School of Computing

Faculty of Engineering

UNIVERSITI TEKNOLOGI MALAYSIA

DATA STRUCTURE & ALGORIHTMA (SECJ2013)

SEMESTER 1 2022/2023

Mini PROJECT Documentation
FINAL YEAR PROJECT MANAGEMENT

By
Yeon Chun Teck (A20EC0148) - Leader
Ng Kai Zheng (A20EC0101)
Amira Azkia Rahmatulloh (A20EC4062)

SECTION 09

Lecturer:
Dr. Johanna Binti Ahmad

Date:
25 January 2023

PART 1: INTRODUCTION

1.1 Synopsis Project

Finishing the final year project is a critical step in graduating from university and receiving a degree. Students in their final year must choose the field and subject of their project, and they are given a time limit to complete it and then graded in order to graduate.

The Final Year Project Management System is a digital system meant to aid in the management of final-year projects for both students and instructors by implementing a queue along with the linked list. Students can manage their final year projects with the system's assistance by receiving suggested fields for their final thesis, storing their information (name, matric number, course code, final year field and topics), tracking their progress, and overseeing the submission report.

Furthermore, a system feature allows lecturers to evaluate their final year project, check student information, and grade final reports. The system also provides real-time updates on project progress, which helps students stay on track and instructors monitor the development of their students.

Finally, this system intends to improve both students' and lecturers' project management processes, making them more efficient and lowering the administrative workload. Students can focus on their research and ensure that their projects are completed on time with the Final Year Project Management System, while supervisors can easily keep track of their students' progress and provide help as needed. Overall, this approach will aid in the improvement of final-year project quality while also making the project management process more manageable for all participants.

1.2 Objective of The Project

1. To provide students with an efficient and streamlined process for submitting, tracking, and managing their final-year projects.
2. To allow lecturers to review and approve project proposals, provide feedback on progress, and grade final reports easily.
3. To give real-time updates on project progress to both students and lecturers.
4. To make the project selection process more efficient by providing students with a list of suggested project fields.
5. To help the department keep track of the popular project fields that students are choosing for their final year projects.

6. To improve the overall quality of final-year projects and make the project management process more manageable for all stakeholders.
7. To provide the students with a reference for suggesting project fields to students who are struggling to find a topic.

PART 2: SYSTEM ANALYSIS AND DESIGN (USE CASE, FLOWCHART AND CLASS DIAGRAM)

2.1 System Requirement

2.1.1 Use Case Diagram

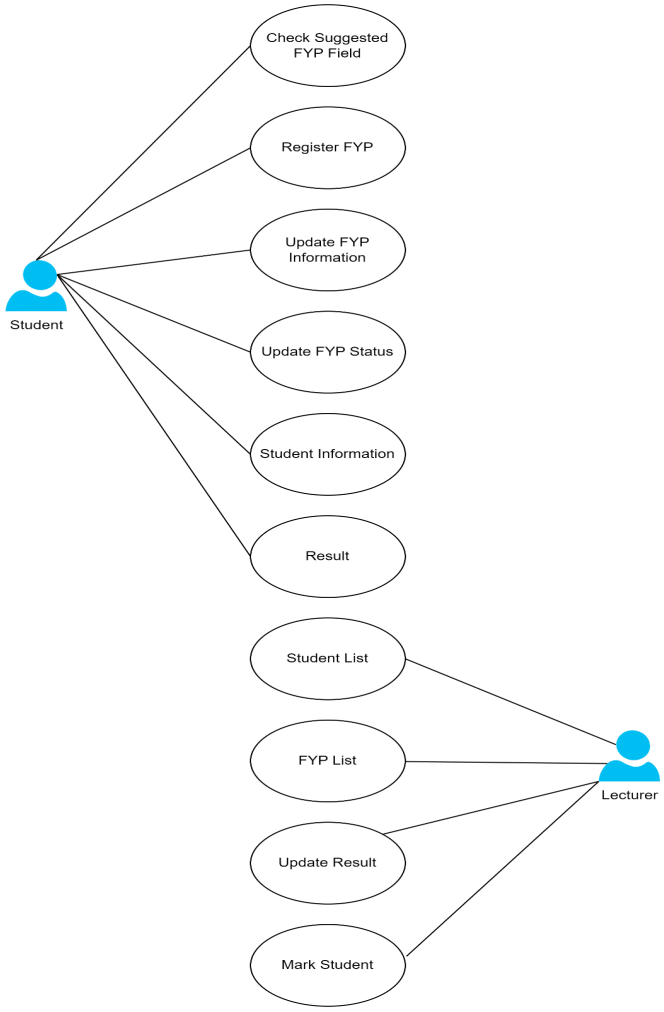


Figure 1: Use Case Diagram for Final Year Project Management System

2.1.2 Use Case Description

Actor/User	Task
Student	As a student, he can check suggested FYP Field for considering FYP title before deciding what FYP to do. In this system, he could register his FYP and update the information of FYP that includes the title and the field of FYP if he would like to modify FYP. He would also need to update the status of his FYP. Furthermore, he could check his information and the result of his FYP
Lecturer	As a lecturer, he could display a list of students who have records in the system and FYP list. Besides that, he could also mark an FYP of a student and update the result.

2.1.3 Detail Description

The system has 7 main use cases.

Use Case	Purpose
Check Suggested FYP Field	Choose the FYP Fields that are suggested in the system to view the information of the suggested FYP Field. Students could only register once.
Register FYP	Enter student name, student matric number, student courses, FYP title and FYP field to register FYP in the system. This case would save FYP Title Submission Date.
Update FYP Information	Modify FYP title and FYP field
Update FYP Status	Update the status of FYP according to the progress. The status includes “Title Submitted”, ”Proposal Submitted”, “Final Report Submitted”, and “Presentation Submitted”. The status of FYP couldn’t be updated if FYP had been marked
Student Information	To view the information of student and FYP including the status
Result	To view the result of the FYP.
Student List	To view a list of student information who have records in the system. The information includes student matric numbers, student names, student courses, FYP titles, FYP fields, FYP title submission dates and FYP status.
FYP List	To view a list of FYP information with corresponding student information. The information includes student matric numbers, student names, FYP titles, FYP status, and

	its mark.
Update Result	To update FYP results
Mark Student	To mark the students who have done presentations. The student who has done the presentation would be marked first.

2.2 System Design

\2.2.1 Class Diagram of System

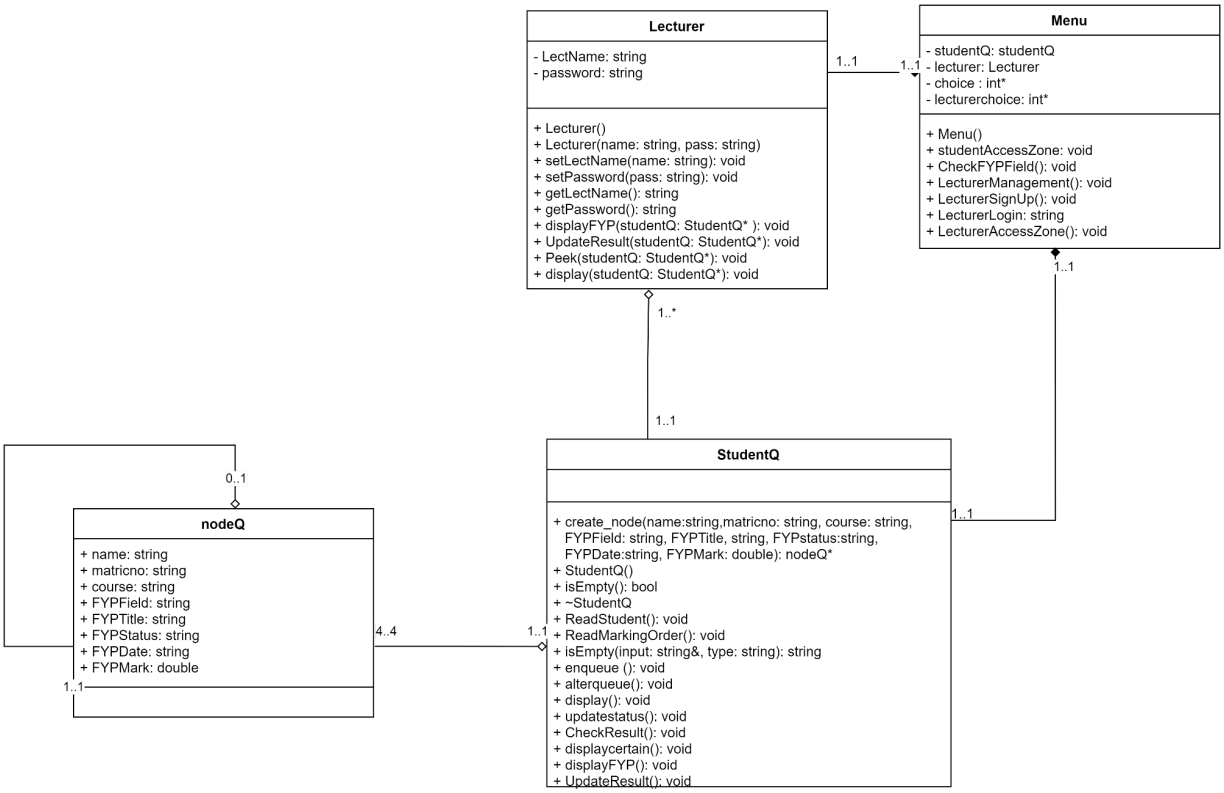


Figure 2 shows the class diagram of the Final Year Project Management System

2.2.2 Flowchart for Each Module/Task

Flowchart 1: Access menu for Student

Prepared By: Ng Kai Zheng

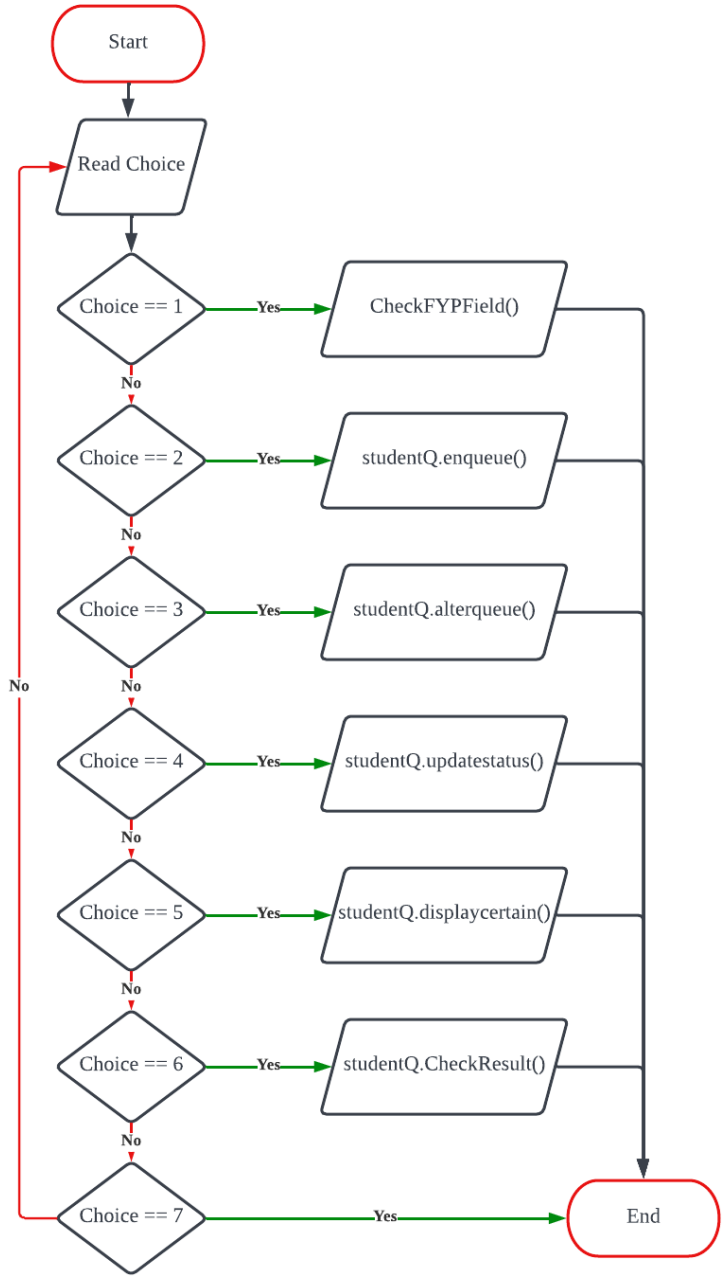


Figure 3 shows a flowchart of StudentAccessZone()

Flowchart 2: Access menu for Lecturer

Prepared By: Yeo Chun Teck

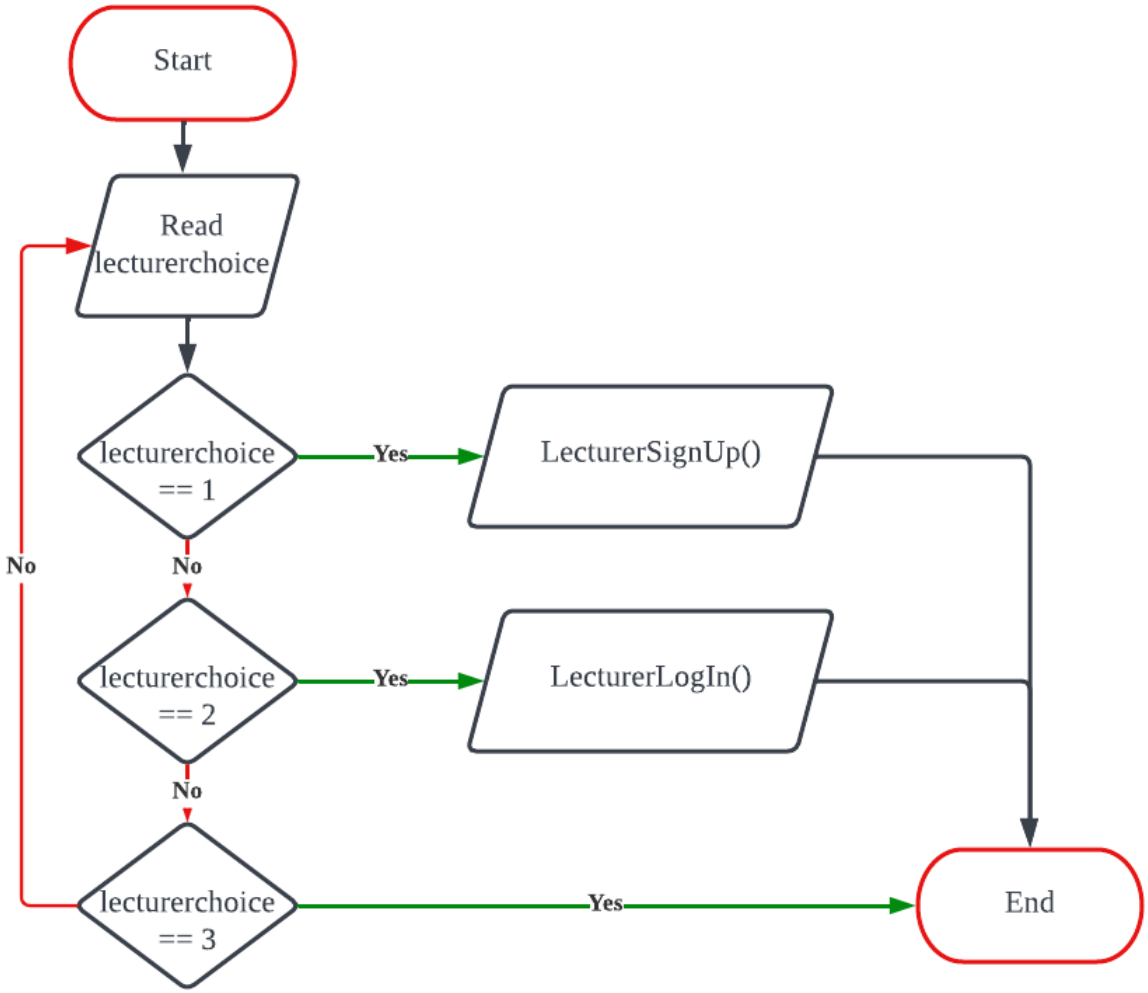


Figure 4 shows a flowchart of LecturerAccessZone()

Flowchart 3: Check suggested Final Year Project Field

Prepared By: Amira

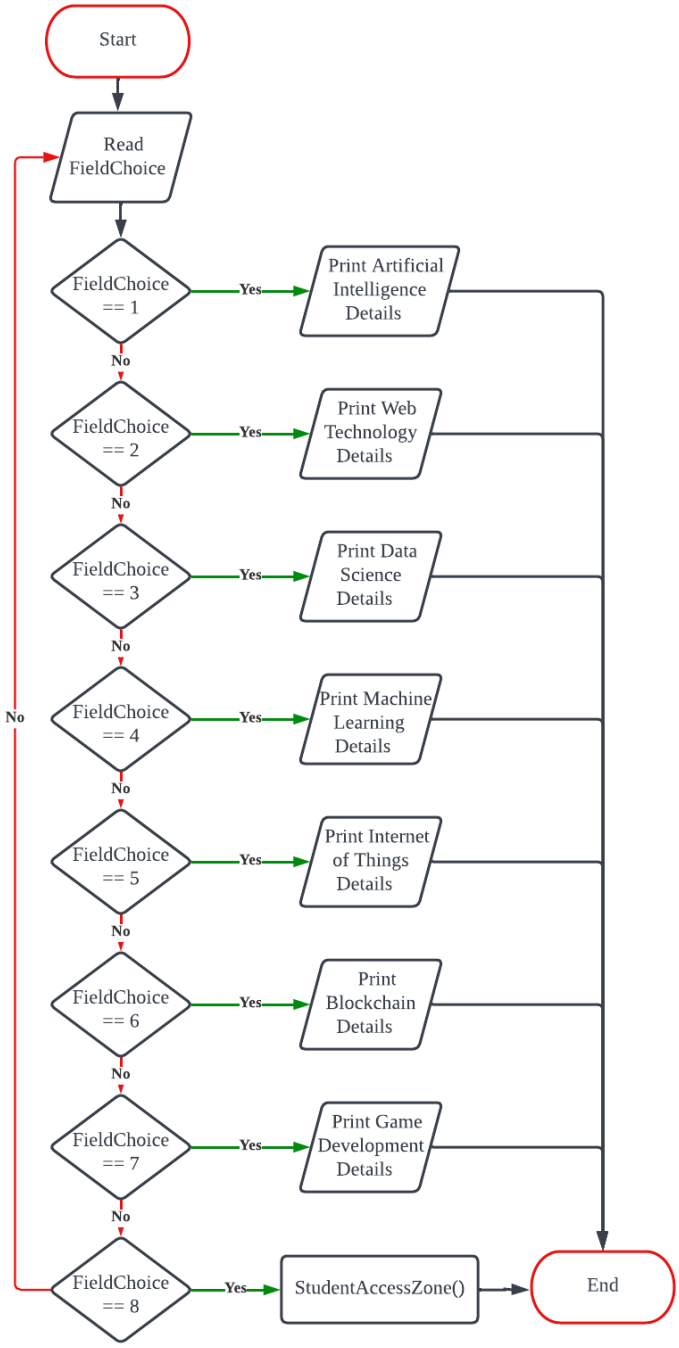


Figure 5 shows a flowchart of CheckFYPField()

Flowchart 4: Enter Final Year Project and Student Information

Prepared By: Ng Kai Zheng

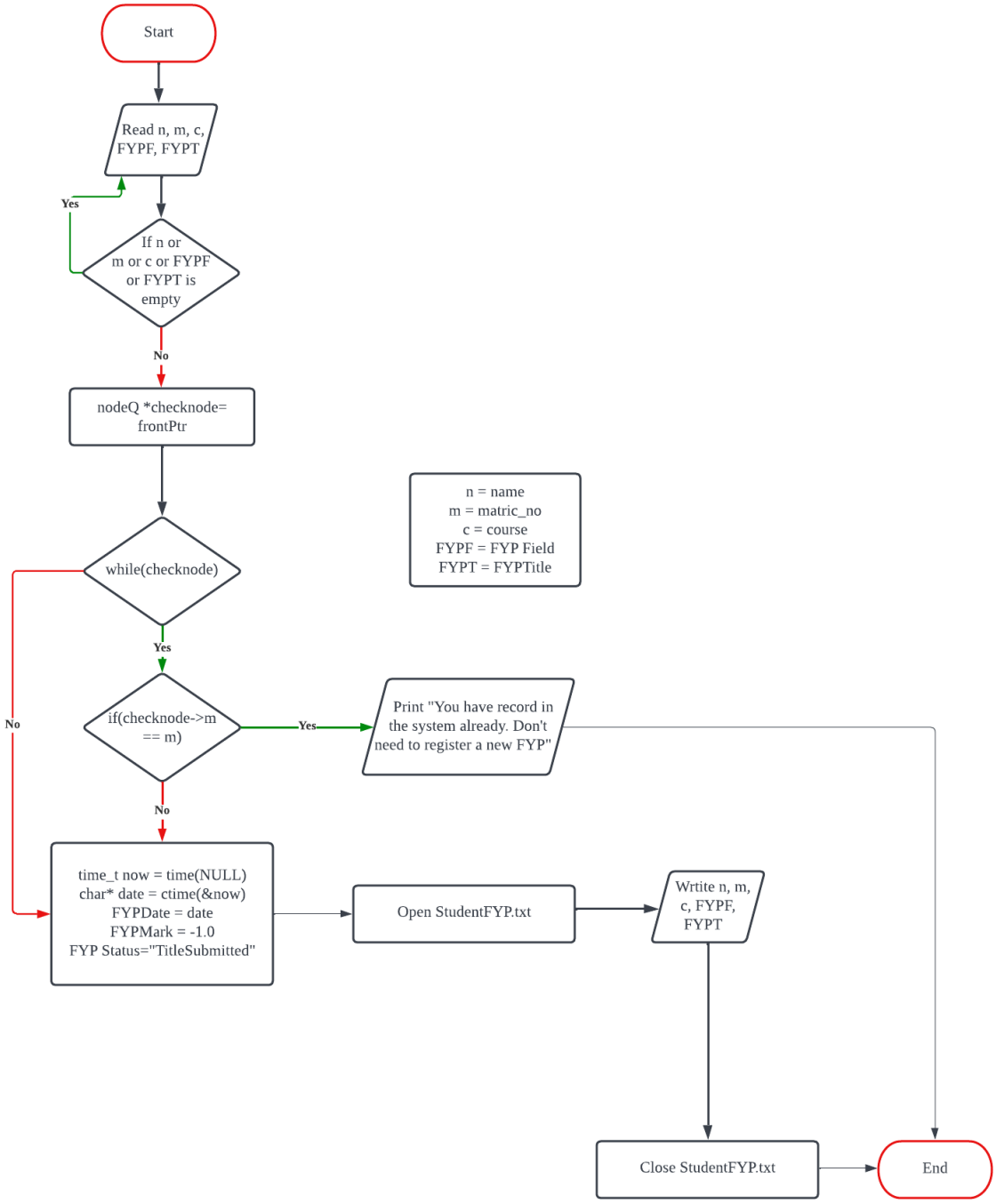


Figure 6 shows a flowchart of enqueue()

Flowchart 5: Update FYP Information

Prepared By: Yeo Chun Teck

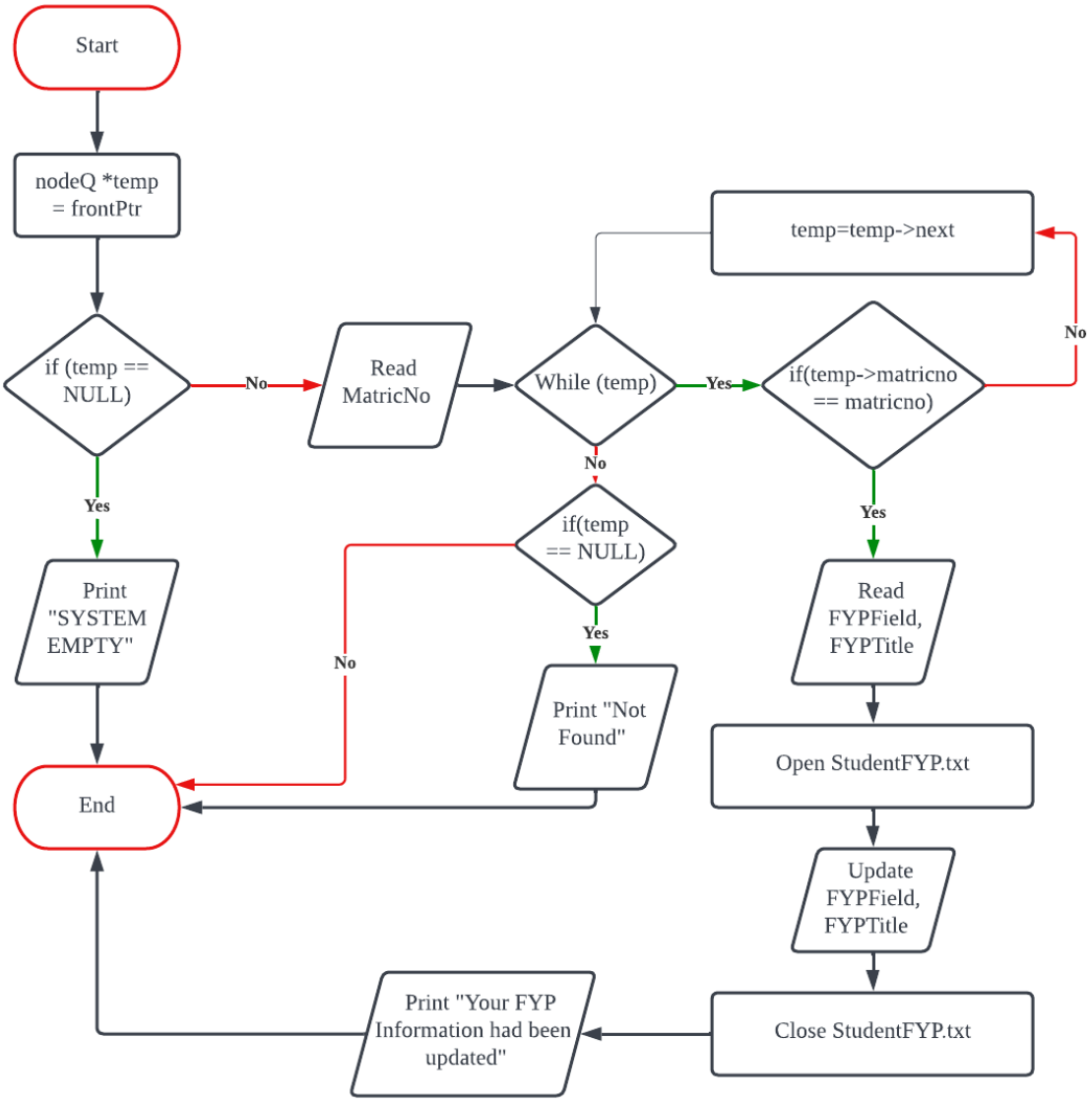


Figure 7 shows a flowchart of alterqueue()

Flowchart 6: Update FYP Status

Prepared By: Amira

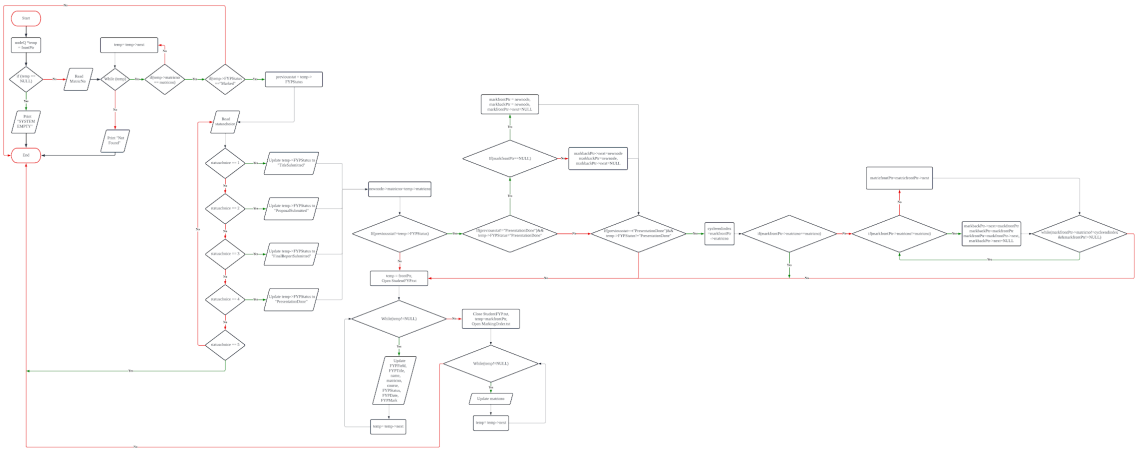


Figure 8 shows a flowchart updatestatus()

See More: https://lucid.app/lucidchart/26cc07f6-616d-45c3-aa6e-7094e9781987/edit?viewport_loc=-3178%2C-1817%2C5305%2C4932%2C0_0&invitationId=inv_34bc6602-6560-40ad-beea-b924ea42c29c

Flowchart 7: Display Certain Student Information

Prepared By: Ng Kai Zheng

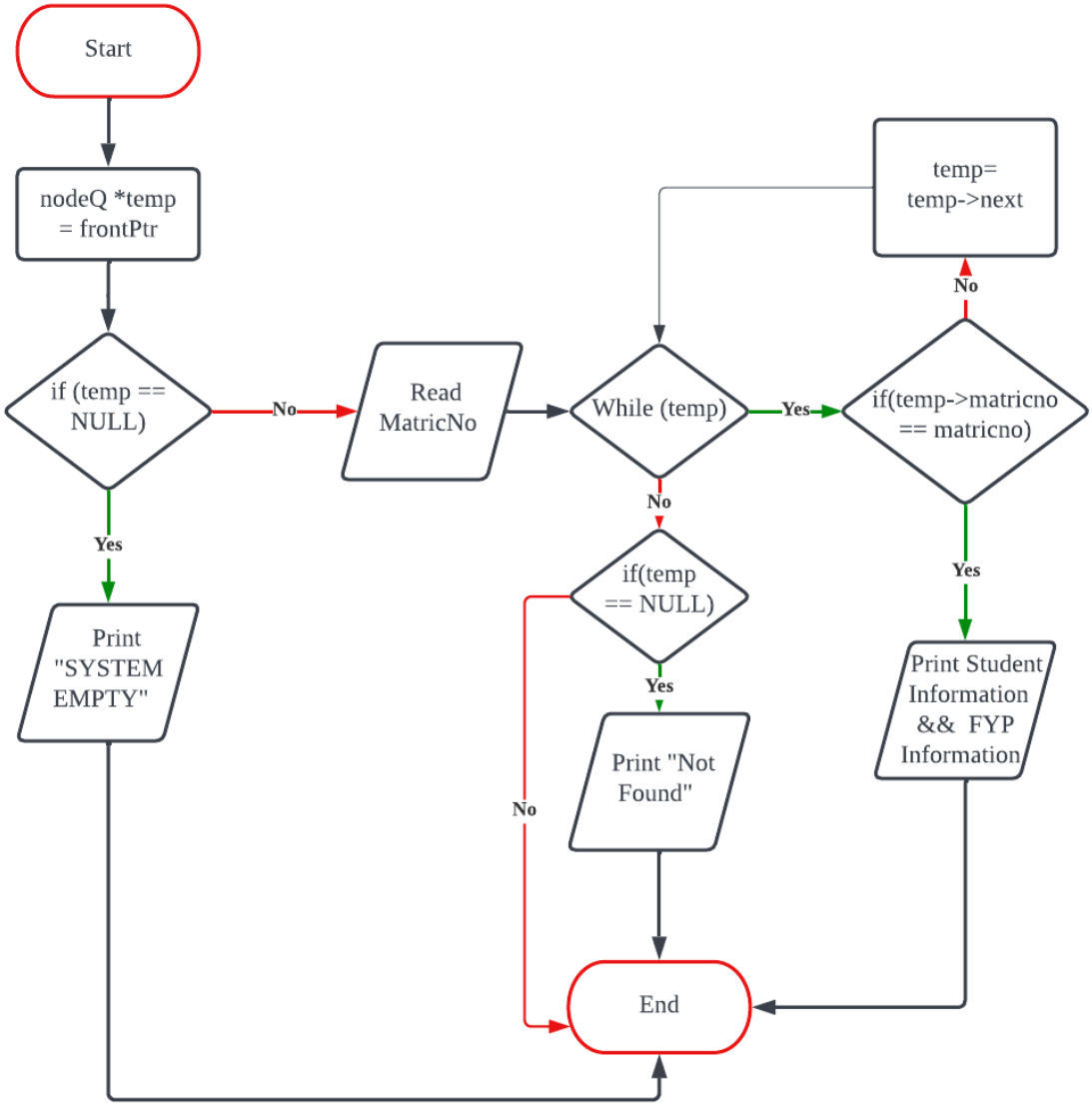


Figure 9 shows a flowchart of displaycertain()

Flowchart 8: Check Student FYP Result

Prepared By: Yeo Chu Teck

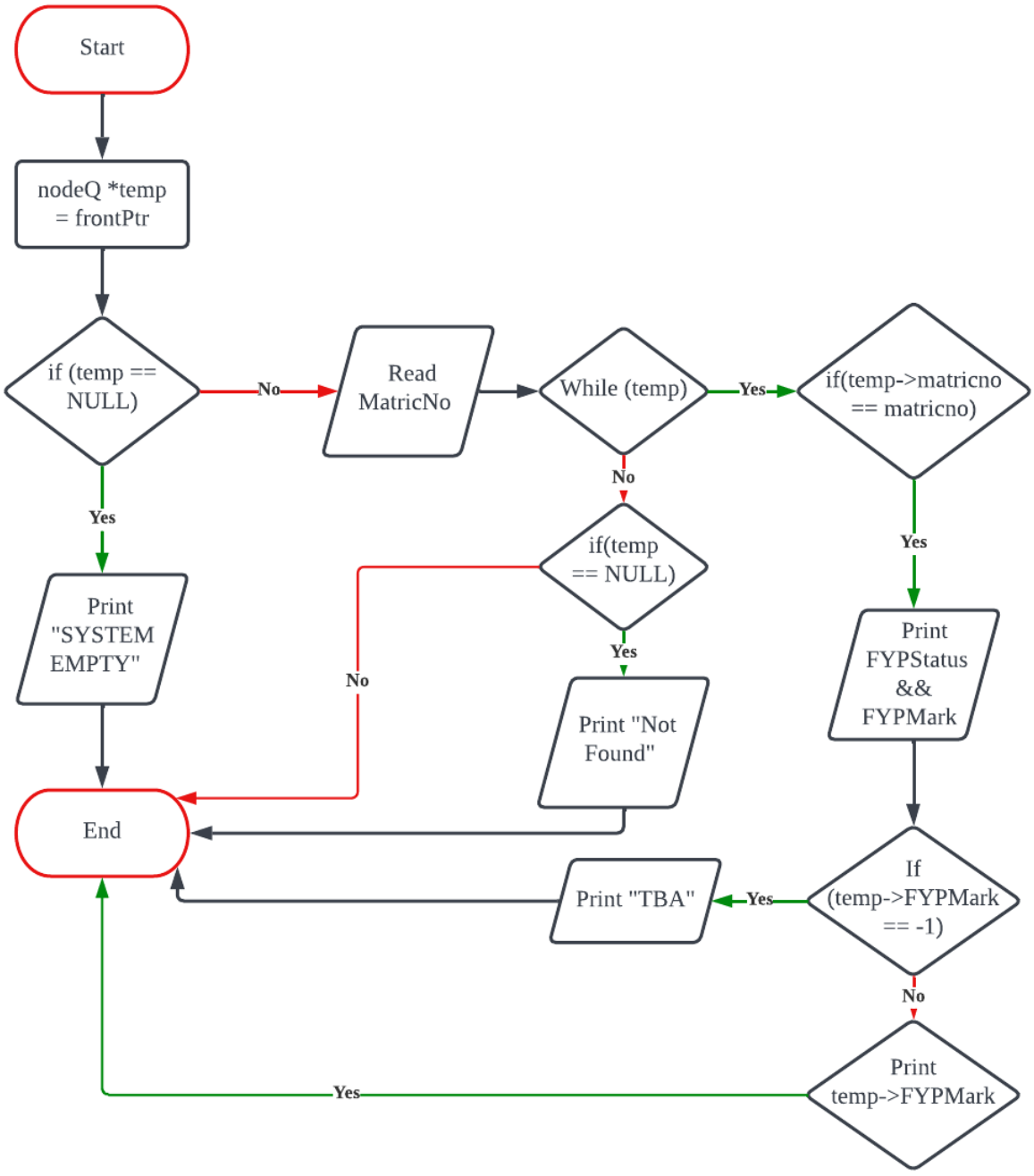


Figure 10 shows a flowchart of CheckResult()

Flowchart 9: Lecturer Sign Up

Prepared By: Amira

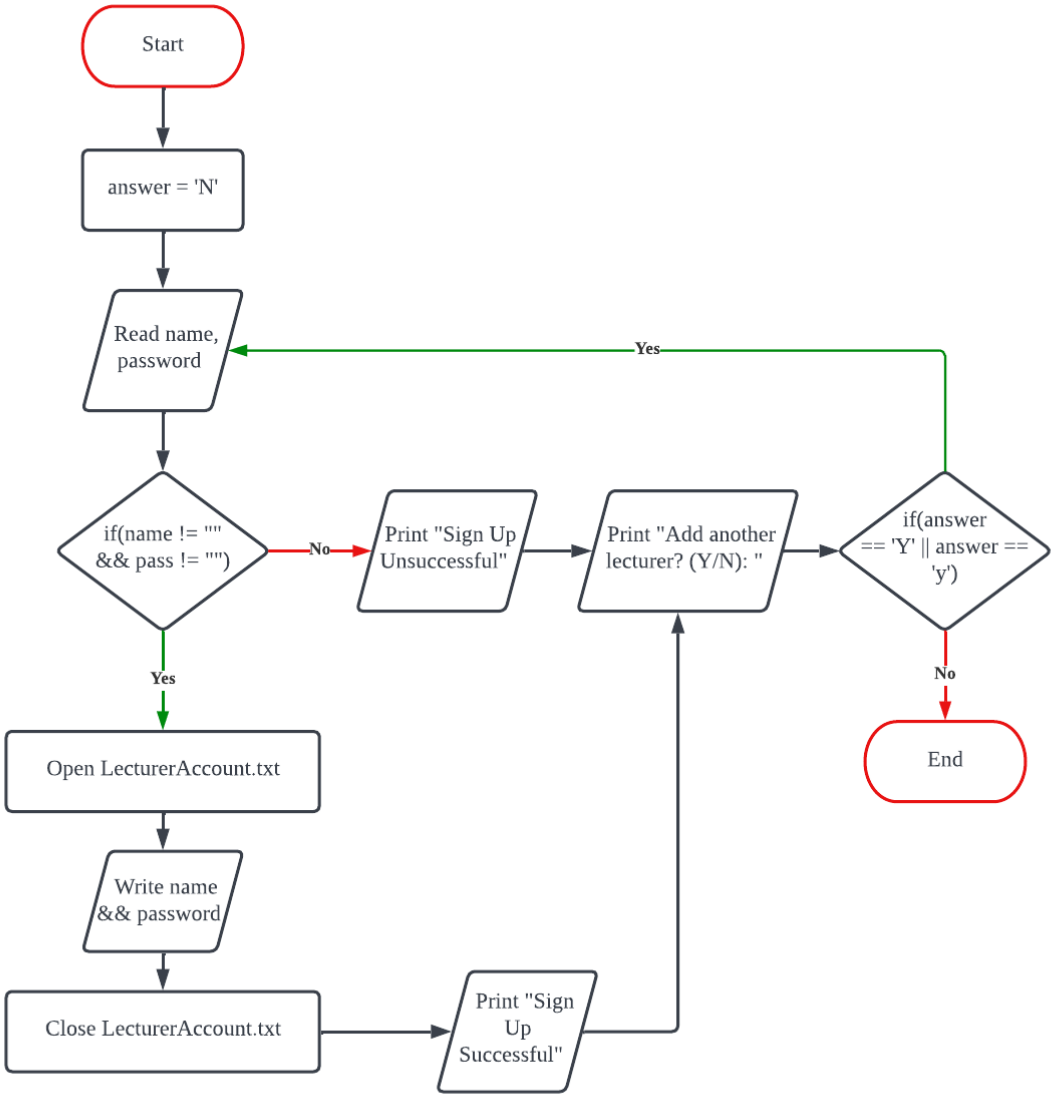


Figure 11 shows a flowchart of LecturerSignUp()

Flowchart 10: Lecturer Log In to system

Prepared By: Ng Kai Zheng

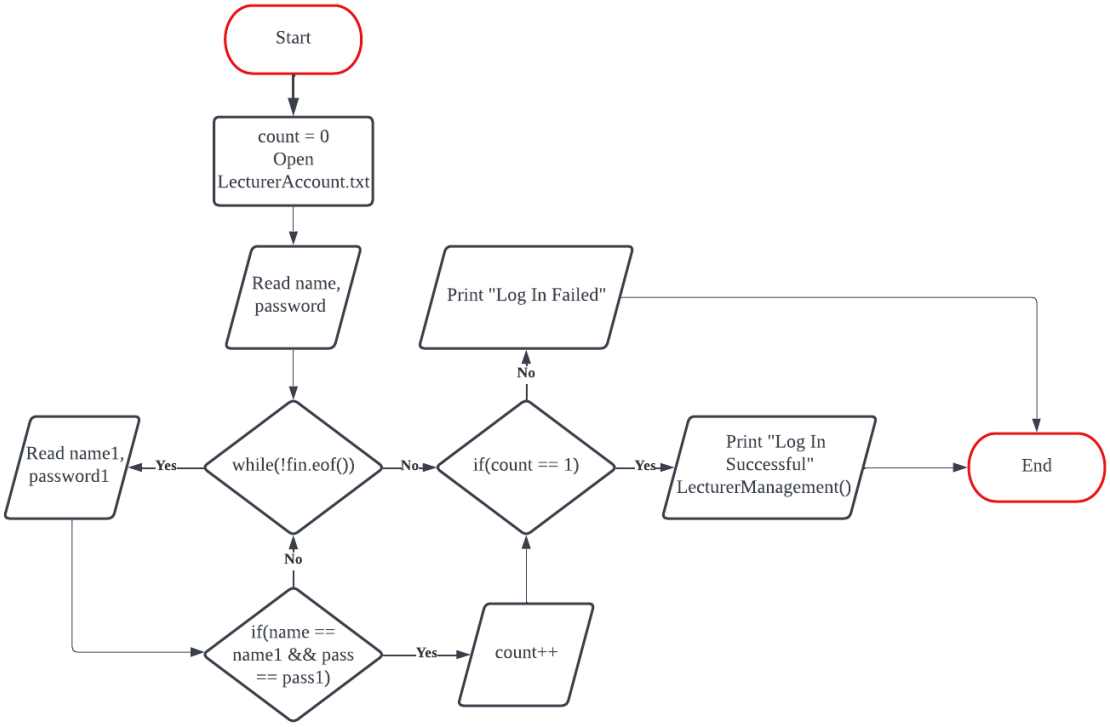


Figure 12 shows a flowchart of LecturerLogIn()

Flowchart 11: Lecturer Management Menu

Prepared By: Yeo Chun Teck

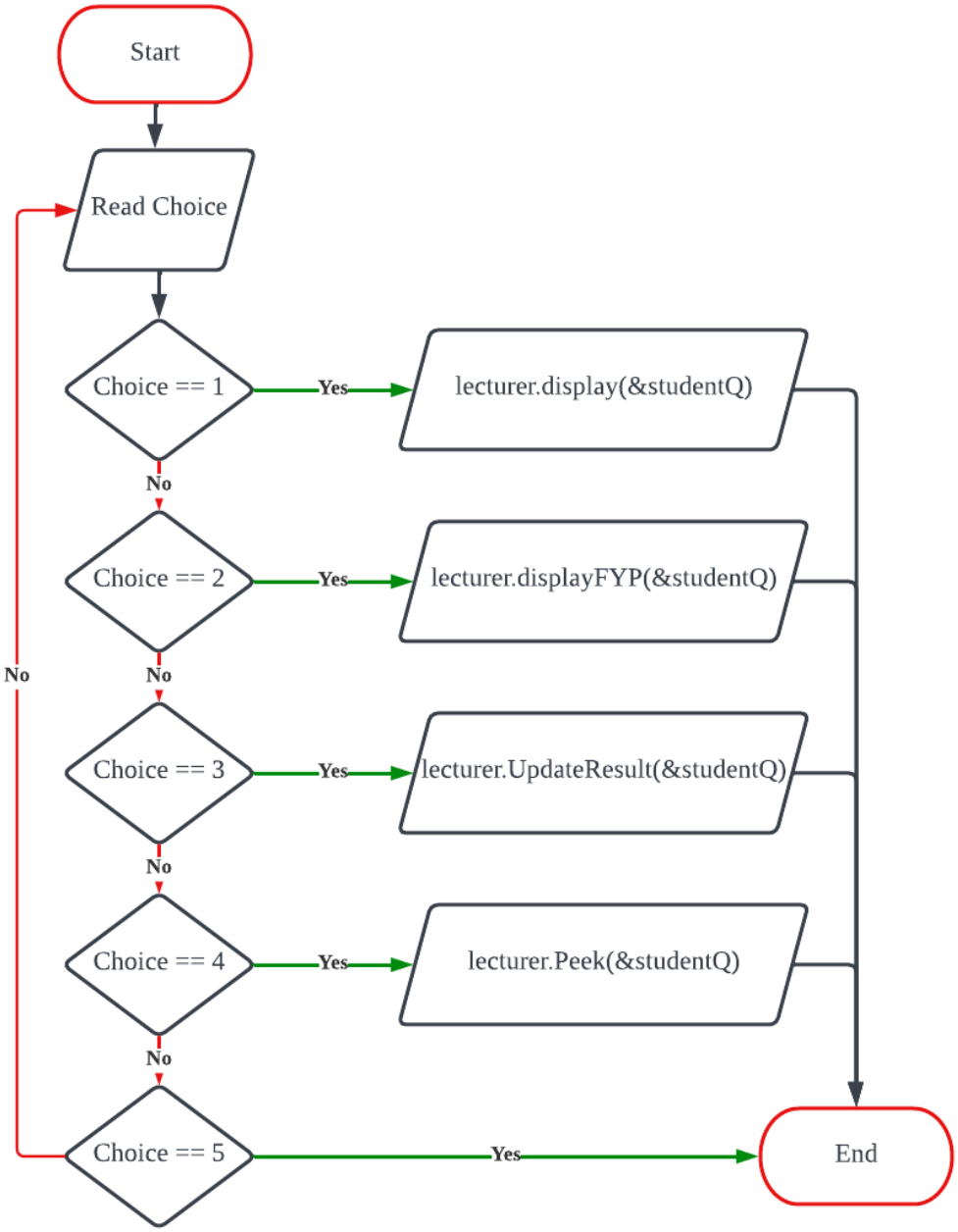


Figure 13 shows a flowchart of LecturerManagement()

Flowchart 12: Display Student List

Prepared By: Amira

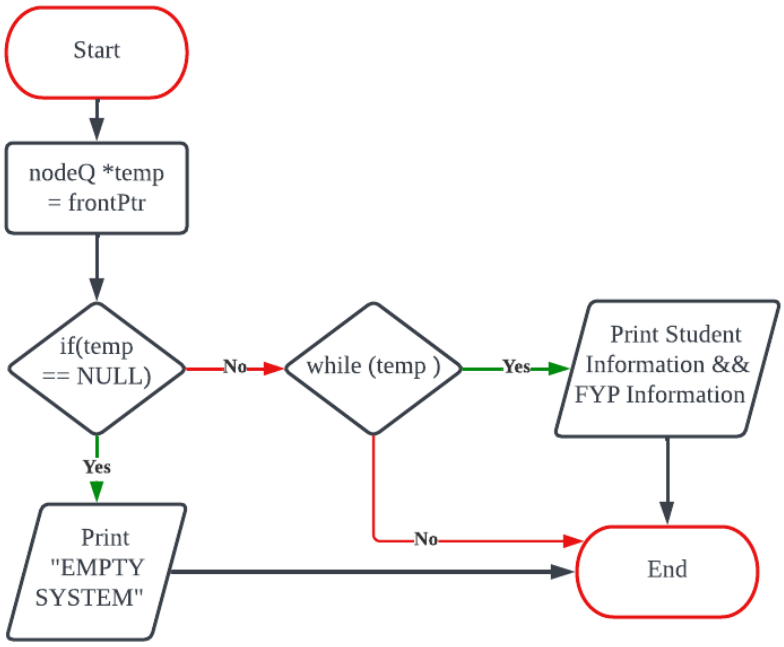


Figure 14 shows a flowchart of display()

Flowchart 13: Display FYP List

Prepared By: Ng Kai Zheng

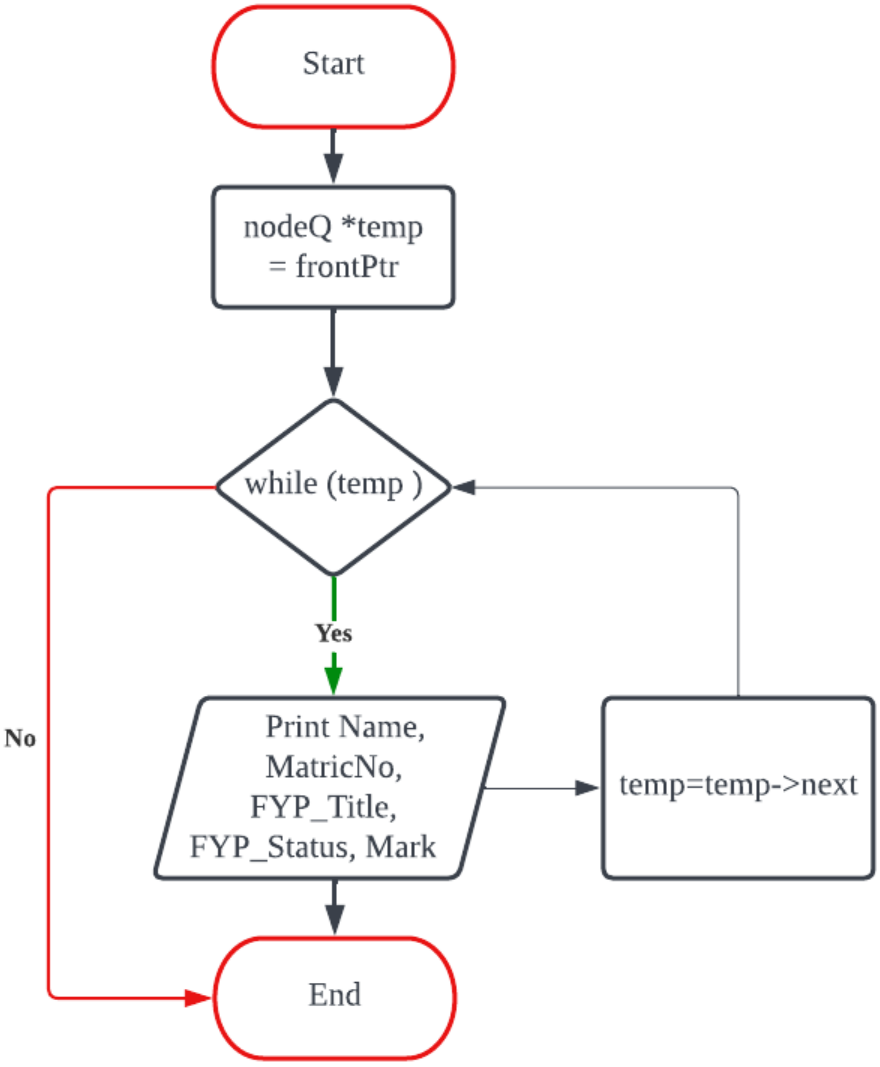


Figure 15 shows a flowchart of displayFYP()

Flowchart 14: Update Student FYP Mark

Prepared By: Yeo Chun Teck

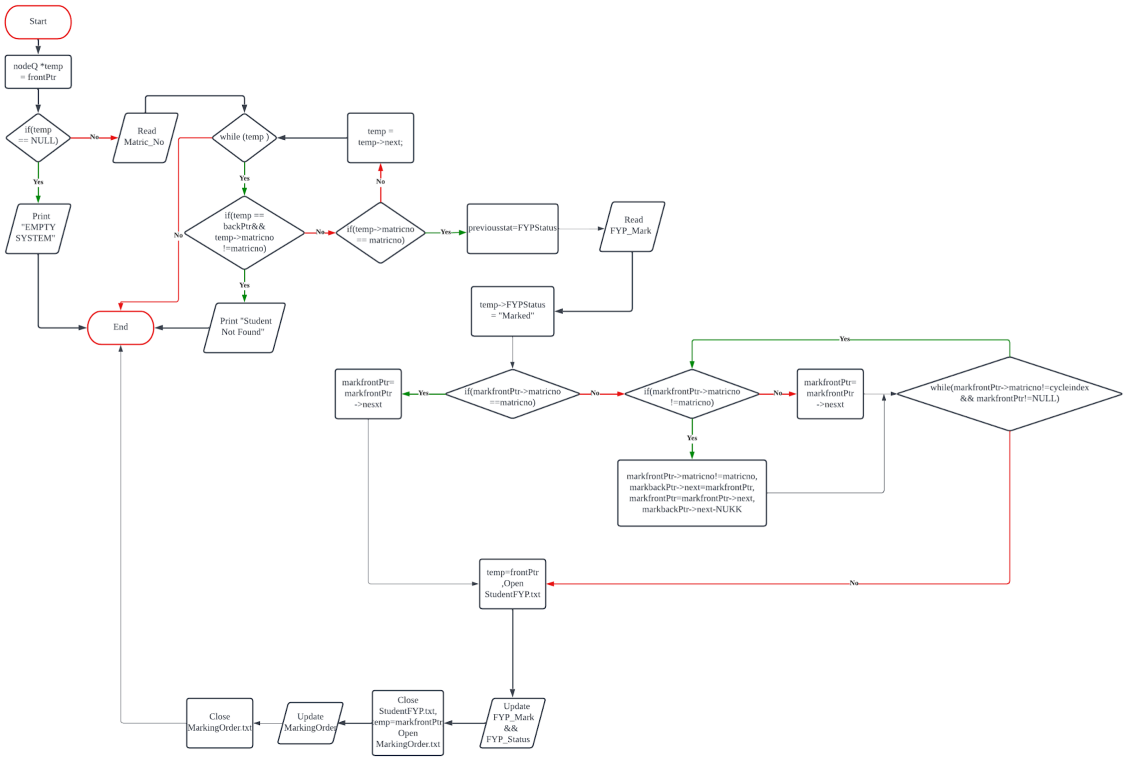


Figure 16 shows a flowchart of UpdateResult()

See More:
https://drive.google.com/file/d/1pBi5X0eLg8BLb5uX9dcTVGofwSbx9R9T/view?usp=sharing

Flowchart 15: Mark Student Menu

Prepared By: Amira

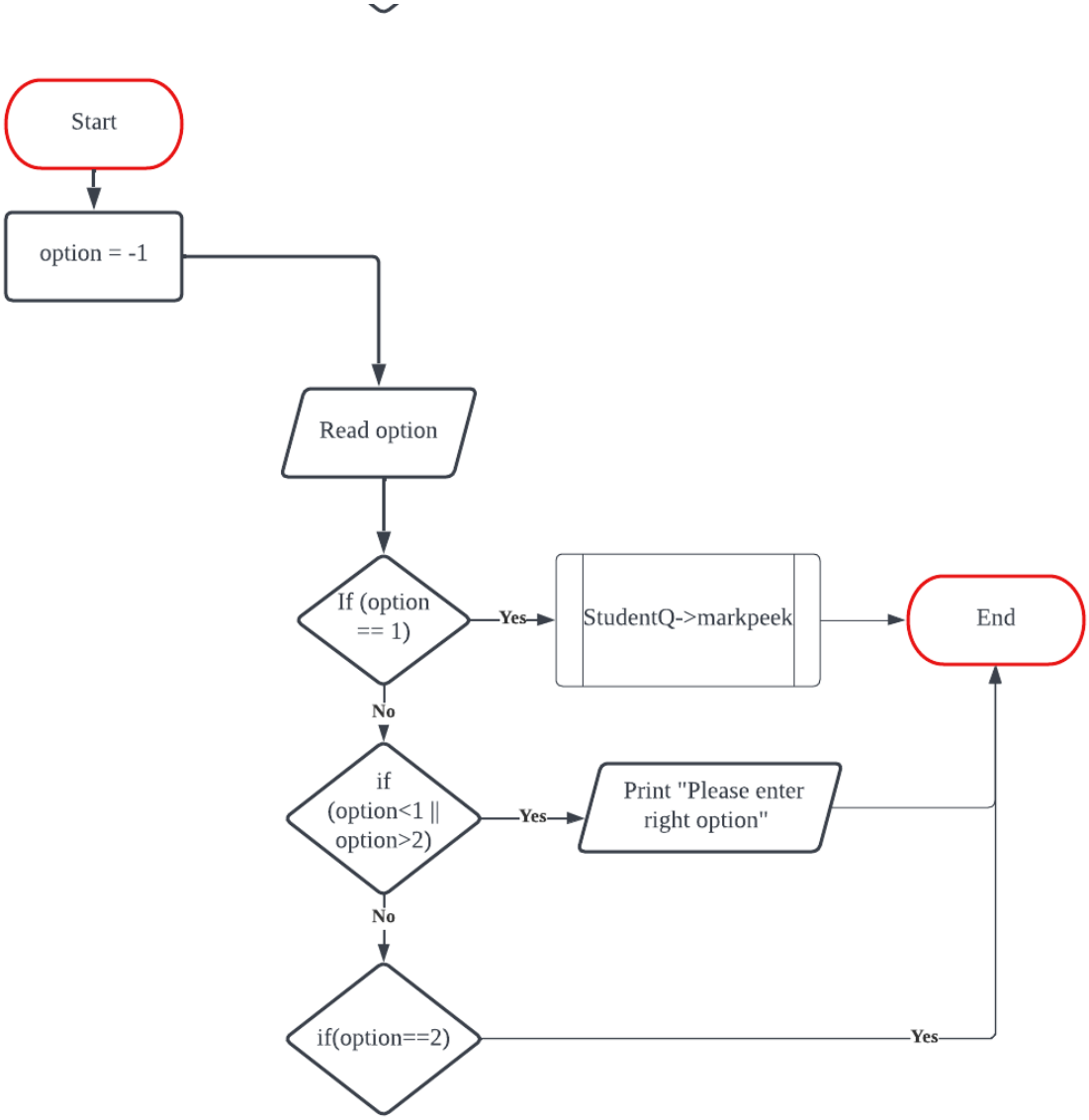


Figure 17 shows a flowchart of Peek()

Flowchart 16: Mark Student FYP

Prepared By: Ng Kai Zheng

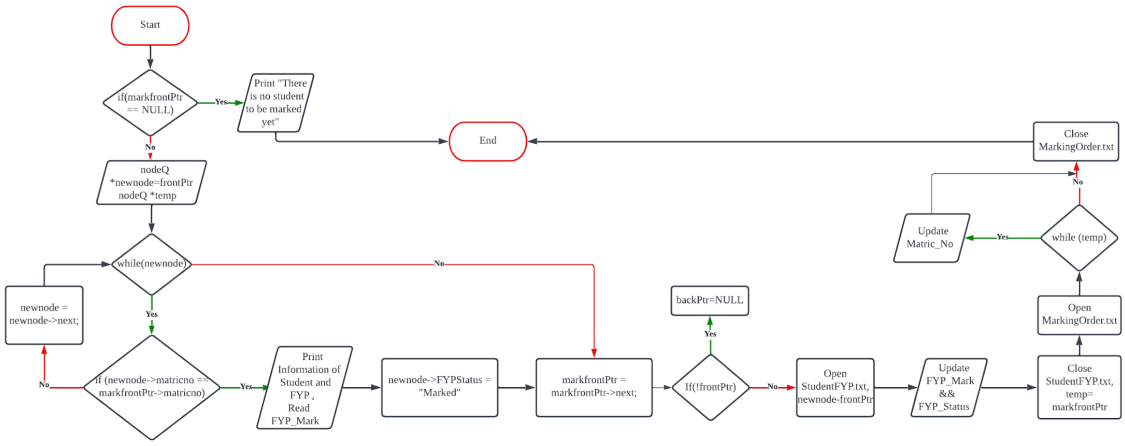


Figure 18 shows a flowchart of markpeek()

PART 3: SYSTEM PROTOTYPE

Prepared By: AMIRA AZKIA RAHMATULLOH

This program manages the final year project for both students and lecturers. When users launch the system, they are greeted by the main menu and asked to select the type of user they are.

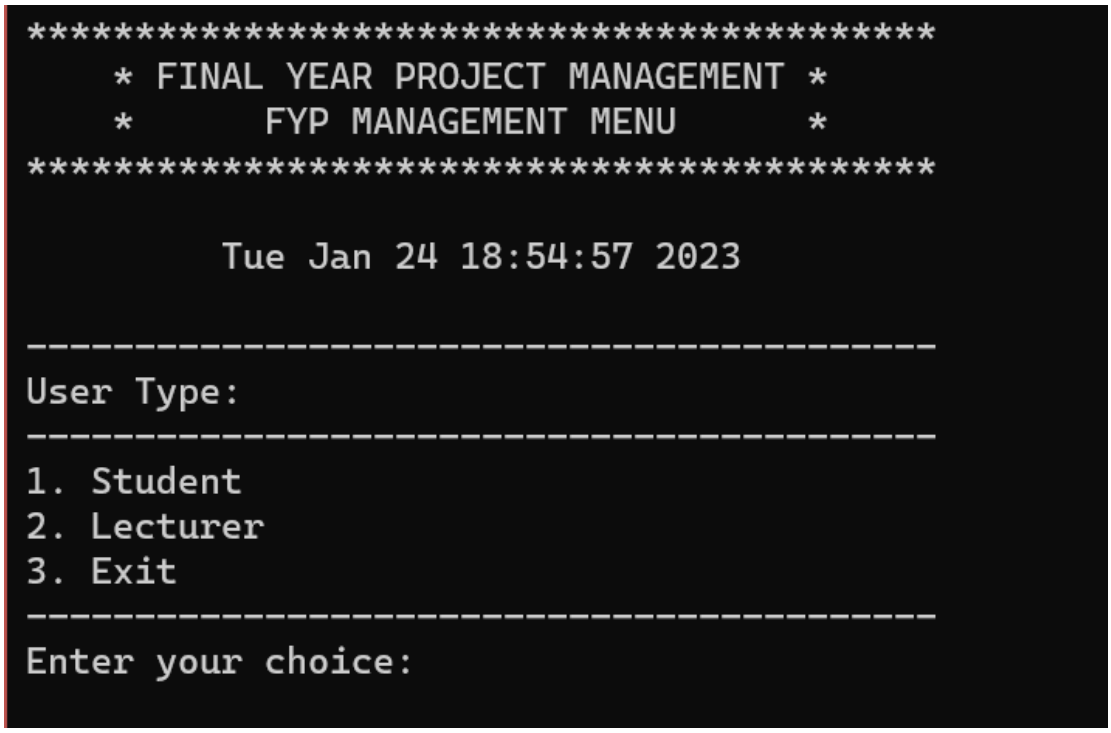


Image 1

According to the preceding illustration, once the user has indicated which type they are, they will be sent to a distinct menu based on that selection.

3.1 Student

If the user selects option 1, the student menu will appear, as shown in **Image 2**, where they can check the proposed FYP field, enter their own FYP information, update both FYP information and status, display their information, and check their results.

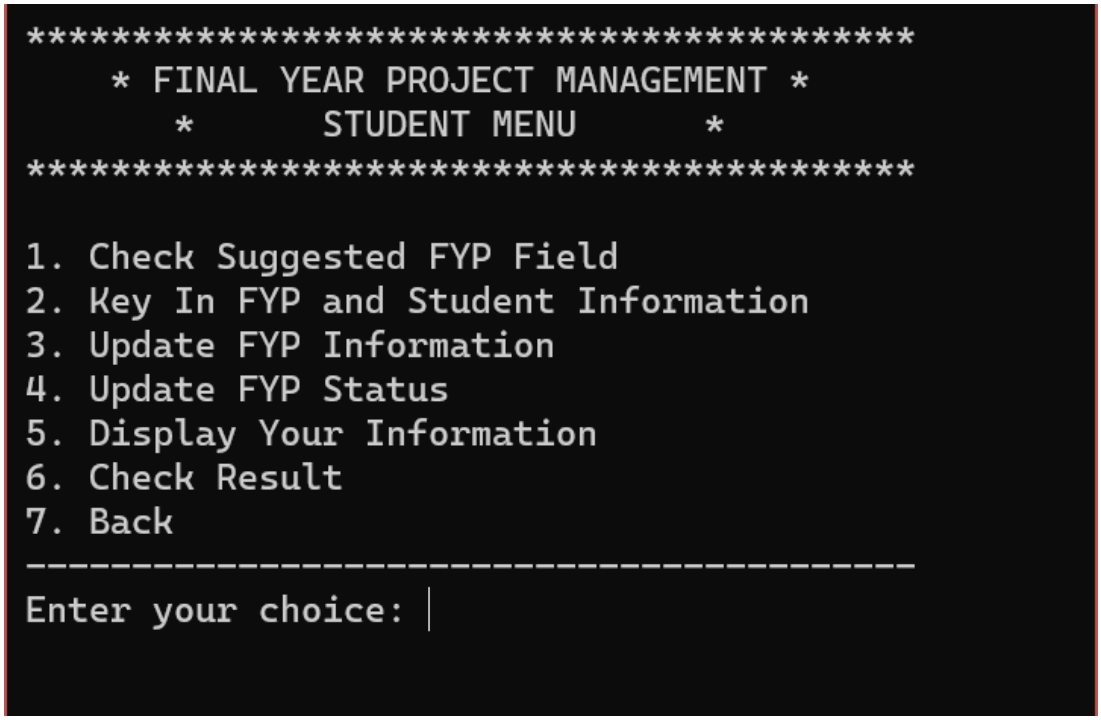


Image 2

By selecting option 1, the student can view all of the suggested fields for their FYP. When the student selects the appropriate number for the field, as illustrated in **Image 3**, they will receive a brief description of it.

```
****************************************
    * FINAL YEAR PROJECT MANAGEMENT *
    *      SUGGESTED FYP FIELD      *
****************************************

1. Artificial Intelligence
2. Web Technology
3. Data Science
4. Machine Learning
5. Internet of Things
6. Blockchain
7. Game Development
8. Back
----------------------------------------
Enter your choice: 1

Artificial Intelligence (AI): AI is the branch
of computer science that deals with creating ma
chines and systems that can perform tasks that
typically require human intelligence, such as u
nderstanding natural language, recognizing spee
ch, making decisions, and so on. There are many
 different areas of AI research, such as natura
l language processing, computer vision, robotic
s, and so on.
----------------------------------------
Enter your choice: 6

Blockchain: Blockchain is a distributed ledger
technology that enables secure and transparent
transactions without the need for intermediarie
s. It uses cryptography to secure transactions
and provide a tamper-proof record of all transa
ctions on the network.
----------------------------------------
Enter your choice:
```

Image 3

Moving on, as shown in **Image 4**, by selecting number 2 in the student main menu, the student can enter their information, such as name, matric number, and course, as well as FYP information, such as field and title.

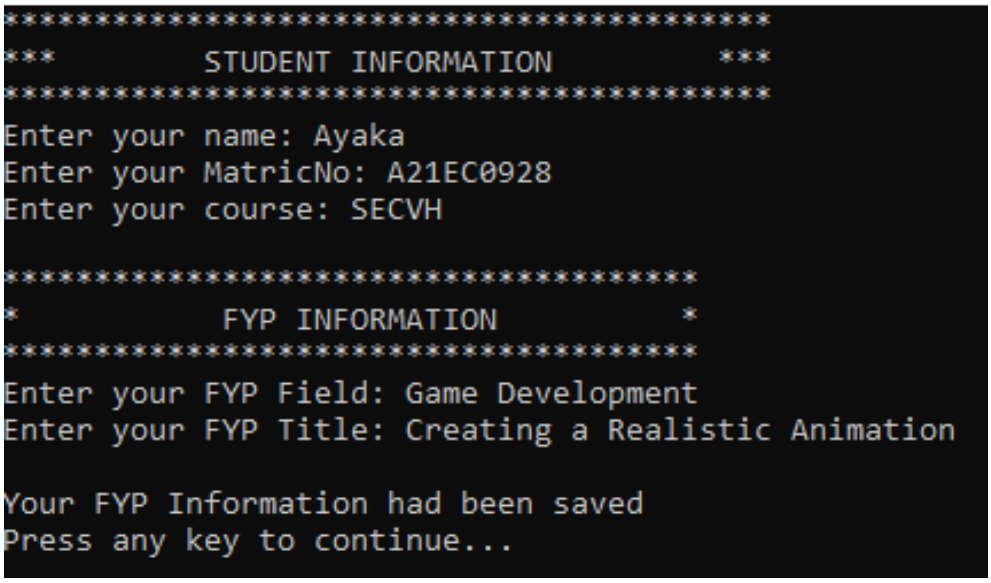


Image 4

When the student information is successfully added, it will be included in the StudentFYP.text as shown in **Image 5.** To identify, the newly added information is highlighted.

StudentFYP.text

```
NG KAI ZHENG
A21EC0101
SECVH
Game Development
ILoveGaming
PresentationDone
Tue Jan 24 14:04:49 2023
-1
Gan Heng Lai
A21EC0148
SECVH
Machine Learning
MachineGoodGood
Title Submitted
Tue Jan 24 14:04:49 2023
-1
Lew Chin Hong
A21EC0188
SECVH
```

```
Web Development
WebWebWeb
PresentationDone
Tue Jan 24 14:04:49 2023
-1
YEO CHUN TECK
A21EC0210
SECJH
Artificial Intelligence
AI Change World
Marked
Tue Jan 24 14:04:49 2023
100
Amirah
A21EC0114
SECTH
Artificial Intelligence
AI Change World
Marked
Tue Jan 24 14:04:49 2023
100
WAI JIA WEN
A21EC0808
SECVH
COMPUTER GRAPHICS
COMPUTER GAMES
TitleSubmitted
Tue Jan 24 14:43:37 2023
-1
Ayaka
A21EC0928
SECVH
Web Technology
Multimedia Web
ProposalSubmitted
Tue Jan 24 20:05:01 2023
-1
```

Image 5

To check their personal information, students can go to the student main menu and select option 5 (display your information). When the user selects it, they will be required to provide their own matric number, as illustrated in **Image 6**.

Image 6

The student can check their own and FYP information after successfully putting the matric number, as illustrated in **Image 7**. The student may now view when they submitted their FYP title, as well as the FYP status and marks, on this page.

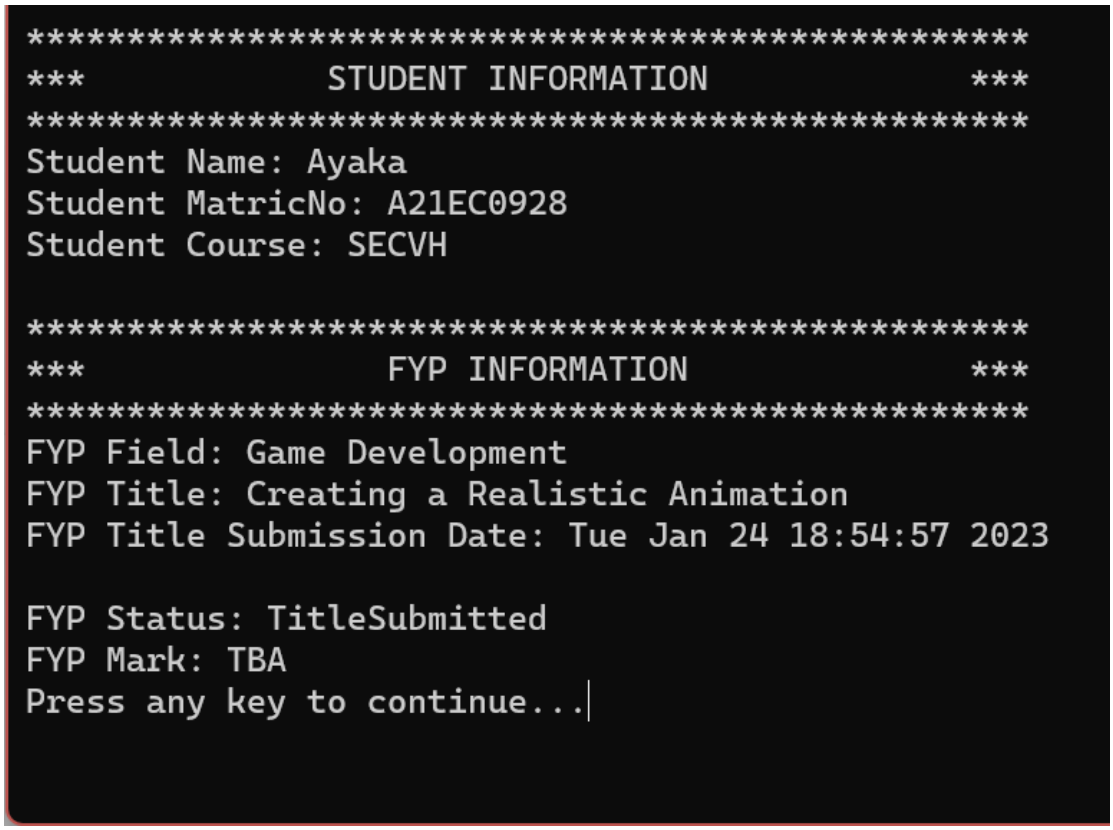


Image 7

If the student wishes to modify their FYP field and title, they can select option 3 (update FYP Information) and enter their new FYP field and title as seen in **Image 8**.

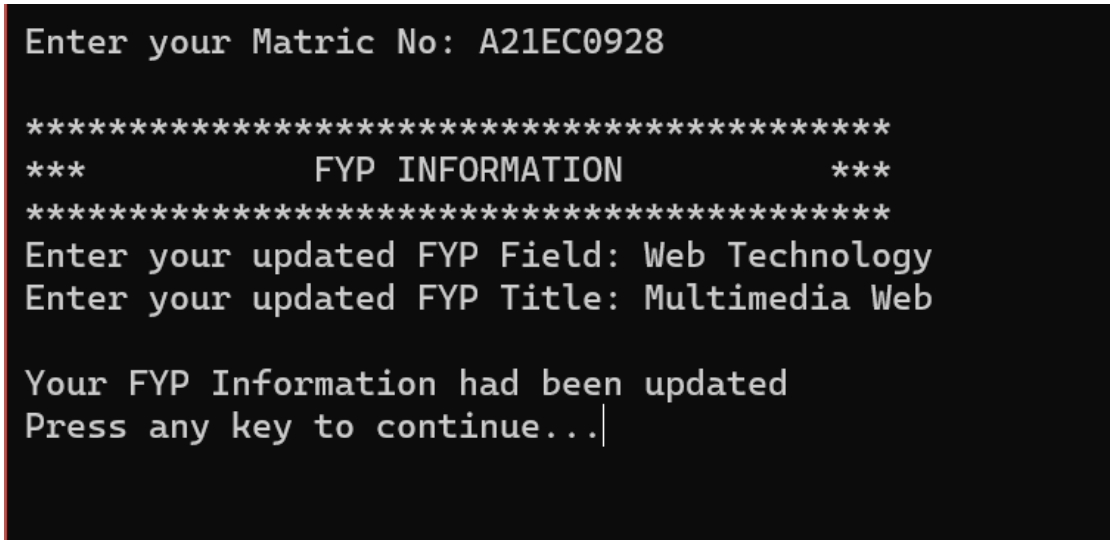


Image 8

Students can change the status of their FYP after selecting the final FYP field and title, as illustrated in **Image 9**.

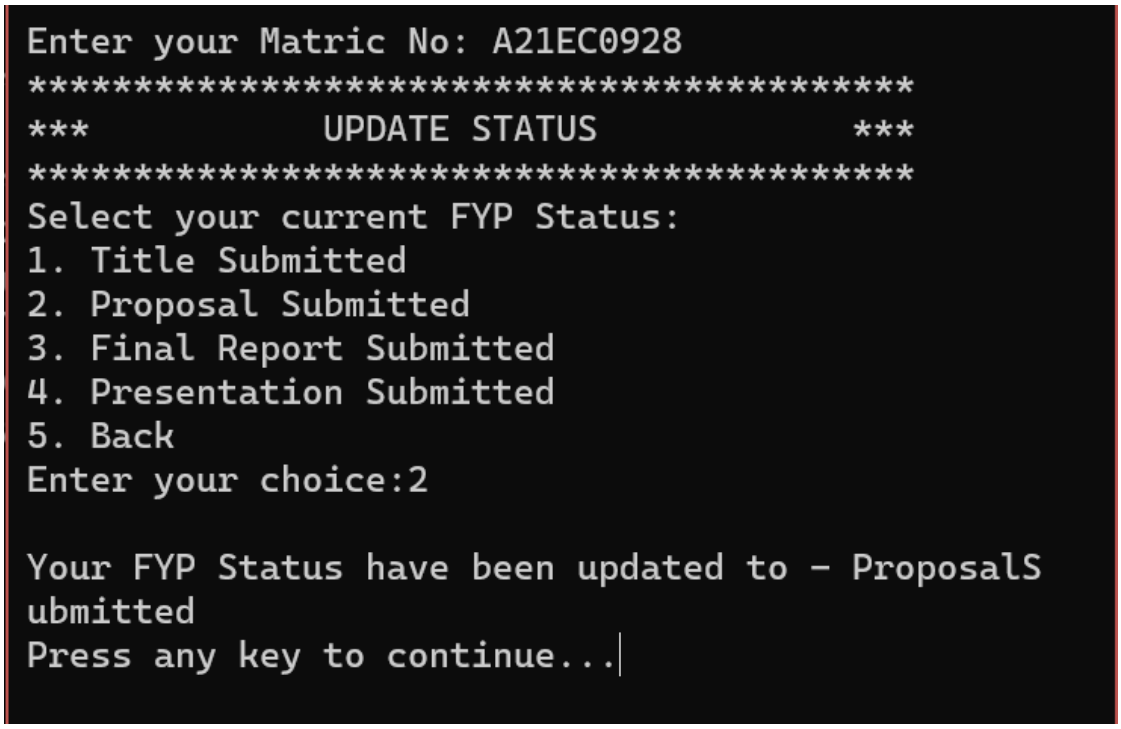


Image 9

When the student selects number 6 (Check Result) from the student menu **Image2**, they can check their FYP status and mark in **Image 10**, based on the option from the previous image.

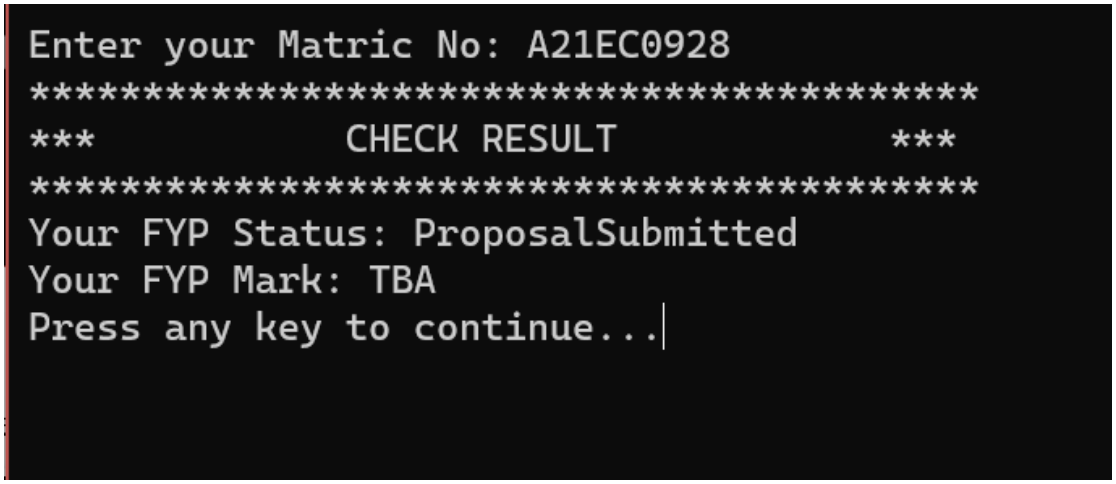


Image 10

Finally, when the student checks their information, they will see a difference as shown in **image 11.**

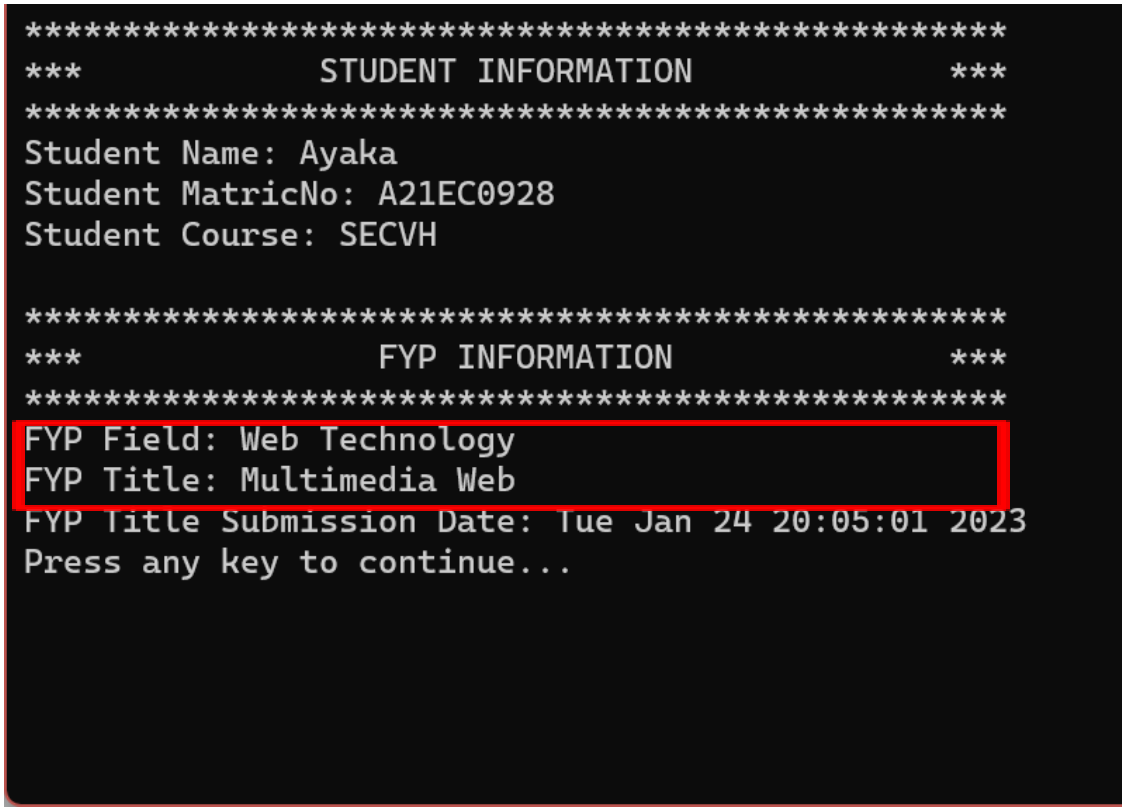


Image 11

If the students update their FYP status to “Presentation Submitted”,as illustrated in **image 12**, their FYP would be in the queue for marking by the lecturer.

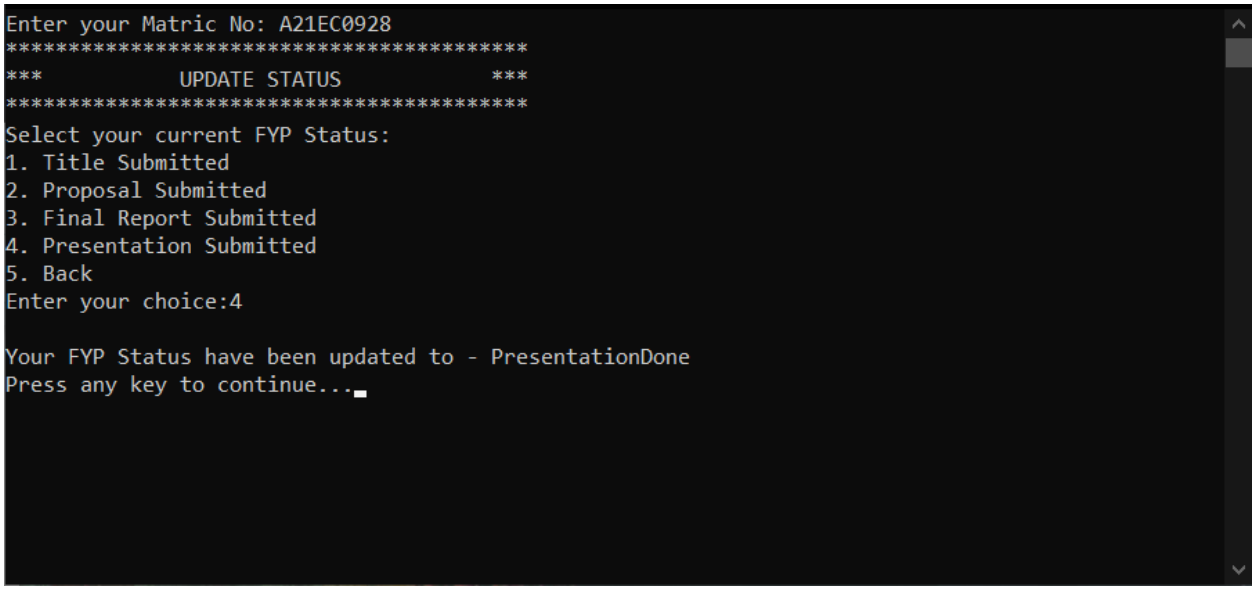
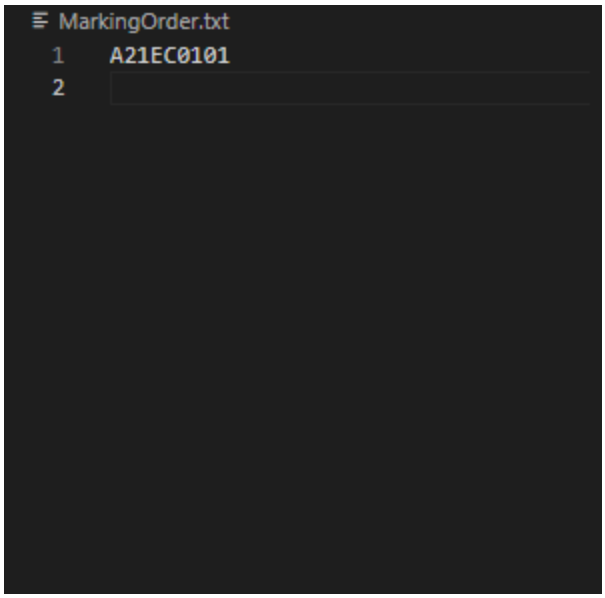


Image 12

Previously, there was one student in the queue for marking as illustrated in **image 13**. After the student changes its state to "Presentation Submitted". There would be a new record which is his Matric Number below the first record in MarkingOrder.txt as illustrated in **image 14**. Now, he is the second person in the queue who had submitted a presentation and is ready to be marked by the lecturer.



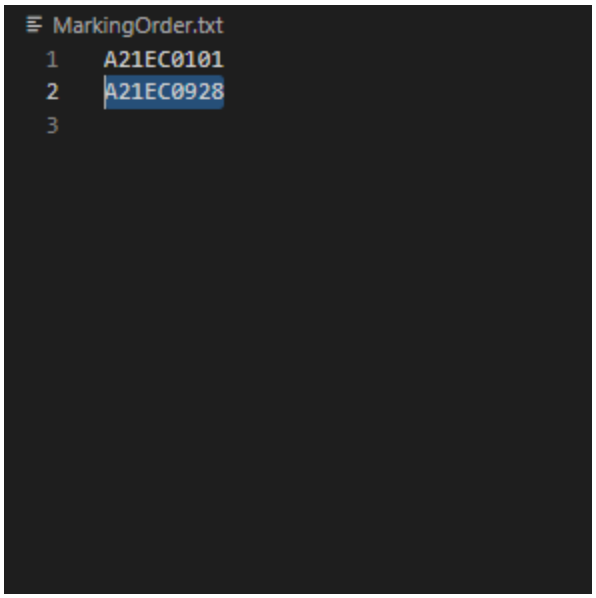


Image 13

Image 14

However, if the student selects the wrong option or just intends to change FYP status from "Presentation Submitted" to others before marking due to some reasons, they are allowed to do it as long as their FYP hasn't been marked. They could change their FYP status as seen in **Image 9** and they would be removed from the queue for marking as seen in **Image 13**

3.2 Lecturer

If the user selects option 2 from **Image 1**, the lecturer menu will appear, where they can select whether they are a first-time or frequent lecturer, as illustrated in **Image 15**.

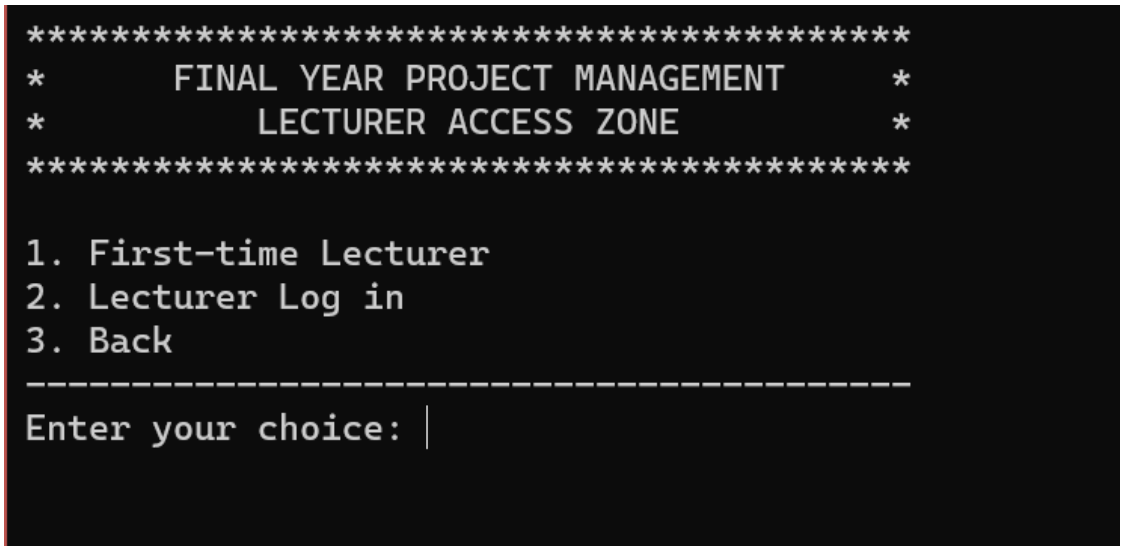


Image 15

To be able to login later, the first-time lecturer will need to enter their name and password as seen in **Image 16**. They can also add another lecturer if they so desire.

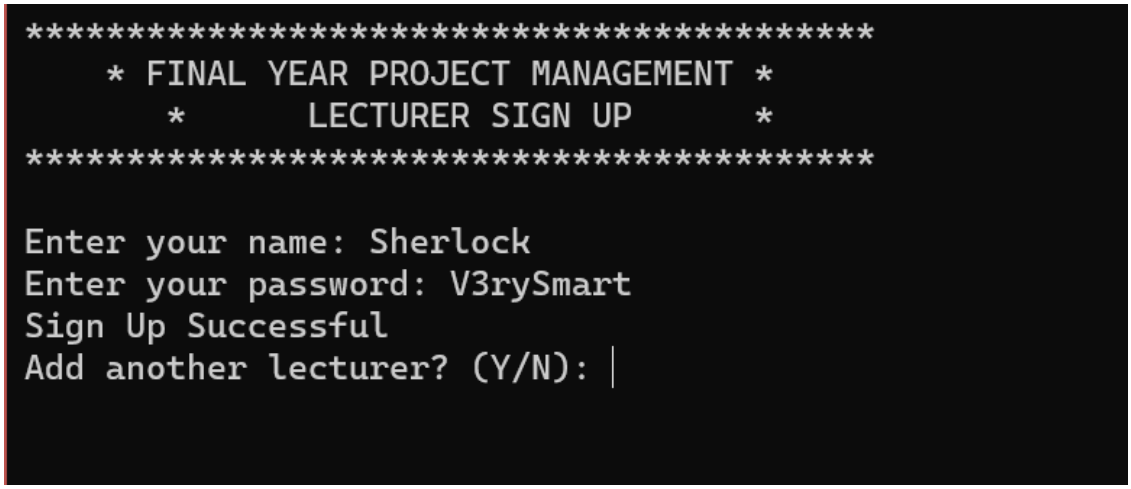


Image 16

As seen below, the prior information will be uploaded to the LecturerAccount.txt file. The highlighted account is the newly added one.

LecturerAccount.txt

```
Lecturer
Lec123
Dr.Ali
Ali123
Sherlock
V3rySmart
```

The methods for logging into the system are similar to the previous image (**Image 16**), with the instructor entering their name and password as seen in **Image 17**.

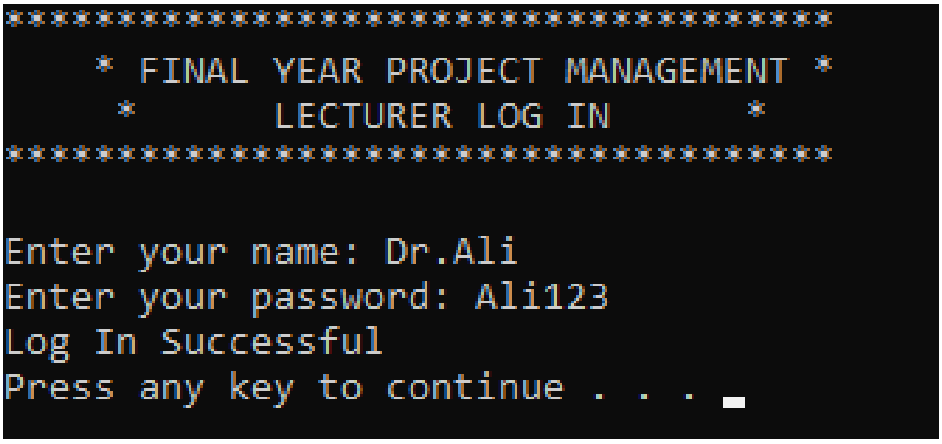


Image 17

After successfully logging in, the lecturer will see the lecturer menu, from which they can choose what to accomplish (see **Image 18**).

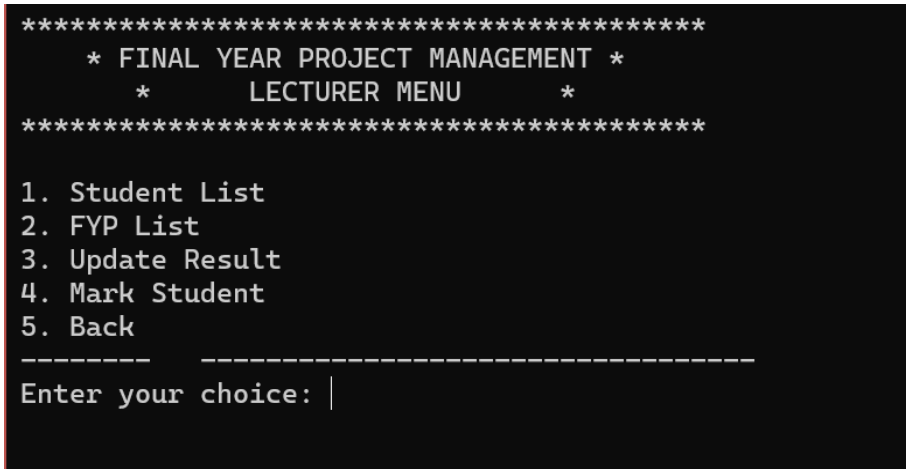


Image 18

When the professor selects number one, the student list, they will view all of the students' entire information as well as their FYP information, as seen in **Image 19**.

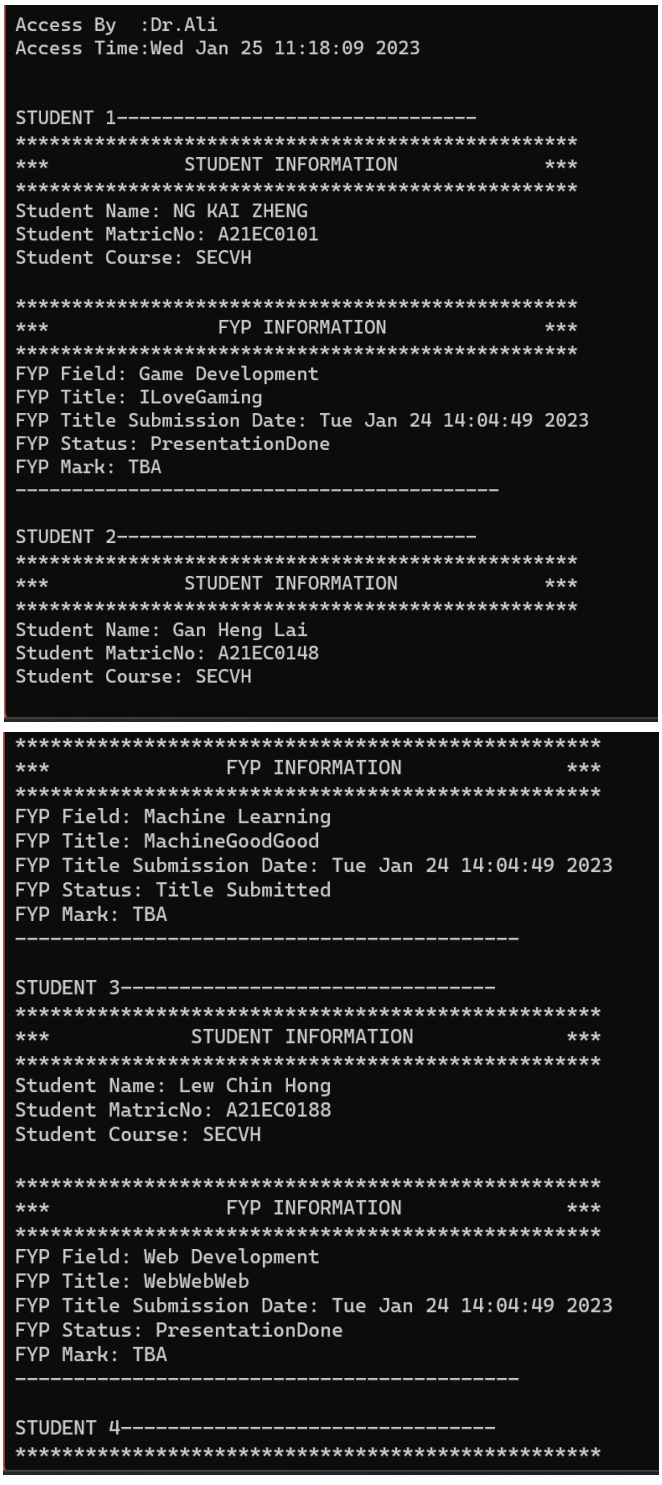


Image 19

When the professor selects number 2, FYP list, they will view the student's information as well as their FYP title, status, and mark, as seen in **Image 20**.

Image 20

Furthermore, the instructor has the authority to mark all unmarked students. The students who are in the queue are ready to be marked. The instructor will first be asked if they want to mark the students as seen in **Image 21**. When they select yes, they will be led to each student's information from the queue shown in **Image 13**.

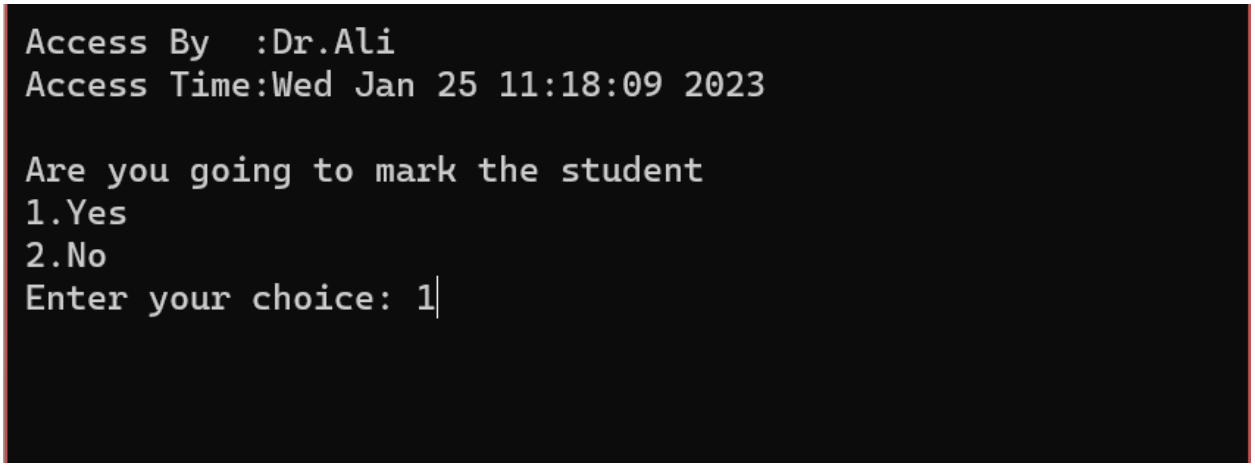


Image 21

Image 22 shows the student's details and FYP, as well as an area for the professor to mark their grades. When the students check their results, they will see the revised mark from the instructor.

```
************************************************
***            STUDENT INFORMATION            ***
************************************************
Student Name: NG KAI ZHENG
Student MatricNo: A21EC0101
Student Course: SECVH

************************************************
***               FYP INFORMATION             ***
************************************************
FYP Field: COMPUTER GRAPHICS AND MULTIMEDIA
FYP Title: ILOVEGAMING
FYP Title Submission Date: Tue Jan 24 14:04:49 2023
FYP Status: PresentationDone
FYP Mark: TBA
****************************************
***           UPDATE RESULT            ***
****************************************
Enter Student FYP Mark: 100
```

Image 22

The record in the queue for marking as illustrated in **Image 13** would be removed from the queue after marking and MarkingOrder.txt would be empty as illustrated in **Image 23**

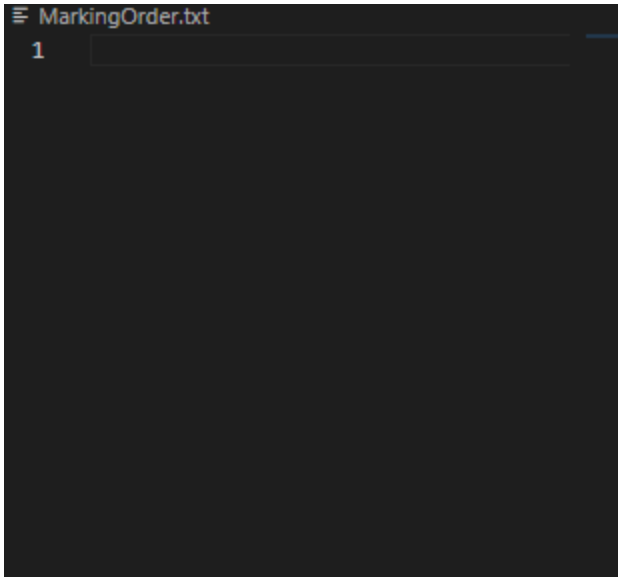


Image 23

Lastly, the lecturer can update the student's mark by selecting the update result from the main menu (**Image 18**). They must first enter the matric number of the student whose results they desire to change, followed by the anticipated mark, as illustrated in **Image 24**.

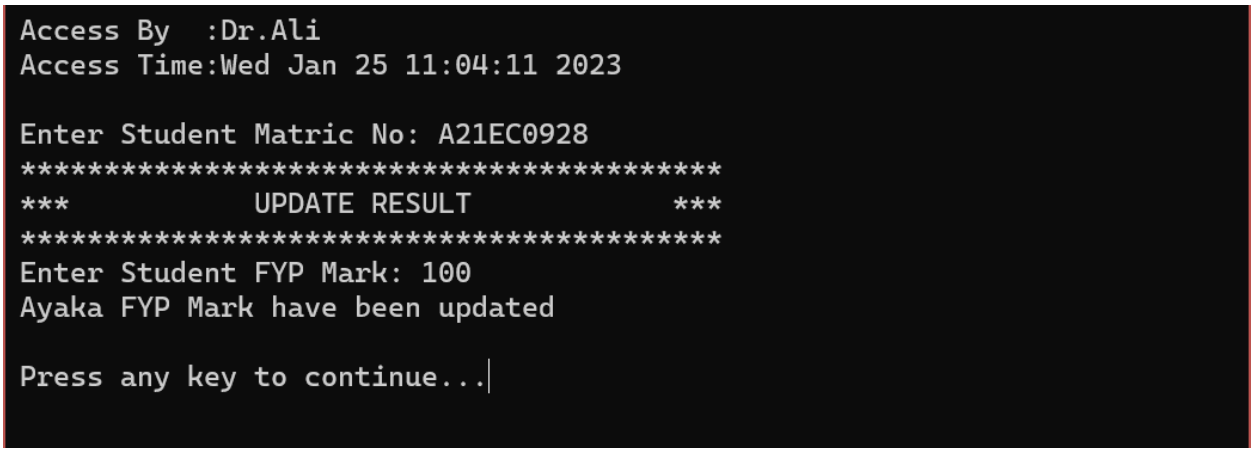


Image 24

PART 4: DEVELOPMENT ACTIVITIES

Meeting Date	Members Participate in the meeting	Activity	Task for each member	Task Achieved (Y/N)
17/1/2023	1. Ng Kai Zheng 2. Yeo Chun Teck 3. Amirah	To allocate member tasks for the Project and to discuss the flow of the program	**Ng Kai Zheng** -Write the main program and implement queue function **Yeo Chun Teck** -help to write the coding and draw the class diagram and write the report **Amira** -write the menu for the user, test program and write the report	Y

18/1/2023	1. Ng Kai Zheng 2. Yeo Chun Teck 3. Amira	Discuss the progress about the program development and continue to write the program and debug the code.	**Ng Kai Zheng** -show members how each function in the program and continue writing the program **Yeo Chun Teck** -explain the flow of the program through the class diagram and help Kai Zheng to debug **Amira** -combine the menu with the program, test with validate input and help in debug	Y
23/1/2023	1. Ng Kai Zheng 2. Yeo Chun Teck 3. Amira	Test the final program checks error, and write the report	**Ng Kai Zheng** - tests the program and write the report **Yeo Chun Teck** - guide all members through the program and explain the functionality of each line of coding **Amira** - write the report, testing the program with validate input	Y

24/1/2023	1. Ng Kai Zheng 2. Yeo Chun Teck 3. Amira	Discuss the final program, complete the report and check everything	**<u>Ng Kai Zheng</u>** - testing program with all possible input (including invalid input), write the report **<u>Yeo Chun Teck</u>** - continue to proceed with the report, check again the code **<u>Amira</u>** - continue to write the report, check the report	Y
25/1/2023	1. Ng Kai Zheng 2. Yeo Chun Teck 3. Amira	Final Checking and submit	**<u>Ng Kai Zheng</u>** - final checking the report **<u>Yeo Chun Teck</u>** - final checking the report, and submit **<u>Amira</u>** - final checking the report	Y

PART 5: APPENDIX

Source Code of Final Year Project Management System

Prepared By: Ng Kai Zheng, Yeo Chun Teck, Amira

Figure 19 below shows the declaration of variables and member functions of Linked list queue implementation in class studentQ.

```
1. //GROUP MEMBER:
2.
3. #include <iostream>
4. #include <fstream>
5. #include <string>
6. #include <iomanip>
7. #include <cstdio>
8. #include <cstdlib>
9. #include <ctime>
10. using namespace std;
11.
12. time_t now = time(NULL);
13. char* date = ctime(&now);
14.
15. class nodeQ
16. {
17.     public:
18.         string name;
19.         string matricno;
20.         string course;
21.         string FYPField;
22.         string FYPTitle;
23.         string FYPStatus;
24.         string FYPDate;
25.         double FYPMark;
26.         nodeQ *next;
27. };
28.
29. class StudentQ
30. {
31.     public:
32.         nodeQ *backPtr, *frontPtr, *markbackPtr, *markfrontPtr;
33.         nodeQ *create_node(string name, string matricno, string course, string
    FYPField, string FYPTitle, string FYPStatus, string FYPDate, double FYPMark)
34.         {
35.             nodeQ *newNode = new nodeQ;
36.             if(!newNode)
37.             {
38.                 cout << "Memory Error" << endl;
39.                 return NULL;
40.             }
41.             else
42.             {
43.             newNode->name = name;
44.             newNode->matricno = matricno;
45.             newNode->course = course;
```

```
46.             newNode->FYPField = FYPField;
47.             newNode->FYPTitle = FYPTitle;
48.             newNode->FYPStatus = FYPStatus;
49.             newNode->FYPDate = FYPDate;
50.             newNode->FYPMark = FYPMark;
51.             newNode->next = NULL;
52.             return newNode;
53.             }
54.         }
55.         StudentQ()
56.         {
57.             backPtr = NULL;
58.             frontPtr = NULL;
59.             markbackPtr=NULL;
60.             markfrontPtr=NULL;
61.         }
62.         bool isEmpty()
63.         {
64.             return (frontPtr == NULL && backPtr == NULL);
65.         }
66.         ~StudentQ()
67.         {
68.             nodeQ *temp = frontPtr;
69.             while (temp)
70.             {
71.                 frontPtr = frontPtr->next;
72.                 delete temp;
73.                 temp = frontPtr;
74.
75.             }
76.         }
77.         void ReadStudent()
78.         {
79.             fstream fin;
80.             fin.open("StudentFYP.txt", ios::in);
81.             string name, matricno, course, FYPField, FYPTitle, FYPStatus,
    FYPDate;
82.             double FYPMark;
83.             while (!fin.eof())
84.             {if(!fin){
85.                     break;}
86.                 getline(fin, name);
87.                 if(name == "") break;
88.                 getline(fin, matricno);
89.                 getline(fin, course);
90.                 getline(fin, FYPField);
91.                 getline(fin, FYPTitle);
92.                 getline(fin, FYPStatus);
93.                 getline(fin, FYPDate);
94.                 fin >> FYPMark;
95.                 fin.ignore();
96.                 nodeQ *newNode = create_node(name, matricno, course, FYPField,
    FYPTitle, FYPStatus, FYPDate, FYPMark);
97.                 if (frontPtr == NULL)
```

```
98.                 {
99.                     frontPtr = newNode;
100.                    backPtr = newNode;
101.                    frontPtr->next = NULL;
102.                }
103.                else
104.                {
105.                    backPtr->next = newNode;
106.                    backPtr = newNode;
107.                    backPtr->next = NULL;
108.                }
109.            }
110.            fin.close();
111.        }
112.
113.
114.        void ReadMarkingOrder(){
115.            fstream fin;
116.            fin.open("MarkingOrder.txt",ios::in);
117.            string  matricno;
118.            while(!fin.eof()){
119.                getline(fin,matricno);
120.                if(matricno=="") break;
121.                nodeQ* temp= new nodeQ;
122.                temp->matricno=matricno;
123.                //temp= frontPtr;
124.                /*while(temp->matricno!=matricno){
125.                    temp=temp->next;
126.                }*/
127.            //nodeQ *newNode= new nodeQ;
128.            /*newNode->name = temp->name;
129.            newNode->matricno = temp->matricno;
130.            newNode->course = temp->course;
131.            newNode->FYPField = temp->FYPField;
132.            newNode->FYPTitle = temp->FYPTitle;
133.            newNode->FYPStatus = temp->FYPStatus;
134.            newNode->FYPDate = temp->FYPDate;
135.            newNode->FYPMark = temp->FYPMark;
136.            newNode->next = NULL;*/
137.
138.            if (markfrontPtr == NULL){
139.                markfrontPtr = temp;
140.                markbackPtr = temp;
141.                markfrontPtr->next = NULL;
142.            }
143.            else{
144.                markbackPtr->next=temp;
145.                markbackPtr=temp;
146.                markbackPtr->next=NULL;
147.            }
148.
149.        }
150.            fin.close();
151.        }
```

```
152.          string isEmpty(string &input, string type)
153.          {
154.              getline(cin, input);
155.              while (input == "")
156.              {
157.                  cout << "Invalid input. Please try again." << endl;
158.                  cout << "Enter your " << type << ": ";
159.                  getline(cin, input);
160.              }
161.              return input;
162.          }
163.          void enqueue()
164.          {
165.              string name, matricno, course, FYPField, FYPTitle, FYPStatus,
    FYPDate;
166.              double FYPMark;
167.
168.              cout << "****************************************" << endl;
169.              cout << "***      STUDENT INFORMATION        ***" << endl;
170.              cout << "****************************************" << endl;
171.              cin.ignore();
172.              cout << "Enter your name: ";
173.              isEmpty(name, "name");
174.              cout << "Enter your MatricNo: ";
175.              isEmpty(matricno, "MatricNo");
176.              nodeQ *checknode= frontPtr;
177.              while(checknode !=NULL){
178.                  if(checknode->matricno==matricno){
179.                      cout<<"You have record in the system already. Don't need
    to register a new FYP"<<endl;
180.                      cout<<"Press any key to continue";
181.                      cin.get();
182.
183.                      return;
184.                  }
185.                      checknode=checknode->next;
186.
187.              }
188.              cout << "Enter your course: ";
189.              isEmpty(course, "course");
190.
191.              cout << "\n*************************************" << endl;
192.              cout << "*        FYP INFORMATION            *" << endl;
193.              cout << "*************************************" << endl;
194.
195.              cout << "Enter your FYP Field: ";
196.              isEmpty(FYPField, "FYP Field");
197.              cout << "Enter your FYP Title: ";
198.              isEmpty(FYPTitle, "FYP Title");
199.              FYPStatus = "TitleSubmitted";
200.              FYPDate = date;
201.              FYPMark = -1.0;
202.              nodeQ *newNode = create_node(name, matricno, course, FYPField,
    FYPTitle, FYPStatus, FYPDate, FYPMark);
```

```
203.
204.                 if (frontPtr == NULL)
205.                 {
206.                     frontPtr = newNode;
207.                     backPtr = newNode;
208.                     frontPtr->next = NULL;
209.                 }
210.                 else
211.                 {
212.                     backPtr->next = newNode;
213.                     backPtr = newNode;
214.                     backPtr->next = NULL;
215.                 }
216.                 fstream fout;
217.                 fout.open("StudentFYP.txt", ios::app);
218.                 nodeQ *temp = backPtr;
219.                 while (temp)
220.                 {
221.                     fout << temp->name << endl
222.                     << temp->matricno << endl
223.                     << temp->course << endl
224.                     << temp->FYPField << endl
225.                     << temp->FYPTitle << endl
226.                     << temp->FYPStatus << endl
227.                     << temp->FYPDate
228.                     << temp->FYPMark << endl;
229.                     temp = temp->next;
230.                 }
231.                 fout.close();
232.                 cout << "\nYour FYP Information had been saved" << endl;
233.                 cout << "Press any key to continue...";
234.                 cin.get();
235.             }
236.
237.             void alterqueue()
238.             {
239.                 nodeQ *temp = frontPtr;
240.                 if (frontPtr == NULL)
241.                 {
242.                     cout << "SYSTEM EMPTY" << endl;
243.                     cout << "Press any key to continue...";
244.                     cin.get();
245.                     return;
246.                 }
247.                 string matricno, FYPField, FYPTitle, FYPStatus, FYPDate;
248.                 cin.ignore();
249.                 cout << "Enter your Matric No: ";
250.                 getline(cin, matricno);
251.                 if(temp->FYPStatus=="Marked")
252.                 {
253.                     cout<<"Your FYP had been marked. You can't change the FYP
    Information."<<endl;
254.                     cin.get();
255.                     return;
```

```
256.                }
257.                while(temp)
258.                {
259.
260.                    if(temp->matricno == matricno)
261.                    {
262.                        cout << "\n****************************************"
    << endl;
263.                        cout << "***           FYP INFORMATION            ***"
    << endl;
264.                        cout << "****************************************" <<
    endl;
265.
266.                        cout << "Enter your updated FYP Field: ";
267.                        isEmpty(FYPField, "updated FYP Field");
268.
269.                        cout << "Enter your updated FYP Title: ";
270.                        isEmpty(FYPTitle, "updated FYP Title");
271.
272.                        temp->FYPField = FYPField;
273.                        temp->FYPTitle = FYPTitle;
274.                        temp = frontPtr;
275.                        fstream fout;
276.                        fout.open("StudentFYP.txt", ios::out);
277.                        while (temp)
278.                        {
279.                            fout << temp->name << endl
280.                            << temp->matricno << endl
281.                            << temp->course << endl
282.                            << temp->FYPField << endl
283.                            << temp->FYPTitle << endl
284.                            << temp->FYPStatus << endl
285.                            << temp->FYPDate << endl
286.                            << temp->FYPMark << endl;
287.                            temp = temp->next;
288.                        }
289.                        fout.close();
290.                        cout<<"\nYour FYP Information had been updated" << endl;
291.                        cout << "Press any key to continue...";
292.                        cin.get();
293.                        break;
294.                    }
295.                    temp = temp->next;
296.                    if(temp == NULL)
297.                    {
298.                        cout << "Not Found" << endl;
299.                        cout << "Press any key to continue...";
300.                        cin.get();
301.                        break;
302.                    }
303.
304.                }
305.
306.            }
```

```
307.          void display()
308.          {
309.              nodeQ *temp = frontPtr;
310.              int counter = 1;
311.              if(temp == NULL)
312.              {
313.                  cout << "EMPTY SYSTEM";
314.                  cout << "Press any key to continue...";
315.                  cin.get();
316.                  return;
317.              }
318.              while (temp != NULL)
319.              {
320.                  cout << "STUDENT " << counter++ <<
    "---------------------------------" << endl;
321.                  cout << "************************************************"
    << endl;
322.                  cout << "***         STUDENT INFORMATION              ***"
    << endl;
323.                  cout << "************************************************"
    << endl;
324.                  cout << "Student Name: " << temp->name << endl
325.                  << "Student MatricNo: " << temp->matricno << endl
326.                  << "Student Course: " << temp->course << endl;
327.                  cout <<
    "\n************************************************" << endl;
328.                  cout << "***               FYP INFORMATION            ***"
    << endl;
329.                  cout << "************************************************"
    << endl;
330.                  cout << "FYP Field: " << temp->FYPField << endl
331.                  << "FYP Title: " << temp->FYPTitle << endl
332.                  << "FYP Title Submission Date: " << temp->FYPDate << endl
333.                  << "FYP Status: " << temp->FYPStatus << endl
334.                  << "FYP Mark: ";
335.                  if(temp->FYPMark == -1) {cout << "TBA" << endl;}
336.                  else {cout << temp->FYPMark << endl;}
337.                  cout << "----------------------------------------" <<
    endl << endl;
338.                  temp = temp->next;
339.              }
340.
341.              cout<<"Press any key to continue...";
342.              cin.ignore();
343.              cin.get();
344.
345.          }
346.      void updatestatus()
347.          {
348.
349.              int statuschoice;
350.              nodeQ *temp = frontPtr;
351.              string matricno;
352.              cin.ignore();
```

```
353.                 if(temp == NULL)
354.                 {
355.                     cout << "EMPTY SYSTEM";
356.                     cout << "Press any key to continue...";
357.                     cin.get();
358.                     return;
359.                 }
360.                 cout << "Enter your Matric No: ";
361.                 getline(cin, matricno);
362.                 while(temp)
363.                 {
364.
365.                     if(temp->matricno == matricno)
366.                     {if(temp->FYPStatus=="Marked")
367.                     {
368.                         cout<<"Your FYP had been marked. You can't change the
    status."<<endl;
369.                         cin.get();
370.                         return;
371.                     }
372.                         cout << "****************************************" <<
    endl;
373.                         cout << "***           UPDATE STATUS             ***"
    << endl;
374.                         cout << "****************************************" <<
    endl;
375.                         cout << "Select your current FYP Status: " << endl;
376.                         cout << "1. Title Submitted" << endl;
377.                         cout << "2. Proposal Submitted" << endl;
378.                         cout << "3. Final Report Submitted" << endl;
379.                         cout << "4. Presentation Done" << endl;
380.                         cout << "5. Back" << endl;
381.                         string previousstat=temp->FYPStatus;
382.                         do
383.                         {
384.                             cout << "Enter your choice:";
385.                             int choice;
386.                             cin >> statuschoice;
387.
388.                             switch(statuschoice)
389.                             {
390.                                 case 1:
391.                                     temp->FYPStatus = "TitleSubmitted";
392.                                     break;
393.                                 case 2:
394.                                     temp->FYPStatus = "ProposalSubmitted";
395.                                     break;
396.                                 case 3:
397.                                     temp->FYPStatus = "FinalReportSubmitted";
398.                                     break;
399.                                 case 4:
400.                                     temp->FYPStatus = "PresentationDone";
401.                                     break;
402.                                 case 5:
```

```
403.                                    return;
404.                                default:
405.                                    cout << "Invalid Choice." << endl;
406.                                    cout << "Please enter a valid choice" <<
    endl;
407.                            }
408.                        }
409.                        while(statuschoice < 1 || statuschoice > 5);
410.                        cout << "\nYour FYP Status have been updated to - " <<
    temp->FYPStatus << endl;
411.                        cout << "Press any key to continue...";
412.                        cin.ignore();
413.                        cin.get();
414.                        nodeQ *newnode=new nodeQ;
415.                        newnode->matricno=temp->matricno;
416.                        if(previousstat!=temp->FYPStatus)
417.                        {
418.
    if(previousstat!="PresentationDone"&&temp->FYPStatus=="PresentationDone")
419.                            {
420.                                if (markfrontPtr == NULL)
421.                                {
422.                                    markfrontPtr = newnode;
423.                                    markbackPtr = newnode;
424.                                    markfrontPtr->next = NULL;
425.                                }
426.                                else
427.                                {
428.                                    markbackPtr->next=newnode;
429.                                    markbackPtr=newnode;
430.                                    markbackPtr->next=NULL;
431.                                }
432.                            }
433.
    if(previousstat=="PresentationDone"&&temp->FYPStatus!="PresentationDone")
434.                            {
435.
436.                                string cycleendindex=markfrontPtr->matricno;
437.                                if(markfrontPtr->matricno==matricno)
438.                                {
439.                                    markfrontPtr=markfrontPtr->next;
440.                                }
441.                                else
442.                                {
443.                                    do
444.                                    {
445.                                       if(markfrontPtr->matricno!=matricno)
446.                                       {
447.                                            markbackPtr->next=markfrontPtr;
448.                                            markbackPtr=markfrontPtr;
449.                                            markfrontPtr=markfrontPtr->next;
450.                                            markbackPtr->next=NULL;
451.                                       }
452.                                       else
```

```
                                        {
                                             markfrontPtr=markfrontPtr->next;
                                        }

   }while(markfrontPtr->matricno!=cycleendindex&&markfrontPtr!=NULL);
                                    }
                                }
                        }
                        temp = frontPtr;
                        fstream fout;
                        fout.open("StudentFYP.txt", ios::out);
                        while (temp!=NULL)
                        {   cout<<temp->name<<endl;
                            fout << temp->name << endl
                            << temp->matricno << endl
                            << temp->course << endl
                            << temp->FYPField << endl
                            << temp->FYPTitle << endl
                            << temp->FYPStatus << endl
                            << temp->FYPDate << endl
                            << temp->FYPMark << endl;
                            temp = temp->next;
                        }
                        fout.close();
                        temp=markfrontPtr;
                        fout.open("MarkingOrder.txt", ios::out);
                        while (temp!=NULL)
                        {
                            fout << temp->matricno << endl;
                            temp=temp->next;
                        }
                        fout.close();
                        return;

                    }
                    temp = temp->next;
                }
                if(temp == NULL)
                {
                    cout << "Not Found" << endl;
                    cout << "Press any key to continue...";
                    cin.get();
                }
            }

        void CheckResult()
            {
                nodeQ *temp = frontPtr;
                string matricno;
                cin.ignore();
                if(frontPtr == NULL){
                    cout << "EMPTY SYSTEM";
                    cout << "Press any key to continue...";
                    cin.get();
```

```
506.                 return;
507.                 }
508.             cout << "Enter your Matric No: ";
509.             getline(cin, matricno);
510.             while(temp)
511.             {
512.                 if(temp->matricno == matricno)
513.                 {
514.                     cout << "************************************" << endl;
515.                     cout << "*              CHECK RESULT              *" <<
    endl;
516.                     cout << "************************************" << endl;
517.                     cout << "Your FYP Status: " << temp->FYPStatus << endl;
518.                     cout << "Your FYP Mark: ";
519.                     if(temp->FYPMark == -1)
520.                     {
521.                         cout << "TBA" << endl;
522.                     }
523.                     else
524.                     {
525.                         cout << temp->FYPMark << endl;
526.                     }
527.                     cout << "Press any key to continue...";
528.                     cin.get();
529.                     break;
530.
531.                 }
532.                 temp = temp->next;
533.             }
534.             if(temp == NULL)
535.             {
536.                 cout << "Not Found" << endl;
537.                 cout << "Press any key to continue...";
538.                 cin.get();
539.             }
540.         }
541.
542.         void displaycertain()
543.         {
544.             nodeQ *temp = frontPtr;
545.             if(temp == NULL)
546.             {
547.                 cout << "EMPTY SYSTEM";
548.                 cout << "Press any key to continue...";
549.                 cin.ignore();
550.                 cin.get();
551.                 return;
552.             }
553.             string matricno;
554.             cin.ignore();
555.             cout << "Enter your Matric No: ";
556.             getline(cin, matricno);
557.
558.             while(temp)
```

```
559.                {
560.                    if(temp->matricno == matricno)
561.                    {
562.                        system("CLS");
563.                        cout <<
    "************************************************" << endl;
564.                        cout << "***           STUDENT INFORMATION
    ***" << endl;
565.                        cout <<
    "************************************************" << endl;
566.                        cout << "Student Name: " << temp->name << endl
567.                        << "Student MatricNo: " << temp->matricno << endl
568.                        << "Student Course: " << temp->course << endl;
569.                        cout <<
    "\n************************************************" << endl;
570.                        cout << "***           FYP INFORMATION           ***"
    << endl;
571.                        cout <<
    "************************************************" << endl;
572.                        cout << "FYP Field: " << temp->FYPField << endl
573.                        << "FYP Title: " << temp->FYPTitle << endl
574.                        << "FYP Title Submission Date: " << temp->FYPDate <<
    endl;
575.                        cout << "Press any key to continue...";
576.                        cin.get();
577.                        break;
578.                    }
579.                    temp = temp->next;
580.                }
581.                if(temp == NULL)
582.                {
583.                    cout << "Not Found" << endl;
584.                    cout << "Press any key to continue...";
585.                    cin.get();
586.                }
587.            }
588.
589.            void displayFYP()
590.            {
591.                nodeQ *temp = frontPtr;
592.                int FYPcounter = 1;
593.                cout <<
    "************************************************************************
    **********************" << endl;
594.                cout << "***                                    STUDENT FINAL YEAR
    PROJECT LIST                              ***" << endl;
595.                cout <<
    "************************************************************************
    **********************" << endl;
596.
597.                cout << setw(4) << left << "No." << setw(20) << "NAME" <<
    setw(12) << "MATRIC_NO" << setw(30) << "FYP_TITLE" << setw(26) << "FYP_STATUS"
    << setw(10) << "FYP_MARK" << endl;
598.
```

```
599.                while(temp)
600.                {
601.                    cout << setw(4) << left << FYPcounter++ << setw(20) <<
    temp->name
602.                    << setw(12) << temp->matricno << setw(30) << temp->FYPTitle
603.                    << setw(26) << temp->FYPStatus << setw(5);
604.                    if(temp->FYPMark == -1)
605.                    {
606.                        cout << "TBA" << endl;
607.                    }
608.                    else
609.                    {
610.                        cout << temp->FYPMark << endl;
611.                    }
612.                    temp = temp->next;
613.                }
614.                cout << "\nPress any key to continue...";
615.                cin.ignore();
616.                cin.get();
617.            }
618.
619.            void UpdateResult()
620.            {
621.                nodeQ *temp = frontPtr;
622.                if(temp == NULL)
623.                {
624.                    cout << "EMPTY SYSTEM";
625.                    cout << "\nPress any key to continue...";
626.                    cin.get();
627.                    return;
628.                }
629.                string matricno;
630.                cin.ignore();
631.                cout << "Enter Student Matric No: ";
632.                getline(cin, matricno);
633.                while(temp)
634.                {
635.                    if(temp == backPtr&& temp->matricno!=matricno)
636.                    {
637.                        cout << "Student Not Found" << endl;
638.                        break;
639.                    }
640.                    if(temp->matricno == matricno)
641.                    {string previousstat= temp->FYPStatus;
642.                        cout << "****************************************" <<
    endl;
643.                        cout << "***           UPDATE RESULT             ***"
    << endl;
644.                        cout << "****************************************" <<
    endl;
645.                        cout << "Enter Student FYP Mark: ";
646.                        cin >> temp->FYPMark;
647.                        cout << temp->name << " FYP Mark have been updated" <<
    endl;
```

```
                        temp->FYPStatus = "Marked";
                        if(previousstat=="PresentationDone")
                        {
                           string cycleindex= markfrontPtr->matricno;
                           if(markfrontPtr->matricno==matricno)
                           {
                            markfrontPtr=markfrontPtr->next;
                           }
                           else{
                                do
                                {
                                    if(markfrontPtr->matricno!=matricno){
                                        markbackPtr->next=markfrontPtr;
                                        markbackPtr=markfrontPtr;
                                        markfrontPtr=markfrontPtr->next;
                                        markbackPtr->next=NULL;
                                    }
                                    else{
                                        markfrontPtr=markfrontPtr->next;
                                    }

   }while(markfrontPtr->matricno!=cycleindex&&markfrontPtr!=NULL);
                           }
                        }

                        temp = frontPtr;
                        fstream fout;
                        fout.open("StudentFYP.txt", ios::out);
                        while (temp)
                        {
                            fout << temp->name << endl
                            << temp->matricno << endl
                            << temp->course << endl
                            << temp->FYPField << endl
                            << temp->FYPTitle << endl
                            << temp->FYPStatus << endl
                            << temp->FYPDate << endl
                            << temp->FYPMark << endl;
                            temp = temp->next;
                        }
                        fout.close();
                        temp=markfrontPtr;
                        fout.open("MarkingOrder.txt",ios::out);
                        while (temp)
                        {
                            fout<<temp->matricno<<endl;
                            temp=temp->next;
                        }
                        fout.close();

                        cin.ignore();
                        break;
                    }
                    temp = temp->next;
```

```
701.                }
702.            }
703.
704.          void markpeek()
705.           {
706.               if(markfrontPtr==NULL){
707.                   cout<< "There is no student to be marked yet"<<endl;
708.                   cout << "\nPress any key to continue...";
709.                   cin.ignore();
710.                   cin.get();
711.               }
712.               else
713.               {
714.                   nodeQ *newnode=frontPtr;
715.                   nodeQ *temp;
716.                   while(newnode!=NULL)
717.                   {
718.                       if(newnode->matricno==markfrontPtr->matricno)
719.                       {
720.                           cout <<
   "************************************************" << endl;
721.                           cout << "***              STUDENT INFORMATION
         ***" << endl;
722.                           cout <<
   "************************************************" << endl;
723.                           cout << "Student Name: " << newnode->name << endl
724.                           << "Student MatricNo: " << newnode->matricno << endl
725.                           << "Student Course: " << newnode->course << endl;
726.                           cout <<
   "\n************************************************" << endl;
727.                           cout << "***              FYP INFORMATION
   ***" << endl;
728.                           cout <<
   "************************************************" << endl;
729.                           cout << "FYP Field: " <<newnode->FYPField << endl
730.                           << "FYP Title: " << newnode->FYPTitle << endl
731.                           << "FYP Title Submission Date: " << newnode->FYPDate
   << endl
732.                           << "FYP Status: " << newnode->FYPStatus << endl
733.                           << "FYP Mark: ";
734.                           if(newnode->FYPMark == -1)
735.                           {
736.                           cout << "TBA" << endl;
737.                           }
738.                           else
739.                           {
740.                           cout << newnode->FYPMark << endl;
741.                           }
742.                           cout << "****************************************"
   << endl;
743.                           cout << "***       UPDATE RESULT              ***"
   << endl;
744.                           cout << "****************************************"
   << endl;
```

```
745.                            cout << "Enter Student FYP Mark: ";
746.                            cin >> newnode->FYPMark;
747.                            cout << newnode->name << " FYP Mark have been
    up5dated" << endl;
748.
749.
750.                            newnode->FYPStatus = "Marked";
751.                            break;
752.                        }
753.                        else
754.                        {
755.                            newnode=newnode->next;
756.                        }
757.                    }
758.                    markfrontPtr=markfrontPtr->next;
759.
760.                    if(!frontPtr)
761.                        backPtr=NULL;
762.
763.                    fstream fout;
764.                    fout.open("StudentFYP.txt",ios::out);
765.                    newnode=frontPtr;
766.                    while (newnode)
767.                    {
768.                        fout << newnode->name << endl
769.                        << newnode->matricno << endl
770.                        << newnode->course << endl
771.                        << newnode->FYPField << endl
772.                        << newnode->FYPTitle << endl
773.                        << newnode->FYPStatus << endl
774.                        << newnode->FYPDate << endl
775.                        << newnode->FYPMark << endl;
776.                        newnode = newnode->next;
777.                    }
778.                    fout.close();
779.
780.                    temp=markfrontPtr;
781.                    fout.open("MarkingOrder.txt", ios::out);
782.                    while (temp)
783.                    {
784.                        fout << temp->matricno << endl;
785.                        temp=temp->next;
786.                    }
787.
788.                    fout.close();
789.                    }
790.            }
791.
792.    };
793.
794.    class Lecturer
795.    {
796.        private:
797.            string LectName;
```

```
798.            string password;
799.            StudentQ *studentQ;
800.        public:
801.            Lecturer()
802.            {
803.                LectName = "";
804.                password = "";
805.                studentQ = new StudentQ;
806.            }
807.            Lecturer(string name, string pass)
808.            {
809.                LectName = name;
810.                password = pass;
811.            }
812.            void setLectName(string name)
813.            {
814.                LectName = name;
815.            }
816.            void setPassword(string pass)
817.            {
818.                password = pass;
819.            }
820.            string getLectName()
821.            {
822.                return LectName;
823.            }
824.            string getPassword()
825.            {
826.                return password;
827.            }
828.            void displayFYP(StudentQ *studentQ)
829.            {
830.                cout << "Access By  :" << this->getLectName() << endl;
831.                cout << "Access Time:" << date << endl;
832.                studentQ->displayFYP();
833.            }
834.            void UpdateResult(StudentQ *studentQ)
835.            {
836.                cout << "Access By  :" << this->getLectName() << endl;
837.                cout << "Access Time:" << date << endl;
838.                studentQ->UpdateResult();
839.                cout << "\nPress any key to continue...";
840.
841.                cin.get();
842.            }
843.            void Peek(StudentQ *studentQ)
844.            {   int option=-1;
845.
846.                cout << "Access By  :" << this->getLectName() << endl;
847.                cout << "Access Time:" << date << endl;
848.                cout<<"Are you going to mark the student"<<endl
849.                    <<"1.Yes\n2.No\n";
850.                cout << "Enter your choice: ";
851.                cin>>option;
```

```
852.                if(option==1){
853.                    system("CLS");
854.                    studentQ->markpeek();
855.                }
856.                if (option<1||option>2)
857.                {
858.                    cout<<"please enter right option\n";
859.                    cout << "\nPress any key to continue...";
860.                    cin.ignore();
861.                    cin.get();
862.                    system("CLS");
863.                }
864.                if(option==2){ return; }
865.            }
866.            void display(StudentQ *studentQ)
867.            {
868.                cout << "Access By  :" << this->getLectName() << endl;
869.                cout << "Access Time:" << date << endl << endl;
870.                studentQ->display();
871.            }
872.    };
873.
874.    class Menu
875.    {
876.        private:
877.            StudentQ studentQ;
878.            Lecturer lecturer;
879.            int choice;
880.            int lecturerchoice;
881.        public:
882.            Menu()
883.            {
884.                studentQ.ReadStudent();
885.                studentQ.ReadMarkingOrder();
886.                choice = 0;
887.                lecturerchoice = 0;
888.            }
889.            void StudentAccessZone()
890.            {
891.
892.                do
893.                {
894.                    system("CLS");
895.                    cout << "****************************************" <<
    endl;
896.                    cout << "     * FINAL YEAR PROJECT MANAGEMENT *" << endl;
897.                    cout << "     *             STUDENT MENU        *" << endl;
898.                    cout << "****************************************" << endl
    << endl;
899.                    cout << "1. Check Suggested FYP Field\n";
900.                    cout << "2. Key In FYP and Student Information\n";
901.                    cout << "3. Update FYP Information\n";
902.                    cout << "4. Update FYP Status\n";
903.                    cout << "5. Display Your Information\n";
```

```
904.                    cout << "6. Check Result\n";
905.                    cout << "7. Back\n";
906.                    cout << "----------------------------------------" <<
    endl;
907.                    cout << "Enter your choice: ";
908.                    cin >> choice;
909.
910.                    switch(choice)
911.                    {
912.                        case 1: system("CLS"); CheckFYPField(); break;
913.                        case 2: system("CLS"); studentQ.enqueue(); break;
914.                        case 3: system("CLS"); studentQ.alterqueue(); break;
915.                        case 4: system("CLS"); studentQ.updatestatus(); break;
916.                        case 5: system("CLS"); studentQ.displaycertain(); break;
917.                        case 6: system("CLS"); studentQ.CheckResult(); break;
918.                        case 7: system("CLS"); break;
919.                        default:
920.                        {
921.                            cout << "Invalid Input" << endl;
922.                            cout << "Please enter a valid choice" << endl;
923.                            cout << "Press any key to continue...";
924.                            cin.ignore();
925.                            cin.get();
926.                            system("CLS");
927.                        }
928.                    }
929.
930.                }while(choice != 7);
931.
932.            }
933.            void CheckFYPField()
934.            {
935.                int FieldChoice;
936.                cout << "****************************************" << endl;
937.                cout << "   * FINAL YEAR PROJECT MANAGEMENT *" << endl;
938.                cout << "   *      SUGGESTED FYP FIELD       *" << endl;
939.                cout << "****************************************" << endl <<
    endl;
940.                cout << "1. Artificial Intelligence\n";
941.                cout << "2. Web Technology\n";
942.                cout << "3. Data Science\n";
943.                cout << "4. Machine Learning\n";
944.                cout << "5. Internet of Things\n";
945.                cout << "6. Blockchain\n";
946.                cout << "7. Game Development\n";
947.                cout << "8. Back\n";
948.
949.                do
950.                {
951.                    cout << "----------------------------------------" <<
    endl;
952.                    cout << "Enter your choice: ";
953.                    cin >> FieldChoice;
954.                    cin.ignore();
```

```
955.                    cout << endl;
956.                    switch(FieldChoice)
957.                    {
958.                        case 1: cout << "Artificial Intelligence (AI): AI is the
    branch of computer science that deals with creating machines and systems that
    can perform tasks that typically require human intelligence, such as
    understanding natural language, recognizing speech, making decisions, and so
    on. There are many different areas of AI research, such as natural language
    processing, computer vision, robotics, and so on.\n"; break;
959.                        case 2: cout << "Web Technology: Web Technology is the
    branch of computer science that deals with the development and design of
    websites and web applications. It involves the use of various technologies such
    as HTML, CSS, JavaScript, and others to create dynamic and interactive
    websites.\n"; break;
960.                        case 3: cout << "Data Science: Data Science is the
    branch of computer science that deals with the extraction of insights and
    knowledge from data. It involves the use of various techniques such as
    statistical analysis, machine learning, and data visualization to analyze and
    make predictions from data.\n"; break;
961.                        case 4: cout << "Machine Learning: Machine Learning is a
    subfield of AI that deals with the development of algorithms and models that
    can learn from data and improve their performance over time. It involves the
    use of techniques such as supervised and unsupervised learning, deep learning,
    and reinforcement learning to develop models that can make predictions and
    decisions.\n"; break;
962.                        case 5: cout << "Internet of Things (IoT): IoT is the
    branch of computer science that deals with the development of interconnected
    devices and systems that can communicate with each other over the internet. It
    involves the use of various technologies such as sensors, microcontrollers, and
    wireless communications to connect devices and enable them to collect and share
    data.\n"; break;
963.                        case 6: cout << "Blockchain: Blockchain is a distributed
    ledger technology that enables secure and transparent transactions without the
    need for intermediaries. It uses cryptography to secure transactions and
    provide a tamper-proof record of all transactions on the network.\n"; break;
964.                        case 7: cout << "Game Development: Game Development is
    the branch of computer science that deals with the development of video games.
    It involves the use of various technologies such as game engines, programming
    languages, and graphics tools to create interactive and engaging game
    experiences.\n"; break;
965.                        case 8: { system("CLS"); StudentAccessZone(); break;}
966.                        default: cout << "Invalid Input\n";
967.                    }
968.                }while(FieldChoice != 8);
969.            }
970.            void LecturerManagement()
971.            {
972.                do
973.                {
974.                    system("CLS");
975.                    cout << "*****************************************" <<
    endl;
976.                    cout << "     * FINAL YEAR PROJECT MANAGEMENT *" << endl;
977.                    cout << "     *      LECTURER MENU       *" << endl;
```

```
978.                    cout << "****************************************" << endl
    << endl;
979.                    cout << "1. Student List\n";
980.                    cout << "2. FYP List\n";
981.                    cout << "3. Update Result\n";
982.                    cout << "4. Mark Student\n";
983.                    cout << "5. Back\n";
984.                    cout << "----------------------------------------" <<
    endl;
985.                    cout << "Enter your choice: ";
986.                    cin >> choice;
987.                    system("CLS");
988.
989.                    switch(choice)
990.                    {
991.                        case 1: lecturer.display(&studentQ); break;
992.                        case 2: lecturer.displayFYP(&studentQ); break;
993.                        case 3: lecturer.UpdateResult(&studentQ); break;
994.                        case 4: lecturer.Peek(&studentQ); break;
995.                        case 5: break;
996.                        default: cout << "Invalid Input\n";
997.                    }
998.
999.                }while(choice != 5);
1000.           }
1001.           void LecturerSignUp()
1002.           {
1003.               char answer = 'N';
1004.               fstream fout;
1005.               string name, pass;
1006.               fout.open("LecturerAccount.txt", ios::app);
1007.               cin.ignore();
1008.               do
1009.               {
1010.                   system("CLS");
1011.                   cout << "****************************************" <<
    endl;
1012.                   cout << "     * FINAL YEAR PROJECT MANAGEMENT *" << endl;
1013.                   cout << "     *      LECTURER SIGN UP          *" << endl;
1014.                   cout << "****************************************" << endl
    << endl;
1015.                   cout << "Enter your name: ";
1016.                   getline(cin, name);
1017.                   cout << "Enter your password: ";
1018.                   getline(cin, pass);
1019.                   if(name != "" && pass != "")
1020.                   {
1021.                       fout << name << endl;
1022.                       fout << pass << endl;
1023.                       cout << "Sign Up Successful\n";
1024.                   }
1025.                   else
1026.                   {
1027.                       cout << "Sign Up Unsucessful\n";
```

```
1028.                 }
1029.                 cout << "Add another lecturer? (Y/N): ";
1030.                 cin >> answer;
1031.                 cin.ignore();
1032.                 if(answer == 'Y' || answer == 'y') system("CLS");
1033.                 else
1034.                 {
1035.                     system("CLS");
1036.                     break;
1037.                 }
1038.              }while(answer == 'Y' || answer == 'y');
1039.             fout.close();
1040.         }
1041.         void LecturerLogIn()
1042.         {
1043.             string name, pass;
1044.             string name1, pass1;
1045.             int count = 0;
1046.             fstream fin;
1047.             fin.open("LecturerAccount.txt", ios::in);
1048.             cin.ignore();
1049.
1050.             system("CLS");
1051.             cout << "**************************************" << endl;
1052.             cout << "   * FINAL YEAR PROJECT MANAGEMENT *" << endl;
1053.             cout << "   *      LECTURER LOG IN      *" << endl;
1054.             cout << "**************************************" << endl << endl;
1055.             cout << "Enter your name: ";
1056.             getline(cin, name);
1057.             cout << "Enter your password: ";
1058.             getline(cin, pass);
1059.
1060.             lecturer.setLectName(name);
1061.             lecturer.setPassword(pass);
1062.
1063.             while(!fin.eof())
1064.             {
1065.                 getline(fin, name1);
1066.                 if(name1 == ""){ break;}
1067.                 getline(fin, pass1);
1068.                 if(name == name1 && pass == pass1)
1069.                 {
1070.                     count++;
1071.                     break;
1072.                 }
1073.             }
1074.             if(count == 1)
1075.             {
1076.                 cout << "Log In Successful\n";
1077.                 system("PAUSE");
1078.                 system("CLS");
1079.                 LecturerManagement();
1080.             }
1081.             else
```

```
1082.               {
1083.                   cout << "Log In Failed\n";
1084.                   system("PAUSE");
1085.                   system("CLS");
1086.                   return;
1087.               }
1088.               fin.close();
1089.           }
1090.           void LecturerAccessZone()
1091.           {
1092.               cin.ignore();
1093.               do
1094.               {
1095.                   cout << "*****************************************" <<
    endl;
1096.                   cout << "*    FINAL YEAR PROJECT MANAGEMENT     *" << endl;
1097.                   cout << "*           LECTURER ACCESS ZONE       *" << endl;
1098.                   cout << "*****************************************" << endl
    << endl;
1099.                   cout << "1. First-time Lecturer\n";
1100.                   cout << "2. Lecturer Log in\n";
1101.                   cout << "3. Back\n";
1102.                   cout << "-----------------------------------------" <<
    endl;
1103.                   cout << "Enter your choice: ";
1104.                   cin >> lecturerchoice;
1105.                   switch(lecturerchoice)
1106.                   {
1107.                       case 1: {system("CLS"); LecturerSignUp(); break;}
1108.                       case 2: {system("CLS"); LecturerLogIn(); break;}
1109.                       case 3: {system("CLS"); break;}
1110.                       default:
1111.                       {
1112.                           cout << "Invalid Input\n";
1113.                           cout << "Please enter a valid choice" << endl;
1114.                           cout << "Press enter to continue...";
1115.                           cin.ignore();
1116.                           cin.get();
1117.                           system("CLS");
1118.                       }
1119.                   }
1120.
1121.
1122.               }while(lecturerchoice != 3);
1123.           }
1124.   };
1125.
1126.   int main()
1127.   {
1128.       Lecturer lecturer;
1129.       Menu menu;
1130.       StudentQ studentQ;
1131.       studentQ.ReadStudent();
1132.       studentQ.ReadMarkingOrder();
```

```
1133.       int choice;
1134.       do
1135.       {
1136.           cout << "*****************************************" << endl;
1137.           cout << "       * FINAL YEAR PROJECT MANAGEMENT *" << endl;
1138.           cout << "       *      FYP MANAGEMENT MENU       *" << endl;
1139.           cout << "*****************************************" << endl <<
    endl;
1140.           cout << "       " << date << endl;
1141.           cout << "-----------------------------------------" << endl;
1142.           cout << "User Type:\n";
1143.           cout << "-----------------------------------------" << endl;
1144.           cout << "1. Student\n";
1145.           cout << "2. Lecturer\n";
1146.           cout << "3. Exit\n";
1147.           cout << "-----------------------------------------" << endl;
1148.           cout << "Enter your choice: ";
1149.           cin >> choice;
1150.
1151.           if(choice == 1)
1152.           {
1153.               system("CLS");
1154.
1155.               menu.StudentAccessZone();
1156.           }
1157.           else if(choice == 2)
1158.           {
1159.               system("CLS");
1160.               menu.LecturerAccessZone();
1161.           }
1162.           else if(choice == 3)
1163.           {
1164.               system("CLS");
1165.               break;
1166.           }
1167.           else
1168.           {
1169.               cout << "Invalid choice" << endl;
1170.               cout << "Please enter a valid choice" << endl;
1171.               cout << "Press enter to continue...";
1172.               cin.ignore();
1173.               cin.get();
1174.               system("CLS");
1175.           }
1176.       }while(choice != 1 || choice != 2 || choice != 3);
1177.
1178.       cout << "\n*************************************" << endl;
1179.       cout << "Thank you for using FYP Management System" << endl;
1180.       cout << "          Have a nice day            " << endl;
1181.       cout << "*************************************" << endl;
1182.       system("pause");
1183.       return 0;
1184.   }
```

Figure 20 shows the external file of rStudentFYP.txt that act as input file in using for student list in the lecturer side.

StudentFYP.txt

```
NG KAI ZHENG
A21EC0101
SECVH
Game Development
ILoveGaming
PresentationDone
Tue Jan 24 14:04:49 2023
-1
Gan Heng Lai
A21EC0148
SECVH
Machine Learning
MachineGoodGood
Title Submitted
Tue Jan 24 14:04:49 2023
-1
Lew Chin Hong
A21EC0188
SECVH
Web Development
WebWebWeb
Marked
Tue Jan 24 14:04:49 2023
100
YEO CHUN TECK
A21EC0210
SECJH
Artificial Intelligence
AI Change World
Marked
Tue Jan 24 14:04:49 2023
100
Amirah
A21EC0114
SECTH
Artificial Intelligence
AI Change World
Marked
Tue Jan 24 14:04:49 2023
100
WAI JIA WEN
```

A21EC0808
SECVH
COMPUTER GRAPHICS
COMPUTER GAMES
TitleSubmitted
Tue Jan 24 14:43:37 2023
-1
Ayaka
A21EC0928
SECVH
Web Technology
Multimedia Web
Marked
Tue Jan 24 20:05:01 2023
100