1. Aim

The aim of this micro project is to develop a simplified Election Management System (EMS) using C++. This system will streamline key election processes, including voter registration, candidate management, and vote counting.

One key part of the project is the voter registration system. It will ensure that each voter is unique and include checks like age and ID verification. Voters will also be able to check if they are registered, making the process clear and accessible.

The project will also create a simple interface for voting. It will make sure that each voter can only vote once, which is important for fairness. Counting the votes accurately is essential, and the system will include ways to tally the votes for each candidate, showing results either during or after the election.

2. Proposed Methodology

1. **Class Structure**
   Election Class: Contains attributes for voter details (ID, name, age, contact, city).
   Tracks if a voter has already voted.
   Voter Class: Manages an array of Election objects.
   Contains methods for voter registration, displaying details, and casting votes.
2. **Data Initialization**
   Initialize the total count of registered voters to zero in the Voter class constructor.
3. **Registration Process**
   Implement fill_registration_form():Prompt for the number of voters registering.
   Collect details (age, ID, name, contact, city) for each voter.
   Validate eligibility (age >= 18) before registration.
4. **Display Voter Details**
   Implement display_voter_details(): Show all registered voters' details.
   Provide feedback if no voters are registered.
5. **Voting Process**
   Implement cast_vote(): Prompt for voter ID to identify the voter.
   Check if the voter has already voted.
   Display a list of political parties for selection.
   Update vote counts based on user selection and mark the voter as having voted.
6. **User Interaction**
   Display options (register, view details, vote, exit) in a loop.
   Use numeric input for user choices.
7. **Feedback and Output**
   Provide clear feedback messages for successful actions (registration, voting).
   Display voting results after each vote cast.

8. Exit Strategy

Allow users to exit the system gracefully while thanking them for participation.

3. Action Plan

Sr. No.	Detail of activity	Plan Start Date	Plan Finish Date
1.	Collect information on existing election management systems and best practices.	14/10/2024	15/10/2024
2.	Implement the code for the election management system, including classes and methods.	15/10/2024	18/10/2024
3.	Create a report about the system.	19/10/2024	20/10/2024
4.	Test Functionality	21/10/2024	22/10/2024
5.	Review and finalize the report.	24/10/2024	24/10/2024

4. Brief Description of Micro project

The Election Management System is a software application designed to streamline the voting process and manage voter registrations effectively. It features two main classes: Election and Voter.

The Election class represents individual voters, encompassing attributes such as voter ID, name, age, contact number, city, and voting status. It initializes these attributes and keeps track of whether a voter has already cast their vote.

The Voter class manages an array of Election objects, allowing for various operations related to voter registration, displaying voter details, and casting votes. Key functionalities include the ability to fill out registration forms, which checks voter eligibility and stores their information. Additionally, it allows users to display the details of registered voters and facilitates the voting process by verifying voter IDs, presenting a list of political parties, and recording votes while ensuring that each voter can vote only once.

The system features a command-line interface where users can easily navigate options to register, view details, or vote, providing a smooth experience. Input validation ensures that users receive immediate feedback on their actions, contributing to a transparent and efficient election process. Overall, this system enhances the management of elections, ensuring accuracy and accessibility for voters.

4.1. Used Concept

The C++ code for the Election Management System employs key programming concepts to create an organized application. It features Election and Voter classes that encapsulate voter details and manage interactions. Constructors initialize these classes, while data encapsulation keeps voter information organized. The system uses arrays to store multiple voter objects, conditional statements for eligibility checks, and loops for handling multiple registrations and displaying details. Switch statements facilitate party selection during voting, and input/output operations enable user interaction.

4.1.1 Classes and Objects

The system is designed with two main classes: Election and Voter. The Election class encapsulates individual voter details, such as ID, name, age, contact information, and voting status. The Voter class manages a collection of Election objects, allowing for operations like registration and voting. This structure promotes encapsulation by keeping voter data and behaviours together, making the code more modular and easier to maintain.

```
#include <iostream>
using namespace std;
class Voter
{
public:
    string voterID;
    void showID()
    {
      cout << "Voter ID: " << voterID << endl;
    }
};
int main()
{
  Voter voter;
  voter.voterID = "V001";
  voter.showID();
  return 0;
}
```

4.1.2 Data Members and Member Functions

The system uses data members like voterID, voterName, age, and contact to store important information about each voter. Member functions like fill_registration_form() help collect this information, while cast_vote() manages the voting process. These functions make the user experience smoother by providing clear steps for each task, keeping the code organized. This structure also allows for easy updates and changes to specific features without impacting the whole system

4.1.3 Constructor Overloading

The Election class has two types of constructors: a default constructor that sets up an object with basic values and a parameterized constructor that allows you to directly assign voter details when creating an object. This feature, known as constructor overloading, gives flexibility in how you create voter instances, whether for registering new voters or updating existing ones. It makes the code easier to read and use, allowing developers to create objects in different ways depending on their needs.

4.1.4 Basic Voting Logic

The system is designed to track votes for various political parties, maintaining a count for each party as votes are cast. It implements a mechanism to ensure that each voter can only cast one vote, preventing any attempts at duplicate voting.

4.2 Programs or code of relevant concepts and output

```
#include<iostream>
#include<conio.h>
#include<string.h>
const int contact=10;
const int voterID=5;
int votes1=0,votes2=0,votes3=0,votes4=0,votes5=0,votes6=0,votes7=0,votes8=0,votes9=0;
int votes10=0;
using namespace std;

class Election
{
 public:
        string voterID, voterName,city;
        double age;
        int vote;
        long int contact;
        int hasVoted;
        Election() : hasVoted(0) {}

           Election(string vid, string vName, double vage,long int vContact,string vcity)
                    {
                    voterID = vid;
                    voterName = vName;
                    age = vage;
                    contact=vContact;
                    city=vcity;
                    hasVoted=0;
                   }
};

class Voter
{
```

```
protected:
        Election election[50];
        int total;
public:
         Voter()
         {
         total = 0;
         }

          void fill_registration_form();
          void display_voter_details();
          void cast_vote();
          void display_header();
         void display_footer();
};

void Voter::display_header()
 {
 cout << "\n=================================================" << endl;
 cout << "                GENERAL ELECTION VOTING SYSTEM                "<< endl;
 cout << "=================================================" << endl;
 }

 void Voter::display_footer()
 {
   cout << "=========================" << endl;
   cout << " Thank you for participating!       " << endl;
   cout << "=========================" << endl;
 }

 void Voter::fill_registration_form()
 {
     int ch;
    cout << "How many voters want to fill registration for?" << endl;
    cin >> ch;

     for (int i = 0; i < ch; i++)
     {
            string voterID, voterName, city;
            double age;
            long int contact;

          cout << "\n---------------- Fill The Registration Form ----------------- " << endl;
          cout << "Enter your age: ";
          cin >> age;
                if (age >= 18)
                 {
                        cout << "You are eligible for voting!" << endl;
                        cout << "Enter voter ID: ";
                        cin >> voterID;
```

```
                        cout << "Enter voter name: ";
                        cin >> voterName;
                        cout << "Enter voter contact number: ";
                        cin >> contact;
                        cout << "Enter voter city: "<<endl;
                        cin >> city;
                        cout << "Registration Successful!" << endl;
                    election[total] = Election(voterID, voterName, age, contact, city);
                     total++;
                }
                else
                {
                        cout << "You are not eligible for voting!" << endl;
                }
        }
}
void Voter::display_voter_details()
{
        if (total == 0)
        {
           cout << "No voter details exist!" << endl;
           return;
        }

         for (int i = 0; i < total; i++)
         {
             cout << "\nVoter " << i + 1 << endl;
             cout << "Voter Age: " << election[i].age << endl;
             cout << "Voter Name: " << election[i].voterName << endl;
             cout << "Voter ID: " << election[i].voterID << endl;
             cout << "Voter Contact: " << election[i].contact << endl;
             cout << "Voter City: " << election[i].city << endl;
          }
}

void Voter::cast_vote()
{
   string voterID;
   int vote;
   cout << "Enter voter ID: ";
   cin >> voterID;

            for (int i = 0; i < total; i++)
            {
                if (voterID == election[i].voterID)
                {
                    if (election[i].hasVoted)
                    {
                      cout << "This voter has already voted!" << endl;
                      return;
```

```
                    }

cout << "-*-*-*-*-*-*-List of political parties in Maharashtra-*-*-*-*-*-*-" << endl;
                cout << "1. BJP" << endl;
                cout << "2. BSP" << endl;
                cout << "3. CPI" << endl;
                cout << "4. INC" << endl;
                cout << "5. NCP" << endl;
                cout << "6. AAP" << endl;
                cout << "7. MNS" << endl;
                cout << "8. SHS" << endl;
                cout << "9. SP" << endl;
                cout << "10. OTHERS" << endl;
                cout<< "-----------------------------------------------------------"<<endl;
                cout << ">>>>Select your political party<<<<" << endl;

        cout << "Enter your vote" << endl;
         cin >> vote;

                 switch (vote)
                {
                        case 1: votes1++; break;
                        case 2: votes2++; break;
                        case 3: votes3++; break;
                        case 4: votes4++; break;
                        case 5: votes5++; break;
                        case 6: votes6++; break;
                        case 7: votes7++; break;
                        case 8: votes8++; break;
                        case 9: votes9++; break;
                        case 10: votes10++; break;
                        default: cout << "Invalid option!" << endl; return;
                }

                   election[i].hasVoted =1;
                        cout << "Vote cast successfully!" << endl;

                cout << "--------------------- Voting Result ----------------------" << endl;
                cout << "| Number of votes for BJP:\t " <<votes1 << endl;
                cout << "| Number of votes for BSP:\t " << votes2 << endl;
                cout << "| Number of votes for CPI:\t " << votes3 << endl;
                cout << "| Number of votes for INC:\t " << votes4 << endl;
                cout << "| Number of votes for NCP:\t " << votes5 << endl;
                cout << "| Number of votes for AAP:\t " << votes6 << endl;
                cout << "| Number of votes for MNS:\t " << votes7 << endl;
                cout << "| Number of votes for SHS:\t " << votes8 << endl;
                cout << "| Number of votes for SP:\t " << votes9 << endl;
                cout << "| Number of votes for OTHERS:\t  " << votes10 << endl;
                cout << "-----------------------------------------------------------" << endl;
            }
```

```
            }
        cout << "Voter ID does not exist!" << endl;
  }

int main()
{
          int ch;
         Voter v;
          v.display_header();
            do
            {
              cout<<endl;
              cout << "\n-~-~-~-~-~-~- Election Voting System -~-~-~-~-~-~-" << endl;
              cout << "Enter your choice:" << endl;
              cout << "|1. Fill Registration Form" << endl;
              cout << "|2. Display Voter Details" << endl;
                  cout << "|3. Vote" << endl;
                  cout << "|4. Exit" << endl;
                  cin >> ch;

                  switch (ch)
                  {
                      case 1:
                            v.fill_registration_form();
                           break;

                      case 2:
                            v.display_voter_details();
                            break;

                      case 3:
                          v.cast_vote();
                          break;

                        case 4:
                            v.display_footer() ;
                            break;

                      default:
                        cout << "Invalid option!" << endl;
                        break;
                  }
              } while (ch != 4);
        return 0;
  }
```

5.Output

```
==================================================
        GENERAL ELECTION VOTING SYSTEM
==================================================


-~-~-~-~-~- Election Voting System -~-~-~-~-~-
Enter your choice:
|1. Fill Registration Form
|2. Display Voter Details
|3. Vote
|4. Exit
1
How many voters want to fill registration for?
3

---------------------- Fill The Registration Form ----------------------
Enter your age: 23
You are eligible for voting!
Enter voter ID: 12201
Enter voter name: vaishnavi
Enter voter contact number: 937080201
Enter voter city: pune
Registration Successful!

---------------------- Fill The Registration Form ----------------------
Enter your age: 12
You are not eligible for voting!

---------------------- Fill The Registration Form ----------------------
Enter your age: 34
You are eligible for voting!
Enter voter ID: 12202
Enter voter name: puja
Enter voter contact number: 939021345
Enter voter city: solapur
Registration Successful!
```

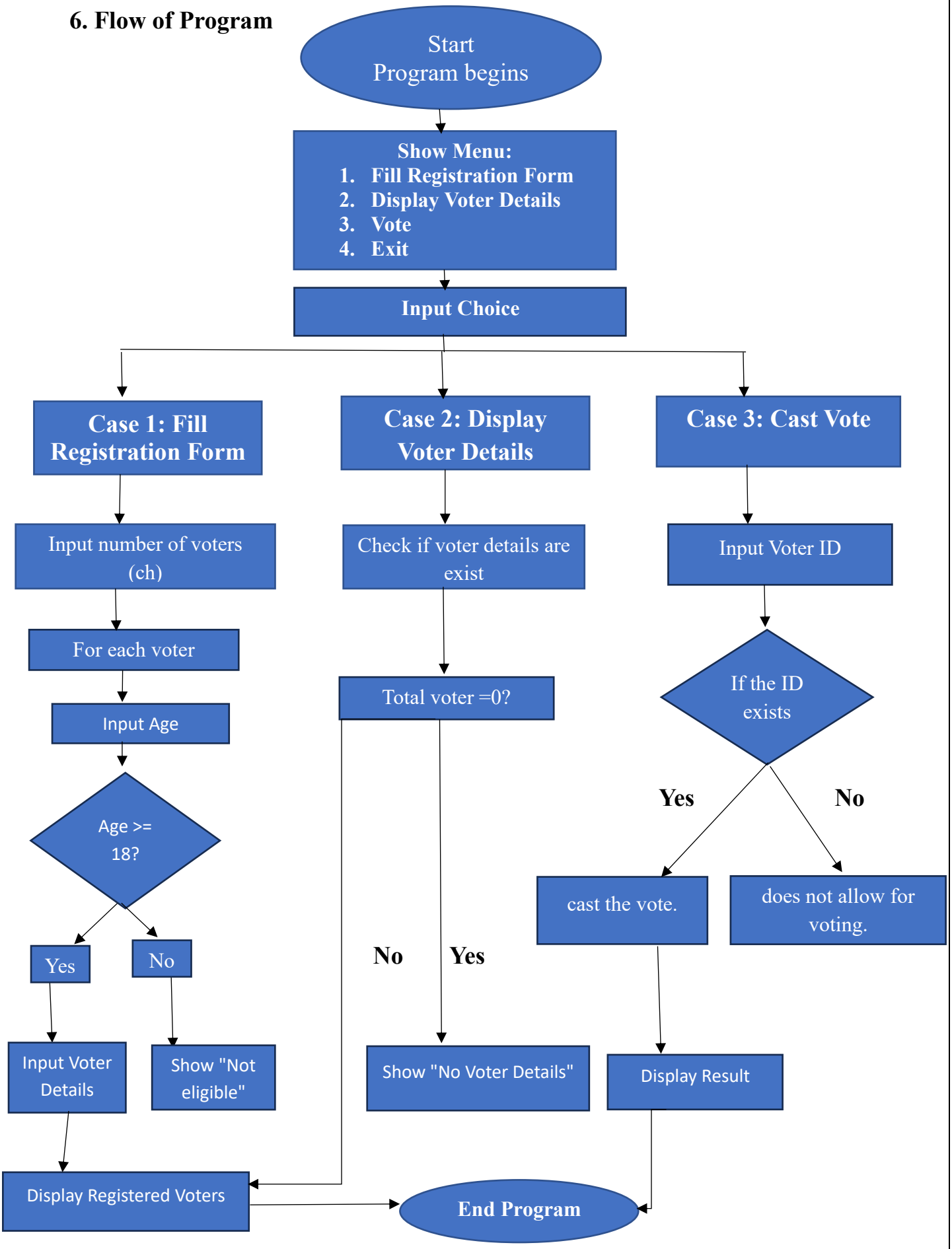
```
-~-~-~-~-~-~- Election Voting System -~-~-~-~-~-~-
Enter your choice:
|1. Fill Registration Form
|2. Display Voter Details
|3. Vote
|4. Exit
2

Voter 1
Voter Age: 23
Voter Name: vaishnavi
Voter ID: 12201
Voter Contact: 937080201
Voter City: pune

Voter 2
Voter Age: 34
Voter Name: puja
Voter ID: 12202
Voter Contact: 939021345
Voter City: solapur
```

```
-~-~-~-~-~- Election Voting System -~-~-~-~-~-
Enter your choice:
|1. Fill Registration Form
|2. Display Voter Details
|3. Vote
|4. Exit
3
Enter voter ID: 12201
-*-*-*-*-*-List of political parties in Maharashtra-*-*-*-*-*-
1. BJP
2. BSP
3. CPI
4. INC
5. NCP
6. AAP
7. MNS
8. SHS
9. SP
10. OTHERS
------------------------------------------------------------

>>>>Select your political party<<<<
Enter your vote
4
Vote cast successfully!
-------------------- Voting Result ---------------------
| Number of votes for BJP:       0
| Number of votes for BSP:       0
| Number of votes for CPI:       0
| Number of votes for INC:       1
| Number of votes for NCP:       0
| Number of votes for AAP:       0
| Number of votes for MNS:       0
| Number of votes for SHS:       0
| Number of votes for SP:        0
| Number of votes for OTHERS:     0
------------------------------------------------------------
```

```
-~-~-~-~-~- Election Voting System -~-~-~-~-~-
Enter your choice:
|1. Fill Registration Form
|2. Display Voter Details
|3. Vote
|4. Exit
3
Enter voter ID: 12201
This voter has already voted!


-~-~-~-~-~- Election Voting System -~-~-~-~-~-
Enter your choice:
|1. Fill Registration Form
|2. Display Voter Details
|3. Vote
|4. Exit
3
Enter voter ID: 12202
-*-*-*-*-*-List of political parties in Maharashtra-*-*-*-*-*-
1. BJP
2. BSP
3. CPI
4. INC
5. NCP
6. AAP
7. MNS
8. SHS
9. SP
10. OTHERS
------------------------------------------------------------

>>>>Select your political party<<<<
Enter your vote
4
Vote cast successfully!
```

```
-------------------- Voting Result ---------------------
| Number of votes for BJP:       0
| Number of votes for BSP:       0
| Number of votes for CPI:       0
| Number of votes for INC:       2
| Number of votes for NCP:       0
| Number of votes for AAP:       0
| Number of votes for MNS:       0
| Number of votes for SHS:       0
| Number of votes for SP:        0
| Number of votes for OTHERS:     0
------------------------------------------------------------


-~-~-~-~-~- Election Voting System -~-~-~-~-~-
Enter your choice:
|1. Fill Registration Form
|2. Display Voter Details
|3. Vote
|4. Exit
4
=========================
 Thank you for participating!
=========================
```

6. Flow of Program



7. Advantages

1. Efficient Voter Management: The system allows for easy registration and tracking of multiple voters, making it straightforward to manage a large number of participants in elections.
2. Data Integrity: By using classes to encapsulate voter information, the system ensures that all data (like voter ID and contact details) is organized and protected from unauthorized access.
3. Simplified Voting Process: With constructors, voters can be registered quickly and easily, streamlining the process of adding new participants to the system.
4. Adaptable Features: The code can be easily extended to accommodate new voting methods or types of voters, allowing for flexibility as election processes evolve.
5. Concurrent Voting: The system enables multiple voters to cast their votes independently, ensuring that one voter's actions do not affect another, which is crucial for a fair election process.

8. Disadvantage

1. Limited Number of Voters: The system can only handle up to 50 voters, which may not be enough for larger elections.
2. Weak Security: There are no strong security features to protect voter information, making it easier for unauthorized access.
3. No Data Saving: If the program is closed, all voter information and votes are lost, so there's no way to keep track of them over time.
4. Input Errors: The system doesn't check very well for mistakes in user input, like entering letters instead of numbers for age or contact info.
5. No Multi-User Support: The program can only handle one voter at a time, which could be a problem if many people want to vote at once.

9. Applications

1. Local Elections: Used for managing votes in town or city elections.
2. School Elections: Helps students vote for class representatives or student councils.
3. Event Planning: Collects votes from members on event dates or activities.
4. Corporate Elections: Used by companies to manage votes for employee representatives or committee members.
5. Online Polls: Can be adapted for websites to conduct online polls on various topics.
6. Feedback Collection: Gathers input from groups on events or services.

10.Conclusion

In conclusion, the Election Management System you've developed effectively demonstrates key programming concepts such as classes, constructors, and object-oriented principles. It provides a structured approach to managing voter registration and voting processes, making it user-friendly and efficient. While it has advantages like easy voter management and the ability to handle multiple voters, it also faces challenges such as limited scalability and security concerns. With potential applications ranging from local elections to community surveys, this system can serve various organizations and events. Overall, enhancing the system with better security measures and data management features could significantly improve its functionality and reliability in real-world scenarios.

11. Reference

1. https://www.geeksforgeeks.org/application-of-oops-in-cpp/
2. https://www.scribd.com/document/638014057/29-MiniProject-ONLINE-VOTING-SYSTEM
3. https://www.miniprojects.com
4. https://www.techtarget.com/searchapparchitecture/definition/object-oriented-programming-OOP