VIRTUAL VOTING MACHINE

AMAN DANG

SUBMITTED BY: AMAN GIRI

TARUN SHARMA

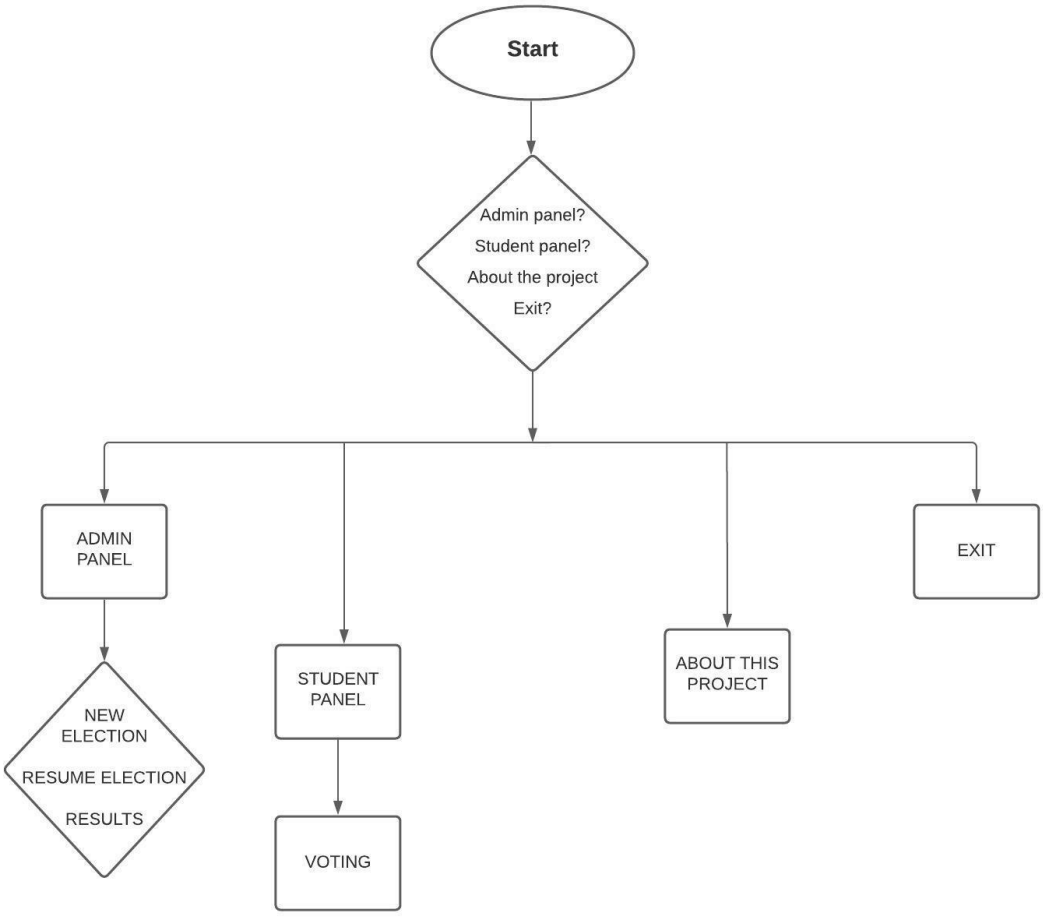
INDEX

INTRODUCTION

Virtual Voting Machine is a project coded in the C programming language which ensures a fair Election within a small group or class. The project gives us the understanding of the various file operations which include writing, reading, modifying etc. The project also gives us the understanding of functions and using various pre-defined functions in the C programming Language.

The project contains a main menu which allows the user to choose their designation i.e. Admin or the student, The students are only able to vote whereas the admin can login and can initiate a new election or can continue the election which was previously stopped or halted for some reason. The menu also contains the information about the project.

FLOWCHART


Start
Admin panel?
Student panel?
About the project
Exit?
ADMIN PANEL
NEW ELECTION
RESUME ELECTION
RESULTS
STUDENT PANEL
VOTING
ABOUT THIS PROJECT
EXIT

HEADER FILES USED

The following header files were used in the source code of this project.

- <stdio.h>
- <stdlib.h>
- <string.h>
- <dos.h>
- <time.h>
- <windows.h>
- <conio.h>

SOURCE CODE

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <conio.h>
#include <windows.h>
#include <dos.h>
#include <time.h>

struct ValidID //structure for basic info collection
{
   int year;
   char branch[6];
   int totalVoters;
};
typedef struct candidate //structure for canditate info
{
   int cid;
   char cname[20];
   int votes;
} CANDIDATE;

struct ValidID ValidID;       //stores Valid user ID parameters
CANDIDATE candidateArray[20]; //to store information all candidates
```

```
int numberOfCandidates;      //Total number of candidates standing for election
char studentVotes[200];      //to store information of votes given by each student

//**gotoxy function defination for gcc compiler
void gotoxy(int x, int y)
{
   COORD c;
   c.X = x;
   c.Y = y;
   SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE), c);
}
//** function to control the styles
void delay(int secs)
{
   // Converting time into milliseconds
   int ms = 1000 * secs;

   //  start time
   clock_t start_time = clock();

   // looping till required time is not achieved
   while (clock() < start_time + ms)
      ;
}
//To extract year from userID entered -- For example, userID:2021bteai00014 year:2021
int extractYear(char userID[15])
{
   int year = 0;
```

```
    char tmp;
    for (int i = 0; i < 4; i++)
    {
        tmp = userID[i];
        year = (year * 10) + (tmp - 48);
    }
    return year;
}
// Extraction of roll number from userid>>> for eg, userID:2021bteai00014 ; roll no. 14
int extractRollNo(char userID[15])
{
    int rollno = 0;
    char tmp;
    for (int i = 9; i < 14; i++)
    {
        tmp = userID[i];
        rollno = (rollno * 10) + (tmp - 48);
    }
    return rollno;
}

//Will check whether the global branch code and inputed branch code is matching or not
int checkBranchCode(char userID[15])
{
    char branchCode[6];
    for (int i = 4; i < 9; i++)
    {
        branchCode[i - 4] = userID[i];
```

```
    }
    branchCode[5] = '\0';
    if (strcmp(branchCode, ValidID.branch) == 0)
        return 1;
    else
        return 0;
}
// Login/password function
int authenticateAdmin()
{
    char username[15], password[6];

    printf("\nEnter username: ");
    scanf("%s", username);
    if ((strcmp(username, "admin")) != 0)
        return 0;
    else

    {
        printf("Enter Password: ");
        int z = 0;
        for (z = 0; z < 5; z++)
        {
            password[z] = getch();
            printf("%c", '*');
        }
        password[z] = '\0';
        if ((strcmp(password, "admin")) != 0)
```

```
            return 0;
        }
        return 1;
}
// Creating text files for storing info of candidates
void createCandidateFiles()
{
    printf("\nCandidate files are being created...\n");
    FILE *fp;
    char filename[20];
    for (int i = 1; i <= numberOfCandidates; i++)
    {
        sprintf(filename, "candidate%d.txt", i);
        fp = fopen(filename, "w");
        fprintf(fp, "0\n%s", candidateArray[i - 1].cname);
        fclose(fp);
    }
    Sleep(15);
    printf("\nCreated Files successfully\n");
}
// To determine max number of votes to the candidates
int getWinner()
{
    int maxV = -1;
    int winnerCid;
    for (int i = 0; i < numberOfCandidates; i++)
    {
        if (candidateArray[i].votes > maxV)
```

```
        {
            winnerCid = candidateArray[i].cid;
            maxV = candidateArray[i].votes;
        }
        else if (candidateArray[i].votes == maxV)
        {
            return -1;
        }
    }
    return winnerCid;
}
// To start a new election
void newelection()
{
    gotoxy(31, 3);
    ;
    printf("\nNew Election Initiation:\n\n");
    gotoxy(31, 7);
    ;
    printf("\nElections for which Year (YYYY): ");
    scanf("%d", &ValidID.year);
    gotoxy(31, 10);
    ;
    printf("\nEnter branch code (e.g., bteai): ");
    scanf("%s", &ValidID.branch);
    gotoxy(31, 13);
    printf("\nEnter Maximum number of voters: ");
    scanf("%d", &ValidID.totalVoters);
```

```
    gotoxy(31, 16);
    printf("\nEnter the Number of candidates standing: ");
    scanf("%d", &numberOfCandidates);

    for (int i = 0; i < ValidID.totalVoters; i++)
    {
        studentVotes[i] = '0';
    }

    for (int i = 0; i < numberOfCandidates; i++)
    {
        candidateArray[i].cid = i + 1;
        printf("Enter name of candidate %d: ", i + 1);
        scanf(" %s", candidateArray[i].cname);
        candidateArray[i].votes = 0;
    }
    return;
}

void saveFile()
{
    printf("Saving Election Info in File...\n");
    FILE *fp = fopen("ElectionInfo.txt", "w");
    if (fp == NULL)
    {
        printf("\nError in file creation\n");
        fclose(fp);
        return;
```

```
    }
    fprintf(
        fp, "%d\n%s\n%d\n%d",
        ValidID.year,
        ValidID.branch,
        ValidID.totalVoters,
        numberOfCandidates);
    fclose(fp);
    printf("Saved Successfully : )");
}
// Opening the file in read mode and getting all the data in it
void loadFile()
{
    FILE *f1, *f2, *f3;
    f1 = fopen("ElectionInfo.txt", "r");
    if (f1 == NULL)
        printf("Not Exist");
    fscanf(f1, "%d", &ValidID.year);
    fseek(f1, 2, SEEK_CUR);
    fscanf(f1, "%s", &ValidID.branch);
    fseek(f1, 2, SEEK_CUR);
    fscanf(f1, "%d", &ValidID.totalVoters);
    fseek(f1, 2, SEEK_CUR);
    fscanf(f1, "%d", &numberOfCandidates);
    fclose(f1);

    //load candidates info and student votes
    for (int i = 0; i < ValidID.totalVoters; i++)
```

```
    {
        studentVotes[i] = '0';
    }
    for (int i = 1; i <= numberOfCandidates; i++)
    {
        int location;
        char filename[20];
        sprintf(filename, "candidate%d.txt", i);
        f2 = fopen(filename, "r+");
        candidateArray[i - 1].cid = i;
        fscanf(f2, "%d", &candidateArray[i - 1].votes);
        fscanf(f2, "%s", candidateArray[i - 1].cname);
        while (!feof(f2))
        {
            fscanf(f2, "%d", &location);
            studentVotes[location - 1] = i + 48;
        }
        fclose(f2);
    }
}
// Main admin panel
void adminPanel()
{
    system("cls");
    while (1)
    {

        if (authenticateAdmin() != 1)
```

```
{
    printf("\n Invalid Username or Password \n");
    break;
}

printf("\n\nLOGGED IN SUCCESSFULLY (Press Enter)");
getch();

while (1)
{
    char inputID[15];
    char input;
    char banInp;
    int WinnerCid, totalVotedNow = 0;
    system("cls");
    printf("\n1.New Election\n2.Continue Previous Election\n3.Result\n4.Logout\nOption:");
    scanf(" %c", &input);

    switch (input)
    {
    case '1':
        newelection();
        saveFile();
        createCandidateFiles();
        system("cls");
        break;
    case '2':
        loadFile();
```

```
            system("cls");
            break;

        case '3':
            WinnerCid = getWinner();
            if (WinnerCid != -1)
            {
                printf("\nWinner is %s with %d votes\n", candidateArray[WinnerCid - 1].cname, candidateArray[WinnerCid - 1].votes);
            }
            else
            {
                printf("\nIts A TIE");
            }
            printf("\nFull Result\n");
            for (int i = 0; i < numberOfCandidates; i++)
            {
                totalVotedNow += candidateArray[i].votes;
                printf("%d. %s -> %d votes\n", candidateArray[i].cid, candidateArray[i].cname, candidateArray[i].votes);
            }
            printf("\nVoting Percentage: %d %%\n\n", (totalVotedNow * 100) / ValidID.totalVoters);
            system("cls");
            break;
        case '4':
            return;
        default:
            printf("Invalid Option");
            getch();
```

```
        }
      }
    }
};
// To check if the entered userid is valid and matches with given prototype
int isValid(char userID[15])
{
   if (strlen(userID) != 14)
      return 0;

   int inputedYear = extractYear(userID);
   int inputedRollNo = extractRollNo(userID);

   if (inputedYear != ValidID.year || checkBranchCode(userID) != 1 || inputedRollNo > ValidID.totalVoters)
      return 0;
   else
      return 1;
}
// To find if the userid is voting again or if it is the first time>> Controls the repetetion of votes
int isVoted(char userID[15])
{
   int location = extractRollNo(userID);
   if (studentVotes[location - 1] == '0')
      return 0;
   else
      return 1;
}
```

```
void saveVote(char userID[15], char voteInput)
{
   char filename[20];
   sprintf(filename, "candidate%d.txt", voteInput - 48);
   FILE *fp = fopen(filename, "r+");
   int location = extractRollNo(userID);
   studentVotes[location - 1] = voteInput;
   candidateArray[voteInput - 49].votes++;
   fseek(fp, 0, SEEK_SET);
   fprintf(fp, "%d\n", candidateArray[voteInput - 49].votes);
   fseek(fp, 0, SEEK_END);
   fprintf(fp, "\n%d", location);
   fclose(fp);
}
//Main student panel
void studentPanel()
{
   char userID[15];
   char voteInput;
   while (1)
   {
      printf("\n\n  To exit press 0");
      printf("\n\n\xB3\xDB\xDB\xDB\xDB\xDB\xDB\xDBIF YOU ARE NOT AWARE OF YOUR USER ID PLEASE CONTACT THE ADMIN");
      printf("\n\n\n\n  Enter user ID:");

      scanf("%s", userID);
      if (strcmp(userID, "0") == 0)
         return;
```

```
    if (isValid(userID) != 1)
    {
        printf("\n  Invalid User ID(Press Enter)");
        getch();
        continue;
    }
    if (isVoted(userID) != 0)
    {
        printf("\n\xB3\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB   THE USER ID ENTERED HAS ALREADY BEEN VOTED FROM\n  FOR FURTHER QUERY CONTACT ADMIN");
        getch();
        continue;
    }
    printf("\n\n  Candidates for election:");
    for (int i = 0; i < numberOfCandidates; i++)
    {
        printf("\n  %d. %s", i + 1, candidateArray[i].cname);
    }
    printf("\n\n  Your Vote(Enter Number):");
    voteInput = getch();
    printf("*");
    if (voteInput - 48 < 1 || voteInput - 48 > numberOfCandidates)
    {
        printf("\nInvalid Vote\nTry Again...");
        getch();
        continue;
    }
    saveVote(userID, voteInput);
    printf("\n\nThank you! Your vote has been recorded.(Press Enter)");
```

```
        getch();
    }
}
//Reading the information about the team and the project from a text file
int projectinfo()
{
    FILE *fp;

    char ch;

    fp = fopen("pinfo.txt", "r");

    /* fopen() return NULL if last operation was unsuccessful */
    if (fp == NULL)
    {
        /* Unable to open file hence exit */
        printf("Unable to open file.\n");
        printf("Please check whether file exists and you have read privilege.\n");
        exit(EXIT_FAILURE);
    }

    // File open success message
    printf("File opened successfully. Reading file contents character by character. \n\n");

    do
    { // Read single character from file
        ch = fgetc(fp);
```

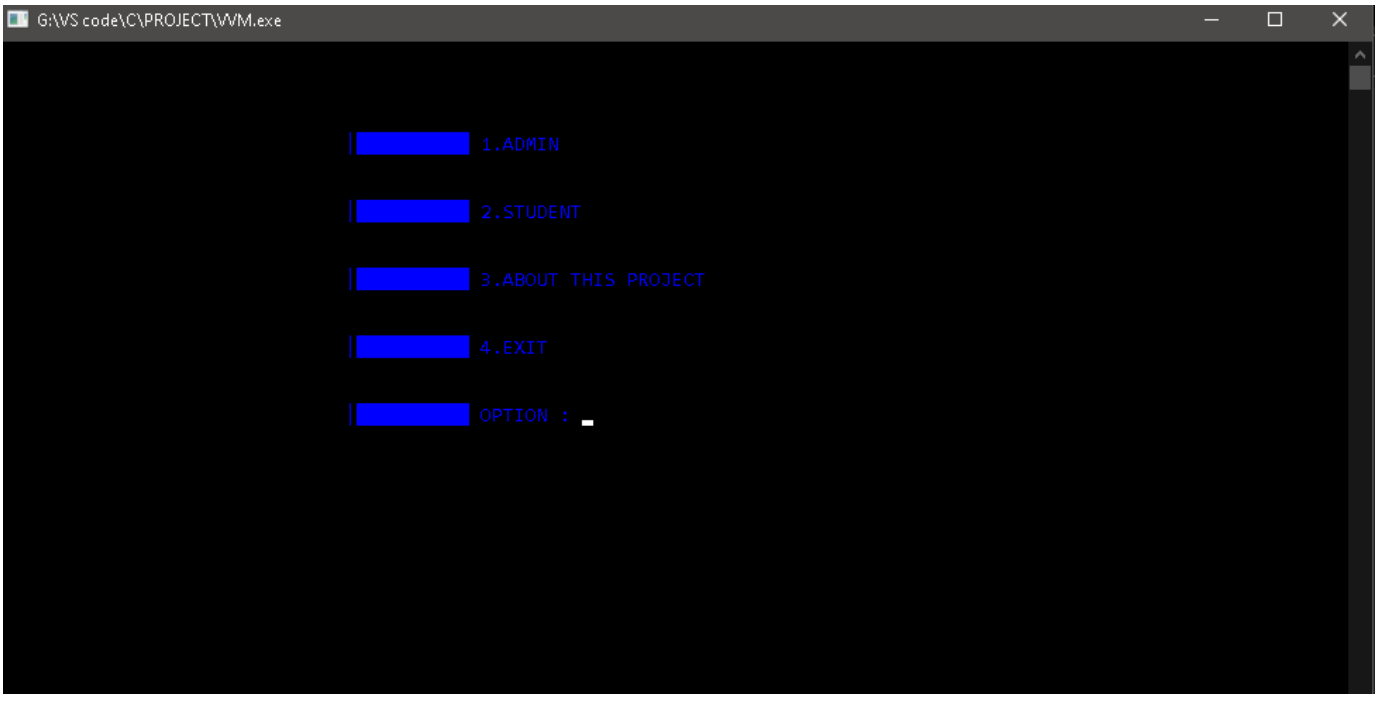
```
        // Print character read on console
        putchar(ch);

    } while (ch != EOF); // Repeat this if last read character is not EOF

    delay(10);
    // Done with this file, close file to release resource
    fclose(fp);

    return 0;
}
//Start of main function
int main()
{
    int j;
    int l;
    system("color 9"); //For color theme >> BLUEISH SHADE 'B'
    gotoxy(5, 10);
    char welcome[20] = "WELCOME";
    l = strlen(welcome);
    for (j = 0; j <= 45; j++)
    {
        Sleep(50); //To display the animation feel to the 'welcome' text
        printf("\xDB");
    }
    for (j = 0; j <= l; j++)
    {
        Sleep(60);
```

```
    printf(" %c", welcome[j]);
}
for (j = 0; j <= 45; j++)
{
    Sleep(50);
    printf("\xDB");
}

while (1)
{

    gotoxy(50, 15);

    delay(2);      //delays the screen contents
    system("cls"); //clears the cnsole
    gotoxy(31, 4);
    printf("\xB3\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB 1.ADMIN");
    gotoxy(31, 7);
    printf("\xB3\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB 2.STUDENT");
    gotoxy(31, 10);
    printf("\xB3\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB 3.ABOUT THIS PROJECT");
    gotoxy(31, 13);
    printf("\xB3\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB 4.EXIT");
    gotoxy(31, 16);
    printf("\xB3\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB\xDB OPTION : ");

    char input;
    scanf(" %c", &input);
```

```
        switch (input)
        {
        case '1':
            system("cls");
            adminPanel();
            break;
        case '2':
            system("cls");
            studentPanel();
            break;
        case '3':
            system("cls");
            projectinfo();
            break;
        case '4':
            return 0;
        default:
            printf("\nInvalid option");
            getch();
        }
    }
    return 0;
}
```

OUTPUT


G:\VS code\C\PROJECT\VVM.exe
W E L C O M E


G:\VS code\C\PROJECT\VVM.exe
1.ADMIN
2.STUDENT
3.ABOUT THIS PROJECT
4.EXIT
OPTION :