

# Student Management System

*Mastering Embedded Systems Diploma*  
[www.learnindepth.com](http://www.learnindepth.com)  
*First Term (Final Project 1)*

*Eng. Michel Adel Fouad*  
*My Profile: <https://www.learn-in-depth.com/online-diploma/meshoadel2018%40gmail.com>*

## Project Description:

It is required to make a student management system using FIFO and add the following options for the user to use:

1. Add the students details manually
2. Add the student details from text file
3. Find the student details by Roll number
4. Find the student details by first name
5. Find the student details by course
6. Find the total number of students
7. Delete the students' details by roll number
8. Update the student's details by roll number
9. Show all information

## Code:

In main.c file we just call the system to start

```
20 * main.c
7 #include "students.h"
8
9 int main() {
10 |     start_system();
11 |     return 0;
12 }
```

Then we use the FIFO driver we implemented before

```
8 #ifndef FIFO_H_
9 #define FIFO_H_
10
11 #define Pprintf(...)    {fflush(stdin);\
12                          fflush(stdout);\
13                          printf(__VA_ARGS__);\
14                          fflush(stdin);\
15                          fflush(stdout);}
16
17 #include "stdio.h"
18 #include "students.h"
19 // Define Data Type
20
21 #define DataType student_struct
22
23 // FIFO components
24 typedef struct {
25     int length;
26     int count;
27     DataType* base;
28     DataType* head;
29     DataType* tail;
30 }FIFO_Queue;
```

```

31 //FIFO error
32 typedef enum {
33     FIFO_NO_ERROR,
34     FIFO_FULL,
35     FIFO_EMPTY,
36     FIFO_NULL,
37     FIFO_NOT_ENOUGH_ITEMS
38 } E_FIFO_RETURN;
39
40 //User Functions
41 E_FIFO_RETURN FIFO_init(FIFO_Queue* My_Queue,DataType* P_Queue,int length); //initialize the FIFO
42 E_FIFO_RETURN FIFO_enqueue(FIFO_Queue* My_Queue,DataType *item); //adds one items to FIFO
43 E_FIFO_RETURN FIFO_dequeue(FIFO_Queue* My_Queue,DataType *store_P); //pop one item from queue and stores it in address
44 E_FIFO_RETURN FIFO_empty(FIFO_Queue* My_Queue); //Checks if queue is empty
45 E_FIFO_RETURN FIFO_full(FIFO_Queue* My_Queue); //Check if queue is full
46 E_FIFO_RETURN FIFO_reset(FIFO_Queue* My_Queue); //resets the FIFO
47 E_FIFO_RETURN FIFO_display(FIFO_Queue* My_Queue,int num); //Display last n items
48 E_FIFO_RETURN FIFO_display_all(FIFO_Queue* My_Queue); //Display all items
49
50
51
52 #endif /* FIFO_H_ */

```

## The System driver (students.h file):

```

2* * students.h
7
8 #ifndef STUDENTS_H_
9 #define STUDENTS_H_
10
11 #define DataType student_struct
12
13 typedef struct student_info{
14     char first_name[50];
15     char last_name[50];
16     unsigned int roll;
17     float gpa;
18     int course_id[10];
19 } student_struct;
20
21 void start_system();
22 void display_student(DataType *ptr);
23 void add_student_manually();
24 void add_student_file();
25 void find_student_roll();
26 void find_student_name();
27 void find_student_course();
28 void delete_student_roll();
29 void update_student_roll();
30
31 #endif /* STUDENTS_H_ */
32

```

# The System driver (students.c file):

We start with includes macros and queue initialization:

```
2* " students.c
7 #include "students.h"
8 #include "FIFO.h"
9 #include "stdio.h"
10 #include "stdlib.h"
11 #include <string.h>
12
13 #define system_size 50 //size of queue
14
15 FIFO_Queue queue1;
16 DataType arr[system_size]={0};
17
```

Then function to start the system:

```
18 void start_system(){
19     FIFO_init(&queue1, arr, system_size);
20     int task=0;
21     Pprintf("=====\\n"
22         "Welcome to student management system\\n");
23     while(1){
24         Pprintf("Choose the task you want to perform\\n"
25             "1. Add the students details manually\\n"
26             "2. Add the student details from text file\\n"
27             "3. Find the student details by Roll number\\n"
28             "4. Find the student details by first name\\n"
29             "5. Find the student details by course\\n"
30             "6. Find the total number of students\\n"
31             "7. Delete the students details by roll number\\n"
32             "8. Update the students details by roll number\\n"
33             "9. Show all information\\n"
34             "10. To exit\\n"
35             "Enter your choice to perform the task : ");
36         scanf("%d", &task);
37         Pprintf("=====\\n");
38         switch(task)
39         {
40             case 1:
41                 add_student_manually();
42                 break;
43             case 2:
44                 add_student_file();
45                 break;
46             case 3:
47                 find_student_roll();
48                 break;
49             case 4:
50                 find_student_name();
51                 break;
52             case 5:
53                 find_student_course();
54                 break;
55             case 6:
56                 Pprintf("=====\\n"
57                     "[INFO] Total number of students = %d\\n"
58                     "[INFO] You can add %d more students\\n"
59                     "=====\\n", queue1.count, system_size-queue1.count);
60                 break;
61             case 7:
62                 delete_student_roll();
63                 break;
64             case 8:
65                 update_student_roll();
66                 break;
67             case 9:
68                 FIFO_display_all(&queue1);
69                 break;
70             case 10:
71                 return;
72                 break;
73             default:
74                 Pprintf("Wrong choice\\n");
75         }
76     }
77 }
78
```

## Function to display a student using a pointer to the student:

```
9=void display_student(DataType *ptr) //Function to print a student using pointer to this student
0 {
1     int i;
2     Pprintf("=====\\n"
3         "Student First Name : %s\\n"
4         "Student Last Name : %s\\n"
5         "Student Roll Number : %d\\n"
6         "Student GPA : %.2f\\n"
7         "Courses Registered : ",ptr->first_name,ptr->last_name,ptr->roll,ptr->gpa);
8     for(i=0;i<10 && ptr->course_id[i]!='\\0';i++)
9         Pprintf("%d ",ptr->course_id[i]);
0     Pprintf("\\n=====\\n");
1 }
~
```

## Function to check if a roll number exists in the system:

```
13=int existing_roll_number(FIFO_Queue* My_Queue,int num) // Function to check if roll number exists
14 {
15     int i;
16     DataType *My_Student=My_Queue->tail;
17     for(i=0;i<My_Queue->count+1;i++)
18     {
19         if(My_Student->roll==num)
20             return 1;
21         if(My_Student == (My_Queue->base + (My_Queue->length)*sizeof(DataType) ) ) // for circular queue
22             My_Student=My_Queue->base;
23         My_Student++; // to navigate between students
24     }
25     return 0;
26 }
```

## First function for the user to use is to add students from a text file:

```
108=void add_student_file() //to add students data from text file
109 {
110     FILE *ptr;
111     char temp_line[250];
112     char loop_char;
113     char temp_element[50];
114     int counter=0,i,j,k,prev_char=0;
115     ptr=fopen("file.txt","r");
116     if(ptr==NULL)
117     {
118         Pprintf("File can't be opened\\n");
119         return;
120     }
121     while(fgets(temp_line,250,ptr)) // to read each line and put it in var temp_line
122     {
123         DataType My_Student;
124         counter=0,prev_char=0;
125         i=0;
126         loop_char=temp_line[i];
127         while(loop_char!='\\n') // loop for every line
128         {
129             loop_char=temp_line[i];
130             if(loop_char==' ' && counter<5) // each space is the difference between each data type
131             {
132                 counter++;
133                 switch(counter)
134                 {
135                     case 1: // after first space we read roll number
136                         for(j=0;j<(i-prev_char);j++)
137                             temp_element[j]=temp_line[j+prev_char]; //copy from prev char to the char before space
138                         temp_element[j]=' ';
139                         prev_char=i+1;
140                         My_Student.roll=atoi(temp_element);
141                         break;
142                     case 2: // second space we read first name
143                         for(j=0;j<(i-prev_char);j++)
```

```

144         temp_element[j]=temp_line[j+prev_char];
145         temp_element[j]='\0';
146         prev_char=i+1;
147         strcpy(My_Student.first_name,temp_element);
148         break;
149     case 3: // then last name
150         for(j=0;j<(i-prev_char);j++)
151             temp_element[j]=temp_line[j+prev_char];
152         temp_element[j]='\0';
153         prev_char=i+1;
154         strcpy(My_Student.last_name,temp_element);
155         break;
156     case 4: // then gpa
157         for(j=0;j<(i-prev_char);j++)
158             temp_element[j]=temp_line[j+prev_char];
159         prev_char=i+1;
160         My_Student.gpa=atof(temp_element);
161         break;
162     case 5: // then courses
163         for(j=0;loop_char!='\n' && j<10;i++)
164         {
165             loop_char=temp_line[i];
166             if(loop_char==' ') // each space is the difference between courses
167             {
168                 for(k=0;k<(i-prev_char);k++)
169                 {
170                     temp_element[k]=temp_line[prev_char+k];
171                 }
172                 temp_element[k]=' ';
173                 My_Student.course_id[j]=atoi(temp_element);
174                 j++;
175                 prev_char=i+1;
176             }
177         }
178         if(loop_char=='\n') // to get last course
179         {
180             for(k=0;k<(i-prev_char);k++)
181             {
182                 temp_element[k]=temp_line[prev_char+k];
183             }
184             temp_element[k]=' ';
185             My_Student.course_id[j]=atoi(temp_element);
186             My_Student.course_id[j+1]='\0';
187         }
188         break;
189     }
190     i++;
191 }
192 }
193 if(exsisting_roll_number(&queue1,My_Student.roll)) // if roll number exists do.t save
194 {
195     Pprintf("[ERROR] Not saved successfully because roll %d number exists\n",My_Student.roll);
196 }
197 else if(FIFO_full(&queue1)==FIFO_FULL) // if queue full don.t save
198 {
199     Pprintf("[ERROR] Not Saved Successfully because system is full\n");
200 }
201 else // add student to the queue
202 {
203     FIFO_enqueue(&queue1,&My_Student);
204     Pprintf("Saved Successfully :%s\n",My_Student.first_name);
205 }
206 }
207 Pprintf("===== \n");
208 "[INFO] Total number of students = %d\n"
209 "[INFO] You can add %d more students\n"
210 "===== \n",queue1.count,system_size-queue1.count);
211 }
212

```

## Function to add students manually:

```

213 void add_student_manually()
214 {
215     if(FIFO_full(&queue1)==FIFO_FULL)
216     {
217         Pprintf("[ERROR] Not Saved Successfully because system is full\n");
218         return;
219     }
220     DataType My_Student;
221     Pprintf("Enter the student Roll Number: ");
222     scanf("%d",&My_Student.roll);
223     if(exsisting_roll_number(&queue1,My_Student.roll))
224     {
225         Pprintf("[ERROR] Not saved successfully because roll %d number exists\n",My_Student.roll);
226         return;
227     }
228     Pprintf("Enter the student first name: ");
229     scanf("%s",My_Student.first_name);
230     Pprintf("Enter the student last name: ");
231     scanf("%s",My_Student.last_name);
232     Pprintf("Enter the student GPA: ");
233     scanf("%f",&(My_Student.gpa));
234     int x,i=0;
235     Pprintf("Enter your registered courses IDs\n");
236     while(i<10)
237     {
238         Pprintf("Enter the course ID or enter 0 if you are done: ");
239         scanf("%d",&x);
240         if(x==0)
241             break;
242         My_Student.course_id[i]=x;
243         i++;
244     }
245     if(i<9)
246         My_Student.course_id[i]='\0';
247     FIFO_enqueue(&queue1,&My_Student);
248     Pprintf("Saved Successfully :%s\n",My_Student.first_name);
249     Pprintf("===== \n");
250     "[INFO] Total number of students = %d\n"
251     "[INFO] You can add %d more students\n"
252     "===== \n",queue1.count,system_size-queue1.count);
253 }

```

## Function to find student by roll number:

```
254
255 void find_student_roll()
256 {
257     int num;
258     Pprintf("Enter the Roll Number you want to search for: ");
259     scanf("%d", &num);
260     int i;
261     DataType *My_Student=queue1.tail;
262     for(i=0; i<queue1.count+1; i++)
263     {
264         if(My_Student->roll==num)
265         {
266             display_student(My_Student);
267             return;
268         }
269         if(My_Student == (queue1.base + (queue1.length)*sizeof(DataType) ) )
270             My_Student=queue1.base;
271         My_Student++;
272     }
273     Pprintf("Roll Number not found\n");
274 }
275
```

## Function to find a student by its first name:

```
276 void find_student_name()
277 {
278     int found=0;
279     char first_name[50];
280     Pprintf("Enter the Student First Name you want to search for: ");
281     scanf("%s", first_name);
282     int i;
283     DataType *My_Student=queue1.tail;
284     for(i=0; i<queue1.count+1; i++)
285     {
286         if(strcmp(My_Student->first_name, first_name)==0)
287         {
288             display_student(My_Student);
289             found=1;
290         }
291         if(My_Student == (queue1.base + (queue1.length)*sizeof(DataType) ) )
292             My_Student=queue1.base;
293         My_Student++;
294     }
295     if(found==0)
296         Pprintf("First Name not found\n");
297 }
298
```

## Function to find students registered a certain course:

```
297 void find_student_course()
298 {
299     int course;
300     int found=0;
301     Pprintf("Enter the course ID you want to search for: ");
302     scanf("%d", &course);
303     int i, j;
304     DataType *My_Student=queue1.tail;
305     for(i=0; i<queue1.count+1; i++)
306     {
307         for(j=0; j<10 && My_Student->course_id[j]!='\0'; j++)
308         {
309             if(My_Student->course_id[j]==course)
310             {
311                 display_student(My_Student);
312                 found=1;
313             }
314         }
315         if(My_Student == (queue1.base + (queue1.length)*sizeof(DataType) ) )
316             My_Student=queue1.base;
317         My_Student++;
318     }
319     if(found==0)
320         Pprintf("Course ID not found\n");
321 }
322
```

## Function to delete student using roll number:

```
323=void delete_student_roll()
324 {
325     int num;
326     Pprintf("Enter the Roll Number you want to delete : ");
327     scanf("%d",&num);
328     int i;
329     DataType *My_Student=queue1.tail;
330     for(i=0;i<queue1.count+1;i++)
331     {
332         if(My_Student->roll==num)
333         {
334             DataType *Temp_p=My_Student;
335             while(My_Student!=queue1.head)
336             {
337                 Temp_p++;
338                 *(My_Student)=*(Temp_p);
339                 if(My_Student == (queue1.base + (queue1.length)*sizeof(DataType) ) )
340                     My_Student=queue1.base;
341                 My_Student++;
342             }
343             queue1.count--;
344             queue1.head--;
345             return;
346         }
347         if(My_Student == (queue1.base + (queue1.length)*sizeof(DataType) ) )
348             My_Student=queue1.base;
349         My_Student++;
350     }
351     Pprintf("Roll Number not found\n");
352 }
353
```

## Function to update student by roll number:

```
354=void update_student_roll()
355 {
356     int num;
357     Pprintf("Enter the Roll Number you want to update: ");
358     scanf("%d",&num);
359     int i;
360     int x;
361     DataType *My_Student=queue1.tail;
362     for(i=0;i<queue1.count+1;i++)
363     {
364         if(My_Student->roll==num)
365         {
366             int upd;
367             Pprintf("Enter the section you want to update: \n"
368                 "1. First Name\n"
369                 "2. Last Name\n"
370                 "3. Roll Number\n"
371                 "4. GPA\n"
372                 "5. Courses registered\n"
373                 "Enter your choice: ");
374             scanf("%d",&upd);
375             switch(upd)
376             {
377                 case 1:
378                     Pprintf("Enter the student first name: ");
379                     scanf("%s",My_Student->first_name);
380                     break;
381                 case 2:
382                     Pprintf("Enter the student last name: ");
383                     scanf("%s",My_Student->last_name);
384                     break;
385                 case 3:
386                     Pprintf("Enter the student Roll Number: ");
387                     scanf("%d",&(My_Student->roll));
388                     break;
389                 case 4:
390                     Pprintf("Enter the student GPA: ");
391                     scanf("%f",&(My_Student->gpa));
392                     break;
393                 case 5:
394                     Pprintf("Enter your registered courses IDs\n");
395                     i=0;
396                     while(i<10)
397                     {
398                         Pprintf("Enter the course ID or enter 0 if you are done: ");
399                         scanf("%d",&x);
400                         if(x==0)
401                             break;
402                         My_Student->course_id[i]=x;
403                         i++;
404                     }
405                     if(i<9)
406                         My_Student->course_id[i]='\0';
407                     break;
408                 default:
409                     Pprintf("Wrong Choice");
410             }
411             Pprintf("Saved Successfully :%s\n",My_Student->first_name);
412             Pprintf("=====\n"
413                 "[INFO] Total number of students = %d\n"
414                 "[INFO] You can add %d more students\n"
415                 "=====\n",queue1.count,system_size-queue1.count);
416             return;
417         }
418         if(My_Student == (queue1.base + (queue1.length)*sizeof(DataType) ) )
419             My_Student=queue1.base;
420         My_Student++;
421     }
422     Pprintf("Roll Number not found\n");
423 }
```



## System Running:

First, we try to add a student manually:

```
Welcome to student management system
Choose the task you want to perform
1. Add the students details manually
2. Add the student details from text file
3. Find the student details by Roll number
4. Find the student details by first name
5. Find the student details by course
6. Find the total number of students
7. Delete the students details by roll number
8. Update the students details by roll number
9. Show all information
10. To exit
Enter your choice to perform the task : 1

=====
Enter the student Roll Number: 7
Enter the student first name: Misho
Enter the student last name: Adel
Enter the student GPA: 4
Enter your registered courses IDs
Enter the course ID or enter 0 if you are done: 1
Enter the course ID or enter 0 if you are done: 22
Enter the course ID or enter 0 if you are done: 33
Enter the course ID or enter 0 if you are done: 44
Enter the course ID or enter 0 if you are done: 55
Enter the course ID or enter 0 if you are done: 5
Enter the course ID or enter 0 if you are done: 66
Enter the course ID or enter 0 if you are done: 0
Saved Successfully :Misho

=====
[INFO] Total number of students = 1
[INFO] You can add 49 more students
=====
```

Then we add another student:

```
Enter your choice to perform the task : 1

=====
Enter the student Roll Number: 2
Enter the student first name: Marco
Enter the student last name: Mina
Enter the student GPA: 3.5
Enter your registered courses IDs
Enter the course ID or enter 0 if you are done: 11
Enter the course ID or enter 0 if you are done: 12
Enter the course ID or enter 0 if you are done: 13
Enter the course ID or enter 0 if you are done: 14
Enter the course ID or enter 0 if you are done: 15
Enter the course ID or enter 0 if you are done: 16
Enter the course ID or enter 0 if you are done: 17
Enter the course ID or enter 0 if you are done: 18
Enter the course ID or enter 0 if you are done: 19
Enter the course ID or enter 0 if you are done: 20
Saved Successfully :Marco

=====
[INFO] Total number of students = 2
[INFO] You can add 48 more students
=====
```

Now we try to add from the following text file:

```
1 Marco Magdy 3.5 1 2 3 4 5
1 Pavly Salah 3 80 12 37 29 63
3 Bolis Karam 3.5 45 21 55 18 46
4 Kerolos Gamal 3.5 452 213 5 18 46134
```

```
=====
Enter your choice to perform the task : 2

=====
Saved Successfully :Marco
[ERROR] Not saved successfully because roll 1 number exists
Saved Successfully :Bolis
Saved Successfully :Kerolos

=====
[INFO] Total number of students = 5
[INFO] You can add 45 more students
=====
```

We can notice that it counts the students and checks for existing roll number correctly.

Now we try to display all students:

```
student_mangement_system.exe (C:\C++ Application\Un\Embedded_Systems\Udiploma\Github\Master-Embedded-System-Udiploma\Unit0_First_Term_Project\Second Project\Debug\student_mangement_system.exe (8/9/23, 3:43 PM))
Enter your choice to perform the task : 2
=====
Student First Name : Misho
Student Last Name : Adel
Student Roll Number : 7
Student GPA : 4.00
Courses Registered : 1 22 33 44 55 5 66
=====
Student First Name : Marco
Student Last Name : Mina
Student Roll Number : 2
Student GPA : 3.50
Courses Registered : 11 12 13 14 15 16 17 18 19 20
=====
Student First Name : Marco
Student Last Name : Magdy
Student Roll Number : 1
Student GPA : 3.50
Courses Registered : 1 2 3 4 5
=====
Student First Name : Bolis
Student Last Name : Karam
Student Roll Number : 3
Student GPA : 3.50
Courses Registered : 45 21 55 18 46
=====
Student First Name : Kerolos
Student Last Name : Gamal
Student Roll Number : 4
Student GPA : 3.50
Courses Registered : 452 213 5 18 46134
=====
```

Now we try the search by roll number:

```
Enter your choice to perform the task : 3
=====
Enter the Roll Number you want to search for: 1
=====
Student First Name : Marco
Student Last Name : Magdy
Student Roll Number : 1
Student GPA : 3.50
Courses Registered : 1 2 3 4 5
=====

Enter your choice to perform the task : 3
=====
Enter the Roll Number you want to search for: 7
=====
Student First Name : Misho
Student Last Name : Adel
Student Roll Number : 7
Student GPA : 4.00
Courses Registered : 1 22 33 44 55 5 66
=====

Enter your choice to perform the task : 3
=====
Enter the Roll Number you want to search for: 6
Roll Number not found
=====
```

## Now we try the search by first name:

```
Enter your choice to perform the task : 4
=====
Enter the Student First Name you want to search for: Marco
=====
Student First Name : Marco
Student Last Name : Mina
Student Roll Number : 2
Student GPA : 3.50
Courses Registered : 11 12 13 14 15 16 17 18 19 20
=====
Student First Name : Marco
Student Last Name : Magdy
Student Roll Number : 1
Student GPA : 3.50
Courses Registered : 1 2 3 4 5
=====

--- --
Enter your choice to perform the task : 4
=====
Enter the Student First Name you want to search for: Misho
=====
Student First Name : Misho
Student Last Name : Adel
Student Roll Number : 7
Student GPA : 4.00
Courses Registered : 1 22 33 44 55 5 66
=====

Enter your choice to perform the task : 4
=====
Enter the Student First Name you want to search for: misho
First Name not found
```

## Now we try the search for student with course ID:

```
Enter your choice to perform the task : 5
=====
Enter the course ID you want to search for: 5
=====
Student First Name : Misho
Student Last Name : Adel
Student Roll Number : 7
Student GPA : 4.00
Courses Registered : 1 22 33 44 55 5 66
=====
Student First Name : Marco
Student Last Name : Magdy
Student Roll Number : 1
Student GPA : 3.50
Courses Registered : 1 2 3 4 5
=====
Student First Name : Kerolos
Student Last Name : Gamal
Student Roll Number : 4
Student GPA : 3.50
Courses Registered : 452 213 5 18 46134
=====

Enter your choice to perform the task : 5
=====
Enter the course ID you want to search for: 7
Course ID not found
```

## Now we try the info function:

```
Enter your choice to perform the task : 6
=====
[INFO] Total number of students = 5
[INFO] You can add 45 more students
=====
```

## Now we try the delete function:

```
Enter your choice to perform the task : 7
=====
Enter the Roll Number you want to delete : 3
=====

Enter your choice to perform the task : 9
=====
Student First Name : Misho
Student Last Name : Adel
Student Roll Number : 7
Student GPA : 4.00
Courses Registered : 1 22 33 44 55 5 66
=====
Student First Name : Marco
Student Last Name : Mina
Student Roll Number : 2
Student GPA : 3.50
Courses Registered : 11 12 13 14 15 16 17 18 19 20
=====
Student First Name : Marco
Student Last Name : Magdy
Student Roll Number : 1
Student GPA : 3.50
Courses Registered : 1 2 3 4 5
=====
Student First Name : Kerolos
Student Last Name : Gamal
Student Roll Number : 4
Student GPA : 3.50
Courses Registered : 452 213 5 18 46134
=====
```

## Now we try the update function:

```
Enter your choice to perform the task : 8
=====
Enter the Roll Number you want to update: 1
Enter the section you want to update:
1. First Name
2. Last Name
3. Roll Number
4. GPA
5. Courses registered
Enter your choice: 4
Enter the student GPA: 3.77
Saved Successfully :Marco
=====
[INFO] Total number of students = 4
[INFO] You can add 46 more students
=====

Enter your choice to perform the task : 3
=====
Enter the Roll Number you want to search for: 1
=====
Student First Name : Marco
Student Last Name : Magdy
Student Roll Number : 1
Student GPA : 3.77
Courses Registered : 1 2 3 4 5
=====
```

This code is tested for several test cases and proved to be working just fine in all of them and test cases given in this report is just little demonstration of the system.