A

PROJECT REPORT ON

UNIVERSITY MANAGEMENT SYSTEM FOR MISAN UNIVERSITY

Submitted by

MOHAMMED TALI QASIM

ENROLL. No. : 2010-525-033

IN PARTIAL FULFILLMENT FOR THE AWARD OF THE DEGREE OF

MASTER OF COMPUTER SCIENCE

(2010-2012)

Under the guidance of

JAWED AHMED

DEPARTMENT OF COMPUTER SCIENCE

JAMIA HAMDARD

(HAMDARD UNIVERSITY)

HAMDARD NAGAR, NEW DELHI – 110062

<u>CERTIFICATE</u>

This is to certify that **MOHAMMED TALI QASIM**, the bonafide students of MSC of batch 2011-2012 have completed the Project titled "**UNIVERSITY MANAGEMENT SYSTEM for MISAN UNIVERSITY**" being submitted for the partial fulfillment of degree of MSC (Computer Science). Their work has been satisfactory and commendable.

DECLARATION

I, Mr. **MOHAMMED TALI QASIM** a student of **Master of Science (CS) (Enrolment No: 2021-523-033)** hereby declare that the dissertation entitled "**University Management System for Misan University**" which is being submitted by me to the Department of Computer Science, Jamia Hamdard, New Delhi in partial fulfillment of the requirements for the award of the degree of **Masters of Science (CS)** is my original work and has not been submitted anywhere else for the award of any Degree, Diploma, Associateship, Fellowship or other similar title or recognition.

(Signature and name of the applicant)

Date:

Place:

Acknowledgements

It is my immense pleasure to take this opportunity to thank all those who helped me directly or indirectly in the preparation of this project.

I gratefully acknowledge the support given by my supervisor Jawed Ahmed. His valuable suggestion and timely advice had been imperative

I would like to thanks my lovable friends whose queries and healthy interaction in the class give me encouragement to finish my project

Special thank to my family for their moral support, I wish them continued good health.
Last but not the least; I thank almighty God for giving us the strength to complete this work.

Mohammed Tali Qasim

2010-525-033

List of Abbreviations

S.NO	Abbreviations	Description
1	M.U.	Maisan University
2	UMS	University Management System
3	DFD	Data Flow Diagram
4	ASP	Active Server Pages
5	ADO	ActiveX Data Objects
6	SQL	Structure Query Language
7	ERD	Entity Relationship Diagram
8	SRS	System Requirements Specification

List of Figures

1

Introduction to Application

1.1) ABSTRACT

UNIVERSITY MANAGEMENT SYSTEM for MISAN UNIVERSITY (M.U.) deals with the maintenance of university, college, faculty, staff and student information within the university. The UMS is an automation system, which is used to store the colleges, HOD, faculty, staff, students, departments, courses, store and information of a colleges.

Starting from registration of a new student in the college, it maintains all the details regarding the attendance and marks of the students. The project deals with retrieval of information through an INTRANET based campus wide portal. It collects related information from all the departments of an organization and maintains files, which are used to generate reports in various forms to measure individual and overall performance of the students.

Development process of the system starts with System analysis. System analysis involves creating a formal model of the problem to be solved by understanding requirements.

1.2) The Project Objectives:

The main objectives of this project are:

- Retrieval of information through an INTRANET based campus wide portal.
- Web portal for students, faculty, HOD and university administration.
- To reduce error in the processing.
- An Effective process for the admission of student and faculty appointment.
- Automation of student/faculty information.

1.3) Vision:

Knowledge pays dividends, especially when it is paired with best-of-breed technology. That's why UMS offers exceptional benefits with an automation system to help the University to reduce the errors of management and cost of audit.

1.4) Mission:

Its mission is to help University to maintain records related to Colleges, Students, Staff, and Faculty etc. in an effective way in order to reduce costs related to record management and periodical audits. It also facilitates University with a control to limit the access to information in a secured manner.

1.5) PURPOSE OF THE SYSTEM

UNIVERSITY MANAGEMENT SYSTEM for MISAN UNIVERSITY (M.U.) deals with the maintenance of university, college, faculty, staff and student information within the university. The UMS is an automation system, which is used to store the colleges, HOD, faculty, staff, students, departments, courses and information of a college.

This project of UMS involved the automation of information about faculty, student, staff etc. that can be implemented in different college managements

The project deals with retrieval of information through an INTRANET based campus wide portal. It collects related information from all the departments of an organization and maintains files, which are used to generate reports in various forms to measure individual and overall performance of the students.

1.6) EXISTING SYSTEM

The system starts with registration of new staff and students. When the subjects are to be allocated to the faculty, the Head of the Department should enter everything in the Excel sheets. Then the staff enters corresponding subject’s attendance and marks of a student then those must also be entered in the Excel sheets and validations are to be done by the user itself. So there will be a lot of work to be done and must be more conscious during the entrance of details. So, more risk is involved.

- **PROBLEMS IN THE EXISTING SYSTEM:**

Storing and accessing the data in the form of Excel sheets and account books is a tedious work. It requires a lot of laborious work. It may often yield undesired results. Maintaining these records as piles may turn out to be a costlier task than any other of the colleges and institutions

- **Risks involved in existing system:**

Present System is time-consuming and also results in lack of getting inefficient results.

Some of the risks involved in the present system are:

- During the entrance of marks and attendance, if any mistake is done at a point, then this becomes cumulative and leads to adverse consequences
- If there is any need to retrieve results it may seem to be difficult to search.

1.7) PROPOSED SYSTEM

UMS (UNIVERSITY MANAGEMENT SYSTEM) makes management to get the most updated information always by avoiding manual accounting process. This system has the following functional divisions.

- Administrator
- User (Students / Faculties / HOD)

Administrator has the functionality of registering new colleges and courses. He also has the rights of creating department, allocating courses to departments, creating faculties, staff, students and allocating subjects to faculties, adds news, store the items of every college, receive fees with report of the fees, view attendance, add details of a library, add transportation and modifications in the data entered.

User of this may be HOD, faculty or students.

- Faculty has the facility of entering the marks, attendance of the students and view assigned subjects.

- Students can check their marks, attendance and subjects but there is no chance of modifications.

Reports must be generated for the existing data i.e. for attendance and marks of the students, which are used to assess the performance of the students. These reports should be viewed by the in charge and user.

2

Analysis Model

2.1) INTRODUCTION

After analyzing the requirements of the task to be performed, the next step is to analyze the problem and understand its context. The first activity in the phase is studying the existing system and other is to understand the requirements and domain of the new system. Both the activities are equally important, but the first activity serves as a basis of giving the functional specifications and then successful design of the proposed system. Understanding the properties and requirements of a new system is more difficult and requires creative thinking and understanding of existing running system is also difficult, improper understanding of present system can lead diversion from solution.

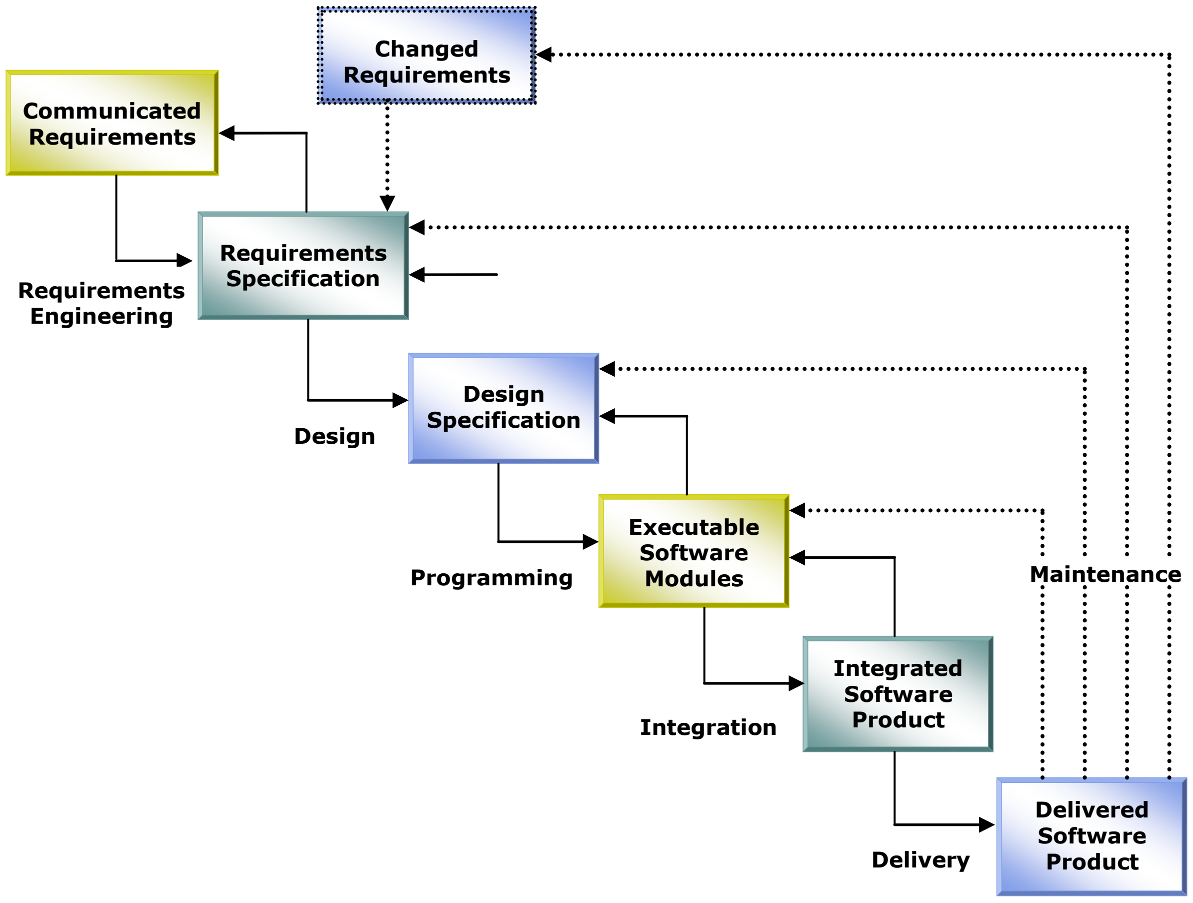
2.2) ANALYSIS MODEL

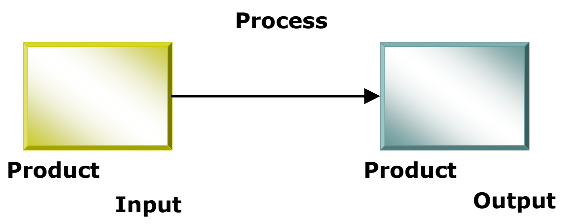
The model that is basically being followed is the WATER FALL MODEL, which states that the phases are organized in a linear order. First of all the feasibility study is done. Once that part is over the requirement analysis and project planning begins. The design starts after the requirement analysis is complete and the coding begins after the design is complete. Once the programming is completed, the testing is done. In this model the sequence of activities performed in a software development project are: -

- Requirement Analysis
- Project Planning
- System design
- Detail design
- Coding
- Unit testing
- System integration & testing

Here the linear ordering of these activities is critical. End of the phase and the output of one phase is the input of other phase. The output of each phase is to be consistent with the overall requirement of the system. Some of the qualities of spiral model are also incorporated like after the people concerned with the project review completion of each of the phase the work done.

WATER FALL MODEL was being chosen because all requirements were known beforehand and the objective of our software development is the computerization/automation of an already existing manual working system.


Changed Requirements
Communicated Requirements
Requirements Specification
Requirements Engineering
Design Specification
Design
Executable Software Modules
Programming
Maintenance
Integrated Software Product
Integration
Delivered Software Product
Delivery



Water Fall Model

2.3) FEASIBILITY STUDY

Preliminary investigation examine project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All system is feasible if they are unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

- **Technical Feasibility**
- **Operational Feasibility**
- **Economical Feasibility**

2.3.1) TECHNICAL FEASIBILITY

Technical Feasibility centers on the existing computer system hardware, software, etc. and to some extent how it can support the proposed addition. This involves financial considerations to accommodate technical enhancements. Technical support is also a reason for the success of the project. The techniques needed for the system should be available and it must be reasonable to use. Technical Feasibility is mainly concerned with the study of function, performance, and constraints that may affect the ability to achieve the system. By conducting an efficient technical feasibility we need to ensure that the project works to solve the existing problem area.

Since the project is designed with ASP.NET with C# as Front end and SQL Server 2000 as Back end, it is easy to install in all the systems wherever needed. It is more efficient, easy and user-friendly to understand by almost everyone. Huge amount of data can be handled efficiently using SQL Server as back end. Hence this project has good technical feasibility

2.3.2) OPERATIONAL FEASIBILITY

People are inherently instant to change and computers have been known to facilitate change. An estimate should be made to how strong a reaction the user staff is likely to have towards the development of the computerized system.

The staff is accustomed to computerized systems. These kinds of systems are becoming more common day by day for evaluation of the software engineers. Hence, this system is operationally feasible. As this system is technically, economically and operationally feasible, this system is judged feasible.

2.3.3) ECONOMICAL FEASIBILITY

The role of interface design is to reconcile the differences that prevail among the software engineer's design model, the designed system meet the end user requirement with economical way at minimal cost within the affordable price by encouraging more of proposed system. Economic feasibility is concerned with comparing the development cost with the income/benefit derived from the developed system. In this we need to derive how this project will help the management to take effective decisions.

Economic Feasibility is mainly concerned with the cost incurred in the implementation of the software. Since this project is developed using ASP.NET with C# and SQL Server which is more commonly available and even the cost involved in the installation process is not high.

Similarly it is easy to recruit persons for operating the software since almost all the people are aware of ASP.NET with C# and SQL Server. Even if we want to train the persons in these area the cost involved in training is also very less. Hence this project has good economic feasibility.

The system once developed must be used efficiently. Otherwise there is no meaning for developing the system. For this a careful study of the existing system and its drawbacks are needed. The user should be able to distinguish the existing one and proposed one, so that one must be able to appreciate the characteristics of the proposed system, the manual one is not highly reliable and also is considerably fast. The proposed system is efficient, reliable and also quickly responding.

3

Software and Hardware Requirements

3.1) Software and Hardware Requirements

3.1.1) Software Requirements

- Servers :
    1. Operating System Server: - Microsoft Windows XP/2003 or Windows 7
    2. Data Base Server: Microsoft SQL Server 2005 and above
- Clients : Microsoft Internet Explorer
- Tools : Microsoft Visual Studio 2008
- Framework : .NET Framework 3.5
- User Interface: ASP.NET with AJAX
- Code Behind : C#

3.1.2) Hardware Requirements

- Processor:
    3. Processor Type: - Pentium III – compatible processor or faster.
    4. Processor Speed : 1.0 GHz, Recommended 2.0 GHz or faster.

- Hard Disk : 40 GB
- RAM:

  Minimum: 512 MB.

  Recommended: 2.048 GB or more

4

Inputs and Outputs

4.1) INPUT AND OUTPUTS

The major inputs and outputs and major functions of the system are follows:

4.1.1) Outputs:

- **Inputs by Administrator:**

➢ Administrator enter his user id and password for login to authenticate in this system

➢ Administrator creates the college.

➢ Administrator adds courses for the college along with its fees information.

➢ Administrator adds the subjects to the courses.

➢ Administrator also registers Faculty, Hod and staff.

➢ Administrator adds the information related to transportation of the college.

- **Inputs by HOD:**

➢ HOD enter his user id and password for login to authenticate in this system

➢ HOD assigns subjects to the faculty.

➢ HOD keeps track of student attendance.

➢ HOD keeps track of marks obtained by students in their exam.

- **Inputs by Faculty:**

➢ Faculty enter his user id and password for login to authenticate in this system

➢ Faculty mark attendance of students.

➢ Faculty assigns marks to the students.

While registration Colleges can able to provide their information like

1. College id
2. College name
3. Address Information of college
4. Password for the college

➢ Administrator can create the various college details in this website.

4.1.2) Outputs:

- ➢ Administrator can have his own home page. Faculty, HOD and students have their own home page after completion of the authentication process.
- ➢ Admin get all colleges and staff and course details.
- ➢ The registered user's data can be stored in centralized database through the system user interface.
- ➢ Various types of information can be displayed to the users like colleges, courses and course subjects etc.
- ➢ After successful submission of log in information users can also change their password.

4.2) PROCESS MODEL USED WITH JUSTIFICATION

ACCESS CONTROL FOR DATA WHICH REQUIRE USER AUTHENTICAION

The following commands specify access control identifiers and they are typically used to authorize and authenticate the user (command codes are shown in parentheses) .

USER NAME (USER)

The user identification is that which is required by the server for access to its file system. This command will normally be the first command transmitted by the user after the control connections are made (some servers may require this).

PASSWORD (PASS)

This command must be immediately preceded by the user name command, and, for some sites, completes the user's identification for access control. Since password information is quite sensitive, it is desirable in general to "mask" it or suppress type out..

5

System Requirement Specification (SRS)

5.1) **System Requirements Specification (SRS)**

The software, Site Explorer is designed for management of web sites from a remote location.

Purpose: The main purpose for preparing this document is to give a general insight into the analysis and requirements of the existing system or situation and for determining the operating characteristics of the system.

Scope: This Document plays a vital role in the development life cycle (SDLC) and it describes the complete requirement of the system. It is meant for use by the developers and will be the basic during testing phase. Any changes made to the requirements in the future will have to go through formal change approval process.

5.2) **Developers Responsibilities Overview:**

The developer is responsible for:

- Developing the system, which meets the SRS and solving all the requirements of the system?
- Demonstrating the system and installing the system at client's location after the acceptance testing is successful.
- Submitting the required user manual describing the system interfaces to work on it and also the documents of the system.
- Conducting any user training that might be needed for using the system.
- Maintaining the system for a period of one year after installation.

5.3) **Output Design:**

Outputs from computer systems are required primarily to communicate the results of processing to users. They are also used to provides a permanent copy of the results for later consultation. The various types of outputs in general are:

- External Outputs, whose destination is outside the organization.
- Internal Outputs whose destination is within organization and they are the
- User’s main interface with the computer.

- Operational outputs whose use is purely within the computer department.
- Interface outputs, which involve the user in communicating directly.

5.4) Output Definition:

The outputs should be defined in terms of the following points:

- Type of the output
- Content of the output
- Format of the output
- Location of the output
- Frequency of the output
- Volume of the output
- Sequence of the output

It is not always desirable to print or display data as it is held on a computer. It should be decided as which form of the output is the most suitable.

5.5) Output Media:

In the next stage it is to be decided that which medium is the most appropriate for the output. The main considerations when deciding about the output media are:

- The suitability for the device to the particular application.
- The need for a hard copy.
- The response time required.
- The location of the users
- The software and hardware available.

Keeping in view the above description the project is to have outputs mainly coming under the category of internal outputs. The main outputs desired according to the requirement specification are:

The outputs were needed to be generated as a hot copy and as well as queries to be viewed on the screen. Keeping in view these outputs, the format for the output is taken from the outputs, which are currently being obtained after manual processing. The standard printer is to be used as output media for hard copies.

5.6) Input Design:

Input design is a part of overall system design. The main objective during the input design is as given below:

- To produce a cost-effective method of input.

- To achieve the highest possible level of accuracy.
- To ensure that the input is acceptable and understood by the user.

5.7) Error Avoidance:

At this stage care is to be taken to ensure that input data remains accurate form the stage at which it is recorded up to the stage in which the data is accepted by the system. This can be achieved only by means of careful control each time the data is handled.

5.8) Error Detection:

Even though every effort is make to avoid the occurrence of errors, still a small proportion of errors is always likely to occur, these types of errors can be discovered by using validations to check the input data.

5.9) Data Validation:

Procedures are designed to detect errors in data at a lower level of detail. Data validations have been included in the system in almost every area where there is a possibility for the user to commit errors. The system will not accept invalid data. Whenever an invalid data is keyed in, the system immediately prompts the user and the user has to again key in the data and the system will accept the data only if the data is correct. Validations have been included where necessary.

The system is designed to be a user friendly one. In other words the system has been designed to communicate effectively with the user. The system has been designed with popup menus.

5.10 User Interface Design:

It is essential to consult the system users and discuss their needs while designing the user interface:

User Interface Systems can be broadly classified as:

1. User initiated interface the user is in charge, controlling the progress of the user/computer dialogue. In the computer-initiated interface, the computer selects the next stage in the interaction.
2. Computer initiated interfaces

In the computer initiated interfaces the computer guides the progress of the user/computer dialogue. Information is displayed and the user response of the computer takes action or displays further information.

.1) User-Initiated Interfaces:

User initiated interfaces fall into tow approximate classes:

1. Command driven interfaces: In this type of interface the user inputs commands or queries which are interpreted by the computer.
2. Forms oriented interface: The user calls up an image of the form to his/her screen and fills in the form. The forms oriented interface is chosen because it is the best choice.

.2) Computer-Initiated Interfaces:

The following computer – initiated interfaces were used:

1. The menu system for the user is presented with a list of alternatives and the user chooses one; of alternatives.
2. Questions – answer type dialog system where the computer asks question and takes action based on the basis of the users reply.

Right from the start the system is going to be menu driven, the opening menu displays the available options. Choosing one option gives another popup menu with more options. In this way every option leads the users to data entry form where the user can key in the data.

5.11) Error Message Design:

The design of error messages is an important part of the user interface design. As user is bound to commit some errors or other while designing a system the system should be designed to be helpful by providing the user with information regarding the error he/she has committed.

This application must be able to produce output at different modules for different inputs.

5.12) **Performance Requirements:**

Performance is measured in terms of the output provided by the application.

Requirement specification plays an important part in the analysis of a system. Only when the requirement specifications are properly given, it is possible to design a system, which will fit into required environment. It rests largely in the part of the users of the existing system to give the requirement specifications because they are the people who finally use the system. This is because the requirements have to be known during the initial stages so that the system can be designed according to those requirements. It is very difficult to change the system once it has been designed and on the other hand designing a system, which does not cater to the requirements of the user, is of no use.

The requirement specification for any system can be broadly stated as given below:

- The system should be able to interface with the existing system
- The system should be accurate
- The system should be better than the existing system

The existing system is completely dependent on the user to perform all the duties.

6

Modules

6.1) Numbers of Modules

1. **Management**

   - Ability to maintain records of Admin, Faculty, Student, Hod etc. of all the colleges.
   - Provides a mechanism for Faculty and Hod to maintain marks and attendance of students and faculty.
   - Ability to keep track of College staff and transportation in an efficient way.

2. **Registration and admission of students**

   • Record the details of the student, including personal details and contact details, qualifications, etc.
   • Manage the process of student admission, which include personal and academic information, College of the study, the program will be studied and that program fees.
   • Maintain records of students registered .

3. **Complete and the collection of fees:**

   - Records the details regarding the fees structure of all the courses entered by the admin.
   - Keep track of the fees received by students for all the colleges.

4. **College Updates (including information regarding PhD):**

• It also provided the college updates to all the users' i.e. Faculty, Hod, and Student which is entered by the admin.
• It also provided the updates regarding the PhD to all the users' i.e. Faculty, Hod, and Student which is also entered by the admin.

5. **Attendance:**

• Allows Faculty and Hod's of all the Colleges to track the attendance of all the students.

6. **Exams:**

  • The organization of examinations by providing the students with the exam

  schedule information through the News section for all the colleges.

  • Maintain records of marks obtained in exams for all the colleges.

7. **Distribution of grades:**

  • Records the score obtained by all the students along with average marks obtained by them.

8. **Transportation:**

  • Scheduling of vehicles and bus.

  • Keep records of all the vehicles for all the colleges.

9. **Library:**

  • keep records of Library Card Details of all the students.

10. **Human resources:**

  • Keep records of staff for all the colleges.

  • The recruitment process methodology.

11. **Accounts:**

  • Store the information of the amount received by the college through fees received by students.

12. **Store:**

  • Keep track of store of all the colleges.

  • Control of goods and serial follow-up expiration date.

  • create a Unique No. for products.

13. **Find job opportunities for students:**

  • an effective process to recruit significantly, making the recruitment process more effective.

  • Display of vacancies on the intranet.

  • Maintain complete records for students who have been appointed in the box employment.

14. **Web portal:**

Web portals for self service for students, teachers, staff and management of the institution, in order to watch all the activities of the campus through these gates.

• Students can see the results of their exams via the Web

• Details of monitoring and fee payment, with reminders sent to the latecomers.

• You can access the tables and quotas School examinations by students and teachers.

• View information about classes, interviews, exams and curricula that are applied at various colleges.

6.2) Guidelines for Modularity

- Make sure modules perform a single task, have a single entry point, and have a single exit point.
- Isolate input-output (I-O) routines into a small number of standard modules that can be shared system-wide.
- Isolate system-dependent functions (e.g., getting date or time) in the application to ease possible future conversions to other computer platforms or to accommodate future operating system revisions.

A module is a bounded contiguous group of statements having a single name and that can be treated as a unit. In other words, a single block in a pile of blocks.

7

.NET Framework

7.1) INTRODUCTION

The Microsoft .NET Framework is a software technology that is available with several Microsoft Windows operating systems. It includes a large library of pre-coded solutions to common programming problems and a virtual machine that manages the execution of programs written specifically for the framework. The .NET Framework is a key Microsoft offering and is intended to be used by most new applications created for the Windows platform.

The pre-coded solutions that form the framework's Base Class Library cover a large range of programming needs in a number of areas, including user interface, data access, database connectivity, cryptography, web application development, numeric algorithms, and network communications. The class library is used by programmers, who combine it with their own code to produce applications.

Programs written for the .NET Framework execute in a software environment that manages the program's runtime requirements. Also part of the .NET Framework, this runtime environment is known as the Common Language Runtime (CLR). The CLR provides the appearance of an application virtual machine so that programmers need not consider the capabilities of the specific CPU that will execute the program. The CLR also provides other important services such as security, memory management, and exception handling. The class library and the CLR together compose the .NET Framework.

7.2) Principal design features

- **Interoperability**

Because interaction between new and older applications is commonly required, the .NET Framework provides means to access functionality that is implemented in programs that execute outside the .NET environment. Access to COM components is provided in the System.Runtime.InteropServices and System.EnterpriseServices namespaces of the framework; access to other functionality is provided using the P/Invoke feature.

- **Common Runtime Engine**

The Common Language Runtime (CLR) is the virtual machine component of the .NET framework. All .NET programs execute under the supervision of the CLR, guaranteeing certain properties and behaviors in the areas of memory management, security, and exception handling.

- **Base Class Library**

The Base Class Library (BCL), part of the Framework Class Library (FCL), is a library of functionality available to all languages using the .NET Framework. The BCL provides classes which encapsulate a number of common functions, including file reading and writing, graphic rendering, database interaction and XML document manipulation.

Simplified Deployment

Installation of computer software must be carefully managed to ensure that it does not interfere with previously installed software, and that it conforms to security requirements. The .NET framework includes design features and tools that help address these requirements.

- **Security**

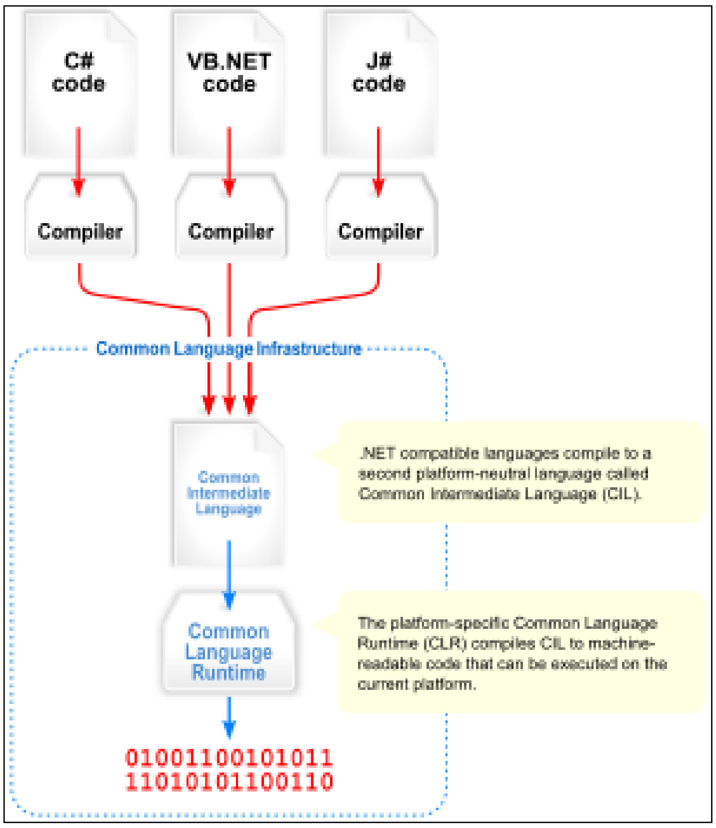
The design is meant to address some of the vulnerabilities, such as buffer overflows, that have been exploited by malicious software. Additionally, .NET provides a common security model for all applications.

- **Portability**

The design of the .NET Framework allows it to theoretically be platform agnostic, and thus cross-platform compatible. That is, a program written to use the framework should run without change on any type of system for which the framework is implemented. Microsoft's commercial implementations of the framework cover Windows, Windows CE, and the Xbox 360.

In addition, Microsoft submits the specifications for the Common Language Infrastructure (which includes the core class libraries, Common Type System, and the Common Intermediate Language), the C# language, and the C++/CLI language to both ECMA and the ISO, making them available as open standards. This makes it possible for third parties to create compatible implementations of the framework and its languages on other platforms.

- **Architecture**


C# code
VB.NET code
J# code
Compiler
Compiler
Compiler
Common Language Infrastructure
Common Intermediate Language
.NET compatible languages compile to a second platform-neutral language called Common Intermediate Language (CIL).
Common Language Runtime
The platform-specific Common Language Runtime (CLR) compiles CIL to machine-readable code that can be executed on the current platform.
01001100101011
11010101100110

Visual overview of the Common Language Infrastructure (CLI)

Common Language Infrastructure.

The core aspects of the .NET framework lie within the Common Language Infrastructure, or CLI. The purpose of the CLI is to provide a language-neutral platform for application development and execution, including functions for exception handling, garbage collection, security, and interoperability. Microsoft's implementation of the CLI is called the Common Language Runtime or CLR.

- **Assemblies**

The intermediate CIL code is housed in .NET assemblies. As mandated by specification, assemblies are stored in the Portable Executable (PE) format, common on the Windows platform for all DLL and EXE files. The assembly consists of one or more files, one of which must contain the manifest, which has the metadata for the assembly. The complete name of an assembly (not to be confused with the filename on disk) contains its simple text name, version number, culture, and public key token. The public key token is a unique hash generated when the assembly is compiled, thus two assemblies with the same public key token are guaranteed to be identical from the point of view of the framework. A private key can also be specified known only to the creator of the assembly and can be used for strong naming and to guarantee that the assembly is from the same author when a new version of the assembly is compiled (required to add an assembly to the Global Assembly Cache).

- **Metadata**

All CLI is self-describing through .NET metadata. The CLR checks the metadata to ensure that the correct method is called. Metadata is usually generated by language compilers but developers can create their own metadata through custom attributes. Metadata contains information about the assembly, and is also used to implement the reflective programming capabilities of .NET Framework.

- **Security**

.NET has its own security mechanism with two general features: Code Access Security (CAS), and validation and verification. Code Access Security is based on evidence that is associated with a specific assembly. Typically the evidence is the source of the assembly (whether it is installed on the local machine or has been downloaded from the intranet or Internet).

Code Access Security uses evidence to determine the permissions granted to the code. Other code can demand that calling code is granted a specified permission.

The demand causes the CLR to perform a call stack walk: every assembly of each method in the call stack is checked for the required permission; if any assembly is not granted the permission a security exception is thrown.

When an assembly is loaded the CLR performs various tests. Two such tests are validation and verification. During validation the CLR checks that the assembly contains valid metadata and CIL, and whether the internal tables are correct. Verification is not so exact. The verification mechanism checks to see if the code does anything that is 'unsafe'. The algorithm used is quite conservative; hence occasionally code that is 'safe' does not pass. Unsafe code will only be executed if the assembly has the 'skip verification' permission, which generally means code that is installed on the local machine.

.NET Framework uses app domains as a mechanism for isolating code running in a process. App domains can be created and code loaded into or unloaded from them independent of other app domains. This helps increase the fault tolerance of the application, as faults or crashes in one app domain do not affect rest of the application. App domains can also be configured independently with different security privileges. This can help increase the security of the application by isolating potentially unsafe code. The developer, however, has to split the application into sub domains; it is not done by the CLR.

- **Class library**

Namespaces in the BCL
System
System. CodeDom
System. Collections
System. Diagnostics
System. Globalization
System. IO
System. Resources
System. Text
System.Text.RegularExpressions

Microsoft .NET Framework includes a set of standard class libraries. The class library is organized in a hierarchy of namespaces. Most of the built in APIs are part of either System.* or Microsoft.* namespaces. It encapsulates a large number of common functions, such as file reading and writing, graphic rendering, database interaction, and XML document manipulation, among others. The .NET class libraries are available to all .NET languages. The .NET Framework class library is divided into two parts: the Base Class Library and the Framework Class Library.

The Base Class Library (BCL) includes a small subset of the entire class library and is the core set of classes that serve as the basic API of the Common Language Runtime. The classes in mscorlib.dll and some of the classes in System.dll and

System.core.dll are considered to be a part of the BCL. The BCL classes are available in both .NET Framework as well as its alternative implementations including .NET Compact Framework, Microsoft Silver light and Mono.

The Framework Class Library (FCL) is a superset of the BCL classes and refers to the entire class library those ships with .NET Framework. It includes an expanded set of libraries, including Win Forms, ADO.NET, ASP.NET, Language Integrated Query, Windows Presentation Foundation, Windows Communication Foundation among others. The FCL is much larger in scope than standard libraries for languages like C++, and comparable in scope to the standard libraries of Java.

- **Memory management**

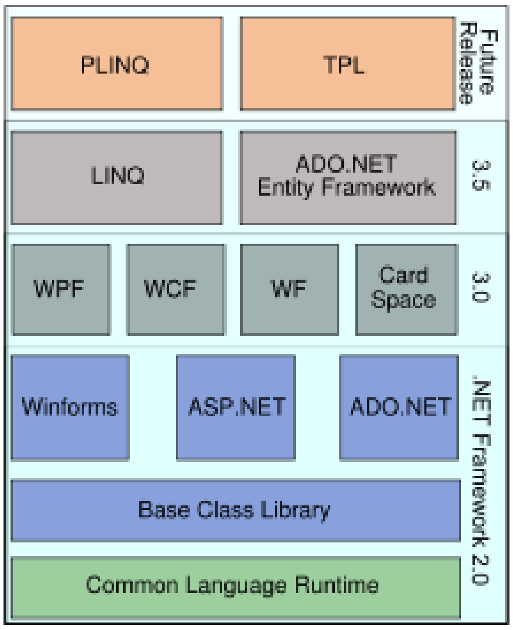
The .NET Framework CLR frees the developer from the burden of managing memory (allocating and freeing up when done); instead it does the memory management itself. To this end, the memory allocated to instantiations of .NET types (objects) is done contiguously from the managed heap, a pool of memory managed by the CLR. As long as there exists a reference to an object, which might be either a direct reference to an object or via a graph of objects, the object is considered to be in use by the CLR. When there is no reference to an object, and it cannot be reached or used, it becomes garbage. However, it still holds on to the memory allocated to it. .NET Framework includes a garbage collector which runs periodically, on a separate thread from the application's thread, that enumerates all the unusable objects and reclaims the memory allocated to them.

The .NET Garbage Collector (GC) is a non-deterministic, compacting, mark-and-sweep garbage collector. The GC runs only when a certain amount of memory has been used or there is enough pressure for memory on the system. Since it is not guaranteed when the conditions to reclaim memory are reached, the GC runs are non-deterministic. Each .NET application has a set of roots, which are pointers to objects on the managed heap (*managed objects*). These include references to static objects and objects defined as local variables or method parameters currently in scope, as well as objects referred to by CPU registers. When the GC runs, it pauses the application, and for each object referred to in the root, it recursively enumerates all the objects reachable from the root objects and marks them as reachable. It uses .NET metadata and reflection to discover the objects encapsulated by an object, and then recursively walk them. It then enumerates all the objects on the heap (which were initially allocated contiguously) using reflection. All objects not marked as reachable are garbage. This is the *mark* phase. Since the memory held by garbage is not of any consequence, it is considered free space. However, this leaves chunks of free space between objects which were initially contiguous. The objects

are then *compacted* together, by using memory to copy them over to the free space to make them contiguous again. Any reference to an object invalidated by moving the object is updated to reflect the new location by the GC. The application is resumed after the garbage collection is over.

The GC used by .NET Framework is actually *generational*. Objects are assigned a *generation*; newly created objects belong to *Generation 0*. The objects that survive a garbage collection are tagged as *Generation 1*, and the Generation 1 objects that survive another collection are *Generation 2* objects. The .NET Framework uses up to Generation 2 objects. Higher generation objects are garbage collected less frequently than lower generation objects. This helps increase the efficiency of garbage collection, as older objects tend to have a larger lifetime than newer objects. Thus, by removing older (and thus more likely to survive a collection) objects from the scope of a collection run, fewer objects need to be checked and compacted.

Versions: Microsoft started development on the .NET Framework in the late 1990s originally under the name of Next Generation Windows Services (NGWS). By late 2000 the first beta versions of .NET 1.0 were released.



The .NET Framework Stack

Version	Version Number	Release Date
1.0	1.0.3705.0	2002-01-05
1.1	1.1.4322.573	2003-04-01
2.0	2.0.50727.42	2005-11-07
3.0	3.0.4506.30	2006-11-06
3.5	3.5.21022.8	2007-11-09

7.3) ASP.NET

- **SERVER APPLICATION DEVELOPMENT**

Server-side applications in the managed world are implemented through runtime hosts. Unmanaged applications host the common language runtime, which allows your custom managed code to control the behavior of the server. This model provides you with all the features of the common language runtime and class library while gaining the performance and scalability of the host server.
The following illustration shows a basic network schema with managed code running in different server environments. Servers such as IIS and SQL Server can perform standard operations while your application logic executes through the managed code.

- **SERVER-SIDE MANAGED CODE**

ASP.NET is the hosting environment that enables developers to use the .NET Framework to target Web-based applications. However, ASP.NET is more than just a runtime host; it is a complete architecture for developing Web sites and Internet-distributed objects using managed code. Both Web Forms and XML Web services use IIS and ASP.NET as the publishing mechanism for applications, and both have a collection of supporting classes in the .NET Framework.
XML Web services, an important evolution in Web-based technology, are distributed, server-side application components similar to common Web sites. However, unlike Web-based applications, XML Web services components have no UI and are not targeted for browsers such as Internet Explorer and Netscape Navigator. Instead, XML Web services consist of reusable software components designed to be consumed by other applications, such as traditional client applications, Web-based applications, or even other XML Web services. As a result, XML Web services technology is rapidly moving application development and deployment into the highly distributed environment of the Internet.

If you have used earlier versions of ASP technology, you will immediately notice the improvements that ASP.NET and Web Forms offers. For example, you can develop Web Forms pages in any language that supports the .NET Framework. In addition, your code no longer needs to share the same file with your HTTP text (although it can continue to

do so if you prefer). Web Forms pages execute in native machine language because, like any other managed application, they take full advantage of the runtime. In contrast, unmanaged ASP pages are always scripted and interpreted. ASP.NET pages are faster, more functional, and easier to develop than unmanaged ASP pages because they interact with the runtime like any managed application.

The .NET Framework also provides a collection of classes and tools to aid in development and consumption of XML Web services applications. XML Web services are built on standards such as SOAP (a remote procedure-call protocol), XML (an extensible data format), and WSDL (the Web Services Description Language). The .NET Framework is built on these standards to promote interoperability with non-Microsoft solutions.

For example, the Web Services Description Language tool included with the .NET Framework SDK can query an XML Web service published on the Web, parse its WSDL description, and produce C# or Visual Basic source code that your application can use to become a client of the XML Web service. The source code can create classes derived from classes in the class library that handle all the underlying communication using SOAP and XML parsing. Although you can use the class library to consume XML Web services directly, the Web Services Description Language tool and the other tools contained in the SDK facilitate your development efforts with the .NET Framework.

If you develop and publish your own XML Web service, the .NET Framework provides a set of classes that conform to all the underlying communication standards, such as SOAP, WSDL, and XML. Using those classes enables you to focus on the logic of your service, without concerning yourself with the communications infrastructure required by distributed software development.

Finally, like Web Forms pages in the managed environment, your XML Web service will run with the speed of native machine language using the scalable communication of IIS.

- **ACTIVE SERVER PAGES.NET**

ASP.NET is a programming framework built on the common language runtime that can be used on a server to build powerful Web applications. ASP.NET offers several important advantages over previous Web development models:

- Enhanced Performance. ASP.NET is compiled common language runtime code running on the server. Unlike its interpreted predecessors, ASP.NET can take advantage

.of early binding, just-in-time compilation, native optimization, and caching services right out of the box. This amounts to dramatically better performance before you ever write a line of code.

- World-Class Tool Support. The ASP.NET framework is complemented by a rich toolbox and designer in the Visual Studio integrated development environment. WYSIWYG editing, drag-and-drop server controls, and automatic deployment are just a few of the features this powerful tool provides.
- Power and Flexibility. Because ASP.NET is based on the common language runtime, the power and flexibility of that entire platform is available to Web application developers. The .NET Framework class library, Messaging, and Data Access solutions are all seamlessly accessible from the Web. ASP.NET is also language-independent, so you can choose the language that best applies to your application or partition your application across many languages. Further, common language runtime interoperability guarantees that your existing investment in COM-based development is preserved when migrating to ASP.NET.
- Simplicity. ASP.NET makes it easy to perform common tasks, from simple form submission and client authentication to deployment and site configuration. For example, the ASP.NET page framework allows you to build user interfaces that cleanly separate application logic from presentation code and to handle events in a simple, Visual Basic -like forms processing model. Additionally, the common language runtime simplifies development, with managed code services such as automatic reference counting and garbage collection.

Manageability. ASP.NET employs a text-based, hierarchical configuration system, which simplifies applying settings to your server environment and Web applications. Because configuration information is stored as plain text, new settings may be applied without the aid of local administration tools. This "zero local administration" philosophy extends to deploying ASP.NET Framework applications as well. An ASP.NET Framework application is deployed to a server simply by copying the necessary files to the server. No server restart is required, even to deploy or replace running compiled code.

- Scalability and Availability. ASP.NET has been designed with scalability in mind, with features specifically tailored to improve performance in clustered and multiprocessor environments. Further, processes are closely monitored and managed by the ASP.NET runtime, so that if one misbehaves (leaks, deadlocks), a new process can be created in its place, which helps keep your application constantly available to handle requests.
- Customizability and Extensibility. ASP.NET delivers a well-factored architecture that allows developers to "plug-in" their code at the appropriate level. In fact, it is possible to extend or replace any subcomponent of the ASP.NET runtime with your own custom-written component. Implementing custom authentication or state services has never been easier.
- Security. With built in Windows authentication and per-application configuration, you can be assured that your applications are secure.

LANGUAGE SUPPORT

The Microsoft .NET Platform currently offers built-in support for three languages: C#, Visual Basic, and Java Script.

WHAT IS ASP.NET WEB FORMS?

The ASP.NET Web Forms page framework is a scalable common language runtime programming model that can be used on the server to dynamically generate Web pages.

Intended as a logical evolution of ASP (ASP.NET provides syntax compatibility with existing pages), the ASP.NET Web Forms framework has been specifically designed to address a number of key deficiencies in the previous model. In particular, it provides:

- The ability to create and use reusable UI controls that can encapsulate common functionality and thus reduce the amount of code that a page developer has to write.

- The ability for developers to cleanly structure their page logic in an orderly fashion (not "spaghetti code").
- The ability for development tools to provide strong WYSIWYG design support for pages (existing ASP code is opaque to tools).

ASP.NET Web Forms pages are text files with an .aspx file name extension. They can be deployed throughout an IIS virtual root directory tree. When a browser client requests .aspx resources, the ASP.NET runtime parses and compiles the target file into a .NET Framework class. This class can then be used to dynamically process incoming requests. (Note that the .aspx file is compiled only the first time it is accessed; the compiled type instance is then reused across multiple requests).

An ASP.NET page can be created simply by taking an existing HTML file and changing its file name extension to .aspx (no modification of code is required). For example, the following sample demonstrates a simple HTML page that collects a user's name and category preference and then performs a form post back to the originating page when a button is clicked:

ASP.NET provides syntax compatibility with existing ASP pages. This includes support for <% %> code render blocks that can be intermixed with HTML content within an .aspx file. These code blocks execute in a top-down manner at page render time.

• CODE-BEHIND WEB FORMS

ASP.NET supports two methods of authoring dynamic pages. The first is the method shown in the preceding samples, where the page code is physically declared within the originating .aspx file. An alternative approach--known as the code-behind method--enables the page code to be more cleanly separated from the HTML content into an entirely separate file.

• INTRODUCTION TO ASP.NET SERVER CONTROLS

In addition to (or instead of) using <% %> code blocks to program dynamic content, ASP.NET page developers can use ASP.NET server controls to program Web pages. Server controls are declared within an .aspx file using custom tags or intrinsic HTML tags that contain a runat="server" attributes value. Intrinsic HTML tags are handled by one of the controls in the System.Web.UI.HtmlControls namespace. Any tag that doesn't explicitly map to one of the controls is assigned the type of System.Web.UI.HtmlControls.HtmlGenericControl.

Server controls automatically maintain any client-entered values between round trips to the server. This control state is not stored on the server (it is instead stored within an <input type="hidden"> form field that is round-tripped between requests). Note also that no client-side script is required.

In addition to supporting standard HTML input controls, ASP.NET enables developers to utilize richer custom controls on their pages. For example, the following sample demonstrates how the <asp:adrotator> control can be used to dynamically display rotating ads on a page.

1. ASP.NET Web Forms provide an easy and powerful way to build dynamic Web UI.
2. ASP.NET Web Forms pages can target any browser client (there are no script library or cookie requirements).
3. ASP.NET Web Forms pages provide syntax compatibility with existing ASP pages.
4. ASP.NET server controls provide an easy way to encapsulate common functionality.

5.ASP.NET ships with 45 built-in server controls. Developers can also use controls built by third parties.

6.ASP.NET server controls can automatically project both uplevel and downlevel HTML.

7.ASP.NET templates provide an easy way to customize the look and feel of list server controls.

8.ASP.NET validation controls provide an easy way to do declarative client or server data validation.

7.4) C#.NET

- **ADO.NET OVERVIEW**

ADO.NET is an evolution of the ADO data access model that directly addresses user requirements for developing scalable applications. It was designed specifically for the web with scalability, statelessness, and XML in mind.
ADO.NET uses some ADO objects, such as the Connection and Command objects, and also introduces new objects. Key new ADO.NET objects include the Dataset, Data Reader, and Data Adapter.

The important distinction between this evolved stage of ADO.NET and previous data architectures is that there exists an object -- the DataSet -- that is separate and distinct from any data stores. Because of that, the DataSet functions as a standalone entity. You can think of the DataSet as an always disconnected recordset that knows nothing about the source or destination of the data it contains. Inside a DataSet, much like in a database, there are tables, columns, relationships, constraints, views, and so forth.

A DataAdapter is the object that connects to the database to fill the DataSet. Then, it connects back to the database to update the data there, based on operations performed while the DataSet held the data. In the past, data processing has been primarily connection-based. Now, in an effort to make multi-tiered apps more efficient, data processing is turning to a message-based approach that revolves around chunks of information. At the center of this approach is the DataAdapter, which provides a bridge to retrieve and save data between a DataSet and its source data store. It accomplishes this by means of requests to the appropriate SQL commands made against the data store.

The XML-based DataSet object provides a consistent programming model that works with all models of data storage: flat, relational, and hierarchical. It does this by having no 'knowledge' of the source of its data, and by representing the data that it holds as collections and data types. No matter what the source of the data within the DataSet is, it is manipulated through the same set of standard APIs exposed through the DataSet and its subordinate objects.

While the DataSet has no knowledge of the source of its data, the managed provider has detailed and specific information. The role of the managed provider is to connect, fill, and persist the DataSet to and from data stores. The OLE DB and SQL Server .NET Data Providers (System.Data.OleDb and System.Data.SqlClient) that are part of the .Net Framework provide four basic objects: the Command, Connection, DataReader and DataAdapter. In the remaining sections of this document, we'll walk through each part of the DataSet and the OLE DB/SQL Server .NET Data Providers explaining what they are, and how to program against them.

The following sections will introduce you to some objects that have evolved, and some that are new. These objects are:

- Connections. For connection to and managing transactions against a database.
- Commands. For issuing SQL commands against a database.
- DataReaders. For reading a forward-only stream of data records from a SQL Server data source.
- DataSet. For storing, Remoting and programming against flat data, XML data and relational data.
- DataAdapters. For pushing data into a DataSet, and reconciling data against a database.

When dealing with connections to a database, there are two different options: SQL Server .NET Data Provider (System.Data.SqlClient) and OLE DB .NET Data Provider (System.Data.OleDb). In these samples we will use the SQL Server .NET Data Provider. These are written to talk directly to Microsoft SQL Server. The OLE DB .NET Data Provider is used to talk to any OLE DB provider (as it uses OLE DB underneath).

Connections:

Connections are used to 'talk to' databases, and are represented by provider-specific classes such as SqlConnection. Commands travel over connections and resultsets are returned in the form of streams which can be read by a DataReader object, or pushed into a DataSet object.

- **Commands:**

Commands contain the information that is submitted to a database, and are represented by provider-specific classes such as SqlCommand. A command can be a stored procedure call, an UPDATE statement, or a statement that returns results. You can also use input and output parameters, and return values as part of your command syntax. The example below shows how to issue an INSERT statement against the Northwind database.

- **DataReaders:**

The DataReader object is somewhat synonymous with a read-only/forward-only cursor over data. The DataReader API supports flat as well as hierarchical data. A DataReader object is returned after executing a command against a database. The format of the returned DataReader object is different from a recordset. For example, you might use the DataReader to show the results of a search list in a web page.

- **DATASETS AND DATAADAPTERS:**

I. DataSets

The DataSet object is similar to the ADO Recordset object, but more powerful, and with one other important distinction: the DataSet is always disconnected. The DataSet object represents a cache of data, with database-like structures such as tables, columns, relationships, and constraints. However, though a DataSet can and does behave much like a database, it is important to remember that DataSet objects do not interact directly with databases, or other source data. This allows the developer to work with a programming model that is always consistent, regardless of where the source data resides. Data coming from a database, an XML file, from code, or user input can all be placed into DataSet objects. Then, as changes are made to the DataSet they can be tracked and verified before updating the source data. The

GetChanges method of the DataSet object actually creates a second DatSet that contains only the changes to the data. This DataSet is then used by a DataAdapter (or other objects) to update the original data source.

The DataSet has many XML characteristics, including the ability to produce and consume XML data and XML schemas.

XML schemas can be used to describe schemas interchanged via WebServices. In fact, a DataSet with a schema can actually be compiled for type safety and statement completion.

II. DATAADAPTERS (OLEDB/SQL)

The DataAdapter object works as a bridge between the DataSet and the source data. Using the provider-specific SqlDataAdapter (along with its associated SqlCommand and SqlConnection) can increase overall performance when working with a Microsoft SQL Server databases. For other OLE DB-supported databases, you would use the OleDbDataAdapter object and its associated OleDbCommand and OleDbConnection objects.

The DataAdapter object uses commands to update the data source after changes have been made to the DataSet. Using the Fill method of the DataAdapter calls the SELECT command; using the Update method calls the INSERT, UPDATE or DELETE command for each changed row. You can explicitly set these commands in order to control the statements used at runtime to resolve changes, including the use of stored procedures. For ad-hoc scenarios, a CommandBuilder object can generate these at run-time based upon a select statement. However, this run-time generation requires an extra round-trip to the server in order to gather required metadata, so explicitly providing the INSERT, UPDATE, and DELETE commands at design time will result in better run-time performance.

1. ADO.NET is the next evolution of ADO for the .Net Framework.
2. ADO.NET was created with n-Tier, statelessness and XML in the forefront. Two new objects, the DataSet and DataAdapter, are provided for these scenarios.
3. ADO.NET can be used to get data from a stream, or to store data in a cache for updates.
4. There is a lot more information about ADO.NET in the documentation.
5. Remember, you can execute a command directly against the database in order to do inserts, updates, and deletes. You don't need to first put data into a DataSet in order to insert, update, or delete it.

Also, you can use a DataSet to bind to the data, move through the data, and navigate data relationships.

7.5) SQL SERVER

A database management, or DBMS, gives the user access to their data and helps them transform the data into information. Such database management systems include dBase, paradox, IMS, SQL Server and SQL Server. These systems allow users to create, update and extract information from their database.

A database is a structured collection of data. Data refers to the characteristics of people, things and events. SQL Server stores each data item in its own fields. In SQL Server, the fields relating to a particular person, thing or event are bundled together to form a single complete unit of data, called a record (it can also be referred to as raw or an occurrence). Each record is made up of a number of fields. No two fields in a record can have the same field name.

During an SQL Server Database design project, the analysis of your business needs identifies all the fields or attributes of interest. If your business needs change over time, you define any additional fields or change the definition of existing fields.

- **SQL SERVER TABLES**

SQL Server stores records relating to each other in a table. Different tables are created for the various groups of information. Related tables are grouped together to form a database.

- **PRIMARY KEY**

Every table in SQL Server has a field or a combination of fields that uniquely identifies each record in the table. The Unique identifier is called the Primary Key, or simply the Key. The primary key provides the means to distinguish one record from all other in a table. It allows the user and the database system to identify, locate and refer to one particular record in the database.

• RELATIONAL DATABASE

Sometimes all the information of interest to a business operation can be stored in one table. SQL Server makes it very easy to link the data in multiple tables. Matching an employee to the department in which they work is one example. This is what makes SQL Server a relational database management system, or RDBMS. It stores data in two or more tables and enables you to define relationships between the table and enables you to define relationships between the tables.

• FOREIGN KEY

When a field is one table matches the primary key of another field is referred to as a foreign key. A foreign key is a field or a group of fields in one table whose values match those of the primary key of another table

.

• REFERENTIAL INTEGRITY

Not only does SQL Server allow you to link multiple tables, it also maintains consistency between them. Ensuring that the data among related tables is correctly matched is referred to as maintaining referential integrity.

DATA ABSTRACTION A major purpose of a database system is to provide users with an abstract view of the data. This system hides certain details of how the data is stored and maintained. Data abstraction is divided into three levels.

Physical level: This is the lowest level of abstraction at which one describes how the data are actually stored.

Conceptual Level: At this level of database abstraction all the attributed and what data are actually stored is described and entries and relationship among them.

View level: This is the highest level of abstraction at which one describes only part of the database.

• ADVANTAGES OF RDBMS

- Redundancy can be avoided
- Inconsistency can be eliminated
- Data can be Shared
- Standards can be enforced

- Security restrictions ca be applied
- Integrity can be maintained
- Conflicting requirements can be balanced
- Data independence can be achieved.

• DISADVANTAGES OF DBMS

A significant disadvantage of the DBMS system is cost. In addition to the cost of purchasing of developing the software, the hardware has to be upgraded to allow for the extensive programs and the workspace required for their execution and storage.
While centralization reduces duplication, the lack of duplication requires that the database be adequately backed up so that in case of failure the data can be recovered.

• FEATURES OF SQL SERVER (RDBMS)

SQL SERVER is one of the leading database management systems (DBMS) because it is the only Database that meets the uncompromising requirements of today's most demanding information systems. From complex decision support systems (DSS) to the most rigorous online transaction processing (OLTP) application, even application that require simultaneous DSS and OLTP access to the same critical data, SQL Server leads the industry in both performance and capability.
SQL SERVER is a truly portable, distributed, and open DBMS that delivers unmatched performance, continuous operation and support for every database.
SQL SERVER RDBMS is high performance fault tolerant DBMS which is specially designed for online transactions processing and for handling large database application.
SQL SERVER with transactions processing option offers two features which contribute to very high level of transaction processing throughput, which are

- The row level lock manager

• ENTERPRISE WIDE DATA SHARING

The unrivaled portability and connectivity of the SQL SERVER DBMS enables all the systems in the organization to be linked into a singular, integrated computing resource.

- **PORTABILITY**

SQL SERVER is fully portable to more than 80 distinct hardware and operating systems platforms, including UNIX, MSDOS, OS/2, Macintosh and dozens of proprietary platforms. This portability gives complete freedom to choose the database server platform that meets the system requirements.

- **OPEN SYSTEMS**

SQL SERVER offers a leading implementation of industry –standard SQL. SQL Server's open architecture integrates SQL SERVER and non –SQL SERVER DBMS with industry's most comprehensive collection of tools, application, and third party software products SQL Server's Open architecture provides transparent access to data from other relational database and even non-relational database.

- **DISTRIBUTED DATA SHARING**

SQL Server's networking and distributed database capabilities to access data stored on remote server with the same ease as if the information was stored on a single local computer. A single SQL statement can access data at multiple sites. You can store data where system requirements such as performance, security or availability dictate.

- **UNMATCHED PERFORMANCE**

The most advanced architecture in the industry allows the SQL SERVER DBMS to deliver unmatched performance.

8

Software Design

8.1) INTRODUCTION

Software design sits at the technical kernel of the software engineering process and is applied regardless of the development paradigm and area of application. Design is the first step in the development phase for any engineered product or system. The designer's goal is to produce a model or representation of an entity that will later be built. Beginning, once system requirement have been specified and analyzed, system design is the first of the three technical activities -design, code and test that is required to build and verify software.

The importance can be stated with a single word "Quality". Design is the place where quality is fostered in software development. Design provides us with representations of software that can assess for quality. Design is the only way that we can accurately translate a employee's view into a finished software product or system. Software design serves as a foundation for all the software engineering steps that follow. Without a strong design we risk building an unstable system – one that will be difficult to test, one whose quality cannot be assessed until the last stage.

During design, progressive refinement of data structure, program structure, and procedural details are developed reviewed and documented. System design can be viewed from either technical or project management perspective. From the technical point of view, design is comprised of four activities – architectural design, data structure design, interface design and procedural design.

8.2) DATA FLOW DIAGRAMS

A data flow diagram is graphical tool used to describe and analyze movement of data through a system. These are the central tool and the basis from which the other components are developed. The transformation of data from input to output, through processed, may be described logically and independently of physical components associated with the system. These are known as the logical data flow diagrams. The physical data flow diagrams show the actual implements and movement of data between people, departments and workstations. A full description of a system actually consists of a set of data flow diagrams. Using two familiar notations Yourdon, Gane and Sarson notation develops the data flow diagrams. Each component in a DFD is labeled with a descriptive name. Process is further identified with a number that will be used for identification purpose. The development of DFD'S is done in several levels. Each process in lower level diagrams can be broken down into a more detailed DFD in the next level. The lop-level diagram is often called context diagram. It consists a single process bit, which plays vital role in studying the current system. The process in the context level diagram is exploded into other process at the first level DFD.

The idea behind the explosion of a process into more process is that understanding at one level of detail is exploded into greater detail at the next level. This is done until further explosion is necessary and an adequate amount of detail is described for analyst to understand the process.

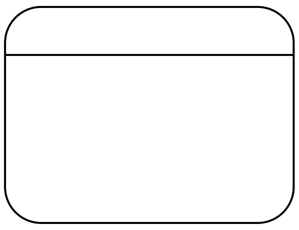
Larry Constantine first developed the DFD as a way of expressing system requirements in a graphical from, this lead to the modular design.

A DFD is also known as a "bubble Chart" has the purpose of clarifying system requirements and identifying major transformations that will become programs in system design. So it is the starting point of the design to the lowest level of detail. A DFD consists of a series of bubbles joined by data flows in the system.

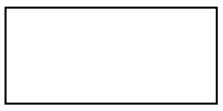
8.3) **DFD SYMBOLS:**

In the DFD, there are four symbols

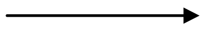
1. A square defines a source(originator) or destination of system data
2. An arrow identifies data flow. It is the pipeline through which the information flows
3. A circle or a bubble represents a process that transforms incoming data flow into outgoing data flows.
4. An open rectangle is a data store, data at rest or a temporary repository of data.



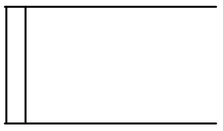
Process that transforms data flow.



Source or Destination of data



Data flow



Data Store

8.4) CONSTRUCTING A DFD:

Several rules of thumb are used in drawing DFD'S:

1. Process should be named and numbered for an easy reference. Each name should be representative of the process.
2. The direction of flow is from top to bottom and from left to right. Data traditionally flow from source to the destination although they may flow back to the source. One way to indicate this is to draw long flow line back to a source.
3. An alternative way is to repeat the source symbol as a destination. Since it is used more than once in the DFD it is marked with a short diagonal.
4. When a process is exploded into lower level details, they are numbered.
5. The names of data stores and destinations are written in capital letters. Process and dataflow names have the first letter of each work capitalized.

A DFD typically shows the minimum contents of data store. Each data store should contain all the data elements that flow in and out.

Questionnaires should contain all the data elements that flow in and out. Missing interfaces redundancies and like is then accounted for often through interviews.

8.5) SALIENT FEATURES OF DFD'S

1. The DFD shows flow of data, not of control loops and decision are controlled considerations do not appear on a DFD.
2. The DFD does not indicate the time factor involved in any process whether the dataflow take place daily, weekly, monthly or yearly.
3. The sequence of events is not brought out on the DFD.

8.6) TYPES OF DATA FLOW DIAGRAMS

1. Current Physical.
2. Current Logical.
3. New Logical.
4. New Physical.

1. CURRENT PHYSICAL:

In Current Physical DFD process label include the name of people or their positions or the names of computer systems that might provide some of the overall system-processing label includes an identification of the technology used to process the data. Similarly data flows and data stores are often labels with the names of the actual physical media on which data are stored such as file folders, computer files, business forms or computer tapes.

2. CURRENT LOGICAL:

The physical aspects at the system are removed as much as possible so that the current system is reduced to its essence to the data and the processors that transforms them regardless of actual physical form.

3. NEW LOGICAL:

This is exactly like a current logical model if the user were completely happy with the user were completely happy with the functionality of the current system but had problems with how it was implemented typically through the new logical model will differ from current logical model while having additional functions, absolute function removal and inefficient flows recognized.

4. NEW PHYSICAL

The new physical represents only the physical implementation of the new system.

8.7) RULES GOVERNING THE DFD'S PROCESS

1) No process can have only outputs.
2) No process can have only inputs. If an object has only inputs than it must be a sink.
3) A process has a verb phrase label.

8.8) DATA STORE

1) Data cannot move directly from one data store to another data store, a process must move data.

2) Data cannot move directly from an outside source to a data store, a process, which receives, must move data from the source and place the data into data store

3) A data store has a noun phrase label.

8.9) SOURCE OR SINK

The origin and /or destination of data.

1) Data cannot move direly from a source to sink it must be moved by a process

2) A source and /or sink has a noun phrase land

8.10) DATA FLOW

1) A Data Flow has only one direction of flow between symbols. It may flow in both directions between a process and a data store to show a read before an update. The later is usually indicated however by two separate arrows since these happen at different type.

2) A join in DFD means that exactly the same data comes from any of two or more different processes data store or sink to a common location.

3) A data flow cannot go directly back to the same process it leads. There must be at least one other process that handles the data flow produce some other data flow returns the original data into the beginning process.

4) A Data flow to a data store means update (delete or change).

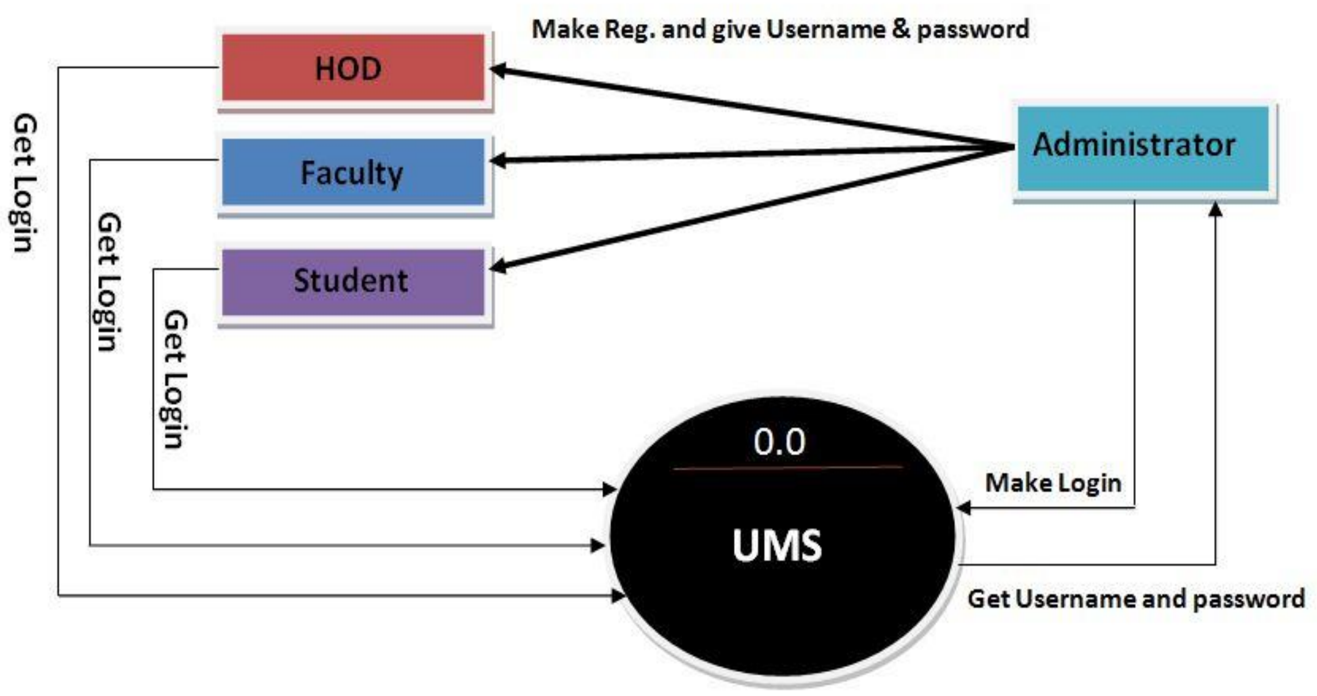
5) A data Flow from a data store means retrieve or use.

A data flow has a noun phrase label more than one data flow noun phrase can appear on a single arrow as long as all of the flows on the same arrow move together as one package.

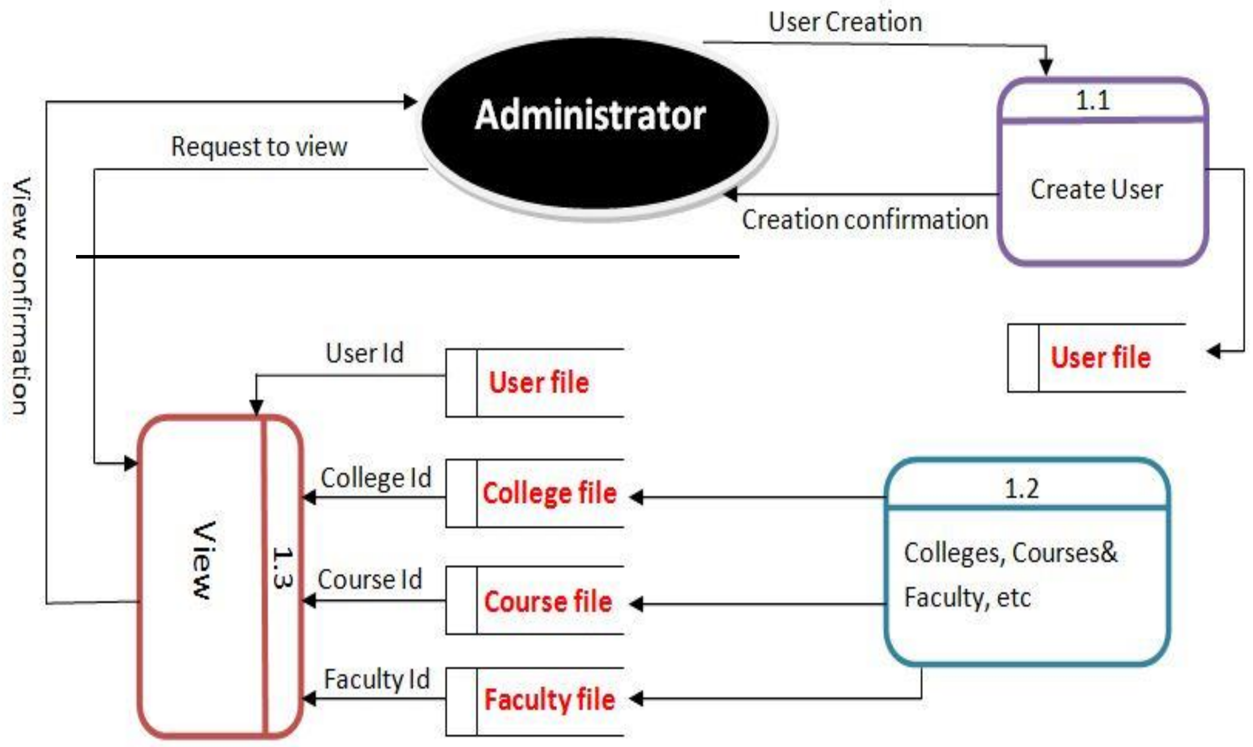
8.11) DFD Diagrams

Context Level (0th level)

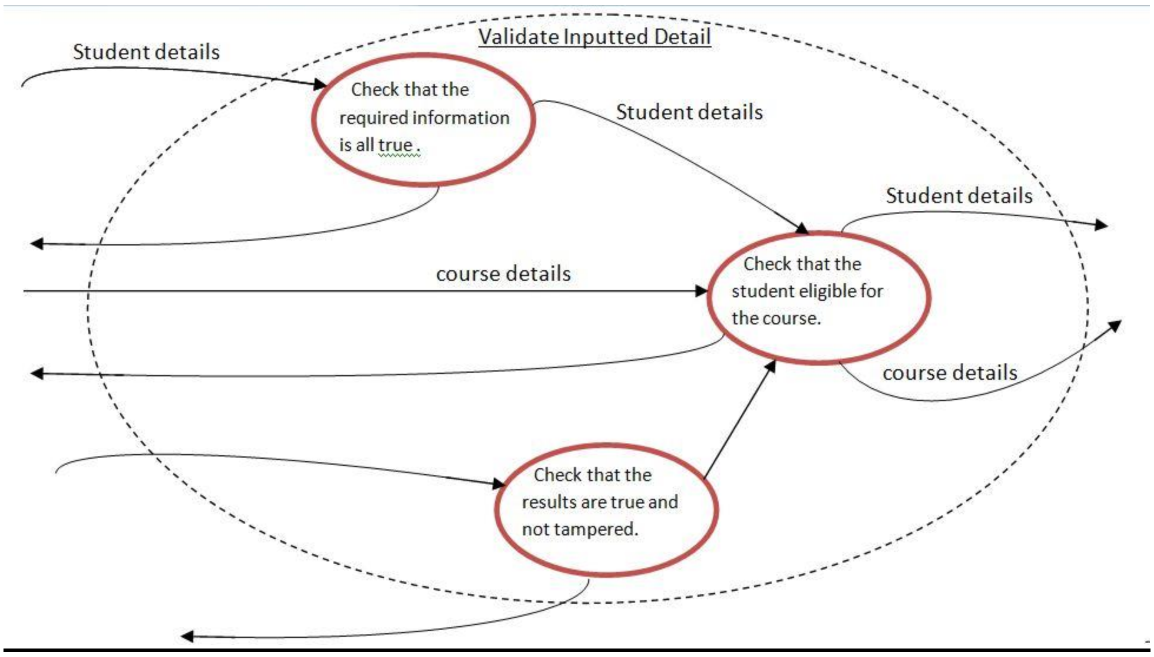
All the above processes together are decomposed and represented in CONTEXT DIAGRAM**.**



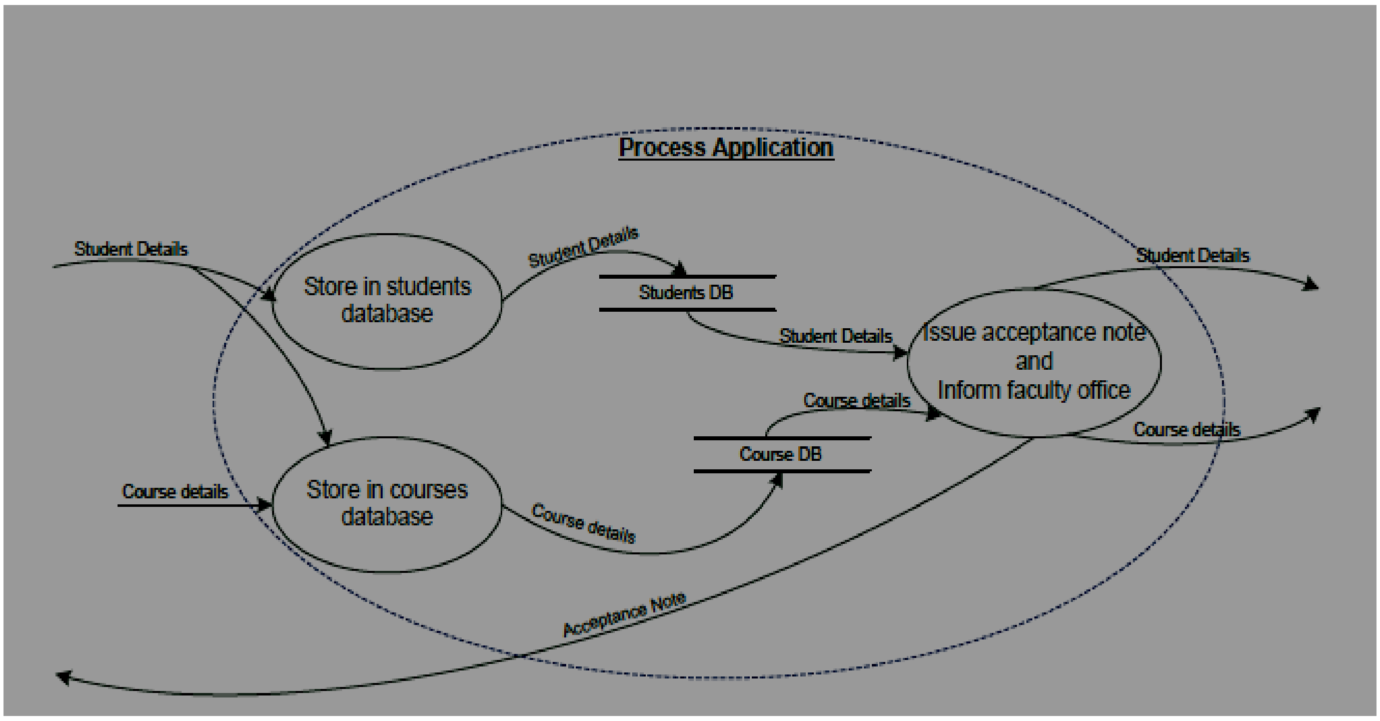
DFD (1st Level)



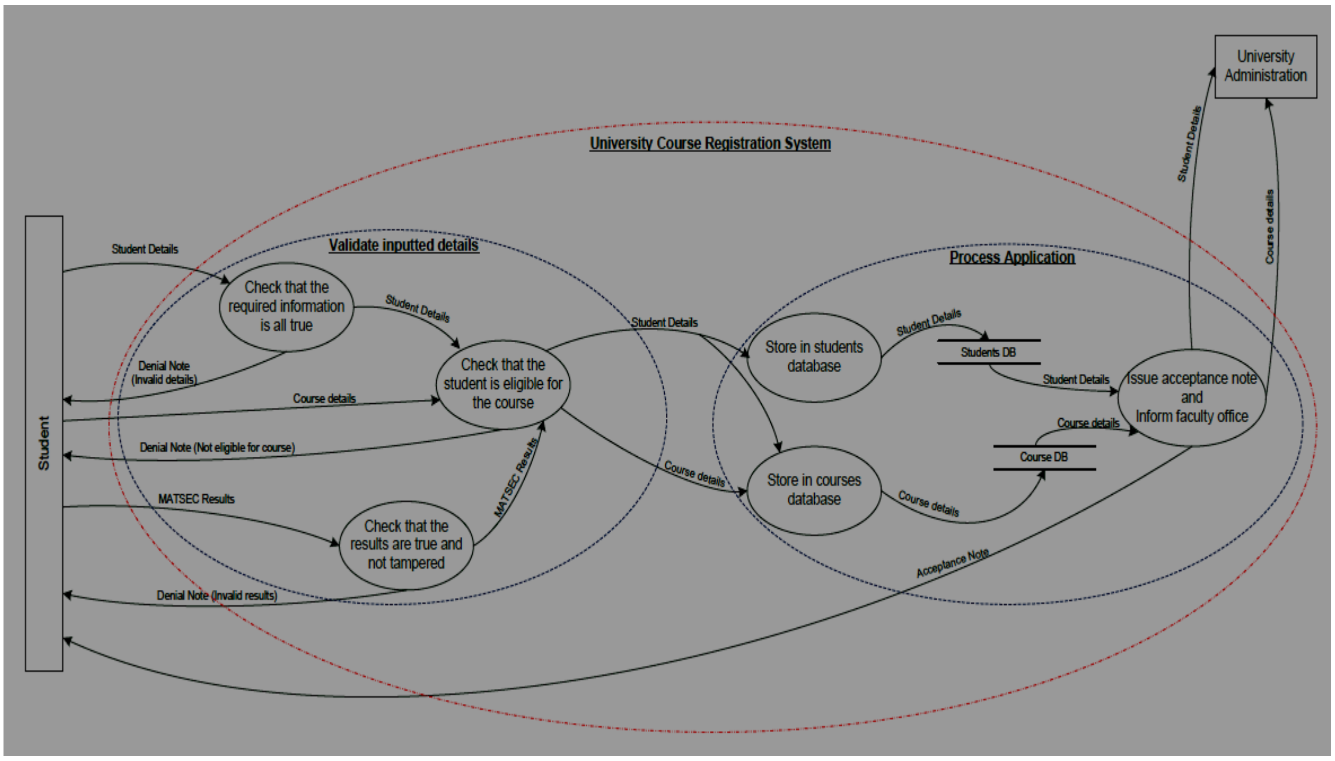
DFD (2nd level)


Validate Inputted Detail
Student details
Check that the required information is all true.
Student details
Student details
course details
Check that the student eligible for the course.
course details
Check that the results are true and not tampered.

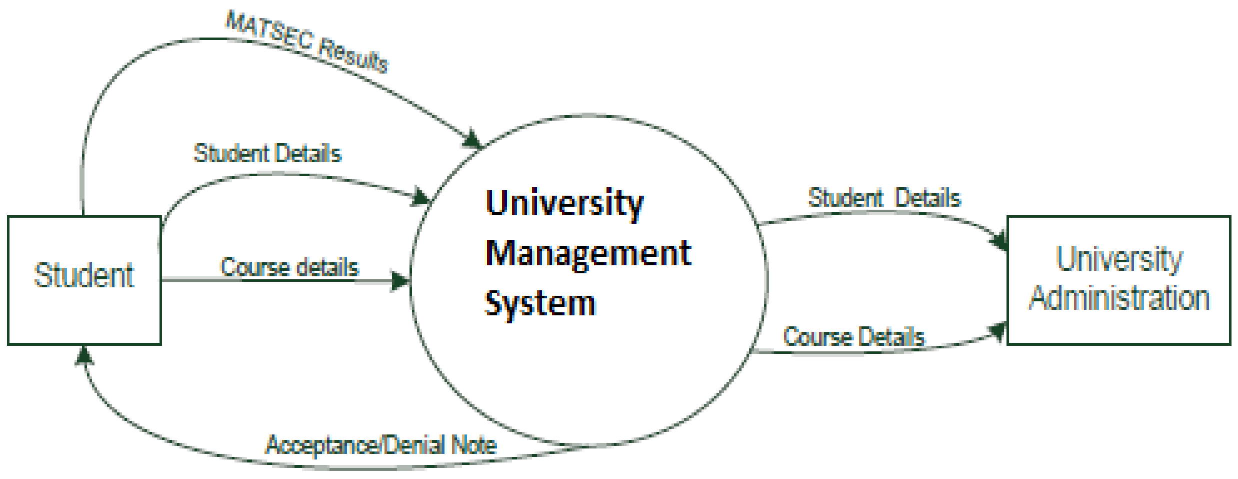
DFD (3rdlevel)



DFD (4th level)


University Administration
University Course Registration System
Student Details
Course details
Student
Student Details
Validate inputted details
Process Application
Check that the required information is all true
Student Details
Student Details
Student Details
Store in students database
Students DB
Denial Note (Invalid details)
Check that the student is eligible for the course
Student Details
Issue acceptance note and Inform faculty office
Course details
Course details
Denial Note (Not eligible for course)
Course DB
Course details
MATSEC Results
Store in courses database
Course details
MATSEC Results
Check that the results are true and not tampered
Acceptance Note
Denial Note (Invalid results)

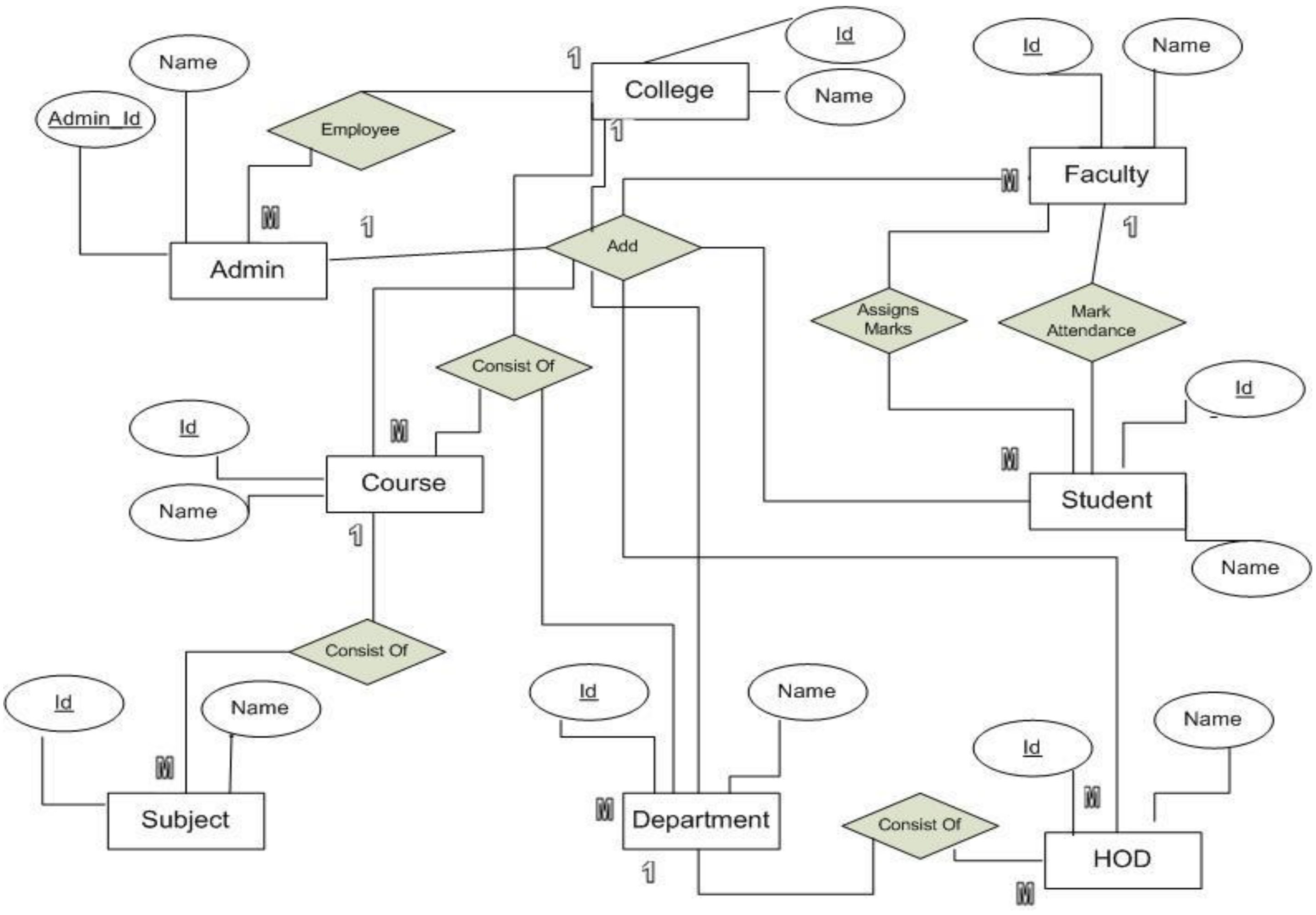
DFD (5th level)



8.12) ER Diagrams

- The relation upon the system is structure through a conceptual ER-Diagram, which not only specifics the existential entities but also the standard relations through which the system exists and the cardinalities that are necessary for the system state to continue.
- The entity Relationship Diagram (ERD) depicts the relationship between the data objects. The ERD is the notation that is used to conduct the date modeling activity the attributes of each data object noted is the ERD can be described resign a data object descriptions.
- The set of primary components that are identified by the ERD are
- Data object
- Relationships
- Attributes
- Various types of indicators.

The primary purpose of the ERD is to represent data objects and their relationships


Name
Admin_Id
Employee
College
Id
Name
Id
Name
Faculty
Admin
Add
Assigns Marks
Mark Attendance
Consist Of
Id
Course
Id
Name
Student
Name
Consist Of
Id
Name
Subject
Id
Name
Department
Id
Name
Consist Of
HOD
1
M
1
1
M
1
M
M
1
M
M
1
M
M

ER Diagram

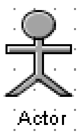
8.13) UML DIAGRAMS

The **use case** diagrams describe system functionality as a set of tasks that the system must carry out and **actors** who interact with the system to complete the tasks.

Use Case:

Each **use case** on the diagram represents a single task that the system needs to carry out. *Buy a Product, Add Client, Make Purchase* and *Validate Order Information* are all examples of use cases. Some use cases may include or extend a task represented by another use case. For example, in order to make a purchase, the order information will need to be validated.

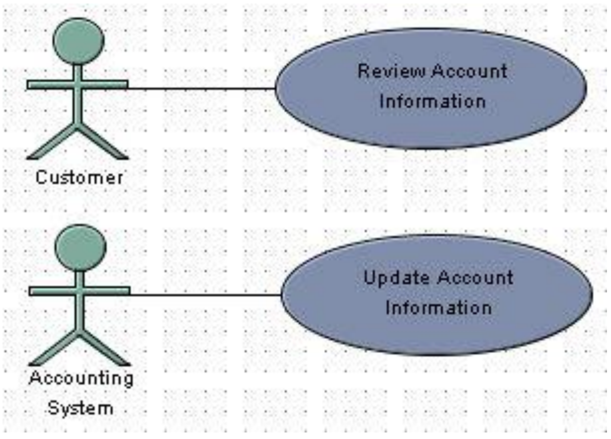
Actor



An **actor** is anything outside the system that interacts with the system to complete a task. It could be a user or another system. The actor "uses" the use case to complete a task. *System Administrator, Credit Authentication System, Accounting System* and *Web Client* are all examples of actors. Often, it is useful to look at the set of use cases that an actor has access to -- this defines the actor's overall role in the system.

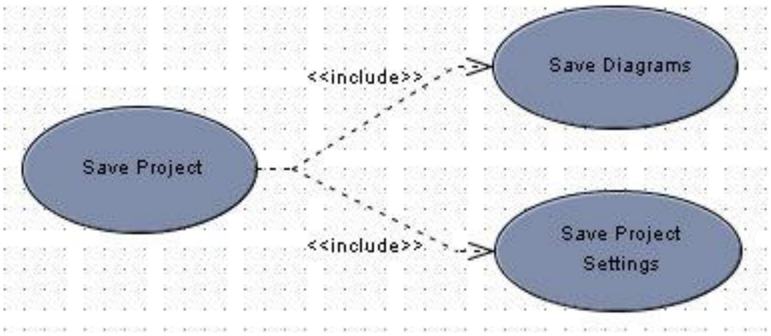
Association:

The **association** is the link that is drawn between and actor and a use case. It indicates which actors interact with the system to complete the various tasks.



Includes:

Use the **includes** link to show that one use case includes the task described by another use case. For example, saving a Visual Case project includes saving the diagrams and saving the project settings. Sometimes the word "Uses" is used instead of "Includes"



Generalization:

The **generalization** link is an informal way of showing that one use case is similar to another use case, but with a little bit of extra functionality. One use case inherits the functionality represented by another use case and adds some additional behavior to it.

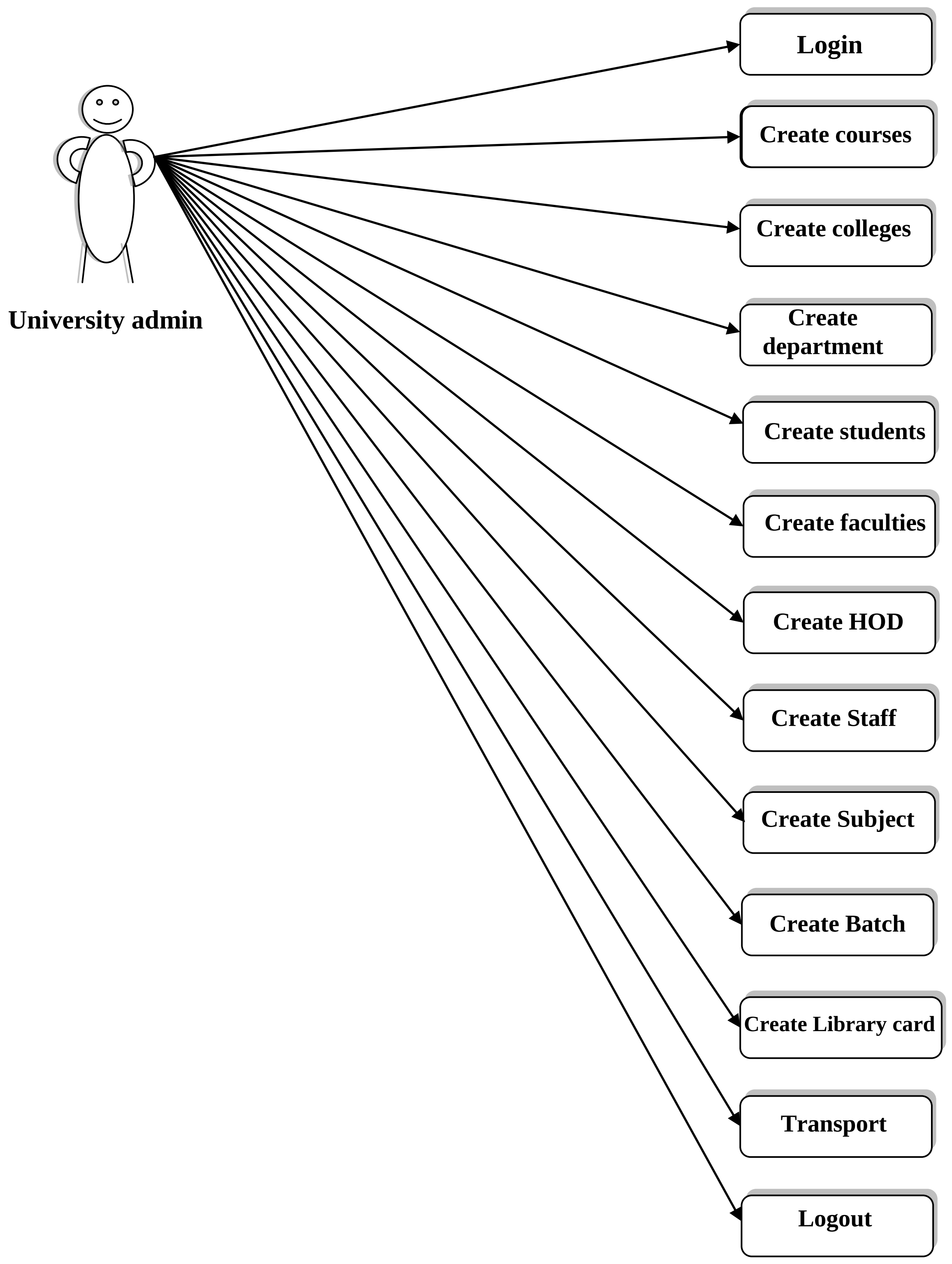
Extends:

The **extends** link is used to show that one use case extends the task described by another use case. It's very similar to generalization, but is much more formalized.

The use case that is extended is always referred to as the **base use case** and has one or more defined **extension points**. The extension points show exactly where extending use cases are allowed to add functionality. The extending use case doesn't have to add functionality at all of the base use case's extension points. The extension link indicates which extension points are being used.

8.14) USE CASE DIAGRAMS

Use case diagram for University Administrator



Use case name	Create courses
Participating actors	administration
Flow of Events	Administrator creates courses and update courses
Entry condition	Admin enter into the system with his own id and pass words
Exit condition	Success fully creates course
Quality Requirements	Successful course creation

Use case name	Create colleges
Participating actors	administration
Flow of Events	Administrator creates colleges, update colleges, delete colleges
Entry condition	Admin enter into the system with his own id and pass words
Exit condition	Success fully creates college
Quality Requirements	Successful college creation

Use case name	Create batches
Participating actors	administrator
Flow of Events	College admin creates batches , delete or update batches
Entry condition	College admin enter into the system with his own id and pass words
Exit condition	Successful batch creation
Quality Requirements	Batch confliction should not occur

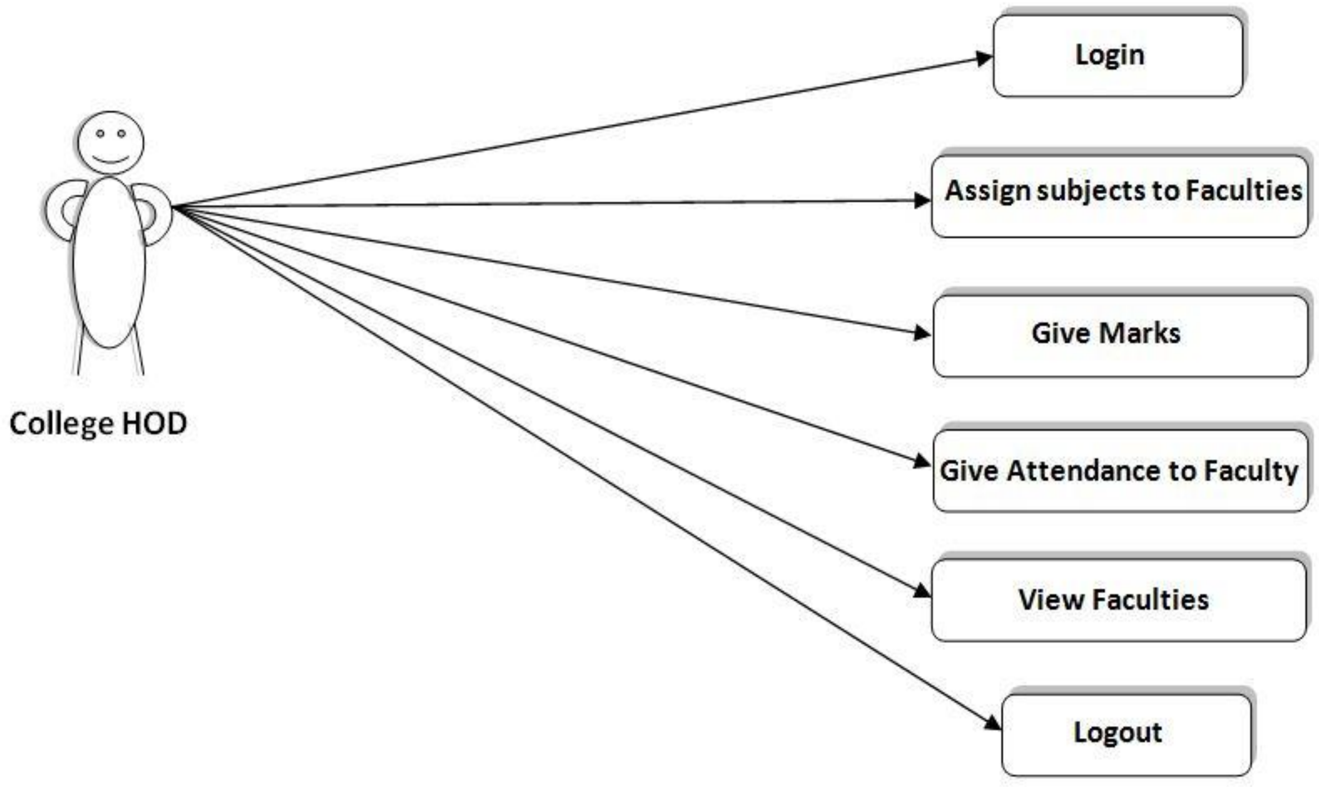
Use case name	Create departments
Participating actors	administrator
Flow of Events	College admin creates departments
Entry condition	College admin enter into with college id and password
Exit condition	Successful department creation
Quality Requirements	Department confliction should not occur

Use case name	Create faculties
Participating actors	administrator
Flow of Events	College admin creates faculties
Entry condition	College admin enters with his id and password
Exit condition	Successful faculty creation
Quality Requirements	Faculty creation

Use case name	Create students
Participating actors	administrator
Flow of Events	College admin creates students or delete student
Entry condition	College admin enters with his id and password
Exit condition	Successful student creation
Quality Requirements	Student creation

Use case name	Reports
Participating actors	University administration
Flow of Events	Administrat creates reports of colleges and courses
Entry condition	Admin enter into the system with his own id and pass words
Exit condition	Success view reports
Quality Requirements	Generation of Reports

Use case diagram for HOD:

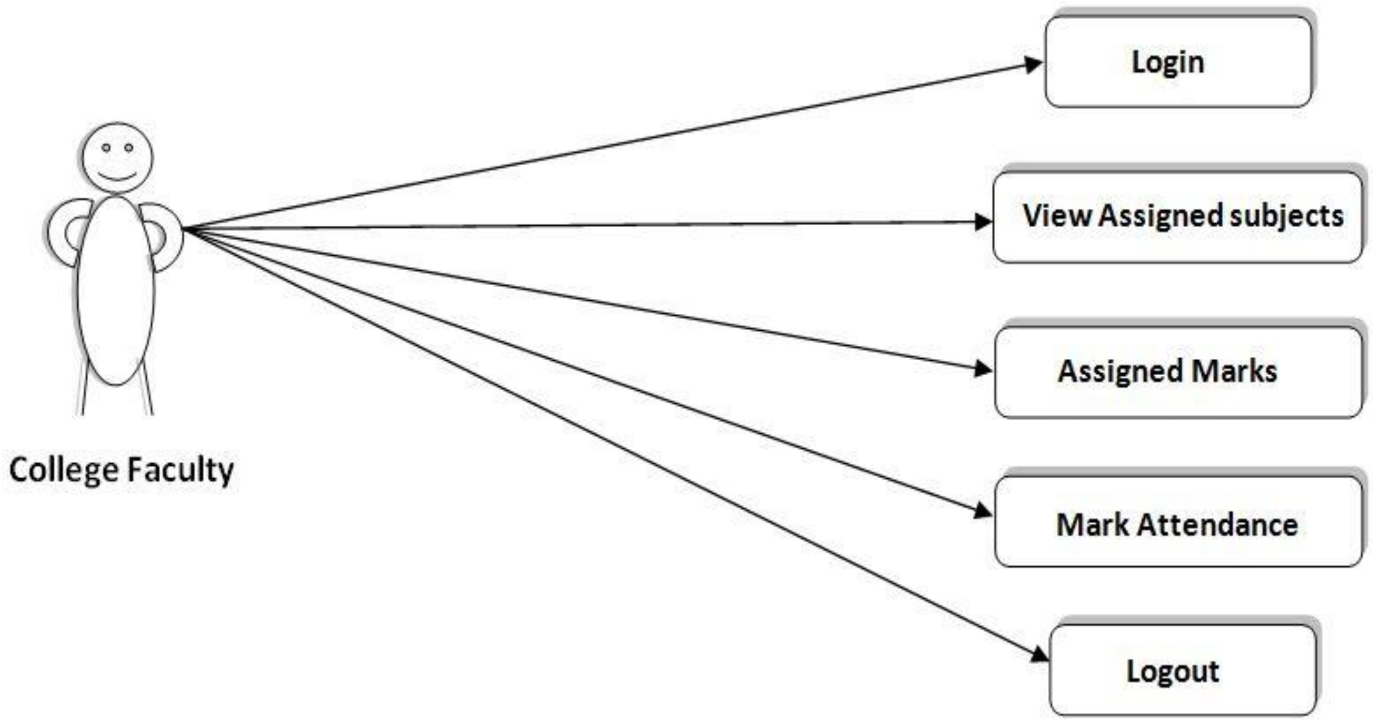


Use case name	assign subjects to faculties
Participating actors	College Hod
Flow of Events	Hod assigns subject to faculties
Entry condition	Hod enters with his collegename and his username and password
Exit condition	After successful assignment
Quality Requirements	Assignment of subjects

Use case name	Give marks
Participating actors	College Hod
Flow of Events	Hod gives marks to students and modify them
Entry condition	Hod enters with his college name and his username and password
Exit condition	After successful giving marks
Quality Requirements	Proper entering of marks

Use case name	Give attendance
Participating actors	College Hod
Flow of Events	Hod gives attendance to students and modify them
Entry condition	Hod enters with his college name and his username and password
Exit condition	After successful giving attendance
Quality Requirements	Proper entering of attendance

Use case diagram for Faculty:

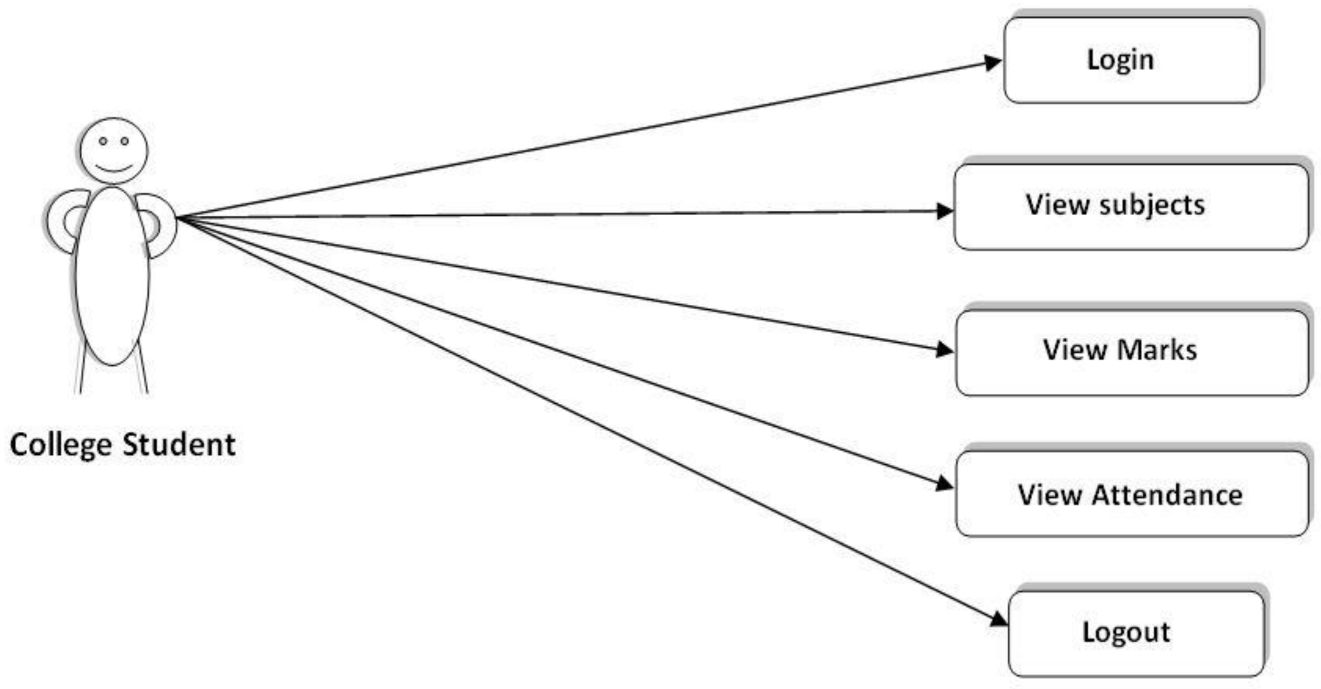


Use case name	view assigned subjects
Participating actors	College faculty
Flow of Events	faculty view assigned subjects
Entry condition	Faculty enters with his collegename and his username and password
Exit condition	After viewing assigned subjects
Quality Requirements	Faculty will view assigned subjects etc

Use case name	Givemarks

Participating actors	College faculty
Flow of Events	Faculty gives marks to students and modify them to particular allocated subject
Entry condition	Faculty enters with his collegename and his username and password
Exit condition	After successful giving marks
Quality Requirements	Proper entering of marks

Use case diagram for student:

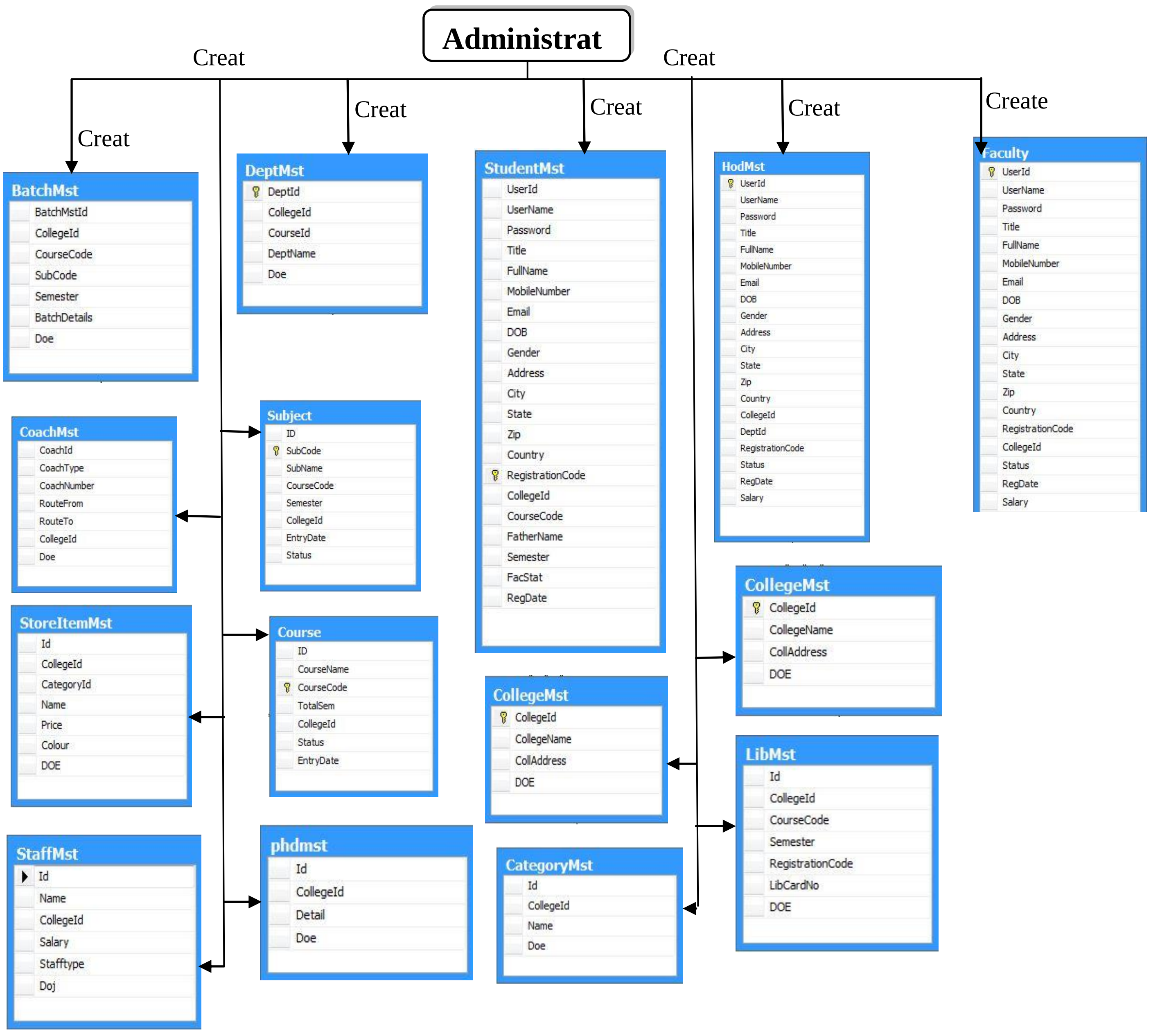


Use case name	View marks
Participating actors	College student
Flow of Events	Student will view his marks
Entry condition	Student enters in to his account with college name and his batch id and rollno
Exit condition	Student after his task is finished
Quality Requirements	Student will view his marks

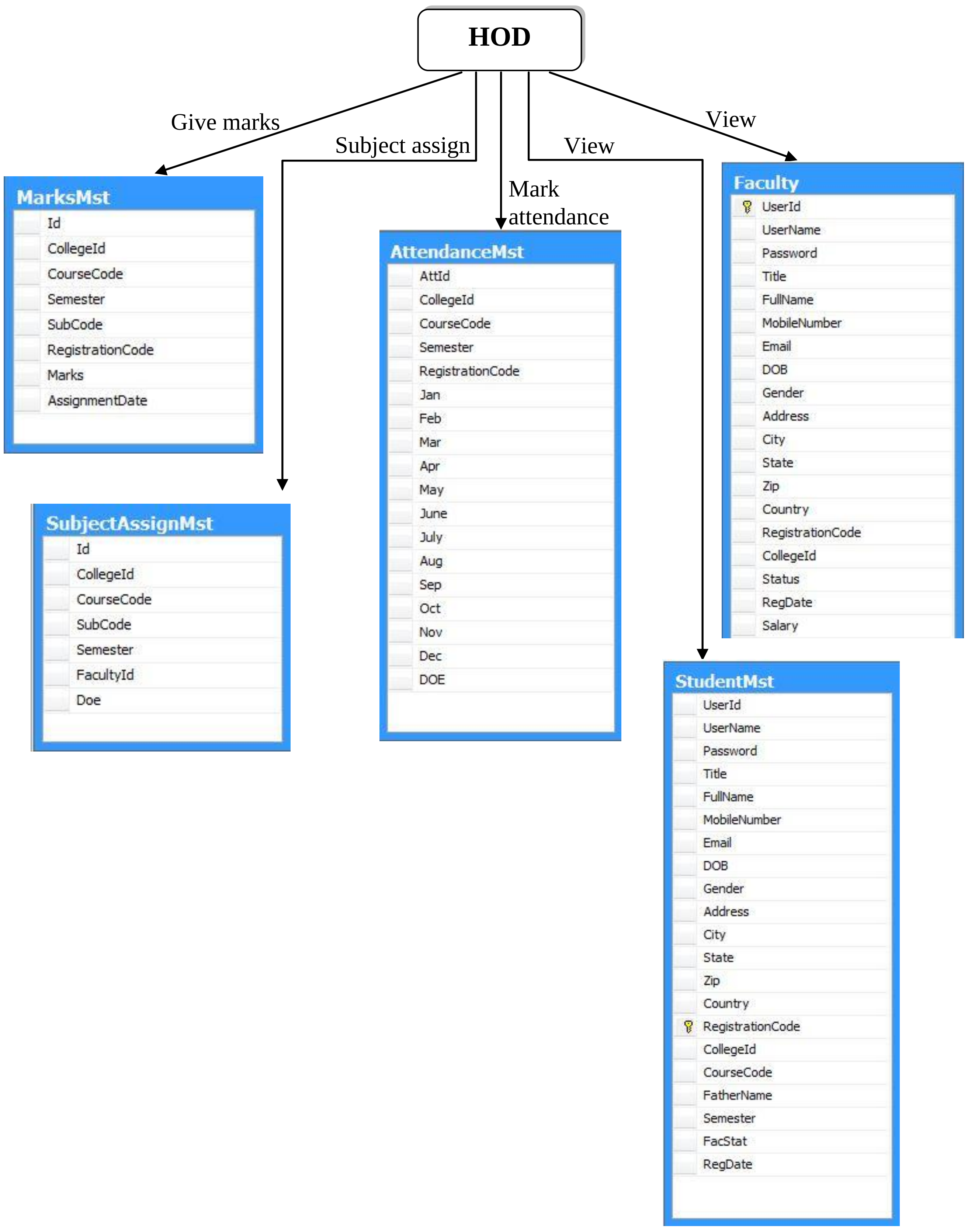
Use case name	View attendance
Participating actors	College student
Flow of Events	Student will view his attendance
Entry condition	Student enters in to his account with college name and his batch id and rollno
Exit condition	Student after his task is finished
Quality Requirements	Proper attendance entering

8.15) CLASS DIAGRAMS

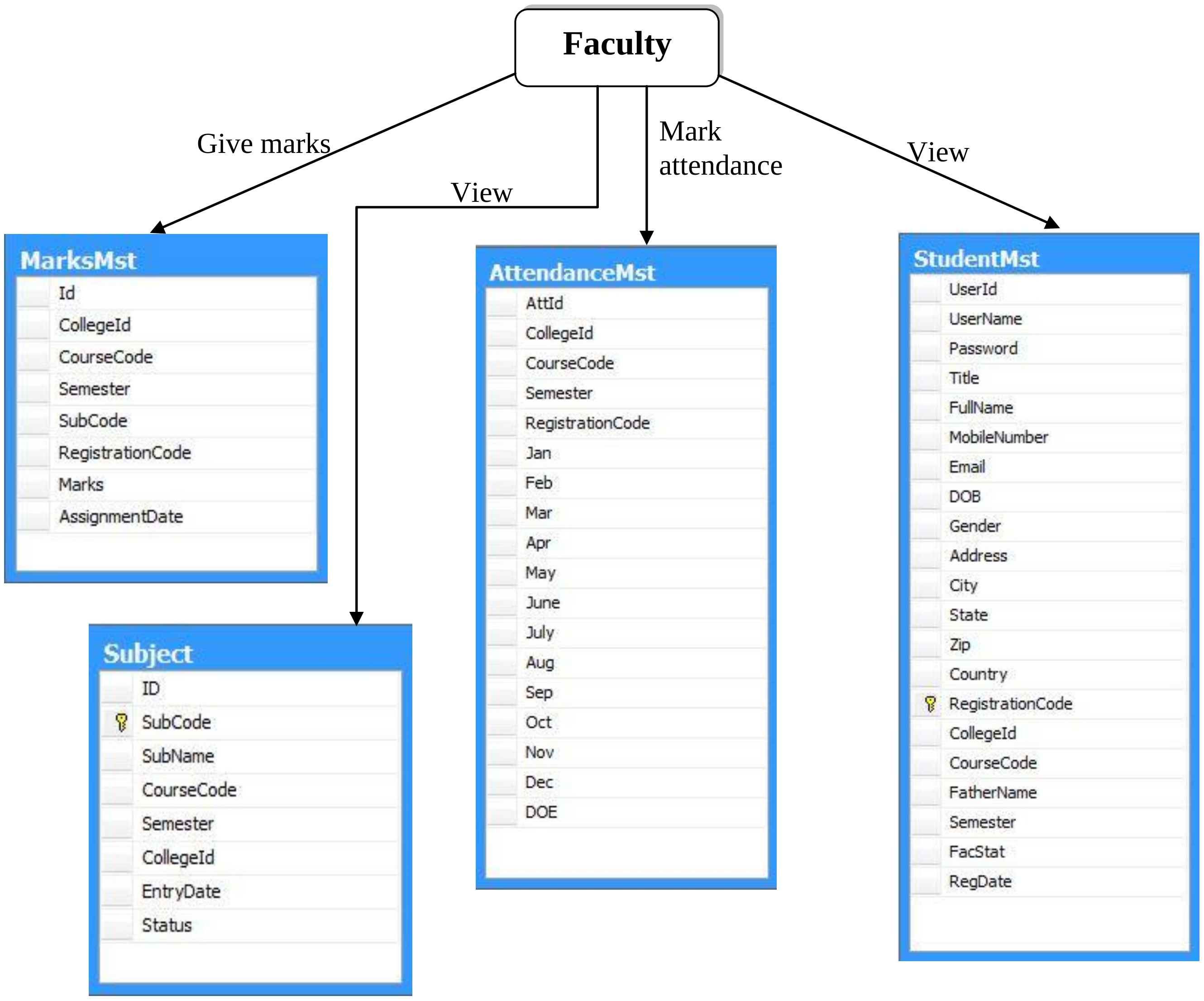
Class Diagram for Administrator:



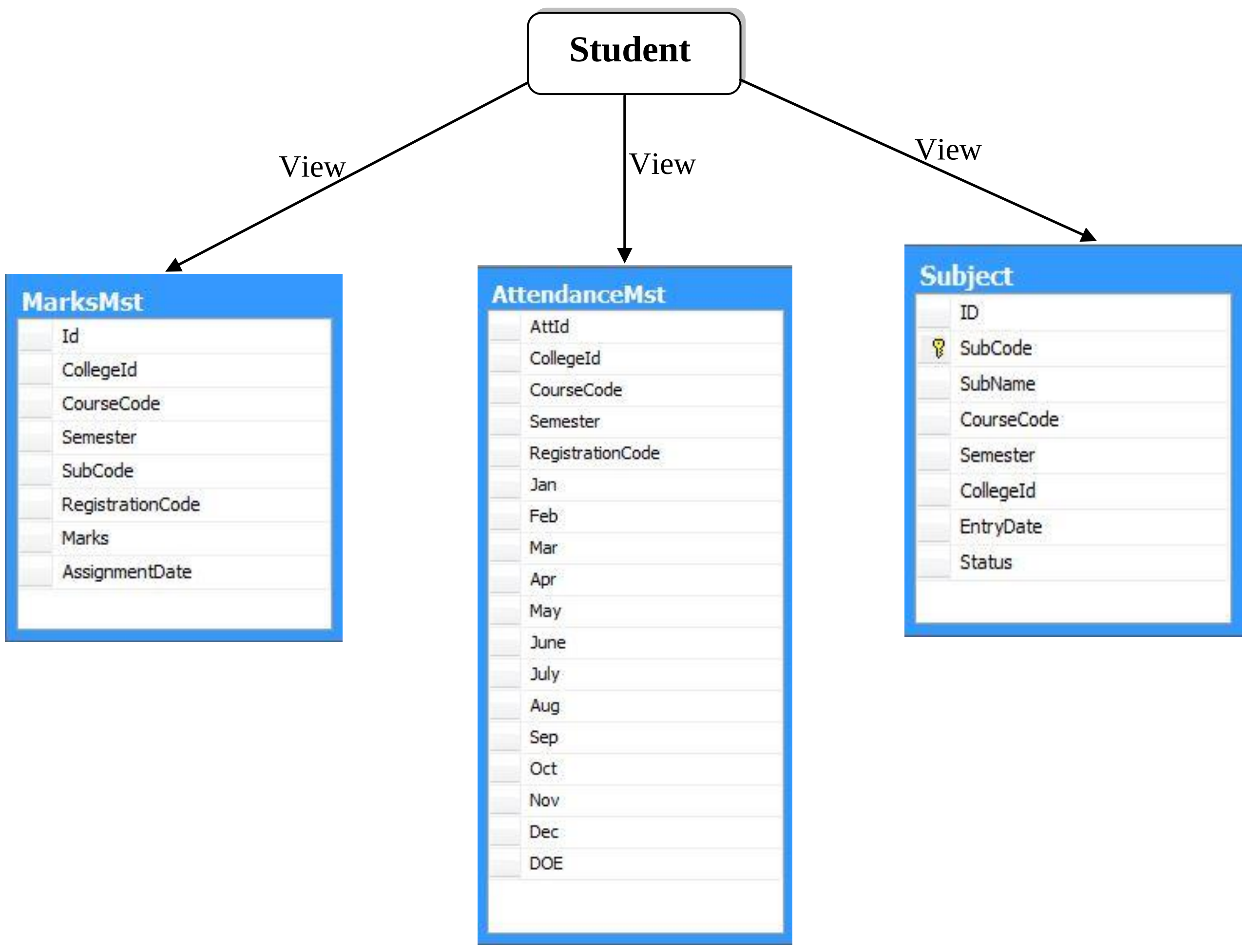
Class Diagram for HOD:



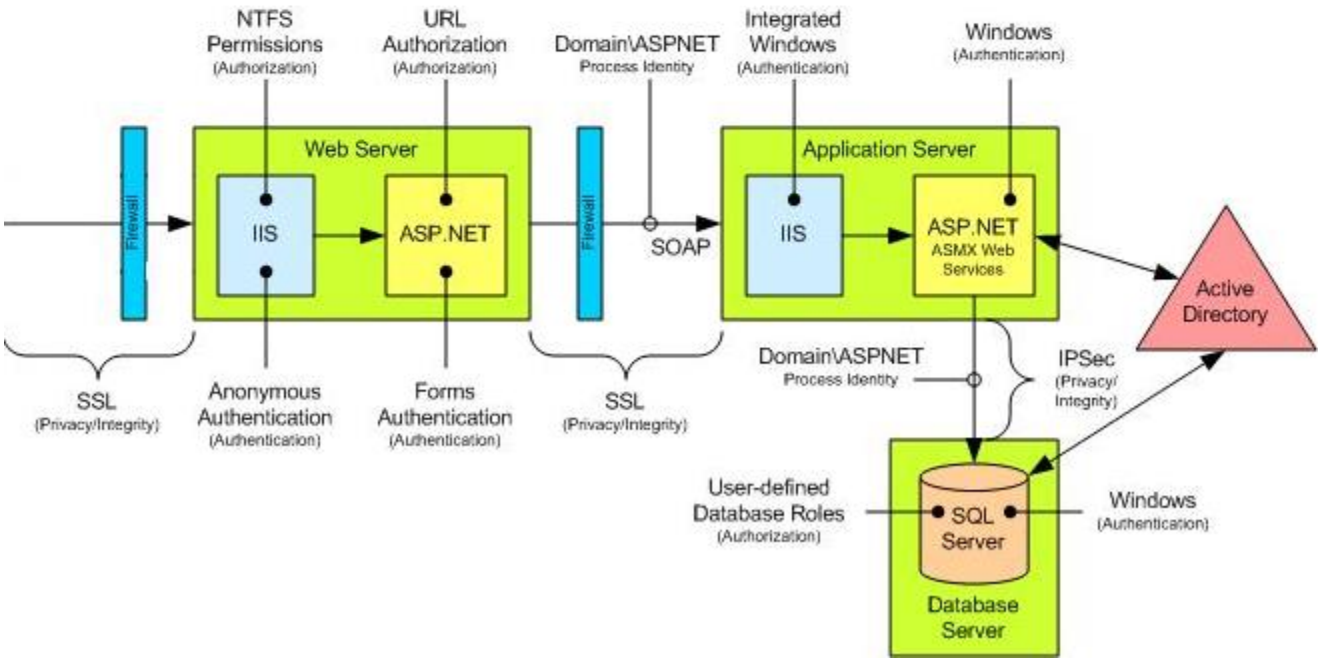
Class Diagram for Faculty:



Class Diagram for Student:



8.16) Deployment Diagram


NTFS Permissions (Authorization)
URL Authorization (Authorization)
Domain\ASPNET Process Identity
Integrated Windows (Authentication)
Windows (Authentication)
Firewall
Web Server
IIS
ASP.NET
Firewall
SOAP
Application Server
IIS
ASP.NET ASMX Web Services
Active Directory
SSL (Privacy/Integrity)
Anonymous Authentication (Authentication)
Forms Authentication (Authentication)
SSL (Privacy/Integrity)
Domain\ASPNET Process Identity
IPSec (Privacy/ Integrity)
User-defined Database Roles (Authorization)
SQL Server
Database Server
Windows (Authentication)

8.17) Database Description:

After carefully understanding the requirements the entire data storage requirements are divided into tables. The tables are normalized to avoid any anomalies during the course of data entry.

8.18) NORMALIZATION:

It is a process of converting a relation to a standard form. The process is used to handle the problems that can arise due to data redundancy i.e. repetition of data in the database, maintain data integrity as well as handling problems that can arise due to insertion, updation, deletion anomalies.

Decomposing is the process of splitting relations into multiple relations to eliminate anomalies and maintain anomalies and maintain data integrity. To do this we use normal forms or rules for structuring relation.

Insertion anomaly: Inability to add data to the database due to absence of other data.

Deletion anomaly: Unintended loss of data due to deletion of other data.

Update anomaly: Data inconsistency resulting from data redundancy and partial update.

Normal Forms: These are the rules for structuring relations that eliminate anomalies.

- **First Normal Form:**

A relation is said to be in first normal form if the values in the relation are atomic for every attribute in the relation. By this we mean simply that no attribute value can be a set of values or, as it is sometimes expressed, a repeating group.

- **Second Normal Form**:

A relation is said to be in second Normal form is it is in first normal form and it should satisfy any one of the following rules.

- Primary key is a not a composite primary key
- No non key attributes are present
- Every non key attribute is fully functionally dependent on full set of primary key.

- **Third Normal Form**:

A relation is said to be in third normal form if their exits no transitive dependencies.

Transitive Dependency: If two non key attributes depend on each other as well as on the primary key then they are said to be transitively dependent.

8.19) Data Dictionary

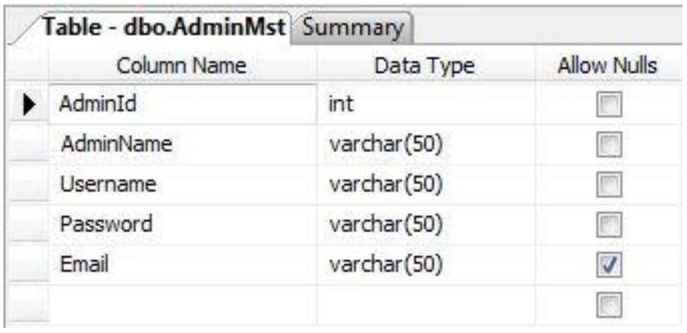


Table - dbo.AdminMst | Summary

Column Name	Data Type	Allow Nulls
AdminId	int	☐
AdminName	varchar(50)	☐
Username	varchar(50)	☐
Password	varchar(50)	☐
Email	varchar(50)	✓
		☐

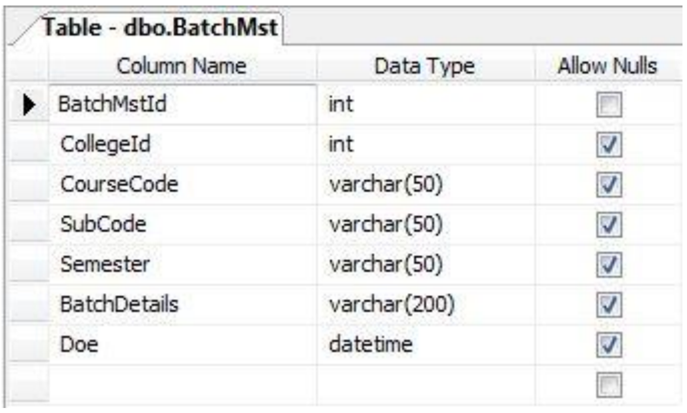


Table - dbo.BatchMst

Column Name	Data Type	Allow Nulls
BatchMstId	int	☐
CollegeId	int	✓
CourseCode	varchar(50)	✓
SubCode	varchar(50)	✓
Semester	varchar(50)	✓
BatchDetails	varchar(200)	✓
Doe	datetime	✓
		☐

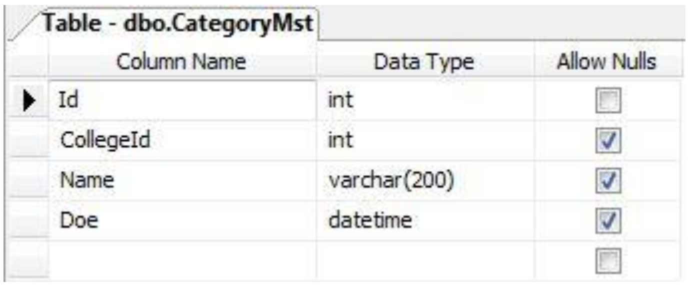


Table - dbo.CategoryMst

Column Name	Data Type	Allow Nulls
Id	int	☐
CollegeId	int	✓
Name	varchar(200)	✓
Doe	datetime	✓
		☐

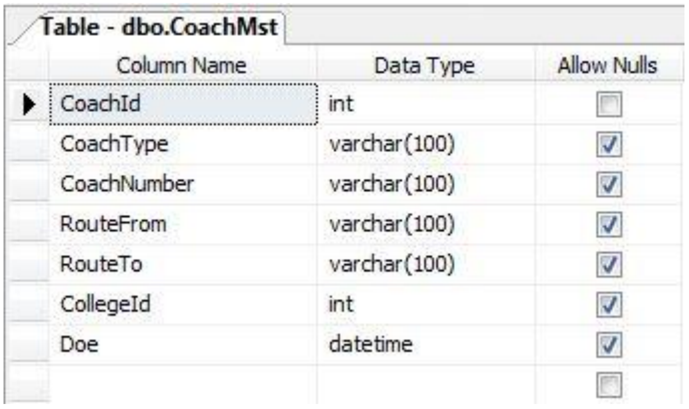


Table - dbo.CoachMst

Column Name	Data Type	Allow Nulls
CoachId	int	☐
CoachType	varchar(100)	✓
CoachNumber	varchar(100)	✓
RouteFrom	varchar(100)	✓
RouteTo	varchar(100)	✓
CollegeId	int	✓
Doe	datetime	✓
		☐

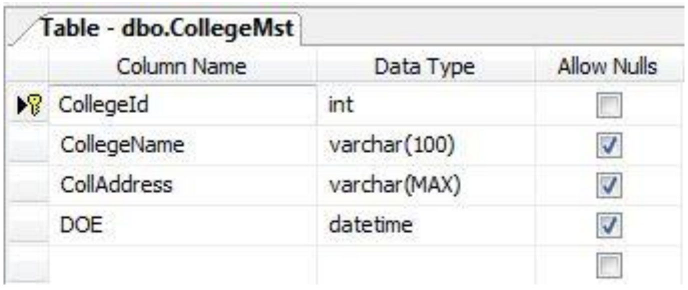


Table - dbo.CollegeMst

Column Name	Data Type	Allow Nulls
CollegeId	int	☐
CollegeName	varchar(100)	✓
CollAddress	varchar(MAX)	✓
DOE	datetime	✓
		☐

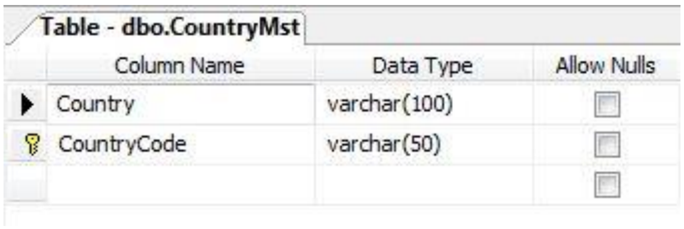


Table - dbo.CountryMst

Column Name	Data Type	Allow Nulls
Country	varchar(100)	☐
CountryCode	varchar(50)	☐
		☐

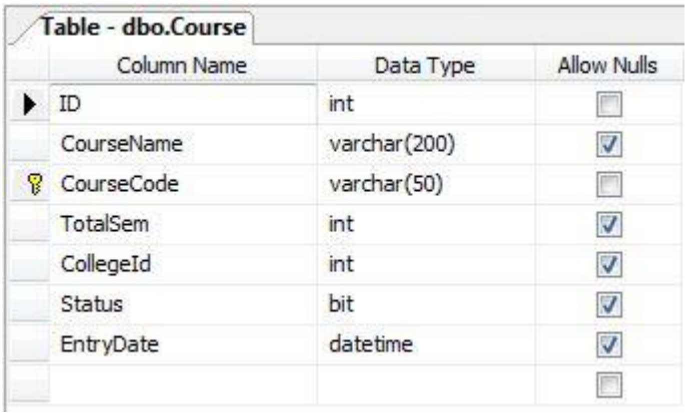


Table - dbo.Course

Column Name	Data Type	Allow Nulls
ID	int	☐
CourseName	varchar(200)	✓
CourseCode	varchar(50)	☐
TotalSem	int	✓
CollegeId	int	✓
Status	bit	✓
EntryDate	datetime	✓
		☐

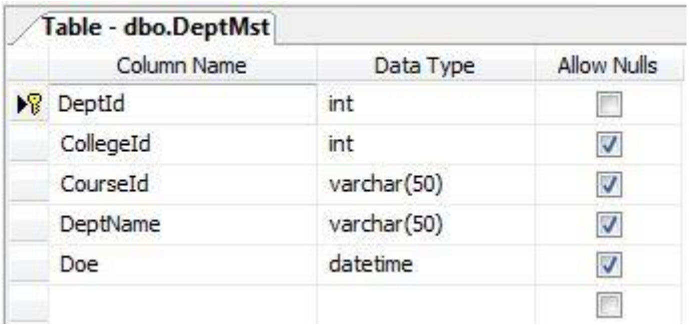


Table - dbo.DeptMst

Column Name	Data Type	Allow Nulls
DeptId	int	☐
CollegeId	int	✓
CourseId	varchar(50)	✓
DeptName	varchar(50)	✓
Doe	datetime	✓
		☐

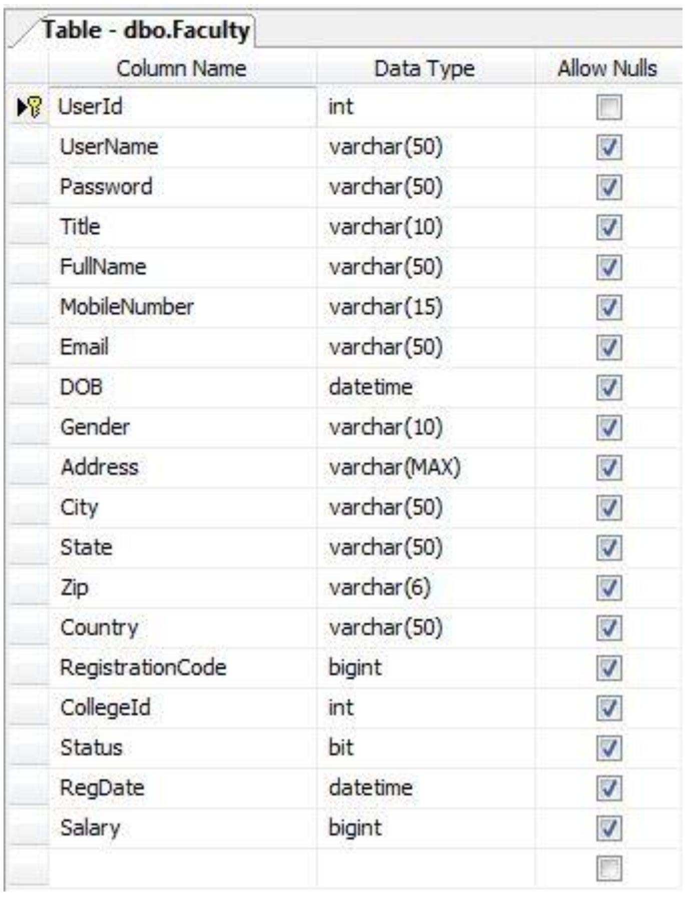


Table - dbo.Faculty

Column Name	Data Type	Allow Nulls
UserId	int	
UserName	varchar(50)	✓
Password	varchar(50)	✓
Title	varchar(10)	✓
FullName	varchar(50)	✓
MobileNumber	varchar(15)	✓
Email	varchar(50)	✓
DOB	datetime	✓
Gender	varchar(10)	✓
Address	varchar(MAX)	✓
City	varchar(50)	✓
State	varchar(50)	✓
Zip	varchar(6)	✓
Country	varchar(50)	✓
RegistrationCode	bigint	✓
CollegeId	int	✓
Status	bit	✓
RegDate	datetime	✓
Salary	bigint	✓

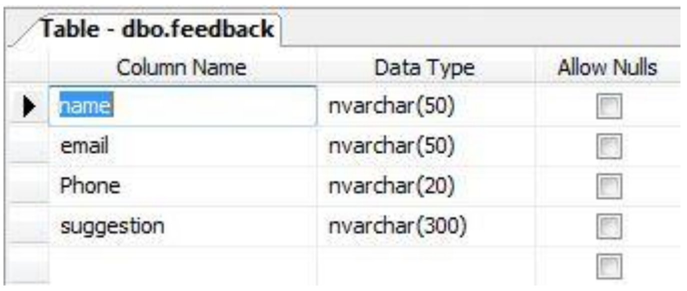


Table - dbo.feedback

Column Name	Data Type	Allow Nulls
name	nvarchar(50)	
email	nvarchar(50)	
Phone	nvarchar(20)	
suggestion	nvarchar(300)	

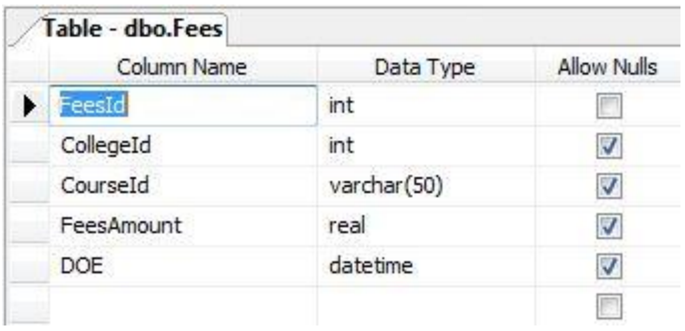


Table - dbo.Fees

Column Name	Data Type	Allow Nulls
FeesId	int	
CollegeId	int	✓
CourseId	varchar(50)	✓
FeesAmount	real	✓
DOE	datetime	✓

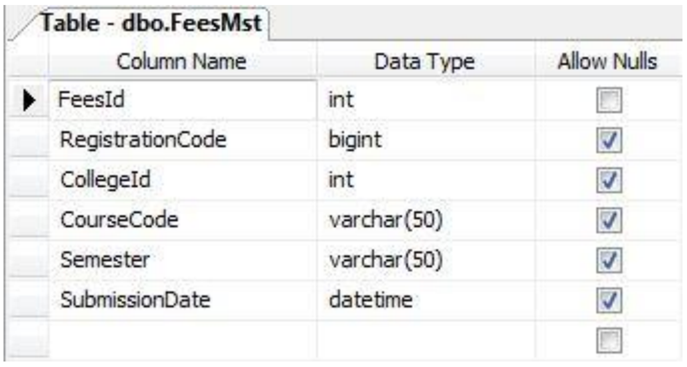


Table - dbo.FeesMst

Column Name	Data Type	Allow Nulls
FeesId	int	☐
RegistrationCode	bigint	✓
CollegeId	int	✓
CourseCode	varchar(50)	✓
Semester	varchar(50)	✓
SubmissionDate	datetime	✓
		☐

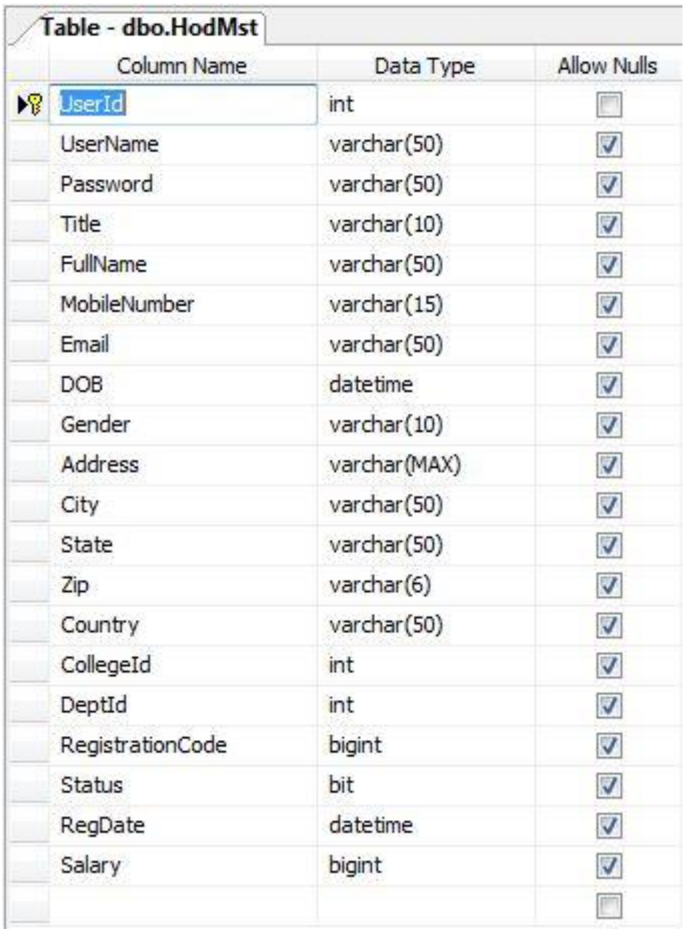


Table - dbo.HodMst

Column Name	Data Type	Allow Nulls
UserId	int	☐
UserName	varchar(50)	✓
Password	varchar(50)	✓
Title	varchar(10)	✓
FullName	varchar(50)	✓
MobileNumber	varchar(15)	✓
Email	varchar(50)	✓
DOB	datetime	✓
Gender	varchar(10)	✓
Address	varchar(MAX)	✓
City	varchar(50)	✓
State	varchar(50)	✓
Zip	varchar(6)	✓
Country	varchar(50)	✓
CollegeId	int	✓
DeptId	int	✓
RegistrationCode	bigint	✓
Status	bit	✓
RegDate	datetime	✓
Salary	bigint	✓
		☐

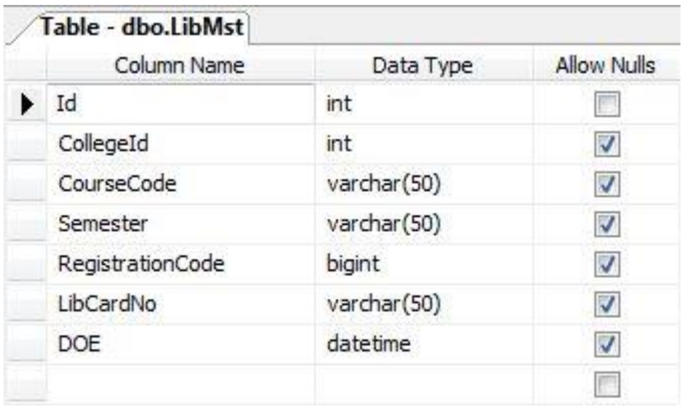


Table - dbo.LibMst

	Column Name	Data Type	Allow Nulls
▶	Id	int	☐
	CollegeId	int	✓
	CourseCode	varchar(50)	✓
	Semester	varchar(50)	✓
	RegistrationCode	bigint	✓
	LibCardNo	varchar(50)	✓
	DOE	datetime	✓
			☐

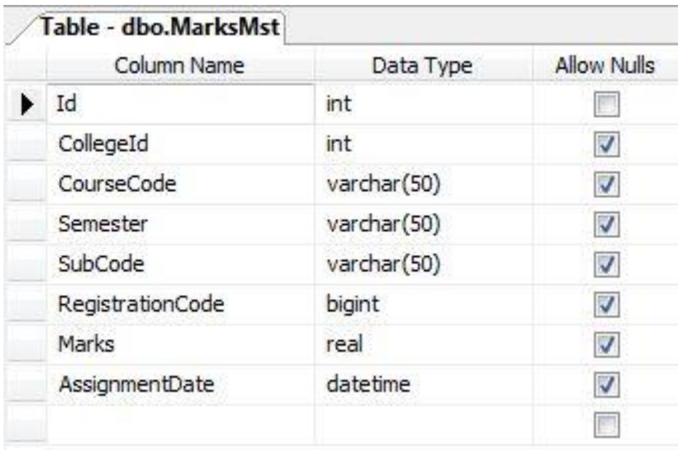


Table - dbo.MarksMst

	Column Name	Data Type	Allow Nulls
▶	Id	int	☐
	CollegeId	int	✓
	CourseCode	varchar(50)	✓
	Semester	varchar(50)	✓
	SubCode	varchar(50)	✓
	RegistrationCode	bigint	✓
	Marks	real	✓
	AssignmentDate	datetime	✓
			☐

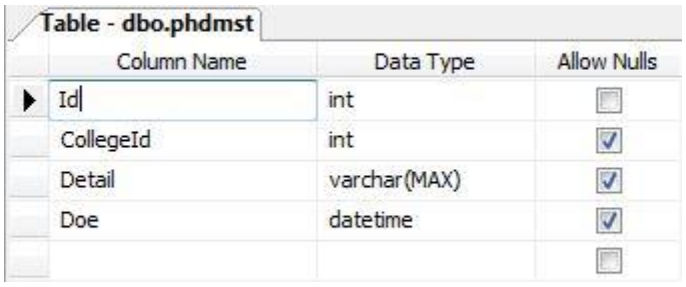


Table - dbo.phdmst

	Column Name	Data Type	Allow Nulls
▶	Id	int	☐
	CollegeId	int	✓
	Detail	varchar(MAX)	✓
	Doe	datetime	✓
			☐

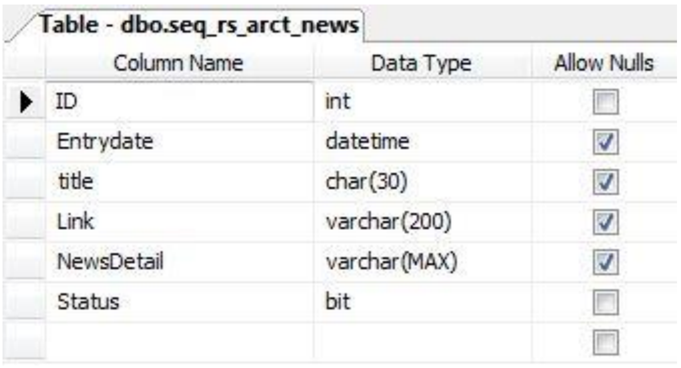


Table - dbo.seq_rs_arct_news

	Column Name	Data Type	Allow Nulls
▶	ID	int	☐
	Entrydate	datetime	✓
	title	char(30)	✓
	Link	varchar(200)	✓
	NewsDetail	varchar(MAX)	✓
	Status	bit	☐
			☐

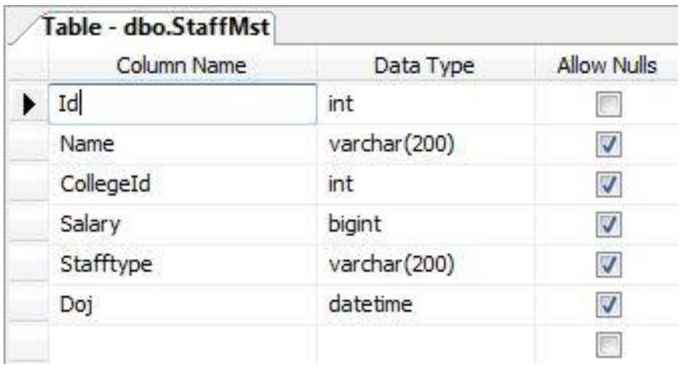


Table - dbo.StaffMst

Column Name	Data Type	Allow Nulls
Id	int	☐
Name	varchar(200)	✓
CollegeId	int	✓
Salary	bigint	✓
Stafftype	varchar(200)	✓
Doj	datetime	✓
		☐

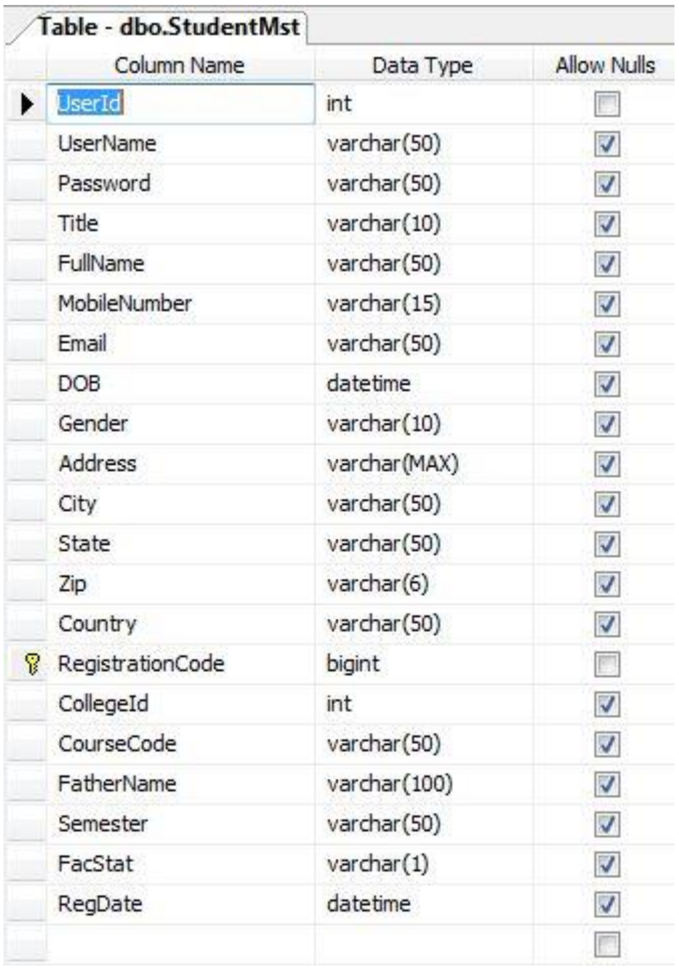


Table - dbo.StudentMst

Column Name	Data Type	Allow Nulls
UserId	int	☐
UserName	varchar(50)	✓
Password	varchar(50)	✓
Title	varchar(10)	✓
FullName	varchar(50)	✓
MobileNumber	varchar(15)	✓
Email	varchar(50)	✓
DOB	datetime	✓
Gender	varchar(10)	✓
Address	varchar(MAX)	✓
City	varchar(50)	✓
State	varchar(50)	✓
Zip	varchar(6)	✓
Country	varchar(50)	✓
RegistrationCode	bigint	☐
CollegeId	int	✓
CourseCode	varchar(50)	✓
FatherName	varchar(100)	✓
Semester	varchar(50)	✓
FacStat	varchar(1)	✓
RegDate	datetime	✓
		☐

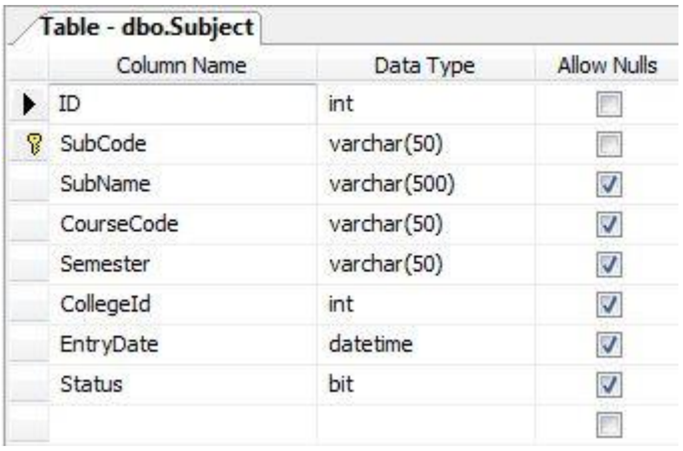


Table - dbo.Subject

Column Name	Data Type	Allow Nulls
ID	int	☐
SubCode	varchar(50)	☐
SubName	varchar(500)	✓
CourseCode	varchar(50)	✓
Semester	varchar(50)	✓
CollegeId	int	✓
EntryDate	datetime	✓
Status	bit	✓
		☐

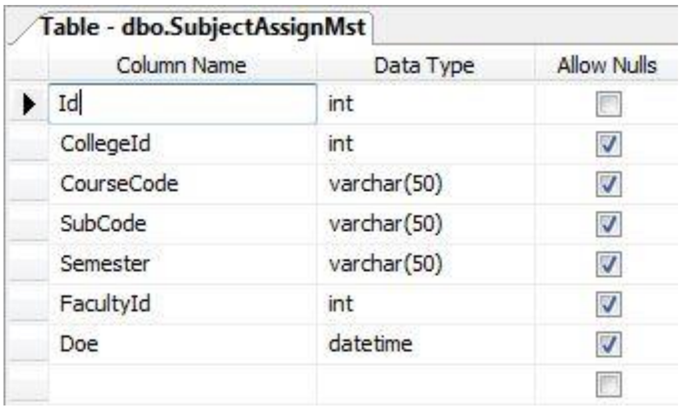


Table - dbo.SubjectAssignMst

Column Name	Data Type	Allow Nulls
Id	int	☐
CollegeId	int	✓
CourseCode	varchar(50)	✓
SubCode	varchar(50)	✓
Semester	varchar(50)	✓
FacultyId	int	✓
Doe	datetime	✓
		☐

8.20) System implementation

The overall goal of the work is to develop a flexible index framework that can be tuned to achieve effective static index selection and online index selection , the goal is to develop a system that can be recommended an evolving set of indexes for incoming queries overtime such that the benefit of index set changes out weight the cost of making those changes. These differentiate between low-cost index set changes and higher cost index set changes.

The benefit of using a potential index over a set of queries, it is necessary to estimate the cost of executing the queries with and without index. The cost model is embedded in to the queries should be answered using a sequential scan or using an existing index. Instead of using the query optimizer to estimate the query cost.

This implementation technique differs from existing tools in determine potential set of indexes to evaluate and quantization basic technique that estimate every costs. All the common goal indexes work in design time. The DBA decide when to run the wizard and over with which workload.

8.21) Basic Design Approach

The goal of the design phase is to transform the requirements specified in the SRS document into a structure that is suitable for the implementation in some programming language. Design involves preparing a detailed analysis of different functions to be supported by the system and identification of data flow among different functions.

Stages of system design are:

1. Conceptual Design
2. DataBase Design
3. Functional Design

1. Conceptual Design:

The conceptual structure of a database is called as schema. A schema can be regarded as a blueprint that portrays, both, kind of data used in building the database and logical relationships.

2. Database Design:

A database is a stored collection of interrelated data, organized on the basis of relationship in the data rather than the convenience of storage structures. It enables sharing of data among various users as and when required. Database Management System is software that provides more flexibility in the storage and retrieval of data and productive information.

Primary key: Collection of one or more fields whose value uniquely identifies a record in a table. This key is used to identify each occurrence of an entity. It never has a null value.

Foreign key: One or more table fields that refer to the primary key field/fields of another table. A foreign key is used to create links with another entity.

3. Functional Design:

Input describes the information to be supplied to this function either by the user on the screen (or) from any data store. Processing describes the operations to be carried out by the function. Output describes the information obtained from the function (or) the action carried out by the function

9

Testing

9.1) INTRODUCTION

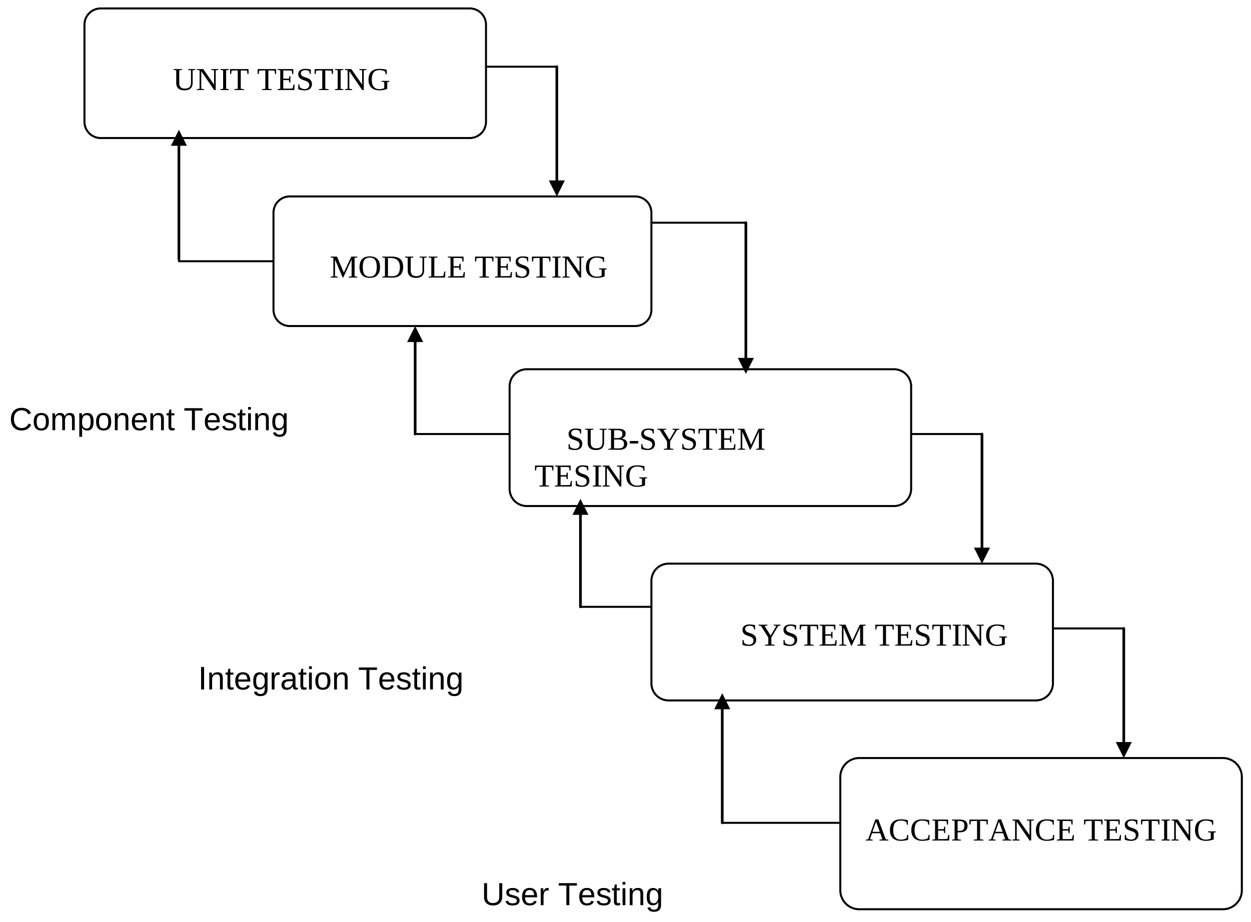
Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. In fact, testing is the one step in the software engineering process that could be viewed as destructive rather than constructive.

A strategy for software testing integrates software test case design methods into a well-planned series of steps that result in the successful construction of software. Testing is the set of activities that can be planned in advance and conducted systematically. The underlying motivation of program testing is to affirm software quality with methods that can economically and effectively apply to both strategic to both large and small-scale systems.

9.2) STRATEGIC APPROACH TO SOFTWARE TESTING

The software engineering process can be viewed as a spiral. Initially system engineering defines the role of software and leads to software requirement analysis where the information domain, functions, behavior, performance, constraints and validation criteria for software are established. Moving inward along the spiral, we come to design and finally to coding. To develop computer software we spiral in along streamlines that decrease the level of abstraction on each turn.

A strategy for software testing may also be viewed in the context of the spiral. Unit testing begins at the vertex of the spiral and concentrates on each unit of the software as implemented in source code. Testing progress by moving outward along the spiral to integration testing, where the focus is on the design and the construction of the software architecture. Talking another turn on outward on the spiral we encounter validation testing where requirements established as part of software requirements analysis are validated against the software that has been constructed. Finally we arrive at system testing, where the software and other system elements are tested as a whole.



9.3) UNIT TESTING

Unit testing focuses verification effort on the smallest unit of software design, the module. The unit testing we have is white box oriented and some modules the steps are conducted in parallel.

1. WHITE BOX TESTING

This type of testing ensures that

- All independent paths have been exercised at least once.
- All logical decisions have been exercised on their true and false sides.
- All loops are executed at their boundaries and within their operational bounds.
- All internal data structures have been exercised to assure their validity.

To follow the concept of white box testing we have tested each form .we have created independently to verify that Data flow is correct, All conditions are exercised to check their validity, All loops are executed on their boundaries.

2. BASIC PATH TESTING

Established technique of flow graph with Cyclomatic complexity was used to derive test cases for all the functions. The main steps in deriving test cases were:

Use the design of the code and draw correspondent flow graph.

Determine the Cyclomatic complexity of resultant flow graph, using formula:

$V(G)=E-N+2$ or

$V(G)=P+1$ or

$V(G)$=Number Of Regions

Where $V(G)$ is Cyclomatic complexity,

E is the number of edges,

N is the number of flow graph nodes,

P is the number of predicate nodes.

Determine the basis of set of linearly independent paths.

3. CONDITIONAL TESTING

In this part of the testing each of the conditions were tested to both true and false aspects. And all the resulting paths were tested. So that each path that may be generate on particular condition is traced to uncover any possible errors.

4. DATA FLOW TESTING

This type of testing selects the path of the program according to the location of definition and use of variables. This kind of testing was used only when some local variable were declared. The *definition-use chain* method was used in this type of testing. These were particularly useful in nested statements.

5. LOOP TESTING

In this type of testing all the loops are tested to all the limits possible. The following exercise was adopted for all loops:

All the loops were tested at their limits, just above them and just below them.

All the loops were skipped at least once.

For nested loops test the inner most loop first and then work outwards.

For concatenated loops the values of dependent loops were set with the help of connected loop.

Unstructured loops were resolved into nested loops or concatenated loops and tested as above.

Each unit has been separately tested by the development team itself and all the input have been validated.

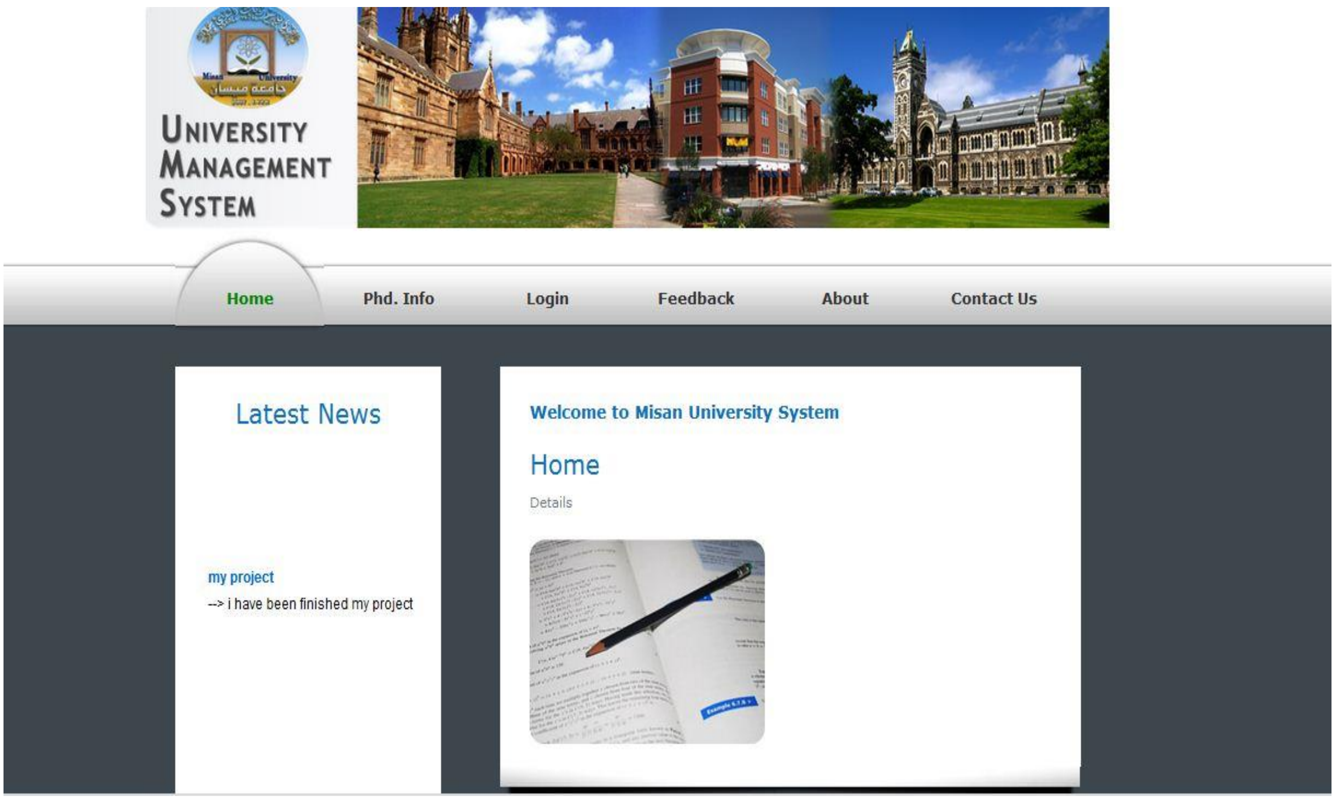
9.4) **TEST CASES**

SNO	TEST CASE ID	DESCRIPTION	EXPECTED VALUE	OBSERVED VALUE	RESULT
1	TC1	PASSWORD NOT GIVEN IN ADMINISTRATION LOGIN FORM	" Invalid User Name or Password"	SAME AS EXPECTED VALUE	SUCCESS
2	TC2	PASSWORD NOT GIVEEN IN HOD LOGIN FORM	" Invalid User Name or Password"	SAME AS EXPECTED VALUE	SUCCESS
3	TC3	PASSWORD IS NOT GIVEN IN FACULTY LOGIN FORM	" Invalid User Name or Password"	SAME AS EXPECTED VALUE	SUCCESS
4	TC4	PASSWORD IS NOT GIVEN IN STUDENT LOGIN FORM	" Invalid User Name or Password"	SAME AS EXPECTED VALUE	SUCCESS

10

Screenshots

UMS Home Page



Welcome to Misan University System

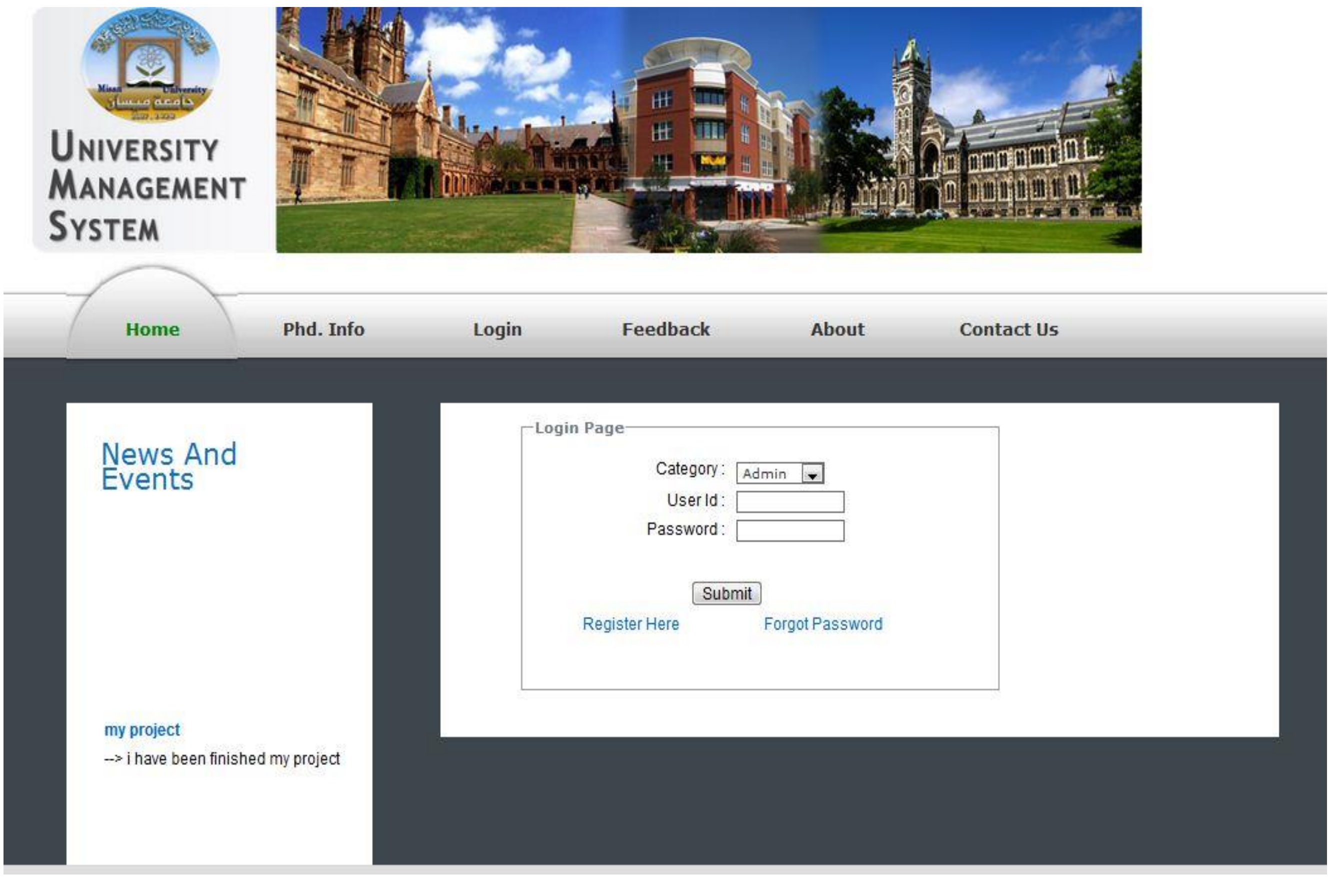
Home

Details

my project

--> i have been finished my project

Login page



News And Events

my project

--> i have been finished my project

Login Page

Category : Admin

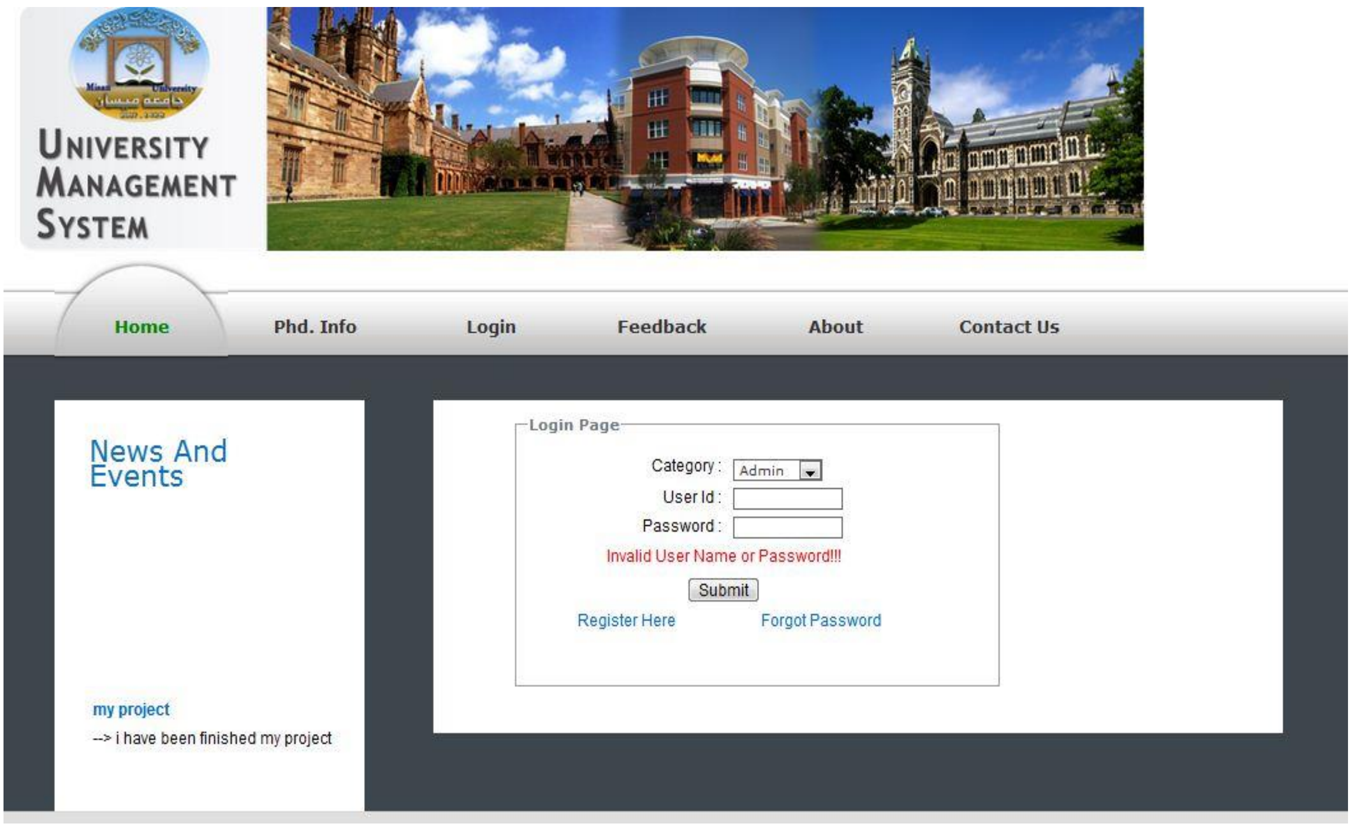
User Id :

Password :

Register Here

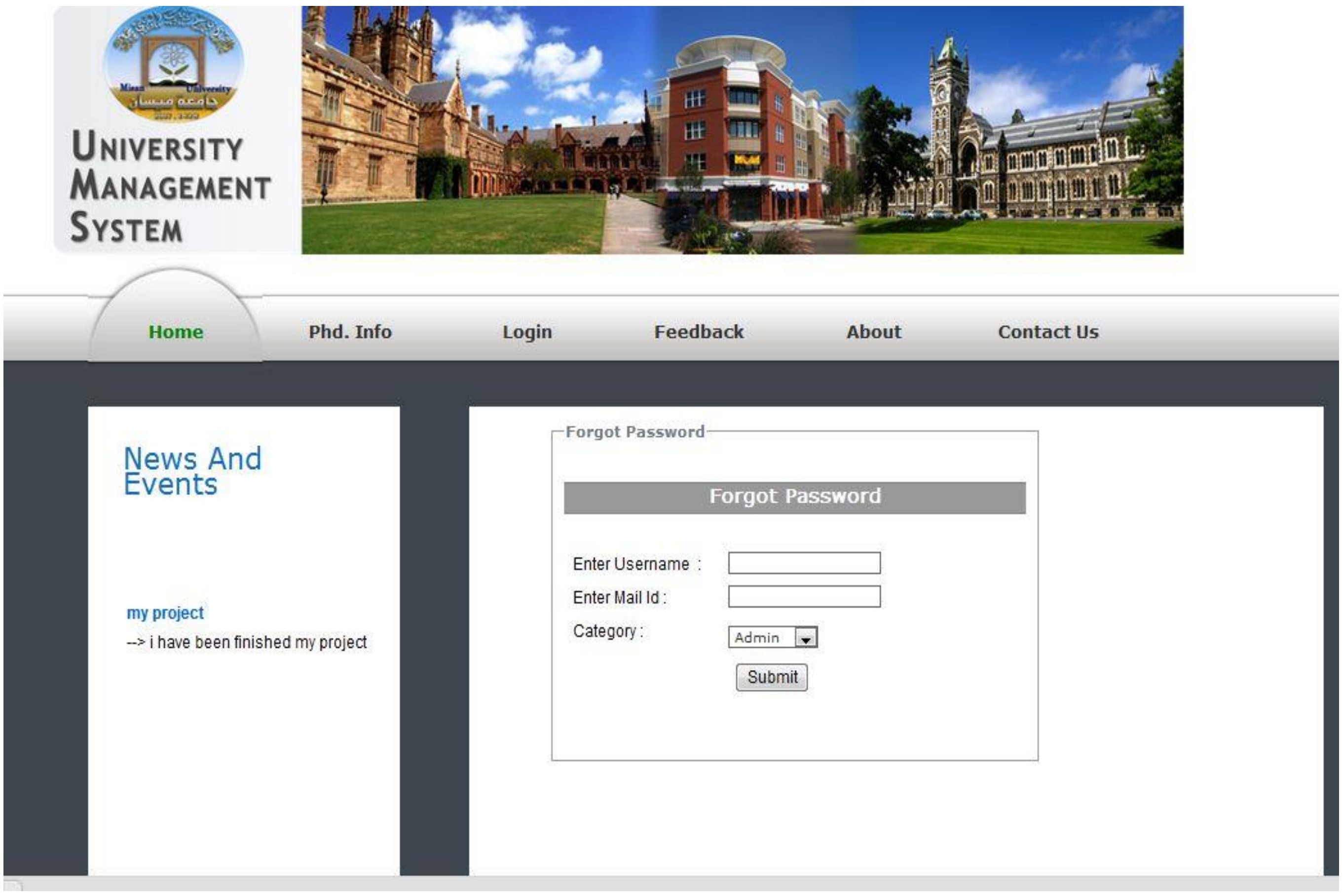
Forgot Password

Administrator enter wrong user name or password



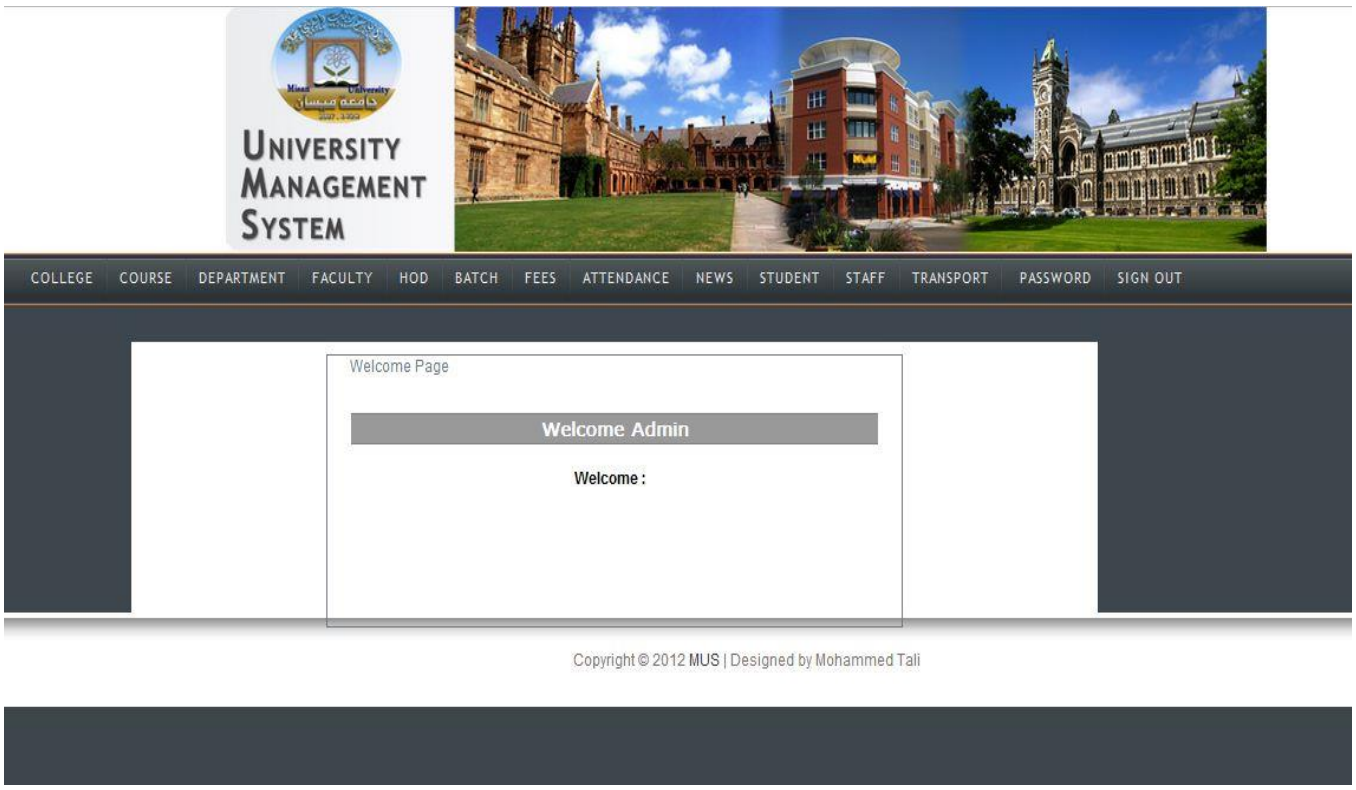
Category : Admin
User Id :
Password :
Invalid User Name or Password!!!

Forgot password page in case the user is forgot the password



Enter Username :	
Enter Mail Id :	
Category :	Admin

Administrator page



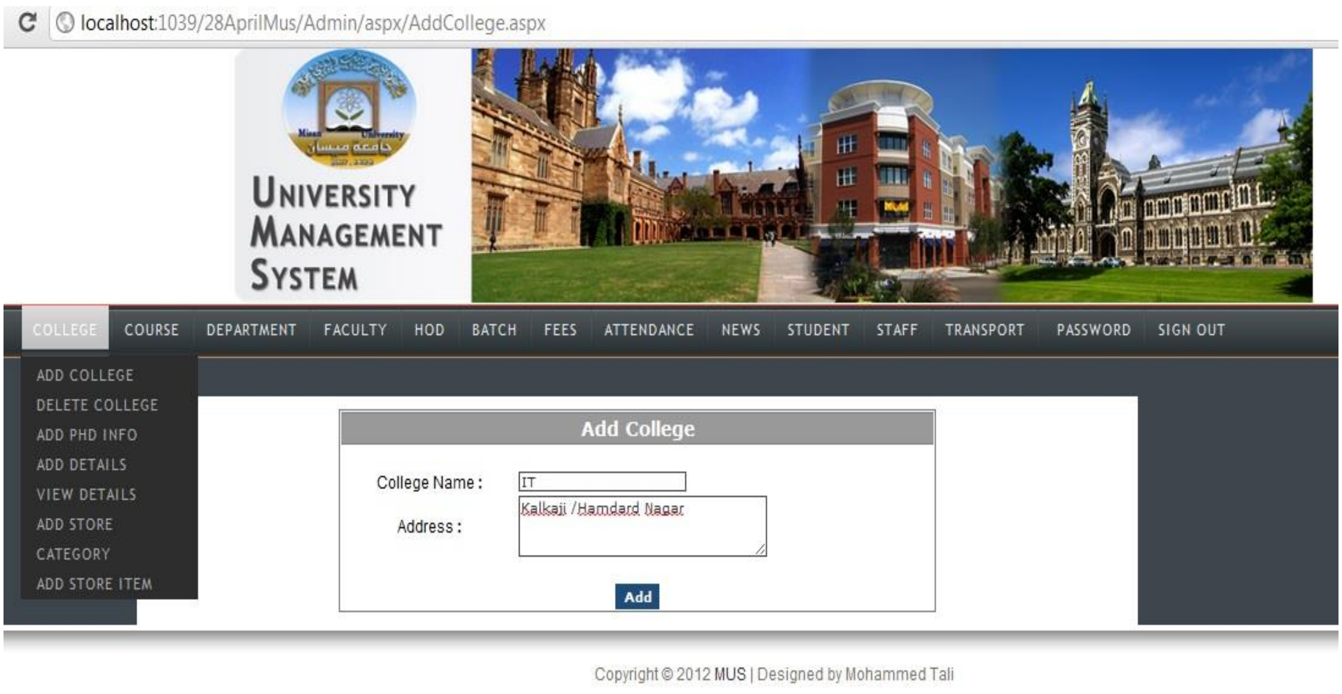
Welcome Page

Welcome Admin

Welcome :

Copyright © 2012 MUS | Designed by Mohammed Tali

New College Creation

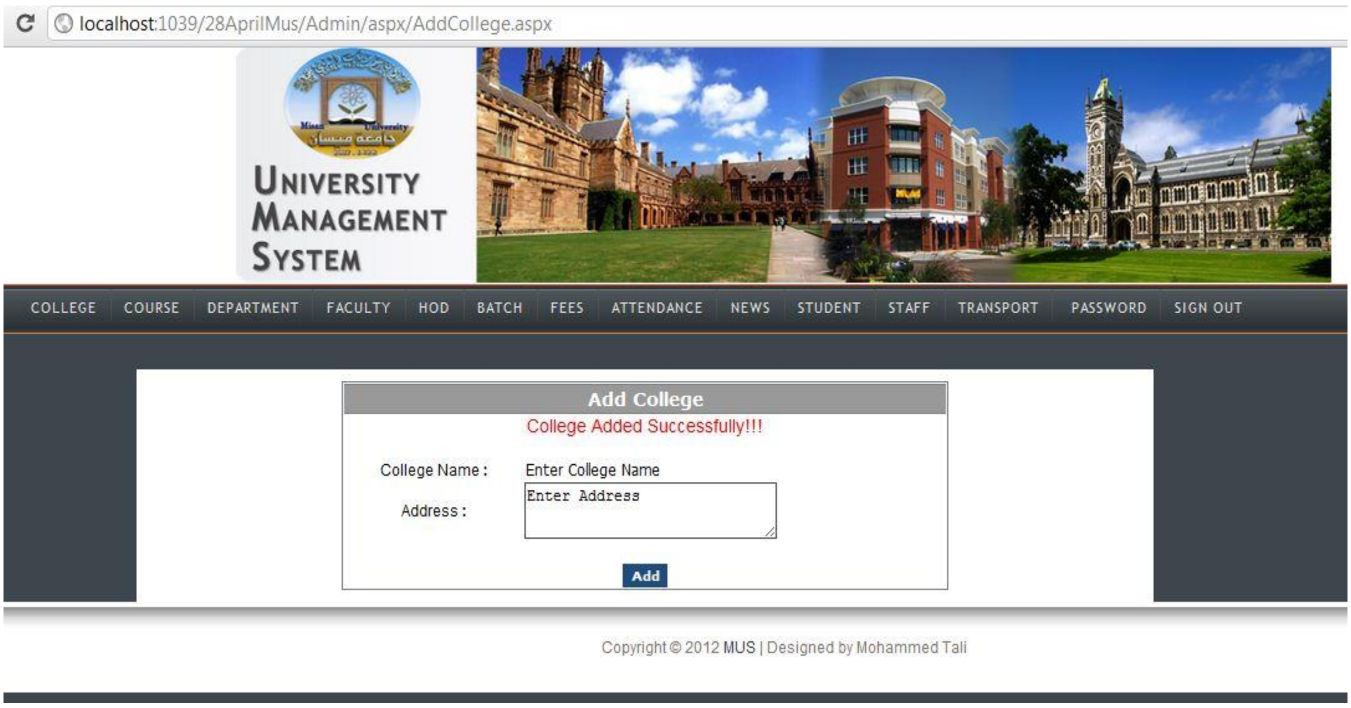


Add College

College Name : IT

Address : Kalkaji /Hamdard Nagar

Collage added successfully


localhost:1039/28AprilMus/Admin/aspx/AddCollege.aspx
UNIVERSITY MANAGEMENT SYSTEM
COLLEGE
COURSE
DEPARTMENT
FACULTY
HOD
BATCH
FEES
ATTENDANCE
NEWS
STUDENT
STAFF
TRANSPORT
PASSWORD
SIGN OUT
Add
Copyright © 2012 MUS | Designed by Mohammed Tali

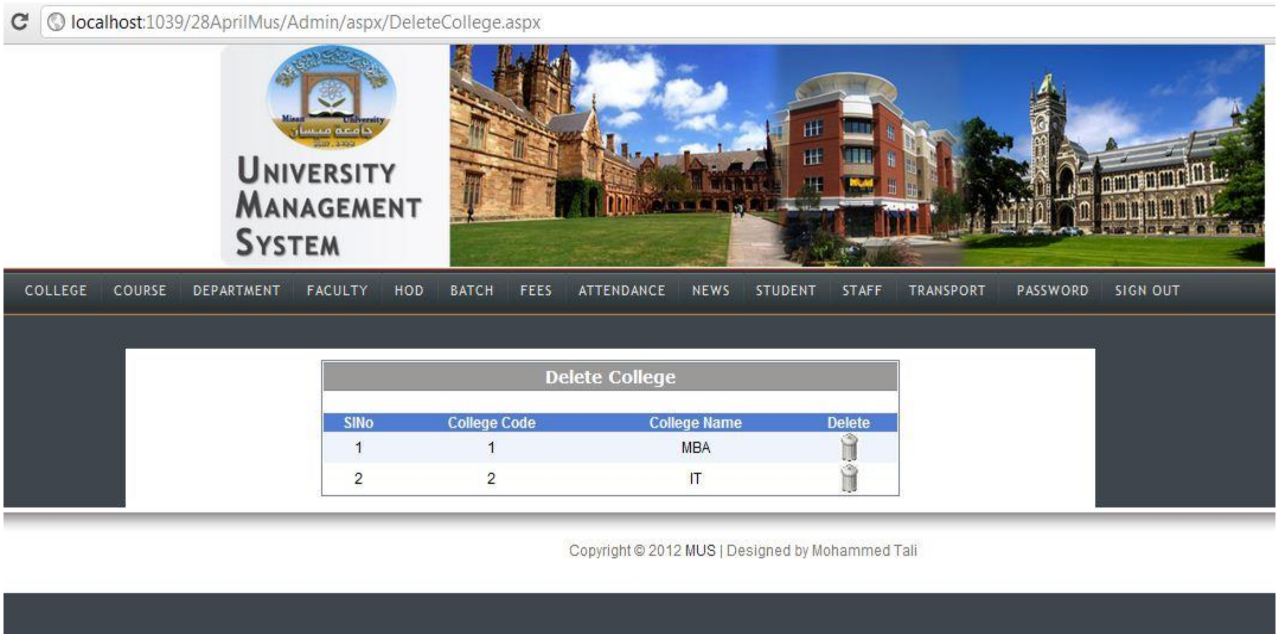
Add College

College Added Successfully!!!

College Name : Enter College Name

Address : Enter Address

Edit collage

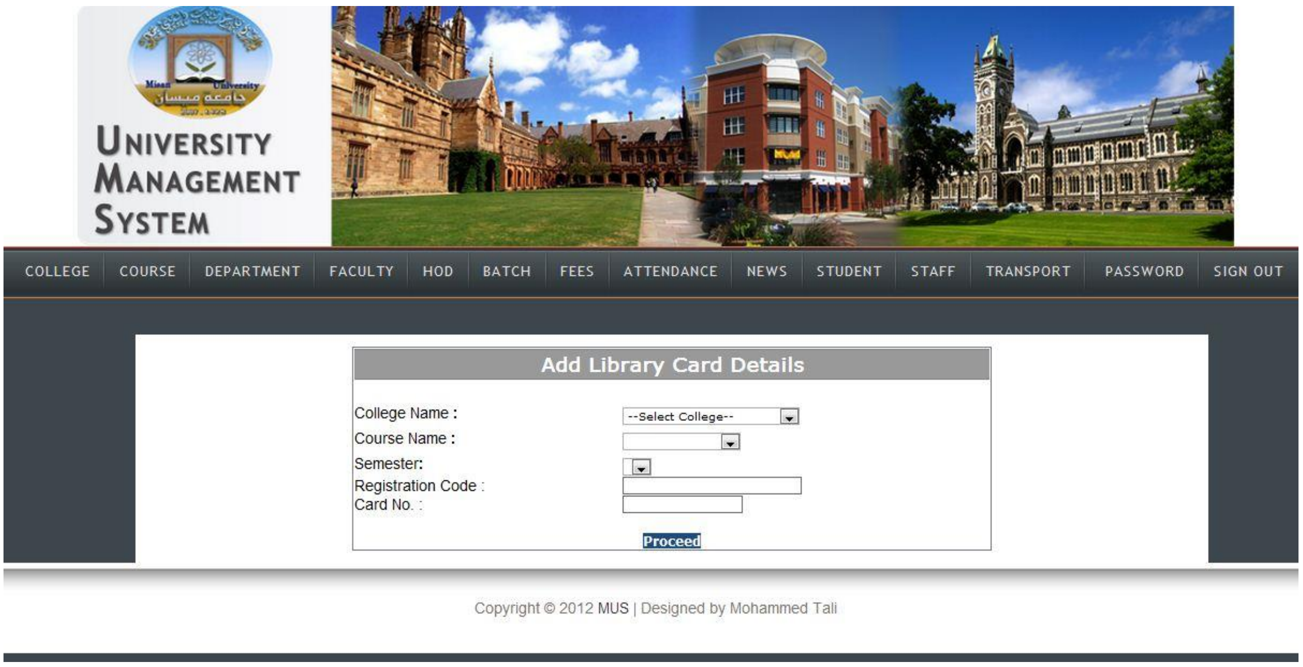


Delete College

SlNo	College Code	College Name	Delete
1	1	MBA	
2	2	IT	

Copyright © 2012 MUS | Designed by Mohammed Tali

Add library card details

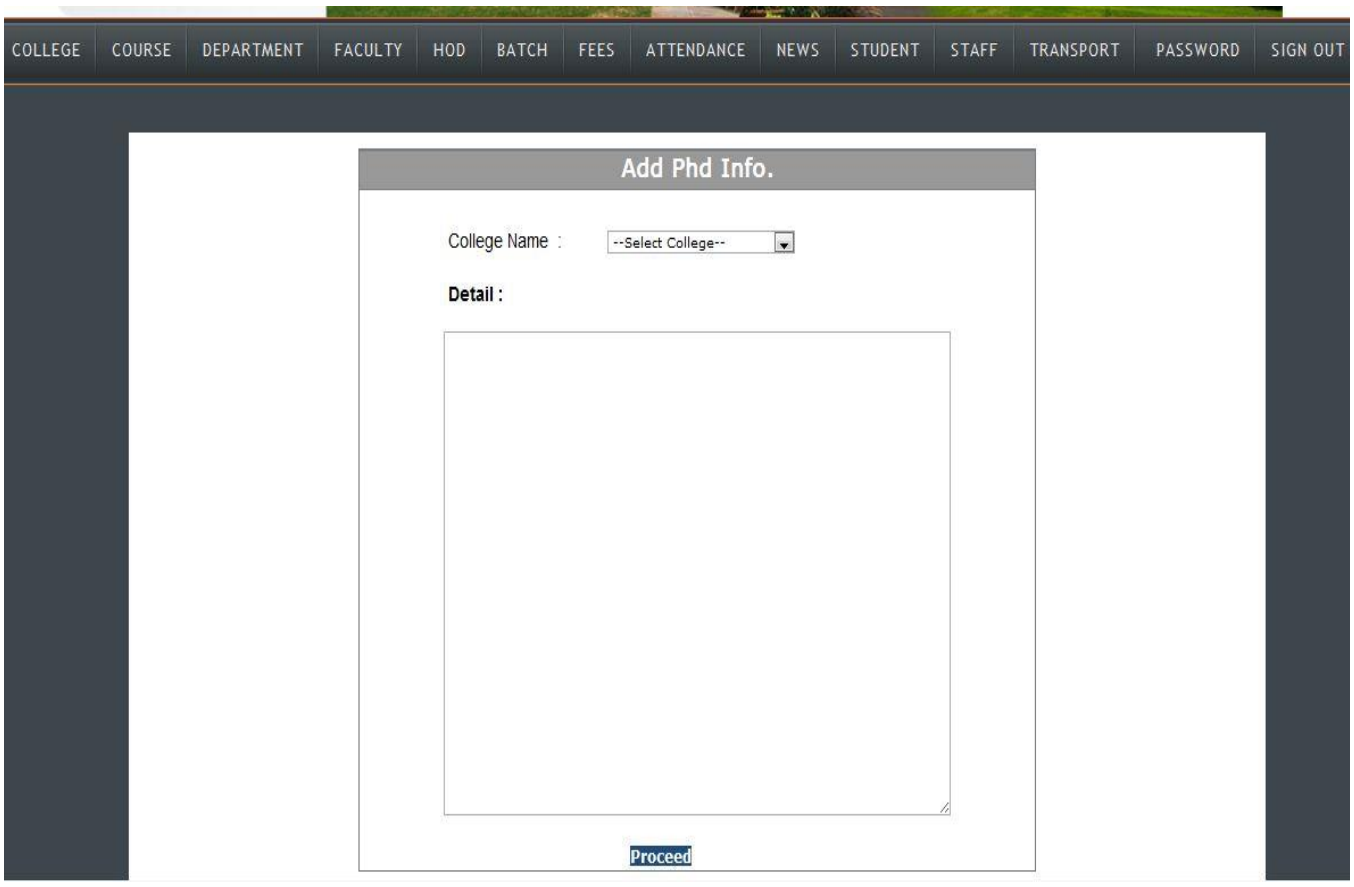


Add Library Card Details

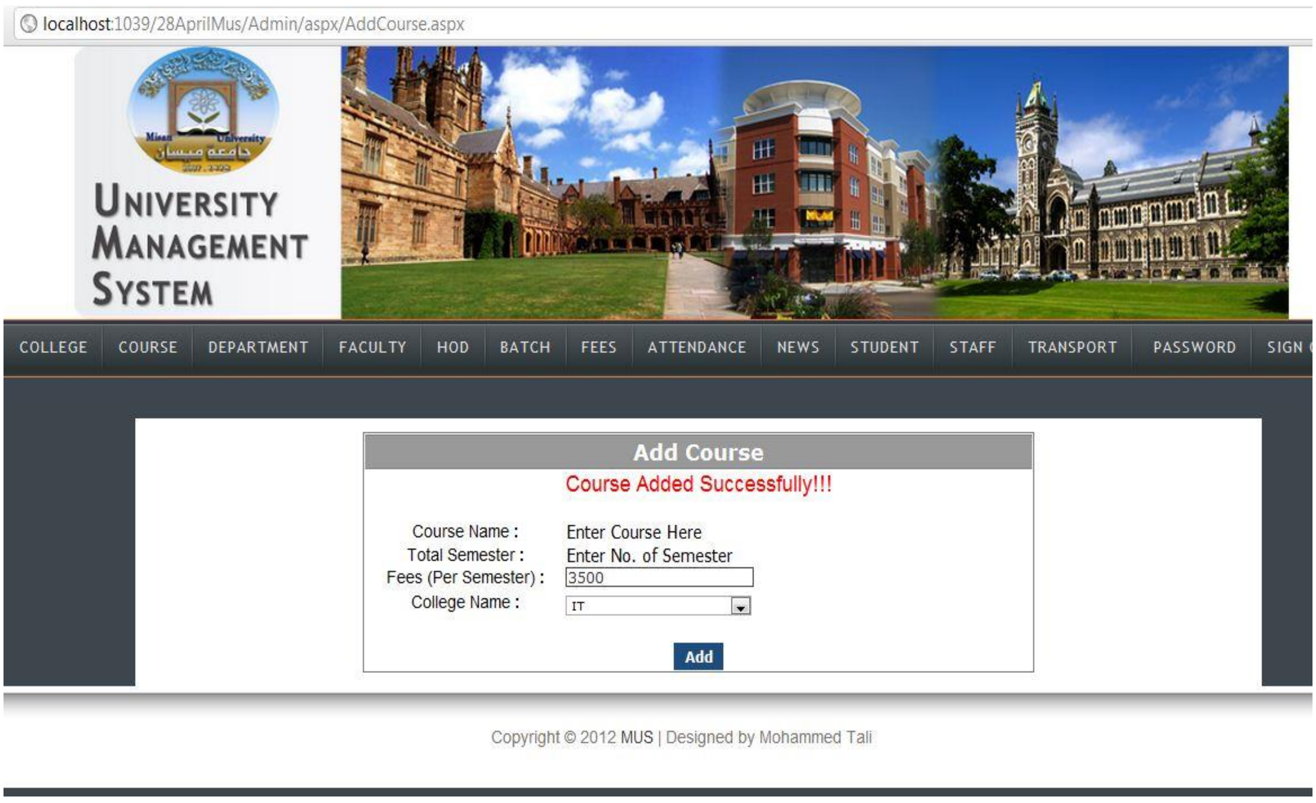
College Name :	--Select College--
Course Name :	
Semester:	
Registration Code :	
Card No. :	

Copyright © 2012 MUS | Designed by Mohammed Tali

Add information about PhD



New Courses Creation


localhost:1039/28AprilMus/Admin/aspx/AddCourse.aspx

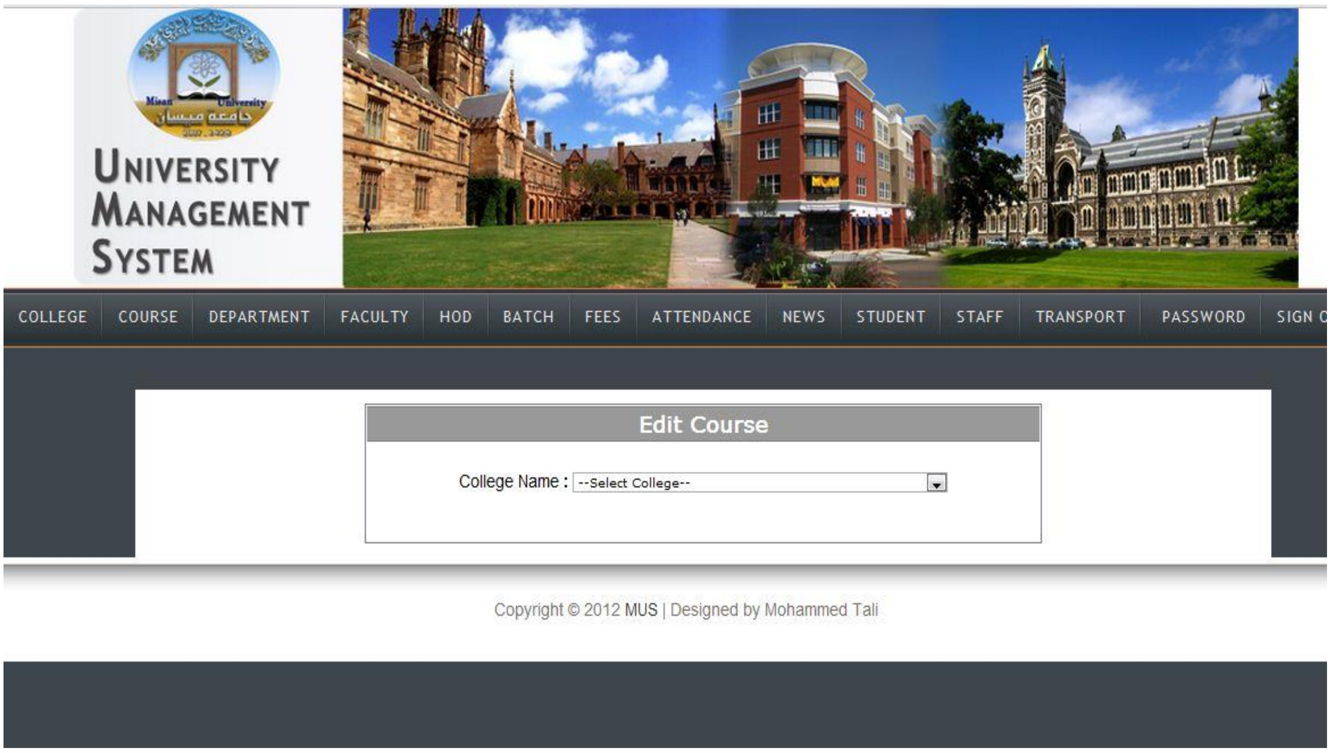
Add Course

Course Added Successfully!!!

Course Name :	Enter Course Here
Total Semester :	Enter No. of Semester
Fees (Per Semester) :	3500
College Name :	IT

Copyright © 2012 MUS | Designed by Mohammed Tali

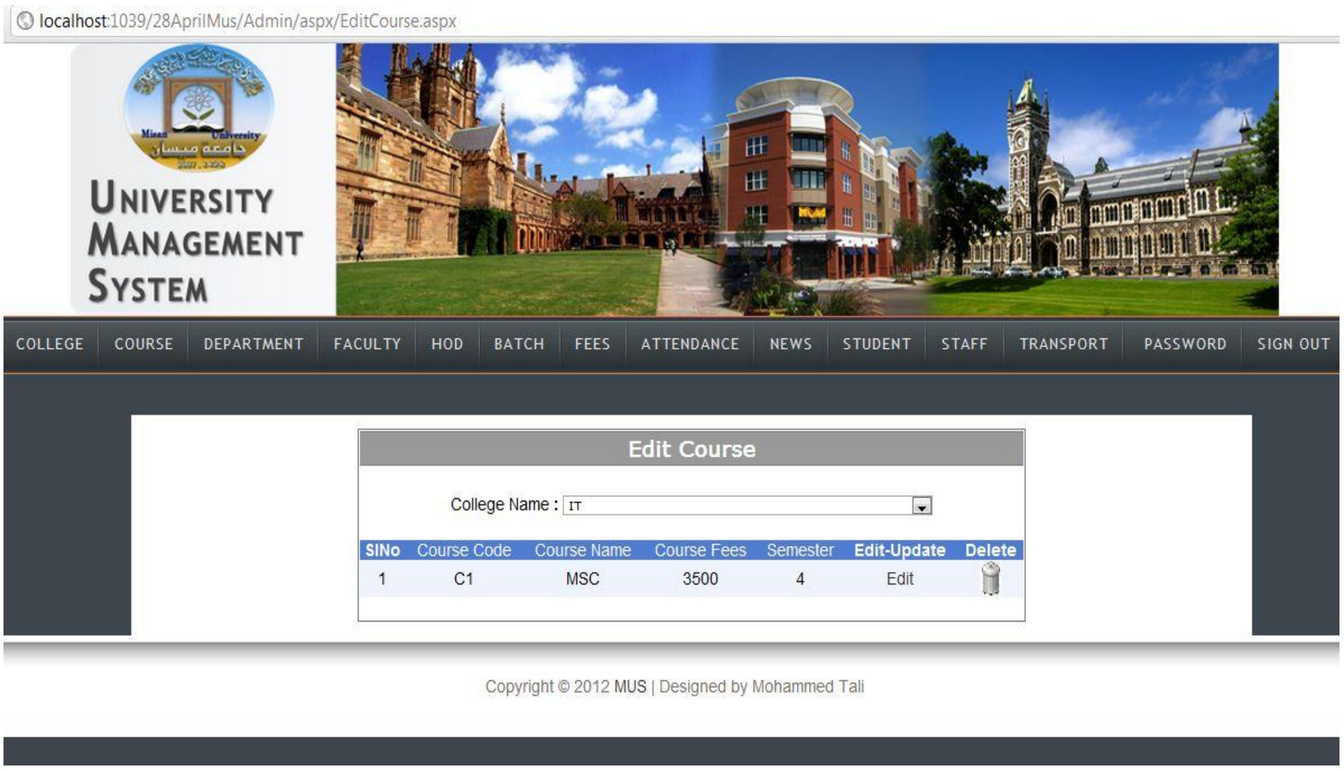
Edit courses



Edit Course

College Name : --Select College--

Copyright © 2012 MUS | Designed by Mohammed Tali



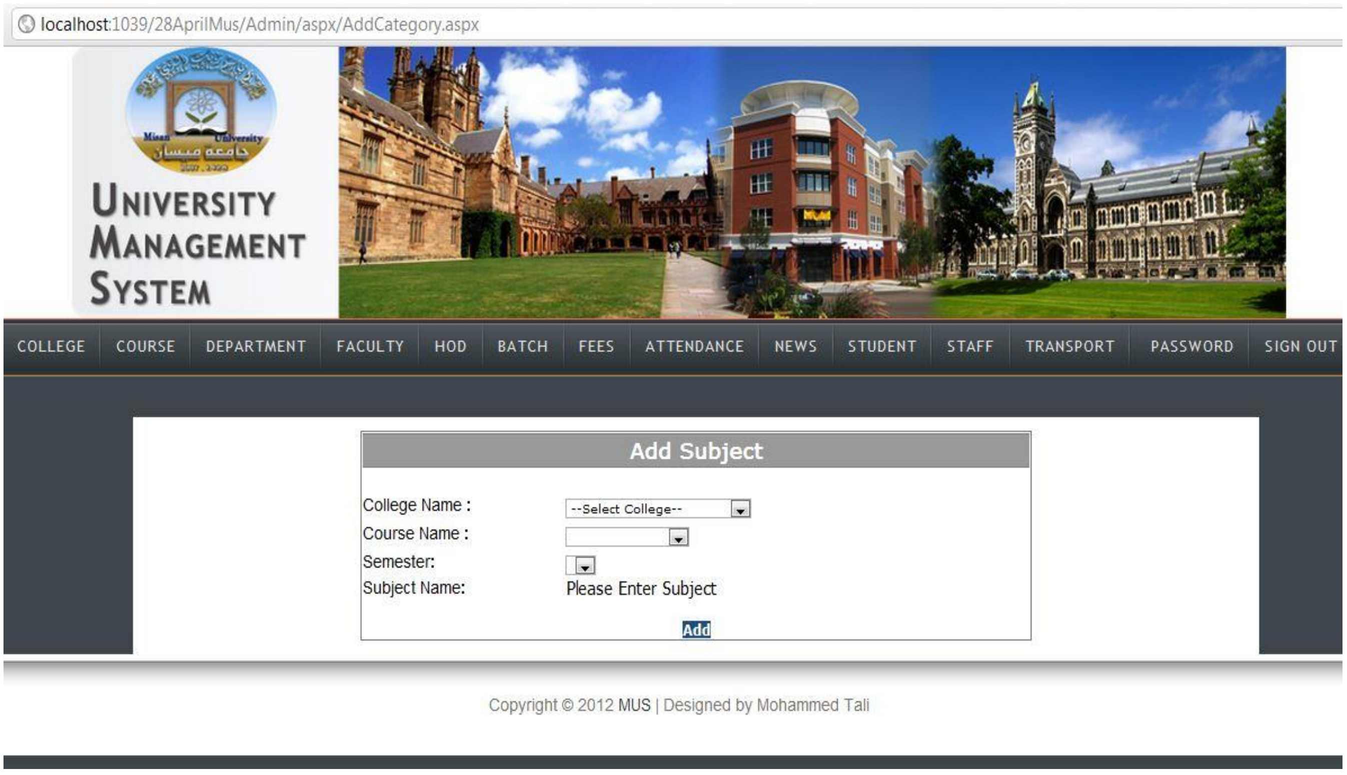
Edit Course

College Name : IT

SINo	Course Code	Course Name	Course Fees	Semester	Edit-Update	Delete
1	C1	MSC	3500	4	Edit	🗑

Copyright © 2012 MUS | Designed by Mohammed Tali

New Subject Creation


localhost:1039/28AprilMus/Admin/aspx/AddCategory.aspx

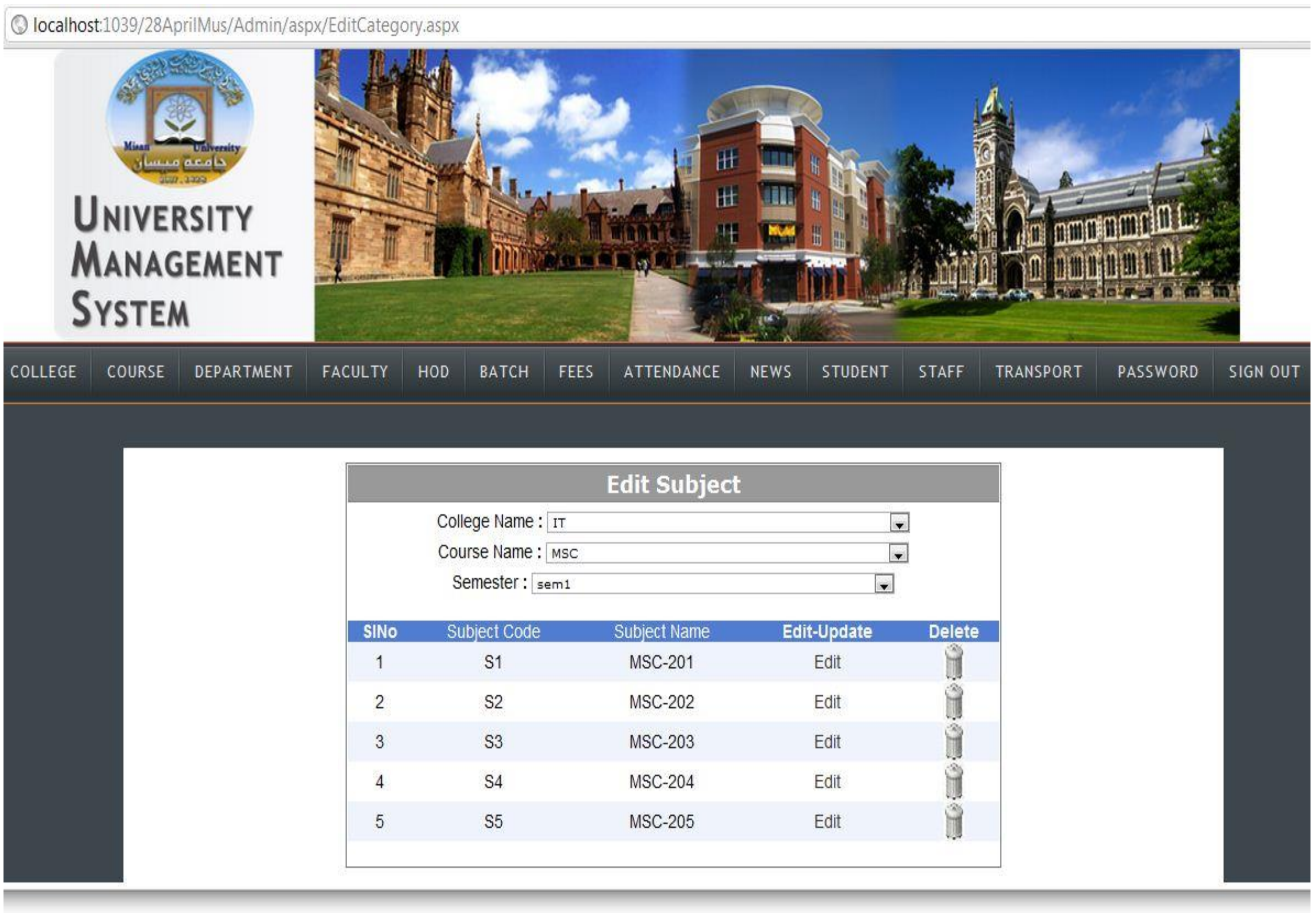
Add Subject

College Name :	--Select College--
Course Name :	
Semester:	
Subject Name:	Please Enter Subject

Add

Copyright © 2012 MUS | Designed by Mohammed Tali

Edit Subject

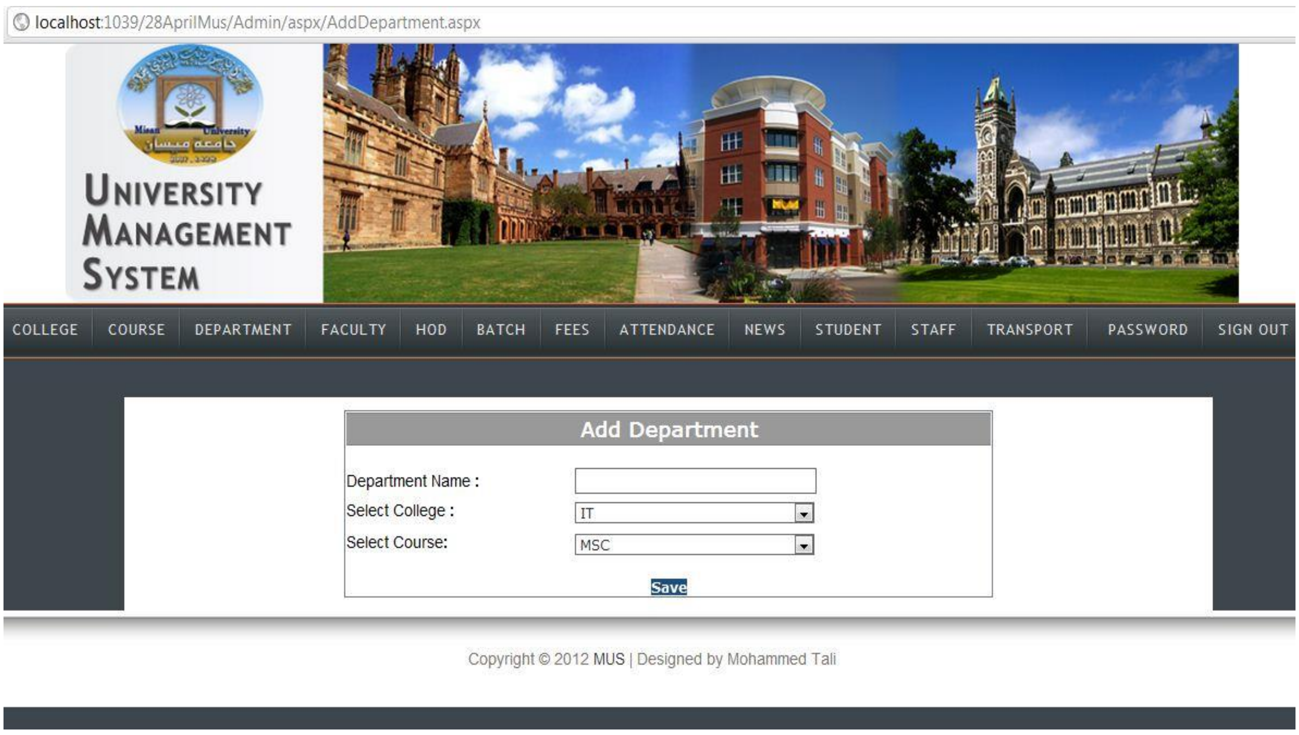

localhost:1039/28AprilMus/Admin/aspx/EditCategory.aspx

Edit Subject

College Name : IT
Course Name : MSC
Semester : sem1

SINo	Subject Code	Subject Name	Edit-Update	Delete
1	S1	MSC-201	Edit	🗑
2	S2	MSC-202	Edit	🗑
3	S3	MSC-203	Edit	🗑
4	S4	MSC-204	Edit	🗑
5	S5	MSC-205	Edit	🗑

New Department Creation



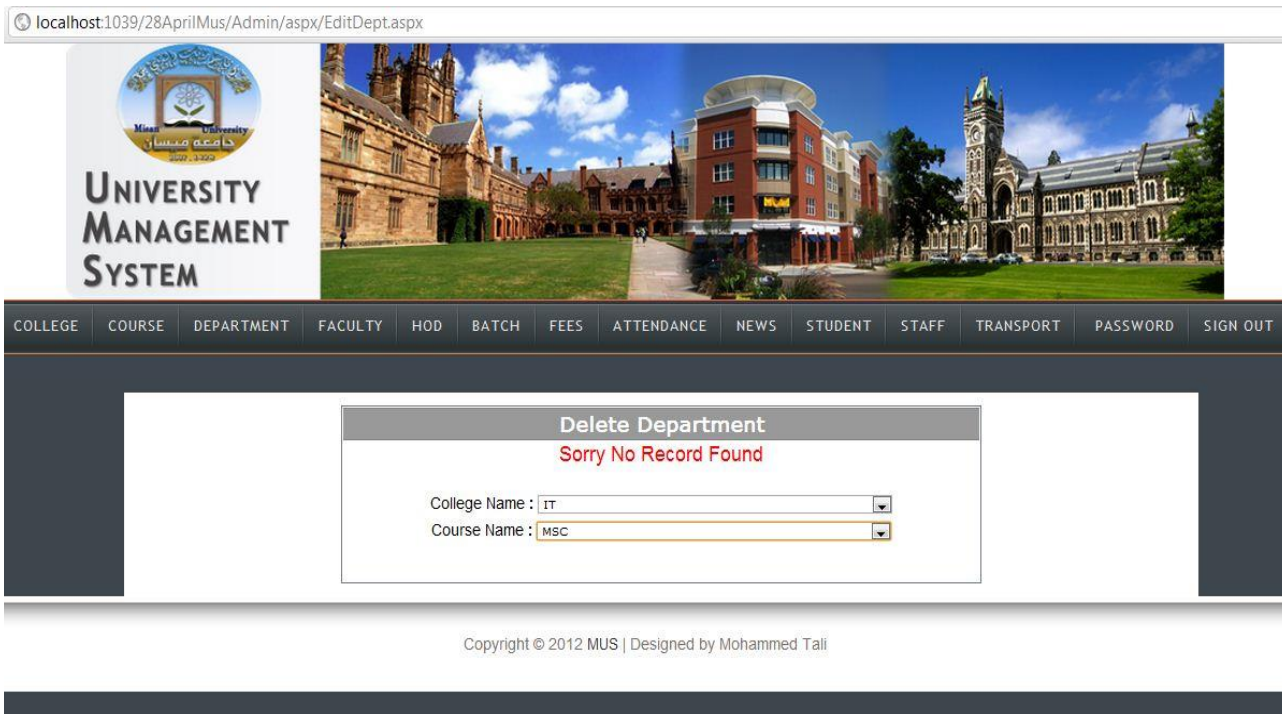
Add Department

Department Name :	
Select College :	IT
Select Course:	MSC

Save

Copyright © 2012 MUS | Designed by Mohammed Tali

In Case there no Department is added yet



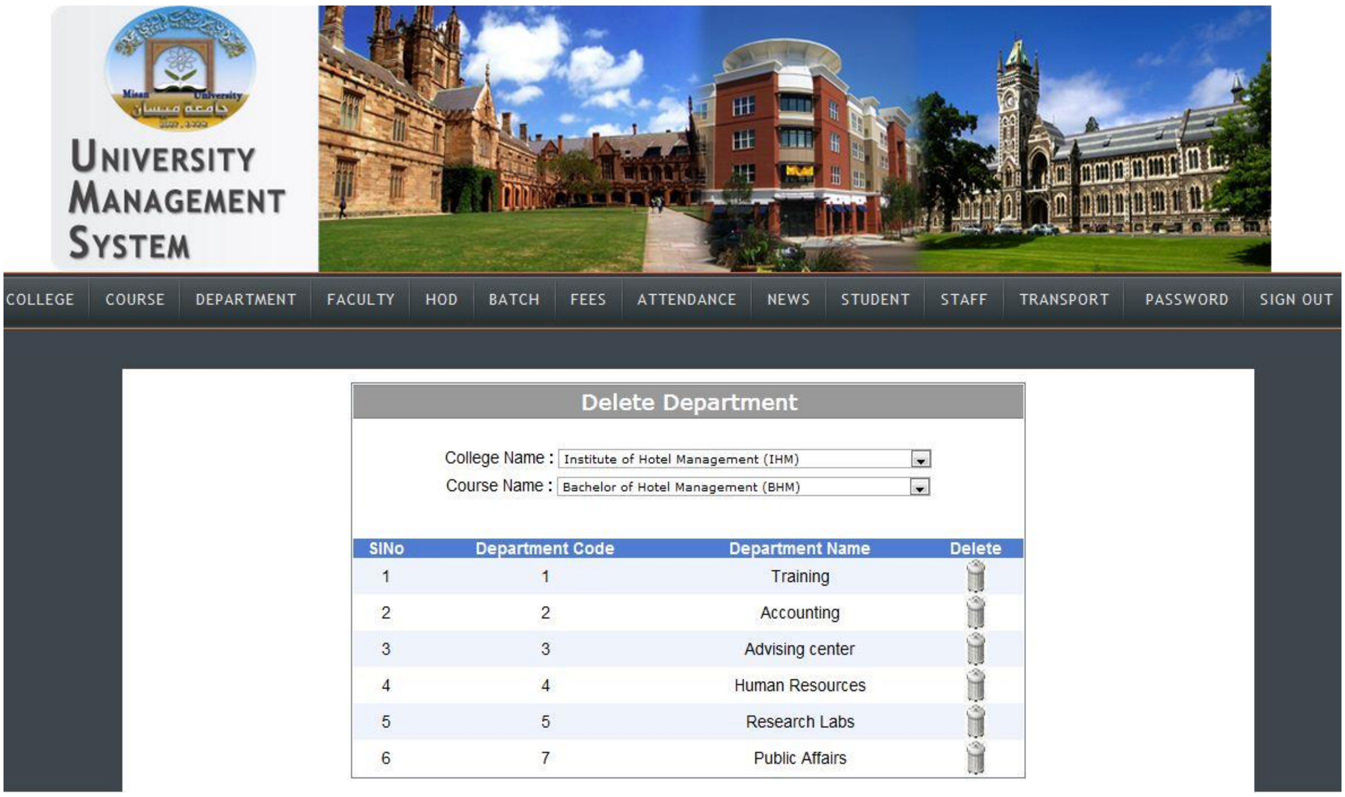
Delete Department

Sorry No Record Found

College Name :	IT
Course Name :	MSC

Copyright © 2012 MUS | Designed by Mohammed Tali

Delete Department



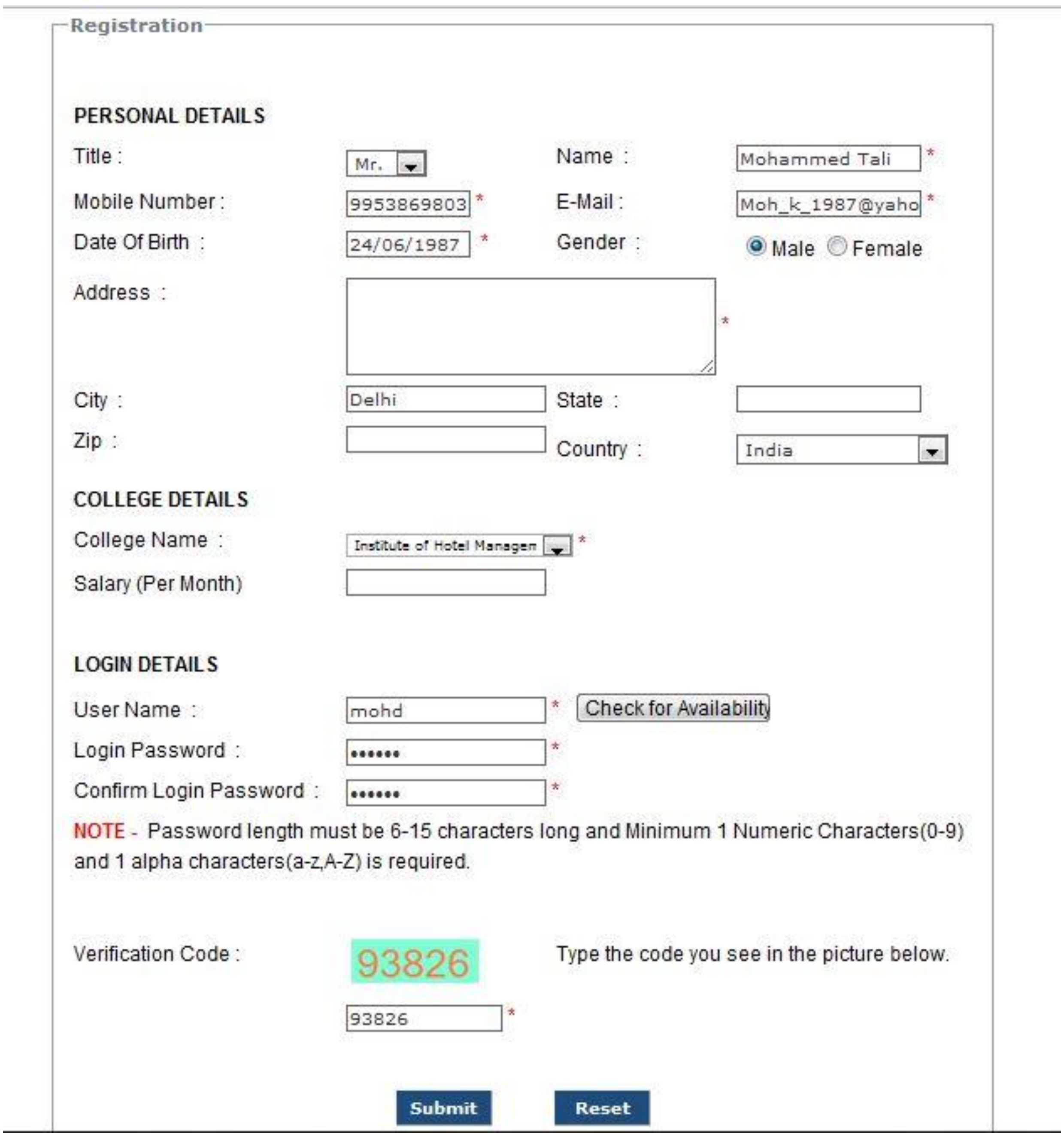
Delete Department

College Name : Institute of Hotel Management (IHM)

Course Name : Bachelor of Hotel Management (BHM)

SlNo	Department Code	Department Name	Delete
1	1	Training	🗑
2	2	Accounting	🗑
3	3	Advising center	🗑
4	4	Human Resources	🗑
5	5	Research Labs	🗑
6	7	Public Affairs	🗑

Registration of the student



Registration

PERSONAL DETAILS

Title :	Mr.	Name :	Mohammed Tali *
Mobile Number :	9953869803 *	E-Mail :	Moh_k_1987@yaho *
Date Of Birth :	24/06/1987 *	Gender :	[✓ Male] Female
Address :	*		
City :	Delhi	State :	
Zip :		Country :	India

COLLEGE DETAILS

College Name :	Institute of Hotel Managem *
Salary (Per Month)	

LOGIN DETAILS

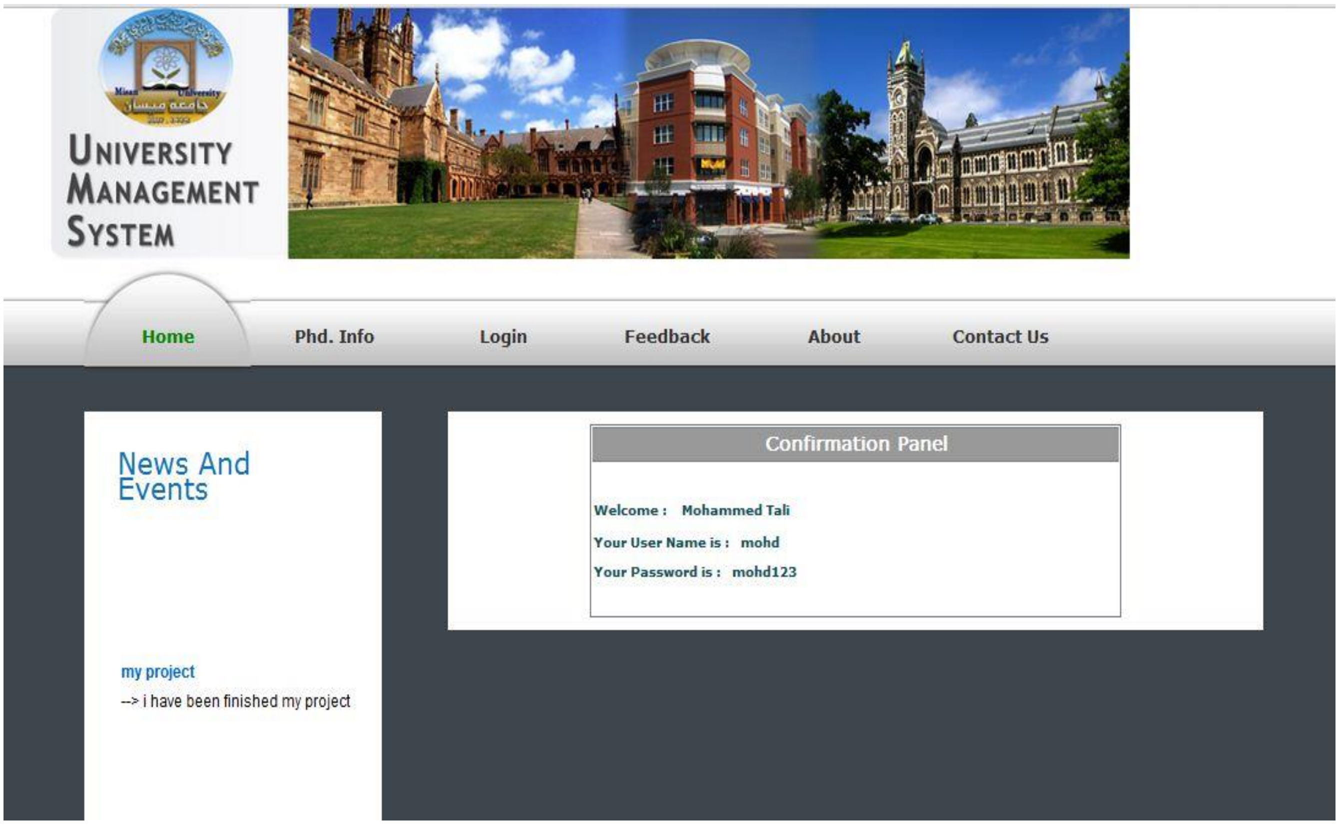
User Name :	mohd *	Check for Availability
Login Password :	•••••• *	
Confirm Login Password :	•••••• *	

NOTE - Password length must be 6-15 characters long and Minimum 1 Numeric Characters(0-9) and 1 alpha characters(a-z,A-Z) is required.

Verification Code : 93826 Type the code you see in the picture below.

93826 *

After the registration process will show the panel to confirm the faculty's name, user name and password.



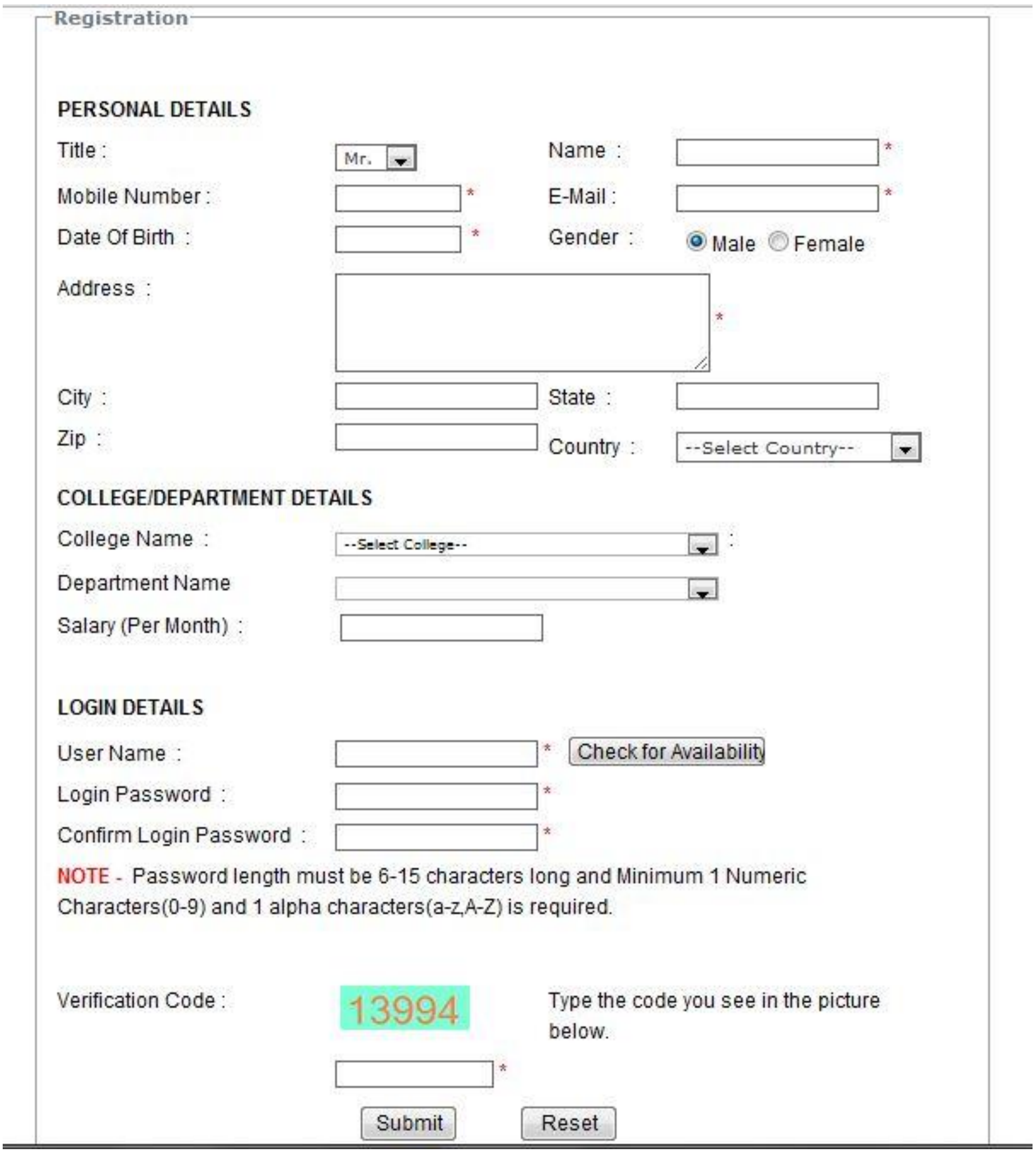
Confirmation Panel

Welcome : Mohammed Tali

Your User Name is : mohd

Your Password is : mohd123

Registration of the HOD


Registration

PERSONAL DETAILS

Title :	Mr.	Name :	*
Mobile Number :	*	E-Mail :	*
Date Of Birth :	*	Gender :	[✓ Male] Female
Address :	*		
City :		State :	
Zip :		Country :	--Select Country--

COLLEGE/DEPARTMENT DETAILS

College Name :	--Select College--
Department Name	
Salary (Per Month) :	

LOGIN DETAILS

User Name :	*	Check for Availability
Login Password :	*	
Confirm Login Password :	*	

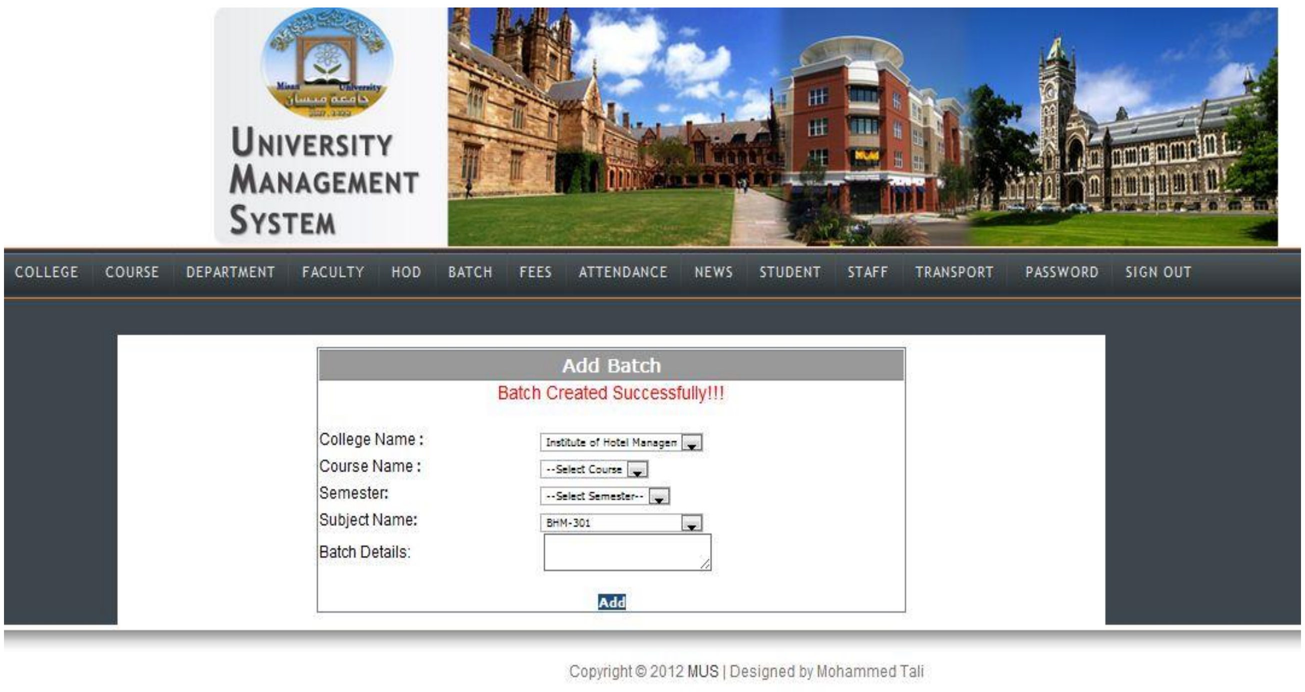
NOTE - Password length must be 6-15 characters long and Minimum 1 Numeric Characters(0-9) and 1 alpha characters(a-z,A-Z) is required.

Verification Code : 13994 Type the code you see in the picture below.

*

Submit Reset

Add Batch

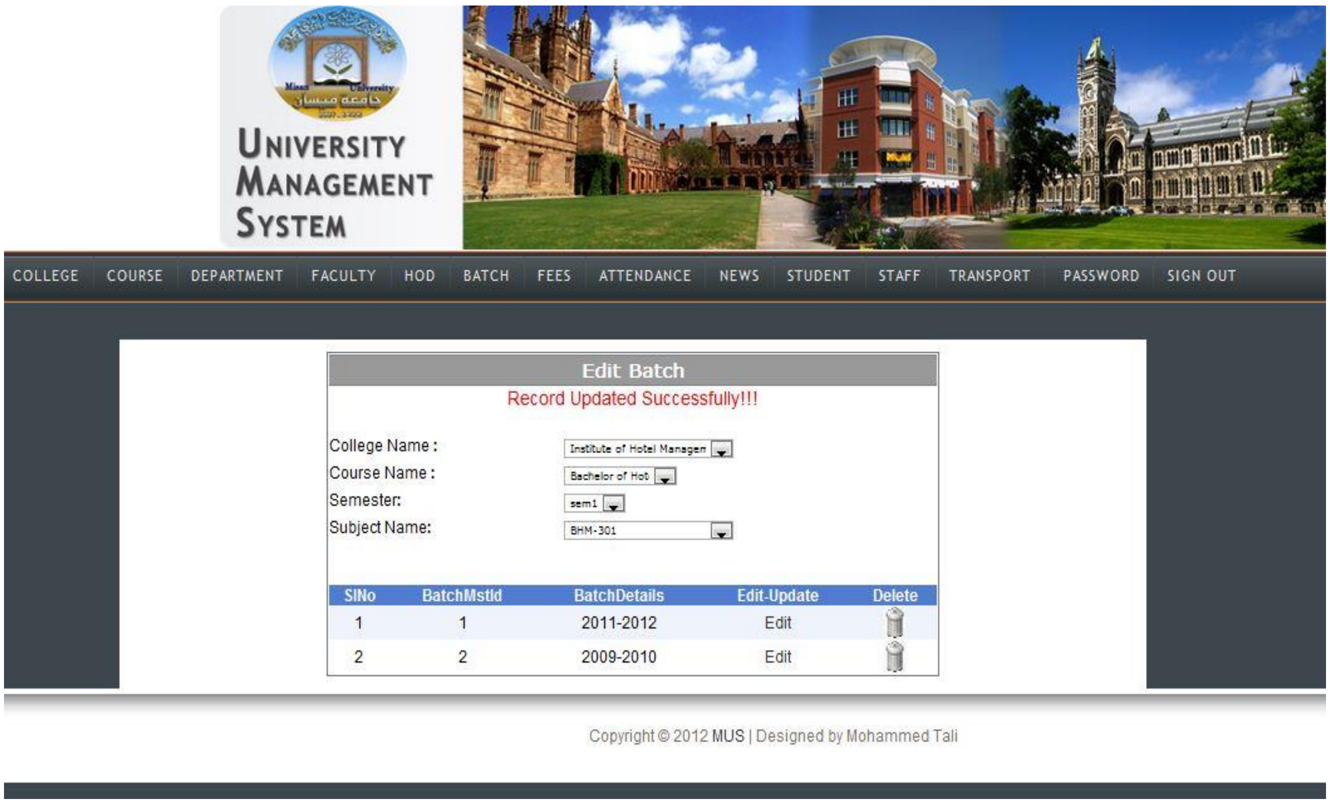


Batch Created Successfully!!!

College Name :	Institute of Hotel Managem
Course Name :	--Select Course
Semester:	--Select Semester--
Subject Name:	BHM-301
Batch Details:	

Copyright © 2012 MUS | Designed by Mohammed Tali

Edit Batch



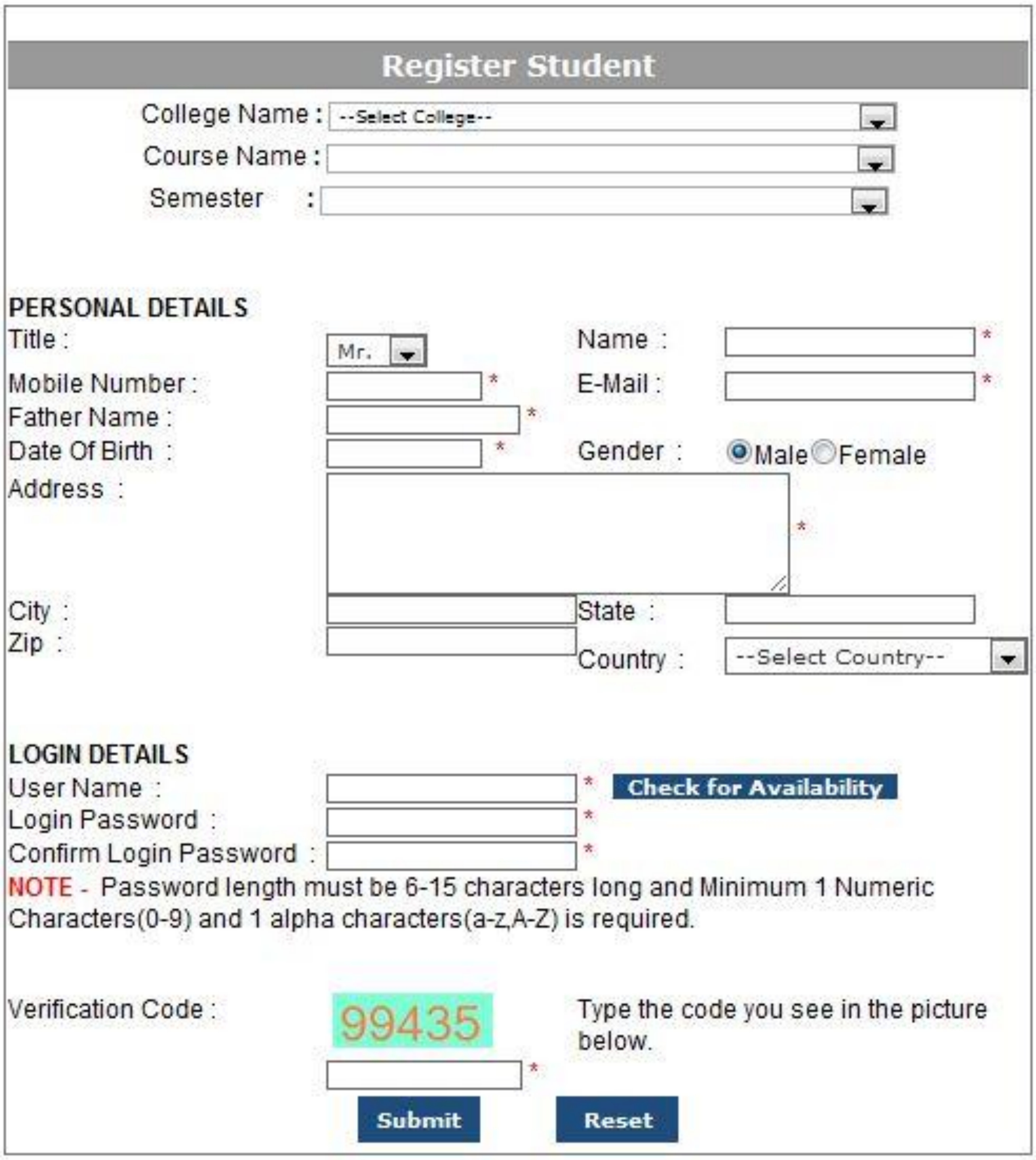
Record Updated Successfully!!!

College Name :	Institute of Hotel Managem
Course Name :	Bachelor of Hot
Semester:	sem1
Subject Name:	BHM-301

SlNo	BatchMstId	BatchDetails	Edit-Update	Delete
1	1	2011-2012	Edit	
2	2	2009-2010	Edit	

Copyright © 2012 MUS | Designed by Mohammed Tali

Registration of the Student



Register Student

College Name : --Select College--

Course Name :

Semester :

PERSONAL DETAILS

Title :	Mr.	Name :	*
Mobile Number :	*	E-Mail :	*
Father Name :	*		
Date Of Birth :	*	Gender :	[✓ Male] Female
Address :	*		
City :		State :	
Zip :		Country :	--Select Country--

LOGIN DETAILS

User Name :	*	Check for Availability
Login Password :	*	
Confirm Login Password :	*	

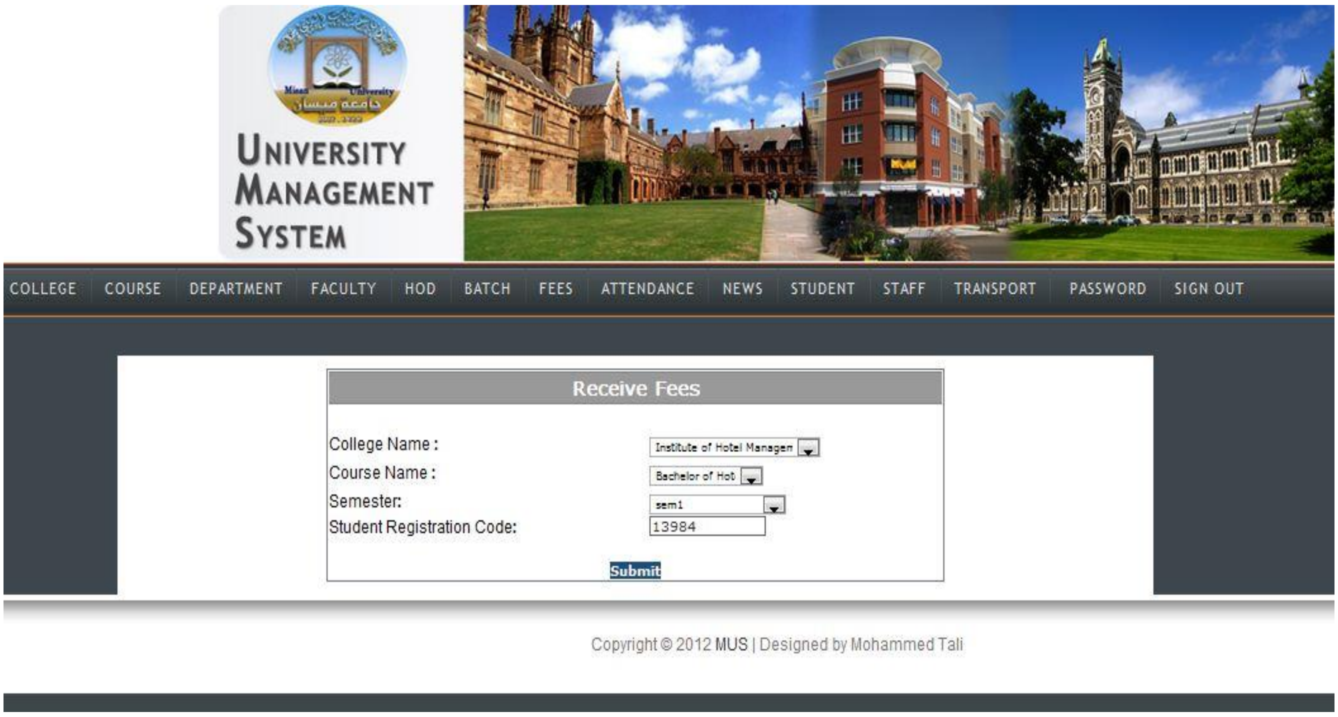
NOTE - Password length must be 6-15 characters long and Minimum 1 Numeric Characters(0-9) and 1 alpha characters(a-z,A-Z) is required.

Verification Code : 99435 Type the code you see in the picture below.

*

Submit Reset

Receive Fees

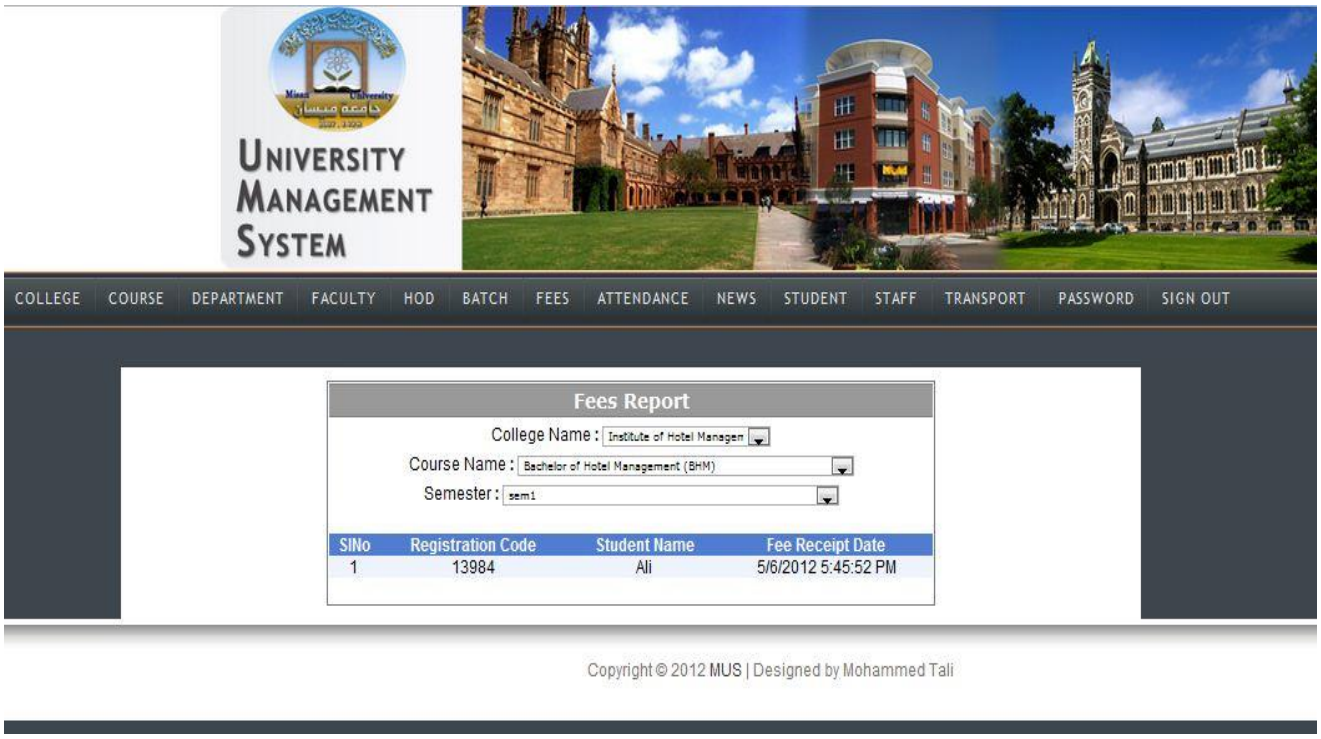


Receive Fees

College Name :	Institute of Hotel Managem
Course Name :	Bachelor of Hot
Semester:	sem1
Student Registration Code:	13984

Copyright © 2012 MUS | Designed by Mohammed Tali

Fees Report



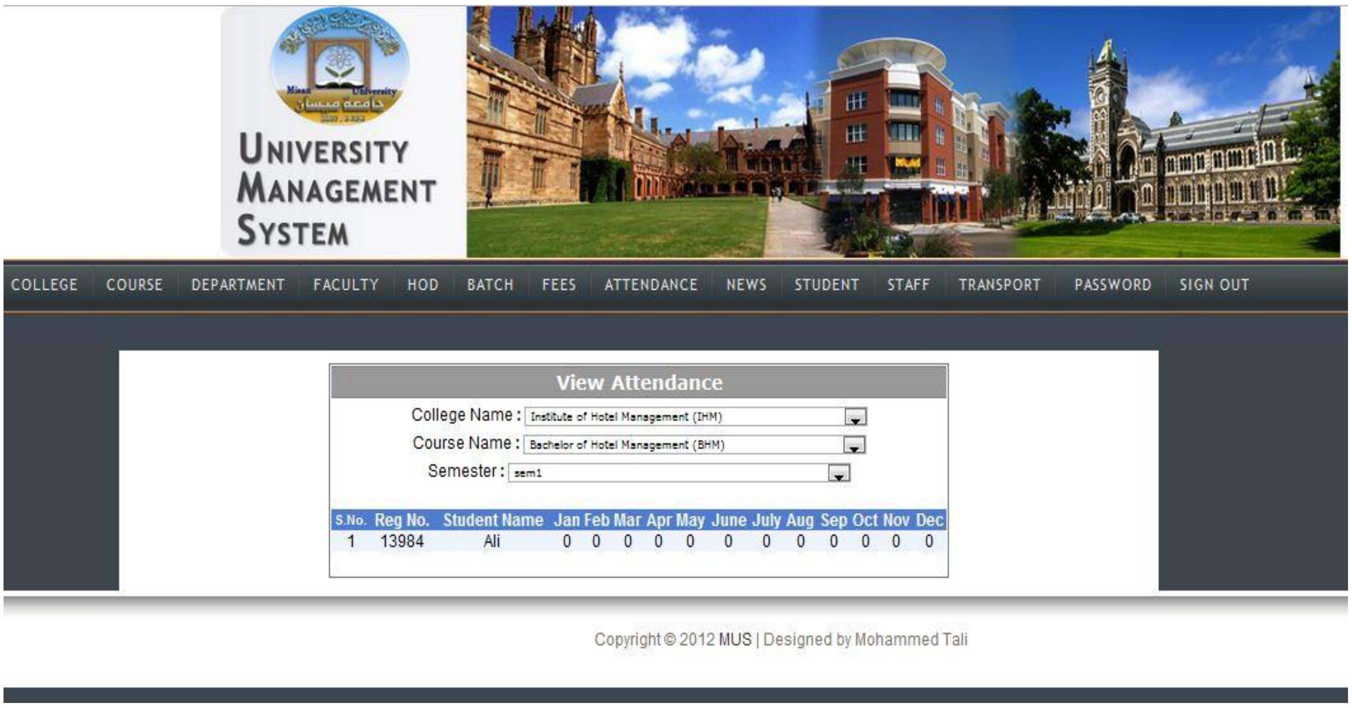
Fees Report

College Name : Institute of Hotel Managem
Course Name : Bachelor of Hotel Management (BHM)
Semester : sem1

SlNo	Registration Code	Student Name	Fee Receipt Date
1	13984	Ali	5/6/2012 5:45:52 PM

Copyright © 2012 MUS | Designed by Mohammed Tali

View Attendance



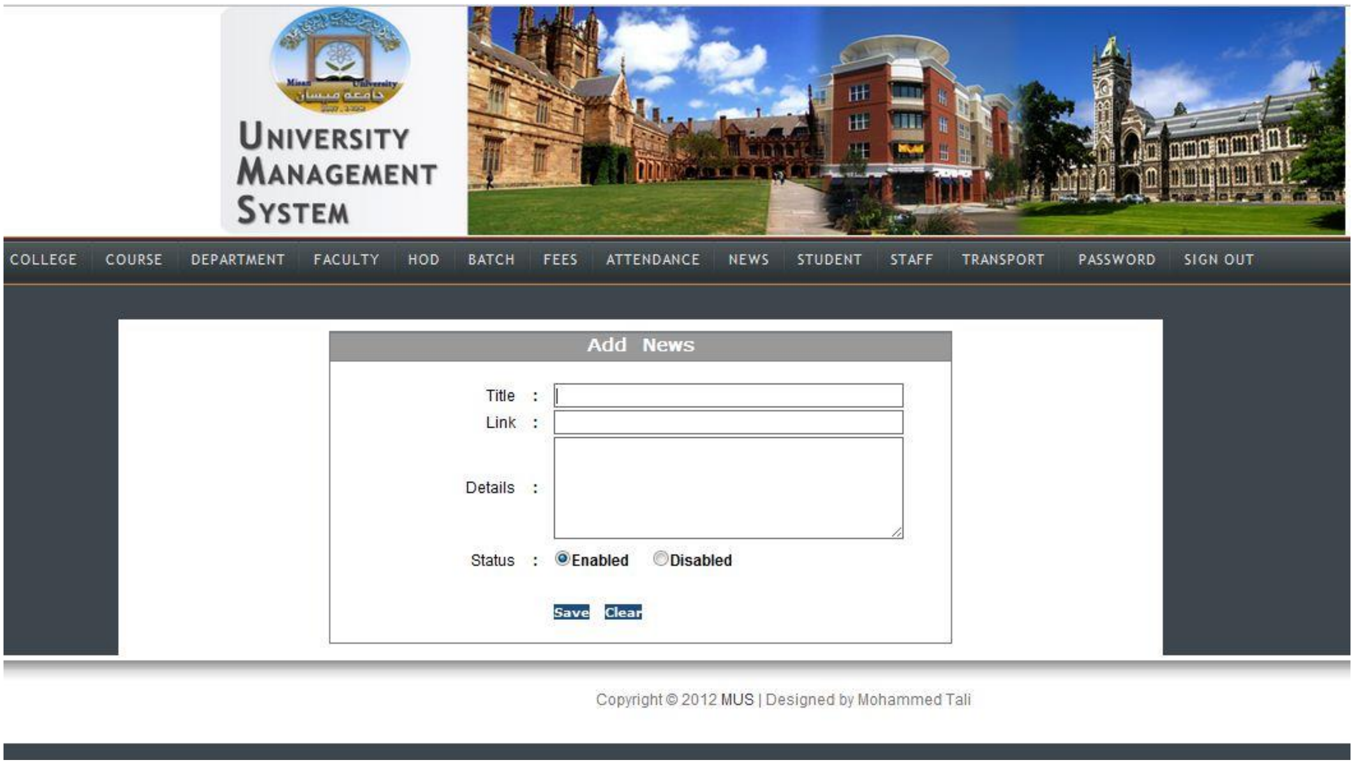
View Attendance

College Name : Institute of Hotel Management (IHM)
Course Name : Bachelor of Hotel Management (BHM)
Semester : sem1

S.No.	Reg No.	Student Name	Jan	Feb	Mar	Apr	May	June	July	Aug	Sep	Oct	Nov	Dec
1	13984	Ali	0	0	0	0	0	0	0	0	0	0	0	0

Copyright © 2012 MUS | Designed by Mohammed Tali

Add News

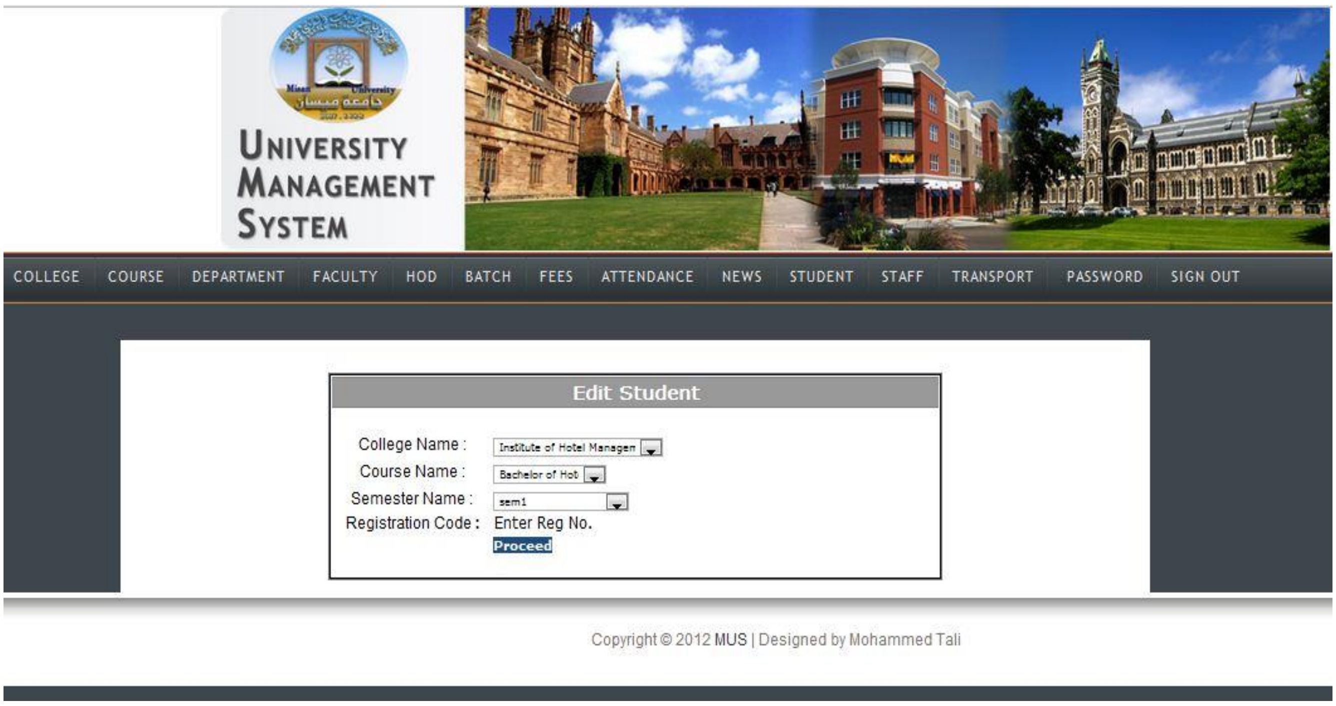


Add News

Title :
Link :
Details :
Status : [✓ Enabled] Disabled

Copyright © 2012 MUS | Designed by Mohammed Tali

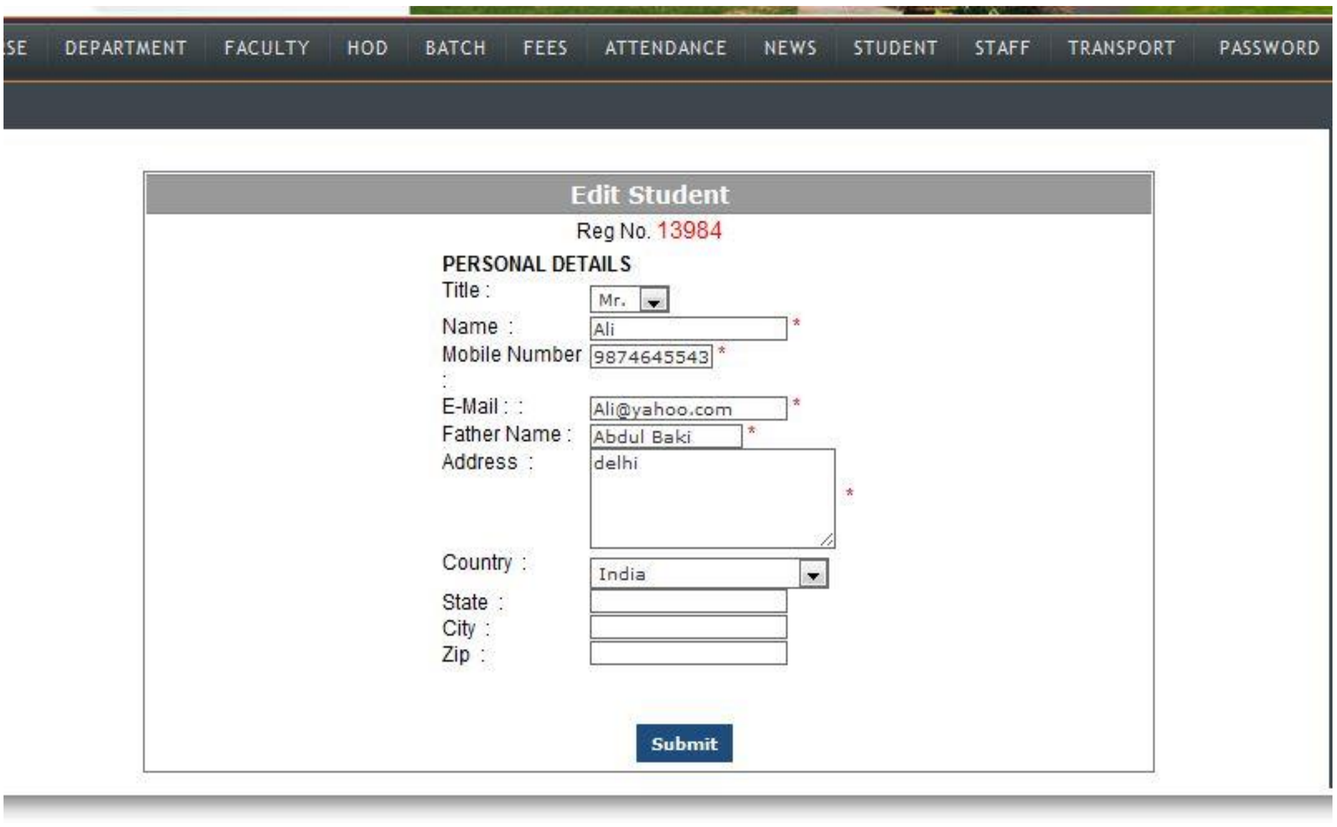
Edit Student



Edit Student

College Name :	Institute of Hotel Managem
Course Name :	Bachelor of Hot
Semester Name :	sem1
Registration Code :	Enter Reg No.

Copyright © 2012 MUS | Designed by Mohammed Tali



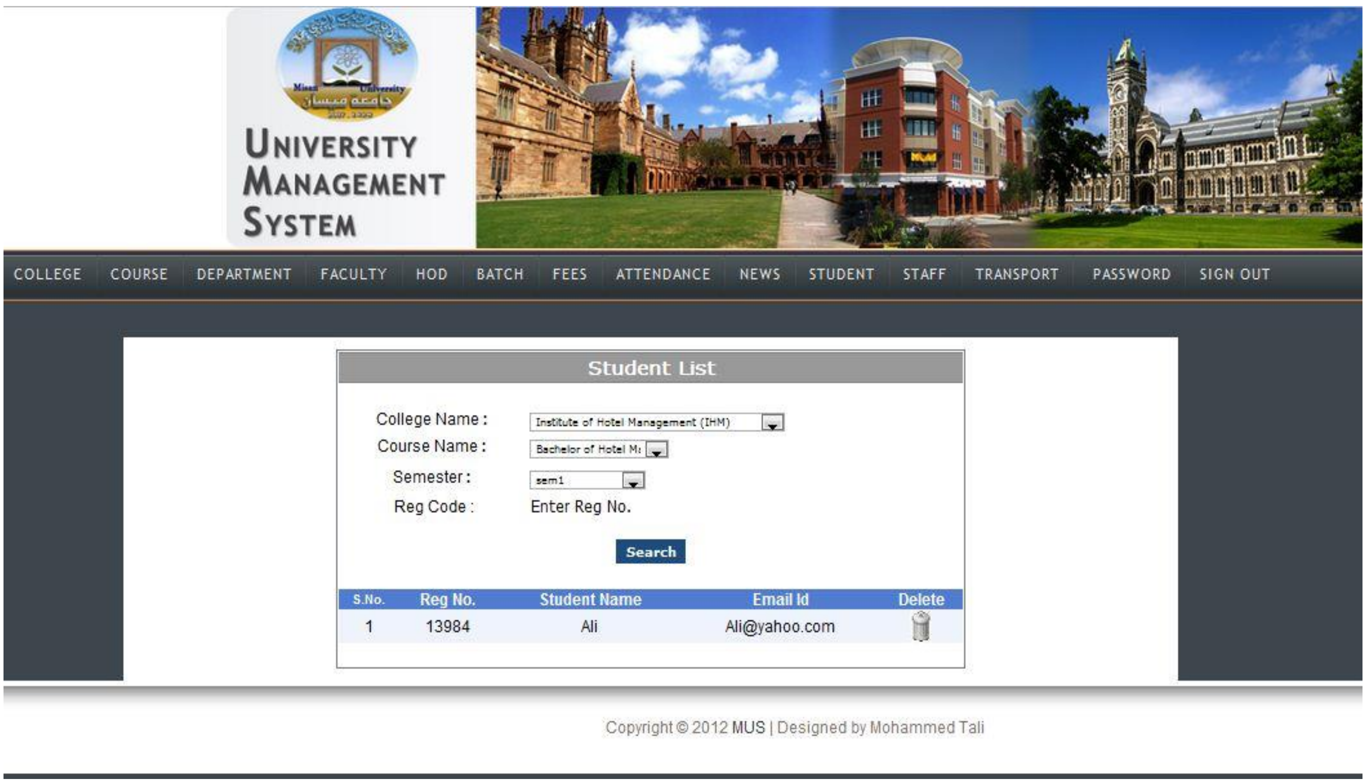
Edit Student

Reg No. 13984

PERSONAL DETAILS

Title :	Mr.
Name :	Ali *
Mobile Number :	9874645543 *
E-Mail : :	Ali@yahoo.com *
Father Name :	Abdul Baki *
Address :	delhi *
Country :	India
State :	
City :	
Zip :	

View a list of students


UNIVERSITY MANAGEMENT SYSTEM
COLLEGE COURSE DEPARTMENT FACULTY HOD BATCH FEES ATTENDANCE NEWS STUDENT STAFF TRANSPORT PASSWORD SIGN OUT
Search

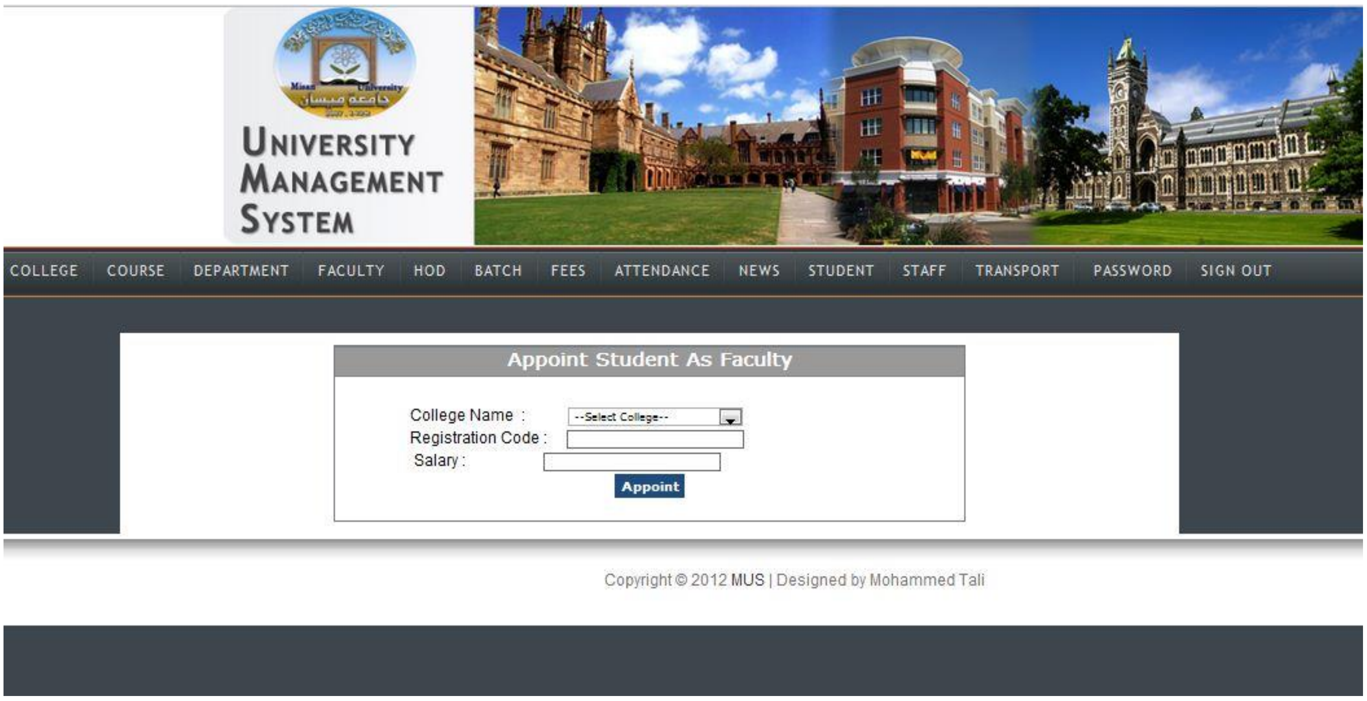
Student List

College Name : Institute of Hotel Management (IHM)
Course Name : Bachelor of Hotel M:
Semester : sem1
Reg Code : Enter Reg No.

S.No.	Reg No.	Student Name	Email Id	Delete
1	13984	Ali	Ali@yahoo.com	

Copyright © 2012 MUS | Designed by Mohammed Tali

The appointment of student as a faculty

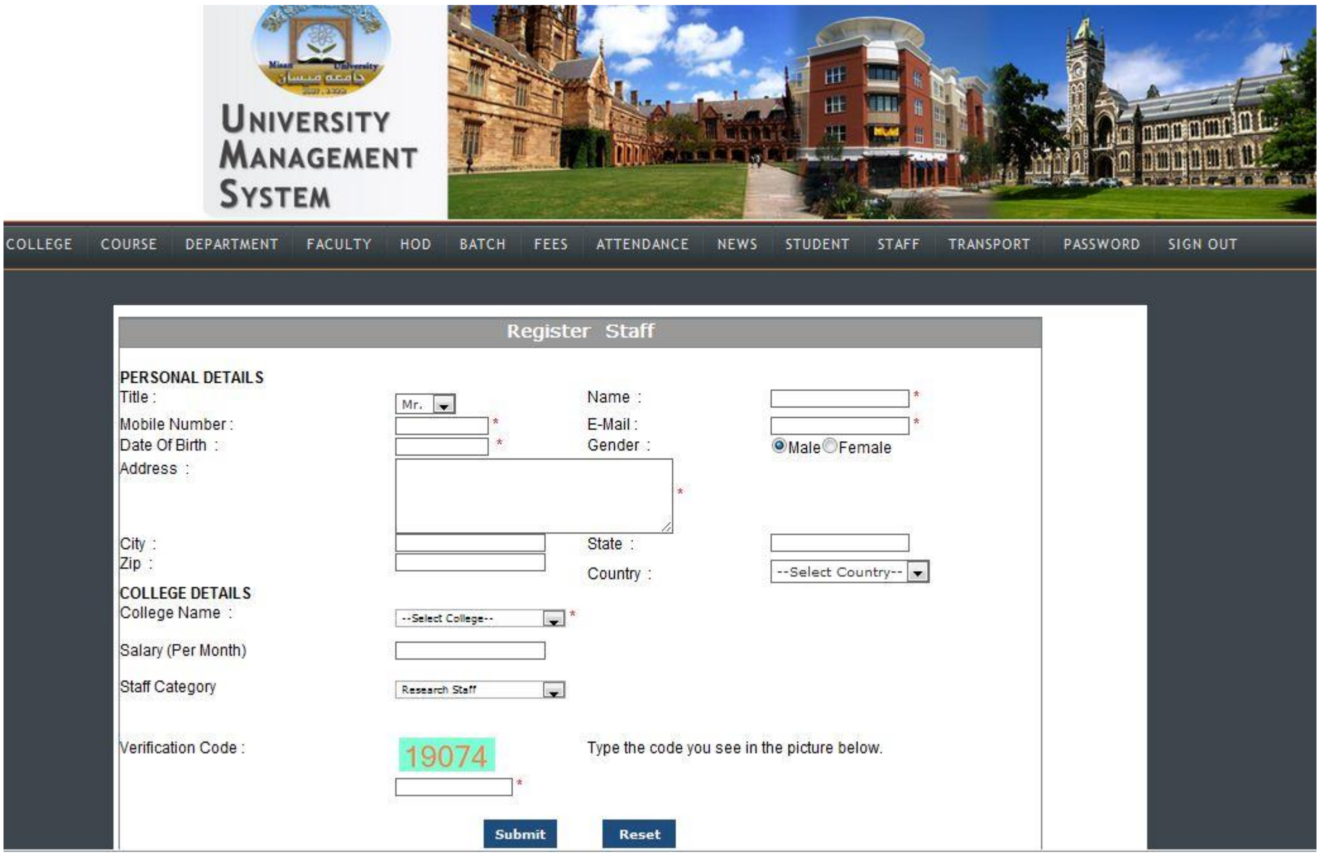

UNIVERSITY MANAGEMENT SYSTEM
COLLEGE COURSE DEPARTMENT FACULTY HOD BATCH FEES ATTENDANCE NEWS STUDENT STAFF TRANSPORT PASSWORD SIGN OUT
Appoint

Appoint Student As Faculty

College Name : --Select College--
Registration Code :
Salary :

Copyright © 2012 MUS | Designed by Mohammed Tali

New Staff Creation


UNIVERSITY MANAGEMENT SYSTEM
COLLEGE COURSE DEPARTMENT FACULTY HOD BATCH FEES ATTENDANCE NEWS STUDENT STAFF TRANSPORT PASSWORD SIGN OUT
Submit
Reset

Register Staff

PERSONAL DETAILS

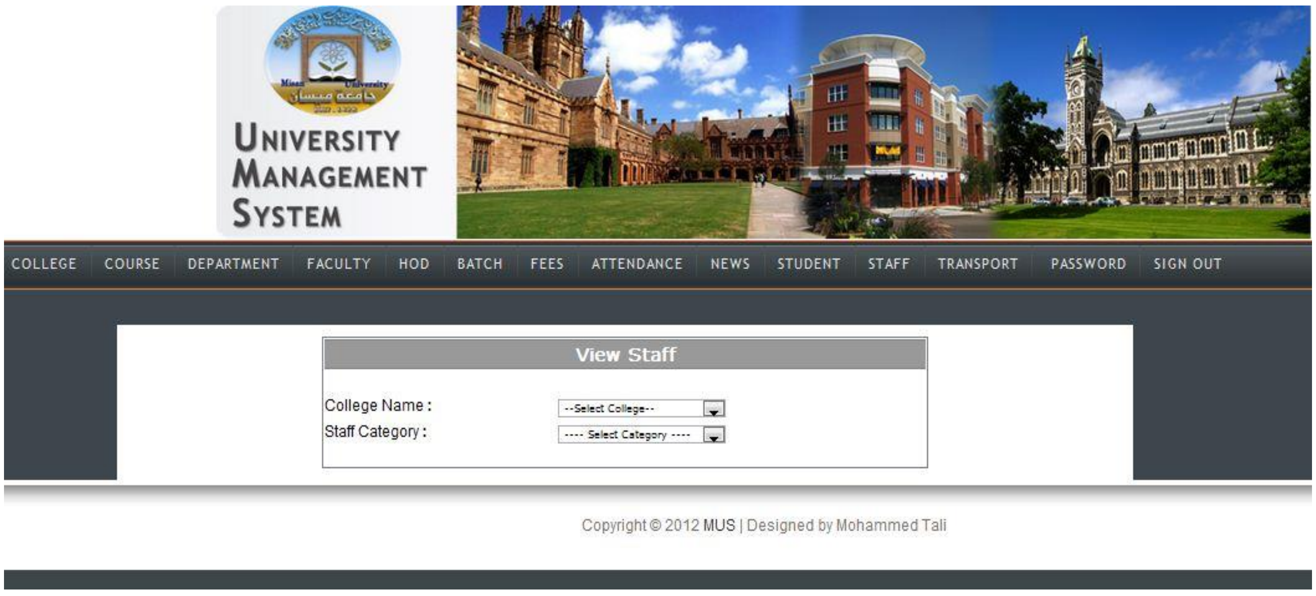
Title :	Mr.	Name :	*
Mobile Number :	*	E-Mail :	*
Date Of Birth :	*	Gender :	[✓ Male] Female
Address :	*		
City :		State :	
Zip :		Country :	--Select Country--

COLLEGE DETAILS

College Name :	--Select College-- *
Salary (Per Month)	
Staff Category	Research Staff
Verification Code :	19074 *

Type the code you see in the picture below.

View Staff

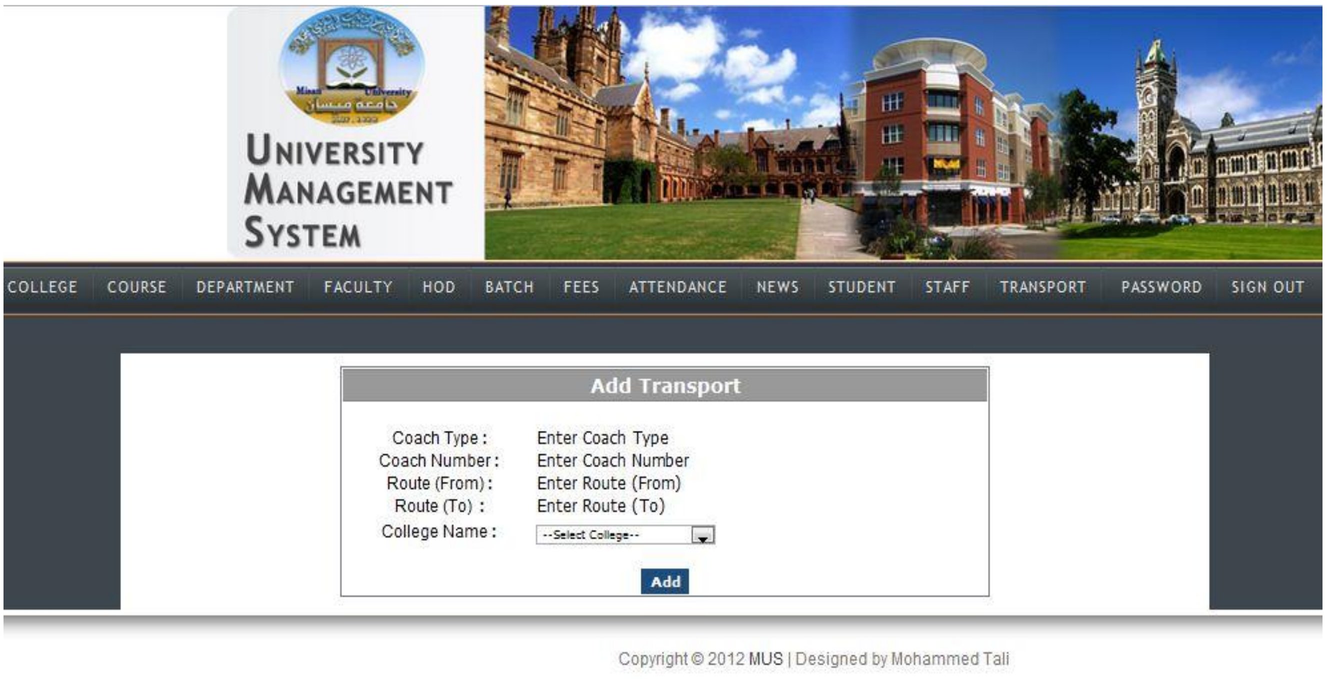

UNIVERSITY MANAGEMENT SYSTEM
COLLEGE COURSE DEPARTMENT FACULTY HOD BATCH FEES ATTENDANCE NEWS STUDENT STAFF TRANSPORT PASSWORD SIGN OUT

View Staff

College Name :	--Select College--
Staff Category :	---- Select Category ----

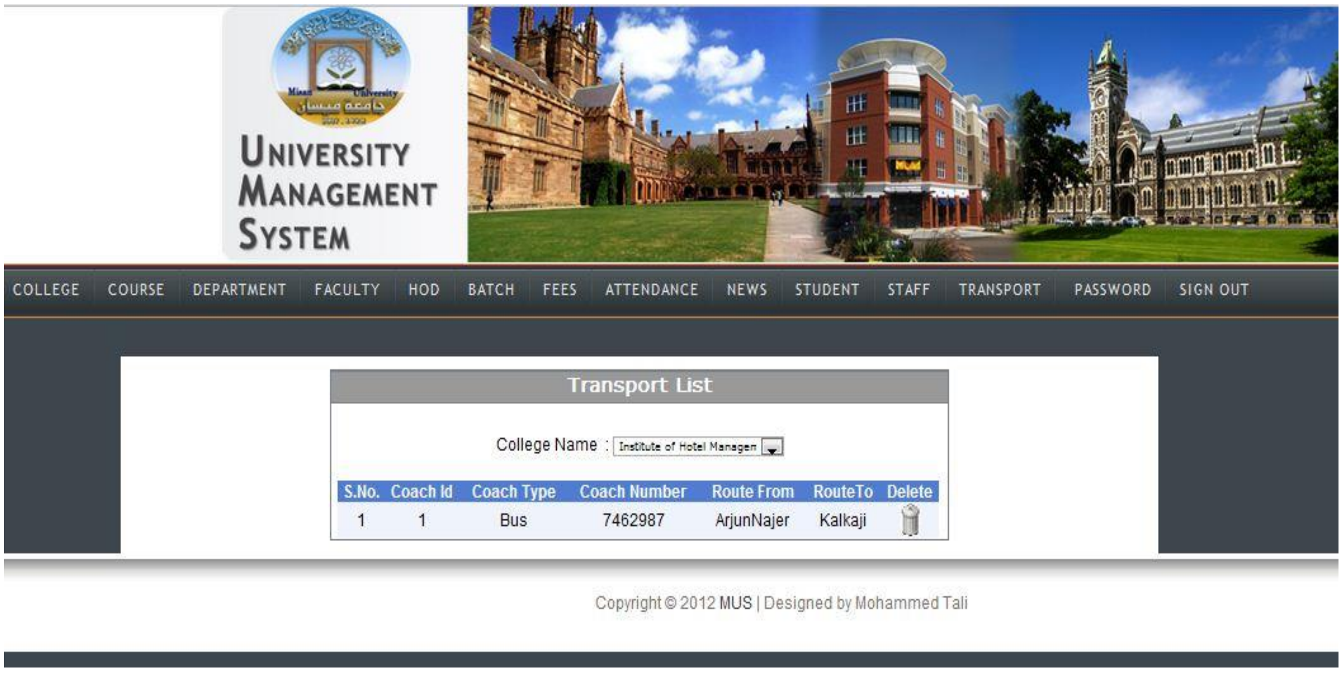
Copyright © 2012 MUS | Designed by Mohammed Tali

Add details of transport of the students which will be included in the card transport of the students



Coach Type :	Enter Coach Type
Coach Number :	Enter Coach Number
Route (From) :	Enter Route (From)
Route (To) :	Enter Route (To)
College Name :	--Select College--

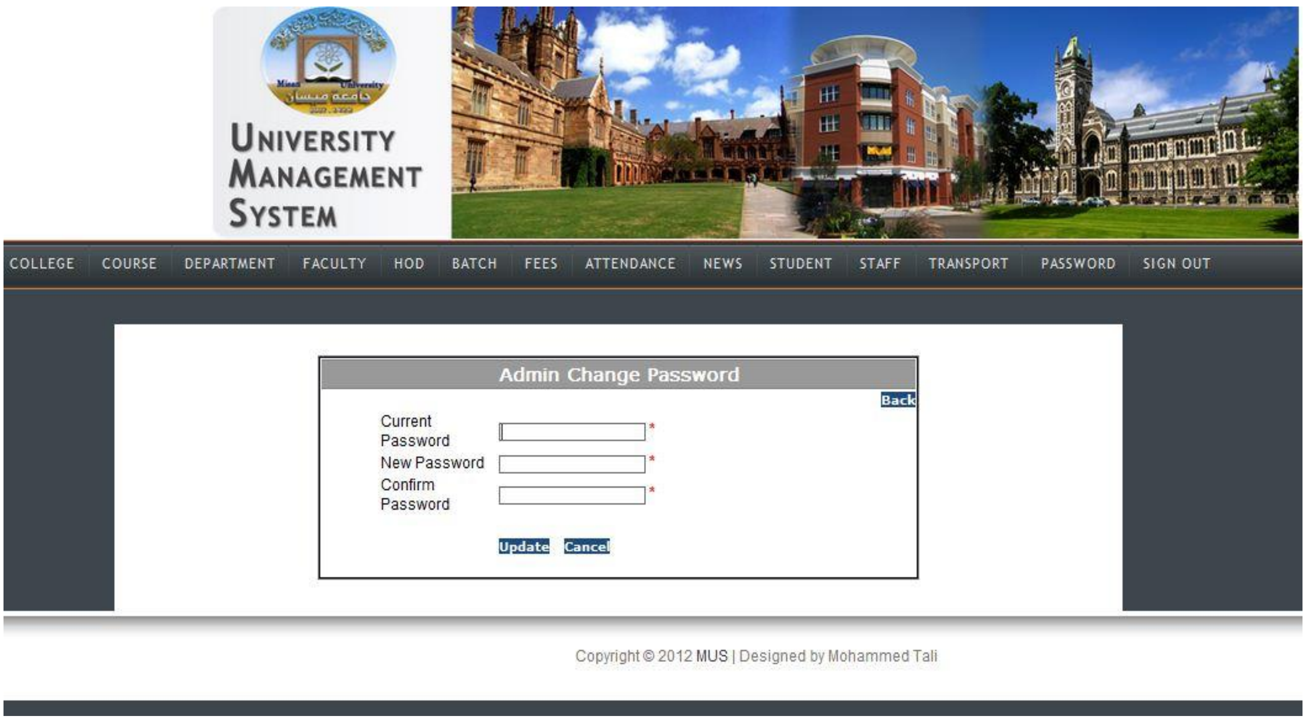
Transport List



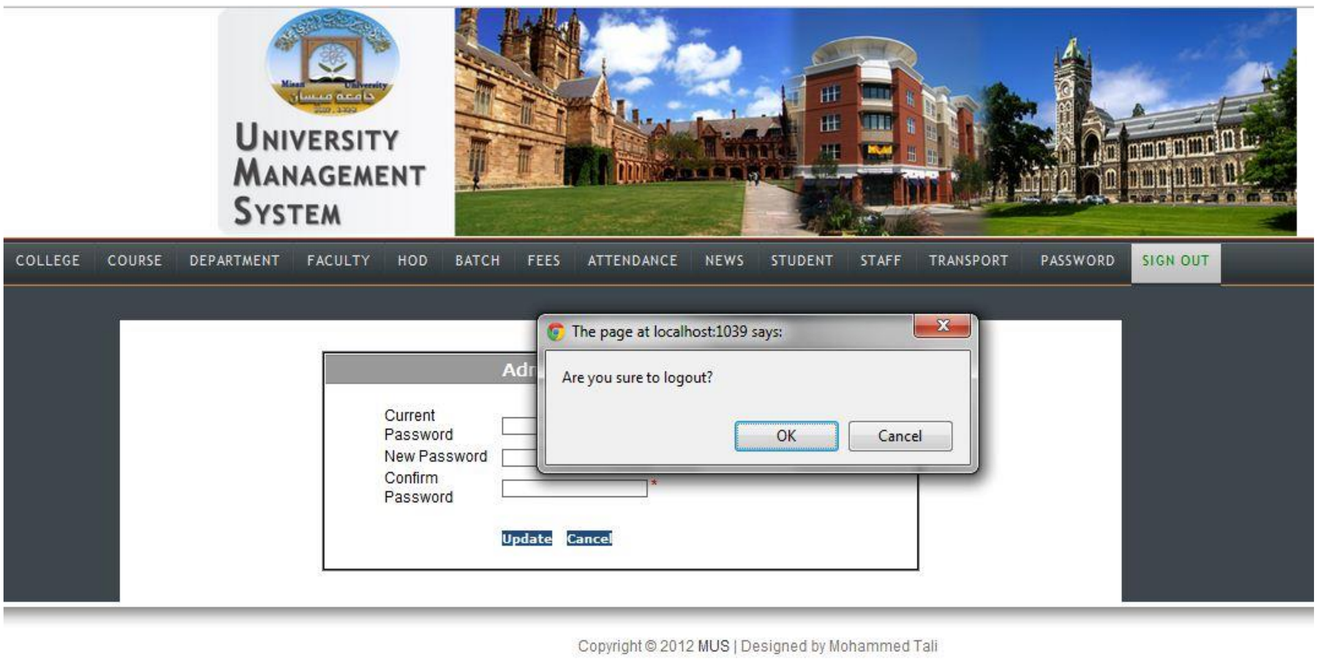
College Name : Institute of Hotel Managem

S.No.	Coach Id	Coach Type	Coach Number	Route From	RouteTo	Delete
1	1	Bus	7462987	ArjunNajer	Kalkaji	🗑

The User can change his Password



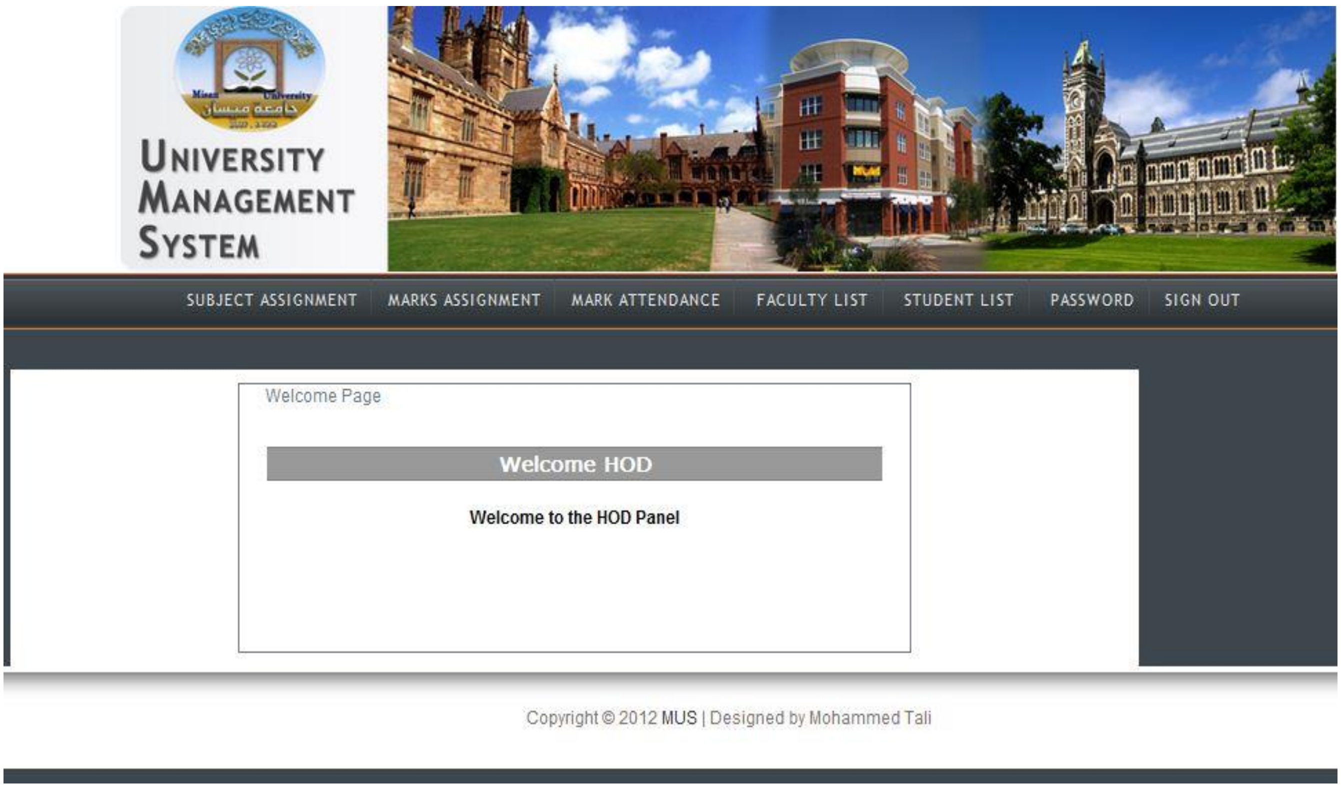
In case the user is made sign out will show a message to confirm the Logout



The page at localhost:1039 says:

Are you sure to logout?

HOD Page

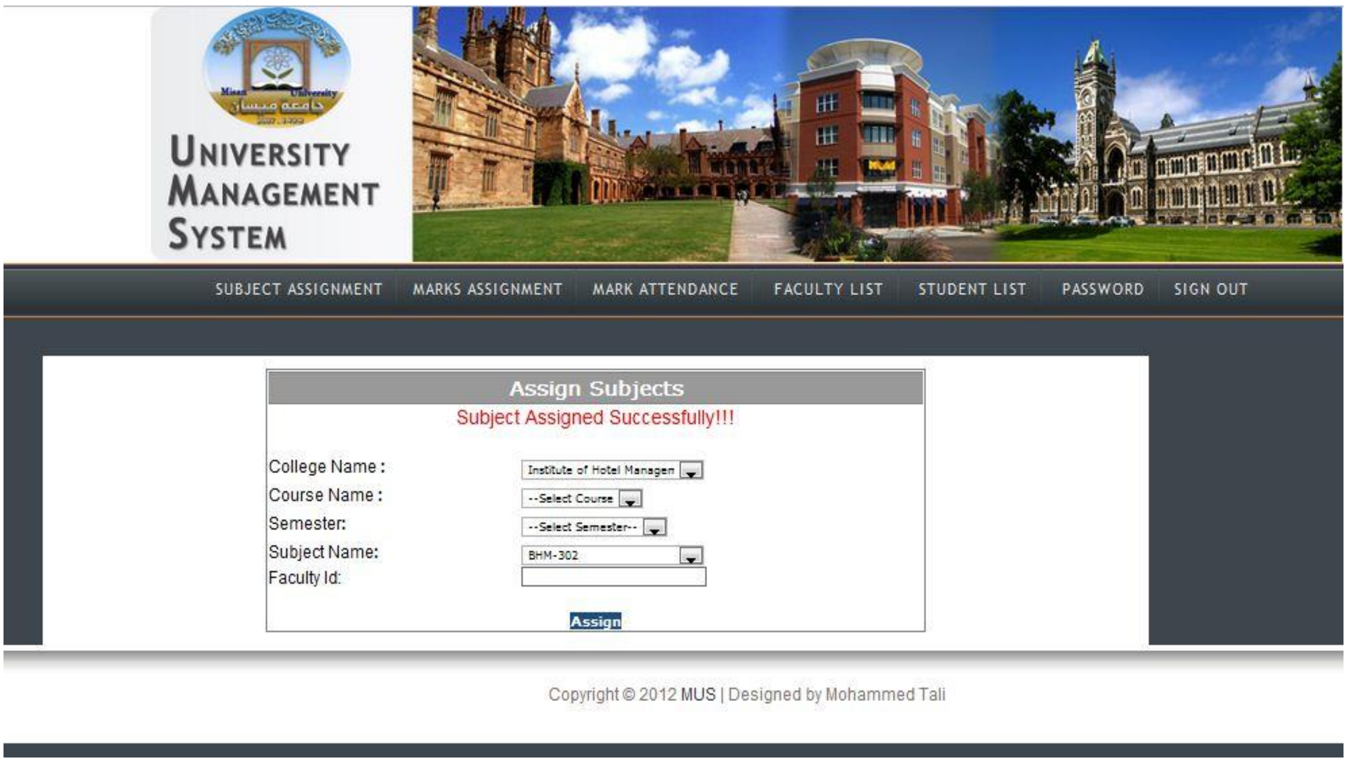


Welcome HOD

Welcome to the HOD Panel

Copyright © 2012 MUS | Designed by Mohammed Tali

The HOD is Assigns Subjects to the Faculty



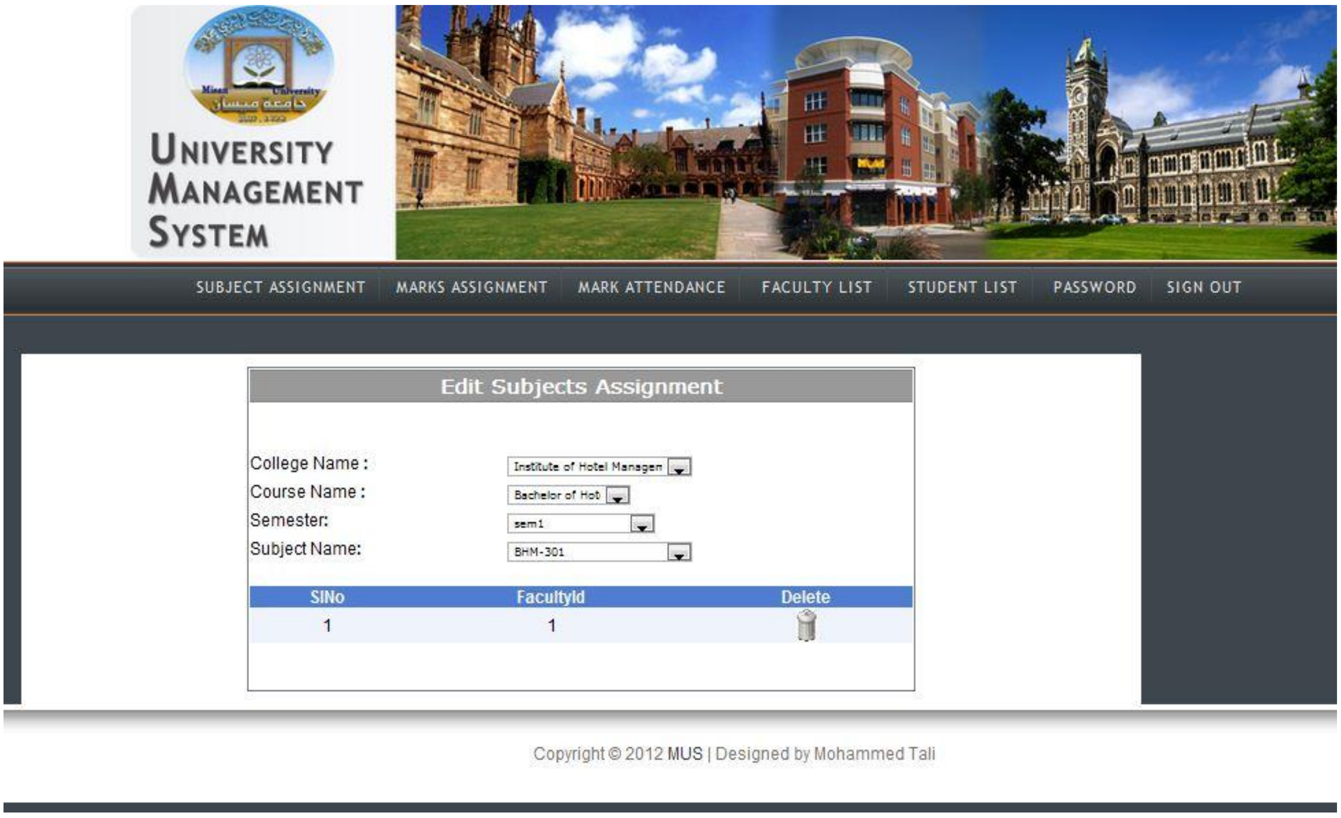
Assign Subjects

Subject Assigned Successfully!!!

College Name :	Institute of Hotel Managem
Course Name :	--Select Course
Semester:	--Select Semester--
Subject Name:	BHM-302
Faculty Id:	

Copyright © 2012 MUS | Designed by Mohammed Tali

Edit Subjects Assignment



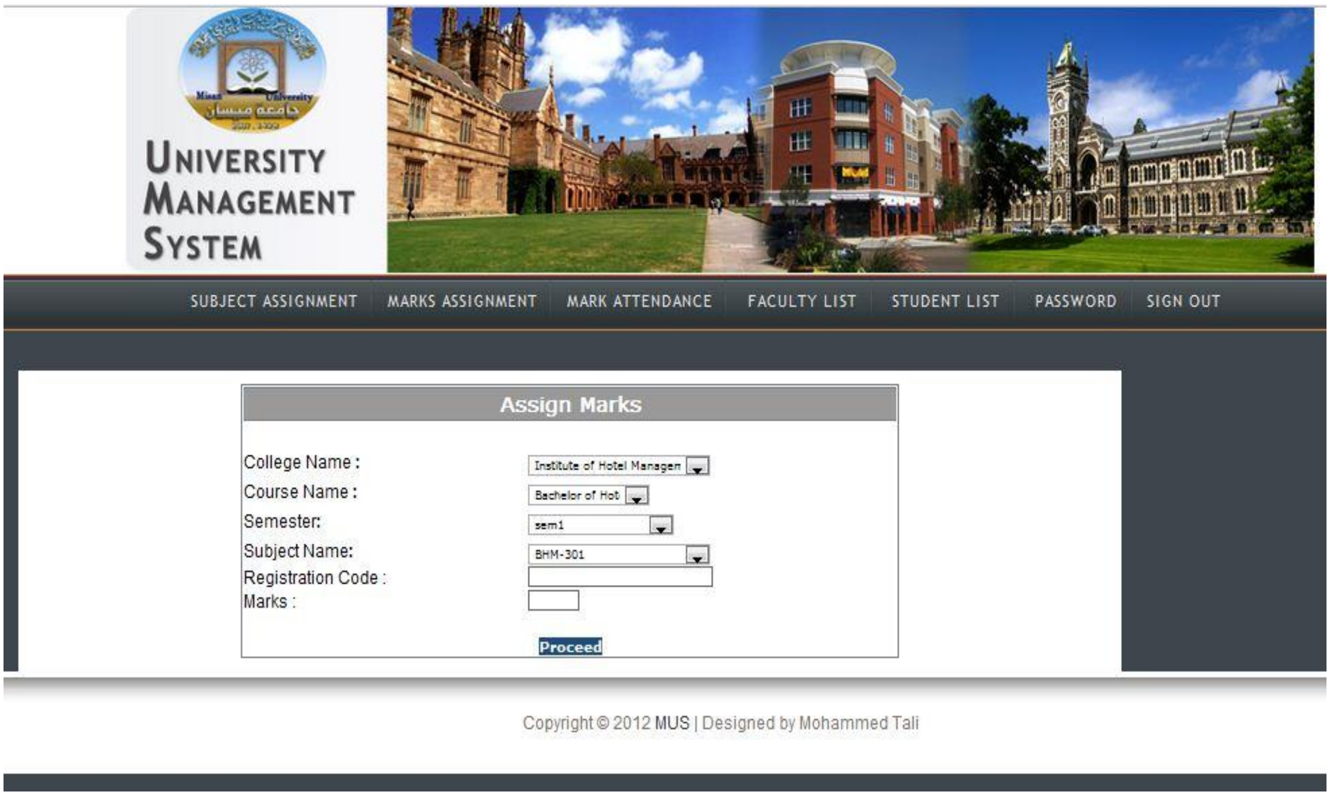
Edit Subjects Assignment

College Name :	Institute of Hotel Managem
Course Name :	Bachelor of Hot
Semester:	sem1
Subject Name:	BHM-301

SlNo	FacultyId	Delete
1	1	🗑

Copyright © 2012 MUS | Designed by Mohammed Tali

The HOD can assigns marks to the students

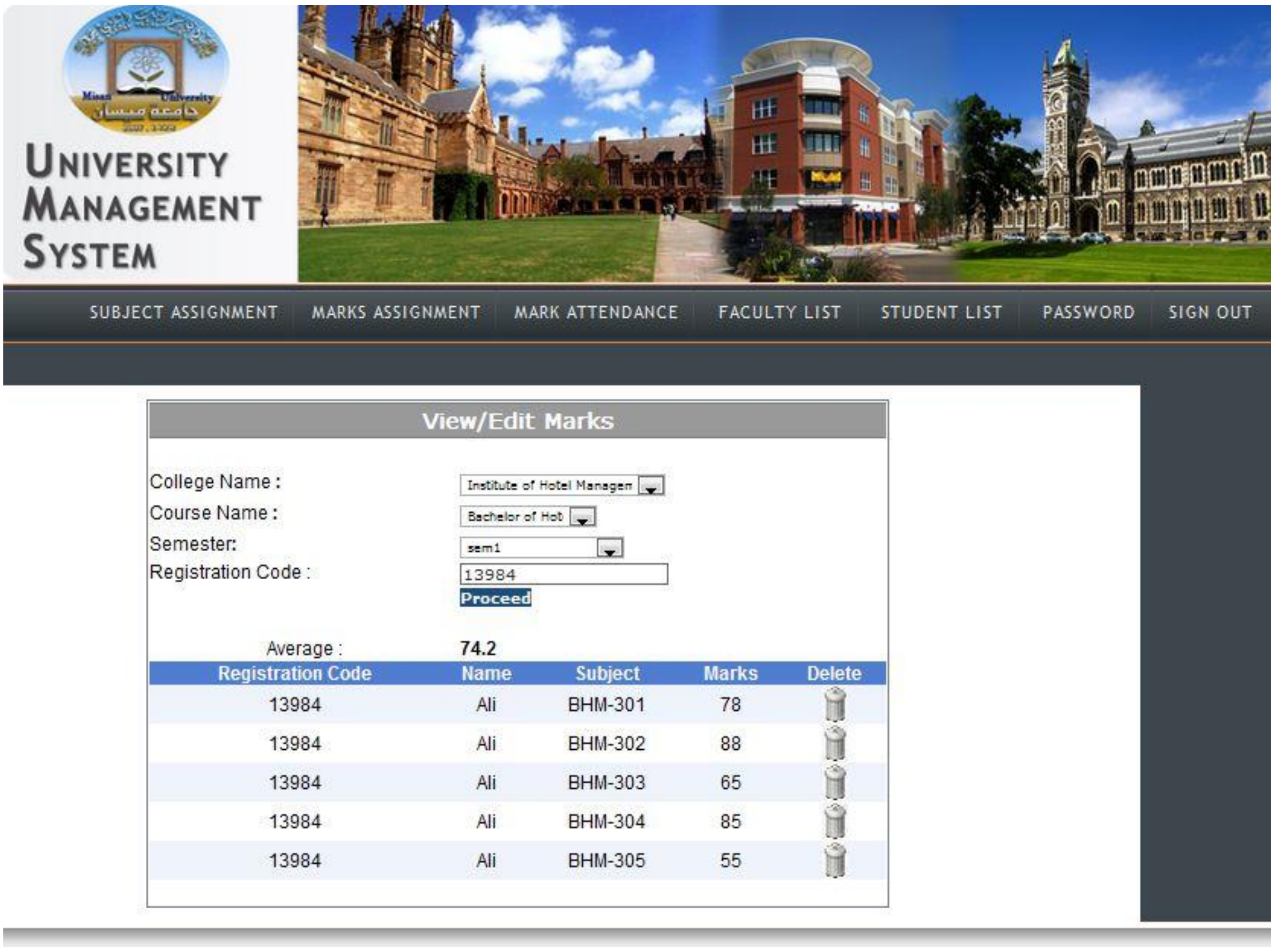


Assign Marks

College Name :	Institute of Hotel Managem
Course Name :	Bachelor of Hot
Semester:	sem1
Subject Name:	BHM-301
Registration Code :	
Marks :	

Copyright © 2012 MUS | Designed by Mohammed Tali

View and edit the marks of the student



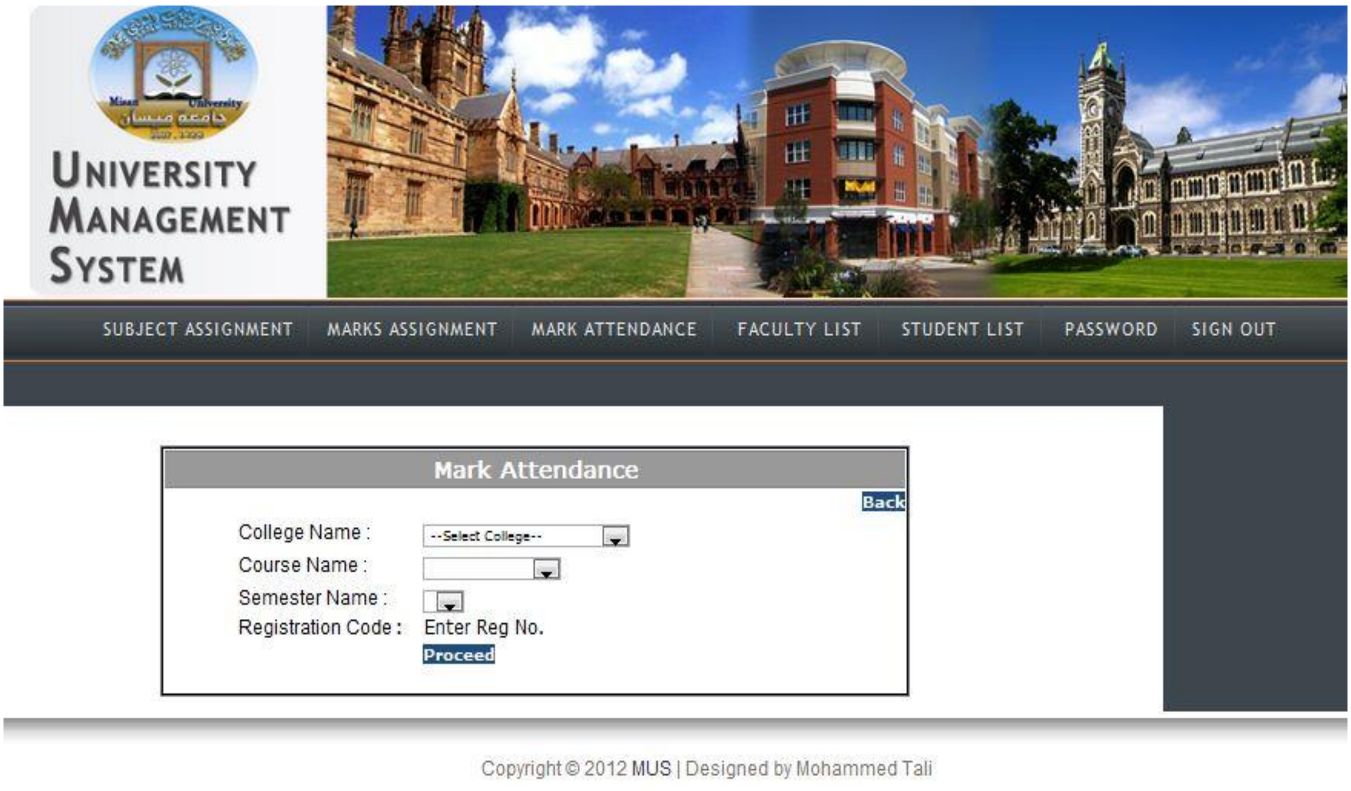
View/Edit Marks

College Name :	Institute of Hotel Managen
Course Name :	Bachelor of Hot
Semester:	sem1
Registration Code :	13984

Average : 74.2

Registration Code	Name	Subject	Marks	Delete
13984	Ali	BHM-301	78	
13984	Ali	BHM-302	88	
13984	Ali	BHM-303	65	
13984	Ali	BHM-304	85	
13984	Ali	BHM-305	55	

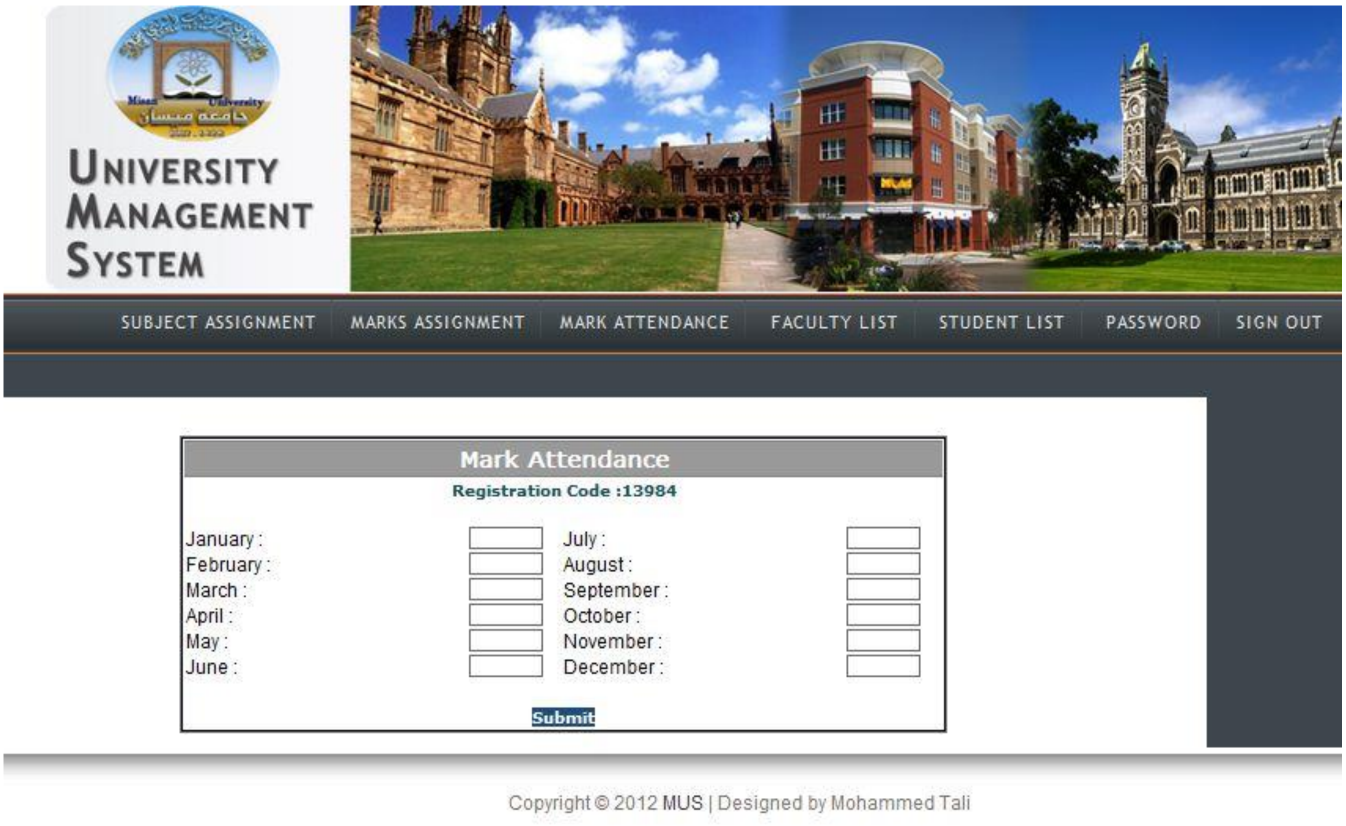
The HOD is marks attendance to the students



Mark Attendance

College Name : --Select College--
Course Name :
Semester Name :
Registration Code : Enter Reg No.

Copyright © 2012 MUS | Designed by Mohammed Tali



Mark Attendance

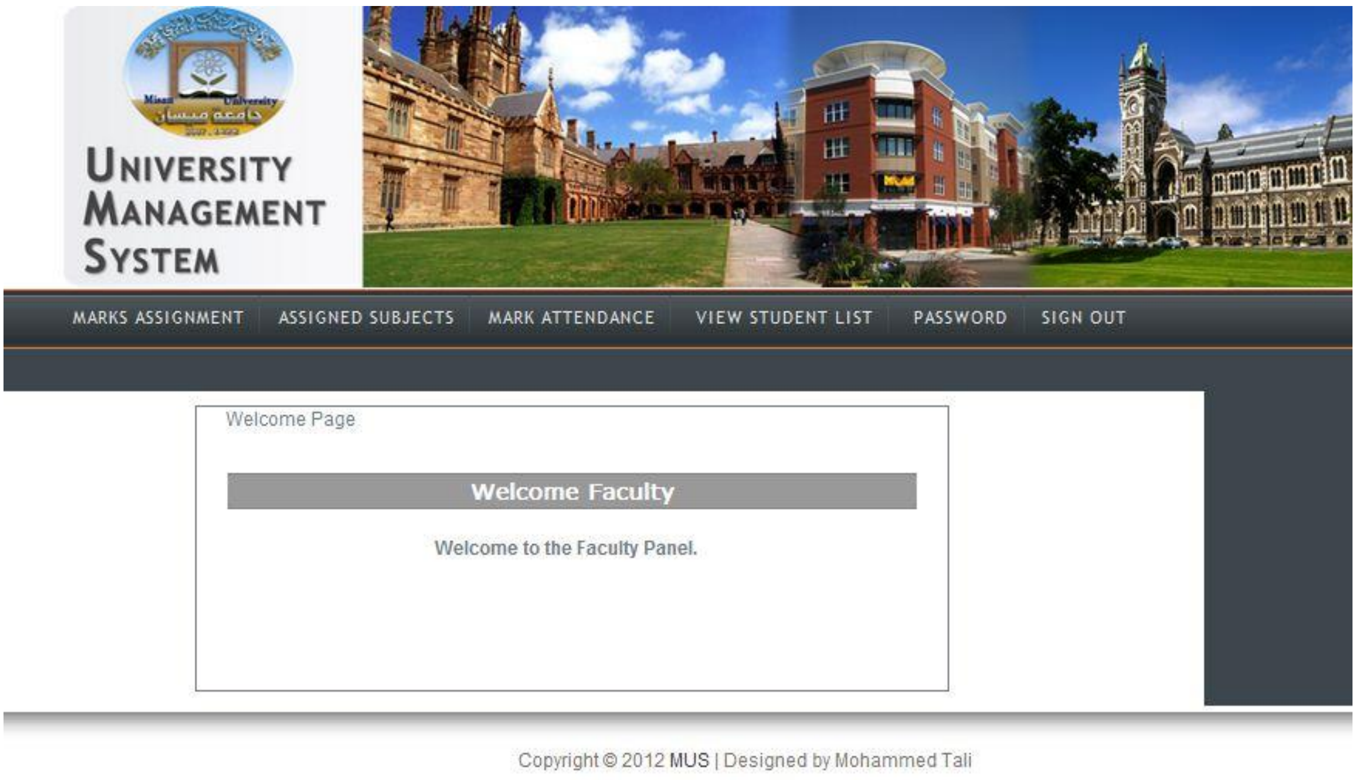
Registration Code :13984

January :		July :	
February :		August :	
March :		September :	
April :		October :	
May :		November :	
June :		December :	

Copyright © 2012 MUS | Designed by Mohammed Tali

Faculty Page

The faculty can assigns the marks of the students, View the assigned subjects, Mark attendance and view the list of the student.



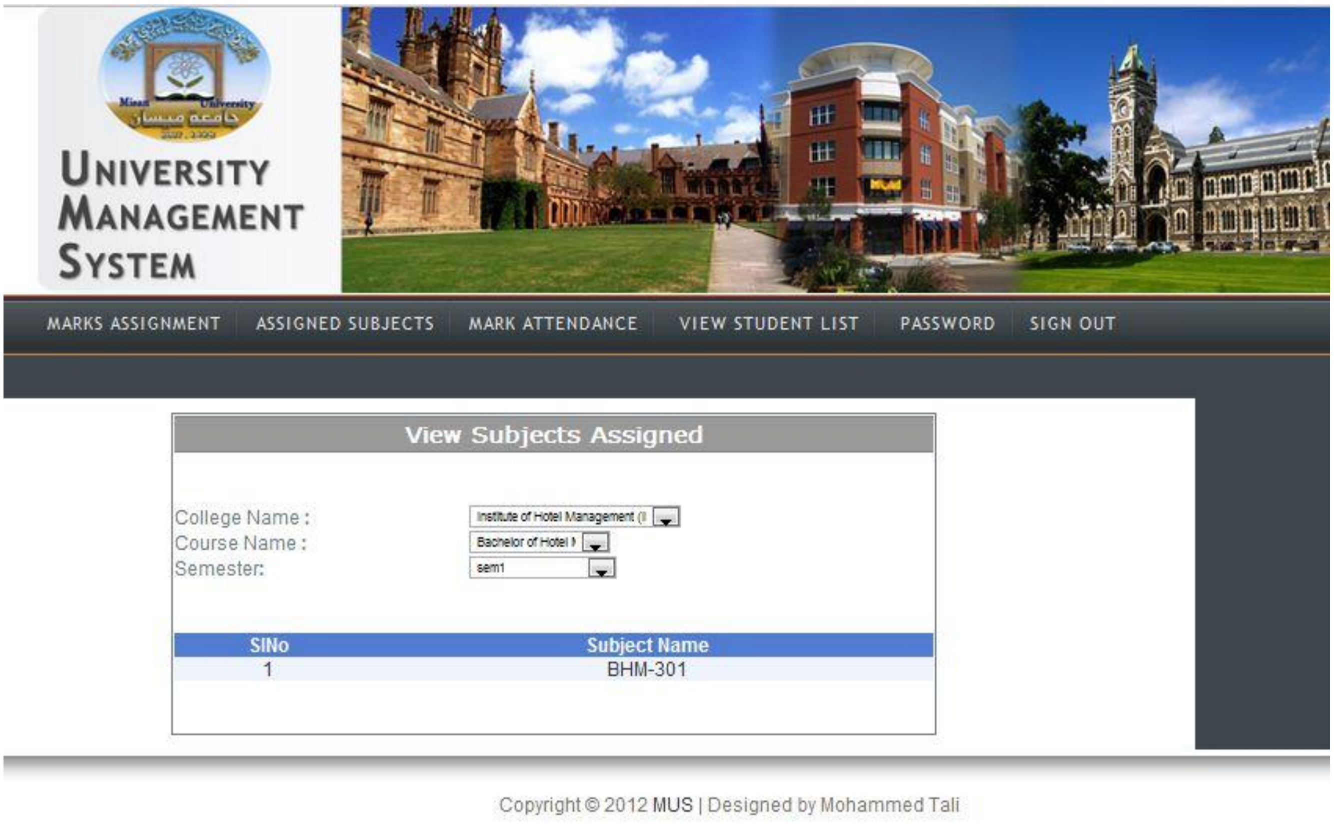
Welcome Page

Welcome Faculty

Welcome to the Faculty Panel.

Copyright © 2012 MUS | Designed by Mohammed Tali

View the assigned subjects



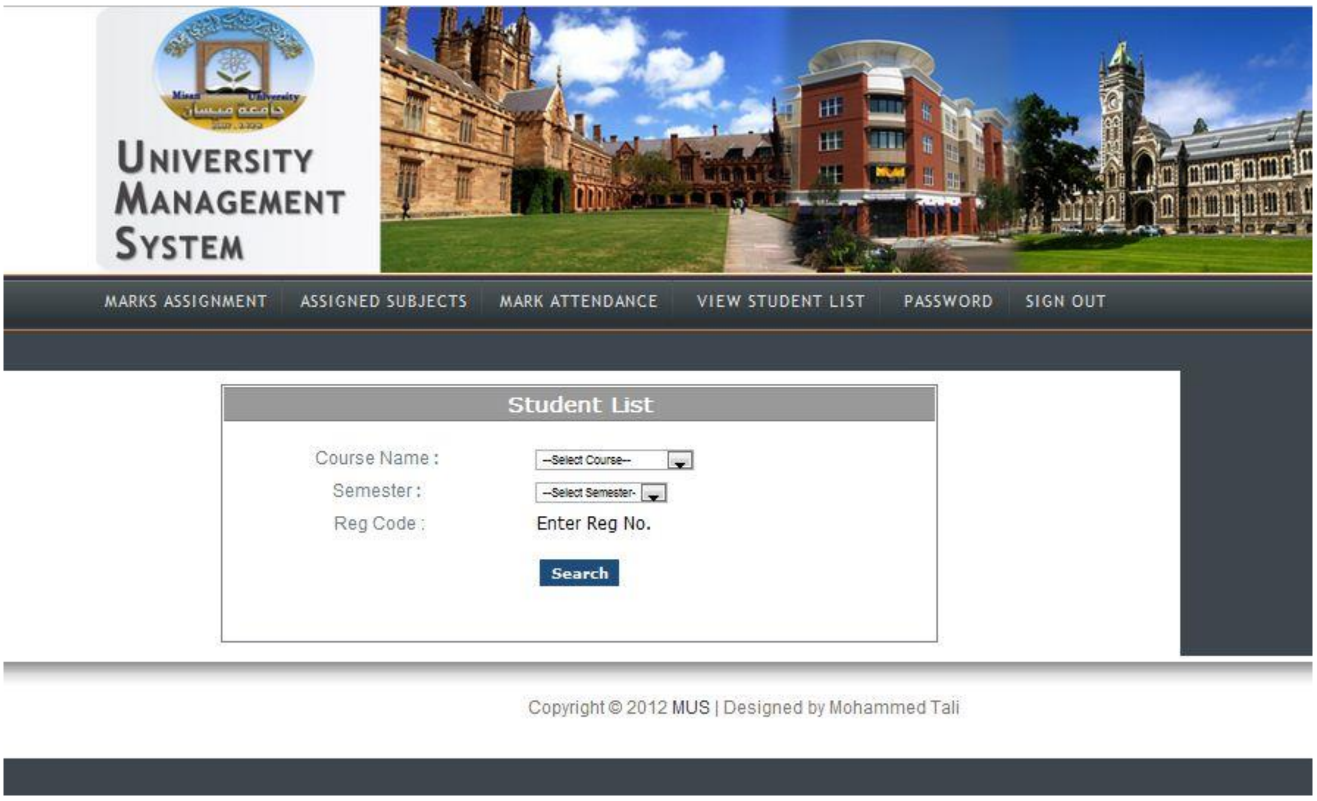
View Subjects Assigned

College Name :	Institute of Hotel Management (I
Course Name :	Bachelor of Hotel M
Semester:	sem1

SINo	Subject Name
1	BHM-301

Copyright © 2012 MUS | Designed by Mohammed Tali

View the list of the students

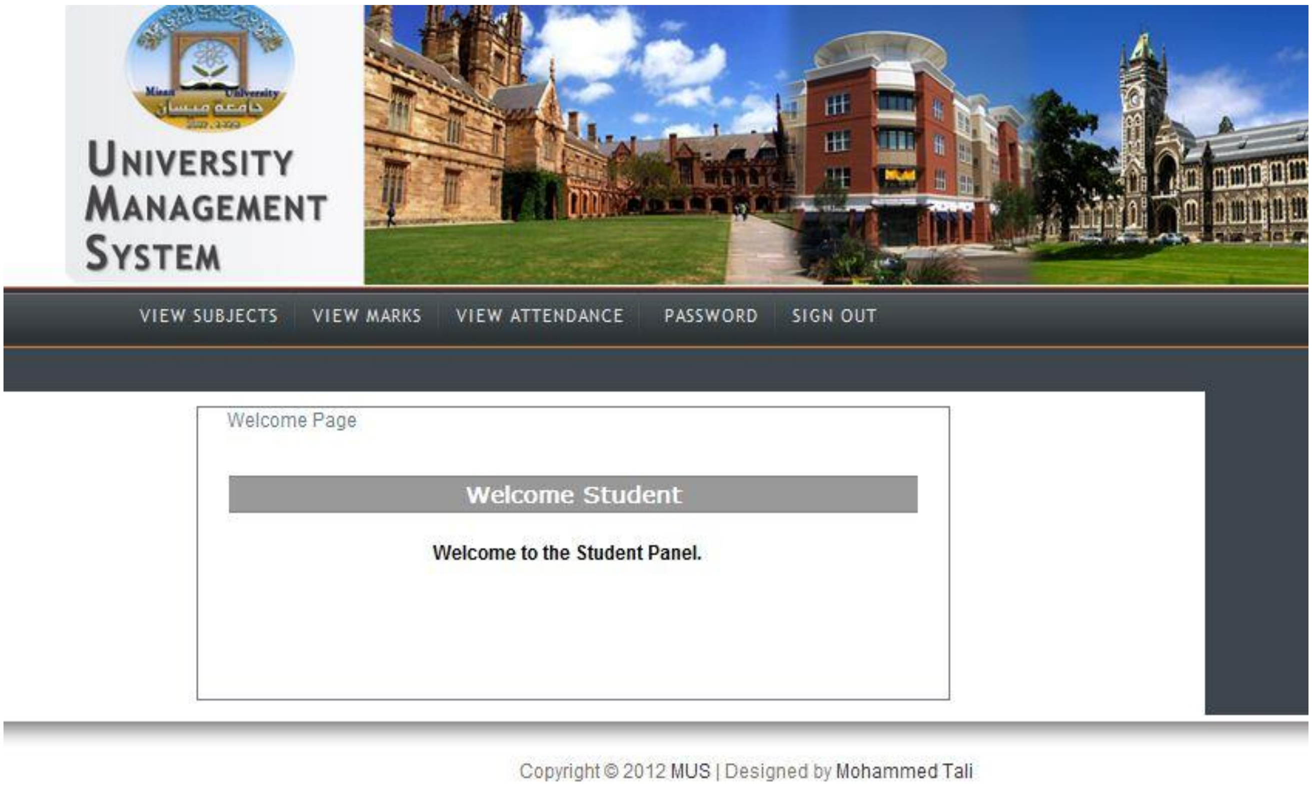


Student List

Course Name :	--Select Course--
Semester :	--Select Semester-
Reg Code :	Enter Reg No.

Copyright © 2012 MUS | Designed by Mohammed Tali

Faculty Page



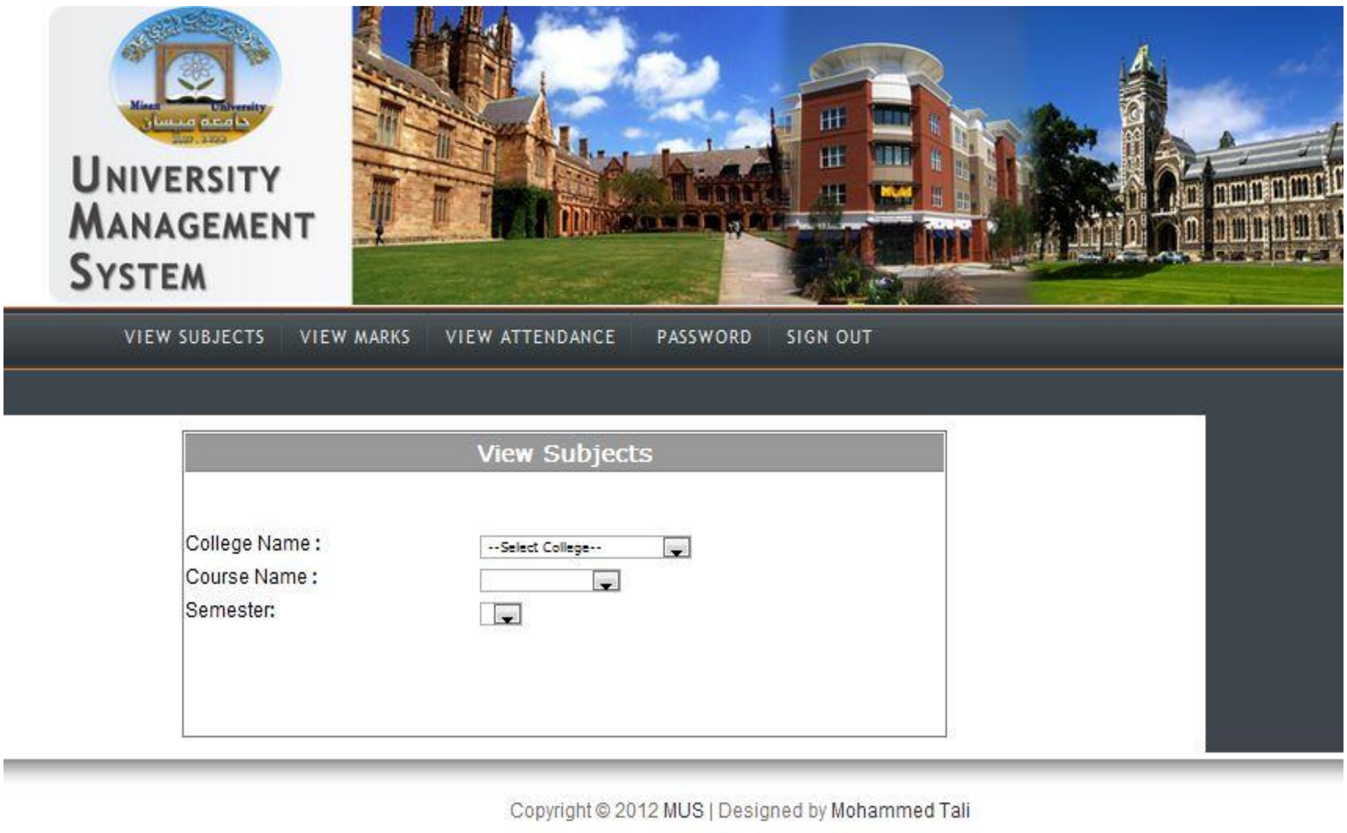
Welcome Page

Welcome Student

Welcome to the Student Panel.

Copyright © 2012 MUS | Designed by Mohammed Tali

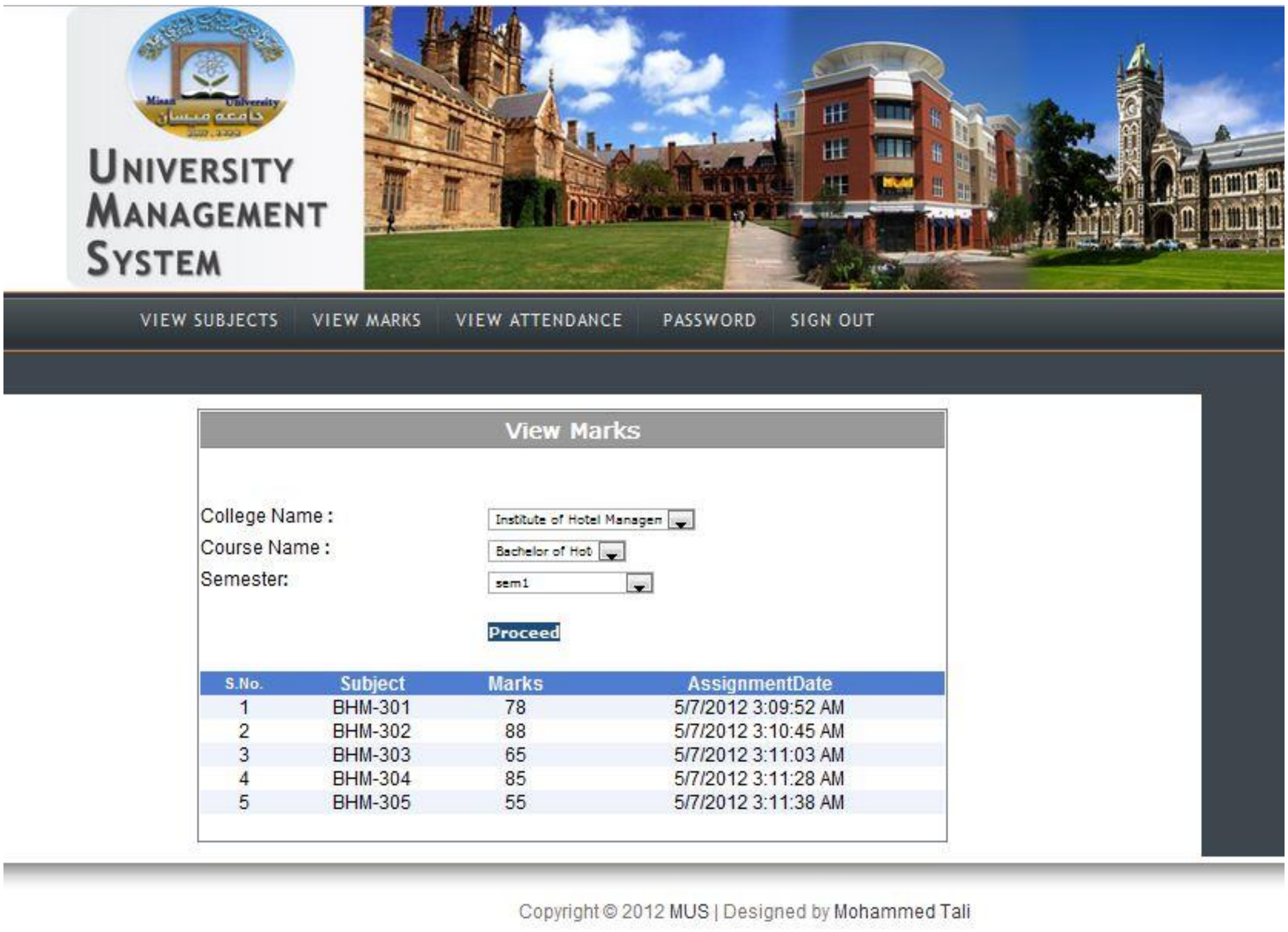
The Student views his/her subjects



College Name : --Select College--
Course Name :
Semester:

Copyright © 2012 MUS | Designed by Mohammed Tali

The Student views his/her marks



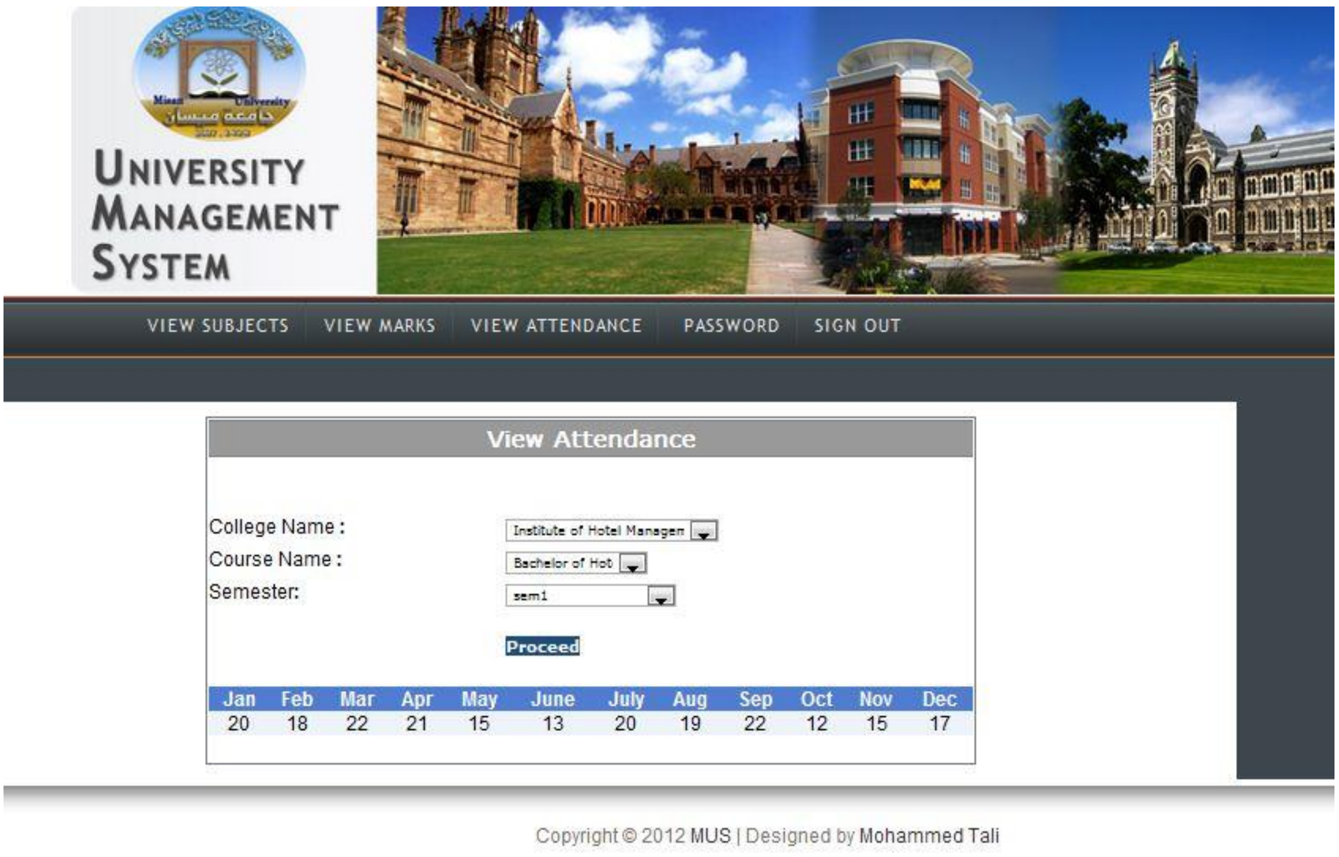
View Marks

College Name : Institute of Hotel Managem
Course Name : Bachelor of Hot
Semester: sem1

S.No.	Subject	Marks	AssignmentDate
1	BHM-301	78	5/7/2012 3:09:52 AM
2	BHM-302	88	5/7/2012 3:10:45 AM
3	BHM-303	65	5/7/2012 3:11:03 AM
4	BHM-304	85	5/7/2012 3:11:28 AM
5	BHM-305	55	5/7/2012 3:11:38 AM

Copyright © 2012 MUS | Designed by Mohammed Tali

The Student views his/her attendance



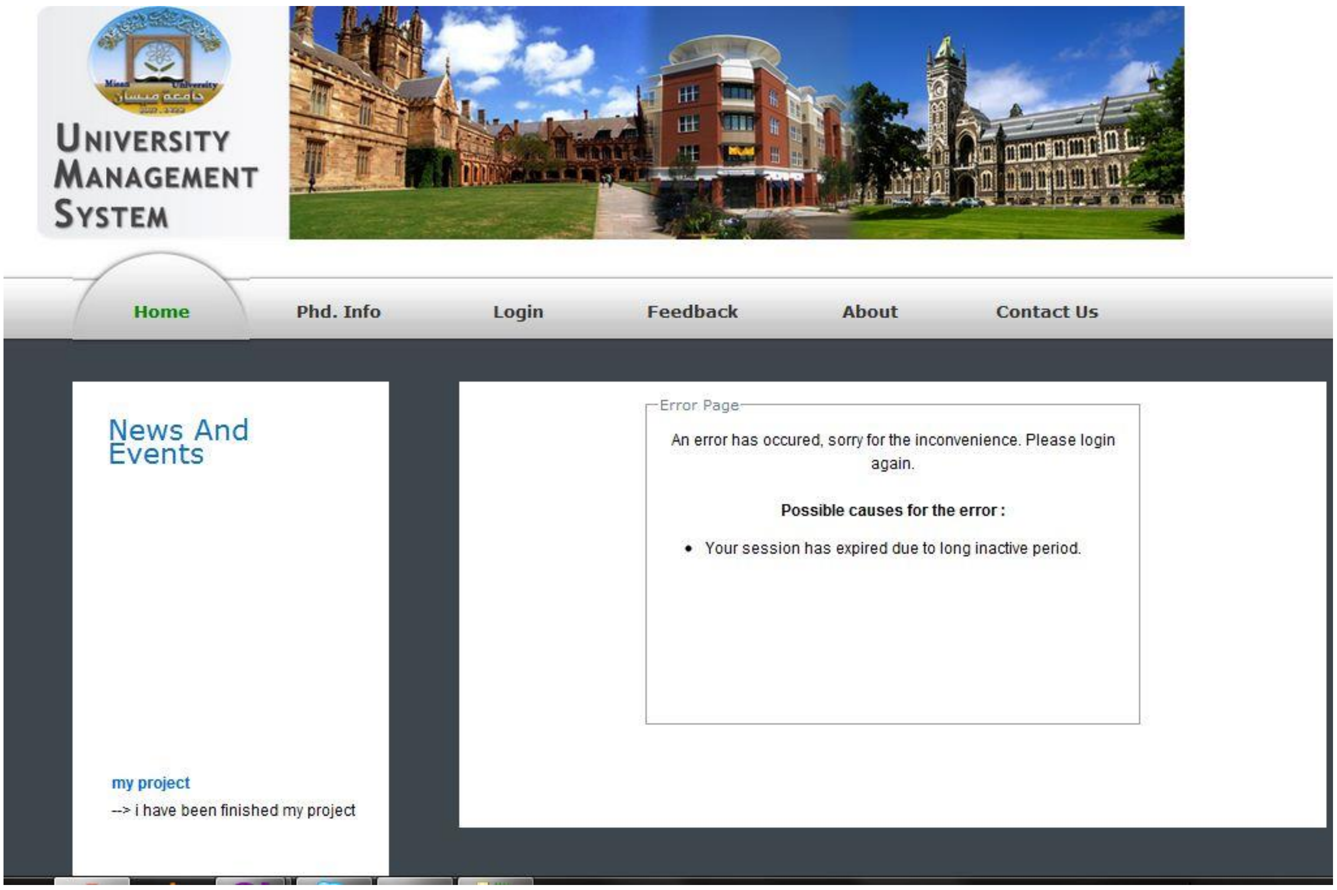
View Attendance

College Name : Institute of Hotel Managem
Course Name : Bachelor of Hot
Semester: sem1

Jan	Feb	Mar	Apr	May	June	July	Aug	Sep	Oct	Nov	Dec
20	18	22	21	15	13	20	19	22	12	15	17

Copyright © 2012 MUS | Designed by Mohammed Tali

In case the user has expired due to long inactive period



Error Page

An error has occured, sorry for the inconvenience. Please login again.

Possible causes for the error :

- Your session has expired due to long inactive period.

my project

--> i have been finished my project

11

System Security

11.1) System Security

Disaster is known as System Security.

System Security can be divided into four related issues:

- Security
- Integrity
- Privacy
- Confidentiality

SYSTEM SECURITY refers to the technical innovations and procedures applied to the hardware and operation systems to protect against deliberate or accidental damage from a defined threat.

DATA SECURITY is the protection of data from loss, disclosure, modification and destruction.

SYSTEM INTEGRITY refers to the power functioning of hardware and programs, appropriate physical security and safety against external threats such as eavesdropping and wiretapping.

PRIVACY defines the rights of the user or organizations to determine what information they are willing to share with or accept from others and how the organization can be protected against unwelcome, unfair or excessive dissemination of information about it.

CONFIDENTIALITY is a special status given to sensitive information in a database to minimize the possible invasion of privacy. It is an attribute of information that characterizes its need for protection.

11.2) SECURITY IN SOFTWARE

System security refers to various validations on data in form of checks and controls to avoid the system from failing. It is always important to ensure that only valid data is entered and only valid operations are performed on the system. The system employees two types of checks and controls:

CLIENT SIDE VALIDATION

Various client side validations are used to ensure on the client side that only valid data is entered. Client side validation saves server time and load to handle invalid data. Some checks imposed are:

- VBScript in used to ensure those required fields are filled with suitable data only. Maximum lengths of the fields of the forms are appropriately defined.
- Forms cannot be submitted without filling up the mandatory data so that manual mistakes of submitting empty fields that are mandatory can be sorted out at the client side to save the server time and load.
- Tab-indexes are set according to the need and taking into account the ease of user while working with the system.

SERVER SIDE VALIDATION

Some checks cannot be applied at client side. Server side checks are necessary to save the system from failing and intimating the user that some invalid operation has been performed or the performed operation is restricted. Some of the server side checks imposed is:

- Server side constraint has been imposed to check for the validity of primary key and foreign key. A primary key value cannot be duplicated. Any attempt to duplicate the primary value results into a message intimating the user about those values through the forms using foreign key can be updated only of the existing foreign key values.
- User is intimating through appropriate messages about the successful operations or exceptions occurring at server side.
- Various Access Control Mechanisms have been built so that one user may not agitate upon another. Access permissions to various types of users are controlled according to the organizational structure.
- Only permitted users can log on to the system and can have access according to their category. User- name, passwords and permissions are controlled o the server side.
- Using server side validation, constraints on several restricted operations are imposed.

12

Maintenance

12.1) **MAINTENANCE**

Maintenance the last phase in the software engineering process. As programs are developed.

A distributing trend has emerged the amount of effort and a resource expended on software maintenance is growing. In total project development maintenances takes 65% of effort .In software maintenance there are four .They are :

- Adaptive Maintenance.
- Corrective Maintenance.
- Perfective Maintenance.
- Preventive Maintenance.

Adaptive Maintenance is applied when changes in the external environment precipitate modifications to software. It deals with adapting the software to new environments.

Perfective Maintenance incorporates enhancements that are requested by user community. It deals with updating the software according to changes in user requirements

Corrective Maintenance acts to correct errors that are uncovered after the software is in use. It deals with fixing bugs in the code

Preventive Maintenance improves future maintainability and reliability and provides a basis for future enhancement. It deals with updating documentation and making the software more maintainable .Tasks performed during the software engineering process define maintainability and have an important impact in the success of any maintenance approach. Reverse Engineering and Reengineering are the tools and techniques used to maintain the project.

There are four major problems that can slow down the maintenance process

- Unstructured code
- Maintenance programmers having insufficient knowledge of the System
- Documentation being absent
- Out of Date , or at best insufficient

The success of the maintence phase relies on these problems being earlier in the life cycle.

13

Limitations and Future Improvement

13.1) **LIMITATIONS**

- The size of the database increases day-by-day, increasing the load on the database back up and data maintenance activity.
- Training for simple computer operations is necessary for the users working on the system.

13.2) **FUTURE IMPROVEMENT**

- It can be implemented to upload files with an huge amount of size with the support of various file formats.

- This System being web-based and an undertaking of Cyber Security Division, needs to be thoroughly tested to find out any security gaps.

- A console for the data centre may be made available to allow the personnel to monitor on the sites which were cleared for hosting during a particular period.

- Moreover, it is just a beginning; further the system may be utilized in various other types of auditing operation viz. Network auditing or similar process/workflow based applications...

14

Conclusion and Bibliography

14.1) CONCLUSION

It has been a great pleasure for me to work on this exciting and challenging project. This project proved good for me as it provided practical knowledge of not only programming in ASP.NET and C#.Net web based application and no some extent Windows Application and SQL Server, but also about all handling procedure related with “University Management system”. It also provides knowledge about the latest technology used in developing web enabled application and client server technology that will be great demand in future. This will provide better opportunities and guidance in future in developing projects independently.

BENEFITS:

The project is identified by the merits of the system offered to the user. The merits of this project are as follows: -

- It’s a web-enabled project.
- This project offers user to enter the data through simple and interactive forms. This is very helpful for the client to enter the desired information through so much simplicity.
- The user is mainly more concerned about the validity of the data, whatever he is entering. There are checks on every stages of any new creation, data entry or update so that the user cannot enter the invalid data, which can create problems at later date.
- Sometimes the user finds in the later stages of using project that he needs to update some of the information that he entered earlier. There are options for him by which he can update the records. Moreover there is restriction for his that he cannot change the primary data field. This keeps the validity of the data to longer extent.
- User is provided the option of monitoring the records he entered earlier. He can see the desired records with the variety of options provided by him.
- From every part of the project the user is provided with the links through framing so that he can go from one option of the project to other as per the requirement. This is bound to be simple and very friendly as per the user is concerned. That is, we can say that the project is user friendly which is one of the primary concerns of any good project.
- Data storage and retrieval will become faster and easier to maintain because data is stored in a systematic manner and in a single database.
- Decision making process would be greatly enhanced because of faster processing of information since data collection from information available on computer takes much less time then manual system.

- Allocating of sample results becomes much faster because at a time the user can see the records of last years.
- Easier and faster data transfer through latest technology associated with the computer and communication.
- Through these features it will increase the efficiency, accuracy and transparency

14.2) BIBLIOGRAPHY

- FOR .NET INSTALLATION

www.support.mircosoft.com

- FOR DEPLOYMENT AND PACKING ON SERVER

www.developer.com

www.15seconds.com

- FOR SQL

www.msdn.microsoft.com

- FOR ASP.NET

Asp.Net 3.5 Unleashed

www.msdn.microsoft.com/net/quickstart/aspplus/default.com

www.asp.net

www.fmexpense.com/quickstart/aspplus/default.com

www.asptoday.com

www.aspfree.com

www.4guysfromrolla.com/index.aspx

- The following books were referred during the analysis and execution phase of the project:

Sql the compleate reference

By Sql Press

Software engineering

By Roger.S.Pressman

Professional asp.net

By Wrox

msdn

By Microsoft