```cpp
#include <iostream>
#include <fstream>
#include <string>
#include <vector>
#include <iomanip>

using namespace std;
const string ClientsFileName = "Clients.txt";

void ShowMainMenue();

struct sClient
{
    string AccountNumber;
    string PinCode;
    string Name;
    string Phone;
    double AccountBalance;
    bool MarkForDelete = false;
};

vector<string> SplitString(string S1, string Delim)
{
    vector<string> vString;
    short pos = 0;
    string sWord; // define a string variable

    // use find() function to get the position of the delimiters
    while ((pos = S1.find(Delim)) != std::string::npos)
    {
        sWord = S1.substr(0, pos); // store the word
        if (sWord != "")
        {
            vString.push_back(sWord);
        }

        S1.erase(0, pos + Delim.length());  /* erase() until positon
          and move to next word. */
    }

    if (S1 != "")
    {
        vString.push_back(S1); // it adds last word of the string.
    }

    return vString;

}

sClient ConvertLinetoRecord(string Line, string Seperator = "#//#")
{
    sClient Client;
    vector<string> vClientData;
```

```
53      vClientData = SplitString(Line, Seperator);
54
55      Client.AccountNumber = vClientData[0];
56      Client.PinCode = vClientData[1];
57      Client.Name = vClientData[2];
58      Client.Phone = vClientData[3];
59      Client.AccountBalance = stod(vClientData[4]);//cast string to
          double
60      return Client;
61  }
62
63  string ConvertRecordToLine(sClient Client, string Seperator = "#//#")
64  {
65
66      string stClientRecord = "";
67      stClientRecord += Client.AccountNumber + Seperator;
68      stClientRecord += Client.PinCode + Seperator;
69      stClientRecord += Client.Name + Seperator;
70      stClientRecord += Client.Phone + Seperator;
71      stClientRecord += to_string(Client.AccountBalance);
72      return stClientRecord;
73  }
74
75  bool ClientExistsByAccountNumber(string AccountNumber, string FileName)
76  {
77
78      vector <sClient> vClients;
79      fstream MyFile;
80      MyFile.open(FileName, ios::in);//read Mode
81
82      if (MyFile.is_open())
83      {
84          string Line;
85          sClient Client;
86
87          while (getline(MyFile, Line))
88          {
89              Client = ConvertLinetoRecord(Line);
90              if (Client.AccountNumber == AccountNumber)
91              {
92                  MyFile.close();
93                  return true;
94              }
95              vClients.push_back(Client);
96          }
97
98          MyFile.close();
99
100     }
101     return false;
102 }
103
104 sClient ReadNewClient()
```

```
105 {
106     sClient Client;
107     cout << "Enter Account Number? ";
108
109     // Usage of std::ws will extract allthe whitespace character
110     getline(cin >> ws, Client.AccountNumber);
111
112     while (ClientExistsByAccountNumber(Client.AccountNumber,
        ClientsFileName))
113     {
114         cout << "\nClient with [" << Client.AccountNumber << "]
            already exists, Enter another Account Number? ";
115         getline(cin >> ws, Client.AccountNumber);
116     }
117
118     cout << "Enter PinCode? ";
119     getline(cin, Client.PinCode);
120
121     cout << "Enter Name? ";
122     getline(cin, Client.Name);
123
124     cout << "Enter Phone? ";
125     getline(cin, Client.Phone);
126
127     cout << "Enter AccountBalance? ";
128     cin >> Client.AccountBalance;
129
130     return Client;
131 }
132
133 vector <sClient> LoadCleintsDataFromFile(string FileName)
134 {
135     vector <sClient> vClients;
136     fstream MyFile;
137     MyFile.open(FileName, ios::in);//read Mode
138
139     if (MyFile.is_open())
140     {
141         string Line;
142         sClient Client;
143
144         while (getline(MyFile, Line))
145         {
146             Client = ConvertLinetoRecord(Line);
147             vClients.push_back(Client);
148         }
149         MyFile.close();
150     }
151     return vClients;
152 }
153
154 void PrintClientRecordLine(sClient Client)
155 {
```

```
156     cout << "| " << setw(15) << left << Client.AccountNumber;
157     cout << "| " << setw(10) << left << Client.PinCode;
158     cout << "| " << setw(40) << left << Client.Name;
159     cout << "| " << setw(12) << left << Client.Phone;
160     cout << "| " << setw(12) << left << Client.AccountBalance;
161 }
162
163 void ShowAllClientsScreen()
164 {
165     vector <sClient> vClients = LoadCleintsDataFromFile
          (ClientsFileName);
166
167     cout << "\n\t\t\t\t\tClient List (" << vClients.size() << ") Client
          (s).";
168     cout <<
          "\n_______________________________________________________";
169     cout << "_____________________________________\n" << endl;
170
171     cout << "| " << left << setw(15) << "Accout Number";
172     cout << "| " << left << setw(10) << "Pin Code";
173     cout << "| " << left << setw(40) << "Client Name";
174     cout << "| " << left << setw(12) << "Phone";
175     cout << "| " << left << setw(12) << "Balance";
176     cout <<
          "\n_______________________________________________________";
177     cout << "_____________________________________\n" << endl;
178
179     if (vClients.size() == 0)
180         cout << "\t\t\t\tNo Clients Available In the System!";
181     else
182
183         for (sClient Client : vClients)
184         {
185
186             PrintClientRecordLine(Client);
187             cout << endl;
188         }
189
190     cout <<
          "\n_______________________________________________________";
191     cout << "_____________________________________\n" << endl;
192 }
193
194 void PrintClientCard(sClient Client)
195 {
196     cout << "\nThe following are the client details:\n";
197     cout << "-----------------------------------";
198     cout << "\nAccout Number: " << Client.AccountNumber;
199     cout << "\nPin Code     : " << Client.PinCode;
200     cout << "\nName         : " << Client.Name;
201     cout << "\nPhone        : " << Client.Phone;
202     cout << "\nAccount Balance: " << Client.AccountBalance;
203     cout << "\n-----------------------------------\n";
```

```
204 }
205
206 bool FindClientByAccountNumber(string AccountNumber, vector <sClient>
      vClients, sClient& Client)
207 {
208     for (sClient C : vClients)
209     {
210
211         if (C.AccountNumber == AccountNumber)
212         {
213             Client = C;
214             return true;
215         }
216
217     }
218     return false;
219 }
220
221 sClient ChangeClientRecord(string AccountNumber)
222 {
223     sClient Client;
224
225     Client.AccountNumber = AccountNumber;
226
227     cout << "\n\nEnter PinCode? ";
228     getline(cin >> ws, Client.PinCode);
229
230     cout << "Enter Name? ";
231     getline(cin, Client.Name);
232
233     cout << "Enter Phone? ";
234     getline(cin, Client.Phone);
235
236     cout << "Enter AccountBalance? ";
237     cin >> Client.AccountBalance;
238     return Client;
239 }
240
241 bool MarkClientForDeleteByAccountNumber(string AccountNumber, vector
      <sClient>& vClients)
242 {
243
244     for (sClient& C : vClients)
245     {
246
247         if (C.AccountNumber == AccountNumber)
248         {
249             C.MarkForDelete = true;
250             return true;
251         }
252
253     }
254
```

```
255     return false;
256 }
257
258 vector <sClient> SaveCleintsDataToFile(string FileName, vector
      <sClient> vClients)
259 {
260     fstream MyFile;
261     MyFile.open(FileName, ios::out);//overwrite
262
263     string DataLine;
264
265     if (MyFile.is_open())
266     {
267         for (sClient C : vClients)
268         {
269
270             if (C.MarkForDelete == false)
271             {
272                 //we only write records that are not marked for
                      delete.
273                 DataLine = ConvertRecordToLine(C);
274                 MyFile << DataLine << endl;
275             }
276
277         }
278
279         MyFile.close();
280     }
281
282     return vClients;
283 }
284
285 void AddDataLineToFile(string FileName, string  stDataLine)
286 {
287     fstream MyFile;
288     MyFile.open(FileName, ios::out | ios::app);
289
290     if (MyFile.is_open())
291     {
292
293         MyFile << stDataLine << endl;
294
295         MyFile.close();
296     }
297 }
298
299 void AddNewClient()
300 {
301     sClient Client;
302     Client = ReadNewClient();
303     AddDataLineToFile(ClientsFileName, ConvertRecordToLine(Client));
304 }
305
```

```
306 void AddNewClients()
307 {
308     char AddMore = 'Y';
309     do
310     {
311         //system("cls");
312         cout << "Adding New Client:\n\n";
313 
314         AddNewClient();
315         cout << "\nClient Added Successfully, do you want to add more
              clients? Y/N? ";
316         cin >> AddMore;
317 
318     } while (toupper(AddMore) == 'Y');
319 
320 }
321 
322 bool DeleteClientByAccountNumber(string AccountNumber, vector
      <sClient>& vClients)
323 {
324     sClient Client;
325     char Answer = 'n';
326 
327     if (FindClientByAccountNumber(AccountNumber, vClients, Client))
328     {
329 
330         PrintClientCard(Client);
331 
332         cout << "\n\nAre you sure you want delete this client? y/n ? ";
333         cin >> Answer;
334         if (Answer == 'y' || Answer == 'Y')
335         {
336             MarkClientForDeleteByAccountNumber(AccountNumber,
                  vClients);
337             SaveCleintsDataToFile(ClientsFileName, vClients);
338 
339             //Refresh Clients
340             vClients = LoadCleintsDataFromFile(ClientsFileName);
341 
342             cout << "\n\nClient Deleted Successfully.";
343             return true;
344         }
345 
346     }
347     else
348     {
349         cout << "\nClient with Account Number (" <<
              AccountNumber << ") is Not Found!";
350         return false;
351     }
352 }
353 
354 bool UpdateClientByAccountNumber(string AccountNumber, vector
```

```
    <sClient>& vClients)
{

    sClient Client;
    char Answer = 'n';

    if (FindClientByAccountNumber(AccountNumber, vClients, Client))
    {

        PrintClientCard(Client);
        cout << "\n\nAre you sure you want update this client? y/n ? ";
        cin >> Answer;
        if (Answer == 'y' || Answer == 'Y')
        {
            for (sClient& C : vClients)
            {
                if (C.AccountNumber == AccountNumber)
                {
                    C = ChangeClientRecord(AccountNumber);
                    break;
                }
            }

            SaveCleintsDataToFile(ClientsFileName, vClients);

            cout << "\n\nClient Updated Successfully.";
            return true;
        }

    }
    else
    {
        cout << "\nClient with Account Number (" <<
          AccountNumber << ") is Not Found!";
        return false;
    }
}

string ReadClientAccountNumber()
{
    string AccountNumber = "";

    cout << "\nPlease enter AccountNumber? ";
    cin >> AccountNumber;
    return AccountNumber;

}

void ShowDeleteClientScreen()
{
    cout << "\n-----------------------------------\n";
    cout << "\tDelete Client Screen";
    cout << "\n-----------------------------------\n";
```

```
406
407     vector <sClient> vClients = LoadCleintsDataFromFile(ClientsFileName);
408     string AccountNumber = ReadClientAccountNumber();
409     DeleteClientByAccountNumber(AccountNumber, vClients);
410 }
411
412 void ShowUpdateClientScreen()
413 {
414     cout << "\n-----------------------------------\n";
415     cout << "\tUpdate Client Info Screen";
416     cout << "\n-----------------------------------\n";
417
418     vector <sClient> vClients = LoadCleintsDataFromFile(ClientsFileName);
419     string AccountNumber = ReadClientAccountNumber();
420     UpdateClientByAccountNumber(AccountNumber, vClients);
421
422 }
423
424 void ShowAddNewClientsScreen()
425 {
426     cout << "\n-----------------------------------\n";
427     cout << "\tAdd New Clients Screen";
428     cout << "\n-----------------------------------\n";
429
430     AddNewClients();
431 }
432
433 void ShowFindClientScreen()
434 {
435     cout << "\n-----------------------------------\n";
436     cout << "\tFind Client Screen";
437     cout << "\n-----------------------------------\n";
438
439     vector <sClient> vClients = LoadCleintsDataFromFile(ClientsFileName);
440     sClient Client;
441     string AccountNumber = ReadClientAccountNumber();
442     if (FindClientByAccountNumber(AccountNumber, vClients, Client))
443         PrintClientCard(Client);
444     else
445         cout << "\nClient with Account Number[" << AccountNumber << "] is not found!";
446 }
447
448 void ShowEndScreen()
449 {
450     cout << "\n-----------------------------------\n";
451     cout << "\tProgram Ends :-)";
452     cout << "\n-----------------------------------\n";
453 }
454
```

```
455 enum enMainMenueOptions
456 {
457     eListClients = 1, eAddNewClient = 2,
458     eDeleteClient = 3, eUpdateClient = 4,
459     eFindClient = 5, eExit = 6
460 };
461
462 void GoBackToMainMenue()
463 {
464     cout << "\n\nPress any key to go back to Main Menue...";
465     system("pause>0");
466     ShowMainMenue();
467
468 }
469
470 short ReadMainMenueOption()
471 {
472     cout << "Choose what do you want to do? [1 to 6]? ";
473     short Choice = 0;
474     cin >> Choice;
475
476     return Choice;
477 }
478
479 void PerfromMainMenueOption(enMainMenueOptions MainMenueOption)
480 {
481     switch (MainMenueOption)
482     {
483     case enMainMenueOptions::eListClients:
484     {
485         system("cls");
486         ShowAllClientsScreen();
487         GoBackToMainMenue();
488         break;
489     }
490     case enMainMenueOptions::eAddNewClient:
491         system("cls");
492         ShowAddNewClientsScreen();
493         GoBackToMainMenue();
494         break;
495
496     case enMainMenueOptions::eDeleteClient:
497         system("cls");
498         ShowDeleteClientScreen();
499         GoBackToMainMenue();
500         break;
501
502     case enMainMenueOptions::eUpdateClient:
503         system("cls");
504         ShowUpdateClientScreen();
505         GoBackToMainMenue();
506         break;
507
```

```
508     case enMainMenueOptions::eFindClient:
509         system("cls");
510         ShowFindClientScreen();
511         GoBackToMainMenue();
512         break;
513
514     case enMainMenueOptions::eExit:
515         system("cls");
516         ShowEndScreen();
517         break;
518     }
519 }
520
521 void ShowMainMenue()
522 {
523     system("cls");
524     cout << "===========================================\n";
525     cout << "\t\tMain Menue Screen\n";
526     cout << "===========================================\n";
527     cout << "\t[1] Show Client List.\n";
528     cout << "\t[2] Add New Client.\n";
529     cout << "\t[3] Delete Client.\n";
530     cout << "\t[4] Update Client Info.\n";
531     cout << "\t[5] Find Client.\n";
532     cout << "\t[6] Exit.\n";
533     cout << "===========================================\n";
534     PerfromMainMenueOption((enMainMenueOptions)ReadMainMenueOption());
535 }
536
537 int main()
538
539 {
540     ShowMainMenue();
541     system("pause>0");
542     return 0;
543 }
```