KFUPM Registrar System

Software Design Document

Version 1.0

Doctor:

Dr. Hamoud Aljamaan

Team 3

Team Members:

✪ **Abdulrahman K. Al-Khaldi**	201673180
Abdulrahman M. Menshawi	201686200
Azzam M. Al-Khaldi	201694080
Mohammad A. Al-Ghamdi	201670960

Document component	Weight	Grade
Organization	5%	
Introduction	5%	
Software Architecture	20%	
User Interface Design	10%	
Structure Design	40%	
Behavior Design	20%	
	Total	

Thursday the 8th of November, 2018

Table of Contents

.1	Introduction:		3
	.1.1	Purpose:	3
	1.2.	Scope:	3
	1.3.	Overview:	3
2.	System Architecture:		3
	2.1.	Sub-systems clarification	3
	2.2.	Client-server model description:	4
	2.3.	Justifications for the architecture choice:	4
3.	User Interface Design:		5
	3.1.	Login page	5
	3.2.	Main page	5
	3.3.	Registration page	6
	3.4.	Add / Drop course(s) page	6
	3.5.	Schedule page	7
4.	Structure Design:		8
	4.1.	Class Diagram	8
	4.2.	Description of each class	8
5.	Behavior Design		12
	5.1.	State Diagrams	12
	5.2.	Description of each state	14
6.	Conclusion		15
7.	References		15

1. Introduction:

KFUPM is one of the greatest universities among Arab universities. Maintaining the systems there is done regularly. However, the registration system is quite old and it's better to be replaced with a new one. This project aims to produce a state of art registrar system to be used by KFUPM only. It's an online system that manages the students' registration, schedule, and grades. It's used as well by faculty members through which they can grade their students, take attendance, etc.....

This section will provide an overview of this Software Design Document, including the purpose, scope and overview.

1.1. Purpose:

The purpose of this document is to describe the design and architecture of the "KFUPM registrar system". It focuses on the static and interaction diagrams of all the components included in the system, as well as data representation. The intended readerships of the document are the developers of the system.

1.2. Scope:

This document describes the detailed design of the "KFUPM registrar system", including the system architecture, class diagrams, and state transition diagrams. In addition, data design will be described, both in the system and in the database. Finally, the design of the user interface will be described.

1.3. Overview:

This document will discuss the system architecture, data design, component design, user interface design, Structure Design, and Behavior Design.

2. System Architecture:

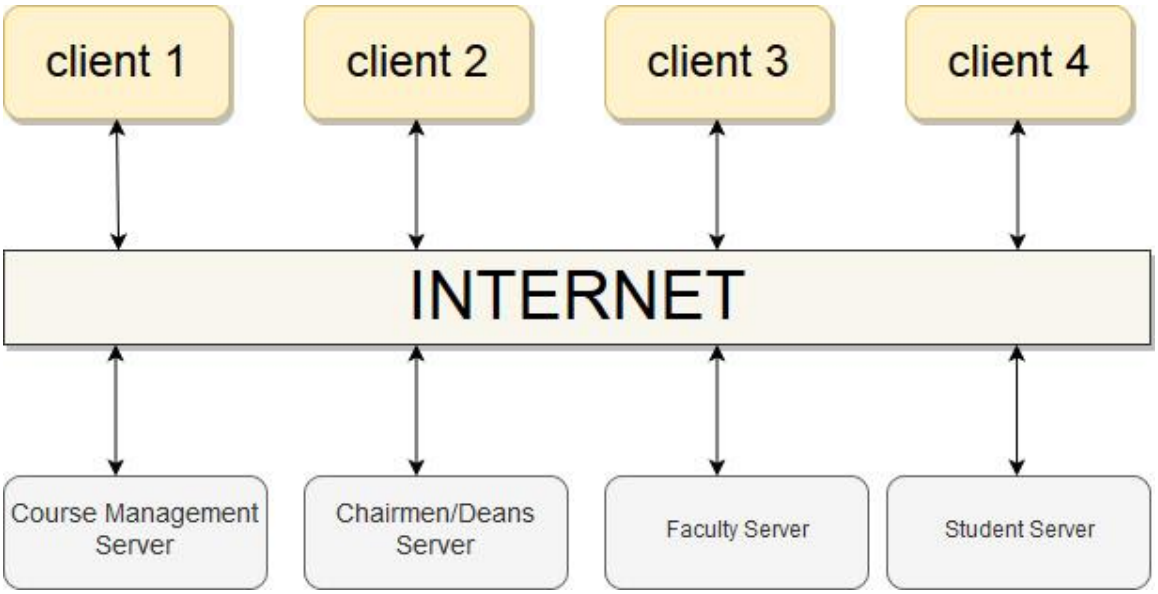
This section will describe the system architecture in terms of the architectural design and will justify the choice along with a brief description for each sub-system.

2.1. Sub-systems clarification

A. Students sub-System: The students sub-system gives the students some functionality to add/drop course search and see some information like advisor, weekly schedule, course offering etc...

B. **Faculty sub-system:** The faculty sub-system allowed the faculties to get information and change information of the students like attendance and grades and see the advising students.
C. **Administrators sub-system**: The Administrator sub-system allows the chairman and the deans to manage and change the information of the department and search for some specified information.
D. **Course Management sub-system:** The course Management sub-system allowed the course management (registrar) to manage and search for specified information.

2.2. Client-server model description:

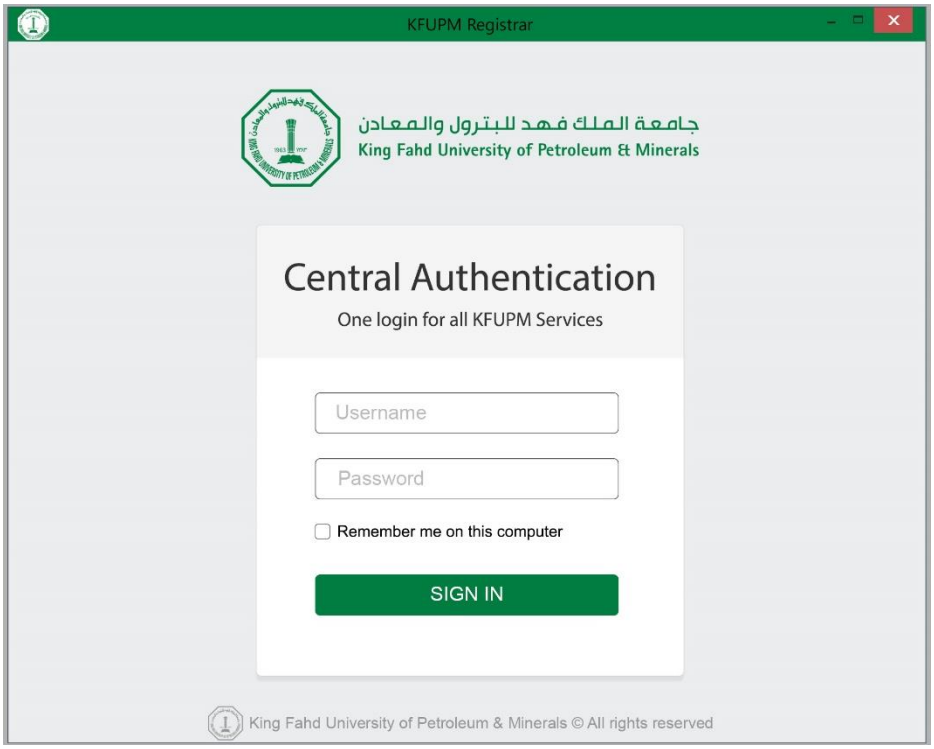


2.3. Justifications for the architecture choice:

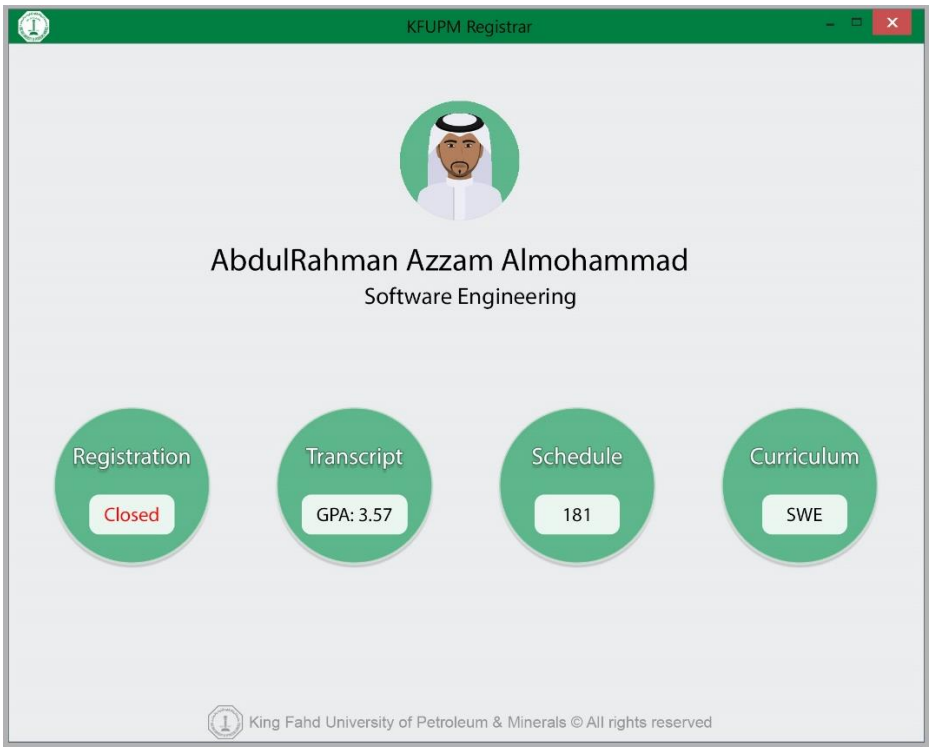
Client-server-architecture model has been chosen because it is easy to distribute the system across a wide range of clients (users) without downloading the system in every user personal computer, so general functionality can be available to all clients. Moreover, if the system needs some maintenance there is no need to shut down the whole system, we can do the maintenance for some part without affecting the other functionality of the system.

3. User Interface Design:

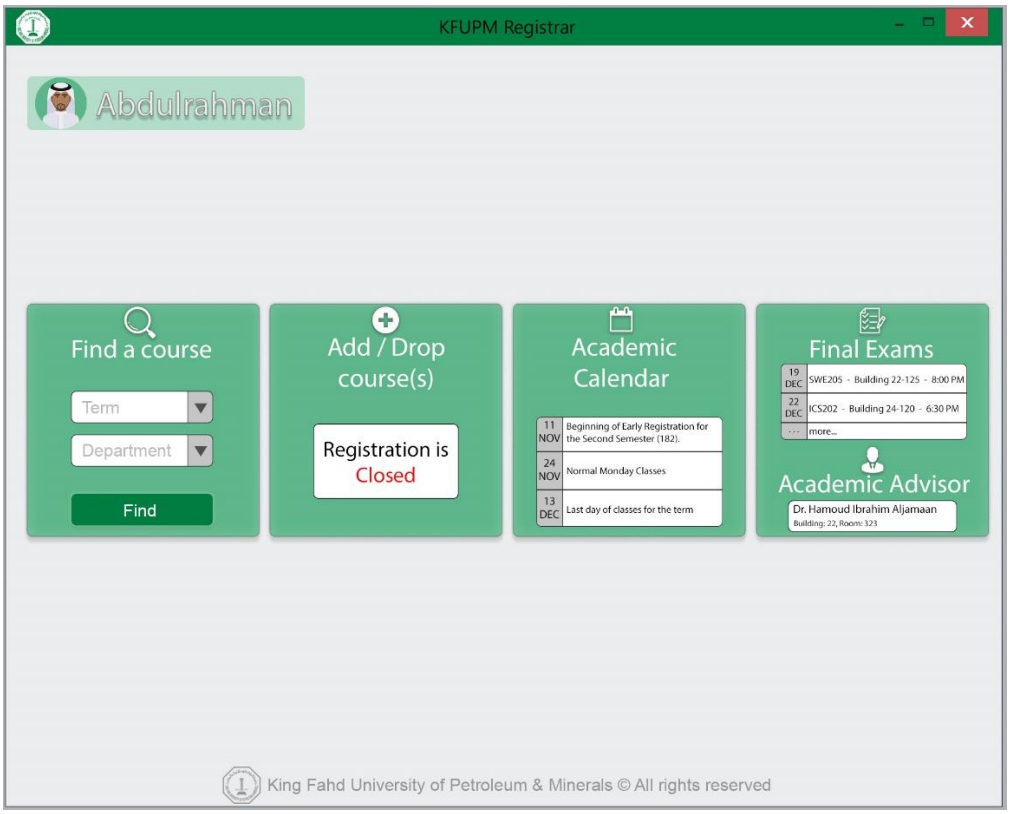
3.1. Login page



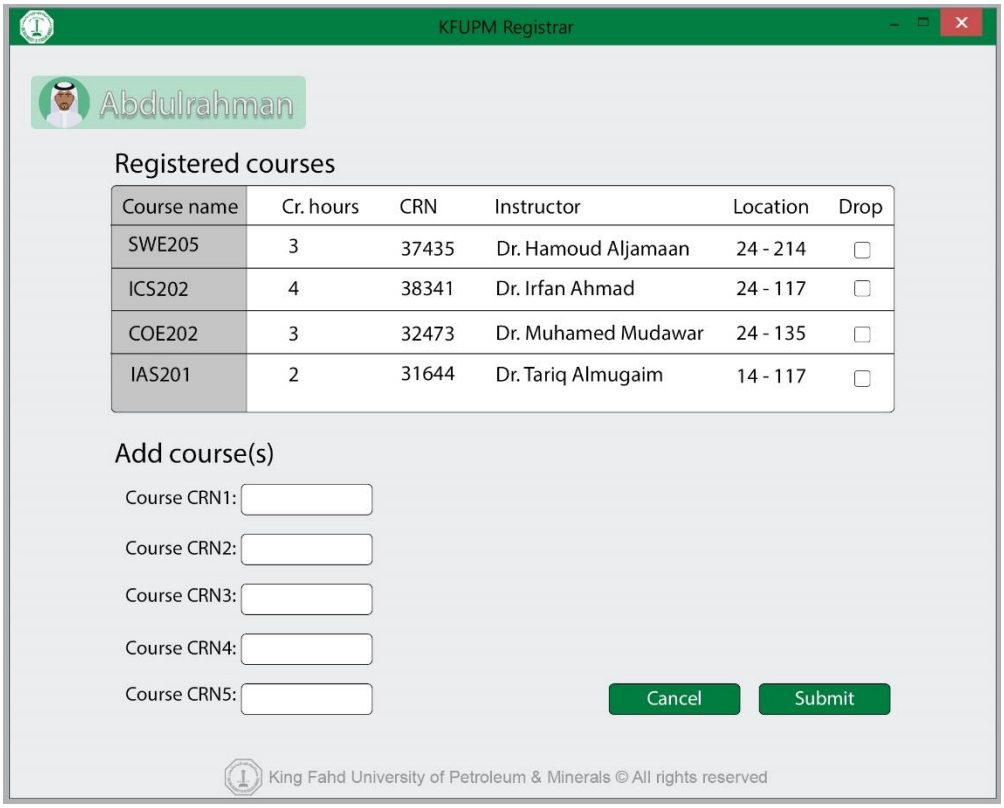
3.2. Main page



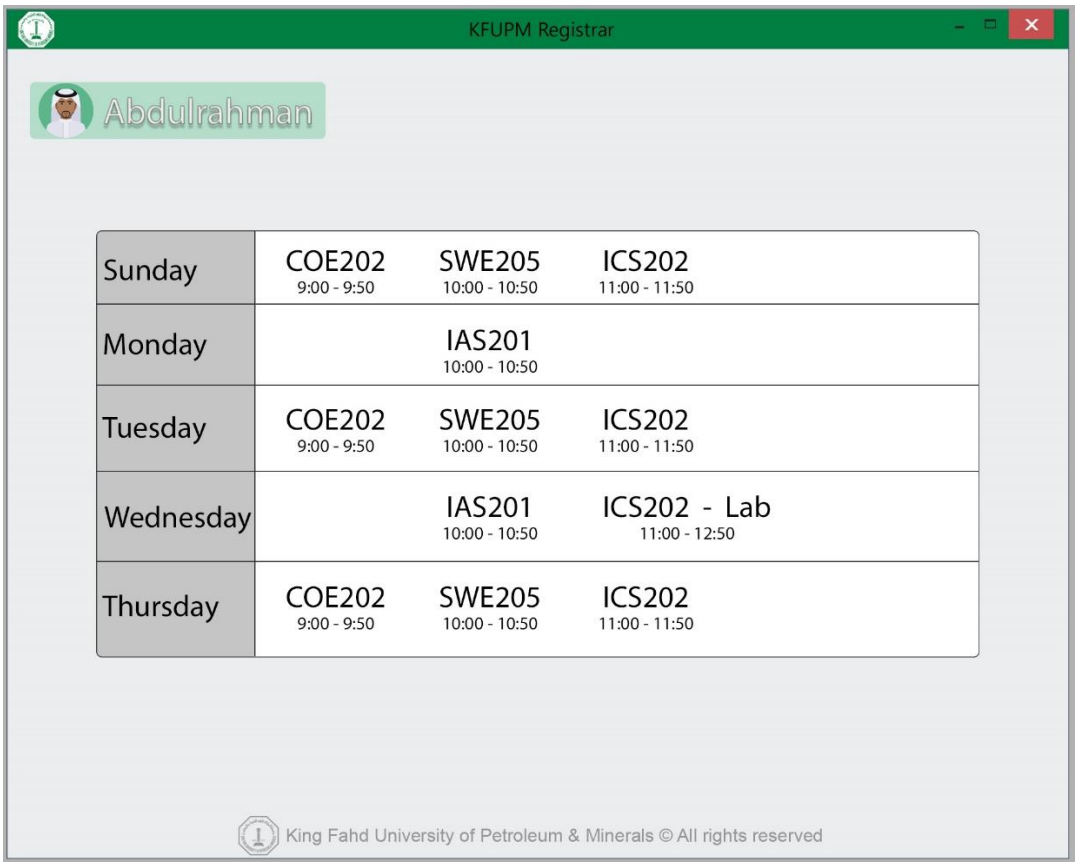
3.3. Registration page



3.4. Add / Drop course(s) page

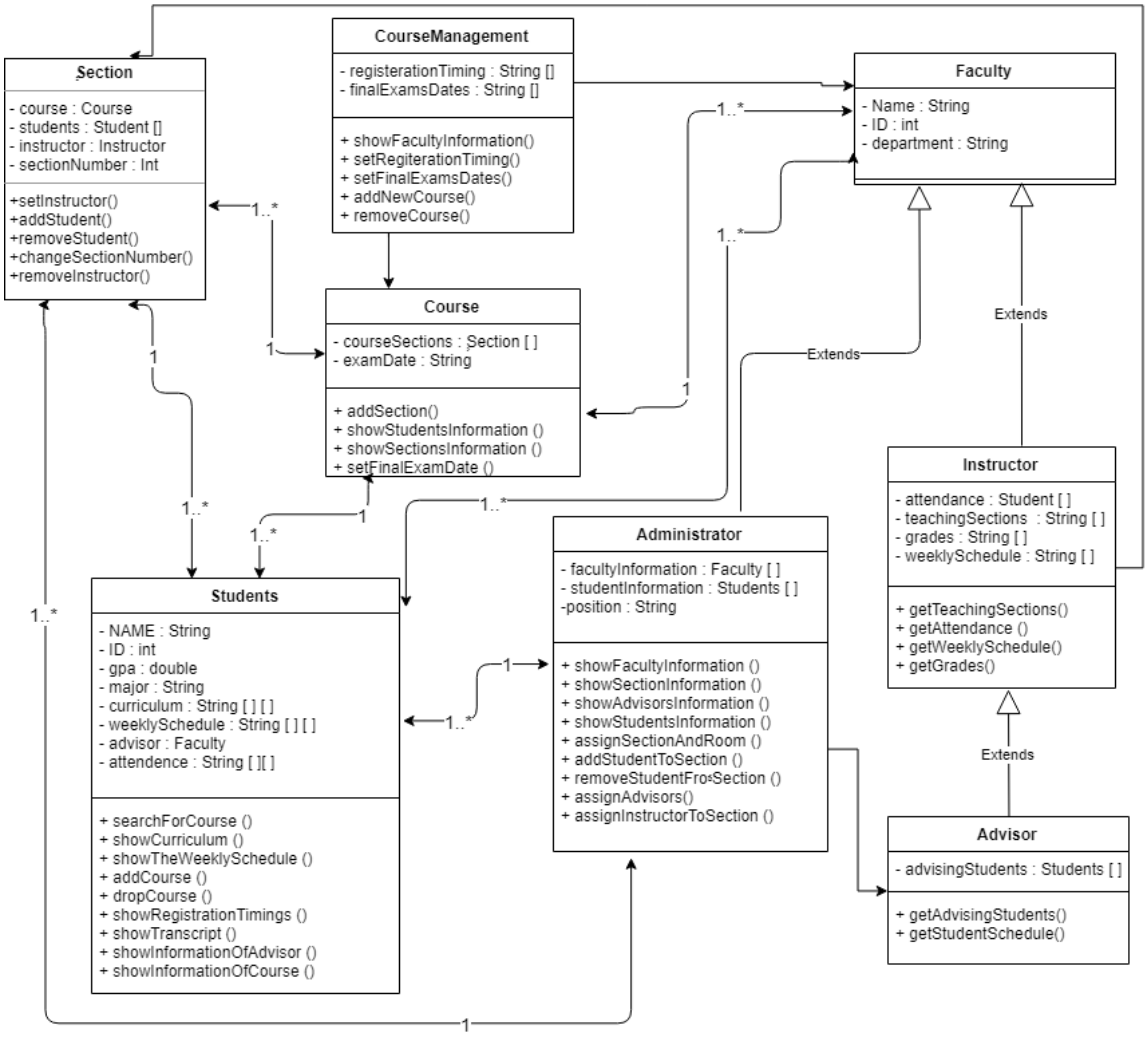


3.5. Schedule page



4. Structure Design:

4.1. Class Diagram



4.2. Description of each class

4.2.1. Course Management Class:

4.2.1.1. Attributes:

- "registerationTiming" is an array of type "String" for all registration times.
- "finalExamsDates" is array of type "String" of all courses' final exams dates.

4.2.1.2. Methods:

- "showFacultyInformation()" return the desired information from the desired faculty member from the "Faculty" class.
- "setRegisetrationTiming()" assign the registration timing for the students.

- "setFinalExamsDates()" assigns the final exams dates for all courses.
- "addNewCourse()" adds a new course of type "Course".
- "removeCourse()" removes an existing course of type "Course".

4.2.2. Course Class:

4.2.2.1. Attributes

- courseSections is an array of type Section of all the course's sections.
- examDate is a string that have the final exams date for the course.

4.2.2.2. Methods:

- addSection( ) give the registrar the ability to add sections as desired.
- removeSection () give the registrar the ability to remove sections as desired.
- showFacultyInformation ( ) the registrar can see the faculty information that help him to add section remove section etc…
- showStudentsInformation() the registrar can see the students information that help him to add student to a section or remove student from section.
- showSectionsInformation ( ) the registrar can see the section information for checking and validating the sections.

4.2.3. Section Class:

4.2.3.1. Attributes:

- "course" is an object of type Course of the assigned course to this section.
- "student" is an array of type Student of all the students in this section.
- "instructor" is an object of type "Instructor" of the assigned instructor to this section.
- "sectionNumber" is a variable of type Integer which assign's the section's number.

4.2.3.2. Methods:

- setInstructor() assign's an instructor for the section.
- "addStudent()" adds a new student of type "Student" to the section.
- "removeStudents( )" to remove student from a section as desired.
- "changeSectionNumber()" reassigns the section number.
- "removeInstructor()" to remove the instructor as desired.

4.2.4. Students class:

4.2.4.1. Attributes:

- Curriculum is a 2D array have the curriculum for the student, and it is organized depend on terms.

- weeklySchedule is a 2D array have the weekly schedule for the student, and it is organized depend on days.
- advisor is a faculty object that have the information of the student advisor like name, office, and office hours etc…
- attendance is a 2D array that have the attendance of the students, and it is organize depend on the courses.

4.2.4.2. Methods:

- "searchForCourse()" has a String parameter and return the course that the student searched for.
- "ShowCurriculum()" show the curriculum of the student as a 2D array.
- "showTheWeeklySchedule()" shows the student his weekly schedule.
- "addCourse()" get the CRN of the desired course and add it to the student.
- "dropCourse()" get the CRN of the desired course and drop it for the student.
- "showRegistrationTimings()" get the desired term and show the registration timing that set by the CourseManegment class.
- "showTranscript()" get the desired term and show the transcript of the student.
- "showInformationOfAdvisor()" return object of Faculty type that have the information of the advisor.
- "showInformationOfCourse()" return an 2D array that have the information of the searched course like timing, instructor, and section number etc….

4.2.5. Faculty class:

4.2.5.1. Attributes:

- "Name" is the name of the faculty member.
- "ID" is the id of the faculty member.
- "Department", which department the faculty member belongs to.

4.2.5.2. Methods: This class is a parent class with no methods.

4.2.6.**Instructor Class which extends Faculty Class**:

4.2.6.1. Attributes:

- "attendance" is a 2D array of type String that have the student's attendance of each course.
- "teachingSections" is an array of type String that have the information for the sections that taught by the faculty like number of students, building number, and room number etc…..
- Grades is a 2D array of type String that have student's grades of each course.
- weeklySchedule is a 2D array have the weekly schedule for the faculty, and it is organized depend on days.

4.2.6.2. Methods:

- “getTeachingSections()” get the course name and return an array of type string that have the information of the section like number of students, building number, and room number etc…
- :getAttendance()” get the attendance of students in the specified section
- “getWeeklySchedule()” show the faculty member his teaching sections and the time of each section
- “getGrades()” get the grades of students in the specified section.

4.2.7. Advisor Class which extends Instructor Class:

4.2.7.1. Attributes:

- “advisingStudents” is array of type Student that have student’s information.

4.2.7.2. Methods:

- “getAdvisingStudents()” show the advisor his advising students and their information in array of type Students.

4.2.8. Administrator class which extends faculty class:

4.2.8.1. Attributes:

- facultyInformation is an array of type faculty that is under his administration.
- studentsInformation is an array of type Student that has all the students studying in his department.
- Position is a string to declare what position does he have (Dean or Chairman)

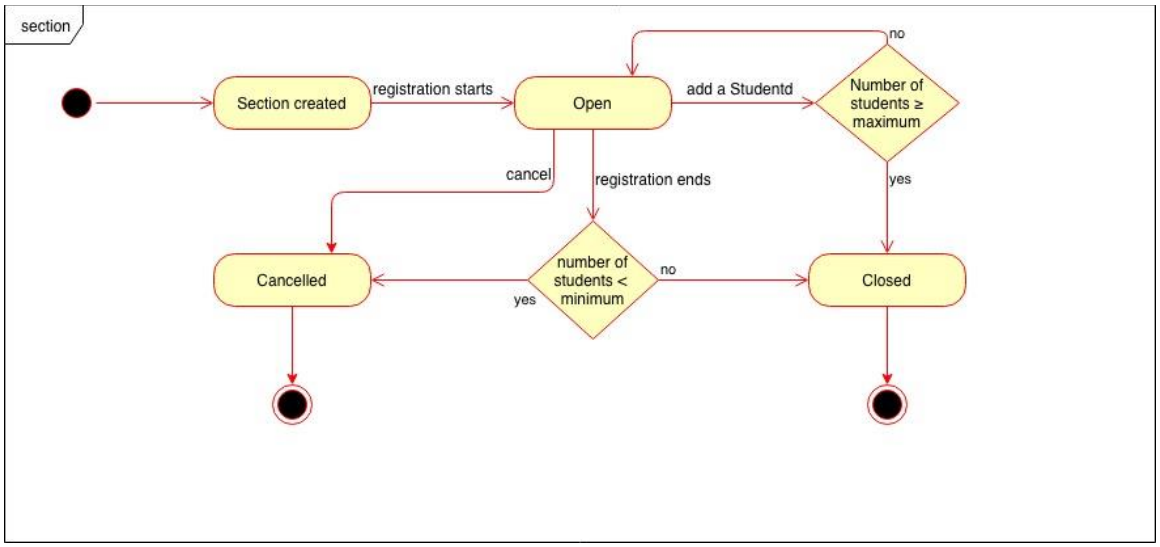
4.2.8.2. Methods:

- “showFacultyInformation()” return an array of all faculty members under his administration.
- “showSectionInformation()” has two parameters (course name, section number) which return’s an array of type Student with all registered students.
- “showAdvisorsInformation()” return an array of type Advisor with all the advisors under his administration.
- "showStudentsInformation ()”return an array of type Student with all registered students under his administration.
- "assignSectionToRoom()” has three parameters (object of type Section, building number and room number) which assign’s a section to a specific building and room.

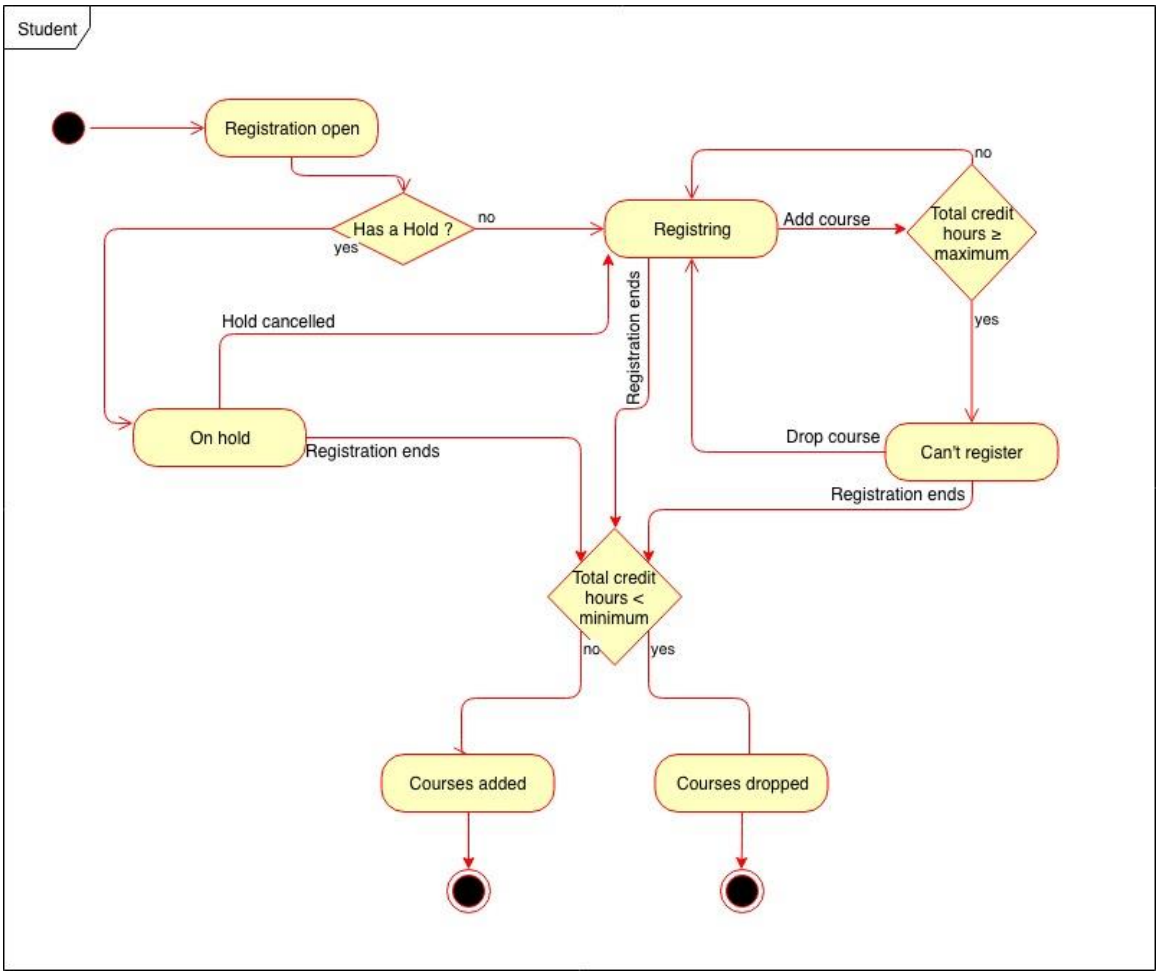
5. Behavior Design

5.1. State Diagrams

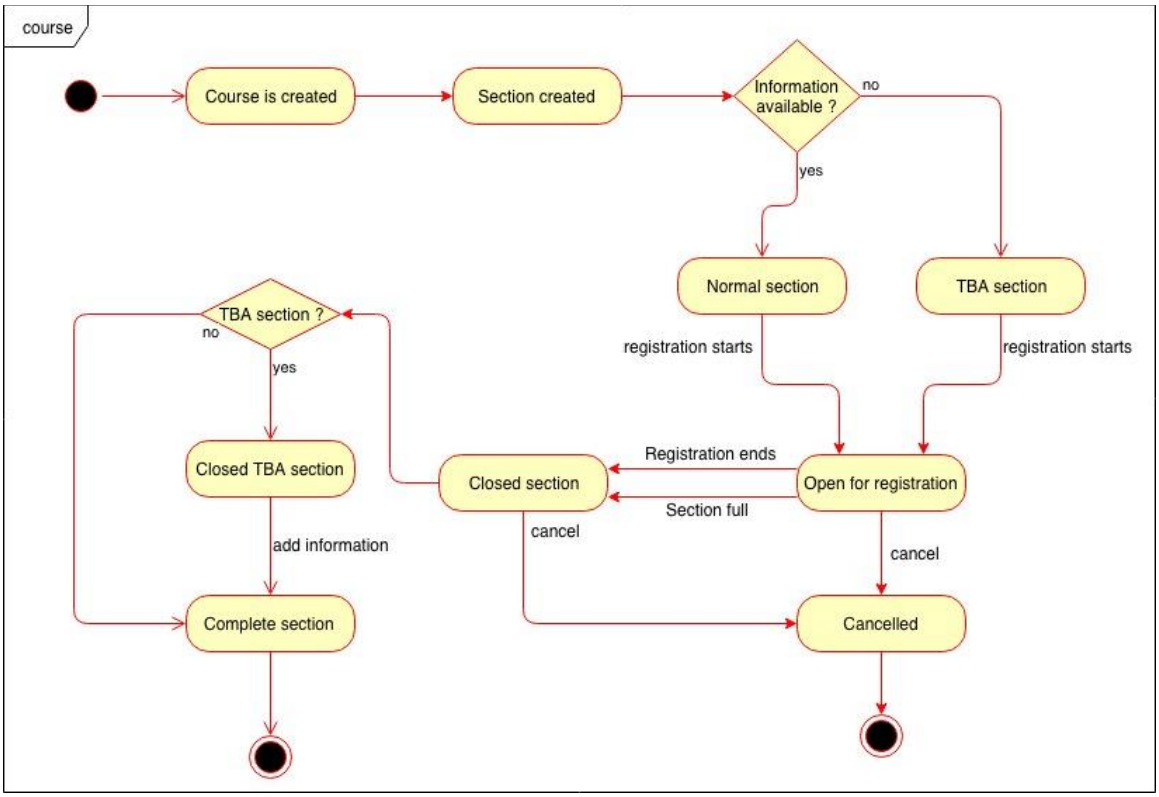
5.1.1. Section



5.1.2.Student


Student
Registration open
Has a Hold ?
yes
no
Registring
Add course
Total credit hours ≥ maximum
no
yes
Hold cancelled
Registration ends
On hold
Registration ends
Drop course
Can't register
Registration ends
Total credit hours < minimum
no
yes
Courses added
Courses dropped

5.1.3.Course



5.2. Description of each state

The description of each state transition machine diagrams is shown above in everyone of them, such that when we go from state to another there is a text that is written near to the relation link and its explain what should to be done before going to the next state.

6. Conclusion

Document History:

Date	Version	Task Description	By
November 1, 2018	1.0	Team meeting to discuss the document and distribute tasks.	All team members
November 4, 2018	1.0	UI design and Introduction.	Abdulrahman Al-Khaldi
November 5, 2018	1.0	Behavior design.	Abdulrahman Menshawi
November 5, 2018	1.0	Structure design.	Azzam Al-Khaldi
November 5, 2018	1.0	System architecture.	Mohammad Al-Ghamdi
November 6, 2018	1.0	Revision for the document.	Abdulrahman Al-Khaldi
November 7, 2018	1.0	Thoroughly revision for the entire document.	All team members

7. References

- Sommerville, Ian. *Software Engineering, Tenth Edition*, course Textbook.
- SWE205 lecture slides.
- https://www.smartdraw.com
- https://www.draw.io
- https://en.wikipedia.org/wiki/Software_architecture