PROJECT REPORT

On

"*Voting System in 'C' Language* "

Submitted By

Mansi S. Ganvir

(**307**)

Guided By:
Roshani Talmale

DEPARTMENT OF FIRST YEAR ENGINEERING

S. B. JAIN INSTITUTE OF TECHNOLOGY MANAGEMENT AND RESEARCH, NAGPUR.

2023-2024

CONTENTS

1. Introduction (Project Details) 01

2. Major Library and Functions 02

3. Source Code (Program) 06

4. Result (Output) 08

5. Conclusion 09

References

Introduction (Project Details):

The "Secure Voting System in C" project endeavors to create a reliable and efficient electronic voting system using the C programming language. In an era where digital solutions are increasingly prevalent, the need for a secure and transparent voting mechanism is paramount. This project aims to address the shortcomings of traditional paper-based voting systems by leveraging the power and versatility of C to develop a robust and secure platform for conducting elections.

- It Develop a user-friendly command-line interface (CLI) for voters to cast their votes securely.

- Implement encryption and authentication mechanisms to safeguard the integrity and confidentiality of the voting process.

- Design data structures and algorithms to efficiently store and process vote data.

- Ensure the fairness and transparency of the voting system by providing audit trails and verification mechanisms.

- Incorporate error handling and fault tolerance to mitigate potential risks and ensure the reliability of the system.

- Optimize performance to handle large-scale elections with a significant number of voters and candidates.

-It Provide documentation and user guides to facilitate the adoption and understanding of the voting system.

Major Library and Functions:

For a voting system project in C, you may require various libraries and functions to handle different aspects of the system. Here are some major libraries and functions that could be useful:

1. *Standard Input/Output Library (stdio.h)*:

- Functions like printf() and scanf() for user input and output.

2. *String Handling Library (string.h)*:

- Functions like strcpy(), strcat(), strcmp() for handling strings, such as candidate names or user inputs.

3. *File Handling Library (stdio.h)*:

- Functions like fopen(), fclose(), fwrite(), fread() for reading from and writing to files, which could be used for storing candidate information or voting records.

4. *Dynamic Memory Allocation (stdlib.h)*:

- Functions like malloc(), calloc(), realloc(), free() for dynamic memory allocation, which could be useful for managing candidate lists or other data structures dynamically.

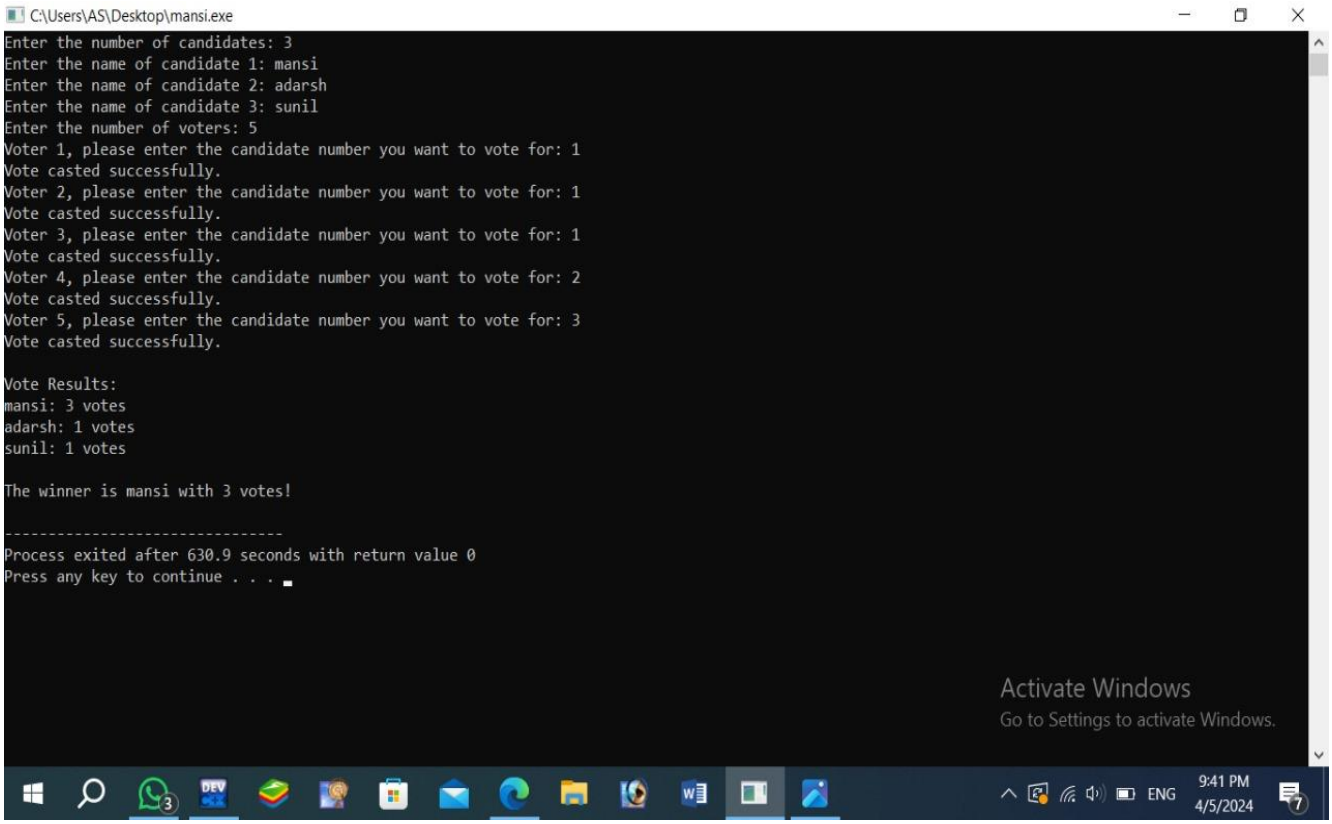
These are some of the major libraries and functions that you might use in a voting system project in C. The specific libraries and functions you choose will depend on the requirements and design of your project.

Source Code (Program):

```
#include <stdio.h>

#define MAX_CANDIDATES 10

int main() {
    int numCandidates;
    =char candidates[MAX_CANDIDATES][50];
 // Assuming each candidate name has a maximum of 50 characters
    int votes[MAX_CANDIDATES] = {0};
    int numVoters;
    int i, j;

    printf("Enter the number of candidates: ");
    scanf("%d", &numCandidates);
```

```
    // Input candidate names
    for (i = 0; i < numCandidates; i++) {
        printf("Enter the name of candidate %d: ", i + 1);
        scanf("%s", candidates[i]);
    }

    printf("Enter the number of voters: ");
    scanf("%d", &numVoters);

    // Vote casting
    for (i = 0; i < numVoters; i++) {
        printf("Voter %d, please enter the candidate number you want to vote for: ", i + 1);
        scanf("%d", &j);

        // Check if the candidate number is valid
        if (j >= 1 && j <= numCandidates) {
            votes[j - 1]++;
            printf("Vote casted successfully.\n");
        } else {
            printf("Invalid candidate number. Vote not counted.\n");
        }
    }

    // Display results
    printf("\nVote Results:\n");
    for (i = 0; i < numCandidates; i++) {
        printf("%s: %d votes\n", candidates[i], votes[i]);
    }

    // Find the winner
    int maxVotes = 0;
    int winnerIndex = -1;
    for (i = 0; i < numCandidates; i++) {
        if (votes[i] > maxVotes) {
            maxVotes = votes[i];
            winnerIndex = i;
        }
    }

    if (winnerIndex != -1) {
        printf("\nThe winner is %s with %d votes!\n", candidates[winnerIndex], maxVotes);
    } else {
        printf("\nNo winner. It's a tie!\n");
    }

    return 0;
}
```

Result & Discussion (Output):



Conclusion:

In conclusion, implementing a voting system project in C can provide a solid foundation for managing elections efficiently. Through proper design and implementation, such a system can handle various voting methods, ensure accuracy, security, and fairness in the electoral process. However, it's crucial to thoroughly test the system to identify and address any potential vulnerabilities or bugs that could compromise its integrity. Additionally, incorporating user-friendly interfaces and robust authentication mechanisms can enhance the usability and trustworthiness of the voting system. Overall, with careful planning and execution, a voting system project in C can contribute to the democratic process by facilitating transparent and reliable elections.

References:

1. www.wikipedia.org/
2. www.Google.com
3. www.beginnersbook.com
4. https://youtube.com