By
M. Ahabb Sheraz

Reg.
2021327

Semester Project
CS103 B

Introducing

E-voting system

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.

Instructions

voter_registration.cpp

1) Users will register as a voter using their IDs (CNICs). This will be done using voter_registration.cpp program.

2) Voters will vote for their parties in the election.cpp program.

election.cpp

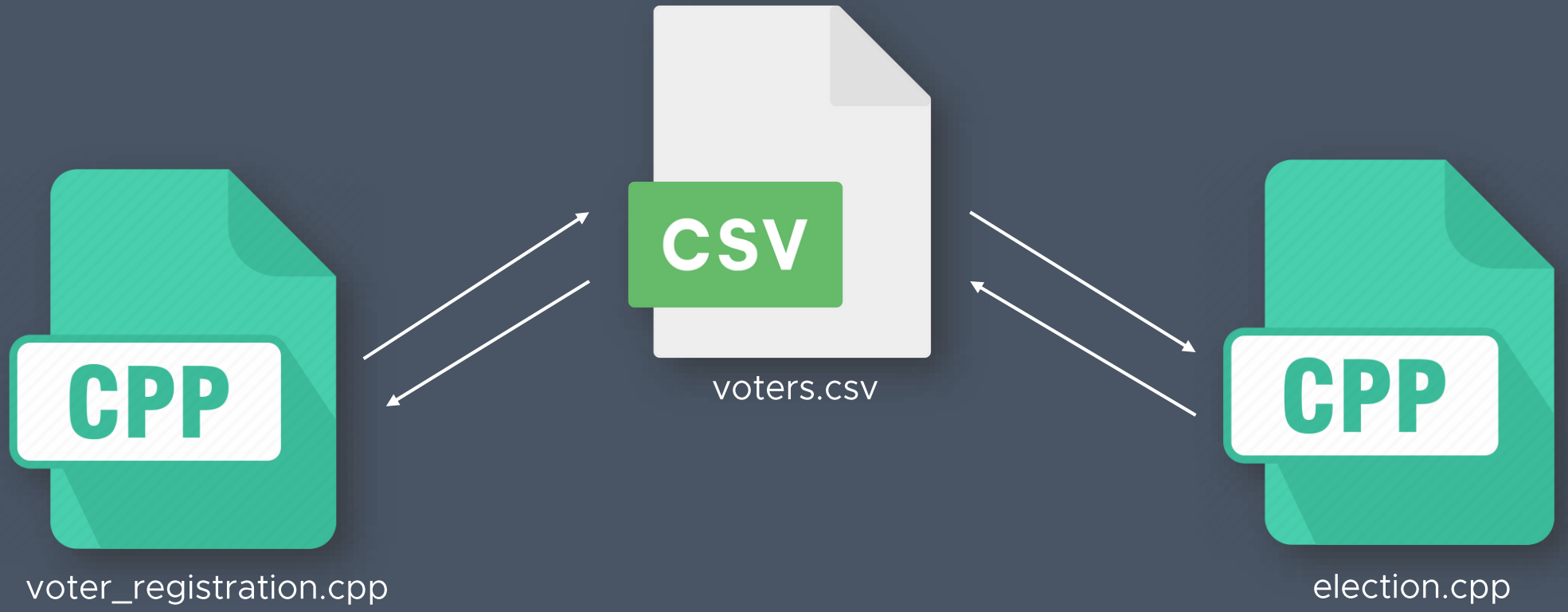
The Algorithm and Code

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

using namespace std;

void create()
{
    // file pointer
    fstream fout;

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

    cout << "Enter the details of 5 students"
         << " (roll name maths phy chem bio):" << endl;

    int i, roll, phy, chem, math, bio;
    string name;

    // Read the input
    for (i = 0; i < 5; i++) {
```

```
PS C:\Users\Lenovo> cd "c:\Users\Lenovo\Documents\e-voting\" ; if ($?) { g++ file3.cpp -o file3 } ; if ($?) { .\file3 }
Enter the details of 5 students (roll name maths phy chem bio):
ahshd
PS C:\Users\Lenovo\Documents\e-voting>
```

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.

authenticate()

```cpp
#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);
```

```cpp
        // 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()

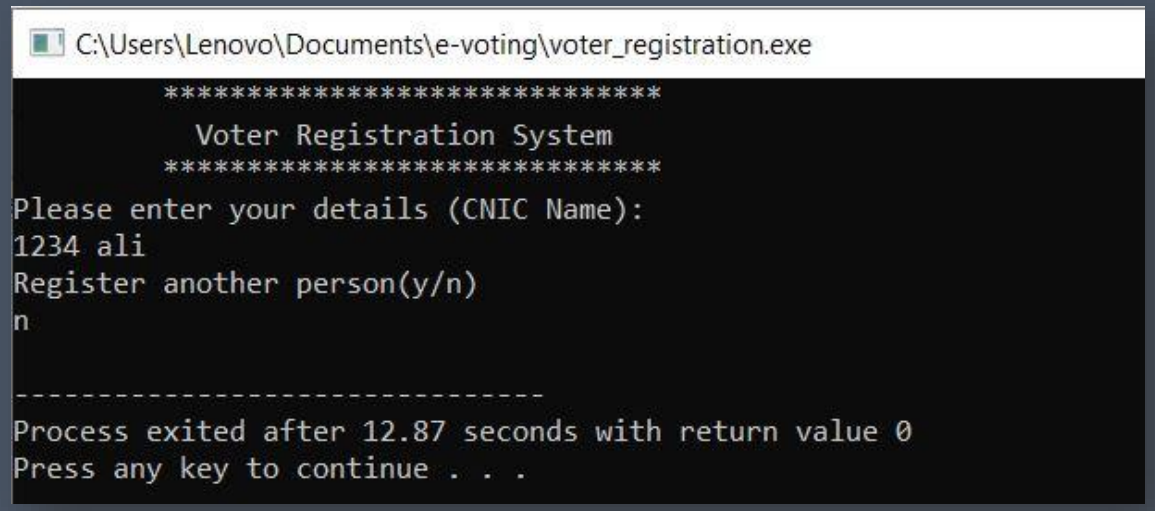
```
65  void voter(int idm, string name)
66  {
67      fstream fout;
68      fout.open("voters.csv", ios::out|ios::app);
69      fout << idm << "," << name << endl;
70      fout.close();
71  }
```

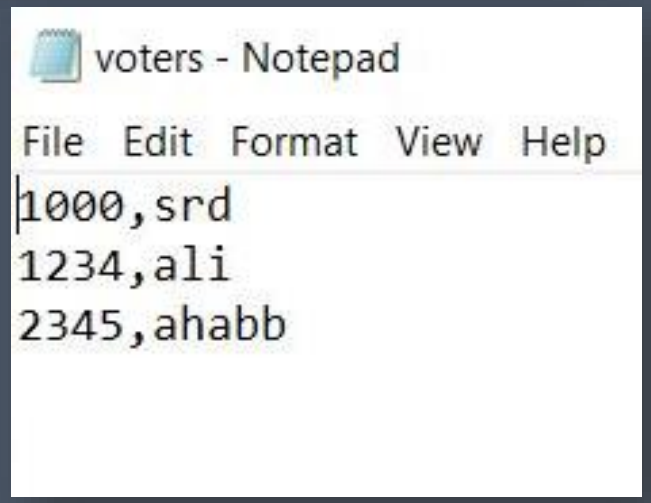
- main()

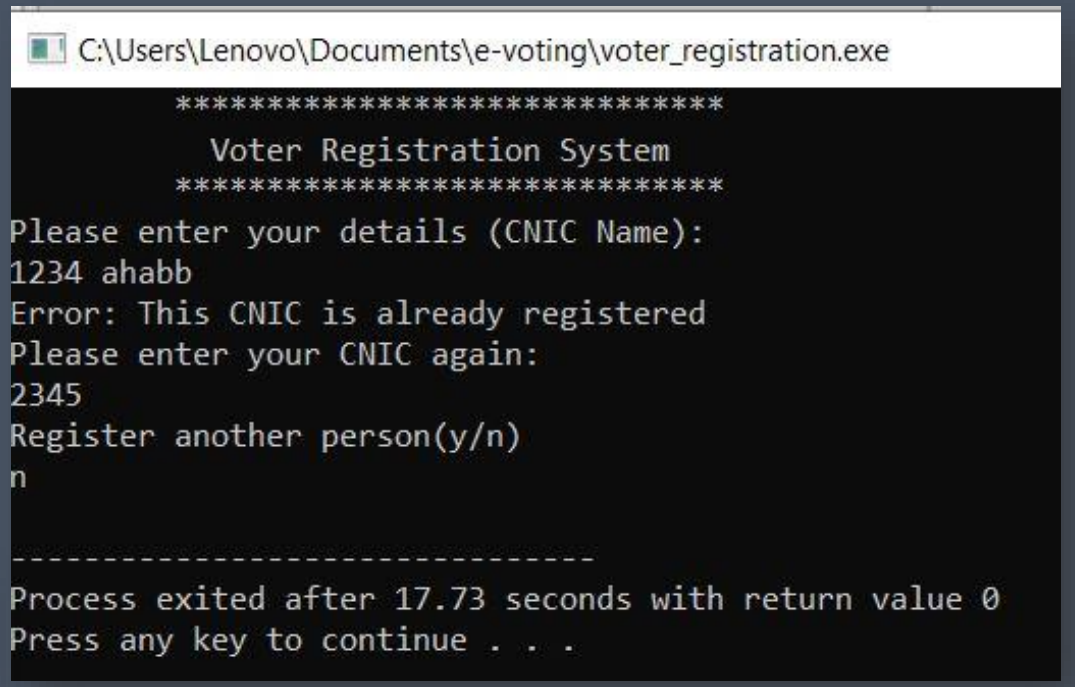
```
77  int main()
78  {
79      // file pointer
80      fstream fout;
81      char op;
82
83      string name;
84      int i = 0, idm;
85      bool registered = false;
86
87      // exception handling to check if voters.csv file exists or not
88      try{
89          if(!fout){
90              throw "404! File voters.csv not found";
91          }
92      }
93      catch(const char* er){
94          cout << er << endl;
95      }
96
97      while(true){
98          cout << "\t ***************************** \t\n"
99          << "\t   Voter Registration System \t\n"
100         << "\t ***************************** \t\n";
101         cout << "Please enter your details"
102          << " (CNIC Name):" << endl;
103
104         cin >> idm >> name;
105
106         registered = authenticate(idm); //returns boolean value on registration status
```

```
106         registered = authenticate(idm); //returns boolean value on registration status
107
108         // Insert the data to file and add newly registered id to voters list
109         if (registered == false){
110             voter(idm, name);
111         }
112
113         cout << "Register another person(y/n)\n";
114         cin >> op;
115         if (op == 'n' || op == 'N'){
116             break;
117         }
118     }
119 }
120
```

Program in action







election.cpp

This code consists of the following functions:

- voter():

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.

- vote_status():

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

election.cpp

- record_vote_n():

Updates vote for National Assembly candidates.

- record_vote_p():

Updates vote for National Assembly candidates.

- na_displayresults():

Displays National Assembly election results.

- pa_displayresults():

Displays Provincial Assembly election results.

voter()

```cpp
#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);
```

```cpp
        // 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()

```
66  bool vote_status(int& x){
67      bool voted = false;
68      fstream fin;
69      int count = 0, id;
70      vector<string> row;
71      string line, word;
72      fin.open("voted.csv", ios::in);
73      while (!fin.eof()) {
74
75          row.clear();
76
77          // read an entire row and
78          // store it in a string variable 'line'
79          getline(fin, line);
80
81          // used for breaking words
82          stringstream s(line);
83
84          // read every column data of a row and
85          // store it in a string variable, 'word'
86          while (getline(s, word, ',')) {
87              // add all the column data
88              // of a row to a vector
89              row.push_back(word);
90          }
91
92          // convert string to integer for comparision
93          //while (!fin.eof()) {
94              id = stoi(row[0]);
95          // Compare the roll number
```

```
94              id = stoi(row[0]);
95          // Compare the roll number
96              if (id == x) {
97                  voted = true;
98                  count = 1;
99                  //count = 1;
100                 cout << "You have already casted a vote" << endl;
101                 break;
102             }
103         if (count == 0){
104             voted = false;
105
106         }
107     }
108     fin.close();
109     return voted;
110 }
111
```

- record_vote_n()

similar for record_vote_p()

```
112  void record_vote_n(string selectparty, int tvotes)
113  {
114      // File pointer
115      fstream fin, fout;
116
117      // Open an existing record
118      fin.open("results.csv", ios::in);
119
120      // Create a new file to store updated data
121      fout.open("resultsnew.csv", ios::out);
122
123      int rollnum, roll1, marks, found = 0, i, votes=0;
124      char sub;
125      int index, new_marks;
126      string line, word, party;
127      vector<string> row;
128
129      // Traverse the file
130      while (!fin.eof()) {
131
132          row.clear();
133
134          getline(fin, line);
135          stringstream s(line);
136
137          while (getline(s, word, ',')) {
138              row.push_back(word);
139          }
140
141          party = row[0];
142          int row_size = row.size();
143
```

```
144          if (party == selectparty) {
145              found = 1;
146              stringstream convert;
147
148              // sending a number as a stream into output string
149              votes += tvotes + stoi(row[1]);
150              convert << votes;
151
152              // the str() converts number into string
153              row[1] = convert.str();
154
155              if (!fin.eof()) {
156                  for (i = 0; i < row_size - 1; i++) {
157
158                      // write the updated data
159                      // into a new file 'resultsnew.csv'
160                      // using fout
161                      fout << row[i] << ",";
162                  }
163
164                  fout << row[row_size - 1] << endl;
165              }
166          }
167          else {
168              if (!fin.eof()) {
169                  for (i = 0; i < row_size - 1; i++) {
170
171                      // writing other existing records
172                      // into the new file using fout.
173                      fout << row[i] << ",";
174                  }
175
176                  // the last column data ends with a '\n'
177                  fout << row[row_size - 1] << endl;
178              }
179          }
180          if (fin.eof())
181              break;
182      }
183      if (found == 0)
184          cout << "error" << endl;
185
186      fin.close();
187      fout.close();
```

```
185
186      fin.close();
187      fout.close();
188
189      // removing the existing file
190      remove("results.csv");
191      // renaming the updated file with the existing file name
192      rename("resultsnew.csv", "results.csv");
193  }
194
```

- na_displayresults()

similar for pa_displayresults()

```
278  void na_displayresults()
279  {
280      // File pointer
281      fstream fin;
282
283      // Open an existing file
284      fin.open("na_results.csv", ios::in);
285
286      int count = 0;
287
288      // Read the Data from the file
289      // as String Vector
290      vector<string> row;
291      string line, word;
292
293      cout << "NA Election results:\n";
294
295      while (!fin.eof()) {
296
297          row.clear();
298
299          // read an entire row and
300          // store it in a string variable 'line'
301          getline(fin, line);
302
303          // used for breaking words
304          stringstream s(line);
305
306          // read every column data of a row and
307          // store it in a string variable, 'word'
308          while (getline(s, word, ',')) {
309              // add all the column data
310              // of a row to a vector
311              row.push_back(word);
312          }
313          cout << row[0] << ": ";
314          cout << row[1] << endl;
315      }
316      fin.close();
317  }
318
```

- main()

```
375  int main()
376  {
377      int id;
378      bool voted, registered;
379      char op, partyselect, admin;
380      string party, adminpw;
381      int partyvote[4] = {0};
382      int pvote[4] = {0};
383
384      // file pointer
385      fstream foutn;
386      fstream foutp;
387
388      // opens an existing csv file or creates a new file.
389      foutn.open("voted.csv", ios::out|ios::app);
390
391      cout << "\t ***************************** \t\n"
392      << "\t   E-Voting System \t\n"
393      << "\t ***************************** \t\n";
394
395      while(true){
396          cout << "Enter your CNIC: ";
397          cin >> id;
398          cout << endl;
399
400          registered = voter(id);
401          voted = vote_status(id);
402
403          if(voted == false && registered == true){
404              cout << "\t ***************************** \t\n"
405              << "\t   Please cast your vote for NA \t\n"
406              << "\t ***************************** \t\n";
407              cout << "\na.PTI\n" << "b.PML(N)\n" << "c.PPP\n" << "d.MMA\n" << "e.TLP\n" << "f.Independent" << endl;
408              cin >> partyselect;
409              if(partyselect == 'a'){
410                  party = "PTI";
411                  partyvote[0]++;
412                  record_vote_n(party, partyvote[0]);
413              }
414              else if (partyselect == 'b')
415              {
416                  party = "PMLN";
417                  partyvote[1]++;
418                  record_vote_n(party, partyvote[1]);
```

```
418             record_vote_n(party, partyvote[1]);
419         }
420         else if (partyselect == 'c')
421         {
422             party = "PPP";
423             partyvote[2]++;
424             record_vote_n(party, partyvote[2]);
425         }
426         else if (partyselect == 'd'){
427             party = "MMA";
428             partyvote[3]++;
429             record_vote_n(party, partyvote[3]);
430         }
431         else if (partyselect == 'e')
432         {
433             party = "TLP";
434             partyvote[4]++;
435             record_vote_n(party, partyvote[4]);
436         }
437         else if (partyselect == 'f'){
438             party = "Indie";
439             partyvote[5]++;
440             record_vote_n(party, partyvote[5]);
441         }
442         else{
443             cout << "invalid input\n";
444         }
445         foutn << id << endl;
446         cout << "Voted successfully.\n";
447 
448         cout << "\t ***************************** \t\n"
449         << "\t   Please cast your vote for PA \t\n"
450         << "\t ***************************** \t\n";
451         cout << "\na.PTI\n" << "b.PML(N)\n" << "c.PPP\n" << "d.MMA\n" << "e.TLP\n" << "f.Independent" << endl;
452         cin >> partyselect;
453         if(partyselect == 'a'){
454             party = "PTI";
455             pvote[0]++;
456             record_vote_p(party, pvote[0]);
457         }
```

```
459             {
460                 party = "PMLN";
461                 pvote[1]++;
462                 record_vote_p(party, pvote[1]);
463 
464             }
465             else if (partyselect == 'c')
466             {
467                 party = "PPP";
468                 pvote[2]++;
469                 record_vote_p(party, pvote[2]);
470             }
471             else if (partyselect == 'd'){
472                 party = "MMA";
473                 pvote[3]++;
474                 record_vote_p(party, pvote[3]);
475             }
476             else if (partyselect == 'e')
477             {
478                 party = "TLP";
479                 pvote[4]++;
480                 record_vote_p(party, pvote[4]);
481             }
482             else if (partyselect == 'f'){
483                 party = "Indie";
484                 pvote[5]++;
485                 record_vote_p(party, pvote[5]);
486             }
487             else{
488                 cout << "invalid input\n";
489             }
490             foutp << id << endl;
491             cout << "Voted successfully.\n";
492         }
493 
494         cout << "\nProceed to other voter.(Y/N)\n";
495         cin >> op;
496         if(op == 'n' || op == 'N'){
497             cout << "Do you want to end the election and display election results.(Y/N)\n";
498             cin >> admin;
499             if(admin == 'Y' || admin == 'y'){
500             cout << "Please enter the admin. password to gain administrative access.\n";
501             cin >> adminpw;
502                 if(adminpw == "1234e")
```

```
487             else{
488                 cout << "invalid input\n";
489             }
490             foutp << id << endl;
491             cout << "Voted successfully.\n";
492         }
493 
494         cout << "\nProceed to other voter.(Y/N)\n";
495         cin >> op;
496         if(op == 'n' || op == 'N'){
497             cout << "Do you want to end the election and display election results.(Y/N)\n";
498             cin >> admin;
499             if(admin == 'Y' || admin == 'y'){
500             cout << "Please enter the admin. password to gain administrative access.\n";
501             cin >> adminpw;
502                 if(adminpw == "1234e")
503                 {
504                     na_displayresults();
505                     pa_displayresults();
506                 }
507                 else{
508                     cout << "Wrong Password\n";
509                 }
510             }
511             break;
512         }
513     }
514 
515 
516     foutn.close();
517     foutp.close();
518 
519     return 0;
520 }
```

Program in action

C:\Users\Lenovo\Documents\e-voting\election2.exe

```
         ******************************
           E-Voting System
         ******************************
Enter your CNIC: 2500

Error: CNIC is not registered.

Proceed to other voter.(Y/N)
y
Enter your CNIC: 5500

Error: CNIC is not registered.

Proceed to other voter.(Y/N)
y
Enter your CNIC: 4500

Welcome: sohaib

         ******************************
           Please cast your vote for NA
         ******************************


a.PTI
b.PML(N)
c.PPP
d.MMA
e.TLP
f.Independent
b
Voted successfully.
         ******************************
           Please cast your vote for PA
         ******************************


a.PTI
b.PML(N)
c.PPP
d.MMA
e.TLP
f.Independent
a
Voted successfully.

Proceed to other voter.(Y/N)
n
Do you want to end the election and display election results.(Y/N)
n

--------------------------------
```

voted - Notepad

File Edit Format View Help

```
0000
2500
1234
```

C:\Users\Lenovo\Documents\e-voting\election2.exe

```
         ******************************
           E-Voting System
         ******************************
Enter your CNIC: 1234

Welcome: ali

         ******************************
           Please cast your vote for NA
         ******************************


a.PTI
b.PML(N)
c.PPP
d.MMA
e.TLP
f.Independent
a
Voted successfully.
         ******************************
           Please cast your vote for PA
         ******************************


a.PTI
b.PML(N)
c.PPP
d.MMA
e.TLP
f.Independent
b
Voted successfully.

Proceed to other voter.(Y/N)
n
Do you want to end the election and display election results.(Y/N)
y
Please enter the admin. password to gain administrative access.
1234e
         ******************************
           NA Election Results
         ******************************
PTI: 1
PMLN: 0
PPP: 0
MMA: 0
TLP: 0
Indie: 0
Indie: 0
         ******************************
           PA Election Results
         ******************************
PTI: 0
PMLN: 1
PPP: 0
MMA: 0
TLP: 0
Indie: 0
Indie: 0

--------------------------------
Process exited after 43.9 seconds with return value 0
Press any key to continue . . .
```

pa_results - Notepad

File Edit Format View Help

```
PTI,0
PMLN,1
PPP,0
MMA,0
TLP,0
Indie,0
```

na_results - Notepad

File Edit Format View Help

```
PTI,1
PMLN,0
PPP,0
MMA,0
TLP,0
Indie,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.

Thank

You!