Subject: C Programming

FINAL PROJECT

BTech CSE 1st Year

Submitted by: Ayushi (590028429) & Harshit (590028439)
Batch: 68
Topic: Online Voting System
Date of submission: 30/11/25

Faculty:

Vinod Kumar

Abstract

This project implements a fully functional Online Voting System using the C programming language. The system is designed to digitally manage and streamline the voting process, ensuring accuracy, security, and convenience for eligible voters.

The system supports essential functionalities such as voter registration, login authentication, vote casting, and vote tracking. Users can register by providing a unique voter ID and password, ensuring no duplicate registrations occur. The system also verifies age eligibility before allowing access, restricting voting to users aged 18 and above. Once logged in, voters can view a list of political parties and cast their votes. The system prevents multiple voting attempts and allows users to verify whom they have voted for .From a programming perspective this project demonstrates core concepts such as:

- *Array manipulation* to store voter details and track votes
- *Structured programming with functions* to organize registration, login, and voting processes
- *Conditional statements* to validate user inputs and control system flow
- *Loops t*o handle repeated interactions and menu navigation
- *String operations* for handling voter IDs, passwords, and party names
- *Logical checks* to enforce voting eligibility and prevent multiple votes

Problem Definition

In traditional voting processes, verifying voter identity, preventing duplicate votes, and managing voter records manually can be slow, inaccurate, and difficult to maintain. Issues such as incorrect entries, repeated voting, and inefficient record handling often occur in offline systems.

To overcome these problems, a simple and secure online voting system is required. The system must ensure that only eligible users (18+ age) can participate, each voter can register with a unique identity, and voting is allowed only after successful authentication. Moreover, the system should store voting information accurately and prevent multiple votes by the same user.

This project aims to develop a computerized system that provides digital voter registration, secure login, party selection, and one-vote-per-user verification, ensuring a smooth and reliable voting experience. Therefore, the following key requirements were identified:

System Requirements:

1. Age Verification

- The system must check the user's age and allow only users aged 18 or above to continue.

2. Voter Registration

- User must register using a unique voter ID and password.
- The system must check for duplicate voter IDs before registration.
- Voter details must be stored (ID, password, vote status, voted party).

3. Login System

- Only registered voters can log in using the correct voter ID and password.
- Incorrect login details should show an error message.

4. Voting Process

- After successful login, the system must display a list of parties:
  - BJP
  - Congress
  - AAP
  - CPI
  - Independent Candidate
- The voter can select one party to cast their vote.
- The system must prevent users from voting more than once.

5. Vote Tracking

- The system must store:
  - Whether the user has voted
  - Which party they voted for

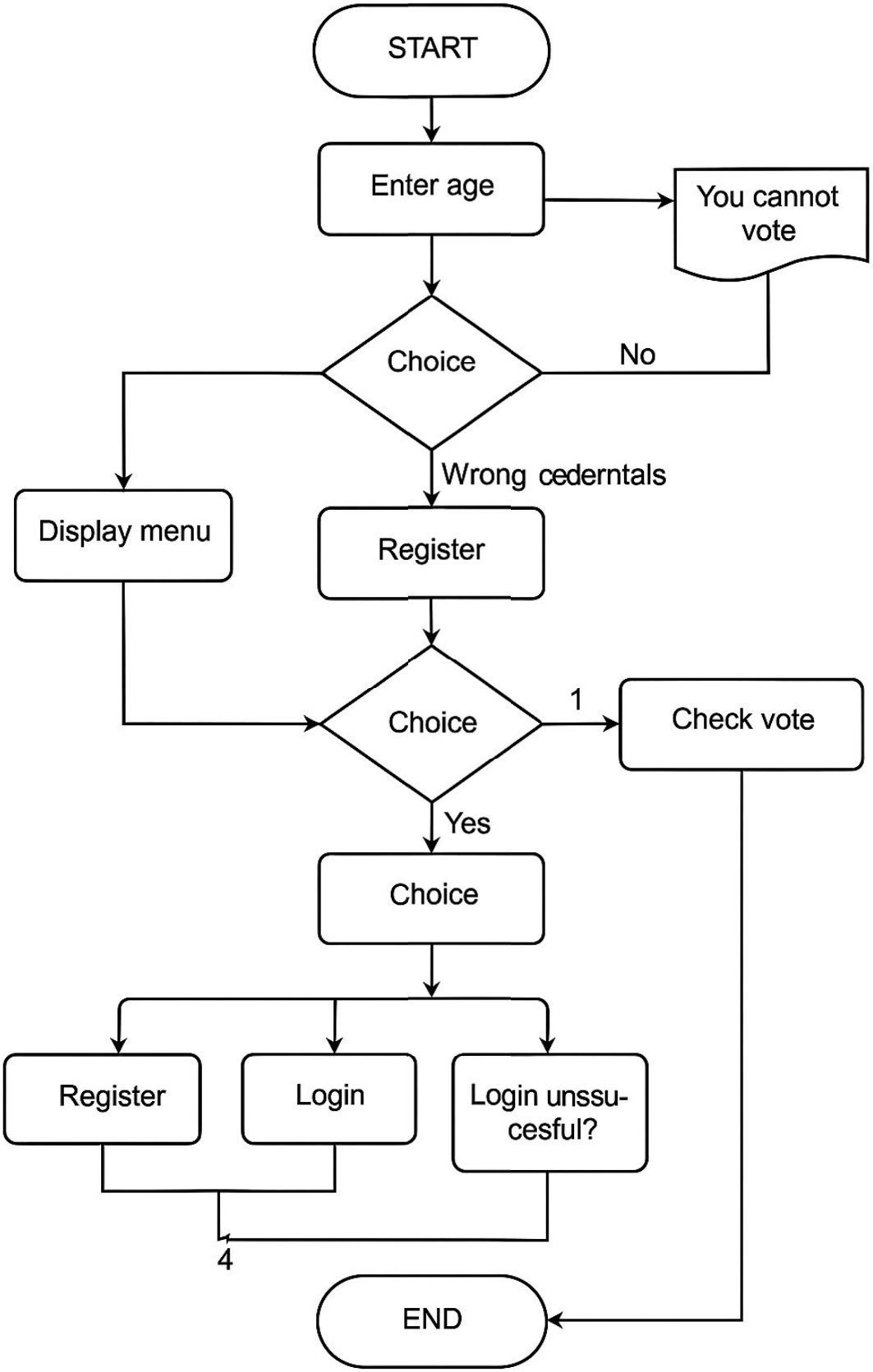
6. Check Vote Option

- The system must allow the user to check whom they voted for, only after login.

7. Data Storage

- The system must store all voter data using:
  - Structures
  - Arrays

Flowchart


START
Enter age
You cannot vote
Choice
No
Wrong cederntals
Display menu
Register
Choice
1
Check vote
Yes
Choice
Register
Login
Login unssu-cesful?
4
END

Algorithm

Step 1: Ask the user for age

. Accept age using formatted input (scanf) .

- If age < 18 → display message "You cannot vote" and terminate program.
- Else → proceed to main menu.

Step 2: Display main menu in a loop

- Options:
    1. Register
    2. Login & Cast Vote
    3. Check Whom You Voted For
    4. Exit

Step 3: Registration (if user chooses 1)

- Accept Voter ID and Password.
- Check if Voter ID already exists using voterIDExists() function.
    - If exists → display "Voter ID already exists. Registration failed."
    - Else → store Voter ID, password, set hasVoted = 0 and votedFor as empty.
- Display "Registration Successful".

Step 4: Login & Cast Vote (if user chooses 2)

- Accept Voter ID and Password.
- Check credentials against registered voters.
    - If invalid → display “Wrong Voter ID or Password.”
    - If valid → display “Login Successful” and proceed.
- Check if user has already voted (hasVoted).
    - If yes → display whom user voted for.
    - If no → display list of parties and accept vote choice.
        - Map choice to party name and update votedFor.
        - Set hasVoted = 1.
        - Display confirmation of successful vote.

Step 5: Check Whom You Voted For (if user chooses 3)

- If user is not logged in → display “You must login first.”
- Else →
    - If user has voted → display party name voted for.
    - If user has not voted → display “You have not voted yet.”

Step 6: Exit (if user chooses 4)

- Display “Thank you for using the Online Voting System.”
- Terminate the program.

Step 7: Loop back to main menu after completing any option except Exit.

Implementation Details

1. Age Verification

To ensure that only eligible voters (18 years or above) can access the system.

```
printf("Enter your age: ");
scanf("%d", &age);

if (age < 18) {
    printf("❌ You are under 18. You cannot vote.\n");
    return 0;
}
```

2. Main Menu

Provides the user with options to interact with the voting system.

```
while (1) {
    printf("\n---- ONLINE VOTING SYSTEM ----\n");
    printf("1. Register\n");
    printf("2. Login & Cast Vote\n");
    printf("3. Check Whom You Voted For\n");
    printf("4. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);
    ...
}
```

3. Voter Registration

- Allows a new user to register as a voter.

```
void registerVoter() {
    ...
    if (voterIDExists(id)) {
        printf("❌ This Voter ID already exists. Registration failed.\n");
        return;
    }
    strcpy(voters[voterCount].voterID, id);
    strcpy(voters[voterCount].password, pass);
    voters[voterCount].hasVoted = 0;
    strcpy(voters[voterCount].votedFor, "");
    voterCount++;
    printf("✓ Registration Successful!\n");
}
```

4. Login

- To authenticate a registered voter before casting a vote.

```
int login() {
    ...
    for (int i = 0; i < voterCount; i++) {
        if (strcmp(voters[i].voterID, id) == 0 &&
            strcmp(voters[i].password, pass) == 0) {
            printf("✓ Login Successful!\n");
            return i;
        }
    }
    printf("❌ Wrong Voter ID or Password.\n");
    return -1;
}
```

5. Casting Vote

- Allows a logged-in voter to cast their vote.
- Prevents multiple voting by the same user.

```
void castVote(int userIndex) {
    if (voters[userIndex].hasVoted) {
        printf("⚠ You have already voted! You voted for: %s\n", voters[userIndex].votedFor);
        return;
    }
    ...
    voters[userIndex].hasVoted = 1;
    printf("✓ You have successfully voted for: %s\n", voters[userIndex].votedFor);
}
```

6. Checking Voted Candidate

- Let a voter check whom they have voted for.

```
void checkYourVote(int userIndex) {
    if (userIndex == -1) {
        printf("❌ You must login first.\n");
        return;
    }
    if (voters[userIndex].hasVoted)
        printf("✓ You voted for: %s\n", voters[userIndex].votedFor);
    else
        printf("⚠ You have not voted yet.\n");
}
```

7. Exit option

Allows the user to exit the system gracefully.

```
case 4:
    printf("Thank you for using the Online Voting System.\n");
    return 0;
```

Testing & Result

Test Case 1: Underage Voter

Input: Age = 16

Expected Output:

```
Enter your age: 16
❌ You are under 18. You cannot vote.
```

Result: Successful

Test Case 2: Register New Voter

Input: Age = 20, Voter ID = "V001", Password = "pass123"

Expected Output:

```
Enter your age: 20

---- ONLINE VOTING SYSTEM ----
1. Register
2. Login & Cast Vote
3. Check Whom You Voted For
4. Exit
Enter your choice: 1

---- VOTER REGISTRATION ----
Enter Voter ID Number: V001
Enter Password: pass123
✓ Registration Successful!
```

Result: successful

Test Case 3: Duplicate Registration

Input: Age = 22, Voter ID = "V001", Password = "anypass"

Expected Output:

```
Enter your age: 22

---- ONLINE VOTING SYSTEM ----
1. Register
2. Login & Cast Vote
3. Check Whom You Voted For
4. Exit
Enter your choice: 1

---- VOTER REGISTRATION ----
Enter Voter ID Number: V001
❌ This Voter ID already exists. Registration failed.
```

Result: Successful

Test Case 4: Login with Correct Credentials

Input: Voter ID = "V001", Password = "pass123"

Expected Output:

```
---- LOGIN ----
Enter Voter ID: V001
Enter Password: pass123
✓ Login Successful!
```

Result: successful

Test Case 5: Login with Wrong Credentials

Input: Voter ID = "V001", Password = "wrongpass"

Expected Output:

```
---- LOGIN ----
Enter Voter ID: V001
Enter Password: wrongpass
❌ Wrong Voter ID or Password.
```

Result: Successful

Test Case 6: Cast Vote Successfully

Input: Logged in as V001, Choice = 2 (Congress)

Expected Output:

```
---- CAST YOUR VOTE ----
1. BJP
2. Congress
3. AAP
4. CPI
5. Independent Candidate
Enter your choice: 2
✓ You have successfully voted for: Congress
```

Result: Successful

Test Case 7: Attempt to Vote Again

Input: Logged in as V001, Choice = 1 (BJP)

Expected Output:

```
---- CAST YOUR VOTE ----
1. BJP
2. Congress
3. AAP
4. CPI
5. Independent Candidate
Enter your choice: 1
⚠ You have already voted! You voted for: Congress
```

Result: Successful

Test Case 8: Check Vote Status

Input: Logged in as V001

Expected Output:

```
✓ You voted for: Congress
```

Result: Successful

Test Case 9: Check Vote Without Login

Input: Not logged in

Expected Output:

```
✗ You must login first.
```

Result: Successful

Problems Faced By Our Group

1. Handling Duplicate Voter IDs

- Problem: Ensuring that no two users can register with same Voter ID.
- Solution: Implemented the voterIDExists() function to check for duplicates before adding a new voter.

2. Preventing Multiple Votes

- Problem: Users could try to cast their vote multiple times.
- Solution: Added a hasVoted flag in the Voter structure. The castVote() function checks this flag and prevents voting if the user has already voted.

3. Secure Login

- Problem: Ensuring that only registered voters with correct credentials can log in.
- Solution: Implemented login validation in the login() function by checking both Voter ID and Password against the registered voters array.

4. Invalid Menu or Voting Choices

- Problem: Users might enter invalid options in menus or vote selection.
- Solution: Added input validation and displayed proper error messages when the user selects invalid options.

Conclusion

The **Online Voting System** project successfully demonstrates the implementation of a secure, reliable, and user-friendly digital platform for voting. The primary objective of the system was to allow eligible voters to register, log in, and cast their votes in a way that ensures fairness and prevents fraudulent activities. Through this program, we were able to incorporate essential features such as age verification to restrict underage users, duplicate voter ID checks to maintain the uniqueness of registrations, and vote tracking to prevent multiple voting attempts.

It has also reinforced concepts related to C programming, including structures, arrays, string manipulation, and input/output operations. By successfully creating a working online voting system, we have demonstrated how programming can solve real-world problems and contribute to efficient digital solutions for processes such as elections.

Future Work

1. Database Integration

- Currently, voter data is stored in arrays, which limits scalability.
- Future versions can integrate a database system (like MySQL or SQLite) to store voter information securely and handle large numbers of users efficiently.

2. Password Encryption

- Passwords are currently stored in plain text, which poses security risks.
- Implementing encryption techniques (e.g., hashing) can protect voter credentials from unauthorized access.

3. Forgot Password / Account Recovery

- Adding a feature for users to recover or reset forgotten passwords would enhance user convenience and accessibility.

4. Real-Time Vote Counting and Results

- Implementing a module for live vote tallying and displaying election results securely can make the system more practical for real-world usage

Snip of code

```c
#include <stdio.h>
#include <string.h>

#define MAX_USERS 100
#define MAX_VOTES 100

// Structure to store voter details
struct Voter {
    char voterID[20];
    char password[20];
    int hasVoted;
    char votedFor[50];
};

// Global array to store all registered voters
struct Voter voters[MAX_USERS];
int voterCount = 0;

// Function to check if voter ID already exists (duplicate check)
int voterIDExists(char id[]) {
    for (int i = 0; i < voterCount; i++) {
        if (strcmp(voters[i].voterID, id) == 0)
            return 1; // found duplicate
    }
    return 0;
}
```

```c
// Function for login
int login() {
    char id[20], pass[20];

    printf("\n---- LOGIN ----\n");
    printf("Enter Voter ID: ");
    scanf("%s", id);
    printf("Enter Password: ");
    scanf("%s", pass);

    for (int i = 0; i < voterCount; i++) {
        if (strcmp(voters[i].voterID, id) == 0 &&
            strcmp(voters[i].password, pass) == 0) {
            printf("✓ Login Successful!\n");
            return i;
        }
    }

    printf(" Wrong Voter ID or Password.\n");
    return -1;
}
```

```c
// Function for login
int login() {
    char id[20], pass[20];

    printf("\n---- LOGIN ----\n");
    printf("Enter Voter ID: ");
    scanf("%s", id);
    printf("Enter Password: ");
    scanf("%s", pass);

    for (int i = 0; i < voterCount; i++) {
        if (strcmp(voters[i].voterID, id) == 0 &&
            strcmp(voters[i].password, pass) == 0) {
            printf("✓ Login Successful!\n");
            return i;
        }
    }

    printf(" Wrong Voter ID or Password.\n");
    return -1;
}
```

```c
// Function to show party list + vote
void castVote(int userIndex) {
    int choice;

    if (voters[userIndex].hasVoted) {
        printf(" You have already voted! You voted for: %s\n", voters[userIndex].votedFor);
        return;
    }

    printf("\n---- CAST YOUR VOTE ----\n");
    printf("1. BJP\n");
    printf("2. Congress\n");
    printf("3. AAP\n");
    printf("4. CPI\n");
    printf("5. Independent Candidate\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);

    switch (choice) {
        case 1: strcpy(voters[userIndex].votedFor, "BJP"); break;
        case 2: strcpy(voters[userIndex].votedFor, "Congress"); break;
        case 3: strcpy(voters[userIndex].votedFor, "AAP"); break;
        case 4: strcpy(voters[userIndex].votedFor, "CPI"); break;
        case 5: strcpy(voters[userIndex].votedFor, "Independent"); break;
        default:
            printf("Invalid choice.\n");
            return;
    }

    voters[userIndex].hasVoted = 1;
    printf("✓ You have successfully voted for: %s\n", voters[userIndex].votedFor);
}
```

```c
// Function to check whom user voted for
void checkYourVote(int userIndex) {
    if (userIndex == -1) {
        printf(" You must login first.\n");
        return;
    }

    if (voters[userIndex].hasVoted)
        printf("✓ You voted for: %s\n", voters[userIndex].votedFor);
    else
        printf(" You have not voted yet.\n");
}

int main() {
    int age;
    int choice;
    int loggedInUser = -1;
```

```c
// Step 1: Ask AGE first
printf("Enter your age: ");
scanf("%d", &age);

if (age < 18) {
    printf(" You are under 18. You cannot vote.\n");
    return 0;
}
```

```c
    // Age 18+ → Show Menu
    while (1) {
        printf("\n---- ONLINE VOTING SYSTEM ----\n");
        printf("1. Register\n");
        printf("2. Login & Cast Vote\n");
        printf("3. Check Whom You Voted For\n");
        printf("4. Exit\n");
        printf("Enter your choice: ");
        scanf("%d", &choice);

        switch (choice) {
            case 1:
                registerVoter();
                break;

            case 2:
                loggedInUser = login();
                if (loggedInUser != -1)
                    castVote(loggedInUser);
                break;

            case 3:
                checkYourVote(loggedInUser);
                break;

            case 4:
                printf("Thank you for using the Online Voting System.\n");
                return 0;

            default:
                printf("Invalid choice. Try again.\n");
        }
    }

    return 0;
}
```

Snip of Output

```
Enter your age: 19

---- ONLINE VOTING SYSTEM ----
1. Register
2. Login & Cast Vote
3. Check Whom You Voted For
4. Exit
Enter your choice: 1

---- VOTER REGISTRATION ----
Enter Voter ID Number: XCV897
Enter Password: 234
? Registration Successful!
```

```
---- ONLINE VOTING SYSTEM ----
1. Register
2. Login & Cast Vote
3. Check Whom You Voted For
4. Exit
Enter your choice: 2

---- LOGIN ----
Enter Voter ID: XCV897
Enter Password: 234
? Login Successful!

---- CAST YOUR VOTE ----
1. BJP
2. Congress
3. AAP
4. CPI
5. Independent Candidate
Enter your choice: 1
? You have successfully voted f
```

```
---- ONLINE VOTING SYSTEM ----
1. Register
2. Login & Cast Vote
3. Check Whom You Voted For
4. Exit
Enter your choice: 3
? You voted for: BJP
```

```
---- ONLINE VOTING SYSTEM ----
1. Register
2. Login & Cast Vote
3. Check Whom You Voted For
4. Exit
Enter your choice: 4
Thank you for using the Online Voting System.

--------------------------------
Process exited after 50.65 seconds with return
```

References

Balagurusamy, E. *Programming in ANSI C*, Tata McGraw-Hill Education.

GeeksforGeeks. "C Programming Language Tutorials." https://www.geeksforgeeks.org/c-programming-language/

College Lecture Notes on Programming in C, School of Computer Science.

Kernighan, B. W., & Ritchie, D. M. *The C Programming Language*, 2nd Edition, Prentice Hall, 1988.

Tutorials Point. "C Programming – Basics and Advanced Concepts." https://www.tutorialspoint.com/cprogramming/index.htm

THANK

YOU