Abstract

The aim of our project is to make the process of registration in the PPU done on line. To grant the student the flexibility that needed in the process of registration.

The result of this project is a web-based software system solution that provides certain requirements according to predetermined polices that enable large number of students to register through this system, surely after complying with all requirements and functions needed. As a result, the end user of our system is the student.

. In few words, we conclude that the produced system is applicable to be implemented in the PPU environment where it could provide facilities to the university and students in an efficient way.

Dedication

To the dearest… our parents

To our holy land of Palestine

To all who gave themselves and lives so others live

To every mother in Palestine

To every giving caring and decent person

To every child who holds a stone in the face of the adversary forces

Acknowledgement

It is one of the greatest experiences of our lives to see how much support and help we would find in a work like this; we would like to thank first our supervisor Eng. Ahlam quri’a for his patience, help, and most of all kindness. We would like to individually thank everyone who helped us to make this work come to life. We would not forget to thank Palestine Polytechnic University with its helpful staff especially in the Computer Center who gave us much of there time and patience too.

Last but not least we would like to thank all the individuals who contributed to this system with their ideas and advice.

TABLE OF CONTENETS

Chapter one (Introduction)

Chapter two (System Specification)

Chapter Six (System Testing)

Chapter Seven (Maintenance)

Chapter eight (Conclusion and further work)

List of tables

List of figures

Chapter 1

Introduction

Chapter 2

System Specification

Chapter 3

Software Requirements Specifications

Chapter 4

Design

Chapter 5

Coding and Implementation

Chapter 6

Testing

Chapter 7

Maintenance

Chapter 8

Conclusion and Further work

1.1 Overview

We are develop this system to generate the process of registration to be as one of the most important internet applications, which has been spotlighted by most universities in the world and this is one of the reasons that encourage us to implement our project.

The main idea in this system that it minimizes the time and cost for both student and university to complete the process of registration, this is the issue that our system demonstrates on and the system must have the minimal characteristics to serve as an appropriate and reliable solution for registration on the internet; common services should be provided by the system such as full registration.

1.2 Research Problem.

We will do our project to make the process of registration which the student must do every semester easier; this will be achieved through using the internet instead of the traditional process.
So our project is how to every student can finish his/her registration with out need to go to the university except the process of payment , which will be achieved through bank and this because its impossible to found a visa card with every student.

The concept of this project covers many things such as the loan that each student has, and the procedures and polices that related to this process, which make this process complete and has no problem, such as which course that allow to the student to take depending on his academic plan, cases that related with the academic situation if he/she has a warning or will graduated in that semester ….etc

1.3 Registration Definition.

The process of registration using the internet instead of the traditional way, to complete all steps that required to grant the insurance of acceptance to the student in that semester

1.4 Overview of the ASP.NET Technology.

✓ *What is the .NET Technology?*

The .NET Framework is a new computing platform that simplifies application development in the highly distributed environment of the Internet. The .NET Framework has two main components:

1. the common language runtime :

The common language runtime is the foundation of the .NET Framework. You can think of the runtime as an agent that manages code at execution time, providing core services such as memory management, thread management, while also enforcing strict type safety and other forms of code accuracy that ensure security and robustness. In fact, the concept of code management is a fundamental principle of the runtime. Code that targets the runtime is known as managed code, while code that does not target the runtime is known as unmanaged code.

2. NET Framework class library :

The class library, the other main component of the .NET Framework, is an, object-oriented collection of reusable types that you can use to develop applications ranging from traditional command-line or graphical user interface (GUI) applications to applications based on the latest innovations provided by ASP.NET, such as Web Forms and XML Web services.

- ✓ *Why choosing the ASP.NET Technology?*

From the portability and flexibility of the .NET Framework as a software development environment and the powerfulness of its database-web based applications; the work team chose this technology to develop the system which is a database access by the web with registration application.

- ✓ *How does it work?*

This is an object oriented technology that contains different classes of developing applications such as web applications and Windows applications. The web applications consist of web forms that support the connection with databases from different resources. Our system (*Registration*) is built on the connection between the web forms and a database built by using the MS SQL Server 2000.

- ✓ The .NET Framework aims:

- o To provide a consistent object-oriented programming environment whether object code is stored and executed locally, executed locally but Internet-distributed, or executed remotely
- o To provide a code-execution environment that minimizes software deployment and versioning conflicts.
- o To provide a code-execution environment that guarantees safe execution of code, including code created by an unknown or semi-trusted third party
- o To provide a code-execution environment that eliminates the performance problems of scripted or interrupted environments.
- o To make the developer experience consistent across widely varying types of applications, such as Windows-based applications and Web-based applications.
- o To build all communication on industry standards to ensure that code based on the .NET Framework can integrate with any other code.

2.1 Introduction

In this chapter the system is described precisely for its requirements, constraints and objectives.

Topics are covered in this chapter:

1. System objectives.
2. Functional and Non-Functional requirements.
3. Constraints.
4. Allocation.
5. Feasibility study (Alternatives, Cost-Benefit analysis, and Evaluation of risks).
6. Resources and Costs.
7. Time Schedule for development.

2.2 System objectives

1. Building the system database that helps the student and university in the registration system.
2. Accompanying with the new technologies by convert the present registration system into a web system.
3. Make the process of registration easier for students.
4. The utilization of registration employee's capabilities in other tasks.
5. Reducing the time and cost for both student and university.
6. Increasing the efficiency of the process of registration.

2.3 Functional description.

2.3.1 Functional definition:

1. Provide student with authenticated registration via specific authorization process.
2. Provide student the ability to apply the policy and standard of the university on line.
3. Give any student some of Information that available in the system.
4. Provide student with help tools and facilities.
5. Provide the administration tools and facilities via specific authorization process.

2.3.2 Requirement specification:

1. Provide student with authenticated registration via specific authorization process.

1.1 Student authentication.

1.2 Student login.

1.3 Student logout.

1.4 New student.

1.5 Forget password.

1.6 Change password.

2. Provide student the ability to apply the policy and standard of the university on line.

2.1 The student can not start the process of registration before he/she pay.

2.2 depending on the student ID, any student cannot start the process of registration if his/her college doesn't start the process of registration.

The allowable number of hours that the student can register in that semester.

The students can not registration more than 12 hours.

The students can registration 21 hours to finish his study at the university.

If the student in the normal status he/she can register 18 hours as maximum, and 12 hours as minimum.

The system will not allow to the student to take any course that depend on another course.

prequest with pass.

co_requist course.

prequest with out pass.

The system will not allow the student to take any courses that its conflict.

Conflict in time.

Conflict in the capacity of the section.

2.5.3 Conflict in the marks.

3. Give any student some of Information that available in the system.

3.1 The student can know the loan.

3.2 Each student can see how much courses he/she pass or fail.

3.3 The student can see the courses that are available in that semester.

3.4 Give the student the academic plane that he belongs to.

4. Provide student with help tools and facilities (user manual).

4.1 System should provide the student by a help.

4.2 The system provides the student by message when they needs.

5. Provide the administration tools and facilities (locally).

5.1 Administrator login.

5.2 Administrator logout.

5.3 Update on plan.

2.4 Non functional requirements

The non-functional requirements define system proprieties and constraints, it is responsible for requirements of the user that are not concerned with the software functionality or programming.

The non-functional requirements are divided into two classifications (process requirements and external requirements).

2.4.1 **Process requirements:**

- Security: A password will be needed for the user login web page. The web page has to be secure, and the passwords will be stored in a protected file.
- Reliability**:** this project will only be expected to be used with Microsoft Internet Explorer.
- High integrity and clarity of the database tables and relations.
- High speed of accessing the site.

2.4.2 **External requirements:**

1. Means of authentication should be implemented; (student id, password) should be used to permit accessibility.
2. Legal and just legal student can login to the system and start the process of registration.
3. Standards: the website should meet the requirements of the student by the same.

2.5 Constraints:

This section covers the constraints, restrictions, and obstacles that faced us during the development process, some of them are described below:

1. The system is to be developed within a fixed time interval; just eighteen weeks are available for developing the system.
2. The increase of total cost for developing and implementing the system, as explained in the feasibility study.

2.6 Allocation of roles of system developers

1. **Leader:** responsible of planning, scheduling and controlling flow of system development processes.
2. **Programmer**: responsible of the system programming, implementation testing so he must have enough experiences in the ASP.NET development environment.
3. **Software engineer:** responsible for the documentation and tracing of the development stages of the software.
4. **Interface designer**: responsible of the web design and user interface of the system.
5. **Technical administrator:** responsible for monitoring, testing and maintaining the system from the moment of implementation.

Note:

The development work team may exchange places for some times in order to benefit from each other’s experiences and skills.

2.7 Feasibility Study

For a system to be developed from scratch, the most important issue is to evaluate its benefits against its cost, doing so requires a feasibility study that shows whether the system is cost effective or not. In both cases, a feasibility study should justify the purpose for developing such a system.

In this section, we show system alternatives that could be adapted; a Cost-Benefit analysis is conducted to justify the decision for developing the system, and an evaluation of the risks that may face the system and the development process.

2.7.1 Alternatives

Which methodology is the best for developing our system? What are the available ones? What technologies are available? And which of them is the most appropriate? These questionnaires and others are covered and explained through this section.

2.7.1.1 Methodology

We can develop our system by working on two different methodologies:

A) Implementing WAN technologies; the idea of our system could be implemented developing distributed software applications that run on a network comprised of several computers and connected by a network connectivity technology (wired or wireless connections).

Advantages:

- High level of security and reliability could be achieved.

Disadvantages**:**

- Work on a limited geographical area.
- More expansive.
- The existence of networking problems.

- More time is needed to install and configure the system.

B) Implementing a web-based client server methodology: the computer running the site will act as a web server on the internet; student can view the system site on by browsing it using client machines connected to the web.

Advantages:

- Cover the entire world without geographical restrictions.
- Faster in registering new student.
- Less technical or equipments configurations needed.
- Less expensive.
- No networking problems.
- Security could be achieved to some degree.
- More scalable.

Disadvantages:

- Security issue is the major challenge.

Conclusion:

By comparing the two available methodologies and putting the need to globalize the scope of the system under light, we have chosen the second methodology (web-based client server) and we worked on it.

2.7.1.2 **Environment**

As described in choosing the methodology, the aim of our system is to make it available for student on the internet.

There are tow language support that:

- CGI (Common Gateway Interface)

Advantage:

1. Runs on many different platforms which will ensure its continued popularity.
2. CGI is still very popular with many big web sites, particularly those running on UNIX operating systems. It also runs on many different platforms.

Disadvantage:

1. CGI requires a lot of server resources, especially in a multi-user situation.
2. It is not easy for a beginner to learn how to program such modules.

- **Active Server Pages (ASP)**

Advantage:

1. A new and powerful server-side technology for creating dynamic web pages.
2. create faster, more reliable dynamic web pages with any of the programming languages supported by the .NET Framework (VB.net)
3. With Microsoft .NET, developing Web applications is much easier.
4. An infrastructure that is already built.
5. ASP.NET allows you to use a far greater selection of full programming languages

Disadvantage:

1. You had a lot of related technologies from HTML, Script, Data access, etc.

2. The tools were terrible.
3. Expensive cost.

Conclusion:

By comparing the two Environments available and putting the need to globalize the scope of the system under light, we have chosen the second language (ASP.NET) and we worked on it.

2.7.2 Cost-benefit analysis

When this system applied, the process of registration will be more easily, more efficiently because the time will be less and the space consuming (builder) less, and the process of registration will be more accurate.

2.7.3 Risk evaluation

There is an opportunity for several risks that may face our system; this section covers them accordingly with the proposed solutions.

Risks:

Nonfunctional risk:

- Required time more than the estimated time.
- New cost may appear and the budget not enough.

Operational risk:

- Incompatibility between systems with environment.
- Insufficient telecommunication data rate (network speed).

Functional risks:

- Such as requirements changes.

Operational risk:

- Some applications don’t perform as estimated such as the system interface is not comfortable or acceptable to the student.

- Appearance of new requirement after or during development stage.

Risk resolution:

- The system must work in environments that have fewer specifications.
- Study the planning process and take right steps to deliver the system in the desired state.
- Study all the requirements and identify the importance of each one.
- Increase the quality of telecommunication services, increase the data rate speed.

2.8 Resources and cost

In this section, resources and cost are described for both the development and implementation requirements.

2.8.1 Development requirement

2.8.1.1 Hardware Development Resources

The hardware requirements are fully satisfied for our project since we have the following available characteristics of the development work station:

1. This project requires a PC that can run Windows XP professional. A PC with a Pentium 3 or Pentium 4 processor can play this role. But to get faster execution and operational speed, we select to work on a (2) PC with Pentium 4 processor.

   A typical PC specification can be:

   - P4 2400 MHz clock speed
   - 512 MB RAM
   - 40 GB hard disk
   - 17’’ LG monitor
   - Keyboard + Mouse
2. Scanner HP USB.
3. UPS 1k VA.

2.8.1.2 Software Development Resources

The PC should contain the following operating system and software:

- Microsoft XP professional, With IIS and server extensions.
- Microsoft Visual Studio.net Includes ADO.NET
- DBMS (Microsoft SQL Server2000)
- Microsoft 2003 Server. With IIS and server extensions.

2.8.1.3 Human Development Resources

Project team consists of three people.

2.8.1.4 Other requirements Resources

We need to obtain some books and references to complete this system these books are relative to the subjects of ASP.NET technology, SQL server and object oriented design, papers, pens, and transportations are estimated to be $25/ week. (Total per 18 weeks is ($450)).

2.8.2 Implementation (Operational) Requirements

The implementation requirements are divided into the hardware requirements, software requirements, human requirements and additional requirements.

2.8.2.1 **Hardware Operational Requirement**

At the real operation of the system, there will be a high load on it because of the expected large number of the students who want to register in the university in that semester. So, the server that will be used should be stable and powerful to do its work in a reliable and efficient way.

1. **The following server specifications can be used:**
   - 2-CPU with 1700 MHz clock speed each.
   - 3GB RDRAM.
   - 100 GB hard disk space.
   - Motherboard Intel.
   - Floppy Disk 1.44MB.
   - CD drives read/write.
   - Keyboard and mouse.
   - 19" monitor.
2. Scanner HP USB.
3. UPS 1k VA.
4. Leased line 2MB.
5. Router.
6. Additional networking requirements, Cabling, Network adapters, Hubs

2.8.2.3 Human Operational Requirement

- The administrator (who make the requirement updates & maintenance for the system).
- Programmer.

2.8.3 Cost Estimation

This section lists the estimation costs for this system including the development costs and the implementation costs.

2.8.3.1 Development Costs

- **Hardware Costs**

Hardware Component	Cost
A PC with Pentium 4– 2400 MHz	700 $*2
Scanner HP USB.	100 $
UPS 1k VA	700 $
Total	2200 $

Table (2.1): shows Development hardware costs

- ✓ **Software Costs**

Software Component	Cost
Windows XP professional	300 $
Microsoft Visual Studio.net	800 $
DBMS (MS SQL Server 2000 support 2-cpu)	10,000 $
Windows 2003 server	1,000 $
Photo Impact 8.0	$20
Macromedia Flash 6.0	free
Microsoft Visual Studio.Net V1.0 2003	$1800
Total	13920 $

Table (2.2): shows the Development software costs for registration system.

- **Human Resources Costs**

NO.	Work Hours/developer	Cost /hour	Total human cost
3 developers	(25 hours)*(18 weeks)= 450 hours	25 $	450 hours*(3 developers)*(25 $)= 33,750 $

Table (2.3): shows the Development human costs for registration system

- **Total Development Costs**

HW. Dev. Costs	SW. Dev. Costs	Human Dev. Costs	Other Dev. Req	Total
2200$	13,920 $	33,750$	450 $	48,320$

Table (2.4): Total development cost

2.8.3.2 Implementation (Operational) Costs

- **Hardware Operational Costs**

Hardware Component	Cost
Server	5,000 $
Scanner HP USB	100 $
UPS 1k VA	700 $
Leased line, 2MB	1,000 $
Router	800 $
Additional networking requirements	600 $
Security system	25,000$
Total	32,200 $

Table (2.5: Operational hardware costs

- **Human Operational Costs**

The system needs the staff of three employees that have enough skills to run the system 24 hours to keep the system database up-to-date.

Human resource	Cost/ year
One administrator	(700 $ /month)*(12 months) 8,400 $
One programmer	(700 $ /month)*(12 months) 8,400 $

Table (2.6: Human Operational costs

- **Total operational cost**

HW. Op. Costs	Human Op. Costs	Other Op. Req. Costs	Total Op. Cost
32,200 $	16,800$	600$	49,600 $

Table (2.7: Total Operational costs

- **Total system cost**

Dev. Costs	Op. Costs	Overhead costs	Total system cost
48,320$	49,600 $	3,000 $	100,920$

Table (2.8: Total system Operational costs

These costs are debatable since the system can face hidden costs, running time and maintenance or network implementation and security costs.

2.9 Time Feasibility

In this section we show how we have allocated the given period over the development stages, the time interval that was available to develop the system was 18 weeks, we have distributed this interval over all of the development process. Table (2.4) shows the time schedule for all development tasks.

- Time Schedule:

  As shown below in Table (2.4), all of the system development tasks are distributed over the available 18 weeks. Some of these tasks were taken place in parallel.

Task	Work	Time in weeks
T1	System specification.	2
T2	Software requirement specification	2
T3	Design	6
T4	Coding and implementation	6
T5	Testing	5
T6	Maintenance	3
T7	Documentation	18

Table (2.9)Time Schedule.

- Gantt chart for time schedule:

week task	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
1	▨	▨																
2			▨	▨														
3				▨	▨	▨	▨	▨	▨									
4						▨	▨	▨	▨	▨	▨							
5											▨	▨	▨	▨	▨			
6														▨	▨	▨		
7	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨	▨

Figure (2.1) Gantt chart

3.1 Introduction

This chapter covers the software requirements specifications, by which the over all functionality for the system software will be described, as well the will be classified according to their roles in the system. Topics that are covered in this chapter are:

1. Functional description.
2. Validation criteria.
3. Information description.

3.2 Functional description.

In this section we describe the major SW functions that meet our system requirements, Table (3.1) shows the functional description for the software system which meet the definition of the requirements which were previously described in chapter two(System Specification):

System requirements	Software Functional Description
1) Provide student with authenticated registration via specific authorization process.	1.1 Student login: This function will be the only method for *Student* to login to the registration home page by *Student* ID and password. 1.2 Student logout: This function enables *Student* to end their session on the home page. 1.3 forget password: this function enable student to remember the password. 1.4 change password: this function enable student to change the password. 1.5 change password: this function enable student to change his/her password.
2) Provide student the ability to apply the policy and standard of the university on line.	2.1 the student can not start the process of registration before he/she pay, and this process occurred in the bank.

	2.2 The allowable number of hours that the student can register in that semester. 2.2.1 If the student will graduate, that's means he needs just 21 hours to finish his study at the university. 2.2.2 If the student will graduate, that's means he needs just 21 hours to finish his study at the university. 2.2.3 If student has nothing of the above points he can take 18 hours as maximum, and 12 hours as minimum. 2.3 The system will not allow to the student to take any course that depend on another course. 2.3.1 there is a course that the student must take the prequest with pass, and the student can not take it before take this course. 2.3.2 there is a course that the student must take with co_requist course. 2.3.3 The system will not allow the student to take any courses that its conflict. 2.4 The system will not allow the student to take any courses that its conflict. 2.4.1 That means if the two course offers in the same time the student can not register it. 2.4.2 The student can not register any course, if the section of this course has full. 2.5 The student can not register any course that he takes it in a previous time and his mark in it is more than 70%.
3) Give any student some of Information that available in the	3.1 the student can know the loan that he/she is granted before starting the process of registration, so he/she can

system.	determine how much he/she will pay depending on the number of hours that he/she will take. 3.2 each student can see how much courses he pass, and which one he fail and which he success in, this will make the student avoid any confuse will happen with him. 3.3 The system give the student the ability to see the courses that is available in that semester, each course has information about it like the name of it, the maximum number it will has, the time of it , the college it relate to , and a counter see how much students is Available during the process of registration. 3.4 Give the student the academic plane that he belongs to, to make his/her process so easier and more accurate.
4) Provide student with help tools and facilities.	4.1 System should provide tools for students to enable them ask for A help.
5) Provide the administration tools and facilities.	5.1 Administrator login: This function will be the only method for administrator to login to registration database (locally). 5.2 Administrator logout: This function enables Administrator to end his session. 5.3 update plane: the administrator have the ability to update on the plan (locally).

Table (3.1) Functional description.

3.2.1 Software requirements specifications

A detailed description for each software requirement is described precisely through this section:

1.1 Login student

***Function*:** Login student.

***Description*:** this function will be for student to login to home registration by login user name and password.

Inputs: user name and password.

***Source*:** student.

***Output*:** student logging into home registration.

***Destination*:** web and database server.

***Require*:** correct password and username.

***Precondition*:** none.

***Post condition*:** login into home registration.

***Side effect*:** none.

1.2 Logout student

***Function*:** Logout student.

***Description*:** this function enables student to end their

sessions on the site.

Inputs: none.

***Source*:** student.

***Output*:** student logout form home registration.

***Destination*:** web and database server.

***Require*:** none.

***Precondition*:** none.

***Post condition*:** none.

***Side effect*:** none.

1.3 new password

Function**:** new password.

Description**:** this function enables student to obtain a password if the student doesn't have a password.

Inputs: student id, new password.

Source**:** student.

Output**:** new password.

Destination**:** web and database server.

Require**:** none.

Precondition**:** none

Post condition**:** none.

Side effect**:** none.

1.4 change password

Function: change password

Description**:** this function enables student to change his/here password.

Inputs: new password.

Source**:** student.

Output**:** new password.

Destination**:** web and database server.

Require**:** none.

Precondition**:** none.

Post condition**:** none.

Side effect**:** none.

1.5 Forget password

***Function*:** Forget password**.**

***Description*:** this function enables student to remember his/here password.

Inputs: answer of the question.

***Source*:** student.

***Output*:** the password.

***Destination*:** web and database server.

***Require*:** none.

***Precondition*:** none.

***Post condition*:** none.

***Side effect*:** none.

2.1 Payment status

Function*:* Payment status**.**

Description: the student must pay in the bank before starting the process of registration

Inputs: starting the process of registration by clicking on the button it referred to.

Source: the student.

Outputs*:* allow to the student to start the process registration**.**

Destination: web and database server.

Requires: correct password and user name.

Pre-condition: pay the fee.

Post-condition: none

Side effect: none

2.2 Start college registration

Function*:* Start college registration**.**

Description: Depending on the student ID, any student cannot start the process of registration if his/her college doesn't start the process of registration.

Inputs: student id.

Source: the student.

Outputs*:* allow to the student to login to the system registration**.**

Destination: web and database server.

Requires: student id.

Pre-condition: none.

Post-condition: none

Side effect: none

2.3.1 Registration of 12 hours.

Function*:* Registration of 12 hours.

Description: If the student has academic warning, and this happen if his accumulate average less than 65%, or his major average less than 70%, in this status he/she can not registration grater than 12 hours.

Inputs: the number of hours that student want to register.

Source: the student.

Outputs: the allowable number of hours that the student can register.

Destination: web and database server.

Requires: correct password and username, knowing the academic statue**.**

Pre-condition: knowing the academic statues of the student, presented courses.

Post-condition: none

Side effect: none

2.3.2 Registration between (0 to 21) hours.

Function: Registration between (0 to 21) hours.

Description: If the student will graduate, that’s means he needs just 21 hours to finish his study at the university.

Inputs: the number of hours that student want to register.

Source: the student.

Outputs: the allowable number of hours that the student can register.

Destination: web and database server.

Requires: none**.**

Pre-condition: none.

Post-condition: none

Side effect: none

2.3.3 Registration of 18 as maximum or 12 as minimum.

Function*:* Registration of 18 as maximum or 12 as minimum.

Description: If the student has nothing of the above points he can take 18 hours as maximum, and 12 hours as minimum.

Inputs: the number of hours that student want to register.

Source: the student.

Outputs: the allowable number of hours that the student can register.

Destination: web and database server.

Requires: none**.**

Pre-condition: none.

Post-condition: none

Side effect: none

2.4.1 Preqest course

***Function**:* Preqest course.

Description: There is a course that the student must take the prequest with pass, and the student can not take it before take this course.

Inputs: the course that the student wants to take.

Source: the student.

***Outputs*:** the same course that the student selects.

Destination: web and database server.

Requires: The course is presented in that semester.

Pre-condition: Success in prerequisite course**,** pay the fee**.**

Post-condition: none.

Side effect: none.

2.4.2 with out Preqest course

Function*:* with out Preqest course**.**

Description: There is a course that the student must take the prequest with pass, and the student can not take it before take this course.

Inputs: the course that the student wants to take.

Source: the student.

Outputs**:** the same course that the student selects

Destination: web and database server.

Requires: The course is presented in that semester

Pre-condition: the student must take the prerequisite course without Success in it.

Post-condition: none.

Side effect**:** none**.**

2.4.3 co_requist course.

Function: co_requist course.

Description: There is a course that the student must take with co_requist course.

Inputs: the course that the student wants to take.

Source: the student.

***Outputs*:** the same course that the student selects

Destination: web and database server.

Requires: The course is presented in that semester.

Pre-condition: none.

Post-condition: none.

Side effect: none**.**

2.5.1 Conflict in time

Function*:* the student can not take two courses if the courses time conflict**.**

Description: if the student want to take tow courses they must not be conflict in there time.

Inputs**:** the courses that the student want to take.

Source**:** the student.

Outputs**:** the same courses.

Destination: web and database server.

Requires: courses with out conflict, courses is presented in that semester

Pre-condition: none.

Post-condition: none.

Side effect**:** none**.**

2.5.2 Conflict in capacity

Function*:* Conflict in section**.**

Description: The student can not register any course, if the section of this course has full.

Inputs: the courses that the student want to take.

Source: the student.

Outputs: the same courses.

Destination: web and database server.

Requires: courses with out conflict, courses is presented in that semester

Pre-condition: none.

Post-condition: none.

Side effect: none**.**

2.6 The mark of the course must be less than 70 if the student wants to register.

Function: The mark of the course must be less than 70 if the student wants to register.

Description: If the student wants to take a course that he takes it before, his mark in that course must be less than 70% other wise he can't register it.

Inputs: the course that the student wants to register.

Source: the student.

Outputs: can not register the same course

Destination: web and database server.

Requires: none.

Pre-condition: none.

Post-condition: none

Side effect: none

3.1 the student can know the loan

Function*:* the student can know the loan.

Description: the system must give the student the value and the type of loan that he/she is granted.

Inputs: none.

Source: the student.

Outputs: the amount of loan that the student he will take**.**

Destination: web and database server.

Requires: none.

Pre-condition: none.

Post-condition: none.

Side effect: none.

3.2 Student can see how much courses he/she pass *or fail.*

Function*:* student can see how much courses he pass *or fail.*

Description: each student can see how much courses he pass, and which one he fail and which he success in, this will make the student avoid any confuse will happen with him.

Inputs: none.

Source: the student.

Outputs: each course with its marks**.**

Destination: web and database server.

Requires: none.

Pre-condition: none.

Post-condition: none

Side effect: none

3.3 course Offer

Function*:* course Offer**.**

Description: The system give the student the ability to see the courses that is available in that semester, each course has information about it like the name of it, the maximum number it will has, the time of it , the college it relate to , and a counter see how much students is available during the process of registration.

Inputs: none.

Source: the student.

Outputs: the course which offers in that semester**.**

Destination: web and database server.

Requires: none.

Pre-condition: none.

Post-condition: none

Side effect: none

3.4the academic plane

***Function**:* the academic plane.

Description: Give the student the academic plane that he belongs to, to make his/her process so easier and more accurate.

Inputs: none.

Source: the student.

Outputs: the academic plane.

Destination: web and database server.

Requires: none.

Pre-condition: none.

Post-condition: none.

Side effect: none.

4.1*viewing help*

***Function**: viewing help*

Description: The student should provide with help.

Inputs: none**.**

Source: system database.

Outputs: help content.

Destination: none.

Requires: *None.*

Pre-condition: none.

Post-condition: none.

Side effect: none**.**

5.1 login administrator

***Function*:** login administrator.

Description: This function will be the only method for administrator to login to registration database (locally).

Inputs: User Name, Password.

Source: registration database administrator**.**

Outputs: Login to his session.

Destination: changes on administrator related account contents.

Requires: administrator name, administrator password**.**

Pre-condition none.

Post-condition: none.

Side effect: none.

5.2 logout administrator

Function*:* logout administrator**.**

Description: This function will be the only method for administrator to login to registration database (locally).

Inputs: none.

Source: Administrator.

Outputs: Session is ended.

Destination: database server.

Requires: none.

Pre-condition: none.

Post-condition: none.

Side effect: none.

5.3 Allow the administrator to update the database locally

Function: updating database locally.

Description: This system allows the administrator to alter the database using authentication and authorization.

Inputs: none.

Source: information of administrator from database.

Outputs: updated database.

Destination: database server.

Requires: none.

Pre-condition: none.

Post-condition: this information can be updated continuously.

Side effect: none.

3.3 Validation Criteria:

As the system has to meet all the requirements that were previously defined and specified, we have included here the criteria by which we ensure requirements validation:

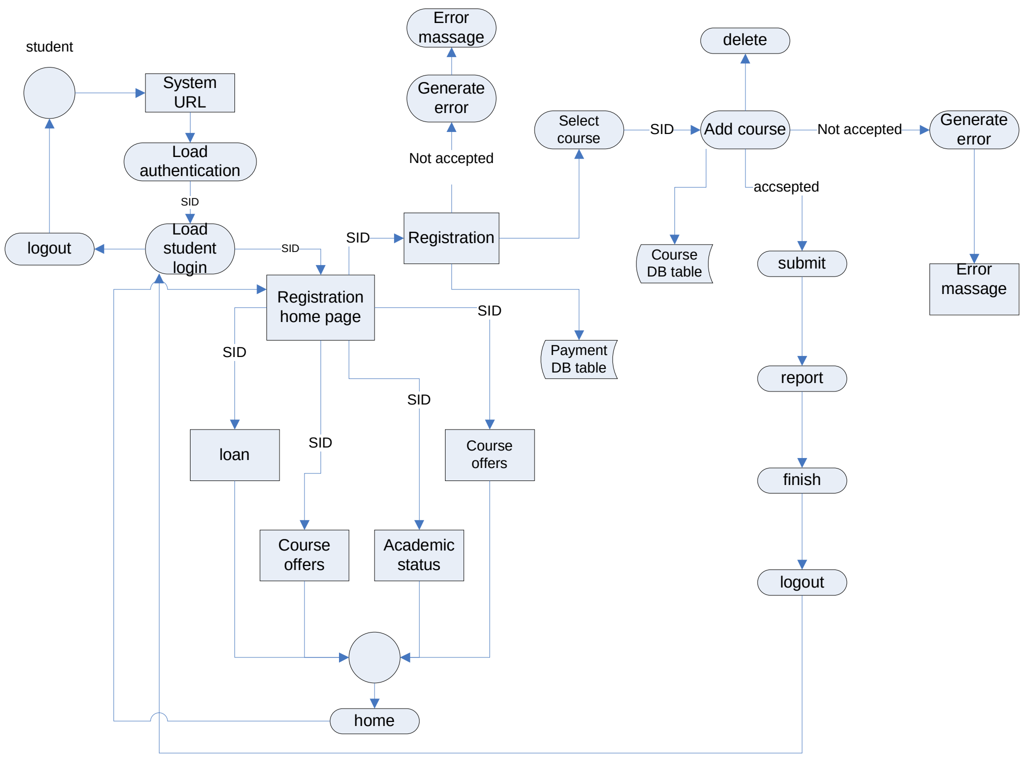
- Passwords have to be six characters at least for student.
- Passwords have to not contain any space.
- No special characters would be allowed within a password string such as (semicolon, etc).
- Student may be the same password and different student id.
- Each student should have a unique session ID whenever he/she enters the site.
- Student id should not contain special characters (hyphen or semicolon).
- Student id should have numbers (0-9).
- Student id should not contain spaces.
- Each student input should be checked against all constraints.
- The expression of each student input should be checked before it is accepted for further processing.

3.4 Information description

During this section, an over all description for the system information is discussed, as well a description for the database requirements and database model is available.

3.4.1 **System Data Flow Diagrams**

The system general data flow diagram is shown in figure (3.1) as data flow diagram, which generally shows the flow of data within the system.



3.4.2 **Data Dictionary**

The terms that are used in developing the system and during the documentation are described precisely in the data dictionary that appears below in table (3.2).

Entity name	Type	Description
o Student registration	Function	This function enables student to register new course according to his/her planning.
o Login student	Function	This function enables student to get into their related account page to view their mark and register for new semester by using their login name, password.
o Logout student	Function	The student should be able end his session.
o Add selected course	Function	This function enables student to determine which course they want to registration.
° Authenticate Inputs	Stored procedure	Validate student ID and password
o ID, password	String	ID (login name) and password is the string pair that is used by student to log into home page.
o Validation expression	String	This string is the code that is used to ensure that any student of the system has typed the right data types and formats into certain fields.
o Session ID	String	Is a random collection of characters that are generated automatically by the site to assign them to each student uniquely to differentiate him from others logging simultaneously.
o Site URL	String	"Universal Recourse Locator", which is the address of the system web site on the WWW.
o Generate report	Function	This is a function that generates a report of the purchased products, their prices, and other delivery details directed to the customer's screen after completing the purchasing process.

o Login administrator	Function	A function that enables the administrator to sign on his account legally.
o Logout administrator	Function	This function makes it possible for the administrator after finishing his work on the system database to sign out his account, and so preventing any unauthorized access.
° WWW	Abbreviation	World Wide Web.
° AVG	Abbreviation	Average.

Table (3.2) Data dictionary.

3.4.3 **System Interface description**

The interface between our system components, functions, modules, and subsystems are to be reliable and integrated, that is, all functions and other components are designed and implemented in a way that ensures the correctness of collaboration between them.

Doing so requires a clear definition of the exchanged parameters and their types and orders, a reliable methodology for dealing with shared memory along with the input/output resources, and other issues that could appear during system design and system testing phases. (Design and Testing phases are covered in chapter 4 and 6 respectively).

According to the user interface, the System will have an interface with three user categories; we have previously classified them as customers, suppliers, and administrators.

In this section we describe the system interface with each user category:

a) student Interface :

- Input:

The interface between student and the system is controlled to ensure that he entered parameters of the same type as they was identified during system development by using validation techniques available, the student will be permitted to input just valid parameters type within an allowed range of values.

- Output:

  The output for student must give them a sense of what is happening in each process, as they choose a certain course; they have to see this course added to their report, that generate after the selected course process in the system registration.

- GUI

  The graphical user interface must be easy and attractive for student; it should provide some what guidance to make the process of registration easy.

  All what we can say here, that GUI must be developed to facilitate customers to investigate easily and registration securely. This topic will be covered in details in chapter 4 (System Design).

3.4.4 **Data base requirements:**

In this section we will demonstrate the tables which will be build, and the data which must store in the data base.

1. Information about Student table :

- ✓ Student number: Each student has a unique student ID.
- ✓ Student name: Contain name of each student.
- ✓ Student level: determine the level of the student.
- ✓ Student password: A unique characters for each student, these characters should encrypt in some encryption methods to increase the level of security.
- ✓ Major number: Each department has a unique number.
- ✓ Plan number: Each plan has a unique number.
- ✓ Year: determine the year that the student register in the university and the type of it must be date.
- ✓ ID: Determine the student identity.

2. Information about mark table :

- ✓ Student number. Each student has a unique student ID.
- ✓ Course number. Each course has a unique student ID.
- ✓ Semester: determine when the student takes the course.
- ✓ Mark: this field determines the mark of the course which the student takes.
- ✓ Year: The year that the student takes the course.
- ✓ Flag: Must be 0 when the course hours collect in the number of the course that the student takes.

3. Information about course offers :

- ✓ Offer serial: Use as index of the table.
- ✓ Course number. Each course has a unique number.
- ✓ Semester: determine when the student takes the course.
- ✓ Instructor number. Each instructor has a unique instructor ID.
- ✓ Section: determine the number of section for each course.
- ✓ Capacity: maximum number of student for each section.

- ✓ Counter: that count the number of student when register any course.
- ✓ Year: Determine the Year that the curse offers in it.
- ✓ Major number: The number of the major.

4. Information about plans:
   - ✓ Plane number. Each plan has a unique number.
   - ✓ Plan year: determine the year of the plan for each major.
   - ✓ Major number. Each major has a unique number.
   - ✓ Plan picture: The picture that demonstrate the tree of the courses.
   - ✓ Required hours: The number of hours in student plan.
5. information about major table :
   - ✓ Major number: Each major has a unique number.
   - ✓ Major name: determine the name of the major.
   - ✓ Department number. Each department has a unique number.
6. Information about department table:
   - ✓ Department number: Each department has a unique number.
   - ✓ Department name: A string that present the name of the college.
   - ✓ College number: each college has a unique number.

7. information about college table :
   - ✓ College number: Each college has a unique number.
   - ✓ College name: A string that present the name of the college.
   - ✓ Start date: The date when the college start the registration.
   - ✓ End date: The date when the college end the registration.

8. Information about previous course:
   - ✓ Course number: Each course has a unique number.
   - ✓ Previous number. Each Previous course has a unique number.
   - ✓ Previous type: The type of the previous course.
   - ✓ Plan number. Each plan has a unique number.

9. Information about instructor table:

- ✓ Instructor number: Each Instructor has a unique number.
- ✓ Instructor name: Contain the information about the full name of each Instructor that includes First Name, Last Name, and Middle Name.

10. Information about student registration table:

- ✓ Student number: Each student has a unique student ID.
- ✓ Semester: determine when the student takes the course.
- ✓ course number: A unique number for the course.

11. Information about student affair table:

- ✓ Student number: Each student has a unique student ID.
- ✓ Year: determine the year that the student register in the university and the type of it must be date.
- ✓ Semester: determine when the student takes the course.
- ✓ Loan value: determine the value of the loan.
- ✓ Loan type: the type of the loan, ex.

12. Information about payment table:

- ✓ Student number: Each student has a unique student ID.
- ✓ Semester: determine when the student takes the course.
- ✓ Payment statues: determine if the student pay or not in the bank.
- ✓ Sum of hours: determine the sum of hours.
- ✓ Year: The year that the student paid in it.

13. Information about course info table:

- ✓ Course number: Each course has a unique number.
- ✓ Course name: A string that present the name of the course.
- ✓ Credit: determine the number hours for each course.
- ✓ Practical: determine if the course has a lab or not.

- ✓ Course type: The type of the course, ex: Prequist course.
- ✓ Course level: determine the course of the level.
- ✓ Plan number: Each plan has a unique number.

14. Information about schedule table:

- ✓ Offers serial: Use as index of the table.
- ✓ Room number: Each room has a unique number.
- ✓ Time: the time that the course will take in it and the entire course in the same semester must be not conflict.
- ✓ Days: Determine the day that the course will be taken in it.

15. Information about forget table: is table contains question number, student number and answer.

- ✓ Question number: A unique number for the question.
- ✓ Student number: A unique number for the student.
- ✓ Answer: The answer of the question.

16. Information about question table:

- ✓ Question number: A unique number for the question.
- ✓ Question name: The question.

4.1 Introduction

This chapter describes the system design, the functional design for all modules in the software system, I/O design, and database design.

Topics that are covered in this chapter:

1. Functional design.
2. I/O Design.
3. Database Design.
4. Test plan.
5. Programming Language and Coding.

4.2 Functional design

This section describes the functional design for each module in the software system, accordingly with the description of the interface, the constraints, and the user interface design in which we use means of diagramming to help us understand it:

1. Student authentication.

a) Description: this function determines if the student has the authentication to login to the system.

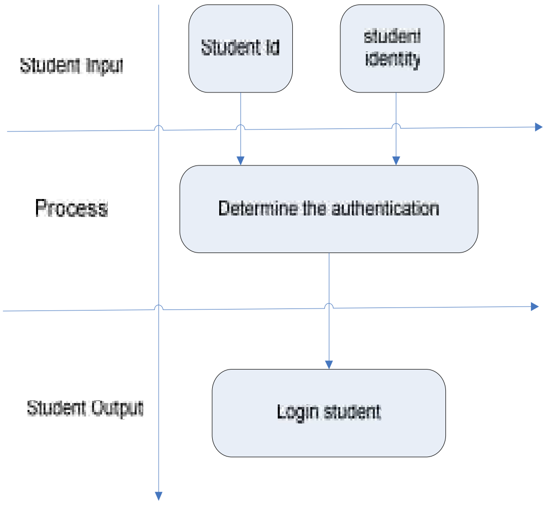
b) Interface:

- ✓ Inputs: student id, student identity.
- ✓ Outputs: login student.

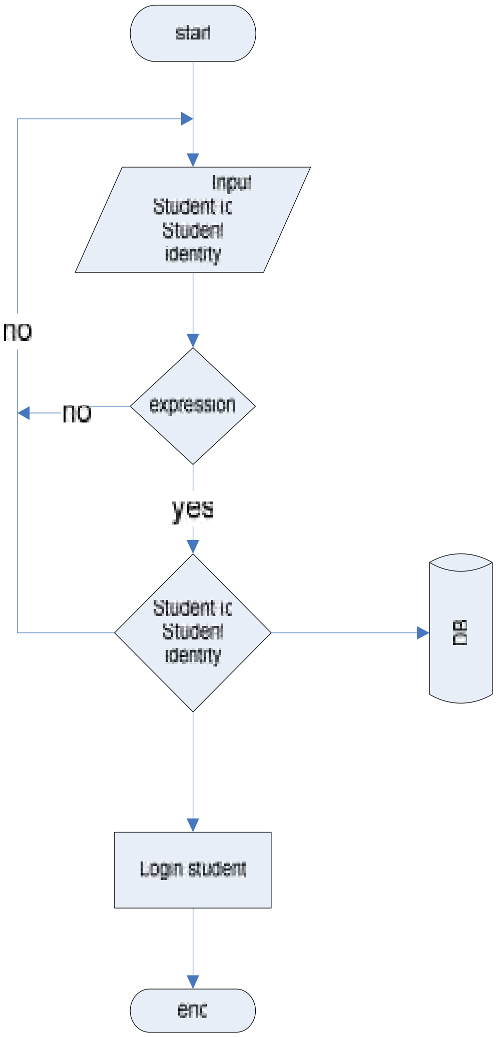
c) Constraints:

- ✓ Student id should have numbers (0-9).
- ✓ Student identity must be number (9).

d) User interface design:



e) Flowchart:


start
Input
Student ic
Student
identity
no
expression
no
yes
Student ic
Student
identity
DB
Login student
enc

2. Student Login.

a) Description: this function enables student to login to the system.

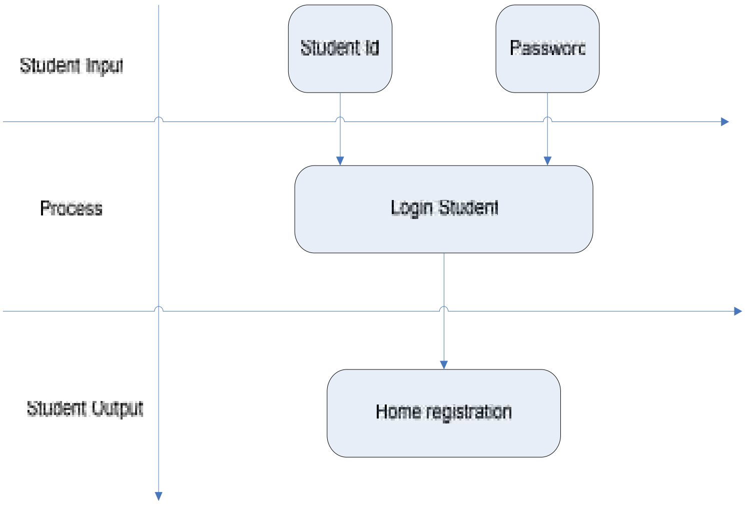
b) Interface:

- ✓ Inputs: student id, password.
- ✓ Outputs: registration home page.

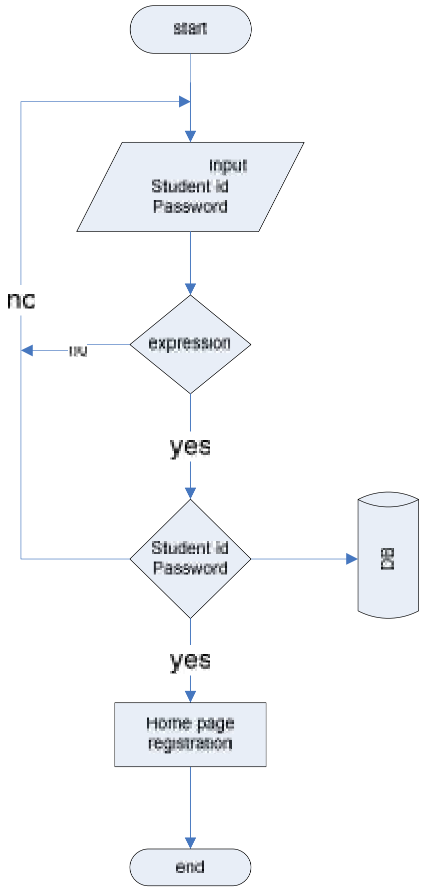
c) Constraints:

- ✓ Student must previously be registered on the university at least for one semester.
- ✓ Student id should have numbers (0-9).
- ✓ The student has the right student id and password.

d) User interface design:



e) Flowchart:


start
input
Student id
Password
nc
no
expression
yes
Student id
Password
DB
yes
Home page
registration
end

3. Student Logout.

a) Description**:** this function enables student to logout from the system.

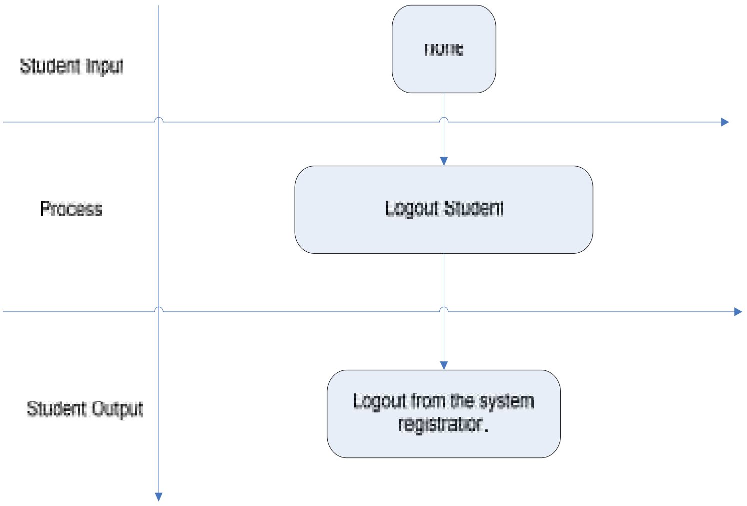
b) Interface:

- ✓ Inputs: select logout.
- ✓ Outputs: logout from the system registration.

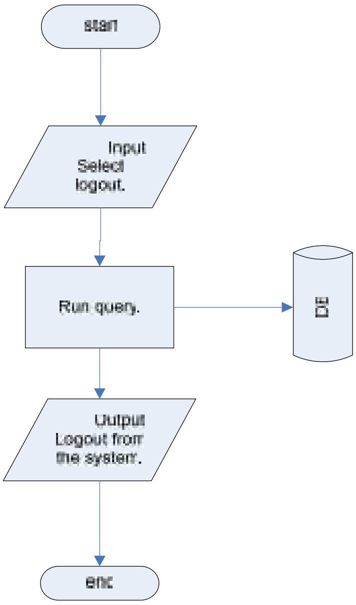
c) Constraints:

- ✓ Registered student must have already logged in.

d) User interface design:



e) Flowchart:


start
Input
Select
logout.
Run query.
DB
Output
Logout from
the system.
end

4. New Student.

a) Description**:** this function enables student to obtain a password.

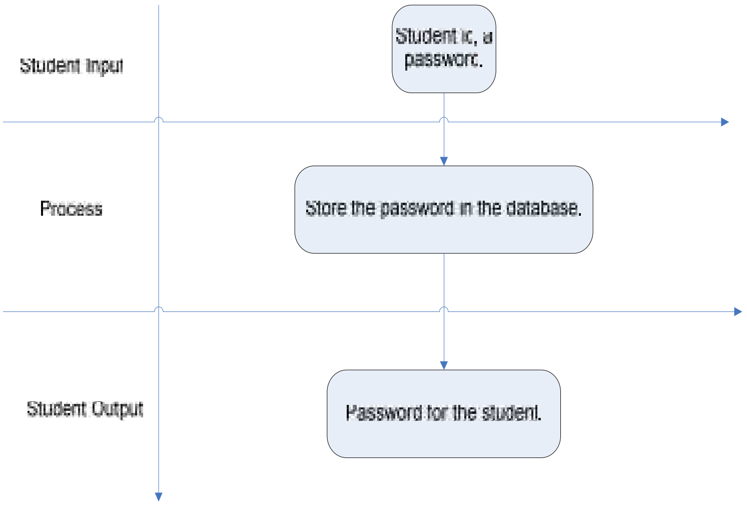
b) Interface:

- ✓ Inputs: student id, password.
- ✓ Outputs: obtain to a password.

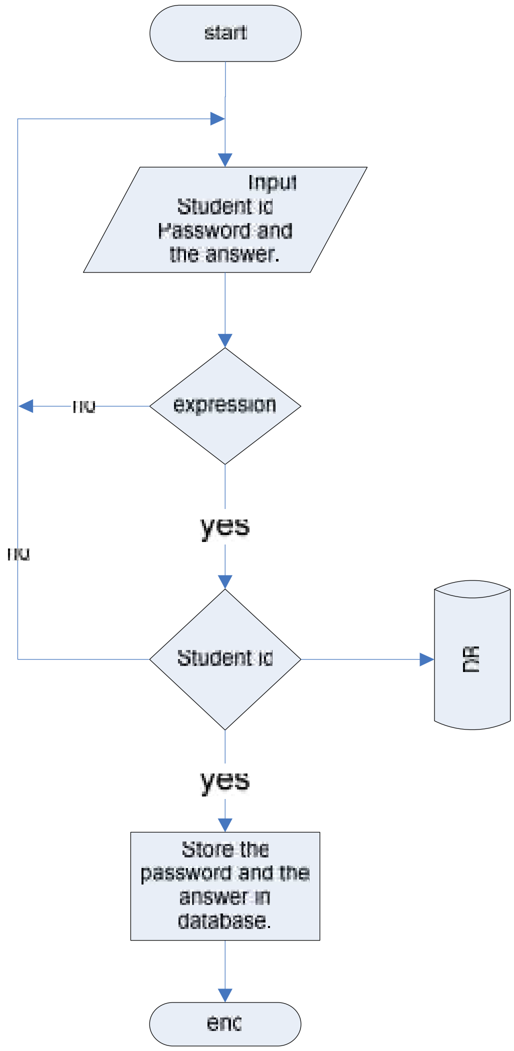
c) Constraints:

- ✓ Password has to be six characters at least for student.
- ✓ Password has to not contain any space.
- ✓ No special characters would be allowed within a password string such as (semicolon, etc).

d) User interface design:


Student Input
Student id, a password.
Process
Store the password in the database.
Student Output
Password for the student.

e) Flowchart:


start
Input
Student id
Password and
the answer.
expression
no
yes
no
Student id
DB
yes
Store the
password and the
answer in
database.
end

5. Change a password.

a) Description**:** this function enables student to change his/her password.

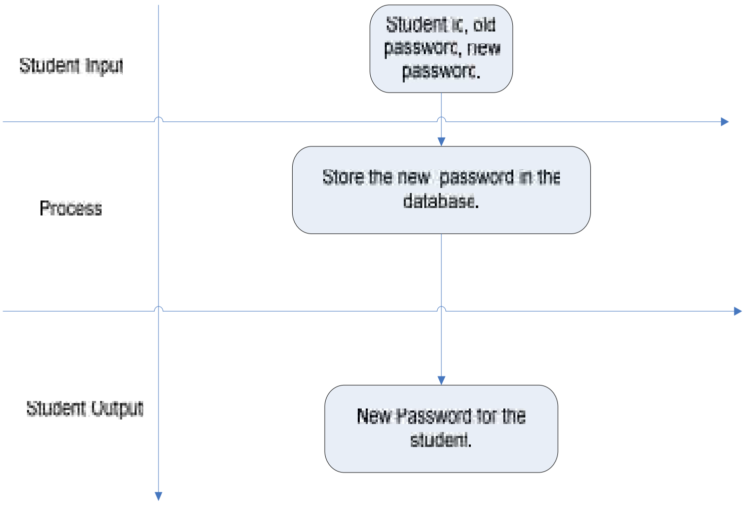
b) Interface:

- ✓ Inputs: student id, old password, new password.
- ✓ Outputs: obtain to the new password.

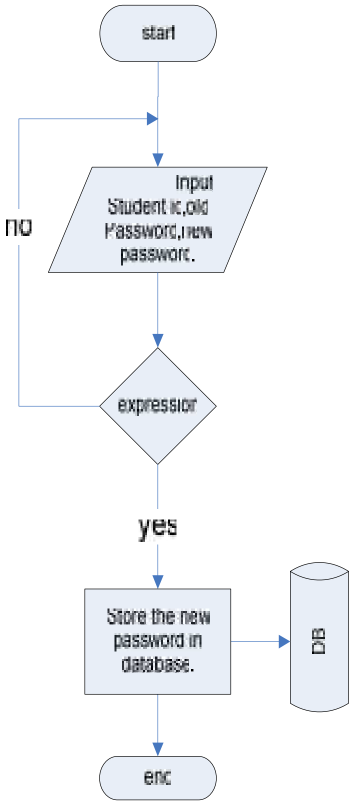
c) Constraints:

- ✓ The new Passwords have to be six characters at least for student.
- ✓ The new Passwords have to not contain any space.
- ✓ No special characters would be allowed within a new password string such as (semicolon, etc).

d) User interface design:


Student Input
Student id, old password, new password.
Process
Store the new password in the database.
Student Output
New Password for the student.

e) Flowchart:


start
Input Student id,old Password,new password.
no
expression
yes
Store the new password in database.
DB
end

6. Forget the password.

a) Description**:** this function enables student to remember his/her password.

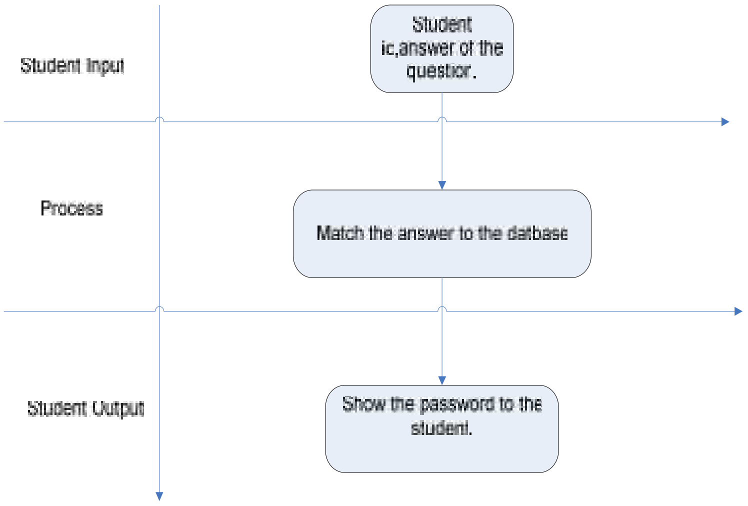
b) Interface:

- ✓ Inputs: student id, the answer of question.
- ✓ Outputs: obtain to the old password.

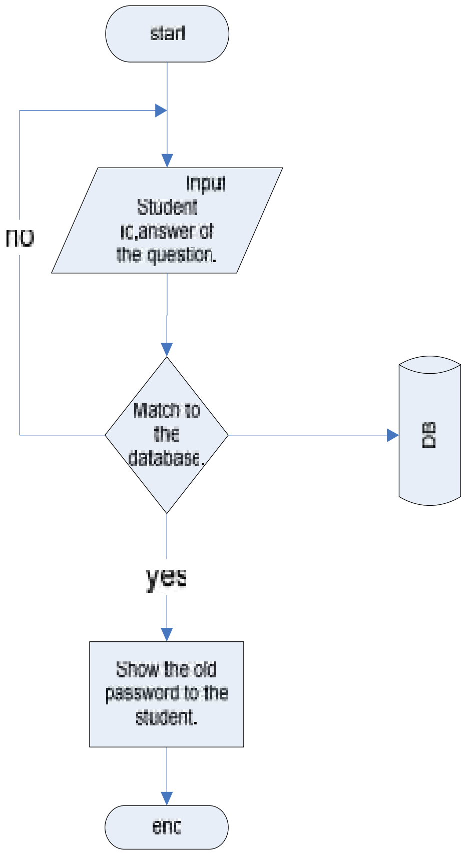
c) Constraints:

- ✓ The answer of the question must match the answer in the database.

d) User interface design:


Student Input
Student ic,answer of the questior.
Process
Match the answer to the datbase
Student Output
Show the password to the student.

e) Flowchart:


start
Input Student id,answer of the question.
no
Match to the database.
DB
yes
Show the old password to the student.
end

7. Process of Registration

a) Description: this function enables every student to select the courses from the list of offers course.

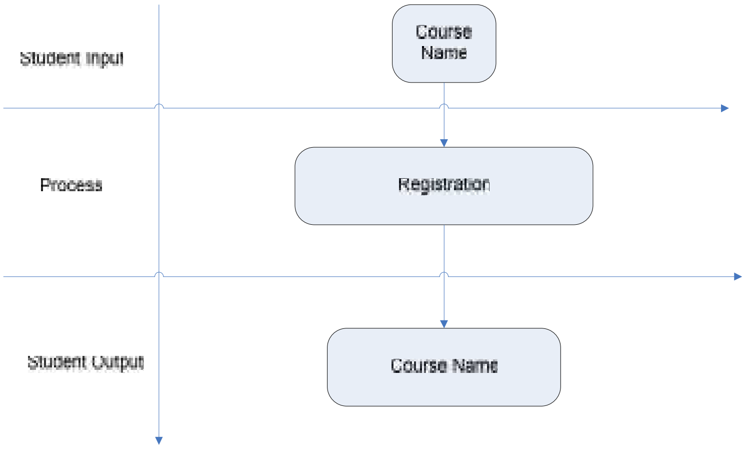
b) Interface:

- ✓ Inputs: course name.
- ✓ Outputs: course name.

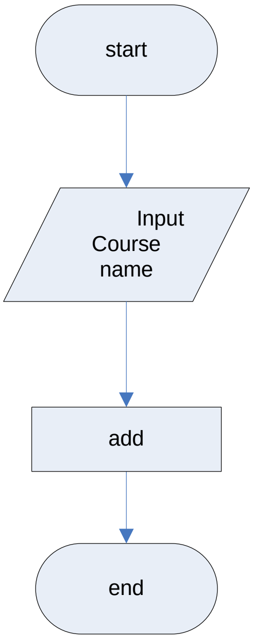
c) Constraints:

- ✓ The student must pay the fee.
- ✓ The student can register only the allowable number of hour depending on his/here academic statues.

d) User interface design:



e) Flowchart:



8. Store the chosen course.

a) Description: this function enables every student to store his/her selected course to register after the submit.

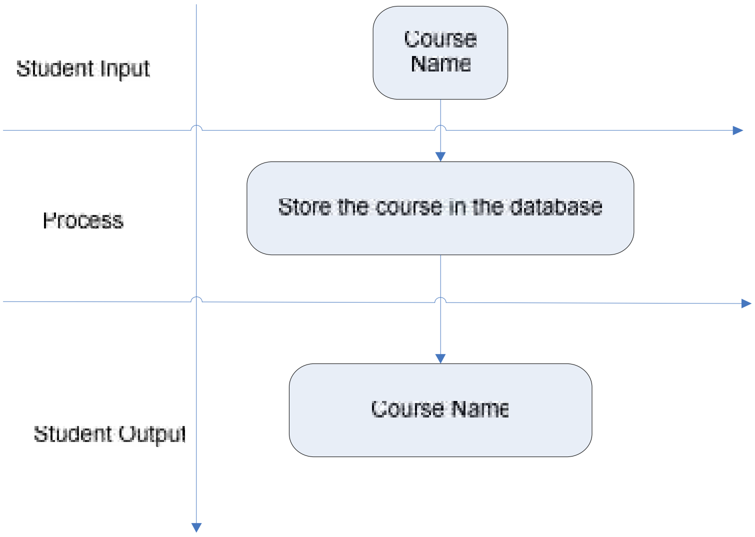
b) Interface:

- ✓ Inputs: course name.
- ✓ Outputs: course name.

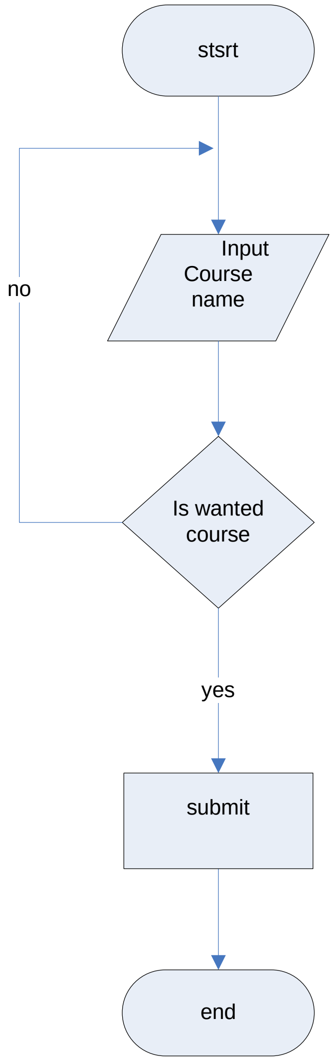
c) Constraints:

- ✓ The add must be done before clicking on the submit button.

d) User interface design:



e) Flowchart:



9. Generate Report

a) Description: The function that generates a report after the process of registration.

b) Interface:

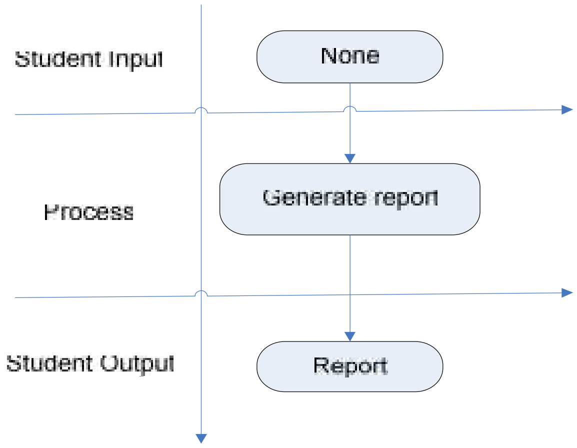
✓ Inputs: complete the process of registration.

✓ Outputs: Report (course name, course number, section, student name, time of each course, and total of hours).

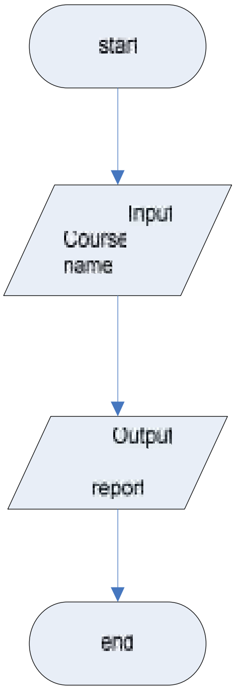
c) Constraints:

✓ None.

d) User interface design:



e) Flowchart:



10. Help

a) Description: This function enables students to find some of information about the system.

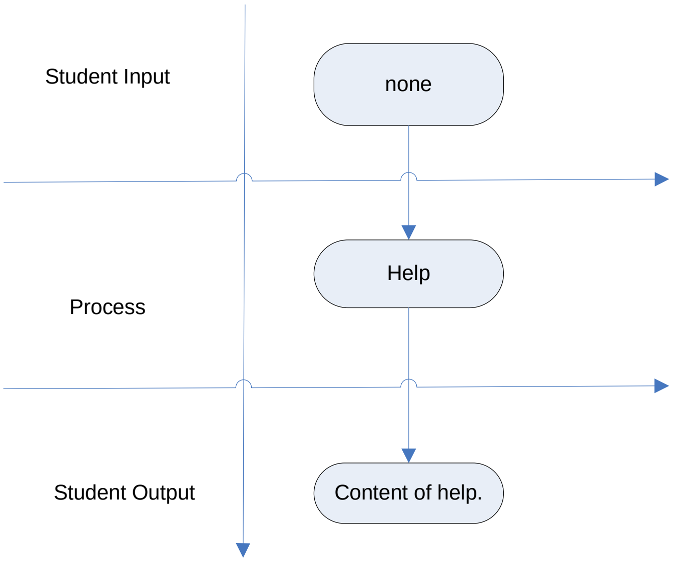
b) Interface:

- ✓ Input: click on help.
- ✓ Output: page contains the step to registration.

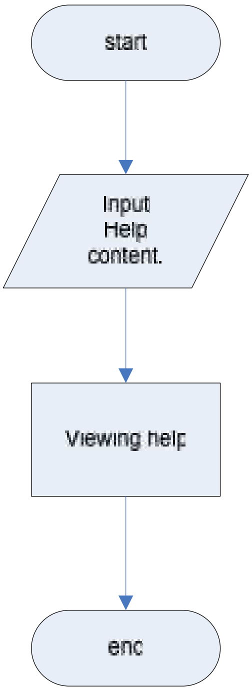
c) Constraints:

- ✓ None.

d) User interface design:



e) Flowchart:



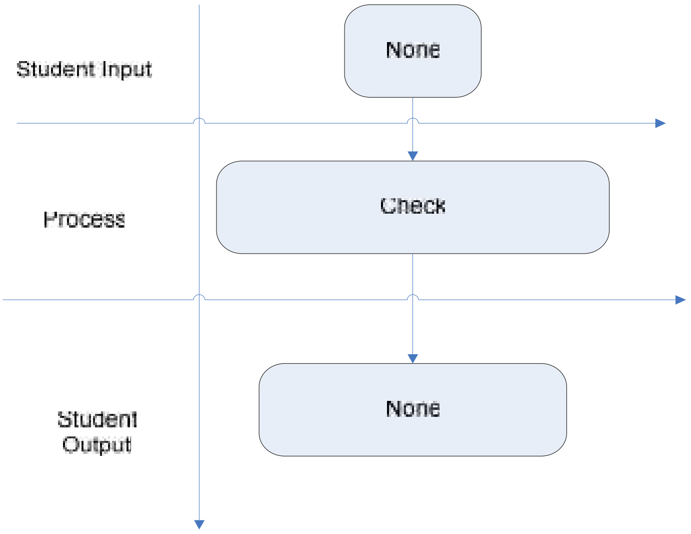
11. Status payment.

   a) Description: This function enables students to login to the registration process only after the payment process.

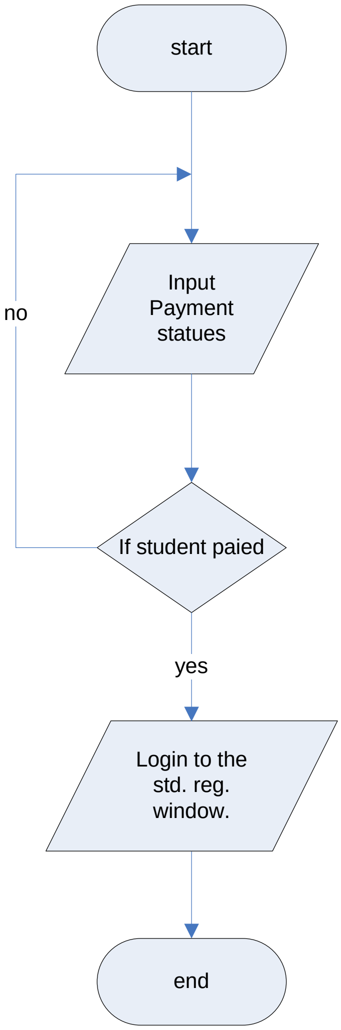
   b) Interface:
      - ✓ Input: A check made by the finance manager.
      - ✓ Output: login to the window of student registration.

   c) Constraints:
      - ✓ The student must pay in the bank.

   d) User interface design:



e) Flowchart:


start
Input Payment statues
no
If student paied
yes
Login to the std. reg. window.
end

12. The number of the hours that the student can register in the semester.

(12-1) the student can register 12 hours only as maximum.

a) Description: This function enables students to register only 12 hours if he/she has an academic warning.

b) Interface:

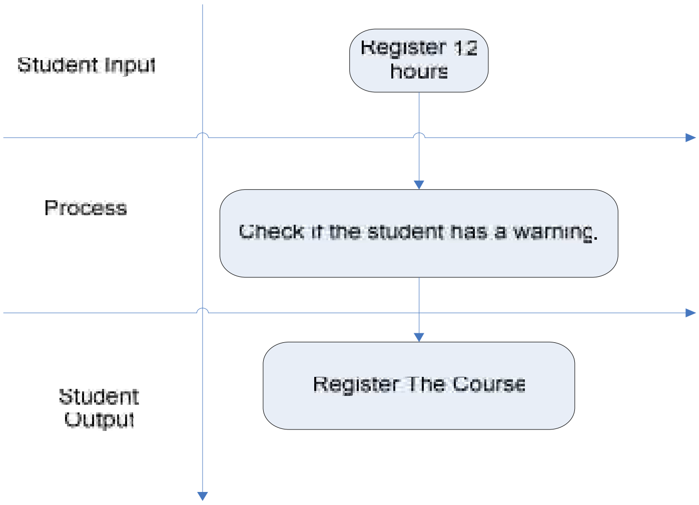
✓ Input: course name.

✓ Output: course name.

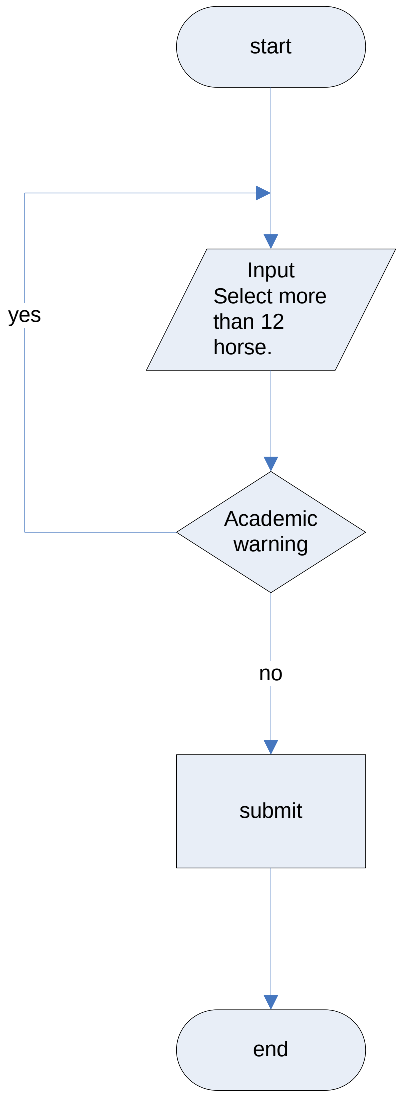
c) Constraints:

✓ The student must register only 12 hours as maximum if he/she has an academic warning.

d) User interface design:



e) Flowchart:


start
yes
Input
Select more
than 12
horse.
Academic
warning
no
submit
end

(12_2) the student can register between 21hours and 0 hours if he/she graduates.

a) Description: This function enables students to register between 21hours and 0 hours if he/she graduates in that semester.

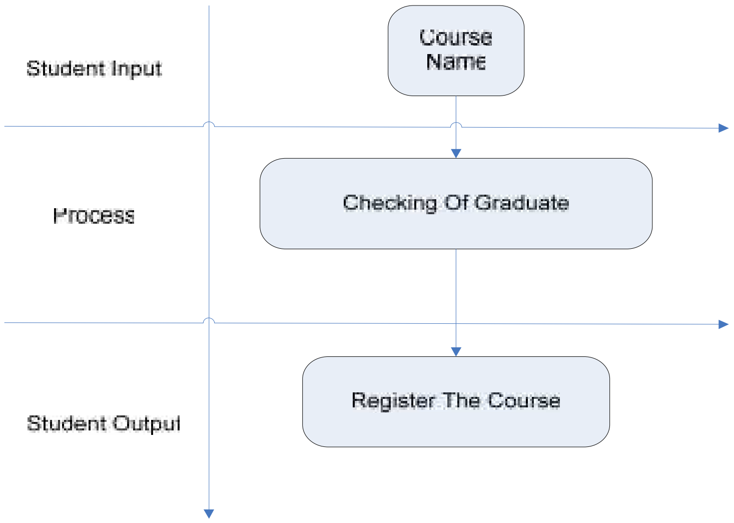
b) Interface:

- ✓ Input: course name.
- ✓ Output: course name.

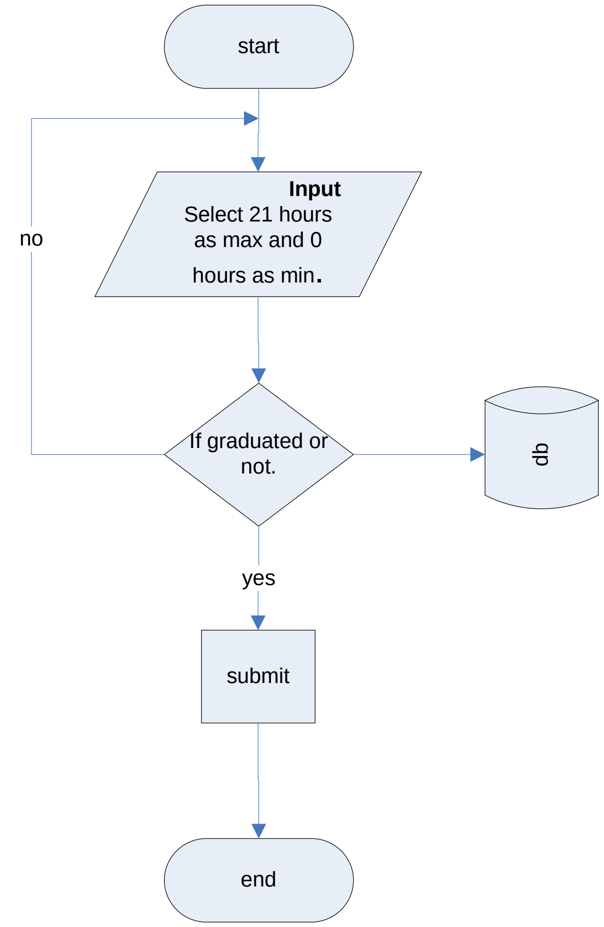
c) Constraints:

- ✓ The student can register 21 hours if he/she graduates in that semester.

d) User interface design:



e) Flowchart:


start
Input
Select 21 hours
as max and 0
hours as min.
no
If graduated or
not.
db
yes
submit
end

(12-3) if the student in the normal state (no warning and not graduated) he/she can register (18) hours as maximum and (12) hours as maximum.

a) Description: This function enables students to register (18) hours as maximum and (12 hours as minimum) in the normal stat.

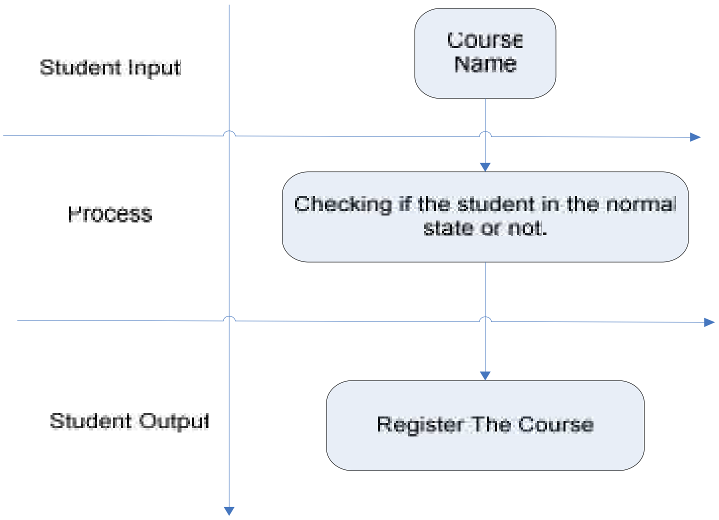
b) Interface:

- ✓ Input: select number of course which the total of hours between 18 and 12.
- ✓ Output: register the course.

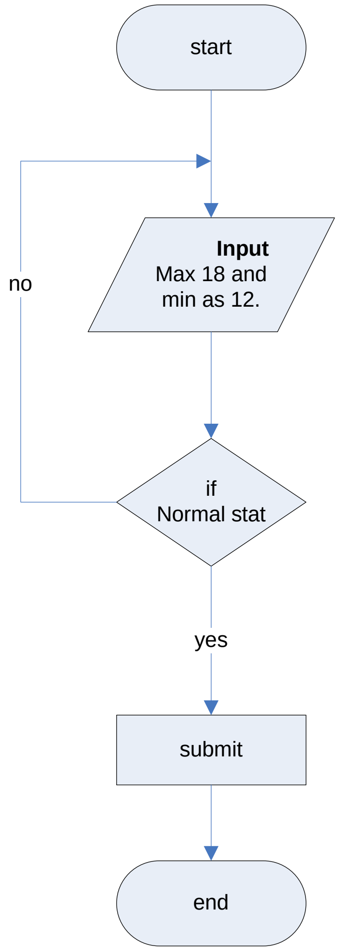
c) Constraints:

- ✓ No warning and not graduated.

d) User interface design:



e) Flowchart:


start
no
Input
Max 18 and
min as 12.
if
Normal stat
yes
submit
end

13. The student can not register any course that he takes it in a previous time and his mark in it is equal or more than 70%.

a) Description: This function disables student to register any course that he takes it in a previous time and his mark in it is equal or more than 70%.

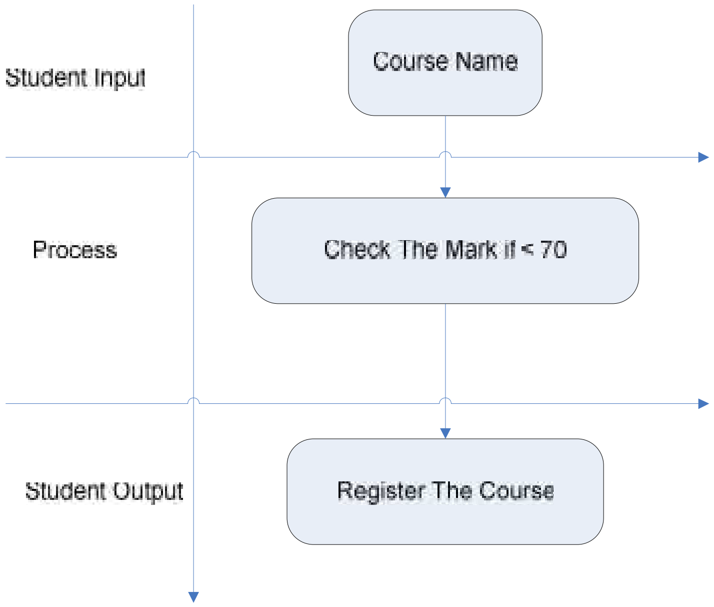
b) Interface:

- ✓ Input: course name
- ✓ Output: course name.

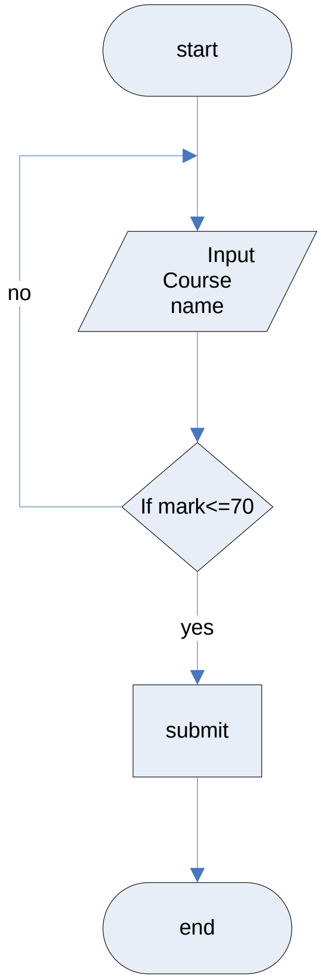
d) Constraints:

- ✓ The mark of the course must be less than 70.

d) User interface design:



e) Flowchart:



14. Academic statues

a) Description: This function provides the student with info about his academic statues.

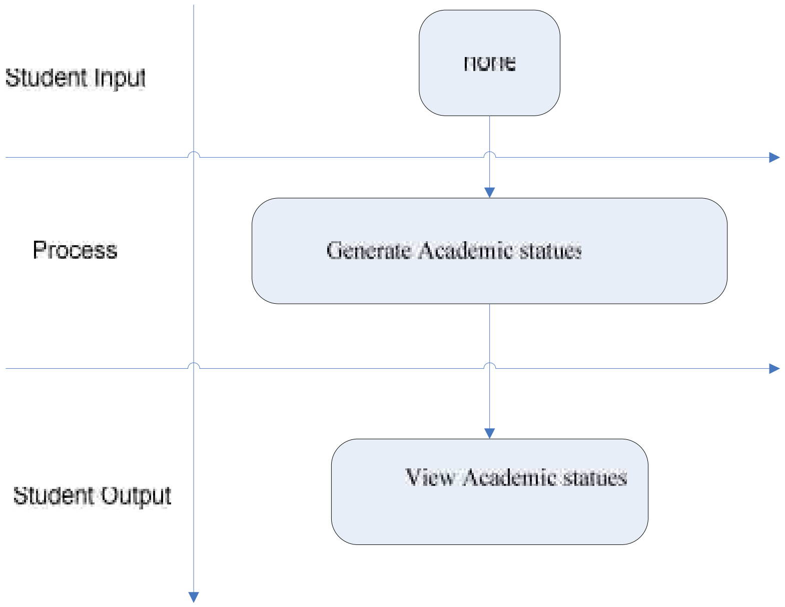
b) Interface:

- ✓ Input: none.
- ✓ Output: the form of academic statues.

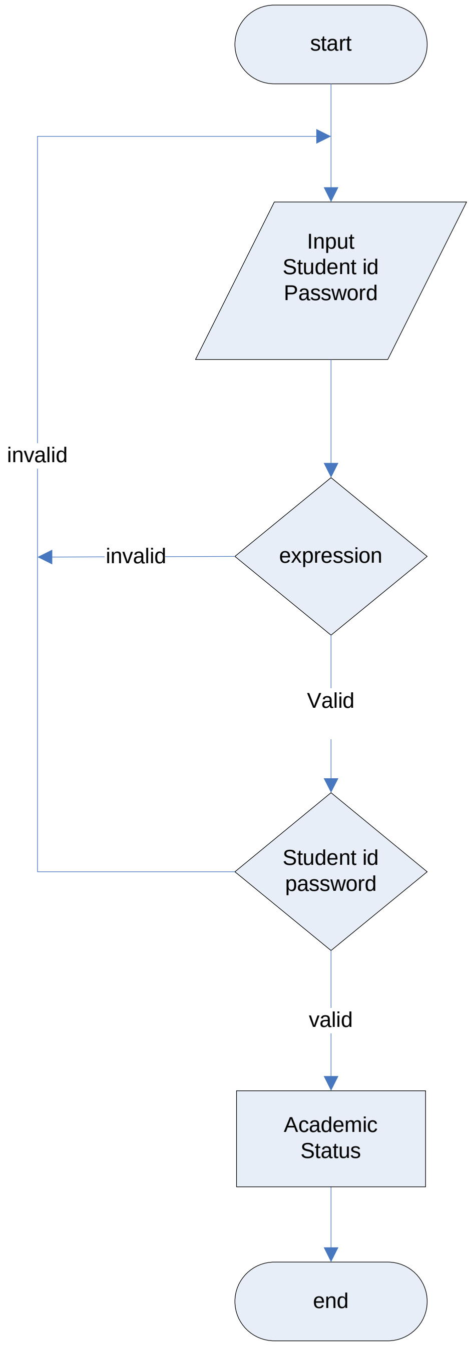
c) Constraints:

- ✓ Student must previously be registered on the university at least for one semester.

d) User interface design:



e) Flowchart:


start
Input
Student id
Password
invalid
invalid
expression
Valid
Student id
password
valid
Academic
Status
end

15. Offer courses

a) Description: This function provides the student with info about the courses that available in that semester.

b) Interface:

✓ Input: none.

✓ Output: the form of Offer courses.

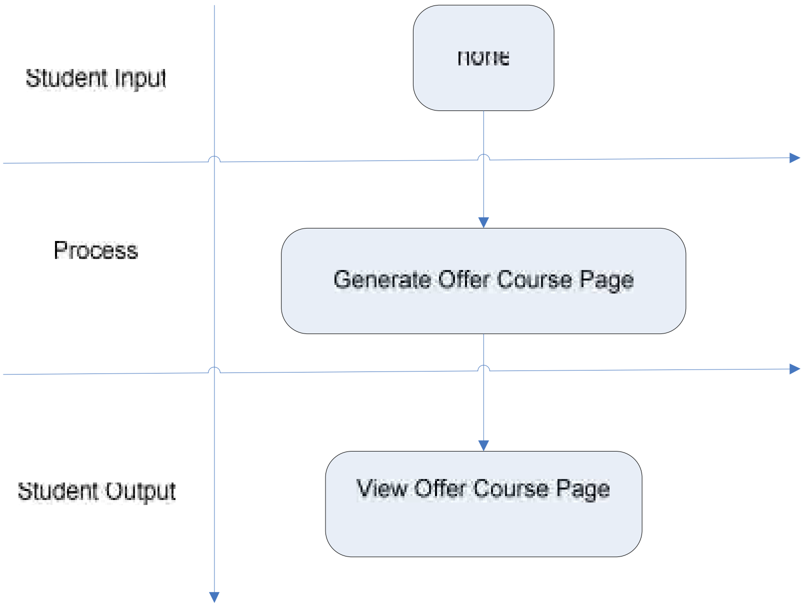
c) Constraints:

✓ Student must previously be registered on the university at least for one semester.

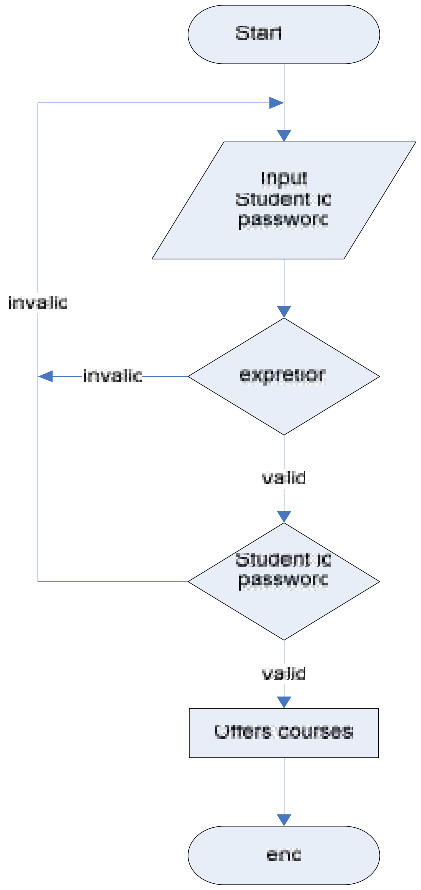
✓ Passwords have to be six characters at least for student.

✓ The student has the right student id and password

d) User interface design:



e) Flowchart:


Start
Input
Student id
password
invalid
invalid
expretion
valid
Student id
password
valid
Offers courses
end

16. Student loan

a) Description: This function provides the student with information about his/her the value

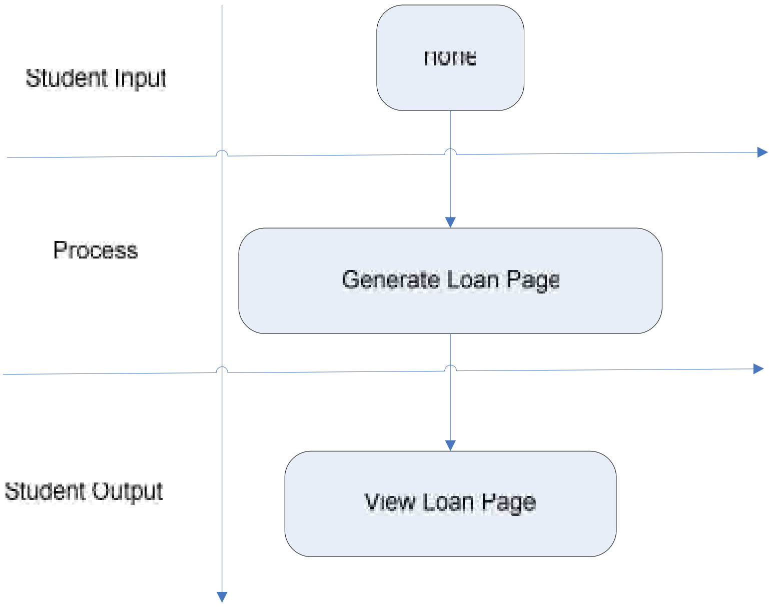
b) Interface:

- ✓ Input: none
- ✓ Output: the form of student loan.

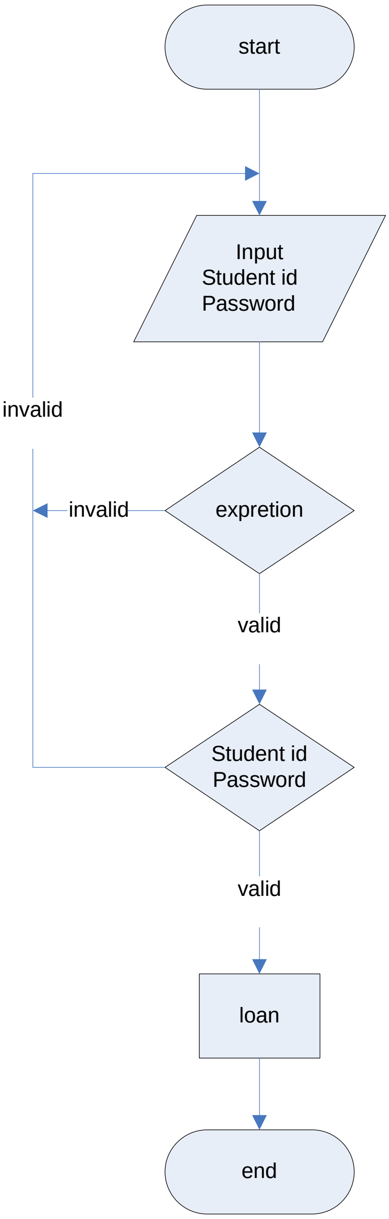
c) Constraints:

- ✓ Student must previously be registered on the university at least for one semester.
- ✓ Passwords have to be six characters at least for student.
- ✓ The student has the right student id and password

d) User interface design:



e) Flowchart:


start
Input
Student id
Password
invalid
invalid
expretion
valid
Student id
Password
valid
loan
end

17. Course plane

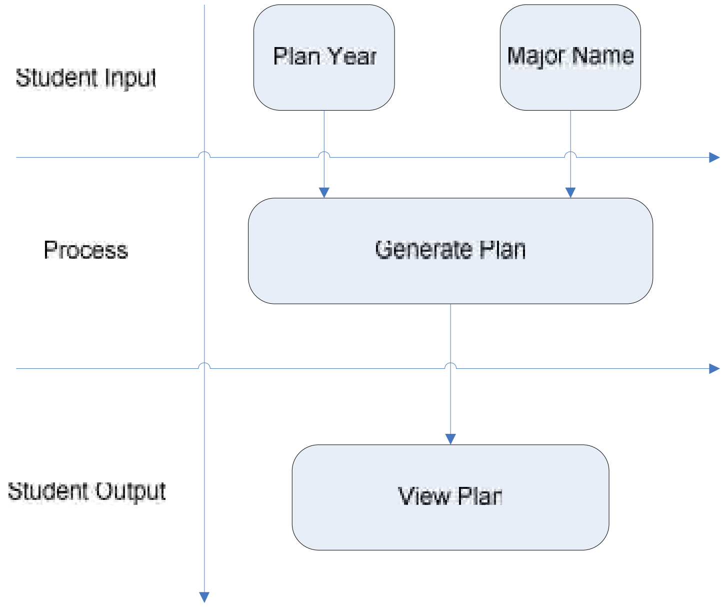
a) Description: This function enable the student to see plan that he belongs to.

b) Interface:

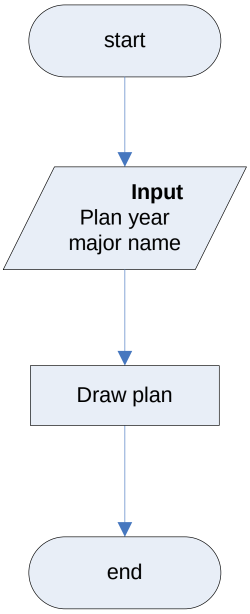
- ✓ Input: plan year, major name.
- ✓ Output: the form contains the draw plan.

c) Constraints: none.

d) User interface design:



e) Flowchart:



18. the dependences of the courses

(18-1) succeed previous course

a) Description: this function disables every student to update his/her selected course to register with out taken the succeed previous course.

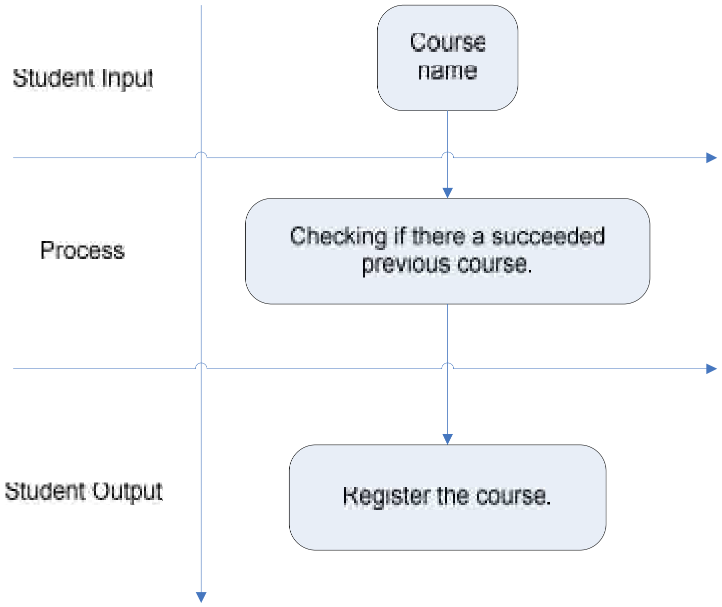
b) Interface:

- ✓ Inputs: course name.
- ✓ Outputs: course name.

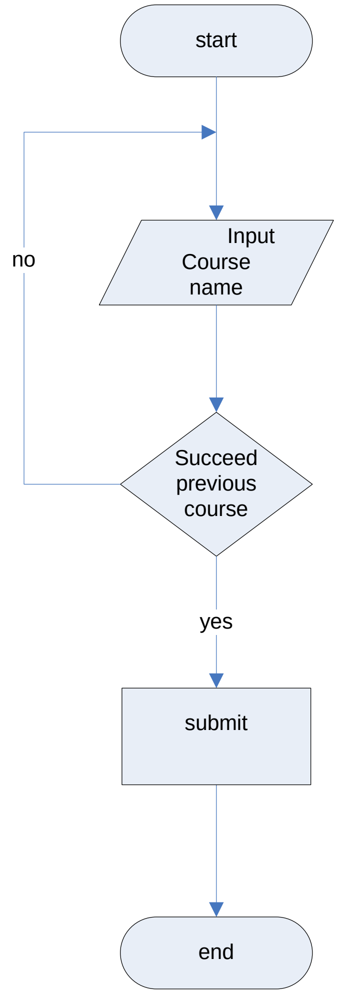
c) Constraints:

- ✓ The student must taken the previous course.

d) User interface design:



e) Flowchart


start
no
Input
Course
name
Succeed
previous
course
yes
submit
end

(18-2) previous course with out succeed

a) Description: this function enables every student to update his/her selected course to register with out succeed.

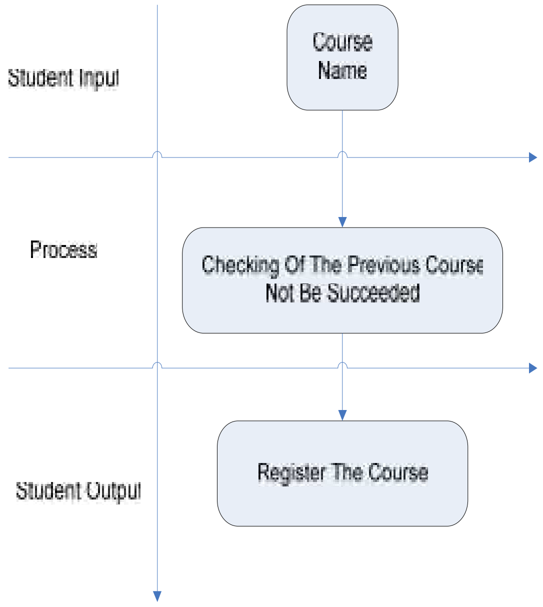
b) Interface:

- ✓ Inputs: course name.
- ✓ Outputs: course name.

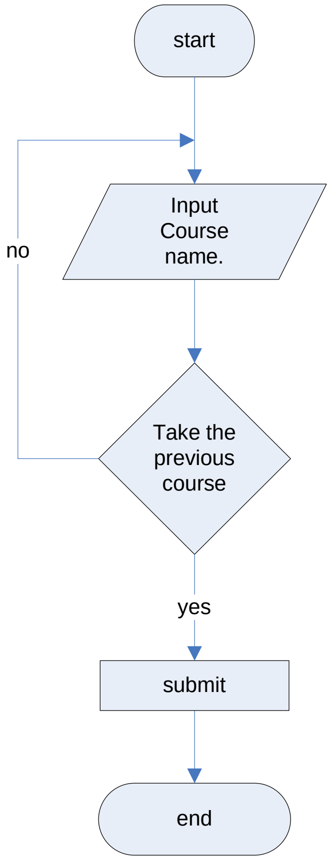
c) Constraints:

- ✓ None.

d) User interface design:



e) Flowchart:


start
Input Course name.
no
Take the previous course
yes
submit
end

(18-3) co_requist

a) Description: this function enables every student to update his/her selected course to register in the semester together.

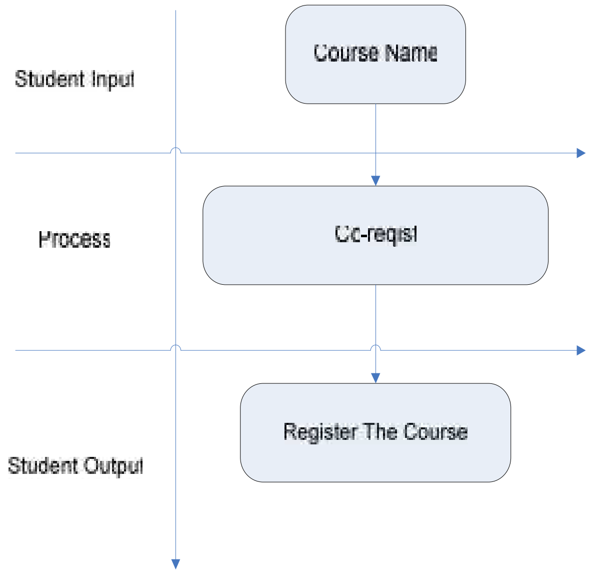
b) Interface:

- ✓ Inputs: course name.
- ✓ Outputs: course name.

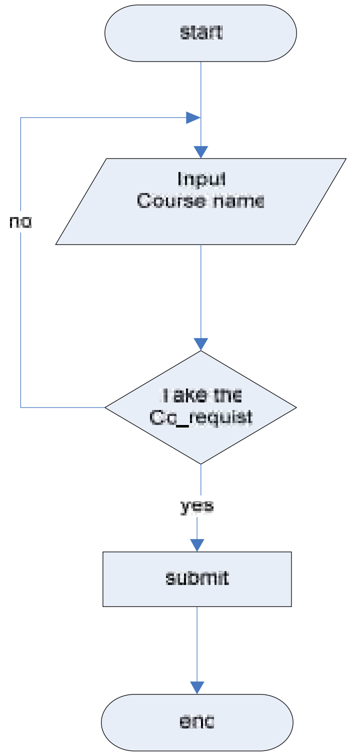
c) Constraints:

- ✓ None.

d) User interface design:



e) Flowchart:


start
Input
Course name
no
Take the
Co_requist
yes
submit
end

19. Conflict in Time.

a) Description: this function disables every student to take any two courses that has a conflict in time.

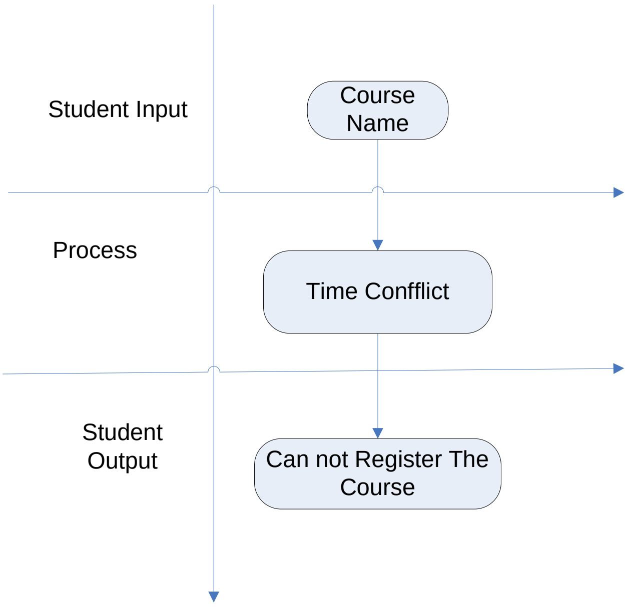
b) Interface:

- ✓ Inputs: course name.
- ✓ Outputs: course name.

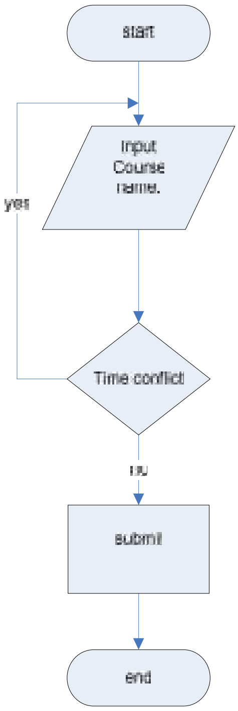
c) Constraints:

- ✓ The course must not conflict in time.

d) User interface design:



e) Flowchart:



20. Conflict in capacity.

a) Description: this function disables student to add his/her selected course to register in the semester, if the section is full.

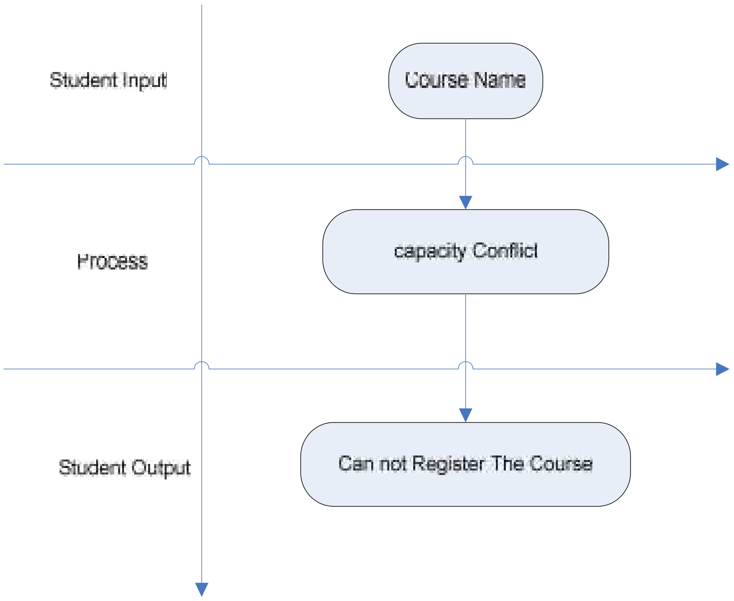
b) Interface:

- ✓ Inputs: course name.
- ✓ Outputs: course name.

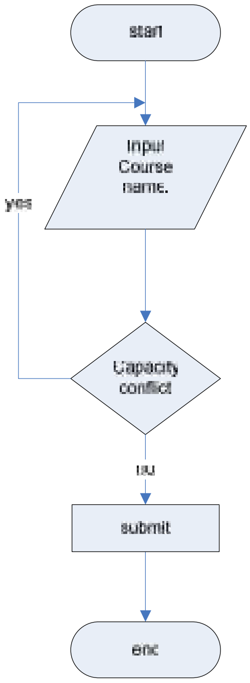
c) Constraints:

- ✓ The course must not conflict in capacity.

d) User interface design:



e) Flowchart:


start
Input Course name.
yes
Capacity conflict
no
submit
end

21. Administrator login:

a) Description: This function will be the only method for administrator to login to system registration database (locally).

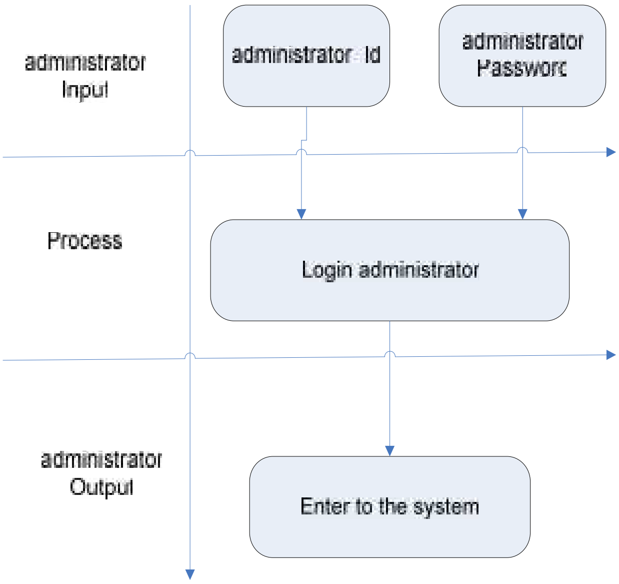
b) Interface:

- ✓ Inputs: administrator id, Password.
- ✓ Outputs: Login to his session.

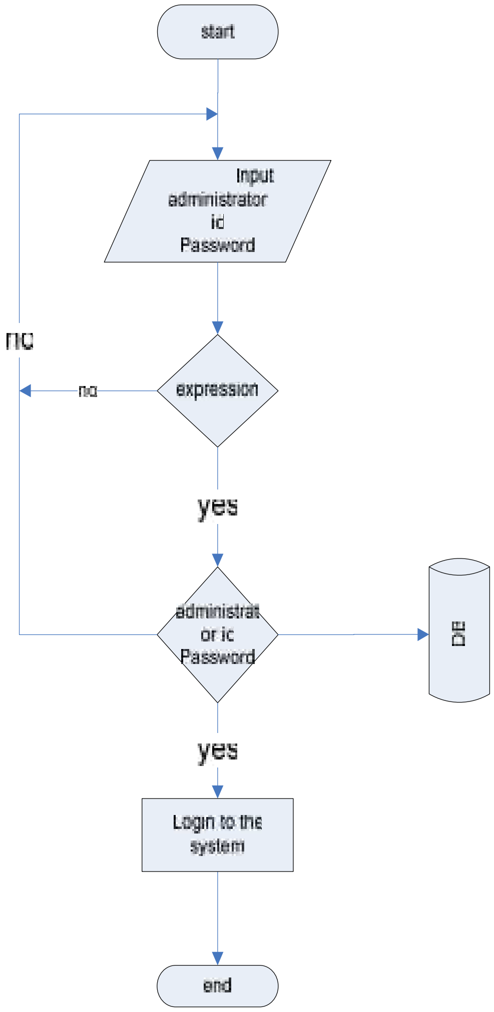
c) Constraints:

- ✓ Student id should have numbers (0-9).
- ✓ The student has the right student id and password.

d) User interface design:



e) Flowchart:


start
Input administrator id Password
no
no
expression
yes
administrator id Password
DB
yes
Login to the system
end

22. **Administrator Logout.**

a) Description**:** this function enables administrator to logout from the system.

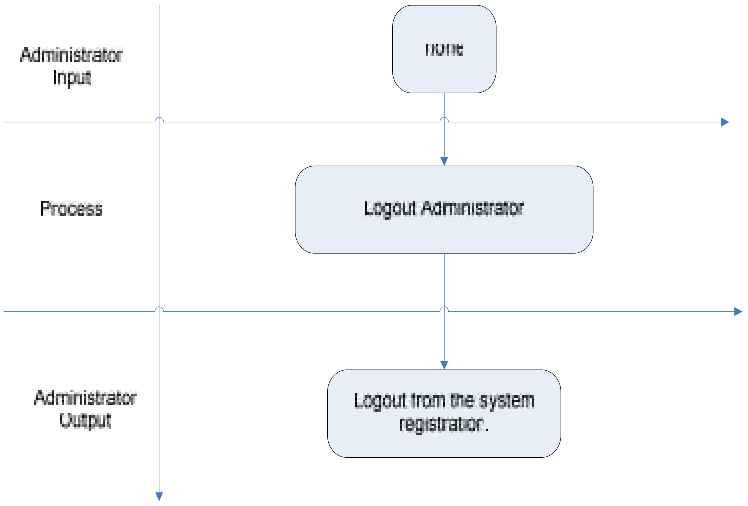
b) Interface:

- ✓ Inputs: select logout.
- ✓ Outputs: logout from the system registration.

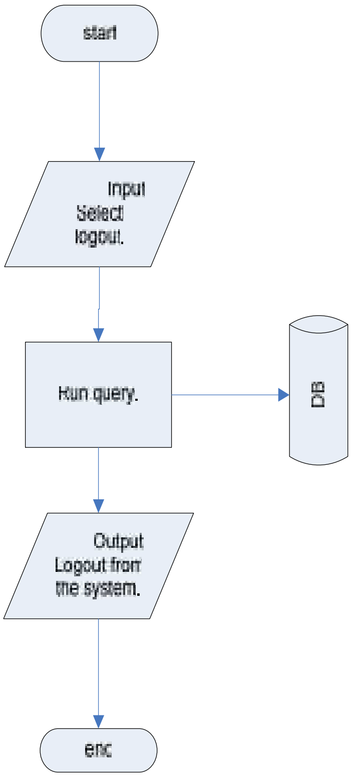
c) Constraints:

- ✓ none.

d) User interface design:



e) Flowchart:


start
Input
Select
logout.
Run query.
DB
Output
Logout from
the system.
end

23. Update on the plan by the Administrator:

a) Description: This function will be allowing the administrator to update on the plan (locally).

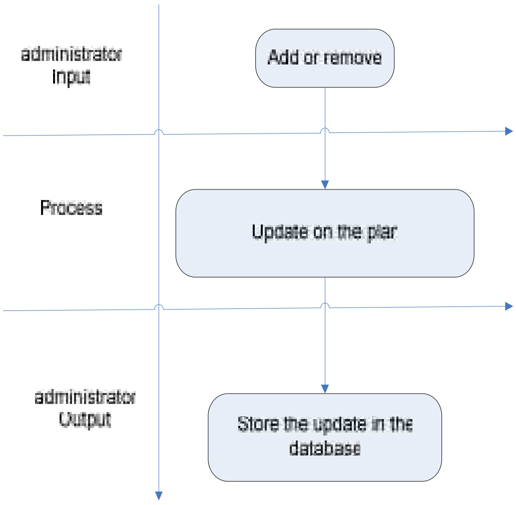
b) Interface:

- ✓ Inputs: exchange on the plan.
- ✓ Outputs: store the exchange in the database.

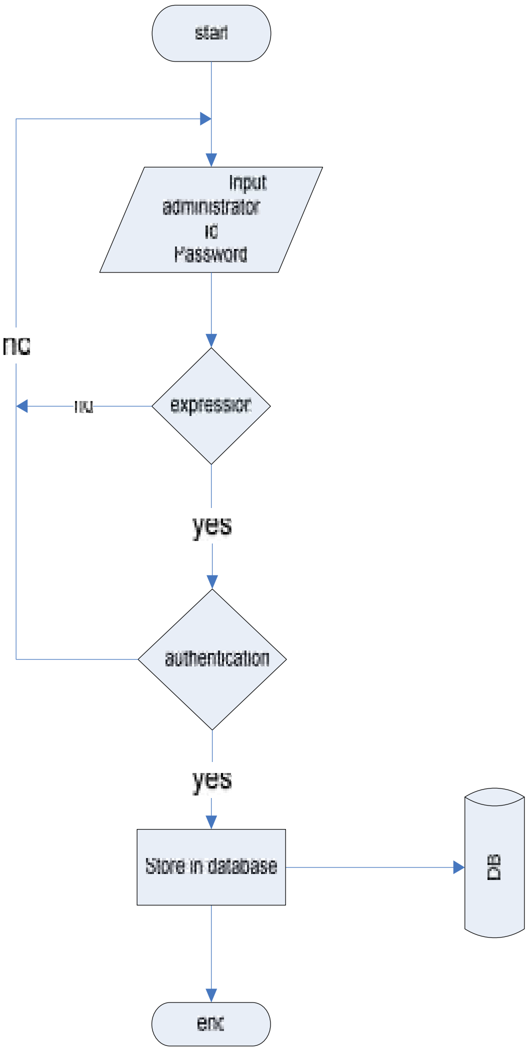
c) Constraints:

- ✓ The exchange must be from the authorization of the administrator .

d) User interface design:



e) Flowchart:


start
Input administrator id Password
no
no
expression
yes
authentication
yes
Store in database
DB
end

4.3 **Input/output design**.

I/O design includes the detailed graphical description of all I/O forms.

4.3.1 Input screen design.

✓ Student authentication: This screen check the authentication of the student to login to the student login home; by entering the student ID and student identity, and then click on the button" submit".

If the entering data is correct the system will allow the student to enter to the home page registration.

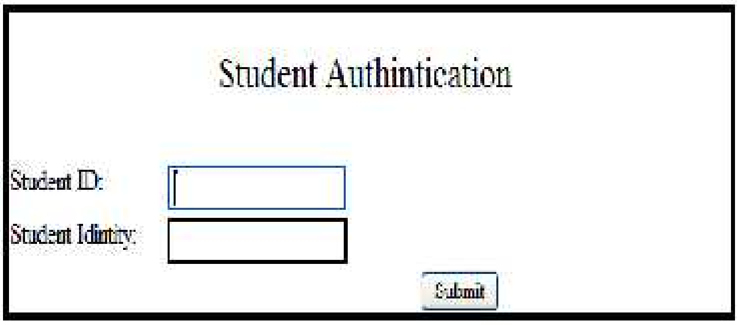


Figure (4.1) Student Authintication form

✓ Student login: In this screen the student enter id and password to login to the home page if he/she has a password or he/she can obtain a password if he/she doesn't have a password, also he/she can remember the password.

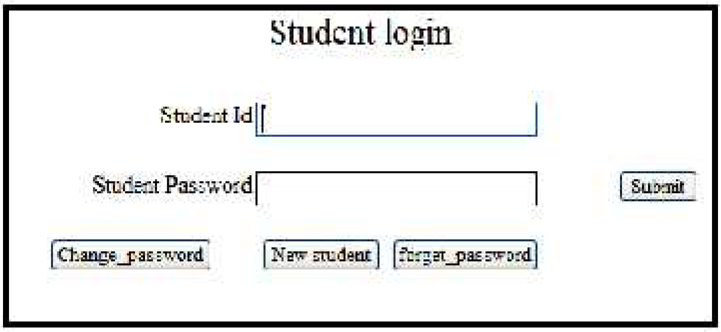


Figure (4.2) Student Login form.

✓ Change password: This screen enable every student to change his/her password by enter the student id, old password, new password, and then confirm the password, if the new password does not match with the validation criteria the system will not allows the student to change it.

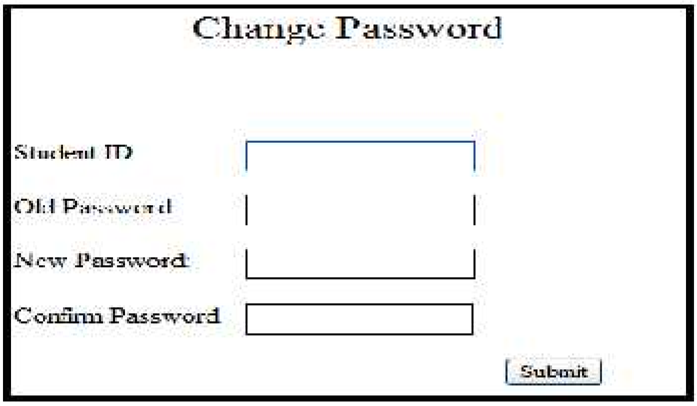


Figure (4.3) Change password form.

✓ Forget password: This screen enable every student to remember his/her password by enter the student id, the answer of the question, then the student can see his/her password as a result after that he/she can return to the student authentication page.

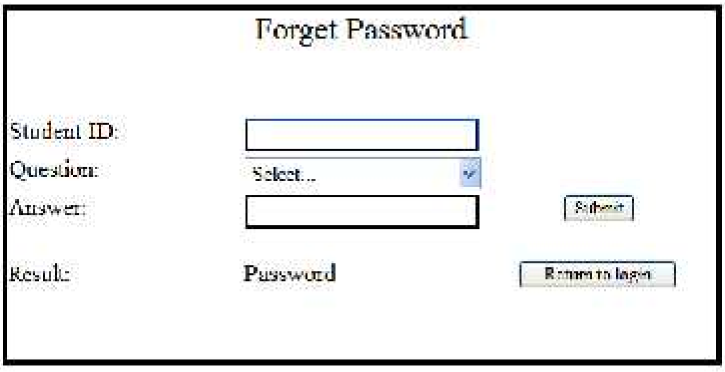


Figure (4.4) Forget password form

✓ New student: This screen enable every student to obtain a password if he/she does not has a password before that, then the student confirm the password and select a question and answer for it.

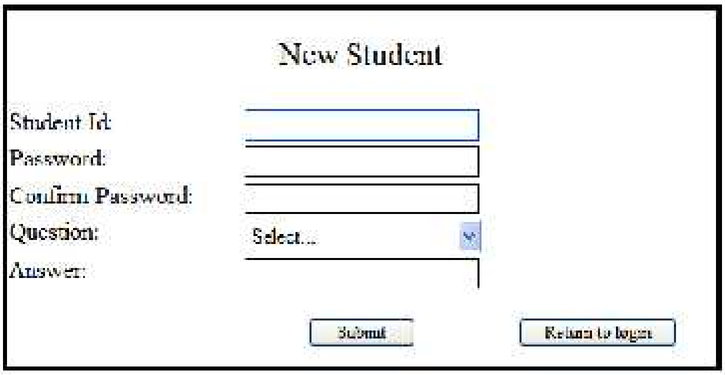


Figure (4.5) New student form

✓ Student registration: In this screen the student can select any course from the dropdown list that contain the offers course then click on (add) to register the course, and click on (delete) to delete the course, but the course doesn't store in the database before click on the submit.

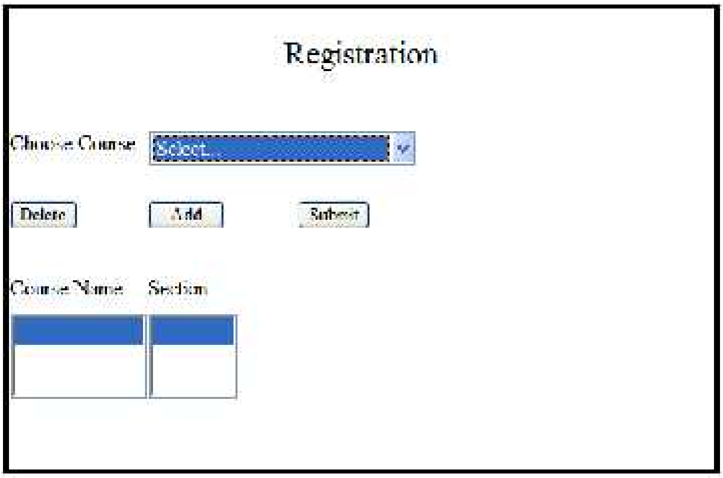
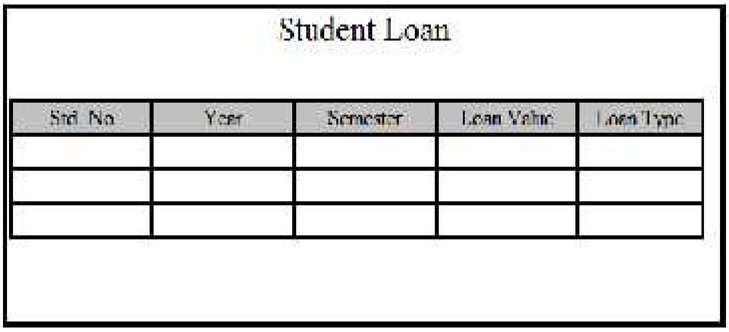


Figure (4.6) registration form

4.3.2 Output screen design.

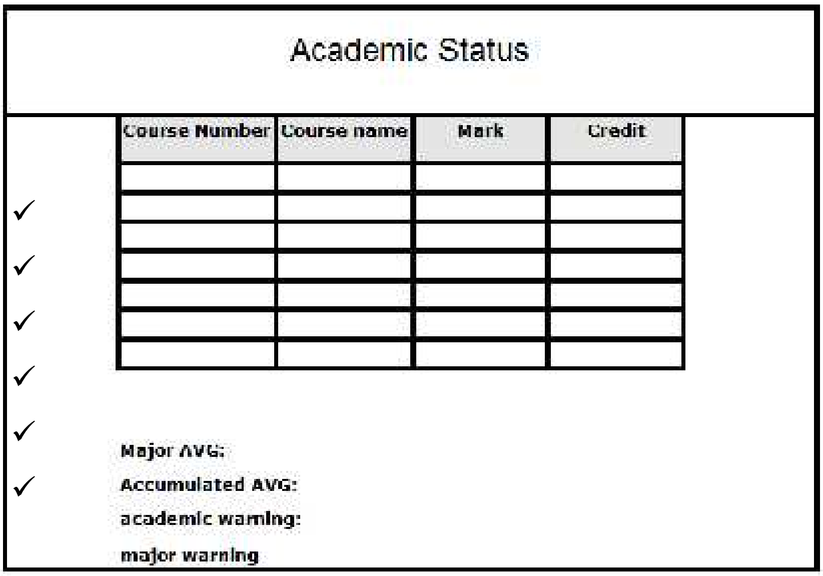
✓ Student loan: in this screen the student can see the value of the loan and its type depending on the student number which entering in the student login.



Student Loan

Std. No	Year	Semester	Loan Value	Loan Type

✓ Academic statues: After the student login to the home page registration and click on the button of academic statues the bellow screen will appear to him, which contain all the courses that the student taken and there mark, course number, student major average, student accumulated average, and if there are academic warning or not.



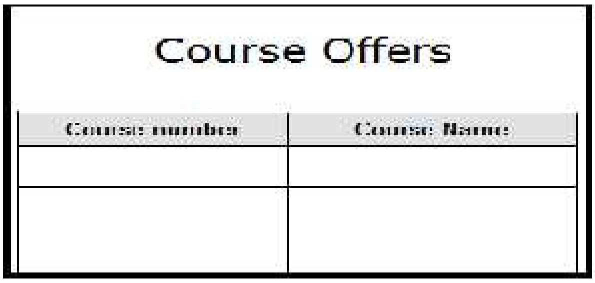
Academic Status

Course Number	Course name	Mark	Credit

✓
✓
✓
✓
✓
✓

Major AVG:
Accumulated AVG:
academic warning:
major warning

- Offers course: This screen appear to the student after click on the button "offer course" from the home page registration, from this page the student can know information about the offers courses by choosing his/her plan year and clicking the button submit.



Course Offers

Course number	Course Name

4.4 Database design

This section includes the database design which is described in the data dictionary, database relations diagram and the UML conceptual model for the “registration” database.

4.4.1 Data dictionary

This part is a description of the database and tables used in the system; it contains the overall description of tables and the specification of each table and its fields.

4.4.1.1 Tables’ description

After analyzing the requirements of the database of the system and depending on security and integrity rules, there was the following database needed for the system to operate on the best way it is expected. The tables of the database are shown in the following table:

Table name	**Description**
student	It contains student number, student name, major number, student level, plan number, student password, ID and year.
Mark	It contains the student number, course number, semester, mark ,year and flag.
Course Info	It contains course number, course name, plan number, credit, practical, course type, and course level
Plans	It contains plan number, plan year, major number, plan picture, required hours.
Major	It contains the major number, major name, and the dept number.
Department	It contains the dept number, dept name, and collage number.
College	It contains the collage number, collage name, start date, and end date.
Schedule	It contains the offer serial, room number, times, and days.
Previous Courses	It contains the course number, previous number, plan number, and previous type.
Students affairs	It contains student number, year, semester, loan value, and loan type.
Course Offer	It contains offer serial, course number, instructor number, semester, section, year, counter, capacity, and major number.
Instructor	It contains instructor number, and instructor name.
Payment	It contains student number, payment status, sum of hours, semester, and year.

student register	It contains student Number, sections, course number, and the semester.
question	It contains question number, and question name.
forget	It contains question number, student number, and answer.

Table (4.1) system tables description

4.4.1.2 Tables

This is a description of each table and its fields:

1. The student table “student”: this table contains student number, student name, student level, student password, major number, plan number and year.

Field Name	Data Type	Null	Key	References	Length	Description
Student No	numeric	No	PK		9	A unique number of the student.
Student Name	Nvarchar	No			50	The student name that is officially registered.
Major Number	Integer	No	FK	Major(major_no)	4	The major number that exists in the major table.
Student Level	Integer	No			4	Brief description about the student level.
ID	decimal	No			9	Determine the student identity.
Student Password	Nvarchar	No			50	The password of the student.
Plan Number	Integer	No			4	The number of the plan that the student take.
Year	Datetime	No			8	The year that the student register in the university.

Table (4.2) the student information table.

2. Mark table “mark”: this table provides the student the ability to take the mark of the course that the student take in the previous semester by enter the plan number which he/she belong to.

Field Name	**Data Type**	**Null**	**Key**	**References**	**Length**	**Description**
Student No	numeric	No	PK,FK	Student(student_no)	9	A unique number of the student.
Course Number	Integer	No	PK		4	A unique number of the course
Semester	Integer	No	PK		4	The number of the semester that the student takes the course.
Year	Datetime	No	PK		8	The year that the student take the course.
Mark	Integer	No			4	The mark of the student in that course.
Flag	Bit	No			1	Must be 0 when the course hours collect in the number of the course that the student takes.

Table (4.3) the mark table.

3. The plans table “plans": this table contains the plan number, major number, plan year, plan picture, and required hours.

Field Name	Data Type	Null	Key	References	Length	Description
Plan _No	Integer	No	PK		4	A unique number of the plan.
Plan_Year	Datetime	No			8	The year which the plan determine.
Major_No	Integer	No	Fk	Major(major_no)	4	The number of the major.
Plan_pic	Nvarchar	No			50	The picture that demonstrate the tree of the courses.
Required_houres	Integer	NO			4	The number of hours in student plan.

Table (4.4) the plans table.

4. The Course_info table “course_info”: this table contains plan number, course number, course name credit, practical, course type and the course level.

Field Name	Data Type	Null	Key	References	Length	Description
Plan Number	Integer	No	PK,FK	Plans(plan_no)	4	A unique number of the plan.
Course Number	Integer	No	PK		4	A unique number of the course
Course Name	Nvarchar	No			50	The name of the course.
Credit	Integer	No			4	Determine the number hours for each course.
Practical	Integer	No			4	Determine if the course has a practical or not.
Course Type	Nvarchar	No			50	Determine the type of the course.
Course Level	Integer	No			4	The level that the course is taken in.

Table (4.5)the course info table.

5. The Major table “major”: this table contains the major number, major name, and the department number.

Field Name	Data Type	Null	Key	References	Length	Description
Major Number	Integer	No	PK		4	A unique number of the major.
Major Name	Nvarchar	No			50	The name of the major.
Dept Number	Integer	No	FK	Department(dept_no)	4	The number of the department.

Table (4.6) the major table.

6. The Department table: this table contains the department number, department name, Collage Number.

Field Name	Data Type	Null	Key	References	Length	Description
Dept Number	Integer	No	PK		4	A unique number for each dept.
Dept Name	Nvarchar	No			50	The name of the department.
Collage Number	Integer	No	FK	Collage(collage_no)	4	A unique number for each collage.

Table (4.7) the department table.

7. The college table “college”: this table contains the college number, college name, start date, and end date.

Field Name	**Data Type**	**Null**	**Key**	**References**	**Length**	**Description**
College Number	Integer	No	PK		4	The unique number of the college
College Name	Nvarchar	No			50	The unique name of the college
Start date	datetime	No			8	The date when the college start the registration.
End date	datetime	No			8	The date when the college end the registration.

Table (4.8) the college table.

8. The Schedule table” Schedule”: this table contains the Offer Serial, Room Number, Times and the days.

Field Name	Data Type	Null	Key	References	Length	Description
Offer Serial	Integer	No	PK,FK	Course_offers (Offer_Serial)	4	Use as index of the table.
Room Number	Integer	No			4	The number for each room.
Times	Datetime	No			8	Determine the time that the course will be taken in it.
days	Datetime	No				Determine the day that the course will be taken in it.

Table (4.9) the Schedule table.

9. The Previous Curse table" Previous Curse ": this table contains the Course Number, Previous Number, Previous type and the plan number.

Field Name	Data Type	Null	Key	References	Length	Description
Course Number	Integer	No	PK		4	A unique number for each course.
Previous Number	Integer	No	PK		4	A unique number for each previous.
Plan Number	Integer	No	PK,FK	Plans(plan_no)	4	A unique number for each plan.
Previous type	Integer	No			4	The type of the previous course.

Table (4.10) the Previous Curse table.

10. The Student Affairs table “student affairs”: this table contains student number, year, semester number, loan value and the loan type.

Field Name	Data Type	Null	Key	References	Length	Description
Student Number	Numeric	No	PK,FK	Student(student_no)	9	A unique number for each student.
Year	Datetime	No			8	Determine the Year that the students take the loan.
Semester	Integer	No			4	Determine the number of the semester that the student take the loan.
Loan Value	Nvarchar	No			50	Determine the value of the loan.
loan type	Nvarchar	No			50	Determine the type of the loan.

Table (4.11) the student affairs table.

11. The Course Offers table “course offers”: this table contains offer serial, course number, instructor number, semester, section, capacity, counter, major number and year.

Field Name	Data Type	Null	Key	References	Length	Description
Offer Serial	Integer	No	PK,FK	Schedual(Offer Serial)	4	Use as index of the table.
Course Number	Integer	No			4	The number of the course.
Instructor Number	Integer	No	FK	Instructor(Instructor_no)	4	The number of the inst.
Semester	Integer	No			4	Describe the number of the semester that the course offers in it.
Section	Integer	No			4	Describe the number of the Section.
Capacity	Integer	No			4	The maximum number of student in this section.
Counter	Integer	Yes			4	Count the number of student that register in this section.
Major number	Integer	No	FK	Major(Major_no)	4	The number of the major.
Year	Datetime	No			8	Determine the Year that the curse offers in it.

Table (4.12) the course offers table.

12. The Instructor table” Instructor”: this table contains Instructor Number and the Instructor Name.

Field Name	Data Type	Null	Key	References	Length	Description
Instructor Number	Integer	No	PK		4	A unique number for the instructor
Instructor Name	Nvarchar	No			50	Contain the instructor name.

Table (4.13) the Instructor table.

13. The Payment table” Payment”: this table contains Student Number, semester, payment status, sum of hours and year.

Field Name	Data Type	Null	Key	References	Length	Description
Student Number	Integer	No	PK,FK	Student(student_no)	4	A unique number for the student.
Semester	Integer	No	PK		4	The number of the semester that the student paid.
Payment Statues	Integer	No			4	Determine if the student paid or not.
Sum Of Hours	Integer	No			4	Determine the sum of hours that the student paid.
year	datetime	No			8	The year that the student paid in it.

Table (4.14) the Payment table.

14. The student register table" student register": this table contains student Number, course number, and the semester.

Field Name	**Data Type**	**Null**	**Key**	**References**	**Length**	**Description**
Student number	Numeric	No	PK,FK	Student(student_no)	9	A unique number for the student.
sections	Integer	No			4	Describe the number of the Section.
Course number	Integer	No	PK		4	A unique number for the course.
Semester	Integer	No	PK		4	Determine the semester that the student register the course.

Table (4.15) the student register table.

15. The question table" question": this table contains question number, and question name.

Field Name	**Data Type**	**Null**	**Key**	**References**	**Length**	**Description**
Question number	Integer	No	PK		4	A unique number for the question.
Question name	nvarchar	No			50	The question.

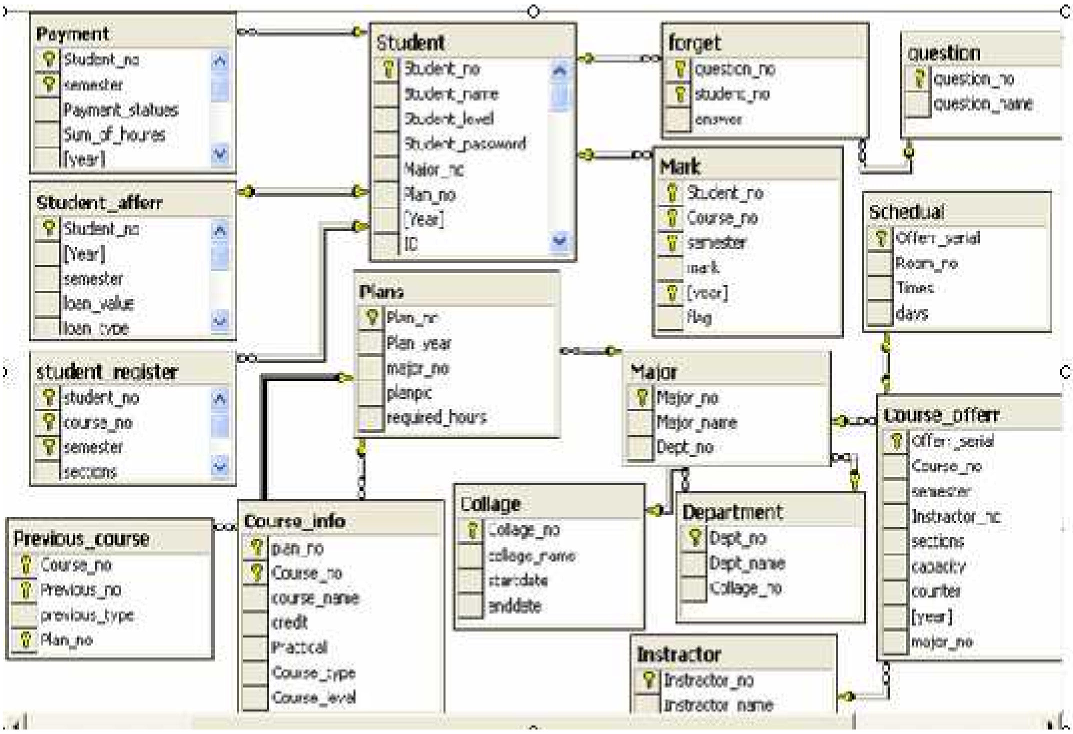
Table (4.16) the question table.

16. The forget table” forget”: this table contains question number, student number and answer.

Field Name	Data Type	Null	Key	References	Length	Description
Question number	Integer	No	PK,FK	Question(question_no)	4	A unique number for the question.
student number	numeric	No	PK,FK	Student(student_no)	9	A unique number for the student.
answer	nvarchar	No			50	The answer of the question.

Table (4.17) the forget table.

4.4.2 Database model



4.5 Test Plan

In test plan we will describe the methodology that we have adapted to test the system.

The following steps describe the process of the system testing:

Testing steps:

1- Unit and Module testing:

In this unit we will use the unit testing to ensure that each function will operate exactly as expected.

2- Integration testing:

The integration of all unit system will be tested so that to ensure that the subsystems work together properly exactly.

3- System testing:

In this unit the system will be tested to ensure that all functions will operate exactly, meets its specification, and that all errors were detected.

time Process	1st week	2nd week	3rd week	4th week	5th week
Unit and module testing.	▨	▨			
Integration testing.		▨	▨	▨	
Stress testing.			▨	▨	
Acceptance testing.				▨	▨

4.6 Programming Language and Coding

There are many languages that can be used to develop a system such ours, but the most effective languages is ASP.NET using Microsoft Visual Studio .NET as the development tool, here we describe the advantages of it, and why our selection was on it:

1- Device Independent:

The ASP.Net is a device independent languages, this means that any user can open the page that written using this languages from any explorer without any additional components or drivers.

2- Server side

One of the most powerful advantages of the ASP.NET technology is that it does not need to make any efforts on the client side, all operations and functions will work on the server.

3- Security

The ASP.Net has a high level of security during transmitting data; they provide us with many algorithms and techniques.

In the ASP.NET there are a build in valuators that ensure the user's input before any generation on the server, so if there any unusual input the webpage it self will not return to the server.

5.1 Introduction

This system is an internet application that depends on a number of technologies that need to be installed maintained and updated continuously. There exist a large number of software development packages that belong to different companies such as Microsoft.

This system is built basically on a group of Microsoft technologies such as the MS SQL server 2003 and ASP.NET 2003. We also included some other applications that serve for the user interface design such as Macromedia Flash and InfoPath.

This chapter describes the packages of software and hardware used in the system for the operation phase and how they are installed and prepared for work, the database and web interfacing, and the user interface implementation.

5.2 Setting up the Required Software and Hardware

5.2.1 Setting up the Hardware

The hardware equipments needed for the operation of this system are:

- A Server characterized computer.
- An internet connection (Leased line).
- A Router for the system network management.
- Networking equipment such as cabling, network adapters, and hubs.
- The development PC is kept also for the updating and development of system.

For more details see (2.8.1.1 Hardware requirements).

5.2.2 Setting up the Software for the system:

5.2.2.1 **Operating System (OS):**

The operating system required for the server set is Microsoft windows 2003 server, and on the development PC we need Microsoft windows XP professional. Windows 2003 server is a powerful networking operating system that supports the server hardware equipment efficiently and supports the ASP.NET environment applications.

5.2.2.2 **.NET framework components:**

To run a system developed using Microsoft .NET technology tools, we need the .NET technology components to be available and integrated with the platform substrate, so when running windows 2003 application services such as component services, message queuing, internet information services(IIS), and windows management instrumentation(WMI), are available to the developer.

The .NET Framework class library exposes features of the runtime and simplifies the development of the .NET based applications; the issue here is that these features should be available for the runtime which must be provided to make the system work probably.

5.2.2.3 Microsoft SQL server 2000:

- **Installation:**

  An Enterprise edition of the Microsoft SQL server 2000 meets our system requirement in creating, accessing, and managing the system database. So Microsoft SQL server 2000 is to be installed on the computer running the system (system computer and the Bank server).

- **Configuration of SQL server 2000:**

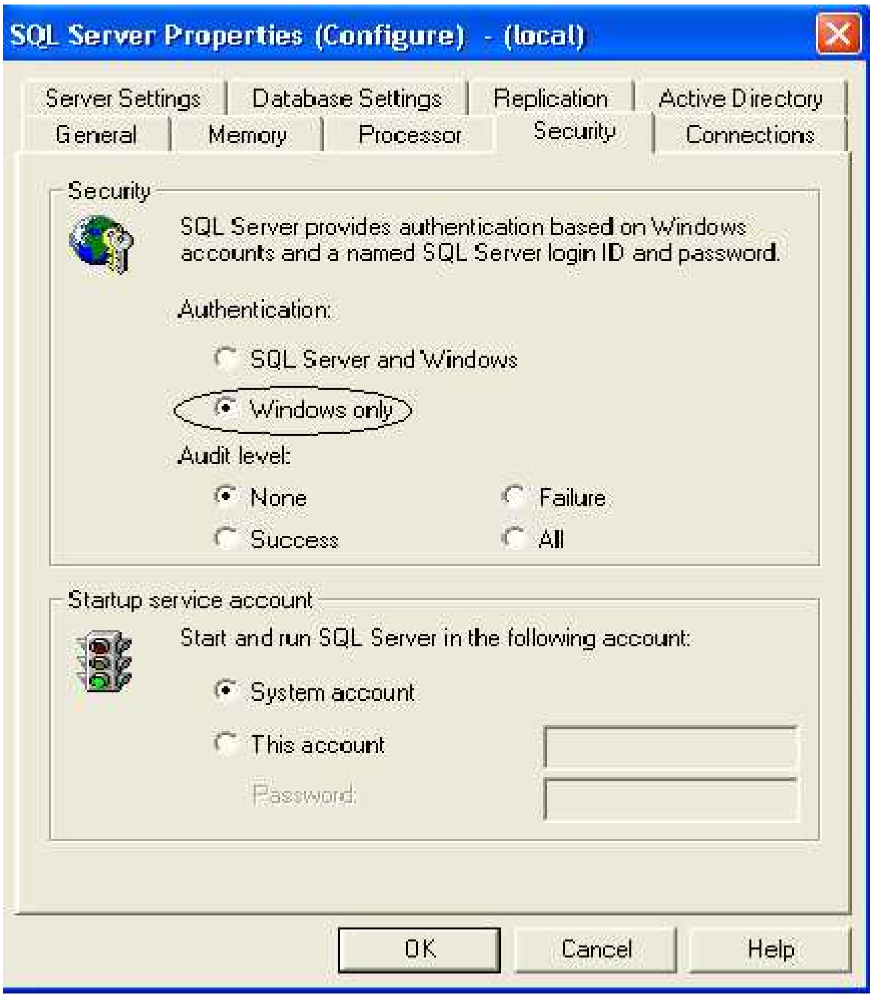
  When using .NET technology ADO.NET is used for database connections, ADO.net which is a new technology that is based on the usefulness of Microsoft ActiveX Data Object (ADO), however. It is a new technology for manipulating data, it contains numerous improvements over the previous version of ADO, and it is greatly simplifies the process of connecting a web application to a database.

Unlike ADO, ADO .NET is specifically designed for data connections located in a disconnected environment, so it the best choice when developing and implementing internet based applications.

The major point here is that why SQL server 2000? And how to configure it after installation?

As we described in this section, that the connectivity and manipulation of the database in the .Net is provided by the ADO.Net technology, now because SQL server 2000 is integrated with the .NET technology tools, it certainly should be compatible for the access by ADO.NET. However, this integration appears to be more efficient and secure especially in such systems (web applications).

By configuring the SQL server 2000 to the windows only authentication mode, which is the preferred method to use when connecting a web application to SQL server 2000 DBMS, this method does not need any user name or password to be transferred back and forth between servers, only the confirmation that the user has been authenticated by trusted sources is required to process the database request. Figure (5.1) shows the configuration of SQL server 2000 authentication mode to be "windows only authentication".



Security

SQL Server provides authentication based on Windows accounts and a named SQL Server login ID and password.

Authentication:

○ SQL Server and Windows

[✓ ● Windows only]

Audit level:

● None ○ Failure

○ Success ○ All

Startup service account

Start and run SQL Server in the following account:

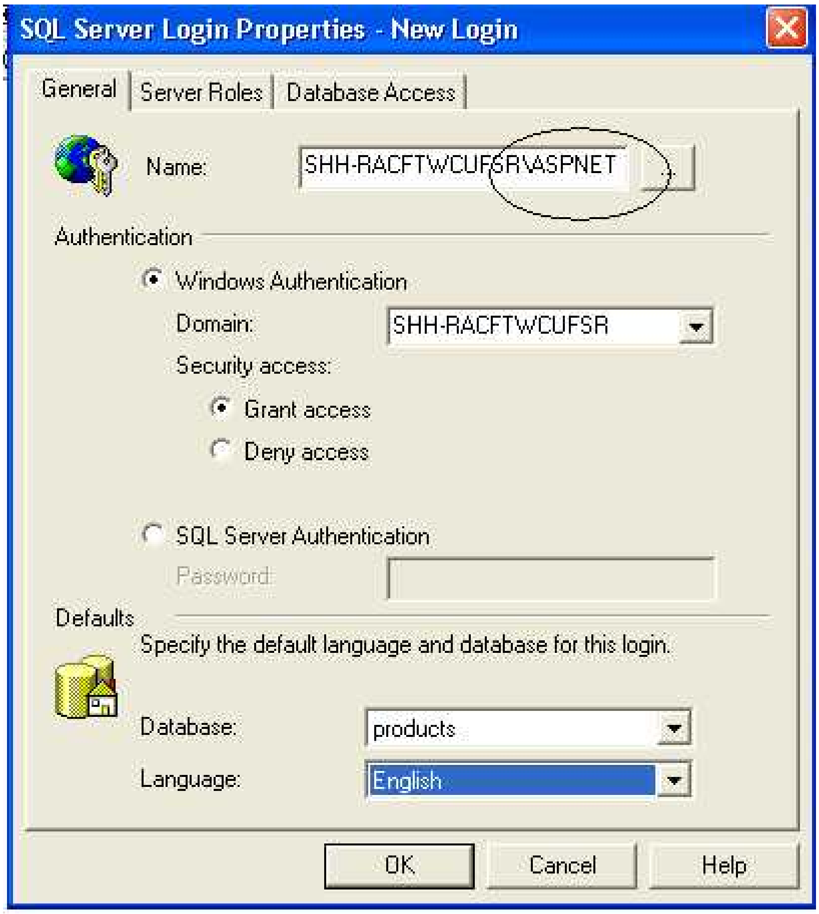
● System account

○ This account

Password:

Figure (5.1) "windows only authentication".

The last configuration of the SQL server 2000 is to work effectively and ensuring that the integration more comfortable is to add a new account in the login group of the SQL server. This account (ASPNET) is created by the .NET framework and it should be added to the login group of the SQL server. Figure (5.2) shows how to add this account to the logins group of the SQL Server.



SQL Server Login Properties - New Login

Name: SHH-RACFTWCUFSR\ASPNET

Authentication

● Windows Authentication

Domain: SHH-RACFTWCUFSR

Security access:

● Grant access

○ Deny access

○ SQL Server Authentication

Password:

Defaults

Specify the default language and database for this login.

Database: products

Language: English

Figure (5.2) Adding ASPNET account on SQL Server logins group

- **Establishing and configuring database connection**

  At this stage, the path to the data stored in the database could be opened and used as a two channel path.

  In fact, to configure a connection, it is better to do it programmatically i.e. by coding which will be described later in this chapter, but using wizards that create connections is preferred at the starting point of the database connection establishment process.

  The core here is whether a connection secure or not, but in configuring the SQL server 2000 we have chosen the windows only authentication, now in this stage we have to integrate the connection security with the SQL server 2000 security mode, thereby making the connection secure, this could done by wizard, as it appears in figure (5.3) shown bellow, we select the integrated security option to be used when creating a connection to the database using Visual Studio.NET

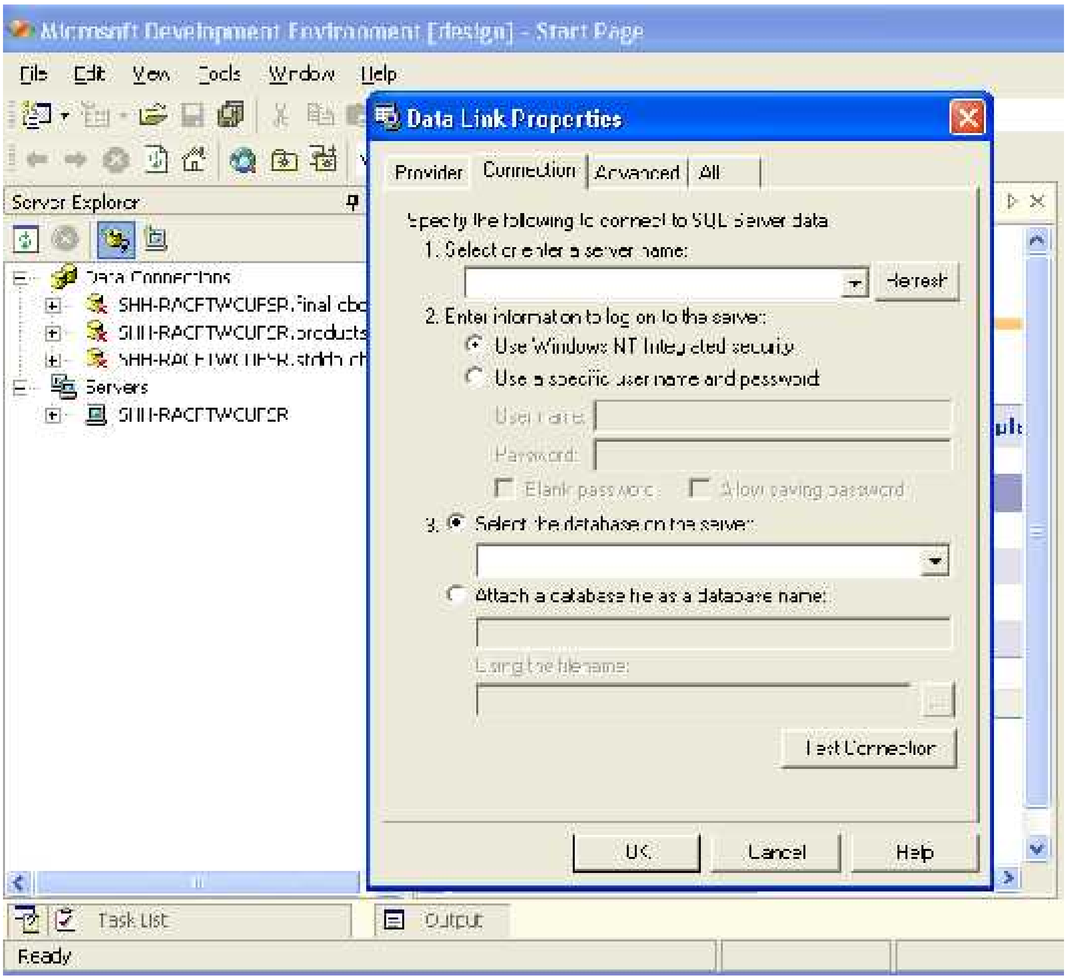

Microsoft Development Environment [design] - Start Page
File Edit View Tools Window Help
Server Explorer
Data Connections
SHH-RACETWCUFSR.Final cbd
SHH-RACETWCUFSR.products
SHH-RACETWCUFSR.stddb ch
Servers
SHH-RACETWCUFSR
Data Link Properties
Provider Connection Advanced All
Specify the following to connect to SQL Server data:
1. Select or enter a server name:
Refresh
2. Enter information to log on to the server:
● Use Windows NT Integrated security
○ Use a specific user name and password:
User name:
Password:
□ Blank password □ Allow saving password
3. ● Select the database on the server:
○ Attach a database file as a database name:
Using the filename:
Test Connection
OK Cancel Help
Task List
Output
Ready

Figure (5.3) Creating Integrated widows security data link

5.2.2.4 **Supporting Software:**

Many other software tools where required to improve our system, these are used to support the appearance of GUI of our web application. So we used macromedia Flash 6.0, Microsoft front page XP, a dope Photoshop, and Photo Impact 8.0.

5.3 Building Database:

Database table are to build in system server computer. Accordingly must run SQL server 2000 as the DBMS, the tables as described in chapter four (design / database design) are to be created using SQL server 2000 wizard tools.

5.4 Database and web interfacing

The interfacing in this system is done through the ADO.NET technology using the stored procedures in the SQL server. The web forms developed in the Visual Basic.NET are linked to the database using SQL connections and SQL commands that can be mapped to stored procedures in the database of the university.

SQL commands are created in the ASP.NET web forms and related to a certain connection; they are mapped to a stored procedure in order to approve security over the system, the usage of parameters is optional, the command then is assigned to a SqlDataReader or a SqlDataAdapter. These adapters and readers are used to access the database and read and manipulate data.

5.5 Operating the system:

To operate the system properly. Many steps should be executed before the system operates as expected in its environment:

- Configuring the network (physical connection).
- Assigning appropriate IP addresses to the computer in the network.
- Setting up the .NET framework.
- Building the system Database.
- Creating the Database connection with security options.
- Setting up the system on the server (publishing through IIS).

5.6 Coding:

In our case a web-based application we used visual Studio.NET, which simplifies the development of powerful, reliable software application by providing familiar and shared development environment. It contain pre-built component, programming wizard, as well the ability to use component built using various language. In Visual Studio.Net there is a single integrated development environment (IDE), which provides a sense of what you see is what you get (the visual programming environment).

The usage of this tool for the purpose of programming and coding reduce the time and efforts and thereby increasing the performance.

When using Visual Studio.Net as a programming environment we gain the benefits of the separation between writing the logical code (the program functionality) from one side and the design of the appearance and graphical user interface (GUI) from the other side.

5.7 Security implementation

Web sites are debatable when it comes to security since every application has different requirements from the others. They vary from highly protected to non-protected sites. The security applying techniques are also variant and available but the great challenge is what to choose that is suitable for your web site.

The security techniques applied here are mostly applied through the .NET Framework; following are the used techniques in the registration system:

1. The security options in the “Web.config” file in the ASP.NET is set to “Windows” security mode. This option relies on the security options applied by the windows itself and the NTFS file system that applies encryption over data and controlled access to the data in the server station. In this particular file we can allow certain users to access the application.

2. There are some programmatic techniques that can guarantee the safe access to the different web forms specially the one that require the password and user name checking. These techniques include using the “session” object to stop the user from going back to a password protected web form without entering the right user name and password every time he/she needs to access that page.
3. Using the authentication over the registration and updating pages of the user and generally over the administrator’s pages.

4. The SQL server logins is a technique that guarantees the access to the database but only if the user is authorized. This can provide another level of safety over the data.

5. Using the stored procedures provides an isolation of the commands from the Visual Basic Code Behind Page, this can guarantee the

stability of commands and in case of viewing those pages the database is secured from being manipulated.

6.1 Introduction

The process of testing the system will be done to ensure that the system meets its specifications and this process is one of the most important stages in the software system development.

A certain testing procedures should be performed on system and its components to insure that all the purpose of delivering a system that works properly as expected. And then an acceptance testing that may be stated as a result for the success of the testing process.

This chapter covers the testing for:

1. System units and module testing.
2. Subsystems testing
3. Integration testing.
4. System testing.
5. Acceptance testing.

6.2 Testing Schedule

Estimation time Testing process	1st week	2nd week	3rd week	4th week	5th week
Unit and module testing	░	░			
	▓	▓			
Integration testing.		░	░	░	
		▓	▓	▓	
Stress testing.			░	░	
			▓	▓	
Acceptance testing.				░	
					▓

░ Virtual work.
▓ Actual work.

Table (6.1) Testing Schedule

6.3 Unit and module testing:

We have tested the units and modules using the black box testing method, this type of testing depending on the work of suggesting specific problems and states for the system and a proceeding answer for the behavior of the system and then apply these states on the system; we suggested some problems to be tested as shown in these tables.

Inputs	Expected values	Actual values	Notes
Valid student Username And Invalid Password	Error Message	Error Message	Match
Invalid student Username And Valid Password	Error Message	Error Message	Match
Invalid student Username And Invalid Password	Error Message	Error Message	Match
Valid student Username And Password	Open home page	Open home page	Match
Forgetting Some Fields	Error Message	Error Message	Match
Select a course with out taken the successfully previous course.	Error Message	Error Message	Match
Select more than 12 hours with AVG<70.	Error Message	Error Message	Match
Select a course with out taken the	Error Message	Error Message	Match

Co_rquest.			
Select more than 18 hours if he/she will not graduated in that semester.	Error Message	Error Message	Match
Select less than 12 hours in the normal state.	Error Message	Error Message	Match
End the time of the session (30 mites).	Clouse the page.	Clouse the page.	Match
Select less than 12 hours if the student graduated.	Add.	Add.	Match
Select tow course that have a conflict in time.	Error Message	Error Message	Match
When the student has a password and click on new password.	Error Message	Error Message	Match
Select a course with out taken the previous course (with out succeed).	Error Message	Error Message	Match
Login for the registration page with out pay in the bank.	Error Message	Error Message	Match

Table (6.2): shows some problems to be tested student object

6.4 Integration testing:

All units and modulus are integrated and this integration is tested to show if there were defects that appear upon the integration of them. We have tested the integration using top-down testing. Testing here demonstrates on the interfaces between all unit system, and the functionality of the integrated parts.

After testing the integration of all unit systems, the result indicated that they work together properly.

6.5 System Testing:

We are tested the system under several conditions, some errors were detected, and upon these results, after that we are solved these problems and we imposed the system another time to testing techniques to ensure that it disposed all types of defects and problems.

6.6 Acceptance Testing:

After test the system against its requirements, we determine that it achieves its functional requirements, to operate soon in the real environment (university).

It allows student to register from any where, to register in easy and seriously way.

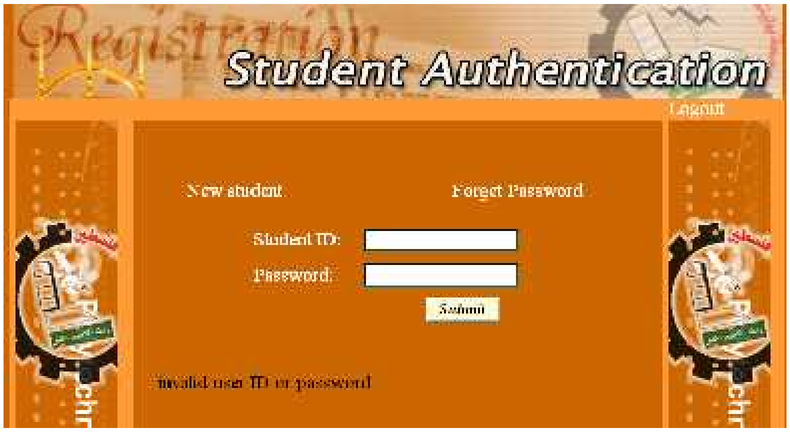
6.7 stress testing:

We are expected that the system will operate as we planning it, when it applies in the real working environment. Because of the highest operational hardware component such as the server and security system that we determine for it and because of the divided which we are determine for each college depending on the student id.

6.8 Sample Snapshots

We have selected some program snapshots to be displayed here to show how the real program behaves when working under certain situations:

1. Here we display a snapshot that shows how the login page for student when inserting an invalid input data:



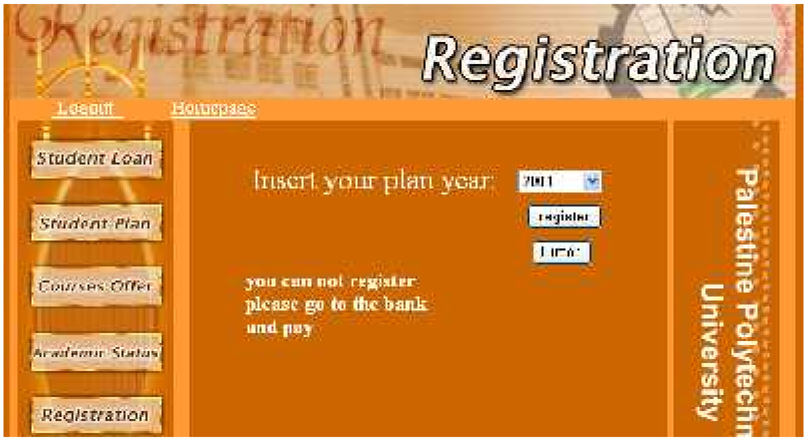
Student ID:

Password:

invalid user ID or password

Figure (6.1) login for student.

2. Here we display a snapshot that shows that the system disables the student to login before he/she paid in the bank.

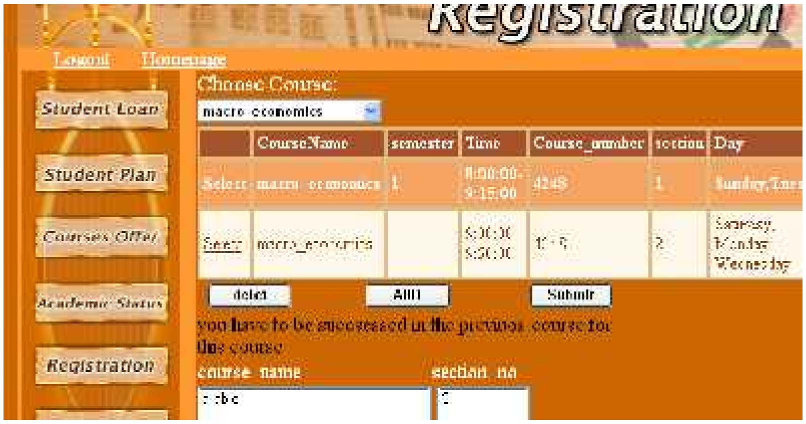


Insert your plan year: 2011

you can not register
please go to the bank
and pay

Figure (6.2) status payment.

3. Here we display a snapshot that shows that the system disables the student to register the course which he/she select with out succeed in the previous course.



Choose Course:

macro economics

	CourseName	semester	Time	Course_number	section	Day
Select	macro economics	1	8:00:00-9:15:00	4248	1	Sunday,Tues
Select	macro_economics		8:00:00-8:50:00	[illegible]	2	Saturday, Monday Wednesday

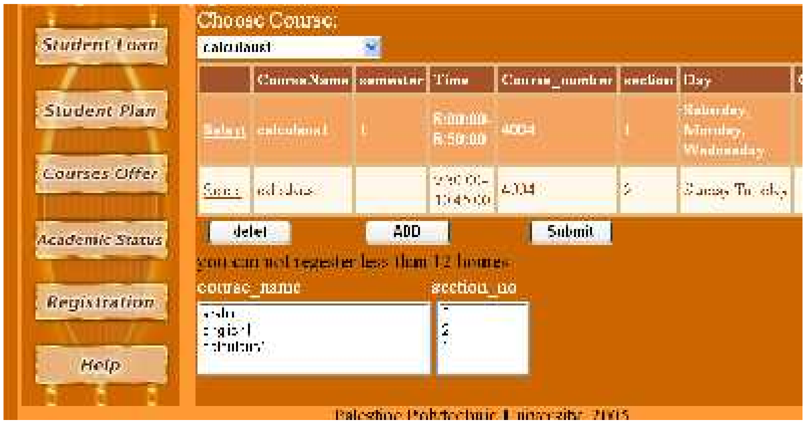
you have to be successed in the previous course for this course

course_name section_no

abc 1

Figure (6.3) succeed in previous course.

4. Here we display a snapshot that shows that the system disables the student to register less than 12 hours in the normal state.



Choose Course:

calculaus1

	CourseName	semester	Time	Course_number	section	Day
Select	calculaus1	1	8:00:00-8:50:00	4004	1	Saturday, Monday, Wednesday
Select	calculaus1		9:30:00-10:45:00	4004	2	Sunday, Tuesday

you can not register less than 12 houres

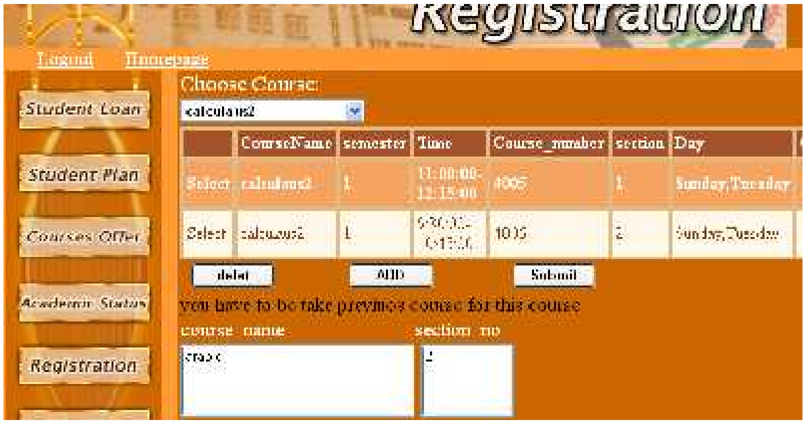
course_name section_no

arabic 1
english 2
calculaus1 1

Palestine Polytechnic University 2005

Figure (6.4) wrong horse registration.

5. Here we display a snapshot that shows that the system disables the student to register the course with out take the previous course (with out succeed).



Choose Course: calculus2

	CourseName	semester	Time	Course_number	section	Day
Select	calculus2	1	11:00:00-12:15:00	4005	1	Sunday,Tuesday
Select	calculus2	1	[illegible]	1005	2	Sunday,Tuesday

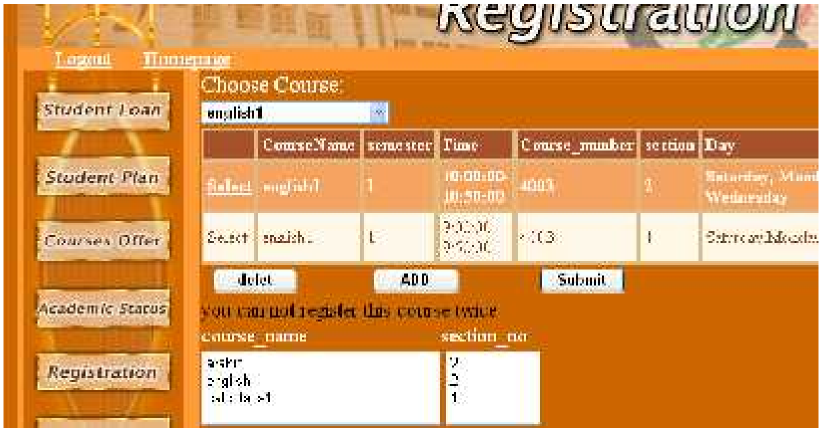
you have to be take previous course for this course

course_name: arabic

section_no: 1

Figure (6.5) the previous course.

6. Here we display a snapshot that shows that the system disables the student to register the course because there is a conflict in a time.



Choose Course: english1

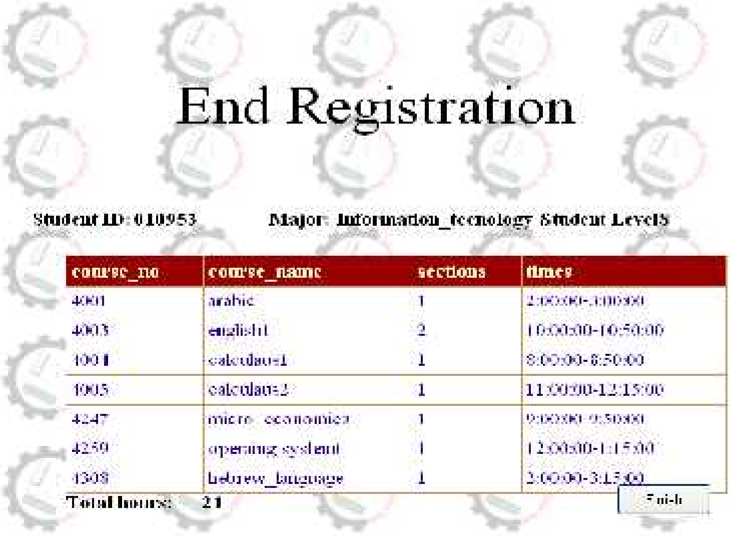
	CourseName	semester	Time	Course_number	section	Day
Select	english1	1	10:00:00-10:50:00	4003	2	Saturday, Mon Wednesday
Select	english1	1	[illegible]	4003	1	Saturday,Monday

you can not register this course twice

course_name	section_no
arabic	2
english	2
calculus1	1

Figure (6.6) student can not register twice.

7. Here we display a snapshot that shows that the system allows the student to register 12 hours if eh/she will be graduated.



End Registration

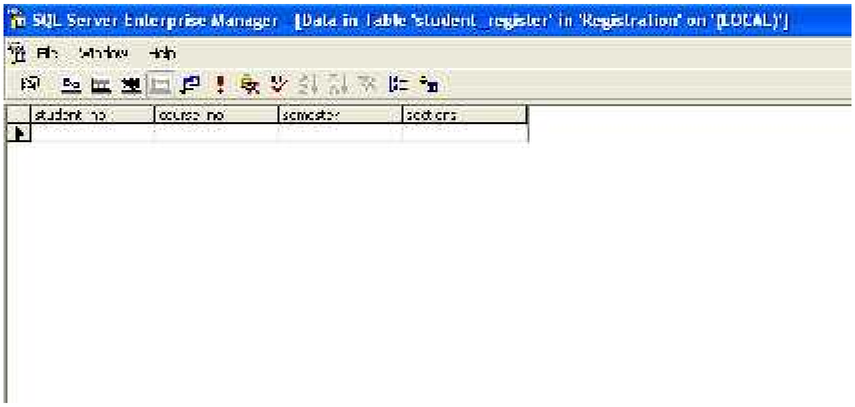
Student ID: 010953 Major: Information_tecnology Student Level5

course_no	course_name	sections	times
4001	arabic	1	2:00:00-3:00:00
4003	english1	2	10:00:00-10:50:00
4004	calculus1	1	8:00:00-8:50:00
4005	calculus2	1	11:00:00-12:15:00
4247	micro_economics	1	9:00:00-9:50:00
4259	operating system	1	12:00:00-1:15:00
4308	hebrew_language	1	2:00:00-3:15:00

Total hours: 21

Figure (6.7) report.

8. Here we display a snapshot that shows the database before the process of registration.

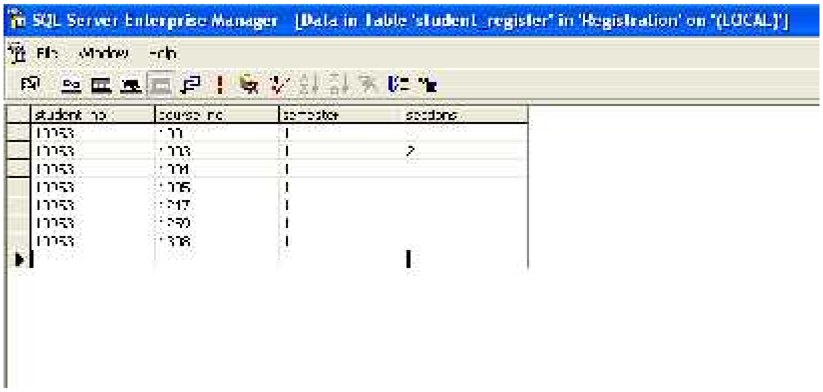


SQL Server Enterprise Manager [Data in Table 'student_register' in 'Registration' on '(LOCAL)']

student_no	course_no	semester	sections

Figure (6.8) database before registration.

9. Here we display a snapshot that shows the database after the process of registration.



SQL Server Enterprise Manager [Data in Table 'student_register' in 'Registration' on '(LOCAL)']
File Window Help

student no	course no	semester	sections
11153	1111	1	
11153	1113	1	2
11153	1114	1	
11153	1115	1	
11153	1217	1	
11153	1250	1	
11153	1308	1	

Figure (6.9) database after registration.

7.1 Introduction

In this chapter, we describe the real working environment within which the system will operate, as well how it could be established, how it could be migrated, and how it is maintained to provided the student and the university by sufficient information and guidance about system deployment and how it could be maintained and some of The information provided in this chapter is built on the distinctiveness of the system which comes from the characteristics of the SQL server and the .NET Framework.

7.2 Migration:

Toward deploying and working on the new system we describe here the steps that must be done:

1. Establishment of the production environment**:**

   By using the visual studio.net you can upgrade the system anytime and improve its design and performance. The solution explorer in the visual studio window allows you to view all the files of the system. You can then choose the web form you want to upgrade or redesign, add or remove any web form or any file necessary for the application and you can add any existing HTML document to the system and give it properties of the .NET files by renaming it from (.html) to (.aspx).

2. Deciding to apply the new system:

   A decision of the deployment of the new system must be taken by the university depending on the plan of the deployment, and this decision depending on if this system will cover all the polices and all standard of the university or not.

7.3 Maintaining the system

The areas of maintenance are divided into two integrated areas:

- SQL server maintenance.
- .NET Framework maintenance.

7.3.1 SQL server maintenance

By double-clicking on any item or clicking the (+) that are on the window you can find the objects of the server in the form of a tree. By signing on its root you can view its components.

The main component in our system is the database of registration system; it contains all the tables and the stored procedures used for the system. Also you can create any user or login and give it the desired privileges over any database on the system.

Through these components you can control the SQL server and the database locally and trace any errors by viewing the log files stored in the database.

7.3.2 The .NET Framework maintenance

The solution explorer in the visual studio window allows you to view all the files of the system to upgrade the system anytime and improve its design and performance. You can add or remove any web form or any file necessary for the application and you can add any existing HTML document to the system and give it properties of the .NET files by renaming (.aspx) rather than (.html) that you chosen the web form which you want to upgrade or redesign.

7.3.3 Backup:

The process of backup means that the programmer can generate a second copes from the database and the code; which determine as a kind of security. The backup on the system database could be configured by means that are provided by the producer company of the DBMS that we have used in our software system which is Microsoft on its DBMS product (SQL Server 200).

8.1 Introduction

This system aimed to put the PPU in the hands of any authentication student on the internet, and enable him/her to register on line which gives it a great importance and hands us over the responsibility of continuing the work over its development and success. But there are some considerations that should be taken in grant that caused the system not to be as perfect as required; following are these considerations, conclusions and recommendations

- This system has been the experience of a limited time period and limited resources for the development and implementation phases.
- This system depends mainly on the team and what they think of such a system, but the system is not run yet.
- The system need to be joined with many parts of its environment and this need to deal with a huge database.

Considering these circumstances the work team decided to take the following conclusion and recommendations.

8.2 Conclusions

The work team has concluded the following:

✓ The system is subject to upgrading and maintenance according to the student and university polices.

✓ The system is subject to the expansion of new ideas and enrichment features such as new services.

✓ As a result of testing process the system will operate properly as expected according to the university polices and requirement and a high level of security are also achieved.

8.3 Recommendations

The work team recommends the following actions as the future work for the system:

- ✓ The containment of a mail server to help more students on line.
- ✓ The ability to make integration between the university and the bank to make the process of payment so easier.
- ✓ The appliance of the full on line registration concepts such as including credit card payments, such as VISA or MASTERCARD.
- ✓ Making the process of accepting a new student in the university and retrieving all information related to him/her possible.

References

1. Registration department in Palestinian polytechnic university.
2. Microsoft Corporation, Developing Microsoft ASP.NET Web Applications Using Visual Studio.NET, 2002.
3. Ian Somerville, software Engineering, Person Education limited 2001.
4. Tay Vaughan, multimedia: making it work, Osborne/Mc Graw_Hill, fourth edition, 1998.