Electronic Voting System

M. Ahabb Sheraz, Reg. no. 2021327, CS103 B Semester Project

Problem Statement:

A valid voter will be casting their ballot for the candidates in provincial and federal general elections. After ending the election, the results of the election will be announced.

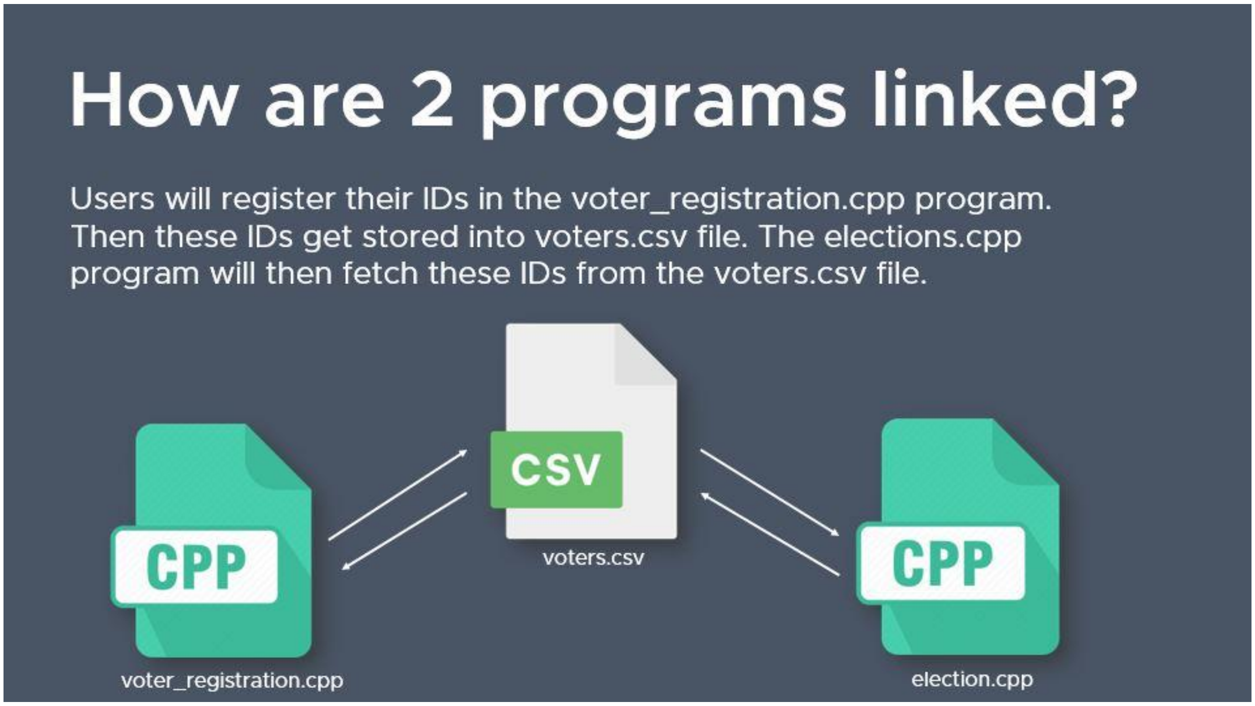
Introduction:

Firstly, Users will register as a voter using their IDs (CNICs). This will be done using voter_registration.cpp program. Voters will be able to vote for their parties in the election.cpp program but only if their IDs are registered.

Algorithm Design and Code:

This section has been divided into two main parts:

- Algorithm design and code of voter_registration.cpp
- Algorithm design and code of election.cpp



How are 2 programs linked?

Users will register their IDs in the voter_registration.cpp program. Then these IDs get stored into voters.csv file. The elections.cpp program will then fetch these IDs from the voters.csv file.

voter_registration.cpp

This code consists of the following functions:

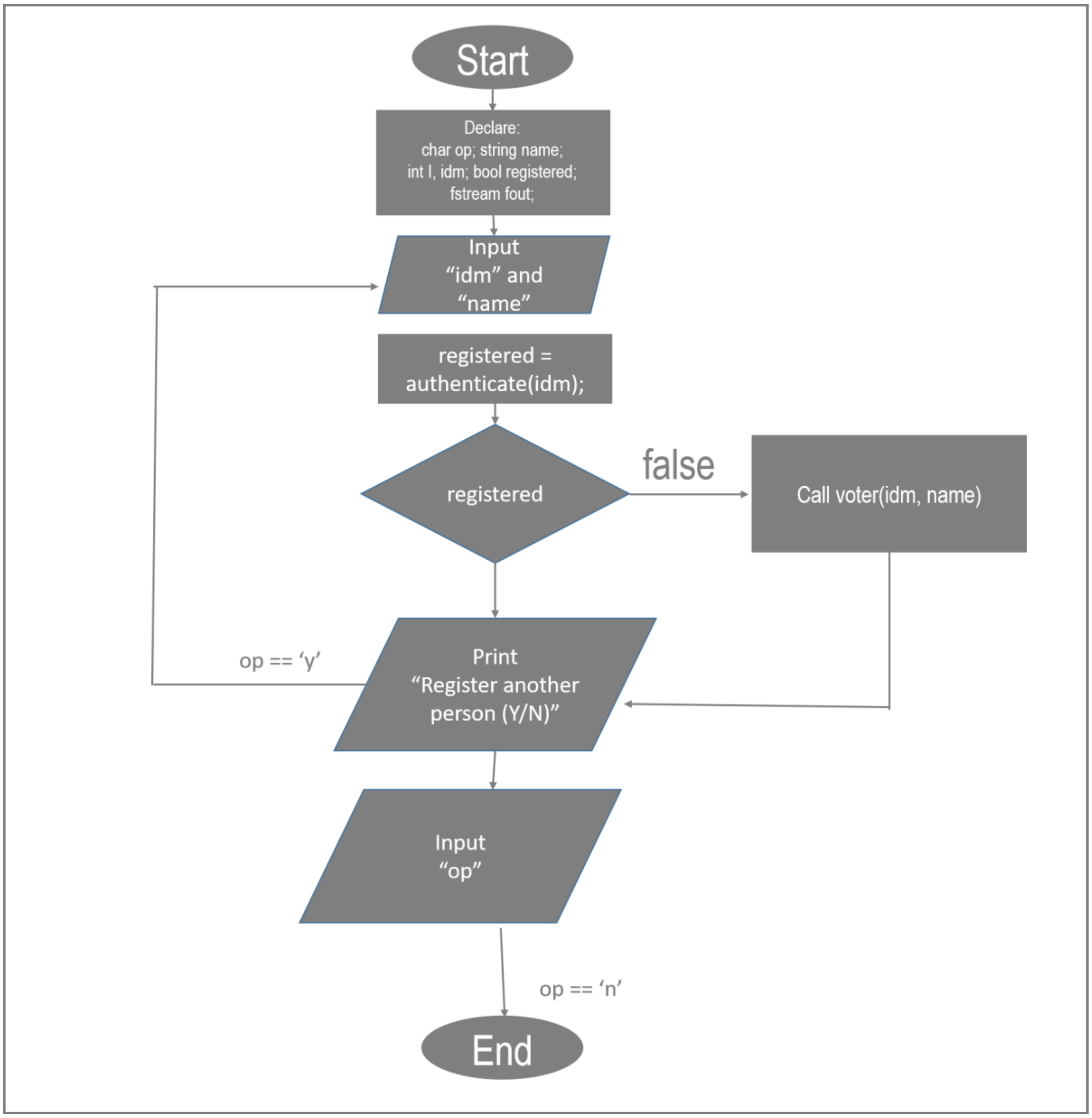
- authenticate():

This function verifies if the typed ID has already been registered or not. It returns the Boolean value of 0 if ID has not been registered before.

- voter():

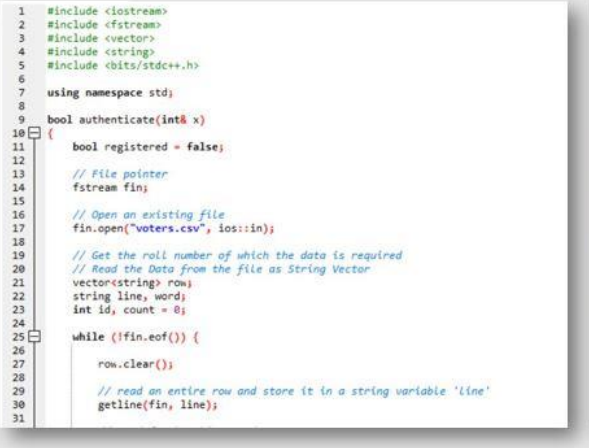
Unregisters IDs get stored into the voters.csv using this function. No return value.

int main()

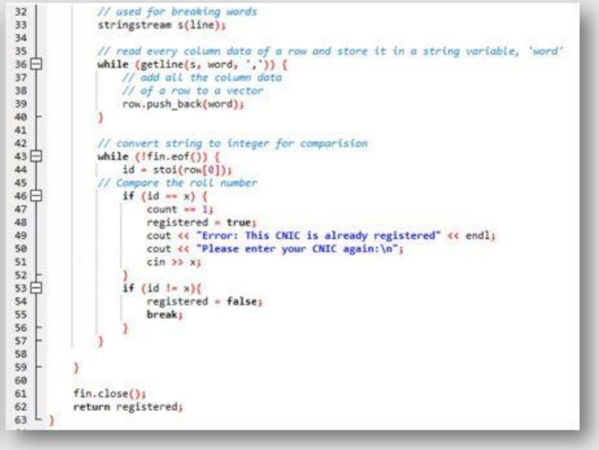


Flowchart of voter_registration

▪ authenticate()

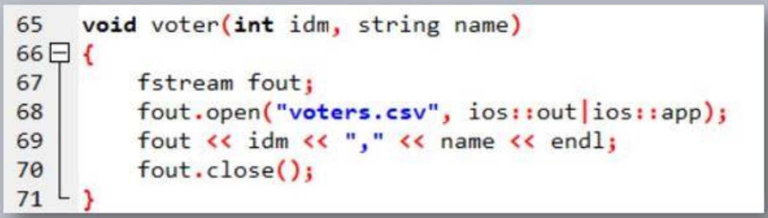


```
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <bits/stdc++.h>

using namespace std;

bool authenticate(int& x)
{
    bool registered = false;

    // File pointer
    fstream fin;

    // Open an existing file
    fin.open("voters.csv", ios::in);

    // Get the roll number of which the data is required
    // Read the Data from the file as String Vector
    vector<string> row;
    string line, word;
    int id, count = 0;

    while (!fin.eof()) {

        row.clear();

        // read an entire row and store it in a string variable 'line'
        getline(fin, line);

```



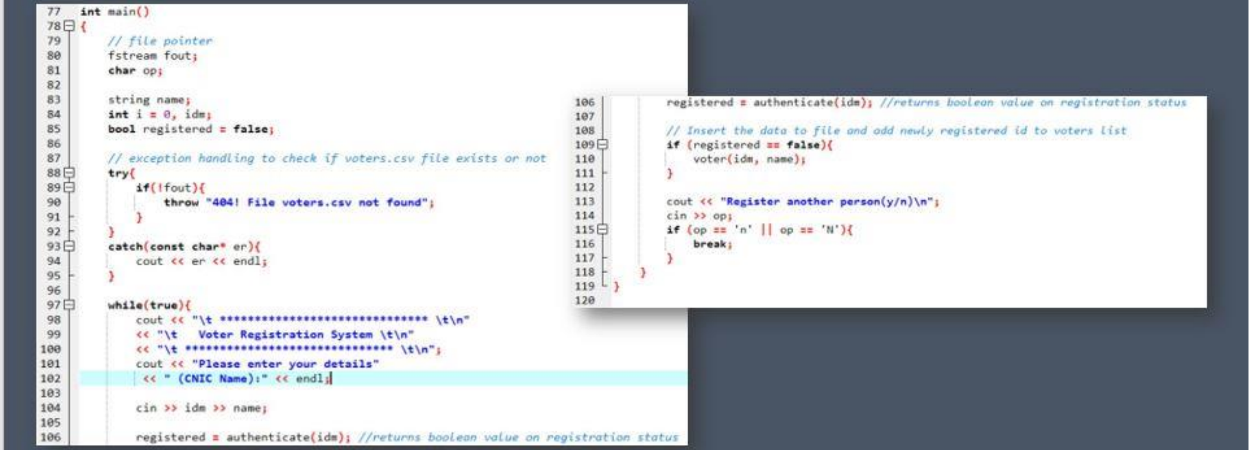
```
        // used for breaking words
        stringstream s(line);

        // read every column data of a row and store it in a string variable, 'word'
        while (getline(s, word, ',')) {
            // add all the column data
            // of a row to a vector
            row.push_back(word);
        }

        // convert string to integer for comparision
        while (!fin.eof()) {
            id = stoi(row[0]);
        // Compare the roll number
            if (id == x) {
                count == 1;
                registered = true;
                cout << "Error: This CNIC is already registered" << endl;
                cout << "Please enter your CNIC again:\n";
                cin >> x;
            }
            if (id != x){
                registered = false;
                break;
            }
        }

    }

    fin.close();
    return registered;
}
```

▪ voter()



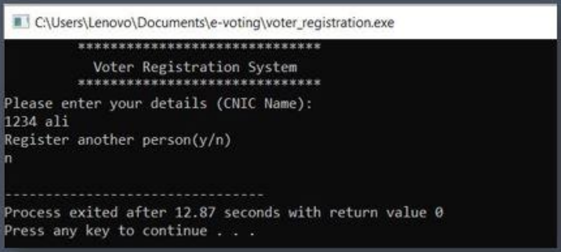
```
void voter(int idm, string name)
{
    fstream fout;
    fout.open("voters.csv", ios::out|ios::app);
    fout << idm << "," << name << endl;
    fout.close();
}
```

- main()

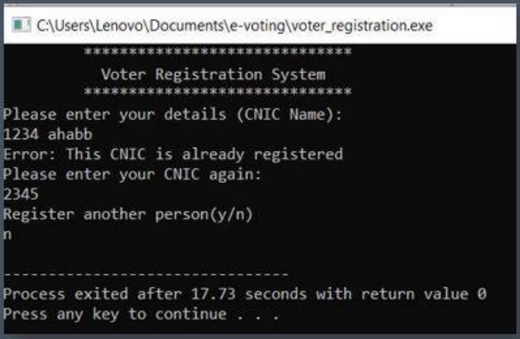


```cpp
int main()
{
    // file pointer
    fstream fout;
    char op;

    string name;
    int i = 0, idm;
    bool registered = false;

    // exception handling to check if voters.csv file exists or not
    try{
        if(!fout){
            throw "404! File voters.csv not found";
        }
    }
    catch(const char* er){
        cout << er << endl;
    }

    while(true){
        cout << "\t ****************************** \t\n"
        << "\t   Voter Registration System \t\n"
        << "\t ****************************** \t\n";
        cout << "Please enter your details"
         << " (CNIC Name):" << endl;

        cin >> idm >> name;

        registered = authenticate(idm); //returns boolean value on registration status

        // Insert the data to file and add newly registered id to voters list
        if (registered == false){
            voter(idm, name);
        }

        cout << "Register another person(y/n)\n";
        cin >> op;
        if (op == 'n' || op == 'N'){
            break;
        }
    }
}
```

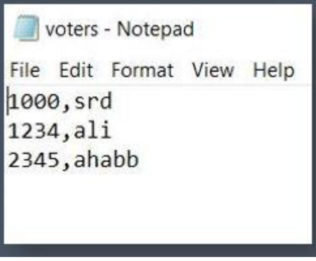
Program in action



```
        ******************************
          Voter Registration System
        ******************************
Please enter your details (CNIC Name):
1234 ali
Register another person(y/n)
n

--------------------------------
Process exited after 12.87 seconds with return value 0
Press any key to continue . . .
```



```
        ******************************
          Voter Registration System
        ******************************
Please enter your details (CNIC Name):
1234 ahabb
Error: This CNIC is already registered
Please enter your CNIC again:
2345
Register another person(y/n)
n

--------------------------------
Process exited after 17.73 seconds with return value 0
Press any key to continue . . .
```



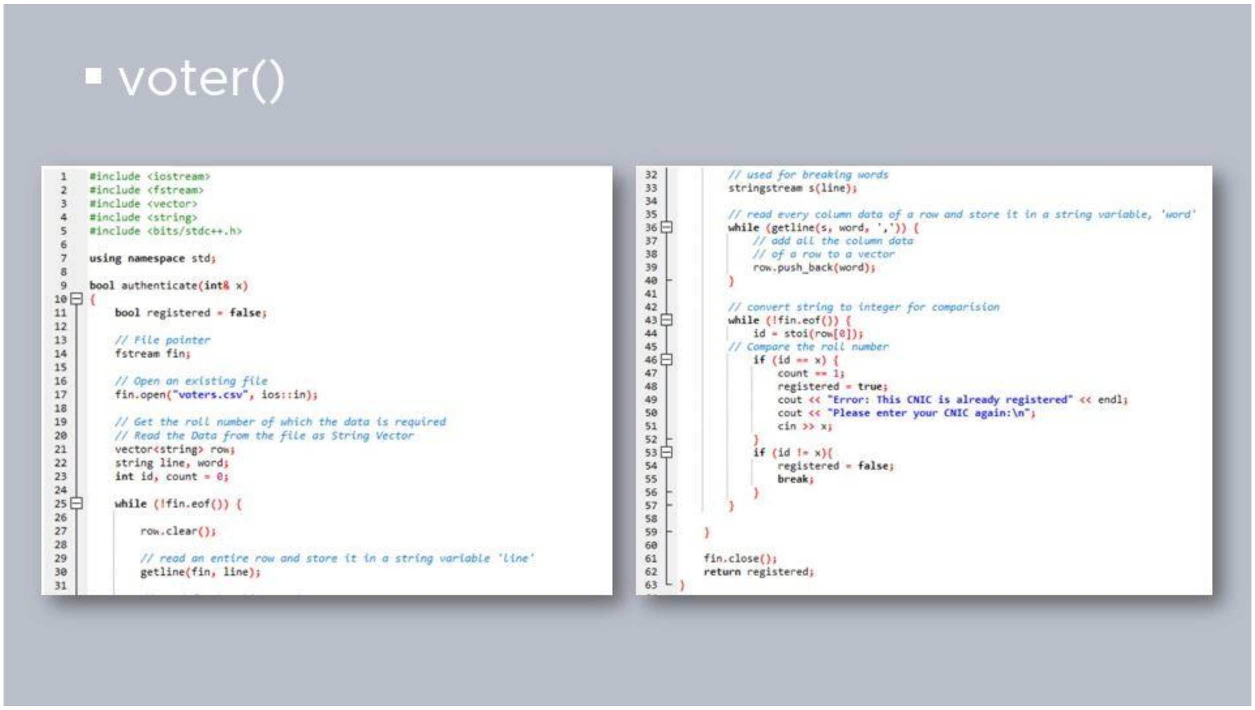
```
1000,srd
1234,ali
2345,ahabb
```

election.cpp

This code consists of the following functions:

- voter():

This function verifies if the typed ID has already been registered or not. In this function, I defined a fstream class named fin which has been linked to voters.csv file to compare the content in the file to the function parameters, given in the form of an address, given by the user.



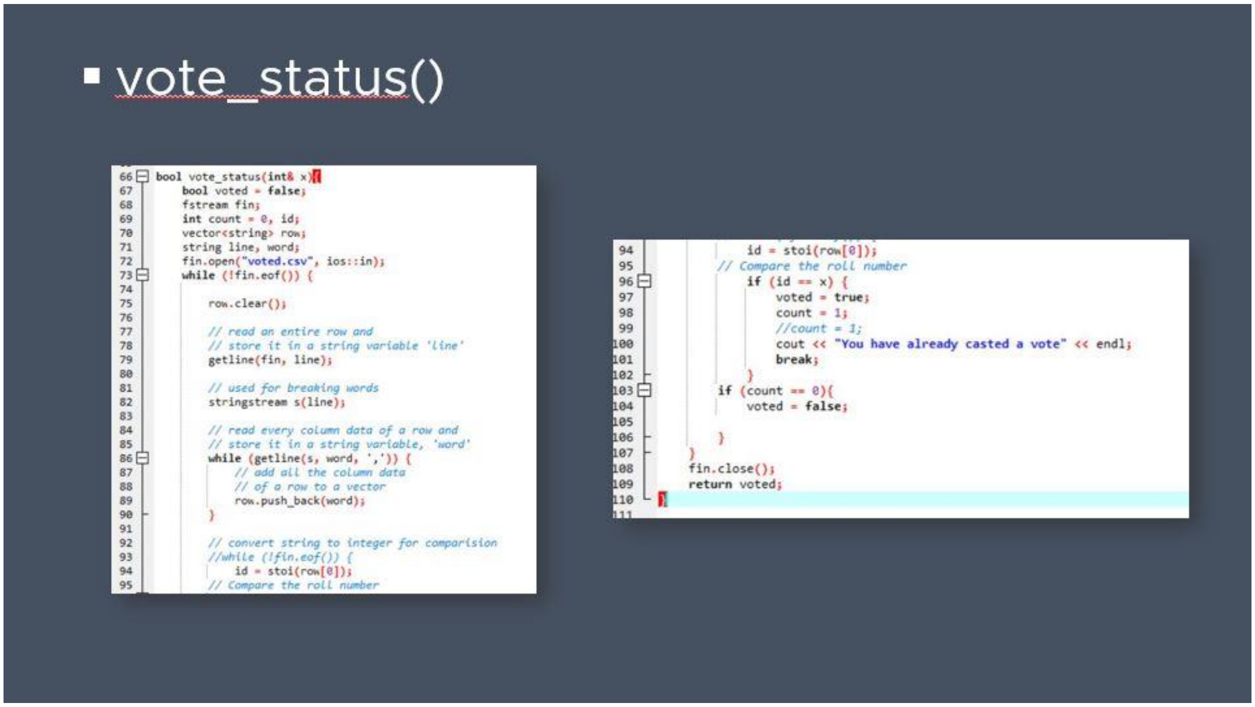
- voter()

```
#include <iostream>
#include <fstream>
#include <vector>
#include <string>
#include <bits/stdc++.h>

using namespace std;

bool authenticate(int& x)
{
    bool registered = false;

    // File pointer
    fstream fin;

    // Open an existing file
    fin.open("voters.csv", ios::in);

    // Get the roll number of which the data is required
    // Read the Data from the file as String Vector
    vector<string> row;
    string line, word;
    int id, count = 0;

    while (!fin.eof()) {

        row.clear();

        // read an entire row and store it in a string variable 'line'
        getline(fin, line);

```

```
        // used for breaking words
        stringstream s(line);

        // read every column data of a row and store it in a string variable, 'word'
        while (getline(s, word, ',')) {
            // add all the column data
            // of a row to a vector
            row.push_back(word);
        }

        // convert string to integer for comparision
        while (!fin.eof()) {
            id = stoi(row[0]);
        // Compare the roll number
            if (id == x) {
                count == 1;
                registered = true;
                cout << "Error: This CNIC is already registered" << endl;
                cout << "Please enter your CNIC again:\n";
                cin >> x;
            }
            if (id != x){
                registered = false;
                break;
            }
        }

    }

    fin.close();
    return registered;
}
```

- vote_status():

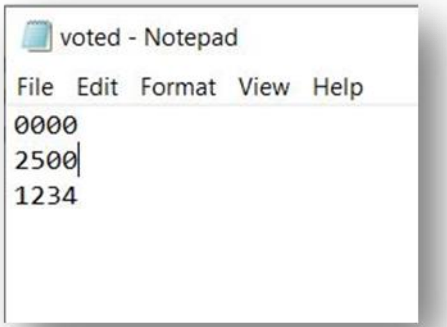
This function will not allow users who have already casted their vote to vote again as records of those who have already voted are kept in another file named voted.csv file. It also returns a Boolean value.



- vote_status()

```cpp
bool vote_status(int& x){
    bool voted = false;
    fstream fin;
    int count = 0, id;
    vector<string> row;
    string line, word;
    fin.open("voted.csv", ios::in);
    while (!fin.eof()) {

        row.clear();

        // read an entire row and
        // store it in a string variable 'line'
        getline(fin, line);

        // used for breaking words
        stringstream s(line);

        // read every column data of a row and
        // store it in a string variable, 'word'
        while (getline(s, word, ',')) {
            // add all the column data
            // of a row to a vector
            row.push_back(word);
        }

        // convert string to integer for comparision
        //while (!fin.eof()) {
            id = stoi(row[0]);
        // Compare the roll number
```

```cpp
            id = stoi(row[0]);
        // Compare the roll number
            if (id == x) {
                voted = true;
                count = 1;
                //count = 1;
                cout << "You have already casted a vote" << endl;
                break;
            }
        if (count == 0){
            voted = false;

        }
    }
    fin.close();
    return voted;
}
```



```
0000
2500
1234
```

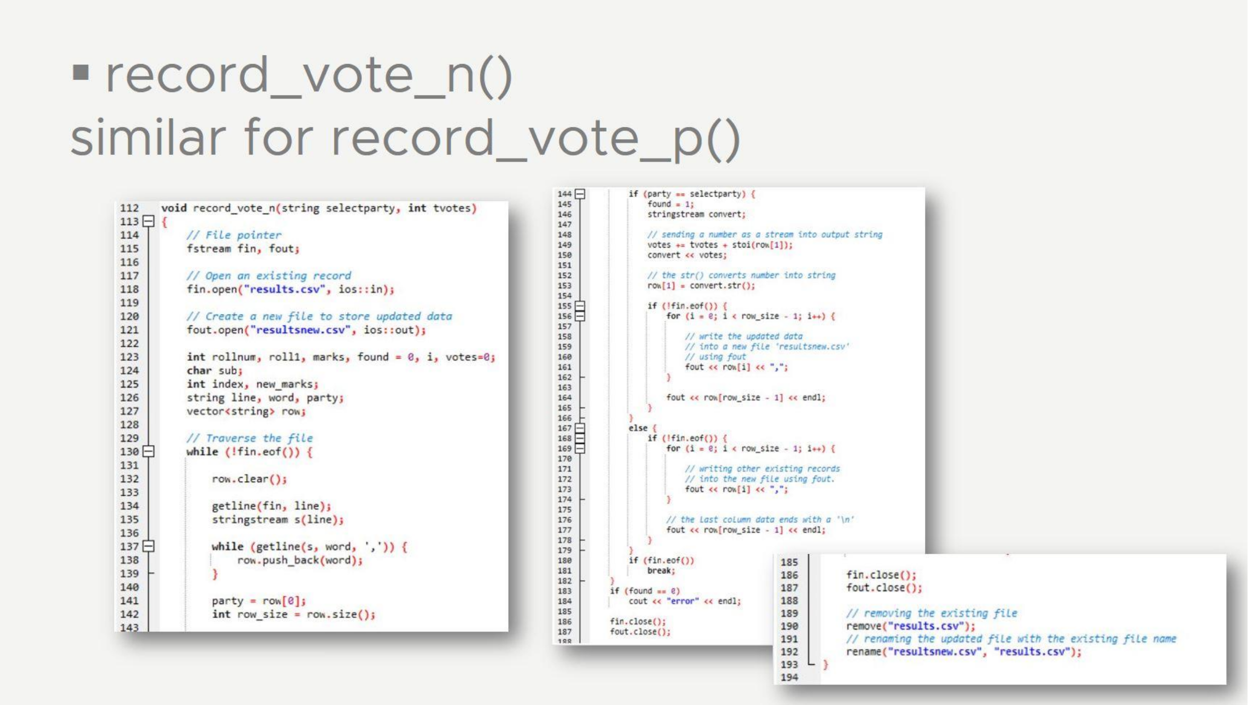
IDs of those who have already voted are kept in voted.csv.

- record_vote_n():

This function updates voting results for National Assembly candidates by updating file na_results.csv. At first, it adds existing data of na_results.csv with new changes to the newly created file na_resultsnew.csv and then deletes the original na_results.csv and renames na_resultsnew.csv to it.

- record_vote_p():

This function updates voting results for Provincial Assembly candidates. Working is very similar to function record_vote_n().



- record_vote_n()

similar for record_vote_p()

```
void record_vote_n(string selectparty, int tvotes)
{
    // File pointer
    fstream fin, fout;

    // Open an existing record
    fin.open("results.csv", ios::in);

    // Create a new file to store updated data
    fout.open("resultsnew.csv", ios::out);

    int rollnum, roll1, marks, found = 0, i, votes=0;
    char sub;
    int index, new_marks;
    string line, word, party;
    vector<string> row;

    // Traverse the file
    while (!fin.eof()) {

        row.clear();

        getline(fin, line);
        stringstream s(line);

        while (getline(s, word, ',')) {
            row.push_back(word);
        }

        party = row[0];
        int row_size = row.size();
```

```
        if (party == selectparty) {
            found = 1;
            stringstream convert;

            // sending a number as a stream into output string
            votes += tvotes + stoi(row[1]);
            convert << votes;

            // the str() converts number into string
            row[1] = convert.str();

            if (!fin.eof()) {
                for (i = 0; i < row_size - 1; i++) {

                    // write the updated data
                    // into a new file 'resultsnew.csv'
                    // using fout
                    fout << row[i] << ",";
                }

                fout << row[row_size - 1] << endl;
            }
        }
        else {
            if (!fin.eof()) {
                for (i = 0; i < row_size - 1; i++) {

                    // writing other existing records
                    // into the new file using fout.
                    fout << row[i] << ",";
                }

                // the last column data ends with a '\n'
                fout << row[row_size - 1] << endl;
            }
        }
        if (fin.eof())
            break;
    }
    if (found == 0)
        cout << "error" << endl;

    fin.close();
    fout.close();
```

```

    fin.close();
    fout.close();

    // removing the existing file
    remove("results.csv");
    // renaming the updated file with the existing file name
    rename("resultsnew.csv", "results.csv");
}
```

- na_displayresults():

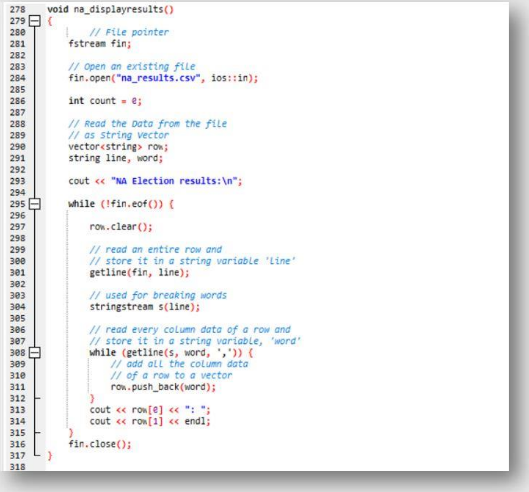
Function displays National Assembly election results.

- pa_displayresults():

Function displays Provincial Assembly election results.

- na_displayresults()

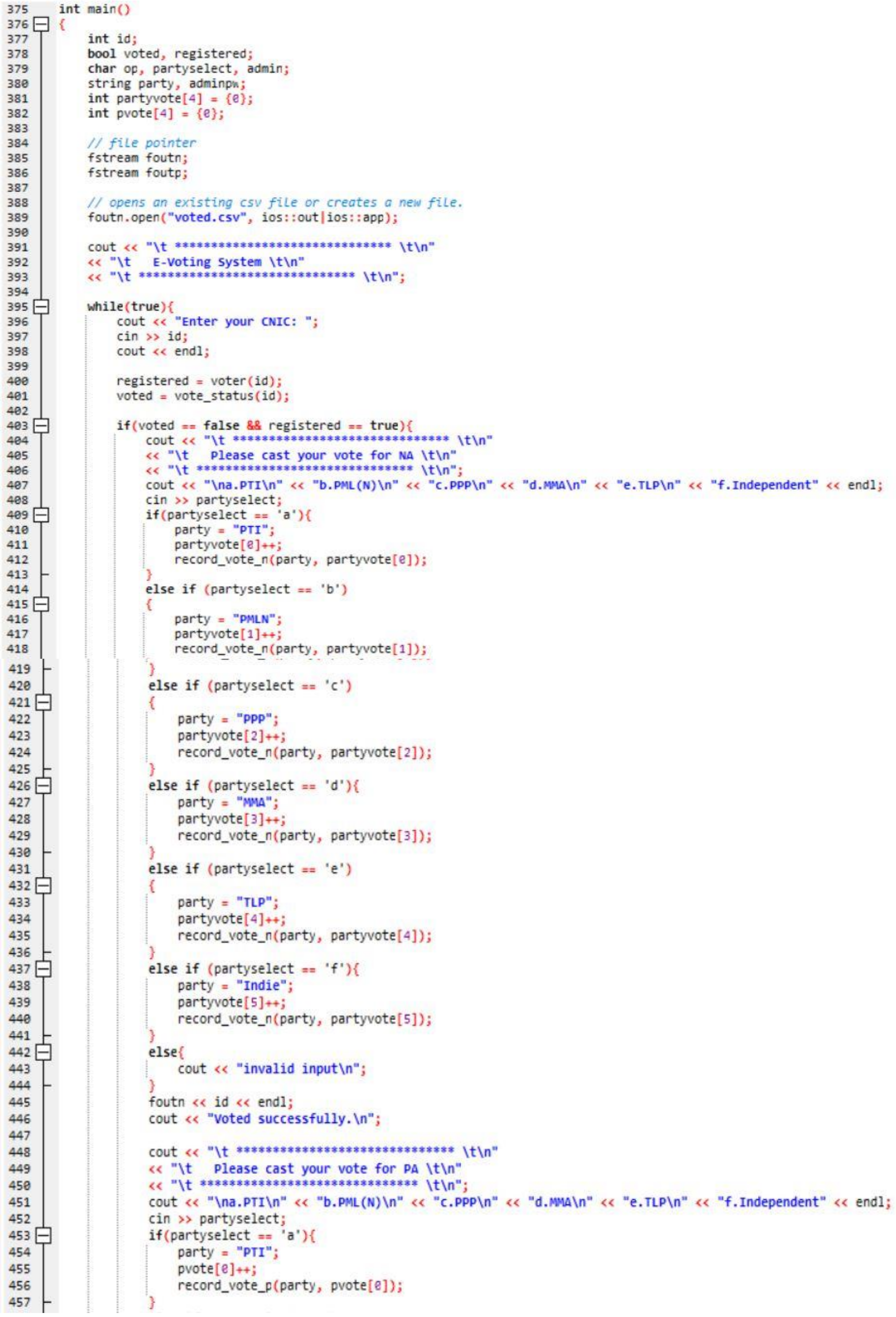
similar for pa_displayresults()



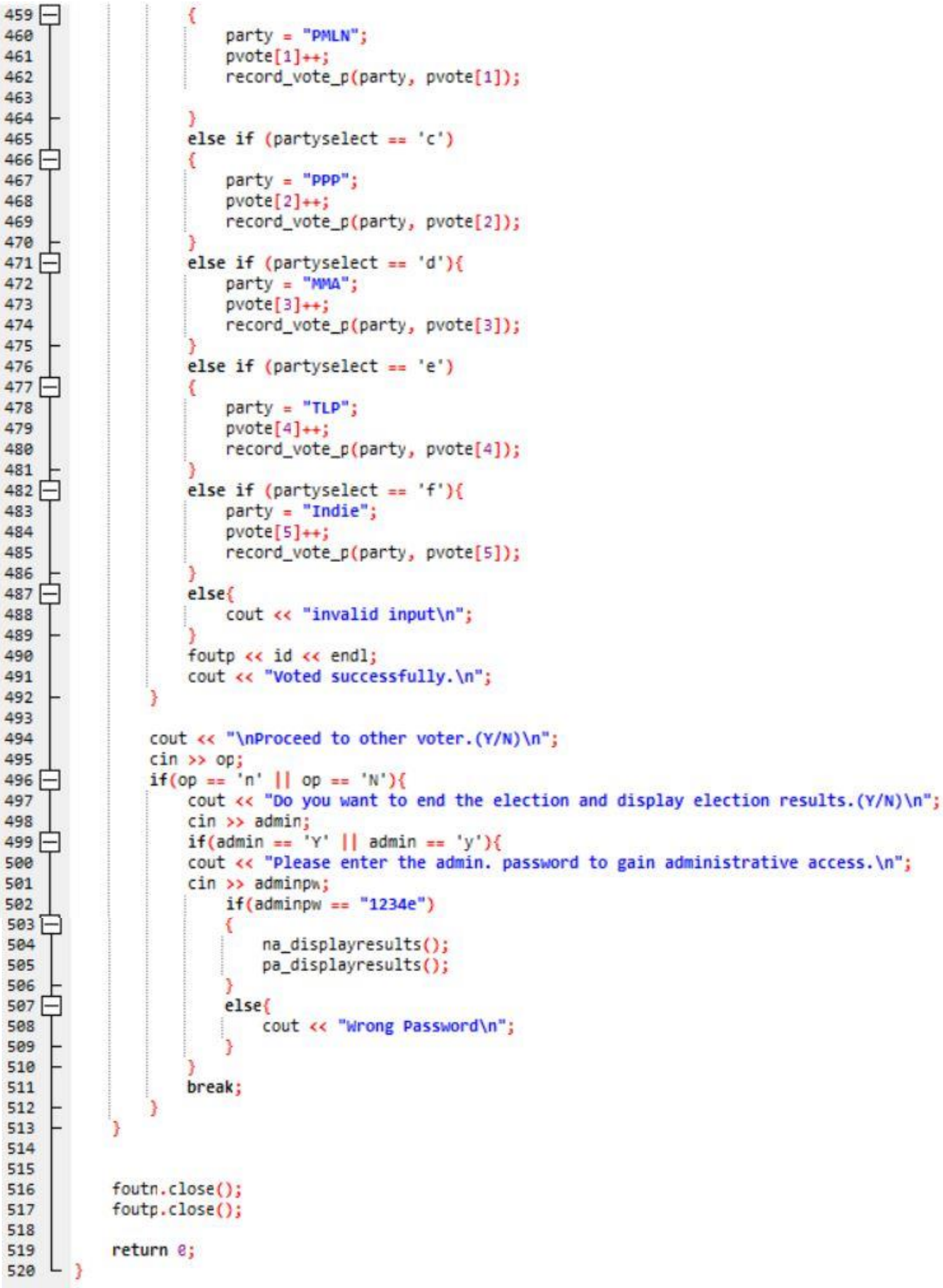
```
void na_displayresults()
{
        // File pointer
    fstream fin;

    // Open an existing file
    fin.open("na_results.csv", ios::in);

    int count = 0;

    // Read the Data from the file
    // as String Vector
    vector<string> row;
    string line, word;

    cout << "NA Election results:\n";

    while (!fin.eof()) {

        row.clear();

        // read an entire row and
        // store it in a string variable 'line'
        getline(fin, line);

        // used for breaking words
        stringstream s(line);

        // read every column data of a row and
        // store it in a string variable, 'word'
        while (getline(s, word, ',')) {
            // add all the column data
            // of a row to a vector
            row.push_back(word);
        }
        cout << row[0] << ": ";
        cout << row[1] << endl;
    }
    fin.close();
}
```

- main():

It is in main function where the user will vote for their party and get verified for voting eligibility.



```cpp
int main()
{
    int id;
    bool voted, registered;
    char op, partyselect, admin;
    string party, adminpw;
    int partyvote[4] = {0};
    int pvote[4] = {0};

    // file pointer
    fstream foutn;
    fstream foutp;

    // opens an existing csv file or creates a new file.
    foutn.open("voted.csv", ios::out|ios::app);

    cout << "\t ****************************** \t\n"
    << "\t   E-Voting System \t\n"
    << "\t ****************************** \t\n";

    while(true){
        cout << "Enter your CNIC: ";
        cin >> id;
        cout << endl;

        registered = voter(id);
        voted = vote_status(id);

        if(voted == false && registered == true){
            cout << "\t ****************************** \t\n"
            << "\t   Please cast your vote for NA \t\n"
            << "\t ****************************** \t\n";
            cout << "\na.PTI\n" << "b.PML(N)\n" << "c.PPP\n" << "d.MMA\n" << "e.TLP\n" << "f.Independent" << endl;
            cin >> partyselect;
            if(partyselect == 'a'){
                party = "PTI";
                partyvote[0]++;
                record_vote_n(party, partyvote[0]);
            }
            else if (partyselect == 'b')
            {
                party = "PMLN";
                partyvote[1]++;
                record_vote_n(party, partyvote[1]);
            }
            else if (partyselect == 'c')
            {
                party = "PPP";
                partyvote[2]++;
                record_vote_n(party, partyvote[2]);
            }
            else if (partyselect == 'd'){
                party = "MMA";
                partyvote[3]++;
                record_vote_n(party, partyvote[3]);
            }
            else if (partyselect == 'e')
            {
                party = "TLP";
                partyvote[4]++;
                record_vote_n(party, partyvote[4]);
            }
            else if (partyselect == 'f'){
                party = "Indie";
                partyvote[5]++;
                record_vote_n(party, partyvote[5]);
            }
            else{
                cout << "invalid input\n";
            }
            foutn << id << endl;
            cout << "Voted successfully.\n";

            cout << "\t ****************************** \t\n"
            << "\t   Please cast your vote for PA \t\n"
            << "\t ****************************** \t\n";
            cout << "\na.PTI\n" << "b.PML(N)\n" << "c.PPP\n" << "d.MMA\n" << "e.TLP\n" << "f.Independent" << endl;
            cin >> partyselect;
            if(partyselect == 'a'){
                party = "PTI";
                pvote[0]++;
                record_vote_p(party, pvote[0]);
            }
```



```
            {
                party = "PMLN";
                pvote[1]++;
                record_vote_p(party, pvote[1]);

            }
            else if (partyselect == 'c')
            {
                party = "PPP";
                pvote[2]++;
                record_vote_p(party, pvote[2]);
            }
            else if (partyselect == 'd'){
                party = "MMA";
                pvote[3]++;
                record_vote_p(party, pvote[3]);
            }
            else if (partyselect == 'e')
            {
                party = "TLP";
                pvote[4]++;
                record_vote_p(party, pvote[4]);
            }
            else if (partyselect == 'f'){
                party = "Indie";
                pvote[5]++;
                record_vote_p(party, pvote[5]);
            }
            else{
                cout << "invalid input\n";
            }
            foutp << id << endl;
            cout << "Voted successfully.\n";
        }

        cout << "\nProceed to other voter.(Y/N)\n";
        cin >> op;
        if(op == 'n' || op == 'N'){
            cout << "Do you want to end the election and display election results.(Y/N)\n";
            cin >> admin;
            if(admin == 'Y' || admin == 'y'){
            cout << "Please enter the admin. password to gain administrative access.\n";
            cin >> adminpw;
                if(adminpw == "1234e")
                {
                    na_displayresults();
                    pa_displayresults();
                }
                else{
                    cout << "Wrong Password\n";
                }
            }
            break;
        }
    }


    foutn.close();
    foutp.close();

    return 0;
}
```

Conclusion

This project was definitely interesting but also challenging. It tested my knowledge of CS103 to its full extend. By making File-based databases using File handling, I was able to complete this task. It was fun creating this project.