: %d\n", vote1);
    fprintf(fp, "Candidate B: %d\n", vote2);
    fprintf(fp, "Candidate C: %d\n", vote3);
    fclose(fp);
    printf("\nVoting Completed Successfully!\n");
    printf("\nEnter Admin Password to view results: ");
    scanf("%d", &password);
    if(password == 1234) {
        float total = vote1 + vote2 + vote3;
        printf("\n--- Voting Results ---\n");
        printf("Candidate A: %d votes (%.2f%%)\n", vote1, (vote1*100)/total);
        printf("Candidate B: %d votes (%.2f%%)\n", vote2, (vote2*100)/total);
        printf("Candidate C: %d votes (%.2f%%)\n", vote3, (vote3*100)/total);
        if(vote1 > vote2 && vote1 > vote3)
            printf("Winner: Candidate A\n");
```

```
        else if(vote2 > vote1 && vote2 > vote3)
            printf("Winner: Candidate B\n");
        else if(vote3 > vote1 && vote3 > vote2)
            printf("Winner: Candidate C\n");
        else
            printf("It's a Tie!\n");
    } else {
        printf("Wrong Password! Access Denied.\n");
    }
    return 0;
}
```

9. Sample Input / Output

Sample Input:

Enter total number of voters: 3

--- Voter 1 ---

Enter Voter ID: 101

Enter your vote (1-3): 1

--- Voter 2 ---

Enter Voter ID: 102

Enter your vote (1-3): 2

--- Voter 3 ---

Enter Voter ID: 101

You have already voted! Access Denied.

--- Voter 3 ---

Enter Voter ID: 103

Enter your vote (1-3): 3

Enter Admin Password to view results: 1234

Sample Output:

--- Voting Results ---

Candidate A: 1 votes (33.33%)

Candidate B: 1 votes (33.33%)

Candidate C: 1 votes (33.33%)

It's a Tie!

10. Result

- **The program executed successfully and produced the expected output.**

11. Applications

- **Educational Institutions: Student council and class representative elections.**
- **Organizations / Clubs: Committee elections and event decisions.**
- **Surveys and Polls: Collect opinions or feedback efficiently.**
- **Community Voting: Local committee elections.**
- **Learning Tool: Helps beginners understand arrays, functions, file handling, and structured programming in C.**

12. Conclusion

The Online Voting System automates the voting process, ensuring accuracy, fairness, and security. It prevents duplicate voting, calculates percentages automatically, and stores results permanently. The project demonstrates structured programming concepts in C and improves problem-solving skills while providing a practical solution for small-scale voting scenarios.

13. Future Enhancements

- **GUI-based version for better user experience.**
- **Database integration (MySQL / SQLite) for scalable storage.**
- **Multi-candidate and multi-category elections.**
- **Enhanced security with encryption and hashed passwords.**
- **Online/web version using client-server architecture.**
- **Real-time vote visualization with bar charts or graphs.**
- **Automated report generation in PDF/CSV format.**

14. References

1. **Programming in C – E. Balagurusamy, 7th Edition, McGraw-Hill Education, 2017.**
2. **Let Us C – Yashavant Kanetkar, 16th Edition, BPB Publications, 2018.**
3. **TutorialsPoint, "C Programming Tutorial," https://www.tutorialspoint.com/cprogramming/**
4. **GeeksforGeeks, "C Programming Examples and Projects," https://www.geeksforgeeks.org/c-programming-examples/**
5. **Stack Overflow, "C Programming File Handling & Voting System Discussions," https://stackoverflow.com/**
6. **W3Schools, "C Structures and File Handling," https://www.w3schools.in/c-tutorial/**