

# REAL-TIME FACE RECOGNITION

## Technical Report

*SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE  
AWARD OF THE DEGREE OF COMPUTER SCIENCE AND ENGINEERING*

## BACHELOR OF TECHNOLOGY



**Submitted by:**  
Jaskirat Singh  
2302566

**Submitted to:**  
Er. Satinderpal Singh  
Assistant Professor

**Dept. of Computer Science Engineering**  
Guru Nanak Dev Engineering College,  
Ludhiana, 141006

# Problem Statement

The LIBRARY MANAGEMENT SYSTEM project has book and student class with data members like book no, bookname, authorname. Books record is stored in a binary file. A student can issue book and deposit it within 15 days. Student is allowed to issue only one book. Student Records are stored in binary file. Administrator can add, modify or delete record.

# Solution

## Source code

```
1  #include <iostream>
2  #include <string>
3  #include <fstream>
4  #include <iomanip>
5  #include <cstring>
6
7  using namespace std;
8
9  class Book {
10 public:
11     char bookname[50], authorname[50];
12     int book_id;
13
14
15     void create_book() {
16         cout << "Enter book id: ";
17         cin >> book_id;
18         cin.ignore();
19         cout << "Enter book name: ";
20         cin.getline(bookname, 50);
21         cout << "Enter author name: ";
22         cin.getline(authorname, 50);
23     }
24
25     void display_book() const {
26         cout << left << setw(15) << book_id << setw(30) << bookname << setw(30) <<
27             authorname << endl;
28     }
29
30     void write_book_file() {
31         ofstream file("book.dat", ios::binary | ios::app);
32         if (!file) {
33             cout << "Unable to open file\n";
34             return;
35         }
36         create_book();
37         file.write(reinterpret_cast<const char*>(this), sizeof(*this));
38         file.close();
39         cout << "Record updated successfully!!\n";
40     }
41
42     void read_book_file() const {
43         ifstream file("book.dat", ios::binary);
```

```

43     if (!file) {
44         cout << "Unable to open file\n";
45         return;
46     }
47     cout << "\n\n\t\tBook List\n\n";
48     cout << left << setw(15) << "Book Number" << setw(30) << "Book Name " <<
49         setw(30) << "Author Name " << endl;
50     cout << "
51         -----\n";
52     while (file.read(reinterpret_cast<char*>(const_cast<Book*>(this)), sizeof(*
53         this))) {
54         display_book();
55     }
56     file.close();
57     cout << "
58         -----\n";
59 }
60
61 void modify_book(int n) {
62     fstream file("book.dat", ios::binary | ios::in | ios::out);
63     if (!file) {
64         cout << "File could not be opened! Press any key to exit.\n";
65         return;
66     }
67     bool found = false;
68     while (file.read(reinterpret_cast<char*>(this), sizeof(*this))) {
69         if (book_id == n) {
70             cout << "Enter new details for book having id " << n << endl;
71             create_book();
72             file.seekp(-static_cast<int>(sizeof(*this)), ios::cur);
73             file.write(reinterpret_cast<const char*>(this), sizeof(*this));
74             cout << "\nRecord modified!\n";
75             found = true;
76             break;
77         }
78     }
79     if (!found) cout << "\nRecord not found!\n";
80     file.close();
81 }
82
83 void delete_book(int n) {
84     ifstream inFile("book.dat", ios::binary);
85     if (!inFile) {
86         cout << "File could not be opened! Press any key to exit.\n";
87         return;
88     }
89
90     ofstream outFile("temp.dat", ios::binary);
91     while (inFile.read(reinterpret_cast<char*>(this), sizeof(*this))) {
92         if (book_id != n) {
93             outFile.write(reinterpret_cast<const char*>(this), sizeof(*this));
94         }
95     }
96     inFile.close();
97     outFile.close();
98     remove("book.dat");

```

```

95         rename("temp.dat", "book.dat");
96         cout << "\nRecord deleted!\n";
97     }
98     int bookid(){return book_id;}
99 };
100
101 class Student:public Book {
102 private:
103     char student_name[50];
104     int student_id;
105
106 public:
107     Student(){
108         strcpy(bookname,"no book issued");
109         strcpy(authername,"-");
110         strcpy(student_name,"-");
111         student_id = book_id = 0;
112     }
113     void create_student() {
114         cout << "Enter student id: ";
115         cin >> student_id;
116         cin.ignore();
117         cout << "Enter student name: ";
118         cin.getline(student_name, 50);
119     }
120
121     void display_student() const {
122         cout << left << setw(15) << student_id << setw(30) << student_name << endl;
123     }
124
125     void write_student_file() {
126         ofstream file("student.dat", ios::binary | ios::app);
127         if (!file) {
128             cout << "Unable to open file\n";
129             return;
130         }
131         create_student();
132         file.write(reinterpret_cast<const char*>(this), sizeof(*this));
133         file.close();
134         cout << "Record updated successfully!!\n";
135     }
136
137     void read_student_file() const {
138         ifstream file("student.dat", ios::binary);
139         if (!file) {
140             cout << "Unable to open file\n";
141             return;
142         }
143         cout << "\n\n\t\tStudent List\n\n";
144         cout << left << setw(15) << "Student id" << setw(30) << "Student Name " <<
145             endl;
146         cout << "
147             -----\n";
148         while (file.read(reinterpret_cast<char*>(const_cast<Student*>(this)),
149             sizeof(*this))) {
150             display_student();
151         }

```

```

149     file.close();
150     cout << "
        -----\n";
151 }
152
153 void delete_student(int n) {
154     ifstream inFile("student.dat", ios::binary);
155     if (!inFile) {
156         cout << "File could not be opened! Press any key to exit.\n";
157         return;
158     }
159
160     ofstream outFile("temp.dat", ios::binary);
161     while (inFile.read(reinterpret_cast<char*>(this), sizeof(*this))) {
162         if (student_id != n) {
163             outFile.write(reinterpret_cast<const char*>(this), sizeof(*this));
164         }
165     }
166     inFile.close();
167     outFile.close();
168     remove("student.dat");
169     rename("temp.dat", "student.dat");
170     cout << "\nRecord deleted!\n";
171 }
172 void issue_book(){
173     int id;
174     cout<<"\nEnter your Id : ";
175     cin>>id;
176
177     fstream file("student.dat", ios::binary | ios::in | ios::out);
178     if (!file) {
179         cout << "File could not be opened! Press any key to exit.\n";
180         return;
181     }
182     bool found = false;
183     while (file.read(reinterpret_cast<char*>(this), sizeof(*this))) {
184         if (book_id && student_id == id) {book_issued_data(); break;}
185         else if(student_id == id){
186             int bookid;
187             cout<<"\nEnter requested book id : ";
188             cin>>bookid;
189             if(book_finder(bookid)){
190                 file.seekp(-static_cast<int>(sizeof(*this)), ios::cur);
191                 file.write(reinterpret_cast<const char*>(this), sizeof(*this));
192                 cout << "\nBook is issued for 15 days!\n";
193                 found = true;
194                 book_issued_data();
195                 break;}
196         }
197         else{
198             cout<<"User not found!!\n";
199             found = true;
200         }
201     }
202     if (!found) cout << "\nBook already issued!\n";
203     file.close();
204 }

```

```

205 }
206 void return_book(){
207     int id;
208     cout<<"\nEnter your Id : ";
209     cin>>id;
210
211     fstream file("student.dat", ios::binary | ios::in | ios::out);
212     if (!file) {
213         cout << "File could not be opened! Press any key to exit.\n";
214         return;
215     }
216     bool found = false;
217     while (file.read(reinterpret_cast<char*>(this), sizeof(*this))) {
218         if (book_id && student_id == id) {
219             strcpy(bookname,"no book issued");
220             strcpy(authurname,"-");
221             book_id = 0;
222             file.seekp(-static_cast<int>(sizeof(*this)), ios::cur);
223             file.write(reinterpret_cast<const char*>(this), sizeof(*this));
224             cout << "\nBook is returned to library!\n";
225             found = true;
226             break;
227         }
228     }
229     if (!found) cout << "\nBook not issued!\n";
230     file.close();
231 }
232
233 bool book_finder(int n){
234     fstream file("book.dat", ios::binary | ios::in | ios::out);
235     if (!file) {
236         cout << "File could not be opened! Press any key to exit.\n";
237         return false;
238     }
239     bool found = false;
240     Book b;
241     while (file.read(reinterpret_cast<char*>(&b), sizeof(b))) {
242         if(b.book_id == n){
243             strcpy(this->bookname,b.bookname);
244             strcpy(this->authurname,b.authurname);
245             this->book_id = b.book_id;
246             found = true;
247             break;
248         };
249     }
250     if(!found)cout<<"book not found!"<<endl;
251     file.close();
252     return found;}
253
254 void book_issued_data(){
255     cout << left << setw(15) << "Student id" << setw(30) << "Student Name "
256         << setw(50)<<"Book Issued"<<endl;
257     cout << "
258         -----\n";
259     cout << left << setw(15) << student_id << setw(30) << student_name <<
260         book_id<<": ("<<bookname<<" by "<<authurname<<")"<<endl;
261 }

```

```

259 };
260
261 int main() {
262     Book b;
263     Student student;
264     int choice = 0, n;
265
266     do {
267         cout << "\n\n\t--- Administrator Menu ---";
268         cout << "\n1. Add New Book";
269         cout << "\n2. Display All Books";
270         cout << "\n3. Modify Book Record";
271         cout << "\n4. Delete Book Record";
272         cout << "\n5. Add New Student";
273         cout << "\n6. Display All Students";
274         cout << "\n7. Delete Student Record";
275         cout << "\n8. Issue book";
276         cout << "\n9. return book";
277         cout << "\n0. Exit";
278         cout << "\nEnter your choice: ";
279         cin >> choice;
280
281         switch (choice) {
282             case 1:
283                 b.write_book_file();
284                 break;
285             case 2:
286                 b.read_book_file();
287                 break;
288             case 3:
289                 cout << "\nEnter Book Number to modify: ";
290                 cin >> n;
291                 b.modify_book(n);
292                 break;
293             case 4:
294                 cout << "\nEnter Book Number to delete: ";
295                 cin >> n;
296                 b.delete_book(n);
297                 break;
298             case 5:
299                 student.write_student_file();
300                 break;
301             case 6:
302                 student.read_student_file();
303                 break;
304             case 7:
305                 cout << "\nEnter Student ID to delete: ";
306                 cin >> n;
307                 student.delete_student(n);
308                 break;
309             case 8: student.issue_book();break;
310             case 9: student.return_book();break;
311             case 0:
312                 cout << "Exiting...\n";
313                 break;
314             default:
315                 cout << "Invalid choice\n";
316         }

```



```
317     } while (choice != 0);
318
319     return 0;
320 }
```