A Project on

“ Digital Voting System (D.V.S) ”

November 2022

Department of Computer Science & Engineering

By:
Preetkamal Singh Sandhu

Index

Digital Voting System (D.V.S)

1. Introduction

This project aims to build a Digital Voting System, which is capable of managing and handling the voter, candidate and voting data in an efficient and better way. It provides a user-friendly and paperless way to cast votes in a poll and access them whenever required, without much human effort. The project is capable of storing new voter, candidate, and poll data, and to update the stored data as and when required. It offers admin access, and the admin is only given the authority to manipulate these data elements, thus adding the much-needed safety and security features to it, while a normal user is capable of casting the vote to his desired candidate.

2. OOP Concepts we have covered

Various concepts of OOP used in this project and studied in the syllabus course are as follows:

1. **Inheritance** :
   Inheritance is the main backbone of this project, where different classes for voter, candidate, etc. have been formulated. Various data objects of such classes can be accessed by the admin class using inheritance.

2. **Polymorphism** :
   Polymorphism is an important concept of OOP and thus it has effectively being implemented in this project. Some of the types of polymorphism used in the project are:

   - Function Overloading
   - Function Overriding
   - Operator Overloading

3. **Abstract Classes and Virtual Functions** :
   The concept of abstract classes and virtual / pure virtual functions has been implemented in order to give different definitions to functions of different classes having similar kind of working in order to make the code simpler and clearer.

4. **Friend Functions** :
   Friend functions are used here to provide the private information of some classes (such as Admin username and password) to some of the functions which require these values to operate.

5. **Constructors and Destructors :**
   Constructors and Destructors are used to initialize data values and to remove the objects once necessary changes have been made to the data.

6. **File Handling** :
   File handling is the backbone of the data storage in this project, used for storing all the candidate, voter, poll data and saving the results of the polls, so that these data elements can be accessed whenever required in the future.

7. **Vectors** :
   Vectors are a special kind of data storage element, similar to arrays, in which we can store data of any type without mentioning the number of elements. We may add more and more elements whenever required. In this project, vectors have been used to effortlessly retrieve the data stored in various files without the need to specify the size of data, as in case of normal arrays. Both 1-D and 2-D vectors have been implemented.

   Some other concepts used related to OOPs and C++ used are as follows:

   - **Inline Function**
   - **Default Arguments in functions**
   - **Console and Text colouring** (Foreground and Background): With the help of console handle in windows using Windows' header file, the console's output colours have been changed to give the project a more graphical and user friendly interface.

   Some other small concepts related to OOP have also been implemented in the project, wherever found helpful and handy.

3. Merits

Some of the merits possessed by this project are:

- It makes the voting process a much simpler and an effortless job.
- Proper security means with the help of admin user and password ensures only an authorized access to the data.
- If wrong admin credentials are entered for more than 5 times, then the software automatically gets terminated, making it more secure.
- If there is an interruption while the polling process is still going on, then it can be again restarted from the point it was left over, without any loss in the data.
- The voting process can be stopped only with a valid admin username and password.
- In the results, there is an option which shows the total percentage of voters who have casted their votes. Thus, it helps to get an idea about the percentage voting that took place for that poll.
- The candidate to whom a voter has casted the vote is never stored anywhere at any point of time, thus helping to maintain adequate privacy for the voter.

4. Demerits :

- Adding the candidate and the voter data for the first time may become a cumbersome task for the admin personnel.
- There may be a chance of stored data files being stolen by some unauthorized party. This problem can be easily handled by applying various encryption techniques.

5. Code and snapshots

```
#include <iostream>
#include <fstream>
#include <sstream>
#include <vector>
#include <conio.h>
#include <windows.h>
#include <math.h>
using namespace std;

class DataHandle{
   protected:
      virtual void GetData()=0;
      virtual void AddData()=0;
};

class Candidate:protected DataHandle{
   private:
      char CandidateID[5],CandidateName[100],PartyName[100],PartySymbol[30];
   protected:
      void GetData();
      void AddData();
      void UpdateData();
      ~Candidate(){}
};

class Voter:protected DataHandle{
   private:
      char VoterID[10],VoterName[150],DOB[10],Gender[5];
   protected:
      void GetData();
      void AddData();
      void UpdateData();
      ~Voter(){}
};

class Poll:protected DataHandle{
   private:
      char PollID[5],PollName[30],PollDate[11],Status[10];
   protected:
      void GetData();
      void AddData();
```

```
        ~Poll(){}
};

class Admin:protected Candidate,protected Voter,protected Poll{
   private:
      string AdminID,AdminPass;
      void SetPass(){
         AdminID="A";
         AdminPass="P";
      }
      friend void ExitPolling(char[],char[]);

   public:
      Admin(){
         SetPass();
      }
      void AdminPage();
      bool login();
};

vector<string> ReadRecord(char[],char[],int);

void UpdateRecord(char[],char[],int,char[],int);

int StrToInt(string str){
   int num = 0;
   for(int i = 0; i < str.length(); i++){
      int digit = int(str[i])-'0';
      int mul = pow(10,str.length()-i-1);
      num+= digit*mul;
   }
   return num;
}

void colour(string s,int code){
   SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), code);
   cout<<s;
   SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 15);
}

void bgcolour(string s,int code1,int code2){
   SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), code1|code2 );
   cout<<s;
   SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 15 );
}

class VoteCast:protected DataHandle{
   private:
      char PollFileName[100];
      char PollID[5];
```

```
        char VoterID[10];
        char CandID[10];
        int VoteCount;
        bool Voted;
        char VoterPass[15];
        vector < vector<string> > Data;
        vector<string>PollRecord ;
        char filename[20],VoterFile[20],TempFile[30];
        int exit;

        friend void TakePoll(char[], char[]);
        friend void ExitPolling(char[], char[]);

        void operator ++(){
           vector<string> Data = ReadRecord(PollFileName,CandID,2);
           int CurrentVotes=StrToInt(Data[1]);
           stringstream UpdatedVotes;
           UpdatedVotes<<(CurrentVotes+1);
           char NewVotes[10];
           strcpy(NewVotes,UpdatedVotes.str().c_str());
           UpdateRecord(PollFileName,CandID,1,NewVotes,2);
        }

    public:
        VoteCast(char File[], char PId[]);
        void GetData();
        int Exit();
        void AddData();
        ~VoteCast(){};
};

Admin DVSOfficer;

inline void cls(){
    system("cls");
}

bool CheckRecord(char FileName[],char ID[],int Cols=4){

    ifstream file;
    file.open(FileName);
    bool Found=false;
    string Col[Cols];

    while(getline(file,Col[0],',') && !Found){
        int i;
        for(i=1;i<(Cols-1);i++){
            getline(file,Col[i],',');
        }
        getline(file,Col[Cols-1],'\n');
```

```
        if(Col[0]==ID){
            Found=true;
        }
    }
    file.close();
    return Found;
}

vector < vector<string> > ReadFile(char FileName[]){
    vector< vector<string> > content;
    vector<string> row;
    string line, word;
    fstream file (FileName, ios::in);

    while(getline(file, line)){
        row.clear();
        stringstream str(line);

        while(getline(str, word, ','))
            row.push_back(word);
            content.push_back(row);
    }
    return content;
}

vector<string> ReadRecord(char FileName[],char ID[],int Cols=4){

    ifstream file;
    file.open(FileName);
    vector<string>Record;
    bool Found=false;
    string Col[Cols];

    while(getline(file,Col[0],',') && !Found){
        int i;

        for(i=1;i<(Cols-1);i++){
            getline(file,Col[i],',');
        }
        getline(file,Col[Cols-1],'\n');

        if(Col[0]==ID){
            Found=true;
            for(i=0;i<Cols;i++){
                Record.push_back(Col[i]);
            }
        }
    }
    file.close();
    return Record;
```

```
}

void UpdateRecord(char FileName[], char ID[], int Index,char New[], int Cols=4){

   fstream fin, fout;
   fin.open(FileName,ios::in);
   fout.open("Data/Temp.csv",ios::out);
   string Row,Col;
   int i;
   vector<string> Record;

   while(!fin.eof()){
      Record.clear();
      getline(fin,Row);
      stringstream s(Row);

      while(getline(s, Col, ',')){
         Record.push_back(Col);
      }
      if(Record[0] == ID){
         Record[Index]=New;

         if(!fin.eof()){
            for(i=0;i<(Cols-1);i++){
               fout<<Record[i]<<",";
            }
            fout<<Record[Cols-1]<<endl;
         }
      }
      else{
         if(!fin.eof()){
            for(i=0;i<(Cols-1);i++){
               fout<<Record[i]<<",";
            }
            fout<<Record[Cols-1]<<endl;
         }
      }
      if(fin.eof()){
         break;
      }
   }
   fin.close();
   fout.close();
   remove(FileName);
   rename("Data/Temp.csv",FileName);
}

void ShowCandidates(char PollFileName[],char PollID[]){
   int i;
   vector < vector<string> > Data = ReadFile(PollFileName);
```

```
    for(i=0;i<Data.size();i++){
        char CandFile[]="Data/Candidate.csv";
        char* IdData;
        IdData = &Data[i][0][0];
        vector<string> CandData = ReadRecord(CandFile,IdData);
        cout<<CandData[0];
        Data[i][0];
        cout<<"\n";
    }
}

void HomePage();
void TakePoll(char[],char[]);

void ExitPolling(char PollFileName[],char PollID[]){
    static int trial;
    string admin,password;
    int i;
    char pass[30];
    bgcolour("\n\t\t EXIT POLLING \n\n",15,0x0040);
    colour("\t Admin User: ",6);
    cin>>admin;
    colour("\t Admin Password: ",6);

    while((pass[i]=getch())!=13){
        cout<<"*";
        i++;
    }
    pass[i]='\0';
    i=0;
    while(pass[i]!='\0'){
        password += pass[i];
        i++;
    }

    if(admin==DVSOfficer.AdminID && password==DVSOfficer.AdminPass){
        cls();
        char filename[]="Data/Polls.csv";
        char Status[]="Conducted";
        UpdateRecord(filename,PollID,3,Status);
        remove("Data/Polls/Temp.csv");
        bgcolour("\n\n\n\n\n\t\t\t\t POLL CONDUCTED SUCCESSFULLY ",15,0x0020);
        colour("\n\n\t\t\t\t-> Press Any Key To Continue:",10);
        getch();
        cls();
        HomePage();
    }
    else{
        cls();
```

```
        bgcolour("\n\t\t EXIT POLLING \n\n",15,0x0040);
        colour("\t  UNAUTHORIZED ACCESS !!\n\n ",4);
        trial++;
        if(trial==5){
           colour("\tYou Have Made 5 Invalid Attempts.\n",14);
           colour("\n\tThe Program is being Terminated for Security Purpose.",14);
           exit(0);
        }
        colour("\t\t -> Press ENTER to Retry\n",6);
        colour("\t\t -> Any Other Key to Continue Polling\n",6);
        colour("\n\t\t\t-> Press Key:",10);
        int choice=getch();
        if (choice==13){
           cls();
           ExitPolling(PollFileName, PollID);
        }
        else{
           cls();
           TakePoll(PollFileName, PollID);
        }
     }
}

void TakePoll(char PollFileName[], char PollID[]){
   VoteCast Vote(PollFileName,PollID);

   while(Vote.exit==0){
      Vote.VoteCast::GetData();
      Vote.Exit();
      if(Vote.exit){
         cls();
         Vote.~VoteCast();
         ExitPolling(PollFileName,PollID);
      }
      else{
         Vote.VoteCast::AddData();
         if(Vote.VoteCount==1){
            ++Vote;
            Vote.~VoteCast();
            cls();
            TakePoll(PollFileName,PollID);
         }
      }
   }
}

void Candidate::GetData(){
   colour("\t Candidate ID: ",14);
   gets(CandidateID);
```

```
    colour("\n\t Candidate Name: ",14);
    gets(CandidateName);

    colour("\n\t Party Name: ",14);
    gets(PartyName);

    colour("\n\t Party Symbol: ",14);
    gets(PartySymbol);
}

void Candidate::AddData(){
    char filename[]="Data/Candidate.csv";

    if(!CheckRecord(filename,CandidateID)){
        ofstream file;
        file.open(filename,ios_base::app);
        file<<CandidateID<<","<<CandidateName<<","<<PartyName<<","<<PartySymbol<<e
ndl;
        file.close();
        bgcolour("\n\n\t\t Candidate Enrolled SUCCESSFULLY ",15,0x0020);
    }
    else{
        bgcolour("\n\n\t\t Candidate ID ALREADY EXISTS !! ",15,0x0040);
    }
}

void Candidate::UpdateData(){
    char filename[]="Data/Candidate.csv";
    colour("\t Enter Candidate ID: ",14);
    gets(CandidateID);
    cls();
    bgcolour("\n\t\t CANDIDATE DATA UPDATION \n\n",15,0x0040);

    if(CheckRecord(filename,CandidateID)){
        vector<string>CandidateRecord = ReadRecord(filename,CandidateID);
        colour("\tCandidate Details: \n",6);
        colour("\n\t\tCandidate Name: ",14);cout<<CandidateRecord[1];
        colour("\n\t\tParty Name: ",14);cout<<CandidateRecord[2];
        colour("\n\t\tParty Symbol: ",14);cout<<CandidateRecord[3];
        int Index;
        colour("\n\n\tChoose a Field to Update:\n\n ",6);
        colour("\t\t1. Update Candidate Name\n",14);
        colour("\t\t2. Update Party Name\n",14);
        colour("\t\t3. Update Party Symbol\n",14);
        colour("\n\t\t-> Any Other Key to Exit to Main Menu",14);
        colour("\n\n\t\t\t Enter Choice: ",10);
        cin>>Index;
        cls();
        char NewData[100];
        bgcolour("\n\t\t CANDIDATE DATA UPDATION \n\n",15,0x0040);
```

```
        switch(Index){
            case 1:{
                colour("\tCurrent Candidate Name: ",14);cout<<CandidateRecord[1];
                cin.ignore();
                colour("\n\n\tEnter Updated Name: ",14);
                gets(NewData);
                UpdateRecord(filename,CandidateID,Index,NewData);
                break;
            }
            case 2:{
                colour("\tCurrent Party Name: ",14);cout<<CandidateRecord[2];
                cin.ignore();
                colour("\n\n\tEnter Updated Name: ",14);
                gets(NewData);
                UpdateRecord(filename,CandidateID,Index,NewData);
                break;
            }
            case 3:{
                colour("\tCurrent Party Symbol: ",14);cout<<CandidateRecord[3];
                cin.ignore();
                colour("\n\n\tEnter Updated Symbol: ",14);
                gets(NewData);
                UpdateRecord(filename,CandidateID,Index,NewData);
                break;
            }
            default:{
                cls();
                DVSOfficer.AdminPage();
                break;
            }
        }
        cls();
        bgcolour("\n\n\t\t DATA UPDATED !! ",15,0x0020);
    }
    else{
        cls();
        bgcolour("\n\n\t\t INVALID Candidate ID !! ",15,0x0040);
    }
}

void Voter::GetData(){
    colour("\t Voter ID: ",14);
    gets(VoterID);

    colour("\n\t Voter Name: ",14);
    gets(VoterName);

    colour("\n\t Voter Gender: ",14);
    gets(Gender);
```

```
    colour("\n\t DOB: ",14);
    gets(DOB);
}

void Voter::AddData(){
    char filename[]="Data/Voter.csv";

    if(!CheckRecord(filename,VoterID)){
        ofstream file;
        file.open(filename,ios_base::app);
        file<<VoterID<<","<<VoterName<<","<<Gender<<","<<DOB<<endl;
        file.close();
        bgcolour("\n\n\t\t Voter Enrolled SUCCESSFULLY ",15,0x0020);
    }
    else{
        bgcolour("\n\n\t\t Voter ID ALREADY EXISTS !!",15,0x0040);
    }
}

void Voter::UpdateData(){
    char filename[]="Data/Voter.csv";
    colour("\t Enter Voter ID: ",14);
    gets(VoterID);
    cls();
    bgcolour("\n\t\t VOTER DATA UPDATION \n\n",15,0x0040);

    if(CheckRecord(filename,VoterID)){
        vector<string>VoterRecord = ReadRecord(filename,VoterID);
        colour("\tVoter Details: \n",6);
        colour("\n\t\tVoter Name: ",14);cout<<VoterRecord[1];
        colour("\n\t\tGender: ",14);cout<<VoterRecord[2];
        colour("\n\t\tDOB: ",14);cout<<VoterRecord[3];
        int Index;
        colour("\n\n\tChoose a Field to Update:\n\n ",6);
        colour("\t\t1. Update Voter Name\n",14);
        colour("\t\t2. Update Gender\n",14);
        colour("\t\t3. Update DOB\n",14);
        colour("\n\t\t-> Any Other Key to Exit to Main Menu",14);
        colour("\n\n\t\t\t Enter Choice: ",10);
        cin>>Index;
        cls();
        char NewData[100];
        bgcolour("\n\t\t Voter DATA UPDATION \n\n",15,0x0040);

        switch(Index){
            case 1:{
                colour("\tCurrent Voter Name: ",14);cout<<VoterRecord[1];
                cin.ignore();
                colour("\n\n\tEnter Updated Name: ",14);
```

```
            gets(NewData);
            UpdateRecord(filename,VoterID,Index,NewData);
            break;
        }
        case 2:{
            colour("\tCurrent Gender: ",14);cout<<VoterRecord[2];
            cin.ignore();
            colour("\n\n\tEnter Updated Gender: ",14);
            gets(NewData);
            UpdateRecord(filename,VoterID,Index,NewData);
            break;
        }
        case 3:{
            colour("\tCurrent DOB: ",14);cout<<VoterRecord[3];
            cin.ignore();
            colour("\n\n\tEnter Updated DOB: ",14);
            gets(NewData);
            UpdateRecord(filename,VoterID,Index,NewData);
            break;
        }
        default:{
            cls();
            DVSOfficer.AdminPage();
            break;
        }
    }
    cls();
    bgcolour("\n\n\t\t DATA UPDATED !! ",15,0x0020);
  }
  else{
    cls();
    bgcolour("\n\n\t\t INVALID Voter ID !! ",15,0x0040);
  }
}

void Poll::GetData(){
  colour("\t Poll ID: ",14);
  gets(PollID);

  colour("\n\t Poll Name: ",14);
  gets(PollName);

  colour("\n\t Poll Date: ",14);
  gets(PollDate);
}

void Poll::AddData(){
  char filename[] = "Data/Polls.csv";

  if(!CheckRecord(filename,PollID)){
```

```
char PollFileName[]="Data/Polls/";
char Str2[]=".csv";
strcat(PollFileName,PollID);
strcat(PollFileName,Str2);

ofstream file;
file.open("Data/Polls.csv",ios_base::app);
file<<PollID<<","<<PollName<<","<<PollDate<<","<<"Upcoming"<<endl;
file.close();

fstream PollFile;
PollFile.open(PollFileName, ios::out | ios::app);
PollFile<<"CandidateID"<<","<<"Votes"<<endl;
PollFile.close();
bool Add=true;

while(Add){
   char CandID[5],CandFile[]="Data/Candidate.csv";
   bgcolour("\n\t\t ADD CANDIDATES FOR THIS POLL \n\n",15,0x0014);
   colour("Enter Candidate ID: ",14);
   gets(CandID);

   if((CheckRecord(CandFile,CandID))&&(!CheckRecord(PollFileName,CandID,2))){
      vector<string> Record=ReadRecord(CandFile,CandID);
      colour("\n\t Candidate Name: ",14);cout<<Record[1];
      colour("\n\t Candidate Party: ",14);cout<<Record[2];
      colour("\n\t Party Symbol: ",14);cout<<Record[3];
      colour("\n\n\t\t -> Press Any Key: ",10);
      getch();
      cls();
      ofstream PFile;
      PFile.open(PollFileName,ios_base::app);
      PFile<<CandID<<","<<"0"<<endl;
      PFile.close();
      bgcolour("\n\n\t\t Candidate Added Successfully ",15,0x0020);
   }
   else if(CheckRecord(PollFileName,CandID,2)){
      bgcolour("\n\n\t Candidate ALREADY ADDED !! ",15,0x0040);
   }
   else{
      bgcolour("\n\n\t Candidate ID INVALID !! ",15,0x0040);
   }
   colour("\n\n\t -> Press Enter to Add New Candidate",6);
   colour("\n\t -> Any Other Key to Finish Adding Candidates",6);
   colour("\n\n\t\t Press Key: ",10);
   int choice=getch();
   if(choice==13){
      cls();
   }
   else{
```

```
            Add=false;
          }
        }
        cls();
        bgcolour("\n\n\t\t Poll Created SUCCESSFULLY ",1,0x0020);
    }
    else{
        bgcolour("\n\n\t\t Poll ID ALREADY EXISTS !! ",15,0x0040);
    }
}

bool Admin::login(){
    static int trial;
    string admin,password;
    int i;
    char pass[30];
    bgcolour("\n\t\t DVS OFFICER LOGIN PAGE \n\n\n",15,0x0040);
    colour("\t Admin User: ",14);
    cin>>admin;
    colour(" \n\t Admin Password: ",14);

    while((pass[i]=getch())!=13){
        cout<<"*";
        i++;
    }
    pass[i]='\0';
    i=0;
    while(pass[i]!='\0'){
        password += pass[i];
        i++;
    }

    if(admin==AdminID && password==AdminPass){
        cls();
        trial=0;
        return true;
    }
    else{
        cls();
        colour("Unauthorized Access !!\n\n ",4);
        trial++;
        if(trial==5){
            colour("\n\tYou Have Made 5 Invalid Attempts.\n",14);
            colour("\n\tThe Program is being Terminated for Security Purpose.",14);
            exit(0);
        }
        login();
    }
}
```

```
VoteCast::VoteCast(char File[], char PId[]){
   strcpy(PollFileName,File);
   strcpy(PollID,PId);
   exit=0;
   VoteCount=0;
   Voted=false;
   strcpy(filename,"Data/Polls.csv");
   strcpy(VoterFile,"Data/Voter.csv");
   strcpy(TempFile,"Data/Polls/Temp.csv");
   Data = ReadFile(PollFileName);
}

void Header(char PollID[5],string PollName, string PollDate){
   SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 15|0x0030 );
   cout<<"\n\n\t\t  WELCOME TO "<<PollName<<"  ";
   SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 15);
   colour("\n\n \tPoll ID: ",14);cout<<PollID;colour("\t Poll Date: ",14);cout<<PollDate;
}

void VoteCast::GetData(){
   PollRecord = ReadRecord(filename,PollID);
   Header(PollID,PollRecord[1],PollRecord[2]);
   colour("\n\n\n\t\t -> Enter Voter ID: ",10);
   cin>>VoterID;
   colour("\n\t\t -> Enter Voter Password: ",10);
   int i=0;
   while((VoterPass[i]=getch())!=13){
      cout<<"*";
      i++;
   }
   VoterPass[i]='\0';
}

int VoteCast::Exit(){
   if(strcmp(VoterID,"EXIT")==0 or strcmp(VoterID,"exit")==0){
      exit=1;
      return exit;
   }
}

void VoteCast::AddData(){
   vector<string> PassData=ReadRecord(VoterFile,VoterID);
   string VoterPassword;
   int i=0;
   while(VoterPass[i]!='\0'){
      VoterPassword += VoterPass[i];
      i++;
   }
   int count=1;
```

```
   if(CheckRecord(VoterFile,VoterID) && !CheckRecord(TempFile,VoterID,2) &&
(VoterPassword==PassData[3])){
        cls();
        char CandFile[]="Data/Candidate.csv";
        while(!Voted){
           int i=1,rows=0;
           cls();
           ifstream file(PollFileName);
           string row;
           while(getline(file,row)){
              rows++;
           }
           file.close();
           Header(PollID,PollRecord[1],PollRecord[2]);
           colour("\n\n \t\t > CANDIDATE LIST: \n\n",6);

           while(i<rows){
              string data=Data[i][0];
              char ID[data.length()+1];
              strcpy(ID,data.c_str());
              vector<string> CandRecord = ReadRecord(CandFile,ID);
              colour("\t\tCandidate ID: ",14);cout<<CandRecord[0];
              colour("\n\t\tCandidate Name: ",14);cout<<CandRecord[1];
              colour("\n\t\tParty Name: ",14);cout<<CandRecord[2];
              colour("\n\t\tParty Symbol: ",14);cout<<CandRecord[3];
              colour("\n\t\t----------------------------\n",3);
              i++;
           }
           if(count==1){
              cin.ignore();
           }
           colour("\n\n\t-> Enter Candidate ID to Vote: ",10);
           gets(CandID);
           count=0;
           if(CheckRecord(PollFileName,CandID,2)){
              cls();
              Header(PollID,PollRecord[1],PollRecord[2]);
              colour("\n\n\tVoter Name: ",6);cout<<PassData[1];
              colour("\n\n\t\tYou Chose the following Candidate: ",6);
              vector<string> CRecord = ReadRecord(CandFile,CandID);
              colour("\n\n\t\tCandidate ID: ",14);cout<<CRecord[0];
              colour("\n\t\tCandidate Name: ",14);cout<<CRecord[1];
              colour("\n\t\tParty Name: ",14);cout<<CRecord[2];
              colour("\n\t\tParty Symbol: ",14);cout<<CRecord[3];
              colour("\n\t\t----------------------------\n",3);
              colour("\n\n\t\t-> Press Enter to Confirm",6);
              colour("\n\t\t-> Any Other Key to Change Choice",6);
              colour("\n\n\t\t\t Press Key:",10);

              int choice=getch();
```

```
            if(choice==13){
               cls();
               Header(PollID,PollRecord[1],PollRecord[2]);
               VoteCount=1;
               bgcolour("\n\n\n\t Your Vote Has Been Added Successfully ",15,0x0020);
               colour("\n\n\t\t-> Press Any Key to Continue:",6);
               ofstream temp;
               temp.open(TempFile,ios_base::app);
               temp<<VoterID<<","<<""<<endl;
               temp.close();
               getch();
               cls();
               Voted=true;
            }
            else{
            }

         }
         else{
            cls();
            Header(PollID,PollRecord[1],PollRecord[2]);
            bgcolour("\n\n\n\t\t INVALID CANDIDATE ID !! ",15,0x0040);
            colour("\n\n\t\t -> Press Any Key to Continue:",10);
            getch();
         }
      }
   }
   else if(CheckRecord(TempFile,VoterID,2) && (VoterPassword==PassData[3])){
      cls();
      bgcolour(" You Have Already Voted !! \n",4,0x0060);
   }
   else{
      cls();
      bgcolour(" Invalid Voter ID/Password !! \n",4,0x0060);
   }
}

void HomePage();
int main();

void Admin::AdminPage(){
   int choice;
   bgcolour("\n\t\t DVS OFFICER ADMIN PAGE \n\n",15,0x0040);
   colour("\t1. Enroll New Candidates \n",6);
   colour("\t2. Register New Voters \n",6);
   colour("\t3. Update Candidate/Voter Data \n",6);
   colour("\t4. Create New Poll \n",6);
   colour("\t5. Start a Poll \n",6);
   colour("\t6. Access A Previous Poll \n",6);
   colour("\t7. Logout Admin Page",6);
```

```
colour("\n\n\t-> Enter Your Choice: ",2);
cin>>choice;

switch(choice){
   case 1:{
      cls();
      bool Add=true;
      cin.ignore();
      while(Add){
         bgcolour("\n\t\t NEW CANDIDATE ENROLLMENT \n\n",15,0x0040);

         Admin C;
         C.Candidate::GetData();
         cls();
         C.Candidate::AddData();
         C.~Candidate();

         colour("\n\n\t-> Press Enter to Enroll New Candidate\n",14);
         colour("\t-> Any Other Key to Exit to Main Menu\n\n",14);
         colour("\t\t-> Press Key:",10);
         int choice=getch();
         if(choice==13){
            cls();
         }
         else{
            Add=false;
         }
      }
      cls();
      AdminPage();
      break;
   }

   case 2:{
      cls();
      bool Add=true;
      cin.ignore();
      while(Add){
         bgcolour("\n\t\t NEW VOTER ENROLLMENT \n\n",15,0x0040);

         Admin V;
         V.Voter::GetData();
         cls();
         V.Voter::AddData();
         V.~Voter();

         colour("\n\n\t-> Press Enter to Enroll New Voter\n",14);
         colour("\t-> Any Other Key to Exit to Main Menu\n\n",14);
         colour("\t\t Press Key:",10);
         int choice=getch();
```

```
        if(choice==13){
            cls();
        }
        else{
            Add=false;
        }
    }
    cls();
    AdminPage();
    break;
}

case 3:{
    cls();
    bgcolour("\n\t\t DATA UPDATION SYSTEM \n\n",15,0x0040);
    colour("\t-> Press 1 to Update Candidate Data\n",14);
    colour("\t-> Press 2 to Update Voter Data\n",14);
    colour("\n\t-> Any Other Key to Exit to Main Menu\n\n",6);
    colour("\t\t Press Key:",10);
    int choice=getch();
    if(choice==49){
        bool Update=true;
        cin.ignore();
        while(Update){
            cls();
            bgcolour("\n\t\t CANDIDATE DATA UPDATION \n\n",15,0x0040);

            Admin C;
            C.Candidate::UpdateData();
            C.~Candidate();

            colour("\n\n\t-> Press Enter to Update Another Record\n",14);
            colour("\t-> Any Other Key to Exit to Main Menu\n\n",14);
            colour("\t\t Press Key:",10);
            int ch=getch();
            if(ch==13){
                cls();
            }
            else{
                Update=false;
            }
        }
        cls();
        AdminPage();
    }
    else if(choice==50){
        bool Update=true;
        cin.ignore();
        while(Update){
            cls();
```

```
            bgcolour("\n\t\t VOTER DATA UPDATION \n\n",15,0x0040);

            Admin V;
            V.Voter::UpdateData();
            V.~Voter();

            colour("\n\n\t-> Press Enter to Update Another Record\n",14);
            colour("\t-> Any Other Key to Exit to Main Menu\n\n",14);
            colour("\t\t Press Key:",10);
            int ch=getch();
            if(ch==13){
               cls();
            }
            else{
               Update=false;
            }
         }
         cls();
         AdminPage();
      }
      else{
         cls();
         AdminPage();
      }
      break;
   }

   case 4:{
      cls();
      bool Add=true;
      cin.ignore();
      while(Add){
         bgcolour("\n\t\t NEW POLL CREATION \n\n",15,0x0040);
         Admin P;

         P.Poll::GetData();
         cls();
         P.Poll::AddData();
         P.~Poll();

         colour("\n\n\t-> Press Enter to Create New Poll\n",14);
         colour("\t-> Any Other Key to Exit to Main Menu\n\n",14);
         colour("\t\t Press Key:",10);
         int choice=getch();
         if(choice==13){
            cls();
         }
         else{
            Add=false;
         }
```

```
    }
    cls();
    AdminPage();
    break;
}

case 5:{
    bool check=true;
    cls();
    cin.ignore();
    while(check){
        char ID[5];
        bgcolour("\n\t\t START A NEW POLL \n\n",15,0x0040);
        colour("Enter Poll ID: ",14);
        gets(ID);
        char PollFileName[]="Data/Polls/";
        char Str2[]=".csv";
        char PID[]="";
        strcat(PID,ID);
        char filename[]="Data/Polls.csv";

        vector<string>PollInfo = ReadRecord(filename,ID);

        if(CheckRecord(filename,ID)){
            cls();
            strcat(PollFileName,ID);
            strcat(PollFileName,Str2);
            check=false;
            bgcolour("\n\t\t POLL DETAILS \n\n",15,0x0040);
            colour("\t POLL ID: ",14);cout<<PollInfo[0];
            colour("\n\t POLL NAME: ",14);cout<<PollInfo[1];;
            colour("\n\t POLL DATE: ",14);cout<<PollInfo[2];;
            colour("\n\t STATUS: ",14);cout<<PollInfo[3];

            if(PollInfo[3]=="Upcoming"){
                string start;
                colour("\n\n\t-> Enter 'Y' to Start Poll\n",6);
                colour("\t-> Any Other Key to Exit\n\n",6);
                colour("\t\tEnter Choice: ",10);
                cin>>start;
                if(start=="Y" or start=="y"){
                    cls();
                    char str[]=".csv";
                    strcat(PollFileName,str);
                    TakePoll(PollFileName,PID);
                }
                else{
                    cls();
                    AdminPage();
                }
```

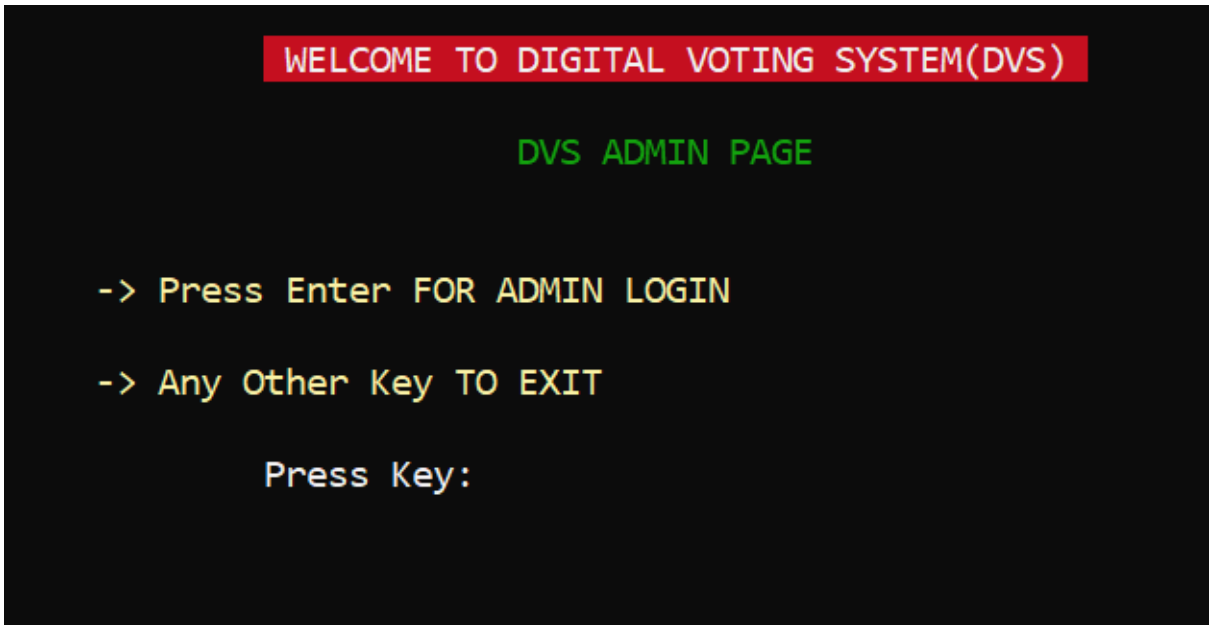
```
            }
            else{
                bgcolour("\n\n\t This Poll Is Already Taken !! \n",4,0x0060);
                colour("\n\t\t-> Press Any Key to Continue:",10);
                getch();
                cls();
                AdminPage();
            }
        }
        else{
            cls();
            bgcolour(" INVALID POLL ID !! \n\n",4,0x0060);
        }
    }
    break;
}

case 6:{
    bool check=true;
    cls();
    cin.ignore();
    while(check){
        char ID[5];
        bgcolour("\n\t\t ACCESS A PREVIOUS POLL \n\n",15,0x0040);
        colour("Enter Poll ID: ",14);
        gets(ID);
        char PollFileName[]="Data/Polls/";
        char Str2[]=".csv";
        char PID[]="";
        strcat(PID,ID);
        char filename[]="Data/Polls.csv";
        char CandFile[]="Data/Candidate.csv";
        char VoterFile[]="Data/Voter.csv";

        vector<string>PollInfo = ReadRecord(filename,ID);
        if(CheckRecord(filename,ID)){
            cls();
            strcat(PollFileName,ID);
            strcat(PollFileName,Str2);
            check=false;
            bgcolour("\n\t\t PREVIOUS POLL DATA \n\n",15,0x0010);
            colour("\t POLL ID: ",14);cout<<PollInfo[0];
            colour("\n\t POLL NAME: ",14);cout<<PollInfo[1];;
            colour("\n\t POLL DATE: ",14);cout<<PollInfo[2];;

            if(PollInfo[3]=="Conducted"){
                int rows=0;
                char str[]=".csv";
                strcat(PollFileName,str);
                ifstream file(PollFileName);
```

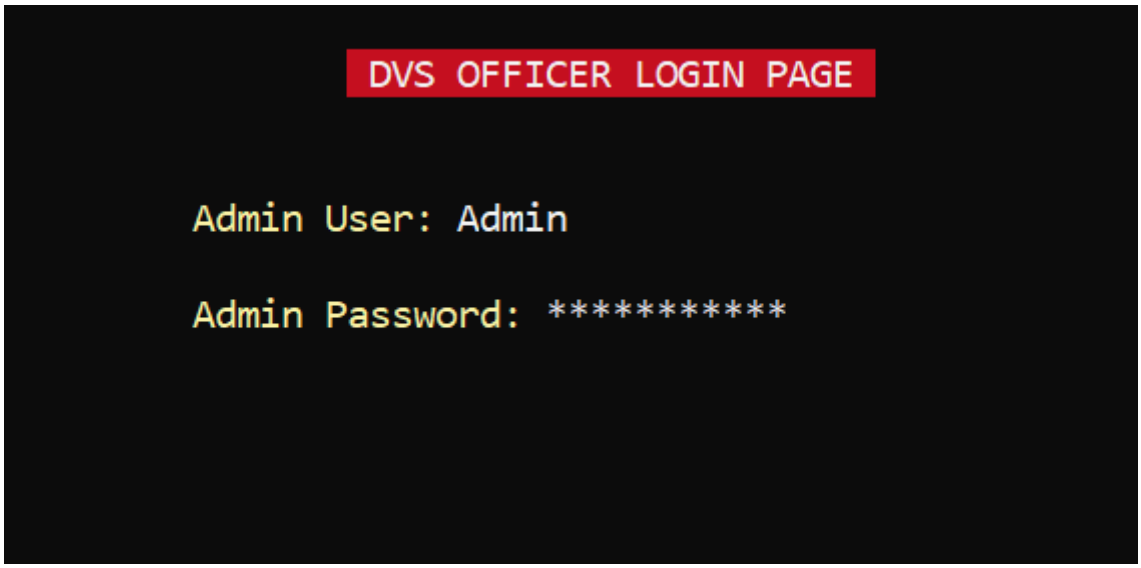
```
string row;
while(getline(file,row)){
   rows++;
}
file.close();

vector < vector<string> > PollData = ReadFile(PollFileName);
int i,j;
string SortedCand[rows];
int SortedVotes[rows];

for(i=1;i<rows;i++){
   SortedCand[i-1]=PollData[i][0];
   SortedVotes[i-1]=StrToInt(PollData[i][1]);
}
for(i=0;i<(rows-1);i++){
   for(j=i+1;j<(rows-1);j++){
      if(SortedVotes[i]<SortedVotes[j]){
         int temp1 =SortedVotes[i];
         SortedVotes[i]=SortedVotes[j];
         SortedVotes[j]=temp1;

         string temp2  =SortedCand[i];
         SortedCand[i]=SortedCand[j];
         SortedCand[j]=temp2;
      }
   }
}

bgcolour("\n\n\t\t\t VOTING RESULTS \n\n",15,0x0020);
float TotalVotes=0, TotalVoters=0;
for(i=0;i<(rows-1);i++){
   TotalVotes+=SortedVotes[i];
}

ifstream VFile(VoterFile);
string VoterRows;
while(getline(VFile,VoterRows)){
   TotalVoters++;
}
VFile.close();

TotalVoters -= 1;
int Percentage=(TotalVotes/TotalVoters)*100;

SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE),10);
cout<<"\t > "<<Percentage;
SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 15);
colour(" % Voters Casted Their Vote\n\n",10);
```

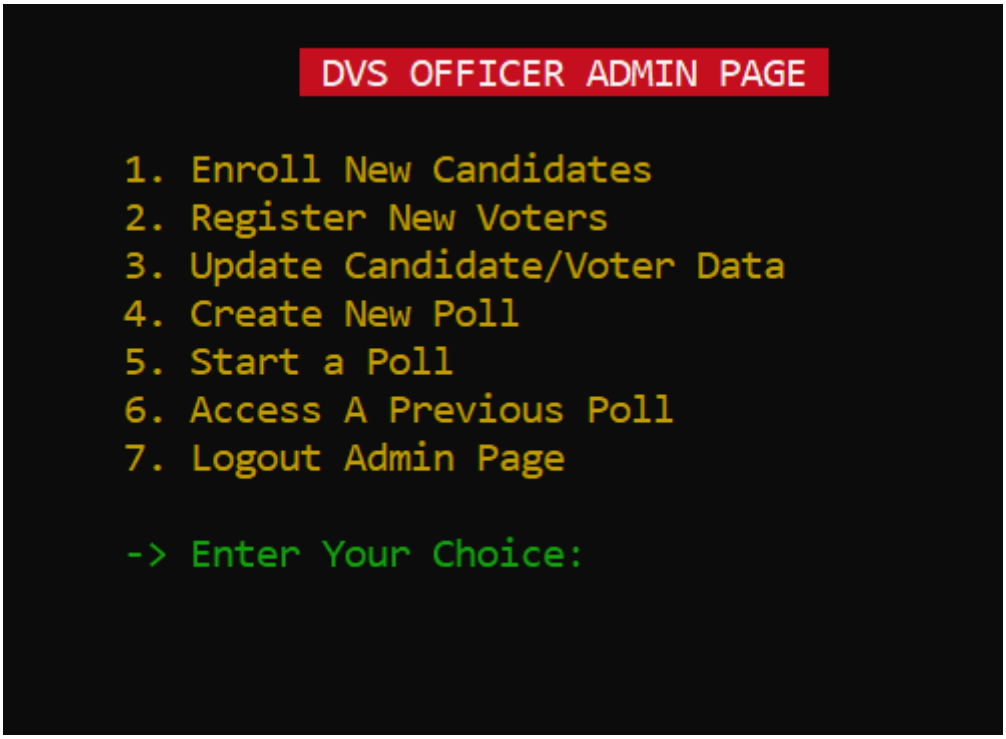
```
                for(i=1;i<rows;i++){
                    string data=SortedCand[i-1];
                    char ID[data.length()+1];
                    strcpy(ID,data.c_str());
                    vector<string> CandData = ReadRecord(CandFile,ID);
                    colour("\t\tCandidate ID: ",14);cout<<CandData[0]<<"\t\t";colour("Votes:
",6);
                    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 2);
                    cout<<SortedVotes[i-1];
                    SetConsoleTextAttribute(GetStdHandle(STD_OUTPUT_HANDLE), 15);
                    colour("\n\t\tCandidate Name: ",14);cout<<CandData[1];
                    colour("\n\t\tParty Name: ",14);cout<<CandData[2];
                    colour("\n\t\tParty Symbol: ",14);cout<<CandData[3];
                    colour("\n\t\t-----------------------------------\n",11);
                }
                colour("\n\n\t Press any Key to Continue:",2);
                getch();
                cls();
                AdminPage();
            }
            else{
                bgcolour("\n\n\t This Poll Is Not Yet Taken !! \n",4,0x0060);
                colour("\n\t\t-> Press Any Key to Continue: ",10);
                getch();
                cls();
                AdminPage();
            }
        }
        else{
            cls();
            bgcolour("INVALID POLL ID !!\n\n",4,0x0060);
        }
    }
    break;
}

case 7:{
    cls();
    bgcolour("\n\n\n\n\n\t\t\t\t LOGGED OUT SUCCESSFULLY ",15,0x0020);
    colour("\n\n\t\t\t\t-> Press Any Key To Continue:",10);
    getch();
    cls();
    HomePage();
    break;
}

default:{
    cls();
    colour("Invalid Choice !!\n\n ",4);
    AdminPage();
```

```
            break;
        }
    }
}

void HomePage(){
    bgcolour("\n\t\t WELCOME TO DIGITAL VOTING SYSTEM(DVS) \n\n",15,0x0040);
    colour("\t\t\t    DVS ADMIN PAGE \n \n \n",2);
    colour("\t-> Press Enter FOR ADMIN LOGIN\n \n",14);
    colour("\t-> Any Other Key TO EXIT \n\n \t\t",14);
    cout<<"Press Key: ";
    int choice=getch();

    if(choice==13){
        cls();
        if(DVSOfficer.login()){
            DVSOfficer.AdminPage();
        }
    }
    else{
        exit(0);
    }
}

int main(){

    colour("",15);
    HomePage();

    return 0;
}
```

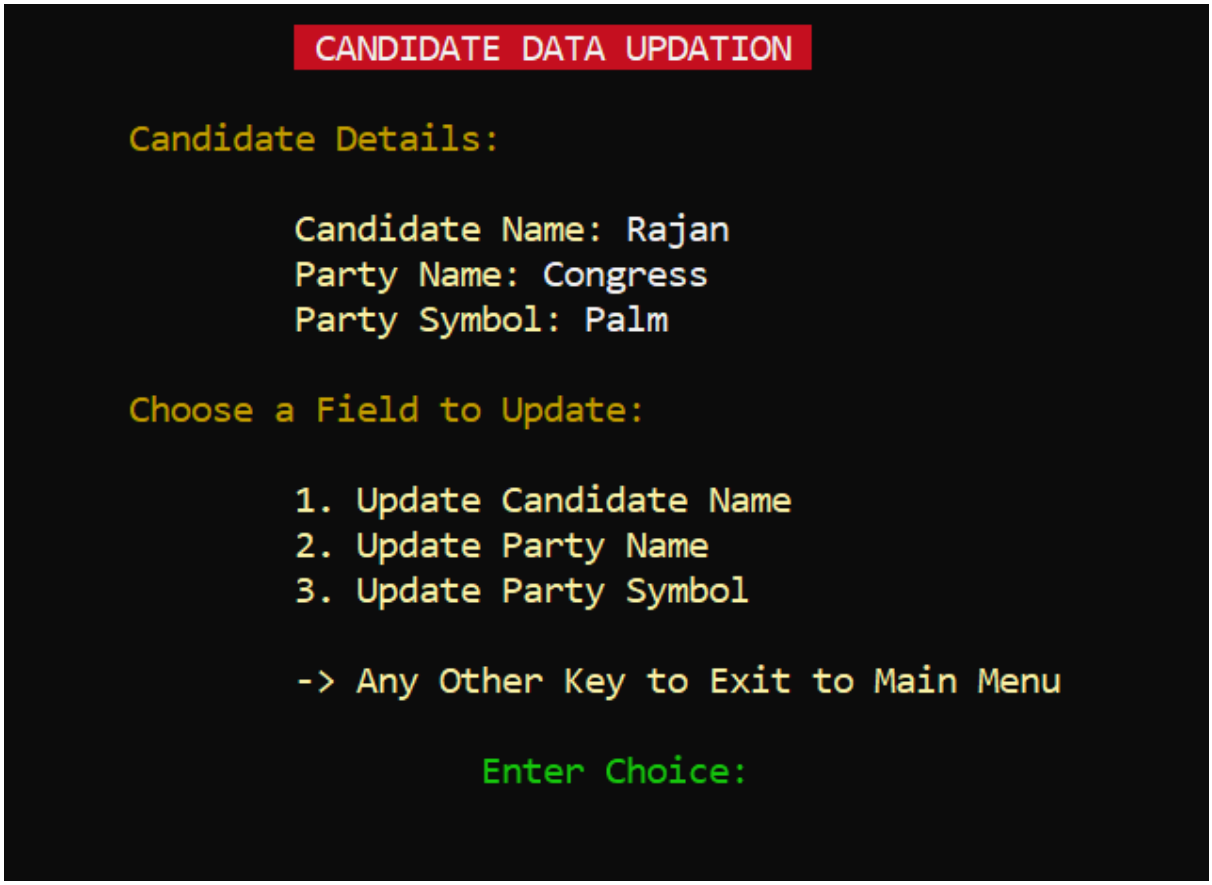
Some snapshots while the project is being implemented:

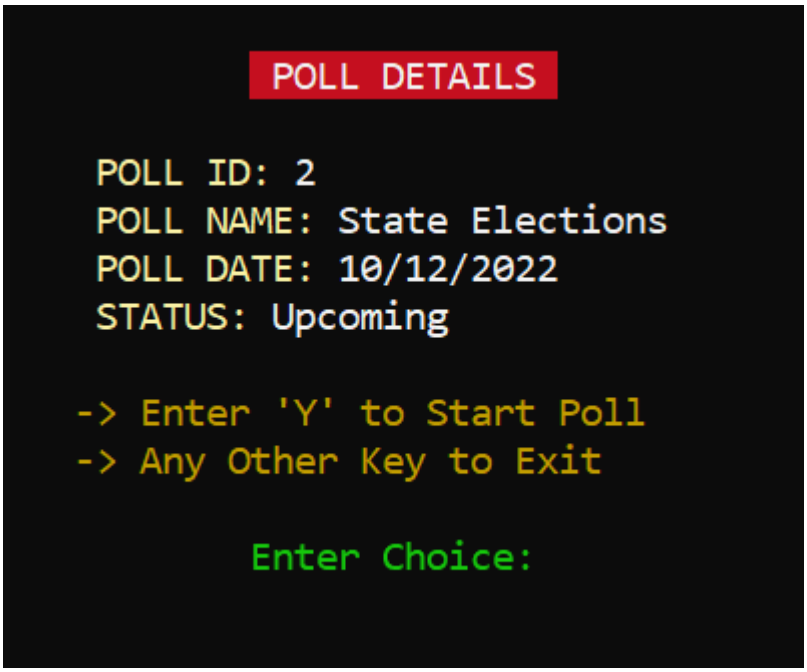


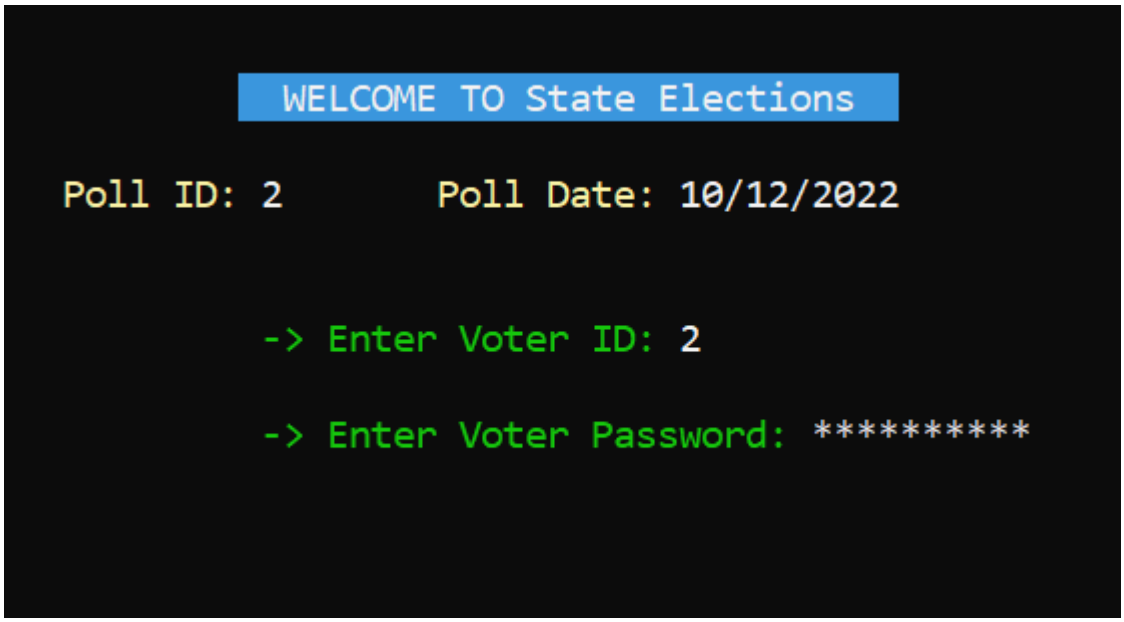



DVS OFFICER ADMIN PAGE
1. Enroll New Candidates
2. Register New Voters
3. Update Candidate/Voter Data
4. Create New Poll
5. Start a Poll
6. Access A Previous Poll
7. Logout Admin Page
-> Enter Your Choice:


NEW CANDIDATE ENROLLMENT
Candidate ID: 1
Candidate Name: Rajan
Party Name: Congress
Party Symbol: Palm


CANDIDATE DATA UPDATION
Candidate Details:
Candidate Name: Rajan
Party Name: Congress
Party Symbol: Palm
Choose a Field to Update:
1. Update Candidate Name
2. Update Party Name
3. Update Party Symbol
-> Any Other Key to Exit to Main Menu
Enter Choice:


POLL DETAILS
POLL ID: 2
POLL NAME: State Elections
POLL DATE: 10/12/2022
STATUS: Upcoming
-> Enter 'Y' to Start Poll
-> Any Other Key to Exit
Enter Choice:


WELCOME TO State Elections
Poll ID: 2
Poll Date: 10/12/2022
-> Enter Voter ID: 2
-> Enter Voter Password: **********


WELCOME TO State Elections
Poll ID: 2
Poll Date: 10/12/2022
> CANDIDATE LIST:
Candidate ID: 1
Candidate Name: Rajan
Party Name: Congress
Party Symbol: Palm
Candidate ID: 2
Candidate Name: Kushal
Party Name: BJP
Party Symbol: Cycle
Candidate ID: 3
Candidate Name: Peter
Party Name: INC
Party Symbol: Pencil
Candidate ID: 4
Candidate Name: Manav Deep
Party Name: TKD
Party Symbol: Hawk
-> Enter Candidate ID to Vote:


WELCOME TO State Elections
Poll ID: 2
Poll Date: 10/12/2022
Voter Name: Pritish
You Chose the following Candidate:
Candidate ID: 3
Candidate Name: Peter
Party Name: INC
Party Symbol: Pencil
-> Press Enter to Confirm
-> Any Other Key to Change Choice
Press Key:

```
        PREVIOUS POLL DATA

POLL ID: 2
POLL NAME: State Elections
POLL DATE: 10/12/2022

                VOTING RESULTS

> 78 % Voters Casted Their Vote

        Candidate ID: 2         Votes: 7
        Candidate Name: Kushal
        Party Name: BJP
        Party Symbol: Cycle
        -------------------------------------
        Candidate ID: 3         Votes: 4
        Candidate Name: Peter
        Party Name: INC
        Party Symbol: Pencil
        -------------------------------------
        Candidate ID: 1         Votes: 3
        Candidate Name: Rajan
        Party Name: Congress
        Party Symbol: Palm
        -------------------------------------
        Candidate ID: 4         Votes: 1
        Candidate Name: Manav Deep
        Party Name: TKD
        Party Symbol: Hawk
        -------------------------------------


Press any Key to Continue:
```